

Game-Theoretic Decision Making in Imperfect-Information Games

Learning Dynamics, Equilibrium Computation, and Complexity

Gabriele Farina

CMU-CS-23-117

May 2023

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee

Tuomas Sandholm, Chair

Vincent Conitzer

Geoffrey Gordon

J. Zico Kolter

Avrim Blum (Toyota Technological Institute at Chicago)

Constantinos Daskalakis (Massachusetts Institute of Technology)

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy*

Copyright © 2023 Gabriele Farina

Parts of this research was sponsored by the National Science Foundation under grant numbers IIS-1718457, IIS-1617590, IIS-1901403, and CCF-1733556; the ARO under awards W911NF-20-1-0081, W911NF-17-1-0082, and W911NF-22-1-0266; a 2019-2020 Facebook Research fellowship. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: computational game theory, extensive-form games, learning dynamics, equilibrium computation.

*To the many scientists and scholars whose work laid the foundations for this dissertation.
Your ingenuity, innovation, and tireless pursuit of knowledge have been a great personal inspiration.*

Abstract

Designing machines that can reason strategically and make optimal decisions even in the presence of imperfect information and adversarial behavior is a fundamental goal of artificial intelligence. Our understanding of game-theoretic decision-making is fairly advanced in *normal-form* games, strategic interactions in which all players act once and simultaneously, selecting (possibly randomizing their choice) an action from a pre-determined finite set of possibilities. In contrast, this dissertation focuses on the more realistic imperfect-information extensive-form games, tree-form decision-making problems in which decision makers can face multiple decisions interleaved with observations about the way previous decisions affected the (partially hidden and possibly adversarial) multiagent environment.

This dissertation seeks to close some recognized open problems in this area, and provide solid theoretical and algorithmic foundations for strategic, game-theoretic decision-making in imperfect-information extensive-form games. We highlight the following contributions.

- We close the long-standing open question of whether extensive-form correlated equilibria can arise from efficient, uncoupled learning dynamics, establishing unconditional positive results.
- We settle the complexity of computing extensive-form perfect equilibria in two-player games, a recognized open problem, showing it matches that of Nash equilibrium.
- We establish learning dynamics with state-of-the-art $\mathcal{O}_T(\log T/T)$ convergence to coarse correlated equilibrium, improving as a corollary known bounds for normal-form games.
- We uncover a kernelized version of the classic multiplicative weights update algorithm. This algorithm leads to learning dynamics for coarse correlated equilibria with theoretical state-of-the-art dependence on the size of the imperfect-information game.
- We give fundamental results about the geometry of correlated strategies, identifying new important classes of games where optimal (*e.g.*, welfare-maximizing) correlated solution concepts can be computed in polynomial time, yielding positive complexity results for extensive-form correlation.
- We give the first learning algorithm with guaranteed convergence to welfare-maximizing correlated solution concepts, and show their state-of-the-art empirical performance.
- We give state-of-the-art algorithms that enable, for the first time, finding exact trembling-hand refinements of Nash equilibrium in large games with up to half a billion terminal states.
- We study the problem of computing optimal team strategies in zero-sum adversarial team games, and give state-of-the-art algorithms and positive complexity results for that setting.
- We give learning dynamics that can incorporate a model of the player to produce strong, human-compatible strategies. These dynamics were a key component in training artificial intelligence for the game of Diplomacy, a 7-player game involving both cooperation and competition with humans.

Acknowledgments

Before delving into the contents of the dissertation, I would like to take a moment to express my heartfelt appreciation to everyone who has been a part of my academic journey and has contributed to the completion of this dissertation.

To my advisor, Tuomas Sandholm, thank you for the guidance, encouragement, and support throughout my graduate studies. Your mentorship has helped me grow both as a researcher and as a person. I am grateful for the countless intellectually stimulating discussions we had, and for the great deal of academic freedom you have always entrusted me with.

To my roommates over the years, Zach, Jay, and Camille, thank you for the laughter, support, and fun memories we have shared. From impromptu BBQs to late-night chats, you have complemented grad school in so many ways. You have often challenged my perspective and helped me adjust my focus on the things that matter. I was lucky to have you as a second family.

To Margarida, thank you for your unwavering love and support. Your patience, kindness, and help have been a source of inspiration and have meant the world to me. I am grateful for every moment we have shared together.

To my committee members, colleagues, mentors, and peers, thank you for your feedback, insights, and collaborations. Your contributions have broadened my perspective and enriched my research. I am grateful for the friendships and connections we have formed along the way. I am especially indebted to my mentors whose example and guidance lit the way: Christian, Nicola, Noam, Vince, Zico, and Costis. A special thanks also to the many awesome collaborators I was lucky to work with and learn from: Ioannis Anagnostides, Brian, Stephen, Haipeng, Ioannis Panageas, Keegan, Misha, Sam, Chun Kai, Fei, John, Chung-Wei, Andrea, Andy, Alberto, Robin, Luca, Federico, Tommaso, Jacob, Noah, Max, Marc, Anton, Adam, Alex, and more.

To the awesome Pittsburgh friends I have not mentioned yet, including Arman, Anna, Pietro, Antonis, Luca, Derya, and more, thanks for all the fun, food, and drinks, and for your warmth, company, and support. A special thanks also to my lab mates and office mates: Ellen, Sid, Carlos, Brandon, and Roger.

To the Computer Science Department at CMU, and to Meta, thank you for your generous financial support, which has allowed me to pursue my research interests and complete this dissertation.

Finally, to my loving family I express my heartfelt gratitude for your steadfast support.



Contents

I	Introduction, notation, and background	1
1	Introduction	3
1.1	Structure and summary of contributions	7
1.2	Research discussed in this dissertation	10
	Mathematical notation	15
2	Imperfect-information extensive-form games	17
2.1	Game trees and information sets	18
2.2	The player's perspective: Tree-form decision processes	21
2.2.1	Extracting a tree-form decision process from the game tree	22
2.2.2	Notation	24
2.3	Strategies and sequence-form representation	26
2.3.1	Sequence-form representation of strategies	27
2.3.2	Deterministic sequence-form strategies and Kuhn's theorem	29
3	No-regret learning in games	31
3.1	Hindsight rationality and Φ -regret	32
3.1.1	Definition of Φ -regret for a player in the repeated game	32
3.1.2	Relationship between Φ -regret and game-theoretic equilibrium	33
3.1.3	Feedback available to the learning player	34
3.2	Mathematical abstraction of a predictive no- Φ -regret algorithm	35
3.2.1	The canonical optimistic learning setup (COLS) for games	38
3.2.2	The important special case of external regret minimization	39
3.2.3	Reducing Φ -regret minimization to external regret minimization	39
3.2.4	Degrees of predictivity	40
3.3	No-external-regret algorithms for probability simplexes	41
3.3.1	Multiplicative weights update (MWU) and its predictive variant (OMWU)	41
3.3.2	Regret matching (RM), regret matching ⁺ (RM ⁺), and variants	44

II	Computation of coarse-correlated and Nash equilibria	47
4	Composability of learning dynamics and predictive counterfactual regret minimization	49
	<i>Dynamics with $\mathcal{O}_T(1/\sqrt{T})$ convergence to coarse-correlated equilibrium, linear-time iterations, state-of-the-art practical performance in many game instances</i>	
4.1	Contributions and related work	49
4.2	Regret circuits	51
4.2.1	Pictorial depiction of regret circuits	51
4.2.2	Cartesian product	52
4.2.3	Convex hull	54
4.2.4	Affine transformation and Minkowski sum	57
4.2.5	Scaled extension	59
4.3	Predictive counterfactual regret minimization paradigm	63
4.3.1	Predictive RM (PRM) and predictive RM ⁺ (PRM ⁺) algorithms	67
4.3.2	Experimental evaluation	68
5	Notions of distance for sequence-form strategies, and prox methods	71
	<i>Notion of distance for sequence-form strategies with both linear-time projections and polynomial diameter • Exact polynomial-time algorithm for Euclidean projections • RVU-predictive dynamics with linear-time iterations, $\mathcal{O}_T(1/T^{3/4})$ convergence to coarse-correlated equilibrium in multiplayer games, and $\mathcal{O}_T(1/T)$ convergence to Nash equilibrium in two-player zero-sum games</i>	
5.1	Contributions and related work	72
5.2	Preliminaries	73
5.2.1	Distance-generating functions and proximal setups	73
5.2.2	Applications	75
5.2.3	Online mirror descent and follow-the-regularized-leader	75
5.2.4	Bilinear saddle points: Excessive gap technique and mirror prox	76
5.3	Euclidean distance-generating function	79
5.3.1	Exact Euclidean projection algorithm	80
5.4	Distance-generating functions with linear-time projections	84
5.4.1	Dilated distance-generating functions	84
5.4.2	Dilatable global entropy distance-generating function	88
5.5	Experimental evaluation	97
5.5.1	MPROX and EGT without aggressive stepsizing	99
5.5.2	EGT with aggressive stepsizing	100
5.A	Appendix: Properties of SMPL functions	101
6	Strongly predictive learning dynamics, and $\mathcal{O}_T(\log T/T)$ convergence in self play	105
	<i>Strongly predictive dynamics for imperfect-information extensive-form games • State-of-the-art $\mathcal{O}_T(\log T/T)$ convergence to coarse-correlated equilibrium in self play</i>	
6.1	Related work	105
6.2	Contributions	106
6.3	Setup and algorithm pseudocode	107

6.4	Regret analysis	108
6.5	Connecting strong predictivity and logarithmic regret	111
6.6	Implementation and iteration complexity	112
6.6.1	Local proximal oracle	113
6.6.2	Linear maximization oracle	115
6.7	Experimental evaluation	116
6.A	Appendix: Proof details	117
6.A.1	Proof of Proposition 6.1	117
6.A.2	Proof of Proposition 6.2	119
6.A.3	Proof of Corollary 6.1	125
6.A.4	Proof of Theorems 6.2 and 6.3	127
7	State-of-the-art regret dependence on game size via kernelization	131
	<i>Kernelized version of the multiplicative weights update algorithm enables reducing learning in imperfect-information extensive-form games to learning in simultaneous game • Dynamics with theoretical state-of-the-art dependence on game size, $\mathcal{O}_T(\log^4 T/T)$ convergence to coarse-correlated equilibria in self play, and linear-time iterations</i>	
7.1	Contributions and related work	131
7.2	A natural reduction: Running OMWU on the vertices of the strategy set	133
7.3	Kernelized multiplicative weights	135
7.3.1	The sequence-form kernel	135
7.3.2	Using the kernel to simulate the Vertex OMWU algorithm	136
7.4	Efficient evaluation of the sequence-form kernel	138
7.4.1	Worst-case linear complexity for a single evaluation	138
7.4.2	Batched computation and amortized complexity	139
7.5	Regret bound of Kernelized OMWU	141
7.6	Experimental evaluation	142
7.A	Appendix: Proof details	144
7.A.1	Proof of Theorem 7.4	144
7.A.2	Proof of Proposition 7.1	146
III	Computation of extensive-form correlated and team equilibria	151
8	Uncoupled learning of extensive-form correlated equilibrium	153
	<i>Efficient, uncoupled no-regret learning dynamics for extensive-form correlated equilibria in imperfect-information extensive-form games • Practical algorithm for EFCE</i>	
8.1	Contributions and related work	153
8.2	Extensive-form correlated equilibrium and its relation with Φ -regret	155
8.2.1	Trigger agents and trigger deviation functions	155
8.2.2	Convergence to the set of EFCEs via no- Φ -regret dynamics	157
8.3	Construction of no-trigger-regret dynamics	159
8.3.1	Canonical trigger deviation matrices	159
8.3.2	Structural decomposition of canonical trigger deviation matrices	163

8.3.3	Complete algorithm and analysis	167
8.4	Final remarks	168
8.A	Appendix: Inductive computation of fixed points of trigger deviation matrices	169
9	Geometry of correlated strategies, and positive complexity results for optimal EFCE	181
	<i>Positive complexity results around optimal correlated solution concepts in imperfect-information extensive-form games • Enables computation of the polytope of expected utilities that can be reached via EFCE</i>	
9.1	Contributions	181
9.2	Preliminaries, notation, and prior work	182
9.2.1	Polytope of correlation plans Ξ	183
9.2.2	Optimal EFCE as a linear program	184
9.2.3	The von Stengel-Forges polytope \mathcal{V}	186
9.2.4	von Stengel and Forges (2008)'s result for two-player games without chance	186
9.3	Characterization of the relationship between Ξ and \mathcal{V}	187
9.4	Scaled-extension-based structural decomposition for \mathcal{V}	188
9.4.1	Examples and intuition	189
9.4.2	Triangle-freeness	192
9.4.3	Two-player games with public chance moves are triangle-free	193
9.4.4	Computation of the decomposition	194
9.4.5	Integrality of the vertices of \mathcal{V} in triangle-free games	202
9.5	Beyond triangle-freeness	205
9.6	Experimental investigation of utilities reached by EFCE	205
9.6.1	Two-player general-sum games	206
9.6.2	Three-player zero-sum games	206
9.A	Appendix: Additional lemmas on the structure of \mathcal{V}	207
9.B	Appendix: Optimal EFCE as a linear program	209
10	Learning optimal extensive-form correlated equilibria	213
	<i>Learning algorithm converging to optimal (e.g., social-welfare-maximizing) EFCE in imperfect-information extensive-form games • Theoretical and practical state-of-the-art technique for learning optimal EFCE</i>	
10.1	Contributions	213
10.2	Optimal EFCE as a bilinear saddle-point problem	214
10.3	No-regret learning algorithm	216
10.3.1	Construction of weakly-predictive algorithms via regret circuits (Chapter 4)	218
10.3.2	Faster rates using RVU-predictive algorithms (Chapter 5)	219
10.3.3	Remarks on last-iterate convergence	220
10.4	Experimental evaluation	221

11 Learning dynamics for team coordination and collusion	223
<i>Models of team coordination and collusion • Practical state-of-the-art learning algorithm converging to team maxmin equilibria with correlation device (TMECoR) in two-team zero-sum games • Positive complexity results to TMECoR</i>	
11.1 Contributions and related work	225
11.2 Failure of minmax theorem for team maxmin equilibrium	225
11.3 TMECoR as a bilinear saddle-point problem	228
11.3.1 Realization vectors and low-dimensional parameterization	229
11.3.2 Connection with correlation plans and triangle-freeness	231
11.4 Experimental evaluation	232
 IV Beyond perfect rationality	 235
12 Positive complexity results for trembling-hand perfect equilibria	237
<i>Settles the complexity of computing extensive-form perfect equilibria in two-player games, showing it is not harder than Nash</i>	
12.1 Preliminaries on trembling-hand refinements	239
12.2 Contributions and related results	239
12.3 Positive complexity results for two-player EFPE	240
12.3.1 Behavioral perturbation matrices	240
12.3.2 EFPE as a trembling linear complementarity problem (LCP)	242
12.3.3 Existence of a negligible positive perturbation (NPP)	245
12.3.4 Computation of extensive-form perfect equilibria	249
12.3.5 Polynomial-time computation in zero-sum games	252
12.A Appendix: Undomination does not prevent sequential irrationality	253
 13 Computing exact trembling-hand refinements in two-player zero-sum games at scale	 255
<i>Scalable algorithm to compute exact trembling-hand refinements in large-scale games</i>	
13.1 Related work	255
13.2 Refined strategies as solutions to trembling linear programs	256
13.2.1 Extensive-form perfect equilibria as trembling linear programs	257
13.2.2 Quasi-perfect equilibria: Definition and formulation	258
13.2.3 One-sided quasi-perfect equilibrium: Definition and formulation	260
13.2.4 Formulations with sparsified payoff matrices	262
13.3 Basis stability	263
13.3.1 Analytic basis stability condition and existence of stable bases	264
13.3.2 Existence of stable bases	265
13.4 A practical algorithm for finding a TLP limit solution	268
13.4.1 Basis-stability oracle	269
13.4.2 Oracle for non-singular basis matrices	270
13.4.3 Oracle for singular basis matrices	272
13.4.4 Limit of strategy	275

13.5	Experimental evaluation	275
13.5.1	Experiments on small and medium-sized benchmark games	275
13.5.2	Experiments on real-world poker endgames	277
14	Quantal response and regularization towards human play	279
	<i>Learning dynamics biased towards to a given model of behavior • Learning dynamics for logit quantal response equilibria in two-player zero-sum games • Construction of AI players for the game of no-press Diplomacy</i>	
14.1	Contributions and related work	279
14.2	Logit quantal responses and KL-anchored responses	282
14.2.1	Logit quantal responses	282
14.2.2	Logit quantal response as an instances of KL-anchored response	283
14.2.3	Learning dynamics for KL-anchored equilibria	284
14.2.4	Imitation-anchored responses in imperfect-information extensive-form games	285
14.3	Modeling uncertainty on the anchoring coefficients	286
14.3.1	A technical lemma needed in the analysis	288
14.3.2	Regret analysis	290
14.3.3	Last-Iterate Convergence in Two-Player Zero-Sum Games	293
14.4	Experimental evaluation in no-press Diplomacy	300
14.4.1	Background on Double Oracle Reinforcement learning for Action exploration (DORA)	300
14.4.2	Training of our bot Diplodocus	300
14.4.3	Experimental setup	301
14.4.4	Performance compared to prior algorithms	302
14.4.5	Experiments against human players	303
	Conclusions and future work	305
	Bibliography	308
	Appendices	324
A	Description of benchmark games used in experimental evaluations	325
A.1	Description of game instances	325
A.1.1	Battleship (B)	325
A.1.2	Liar’s dice (D)	326
A.1.3	Goofspiel (G)	326
A.1.4	Limited-information Goofspiel (GL)	326
A.1.5	Kuhn poker (K)	326
A.1.6	Leduc poker (L)	327
A.1.7	Pursuit-evasion (P)	327
A.1.8	River Endgame (REL)	328
A.1.9	Sheriff (S)	328

A.1.10 Double-dummy bridge endgame (T, TP)	329
A.1.11 Ridesharing game (RS)	329
A.1.12 Small matrix (SM)	330
A.2 Game dimensions	331
B Summary of notation	333
B.1 Tree-form decision processes	334
B.2 Correlated strategies	335
B.3 Mathematical notation	335

List of Algorithms

3.1	Predictive multiplicative weights update algorithm (OMWU)	42
3.2	Regret matching (RM)	44
3.3	Regret matching ⁺ (RM ⁺)	44
3.4	Discounted regret matching (Discounted RM)	45
3.5	Linear regret matching (Linear RM)	45
4.1	Regret circuit for $\mathcal{X} \times \mathcal{Y}$	52
4.2	Regret circuit for $\text{co}\{\mathcal{X}, \mathcal{Y}\}$	55
4.3	Predictive CFR (weakly-predictive no-external-regret algorithm for sequence-form strategy polytope)	65
4.4	(Continued) Predictive CFR (weakly-predictive no-external-regret algorithm for sequence-form strategy polytope)	66
4.5	Predictive regret matching (PRM)	68
4.6	Predictive regret matching ⁺ (PRM ⁺)	68
5.1	Predictive FTRL	76
5.2	Predictive OMD	76
6.1	Log-Regularized Lifted Optimistic FTRL (LRL-OFTRL)	107
6.2	Proximal Newton method (Tran-Dinh, Kyriallidis, and Cevher, 2015)	115
7.1	Vertex OMWU	134
7.2	Kernelized OMWU (KOMWU)	139
8.1	No-external-regret algorithm $\mathcal{R}_{\hat{\sigma}}$ for set $\Lambda_{\hat{\sigma}} := \{\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}} : \mathbf{x}_{\hat{\sigma}} \in \mathcal{Q}_{\neq j}\}$	164
8.2	No-regret algorithm $\tilde{\mathcal{R}}$ for the set $\text{co}\Psi = \text{co}\{\Lambda_{\hat{\sigma}} : \hat{\sigma} \in \Sigma\}$	166
8.3	No-trigger-regret algorithm for sequence-form strategy polytope \mathcal{Q}	167
8.4	Extend($\mathbf{T}, J, j^*, \mathbf{x}$)	172
8.5	FixedPoint(\mathbf{T})	174
9.1	FillOutRow($(\sigma_1, \sigma_2), j_1, \mathcal{S}, \mathcal{D}$)	196
9.2	FillOutColumn($(\sigma_1, \sigma_2), j_2, \mathcal{S}, \mathcal{D}$)	196

9.3	Decompose($(\sigma_1, \sigma_2), \mathcal{S}, \mathcal{D}$)	199
12.1	Find-EFPE	252
13.1	Naïve algorithm for finding a limit solution to a TLP $P(\epsilon)$	268
14.1	DiL-piKL algorithm, for a generic player i	287

List of Figures

2.1	Game tree for the game of Kuhn poker. Utilities are shown for Player 1 only; utilities for Player 2 are the opposite Player 1's. Due to space constraints, we used 'chk.' as an abbreviation for the 'check' action.	18
2.2	Tree-form decision process faced by Player 1 in the game of Kuhn poker.	22
2.3	Small game tree used in the example.	23
2.4	(Left) Tree-form decision process considered in the example. (Right) The constraints that define the sequence-form polytope \mathcal{Q} for Player 1 (besides nonnegativity) in the TFDP shown on the left.	28
2.5	(Right) Examples of deterministic sequence-form strategies for the small tree-form decision process on the left.	30
4.1	Regret circuit for the image $H(\mathcal{X})$ of \mathcal{X} under the affine transformation H	58
4.2	Regret circuit for the Minkowski sum $\mathcal{X} + \mathcal{Y}$	59
4.3	Regret circuit for the scaled extension $\mathcal{X} \stackrel{h}{\triangleleft} \mathcal{Y}$	61
4.4	Sequential decision-making problem used in the example.	63
4.5	Performance of PCFR ⁺ , CFR ⁺ , DCFR, and LCFR on nine games. In all plots, the x axis is the number of iterations of each algorithm. For each game, the top plot shows that the Nash gap on the y axis (on a log scale), the bottom plot shows and the average prediction error (on a log scale).	69
5.1	Performance of the EGT algorithm instantiated with the two entropy DGFs across nine games. The x-axis shows the number of EGT iterations, and the y-axis shows the distance to Nash equilibrium.	99
5.2	Performance of the MPROX algorithm instantiated with the two entropy DGFs across nine games.	100
5.3	Performance of the EGT/AS algorithm instantiated with the two entropy DGFs, as well as aggressive stepsizing, μ balancing, and initial μ fitting.	101
6.1	The lifting operation performed by the LRL-OFTRL algorithm on the strategy set.	108
6.2	The regret of the players when they follow our learning dynamics, LRL-OFTRL. The x -axis indexes the iteration, while the y -axis the regret. The scale on the x -axis is <i>logarithmic</i> . We observe that the regret of each player grows as $\mathcal{O}_T(\log T)$, verifying Theorem 6.3.	117

7.1	Overview of reduction template from \mathcal{Q} to Δ^{Π} . The matrix \mathbf{V} has the vertices Π of \mathcal{Q} as columns.	134
7.2	Experimental comparison of KOMWU with CFR.	143
7.3	Experimental comparison of KOMWU with DOMWU for different choices of learning rates.	144
8.1	Canonical trigger deviation matrices. Entries shaded with dark gray represent the entries of the matrix defined in the second case of Equation (8.5). Given trigger sequence $\hat{\sigma} \in \Sigma^*$, all indices (σ_r, σ_c) such that $\sigma_r, \sigma_c \succ \hat{\sigma}$ are shaded with light gray.	161
8.2	Pictorial depiction of our no-(co Ψ)-reget algorithm for the set of sequence-form strategies \mathcal{Q} . For notational convenience we let $\Sigma^* := \{1, \dots, m\}$.	168
9.1	Overview of the connections among this chapter's results.	189
9.2	Three examples of extensive-form games with increasingly complex information partitions. As usual, the crossed nodes belong to the chance player, the black round nodes belong to Player 1, the white round nodes belong to Player 2, the gray round sets define information sets, and the white squares denote terminal nodes (payoffs are omitted as they are irrelevant). The numbers along the edges define the action names; for clarity we assign unique action names at each information set.	189
9.3	Polytope of expected utilities for the players that can be reached via extensive-form correlated equilibria in three standard two-player general-sum imperfect-information extensive-form games.	206
9.4	Polytope of expected utilities for the players that can be reached via extensive-form correlated equilibria in three standard three-player zero-sum imperfect-information extensive-form games.	207
11.1	Matching pennies game.	226
12.1	Perfect-information game in which a sequentially-irrational Nash equilibrium is highlighted.	237
12.2	Small <i>imperfect</i> -information game in which a sequentially-irrational Nash equilibrium is highlighted.	238
12.3	Modified <i>Guess-the-Ace</i> games. The highlighted equilibrium is undominated, and yet sequentially irrational.	253
13.1	High-level overview of the steps of our practical algorithm for finding a TLP limit solution.	269
13.2	Example of a situation where the basis matrix $\mathbf{B}(\epsilon)$ is singular at $\epsilon = 0$.	273
A.1	The graph on which the search game is played.	327
A.2	<i>Left</i> : Graph configuration 1, used for RS212 RS213 . <i>Right</i> : Graph configuration 2, used for RS222 RS223 . In both cases the position of the two drivers is randomly chosen at the beginning of the game, edge costs are unitary, and the reward for each node is indicated between curly brackets.	330

List of Tables

2.1	Summary of basic notation for TFDPs. In cases where it is important to specify the player to which the different quantities belong, a subscript with the player will be added.	26
4.1	Bottom-up construction rules for sequence-form strategy spaces. e_i denotes the i -th indicator vector, that is, the vector whose entries are all 0 except for the entry in position i , which is set to 1.	67
5.1	(A) Various measures of the size of each of the games that we test algorithms on. (B), (C) The magnitude of the dilated entropy DGF and dilatable global entropy DGF weights.	97
7.1	Properties of various no-regret algorithms for imperfect-information extensive-form games. All algorithms take linear time to perform an iteration. The first set of rows are for non-predictive algorithms. The second set of rows are for predictive algorithms. The regret bounds are per player and apply to multiplayer general-sum games. They depend on the maximum number of actions A available at any decision node, the maximum ℓ_1 norm $\ \mathcal{Q}\ _1 = \max_{q \in \mathcal{Q}} \ q\ _1$ over the player’s sequence-form strategy polytope \mathcal{Q} , the depth D of the decision polytope, and the number of players m . Optimistic algorithms have better asymptotic regret, but worse dependence on the game constants m , A , and $\ \mathcal{Q}\ _1$. Note that our algorithms achieve better dependence on $\ \mathcal{Q}\ _1$ compared to all existing algorithms. †Last-iterate convergence results are for two-player zero-sum games, and some results rely on the assumption of a unique Nash equilibrium—see Section 7.5 for details. ‡We remark that LRL-OFTRL enjoys polynomial-time iterations (ignoring a $\log \log T$ dependence), but <i>not</i> linear-time iterations unlike the other methods in the table. *See C.-W. Lee, Kroer, and Luo (2021).	132
9.1	Additional notation used when dealing with correlation of strategy spaces.	183
10.1	Experimental comparison between our learning-based approach (‘Ours’, Section 10.3.1) and using the commercial LP solver Gurobi on the linear programming formulation of optimal EFCE (Proposition 9.1), both for computing an optimal EFCE within an optimality and feasibility tolerance set to 1% of the payoff range of the game.	221

10.2	Experimental comparison between our learning-based approach ('Ours', Section 10.3.1) and using the commercial LP solver Gurobi on the linear programming formulation of optimal EFCE (Proposition 9.1), both for computing an optimal EFCE within an optimality and feasibility tolerance set to 0.1% of the payoff range of the game.	222
11.1	Comparison between TME, TMECor and Team Nash equilibrium.	224
11.2	Runtime of our learning algorithm (column ' This chapter '), compared to prior state-of-the-art algorithms based on linear programming (' ZS22 ', B. H. Zhang and Sandholm, 2022b) and column generation (' ZFCS22 ', B. H. Zhang, Farina, Celli, and Sandholm, 2022) respectively, on several standard parametric benchmark games. '—': Missing or unknown value.	233
12.1	Complexity of computing trembling-hand equilibrium refinements in two-player games.	240
13.1	Comparison between different methods of computing Nash equilibrium refinements in two-player zero-sum small and medium-sized benchmark games.	276
13.2	Unsparsified and sparsified size of River endgames encountered by Libratus during the "Brains vs AI" competition.	277
13.3	Computation of refined Nash equilibria in real poker endgames using our algorithm.	278
14.1	Performance of different algorithms. Agents above the line were retrained. Agents below the line were evaluated using the models and the parameters provided by the authors. The \pm shows one standard error.	302
14.2	Performance of four different agents in a population of human players, ranked by Elo, among all 43 participants who played at least 5 games. The \pm shows one standard error.	304
A.1	Dimensions of game instances used in this dissertation.	331
A.2	(Continued) Dimensions of game instances used in this dissertation.	332

Part I

**Introduction, notation, and
background**

Chapter 1

Introduction

Designing machines that can reason strategically and make optimal decisions even in the presence of imperfect information and adversarial behavior is a fundamental goal of *artificial intelligence*. The study of strategic decision-making is a rich discipline involving models and techniques from game theory, optimization, statistics, and others, and with applications ranging from markets (including financial markets, auctions, *etc.*), to defense (including cybersecurity, airport security, patrolling routes, *etc.*), to recreational games, to logistics and procurement, and more.

Our understanding of the computational aspects of game-theoretic decision-making is fairly advanced in strategic interactions in which all players act once and simultaneously, selecting (possibly randomizing their choice) an action from a pre-determined finite set of possibilities. These interactions, called *normal-form games*, are arguably the best studied in game theory, and have been the standard model in the theory of learning in games and online learning more generally for a long time.

In contrast, this dissertation focuses on the more realistic and challenging *tree-form* decision-making problems in which decision makers can face multiple decisions interleaved with observations about the way previous decisions affected the (partially hidden and possibly adversarial) multiagent environment. These games go under the name of *imperfect-information extensive-form games*—where the term *extensive-form* is a standard term in the literature, meaning tree-form—and can model both sequential and simultaneous moves, stochastic events (such as drawing a random card from a deck), and uncertainty about the state of the game. By allowing interleaved decisions and observations, imperfect-information extensive-form games present unique challenges compared to (and require different techniques from) normal-form games. For one, due to the presence of observations, the number of strategies in imperfect-information extensive-form games is usually exponential in the number of possible choices, resulting in additional computational challenges and a more nuanced complexity landscape than interactions that terminate after a single action. I believe that tackling the added complexity of imperfect-information extensive-form games

is necessary to bring strategic decision-making to the real world, where unlike single-choice interactions, a decision maker rarely has to face just one choice, much less independently of any observation about the state of the environment.

This dissertation seeks to provide solid theoretical and algorithmic foundations for strategic, game-theoretic decision-making in imperfect-information extensive-form games. We identify three main areas of contributions (a more detailed description of the contributions of each chapter, as well as of the dependency relations among the chapters is given below, in [Section 1.1](#)).

New and state-of-the-art learning dynamics for imperfect-information games Learning dynamics provide positive, constructive answers to the following fundamental question: Can a player that repeatedly plays a game follow rules to refine their own strategy after each match, so as to guarantee reaching a form of game-theoretic equilibrium in the long run? Due to their uncoupled nature, learning dynamics are typically extremely practical methods to compute game-theoretic solutions to large games. While the existence of powerful learning dynamics leading to the most important notions of game-theoretic equilibrium in general normal-form games (such as correlated and coarse-correlated equilibrium) had been a celebrated result for a long time, much less was known in general imperfect-information games.

This dissertation provides several contributions to the theory of learning dynamics in multiplayer imperfect-information games. Here we mention four; more work on learning in imperfect-information extensive-form games is mentioned later in this section.

- We construct the first efficient (*i.e.*, with polynomial-time iteration complexity) learning dynamics for extensive-form correlated equilibrium in imperfect-information extensive-form games. These dynamics resolve a long-standing open problem in the theory of learning in imperfect-information extensive-form games, and beget the practical state of the art for computing extensive-form correlated equilibria in multiplayer imperfect-information extensive-form games.
- We show that learning coarse-correlated equilibria in imperfect-information extensive-form games can be efficiently reduced to learning in normal-form games by means of a *kernelized* version of the classic multiplicative weights update algorithm. This reduction enables transferring results that were previously known to hold only in normal-form games to general imperfect-information extensive-form games, thereby shortening the gap between the two domains. The existence of such a reduction contradicts decades of popular beliefs, and leads to regret bounds with state-of-the-art dependence on the game size, improving on *all* prior learning algorithms for imperfect-information extensive-form games. This technique is also the first to show that $\mathcal{O}_T(\log^4 T/T)$ convergence to coarse-correlated equilibrium can be achieved in imperfect-information extensive-form games while retaining linear-time strategy updates. Before this result, the bound was only known to be possible for

normal-form games (Daskalakis, Fishelson, and Golowich, 2021), and the best-known result for imperfect-information extensive-form games was of order $\mathcal{O}_T(1/\sqrt{T})$, exponentially worse.

- Using a different technique, we further improve the rate of convergence to coarse-correlated equilibrium in both imperfect-information extensive-form games and normal-form games with any number of players. Specifically, we construct learning dynamics leading to coarse-correlated equilibrium in multiplayer imperfect-information extensive-form games at the theoretical state-of-the-art, *near-optimal* rate of $\mathcal{O}_T(\log T/T)$, while enjoying $\mathcal{O}_T(\log \log T)$ per-iteration complexity. We remark that the technique we use to establish the result applies to the larger class of *concave games* (of which imperfect-information extensive-form games and normal-form games are instances), establishing, for the first time, the existence of near-optimal polynomial-time learning dynamics for any such games.
- Finally, we propose new learning dynamics with state-of-the-art empirical convergence to Nash equilibrium in a large number of two-player zero-sum imperfect-information extensive-form games.

Structure of correlated play in imperfect-information extensive-form games Several problems of interest, including the computation of welfare-maximizing correlated solution concepts and optimal strategies for teams in the absence of communication, can be formulated as linear optimization problems over the set of correlated strategies of (possibly subsets of) the players. To enable progress on those problems, a part of this dissertation focuses on developing new technique and foundational results regarding the structure and geometry of correlated distributions of play between players. Here, we highlight the most significant findings.

- We give positive complexity results for the problem of optimizing a linear function over the polytope of correlated strategies between two players. Specifically, we isolate a condition—called *triangle freeness* and automatically satisfied by all two-player games whose chance moves are publicly observed—that guarantees that the set of correlated strategies can be represented via polynomially many linear constraints, rendering optimization tractable. When triangle freeness is not satisfied, we also discuss how the techniques can be generalized to yield state-of-the-art *parameterized* complexity results. These complexity thresholds are unique to imperfect-information extensive-form games and do not have an equivalent in normal-form games, highlighting the intricate and fascinating structure of the former. We also remark that the techniques we introduce in this dissertation also enable to study, for the first time, what equilibrium points can be supported by extensive-form correlated equilibria, highlighting the richness of behavior that arises from introducing a correlation device in imperfect-information extensive-form games.

- We leverage our results regarding the structure of the polytope of correlated strategies to construct the first no-regret learning algorithm with guaranteed convergence to the set of optimal (for example, welfare-optimizing) extensive-form correlated equilibria, leading to the practical state-of-the-art algorithm for the problem.
- Finally, we draw new connections between the study of correlated strategies and the study of optimal strategies in two-team zero-sum imperfect-information extensive-form games. By leveraging these connections, we give the first positive complexity result for the latter problem, as well as the practical state-of-the-art learning algorithm.

Learning and equilibrium computation in the presence of imperfect players Finally, the last part of this dissertation focuses on learning and computational techniques for imperfect (that is, not perfectly rational) players.

- We settle the complexity of computing extensive-form perfect equilibria in two-player games, a recognized open problem (Miltersen and Sørensen, 2006), showing that it is not harder than computing a generic Nash equilibrium. Extensive-form perfect equilibria (EFPEs) are *trembling-hand equilibrium refinements*, subsets of Nash equilibria that satisfy some additional robustness guarantees to small mistakes of the opponents. In particular, EFPEs guarantee that the equilibrium strategies are optimal even in parts of the game tree that are reached only if a player has made a mistake, and are therefore appealing concepts for AI players that need to be played against human players.
- We also give practical state-of-the-art algorithms that enable, for the first time, to find exact trembling-hand refinements of Nash equilibrium (including EFPE and other) at scale in two-player zero-sum imperfect-information extensive-form games. As we show, our algorithm is able to scale to real games with up to half a billion terminal nodes, an unprecedented scale for equilibrium refinements.
- Finally, we introduce a methodology for constraining learning dynamics to remain close to a given anchor policy for each player. As a special case, such anchored learning algorithms can be used to compute logit quantal response equilibria in imperfect-information extensive-form games. We empirically investigate the use of these learning algorithms in the construction of AI agents for the game of no-press Diplomacy, showing that our approach enables the AI agents to comply with human conventions. In a 200-game no-press Diplomacy tournament involving 62 human participants spanning skill levels from beginner to expert, two AI bots trained with our algorithm both achieved a higher average score than all other participants who played more than two games, and ranked first and third according to an Elo ratings model. We also remark that the same methodology was recently used in building Cicero, an AI agent for playing the full version of Diplomacy using natural language communication.

1.1 Structure and summary of contributions

We now summarize the structure and main contributions of each chapter of this dissertation. A detailed list of contributions presented in each chapter is placed at the beginning of the chapter itself, together with a discussion of the most relevant related research.

Chapters 2 and 3 give an approachable overview of imperfect-information extensive-form games and learning in games. When presenting the key ideas of the theory of learning in games, we opted for a modern approach, introducing the concept of predictivity early on as a standard aspect in the definition of a no-regret algorithm rather than as a later development.

Chapter 4 introduces *regret circuits*, a methodology for constructing predictive no-regret algorithms for composite sets obtained via convexity-preserving operations. We use the framework to construct a predictive version of the counterfactual regret minimization (CFR) algorithm, a no-external-regret and parameter-free algorithm for the strategy set of a player in an imperfect-information extensive-form games. We find that our predictive algorithms obtained via regret circuits achieve state-of-the-art empirical convergence to the set of Nash equilibria in two-player zero-sum games in a majority of instances we test on. In addition, the material presented in the chapter will find applications in several other chapters, such as in the construction of the first learning dynamics for extensive-form correlated equilibria (**Chapter 8**), welfare-maximizing equilibria (**Chapter 10**), and team strategies (**Chapter 11**).

Chapter 5 investigates several fundamental questions about the metric structure of strategy spaces in imperfect-information extensive-form games, with applications to online and offline optimization methods. The contributions are multiple, spanning different notions of distance between strategies that arise in practice, and laying foundations applicable to a wide range of optimization techniques. Perhaps most notably, the chapter introduces the first notion of distance in imperfect-information games that enables projecting points onto the strategy polytope in linear time (a key operation needed in projected optimization methods such as mirror descent or projected gradient descent), while at the same time guaranteeing polynomial (in the game tree size) distance between any two strategies.

Chapter 6 provides state-of-the-art regret bounds for learning in imperfect-information extensive-form games with an arbitrary number of players, constructing learning dynamics with guaranteed convergence to coarse correlated equilibria at a rate of $\mathcal{O}_T(\log T/T)$, all while enjoying an $\mathcal{O}_T(\log \log T)$ iteration complexity. This settles, for the positive, the question of whether near-optimal no-regret learning can be achieved in general imperfect-information extensive-form games, and improves the state of the art for nonsequential games as well. To establish the result, we design new techniques that are of independent interest, as they apply to the larger class of *concave* games, which contains imperfect-information extensive-form

games. This chapter depends on certain parts of [Chapter 5](#) as prerequisites, as it builds on new results related to efficient proximal updates for strategies in imperfect-information extensive-form games.

[Chapter 7](#) shows that learning in imperfect-information extensive-form games can be efficiently reduced to learning in nonsequential games by means of a *kernelized* version of the classic multiplicative weights update algorithm. This reduction enables transferring, in a black-box fashion, results that were previously known to apply only in nonsequential games to general imperfect-information extensive-form games, thereby shortening the gap between the two domains. The existence of such a reduction might come as a surprise, as it contradicts decades of popularly held beliefs. Even more surprisingly, it leads to regret bounds with state-of-the-art dependence on the game size, improving on *all* prior algorithms, despite these being designed specifically for imperfect-information extensive-form games. The algorithm presented in this chapter is also the first to show that polylogarithmic regret can be achieved in imperfect-information extensive-form games while retaining linear-time strategy updates.

[Chapter 8](#) provides the first uncoupled, polynomial-time learning dynamics leading to extensive-form correlated equilibrium (EFCE), the natural generalization of correlated equilibrium to imperfect-information extensive-form games. The material presented in this chapter closes an longstanding open problem in the literature, and parts of it were recognized with a paper award at NeurIPS and publication in the Journal of the ACM. This chapter makes use of the formalism of regret circuits introduced in [Chapter 4](#).

[Chapter 9](#) focuses on the structure and geometry of correlated distributions of play between players. As we show in this chapter and in the following ones, several problems of interest, including the computation of welfare-maximizing correlated solution concepts and optimal strategies for teams in the absence of communication, can all be formulated as linear optimization problems over the set of correlated strategies of (possibly subsets of) the players. One of the key results of the chapter is establishing new positive description complexity results, isolating a condition—called *triangle freeness* and automatically satisfied by all two-player games whose chance moves are publicly observed—that guarantees that the set of correlated strategies can be represented via polynomially many linear constraints, rendering optimization tractable. Along the way, several new concepts and techniques of likely independent interest are introduced to study the combinatorial structure of correlation. Beyond triangle freeness, we also discuss how the ideas of the chapter can be extended to yield state-of-the-art parameterized complexity results for those optimization problems over correlated strategies. The complexity thresholds presented in this chapter are very different from nonsequential games and highlight the intricate and fascinating structure of imperfect-information extensive-form games. At the end of the chapter, we use

our techniques to study, for the first time, what equilibrium points can be supported by extensive-form correlated equilibria, highlighting the richness of behavior that arises from introducing a correlation device in imperfect-information extensive-form games.

Chapter 10 combines the structural understanding of correlated strategies gained from [Chapter 9](#) together with the formalism of regret circuits introduced in [Chapter 4](#) to design the first learning algorithm with guaranteed convergence to *optimal* (for example, welfare-maximizing) extensive-form correlated equilibrium in imperfect-information extensive-form games. As we show in the experimental evaluation at the end of the chapter, the use of learning begets the practical state-of-the-art approach for computing optimal correlated solution concepts in multiplayer imperfect-information extensive-form games.

Chapter 11 studies the problem of constructing optimal strategies in two-team zero-sum games, where the players cannot communicate during the game. After discussing alternative notions of game-theoretic optimality for the setting, we isolate one, TMECor, as the most appropriate for the setting. Then, we establish strong ties between the computation of TMECor strategies for a team and the geometry of correlation plans studied in [Chapter 9](#). By leveraging that connection, we provide the current state-of-the-art complexity result for the problem of computing optimal team strategies, as well as the practical state-of-the-art learning algorithm for TMECor. This chapter combines ideas presented in [Chapters 4, 9](#) and [10](#).

Chapter 12 settles the complexity of computing an extensive-form perfect equilibrium—arguably the best-known and most-studied sequentially-rational refinement of the Nash equilibrium—in two-player games, a recognized problem. Extensive-form perfect equilibrium (EFPE) resolves certain unsatisfactory behaviors that can be prescribed by Nash equilibrium when the opponent makes a mistake. This is especially relevant when using Nash equilibrium as the optimization target for AI bots playing games against humans. Our results show a positive result: computing an EFPE is not harder, from a computational complexity standpoint, than computing a Nash equilibrium, showing that the benefits of EFPE come at no additional (theoretical) cost. However, we remark that the technique we use to establish the complexity result in this chapter is hardly scalable, and devote the next chapter to constructing scalable algorithms that can compute EFPE and other equilibrium refinements at scale for the first time.

Chapter 13 presents the first algorithm able to compute exact trembling-hand equilibrium refinements in large imperfect-information extensive-form games, significantly extending some key ideas presented in [Chapter 12](#). We demonstrate the scalability of our algorithm for computing exact extensive-form perfect, quasi-perfect, and one-sided quasi-perfect (a new solution concept we introduce), in poker endgames that were encountered as part

of the Brains-vs-AI competition. These are games with hundreds of million of terminal states, and provide a natural testing ground for research on refinements robust to small mistakes of the (human) players. The scale of the games that can be handled using the technology developed in this chapter is several order of magnitude larger than anything that was possible before, and enables—for the first time—studying the empirical performance of equilibrium refinements in large games.

Chapter 14 introduces a methodology for anchoring learning dynamics to remain close to given strategies for each player. As a special case, such anchored learning algorithms can be used to compute logit quantal response equilibria in imperfect-information extensive-form games. We empirically investigate the use of these learning algorithms in the construction of AI agents for the game of no-press Diplomacy, showing that our approach enables the AI agents to comply with human conventions. This research was recognized with an outstanding paper honorable mention at ICLR. As mentioned earlier in the introduction, in a 200-game no-press Diplomacy tournament involving 62 human participants spanning skill levels from beginner to expert, two AI bots trained with our algorithm both achieved a higher average score than all other participants who played more than two games, and ranked first and third according to an Elo ratings model. In the chapter, we also remark that the same methodology was recently used in building Cicero, an AI agent for playing the full version of Diplomacy using natural language communication, which was recently featured on *Science*.

1.2 Research discussed in this dissertation

This dissertation combines and builds on a selection of articles that I have published during my doctoral studies. In addition, this dissertation includes a number of improvements to the published material, with the goal of offering a cohesive and approachable treatment of the subject. These additions include new text, numerous examples, new figures and diagrams, remarks about connections across different chapters, and technical improvements that I discovered in hindsight and never published. Through this additional amount of work I hope that this dissertation will help future researchers interested in computational aspects of imperfect-information extensive-form games, serving as an approachable reference and partially filling a current void in the literature.

Parts of this dissertation have appeared in the literature as follows (in chronological order for each chapter).

Chapter 4.

- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2019c). “Regret Circuits: Composability of Regret Minimizers”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2021b). “Faster Game Solving via Predictive Blackwell Approachability: Connecting Regret Matching and Mirror Descent”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Chapter 5.

- Kroer, Christian, Gabriele Farina, and Tuomas Sandholm (2018b). “Solving Large Sequential Games with the Excessive Gap Technique”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2019b). “Optimistic Regret Minimization for Extensive-Form Games via Dilated Distance-Generating Functions”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2021a). “Better Regularization for Sequential Decision Spaces: Fast Convergence Rates for Nash, Correlated, and Team Equilibria”. In: *ACM Conference on Economics and Computation*.

Chapter 6.

- Anagnostides, Ioannis, Gabriele Farina, Christian Kroer, Andrea Celli, and Tuomas Sandholm (2022). “Faster No-Regret Learning Dynamics for Extensive-Form Correlated and Coarse Correlated Equilibrium”. In: *Proceedings of the ACM Conference on Economics and Computation (EC)*.
- Anagnostides, Ioannis, Gabriele Farina, Christian Kroer, Chung-Wei Lee, Haipeng Luo, and Tuomas Sandholm (2022). “Uncoupled Learning Dynamics with $O(\log T)$ Swap Regret in Multiplayer Games”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Anagnostides, Ioannis, Gabriele Farina, and Tuomas Sandholm (2023). “Near-Optimal Φ -Regret Learning in Extensive-Form Games”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.

Chapter 7.

- Farina, Gabriele, Chung-Wei Lee, Haipeng Luo, and Christian Kroer (2022). “Kernelized Multiplicative Weights for 0/1-Polyhedral Games: Bridging the Gap Between Learning in Extensive-Form and Normal-Form Games”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.

Chapter 8.

- Celli, Andrea, Alberto Marchesi, Gabriele Farina, and Nicola Gatti (2020). “No-Regret Learning Dynamics for Extensive-Form Correlated Equilibrium”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele, Tommaso Bianchi, and Tuomas Sandholm (2020). “Coarse Correlation in Extensive-Form Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Farina, Gabriele, Andrea Celli, Alberto Marchesi, and Nicola Gatti (2022). “Simple Uncoupled No-regret Learning Dynamics for Extensive-form Correlated Equilibrium”. In: *Journal of the ACM* 69.6. URL: <https://dl.acm.org/doi/10.1145/3563772>.

Chapter 9.

- Farina, Gabriele, Chun Kai Ling, Fei Fang, and Tuomas Sandholm (2019a). “Correlation in Extensive-Form Games: Saddle-Point Formulation and Benchmarks”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele, Chun Kai Ling, Fei Fang, and Tuomas Sandholm (2019b). “Efficient Regret Minimization Algorithm for Extensive-Form Correlated Equilibrium”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele and Tuomas Sandholm (2020). “Polynomial-Time Computation of Optimal Correlated Equilibria in Two-Player Extensive-Form Games with Public Chance Moves and Beyond”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Chapter 10.

- Farina, Gabriele, Andrea Celli, Nicola Gatti, and Tuomas Sandholm (2021). “Connecting Optimal Ex-Ante Collusion in Teams to Extensive-Form Correlation: Faster Algorithms and Positive Complexity Results”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.

- Zhang, Brian Hu, Gabriele Farina, Andrea Celli, and Tuomas Sandholm (2022). “Optimal Correlated Equilibria in General-Sum Extensive-Form Games: Fixed-Parameter Algorithms, Hardness, and Two-Sided Column-Generation”. In: *Proceedings of the ACM Conference on Economics and Computation (EC)*.
- Zhang, Brian Hu, Gabriele Farina, Ioannis Anagnostides, Federico Cacciamani, Stephen McAleer, Andreas Haupt, Andrea Celli, Nicola Gatti, Vincent Conitzer, and Tuomas Sandholm (2023). *Learning and Steering toward Optimal Equilibria and Mechanisms*.

Chapter 11.

- Farina, Gabriele, Andrea Celli, Nicola Gatti, and Tuomas Sandholm (2018). “Ex Ante Coordination and Collusion in Zero-Sum Multi-Player Extensive-Form Games”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Zhang, Brian Hu, Gabriele Farina, and Tuomas Sandholm (2023). “Team Belief DAG Form: A Concise Representation for Team-Correlated Game-Theoretic Decision Making”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.

Chapter 12.

- Farina, Gabriele and Nicola Gatti (2017). “Extensive-Form Perfect Equilibrium Computation in Two-Player Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Chapter 13.

- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2017). “Regret Minimization in Behaviorally-Constrained Zero-Sum Games”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Kroer, Christian, Gabriele Farina, and Tuomas Sandholm (2017). “Smoothing Method for Approximate Extensive-Form Perfect Equilibrium”. In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.
- Farina, Gabriele, Nicola Gatti, and Tuomas Sandholm (2018). “Practical Exact Algorithm for Trembling-Hand Equilibrium Refinements in Games”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele and Tuomas Sandholm (2021a). “Equilibrium Refinement for the Age of Machines: The One-Sided Quasi-Perfect Equilibrium”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

- Farina, Gabriele and Tuomas Sandholm (2022). “Fast Payoff Matrix Sparsification Techniques for Structured Extensive-Form Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Chapter 14.

- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2019a). “Online Convex Optimization for Sequential Decision Processes and Extensive-Form Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Bakhtin, Anton, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, Athul Paul Jacob, Mojtaba Komeili, Karthik Konath, Minae Kwon, Adam Lerer, Mike Lewis, Alexander H. Miller, Sasha Mitts, Adithya Renduchintala, Stephen Roller, Dirk Rowe, Weiyan Shi, Joe Spisak, Alexander Wei, David Wu, Hugh Zhang, and Markus Zijlstra (2022). “Human-level play in the game of Diplomacy by combining language models with strategic reasoning”. In: *Science* 378.6624. URL: <https://www.science.org/doi/pdf/10.1126/science.ade9097>.
- Jacob, Athul Paul, David J. Wu, Gabriele Farina, Adam Lerer, Hengyuan Hu, Anton Bakhtin, Jacob Andreas, and Noam Brown (2022). “Modeling Strong and Human-Like Gameplay with KL-Regularized Search”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Bakhtin, Anton, David J Wu, Adam Lerer, Jonathan Gray, Athul Paul Jacob, Gabriele Farina, Alexander H Miller, and Noam Brown (2023). “Mastering the Game of No-Press Diplomacy via Human-Regularized Reinforcement Learning and Planning”. In: *Proceedings of the International Conference on Learning Representations (ICLR)*.

Mathematical notation

Throughout the dissertation, we make use of the following mathematical notational conventions.

Vectors and matrices

- Vectors and matrices are marked in bold.
- Given a finite set $S = \{s_1, \dots, s_n\}$, we denote as \mathbb{R}^S (resp., $\mathbb{R}_{\geq 0}^S$) the set of real (resp., nonnegative real) $|S|$ -dimensional vectors whose entries are denoted as $\mathbf{x}[s_1], \dots, \mathbf{x}[s_n]$.
- Similarly, given finite sets S, S' , we denote as $\mathbb{R}^{S \times S'}$ (resp., $\mathbb{R}_{\geq 0}^{S \times S'}$) the set of real (resp., nonnegative real) $S \times S'$ square matrices \mathbf{M} whose entries are denoted as $\mathbf{M}[s_r, s_c]$ ($s_r \in S, s_c \in S'$), where s_r corresponds to the row index and s_c to the column index.

Standard sets

- We denote the set of real numbers as \mathbb{R} , the set of nonnegative real numbers as $\mathbb{R}_{\geq 0}$, and the set $\{1, 2, \dots\}$ of positive integers as \mathbb{N} .
- The set $\{1, \dots, n\}$, where $n \in \mathbb{N}$, is compactly denoted as $\llbracket n \rrbracket$.
- The empty set is denoted as $\{\}$.
- Given a finite set S , we denote by Δ^S the simplex $\Delta^S := \{\mathbf{x} \in \mathbb{R}_{\geq 0}^S : \sum_{s \in S} \mathbf{x}[s] = 1\}$. The symbol Δ^n , with $n \in \mathbb{N}$, is used to mean $\Delta^{\llbracket n \rrbracket}$.
- Given a finite set S , we use the symbol $\mathbb{S}^S \subseteq \mathbb{R}_{\geq 0}^{S \times S}$ to denote the set of stochastic matrices, that is, nonnegative square matrices whose columns all sum up to 1. The symbol \mathbb{S}^n , where $n \in \mathbb{N}$, is used to mean $\mathbb{S}^{\llbracket n \rrbracket}$.

Operations on sets

- Given a set S , we denote its convex hull with the symbol $\text{co } S$. The convex hull of the union of finitely many sets S_1, \dots, S_m is denoted $\text{co}\{S_1, \dots, S_m\}$.

- Disjoint union of set is denoted with the symbol \sqcup .

Functions

- Given two functions $f : X \rightarrow Y$ and $g : Y \rightarrow Z$, we denote by $g \circ f : X \rightarrow Z$ their composition $x \mapsto g(f(x))$.
- Given a set S and a function f , the *image of S via f* is denoted as $f(S) := \{f(s) : s \in S\}$.
- Given a proposition p , we denote with $\mathbb{1}_p$ the indicator function of that proposition:

$$\mathbb{1}_p = \begin{cases} 1 & \text{if } p \text{ is true} \\ 0 & \text{otherwise.} \end{cases}$$

Partial orders

- Given a partially ordered set $(S, <)$ and two elements $s, s' \in S$, we use the standard derived symbols $s \preceq s'$ to mean that $(s = s') \vee (s < s')$, $s \succ s'$ to mean that $s' < s$, and $s \succcurlyeq s'$ to mean that $s' \preceq s$.

Asymptotic notation

- We use the symbols $\mathcal{O}, \Omega, \Theta$ to denote the usual asymptotic notation.
- The symbol \mathcal{O}_T is used to denote that only dependence on the parameter T is highlighted, treating all other parameters as constants.

Additional notation about games and other objects used throughout the dissertation will be introduced in later chapters. All notation is summarized in [Appendix B](#) to help the reader.

Chapter 2

Imperfect-information extensive-form games

Imperfect-information extensive-form^[2.a] games model tree-form strategic interactions in which not all actions might be observed by all players. They represent an ample majority of strategic interactions encountered in the real world, ranging from recreational games such as poker, to negotiation, and auctions.

The standard representation of an imperfect-information extensive-form game is through its *game tree*, which formalizes the interaction of the players as a directed tree. In the game tree, each non-terminal node belongs to exactly one player, who acts at the node by picking one of the outgoing edges (each labeled with an action name). Imperfect information is captured in this representation by partitioning the nodes of each player into sets (called *information sets*) of nodes that are indistinguishable to that player given his or her observations. We will describe in detail this representation in [Section 2.1](#).

It should be stressed that one of the distinguishing features of the game tree representation of games is that it encodes the dynamics of the interaction for *all* players, without taking the side of any one player in particular. This makes the game tree a suboptimal representation for shedding light on how *individual* players can *learn* in the game—that is, iteratively refine their strategies to improve performance until eventually, in most cases, game-theoretic equilibrium is reached. Instead, in most of this dissertation we will find it convenient (and insightful) to express learning and optimization of strategies from the point of a particular player as a procedure on the player’s *tree-form decision process (TFDP)*. As the name suggests, the TFDP expresses the interaction from the point of view of the player of interest, making a distinction between the decisions that the player faces, and the observations that the player makes. We introduce TFDPs in [Section 2.2](#), showing how the TFDP of any player can be extracted from the game tree.

^[2.a]The term *extensive-form* is a standard term in the game theory literature, meaning tree-form.

2.1 Game trees and information sets

In this section, we detail how the game tree representation captures the dynamics of a game, and how imperfect information is encoded in this representation. As a running example, we will illustrate the representation by analyzing the game tree of a simplified two-player variant of poker (perhaps the archetype of imperfect-information extensive-form games), known as *Kuhn poker* (Kuhn, 1950). Additional examples will be presented in Section 2.2.

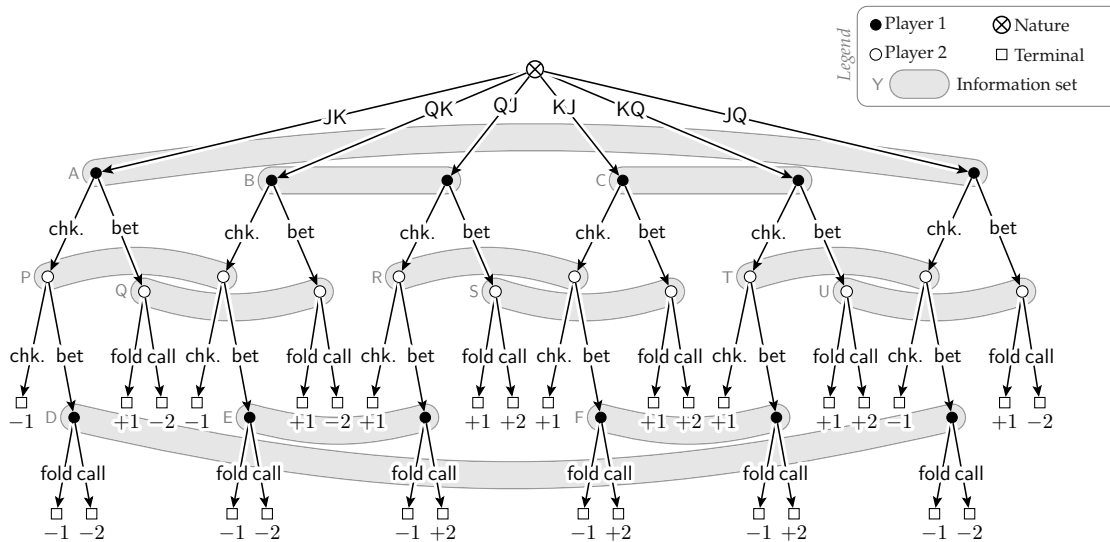


Figure 2.1: Game tree for the game of Kuhn poker. Utilities are shown for Player 1 only; utilities for Player 2 are the opposite Player 1’s. Due to space constraints, we used ‘chk.’ as an abbreviation for the ‘check’ action.

Histories, actions, and payoffs The *game tree* represents the strategic interaction of players as a finite directed tree. The nodes of the game tree are called *histories*. Each history that is not a leaf of the game tree is associated with a unique acting player. In an n -player game, the set of valid players is the set $\llbracket n \rrbracket \cup \{c\} = \{1, \dots, n, c\}$, where c denotes the *chance* (or *nature*) *player*—a fictitious player that selects actions according to a known fixed probability distribution and models exogenous stochasticity of the environment (for example, a roll of the dice, or drawing a card from a deck). The player is free to pick any one of the actions available at the history, which correspond to the outgoing edges at the histories.

The players keep acting until a leaf of the game tree—called a *terminal history*—is reached. Terminal histories are not associated with any acting player; the set of terminal histories is denoted \mathcal{Z} . When the game transitions to a terminal node $z \in \mathcal{Z}$, each player $i \in \llbracket n \rrbracket$ receives a *payoff* according to the payoff function $u_i : \mathcal{Z} \rightarrow \mathbb{R}$.

Example 2.1 (Kuhn poker). In the game tree of Kuhn poker (Figure 2.1), the root history of the tree (the first move in the game) belongs to the nature player c . It models a dealer that privately deals one card to each player from a shuffled deck containing cards {Jack, Queen, King}. The actions of the nature player correspond to the six possible assignments of two cards from the deck, which are annotated on the edges; for example, the leftmost edge JK corresponds to the case in which Player 1 is dealt a Jack and Player 2 is dealt a King. Since the deck is shuffled, each of the six actions are selected with probability $\frac{1}{6}$ by the nature player.

No matter the action selected by the dealer, the game transitions to a history of Player 1, which marks the beginning of what in poker is called a “betting round”. First, Player 1 decides to either check (continue without betting any money) or bet \$1. Then,

- If Player 1 checks, Player 2 can either check, or bet \$1.
 - If Player 2 checks, the game terminates with a *showdown*: the player with the higher card receives from the other player whatever amount the other player bet, plus an *ante* amount of \$1.
 - If, instead, Player 2 *bets* the additional \$1, then Player 1 can either fold his hand or call, that is, raise his bet by \$1.
 - * If Player 1 folds, he has to give Player 2 only the \$1 *ante*;
 - * if Player 1 calls, a showdown with the same dynamics as before.
- If Player 1 bets the \$1, Player 2 can either fold her hand or call.
 - If Player 2 folds her hand, Player 2 gives Player 1 the \$1 *ante*.
 - If, instead, Player 2 calls the bet, she increases her bet by \$1 and a showdown occurs, with the same dynamics as before.

Imperfect information and information sets To model imperfect information, the histories of each player $i \in \llbracket n \rrbracket$ are partitioned into a collection \mathcal{I}_i of so-called *information sets*. Each information set $I \in \mathcal{I}_i$ groups together histories that Player i cannot distinguish between when he or she acts there. In the limit case in which all information sets are singleton, the player never has any uncertainty about which history they are acting at, and the game is said to have *perfect information*.

Since a player always knows what actions are available at a decision node, any two histories h, h' belonging to the same information set I must have the same set of available actions. Correspondingly, we can write \mathcal{A}_I to denote the set of actions available at any node that belongs to information set I .

Example 2.1 (Continued; Kuhn poker). In Kuhn poker, each player observes their own private card and the actions of the opponent, but *not* the opponent’s private card. The twelve

information sets indicated in Figure 2.1 (six for Player 1 denoted A through F, and six for Player 2 denoted P through U) reflect this partial information.

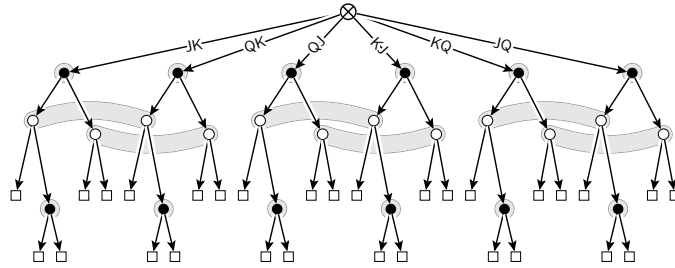
For example, Player 1's histories following actions QK and QJ of the nature player (the dealer) are part of the same information set B, in that Player 1 cannot distinguish between the two histories, having observed only their private Queen card.

As another example, Player 2's information set P captures the uncertainty the player has on the underlying history after having observed a private King card, and a check from Player 1.

Example 2.2. To further illustrate how information sets capture private information, in this example we speculate on how different rules for Kuhn poker would translate into different information set structures.

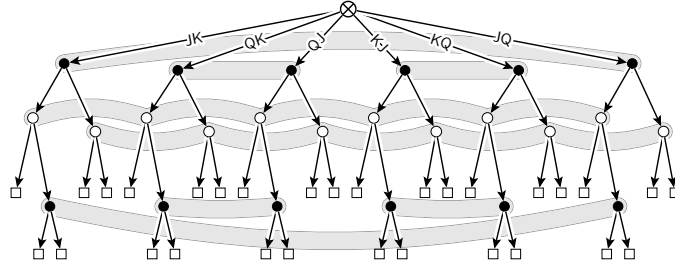
■ **First variation**

Player 1 is revealed the private card of Player 2 by the dealer.



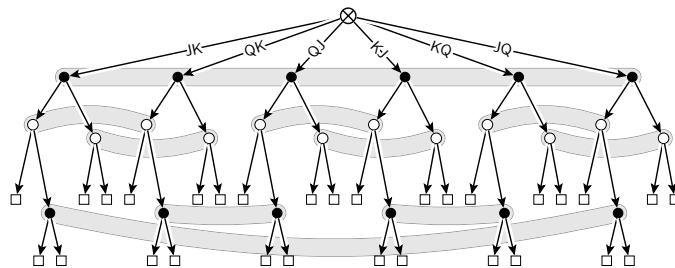
■ **Second variation**

Player 2 does not get to observe her private card.



■ **Third variation**

Player 1 is allowed to look at his private card only if he decides to check.



Perfect recall As is standard in the literature, we assume that the game has *perfect recall*, that is, information sets satisfy the fact that that no player forgets about their actions, and about information once acquired. This condition is formalized as follows.

Definition 2.1 (Perfect recall). A player $i \in \llbracket n \rrbracket$ is said to have *perfect recall* if, for any information set $I \in \mathcal{I}_i$, for any two histories $h, h' \in I$ the sequence of Player i 's actions encountered along the path from the root to h and from the root to h' must coincide (or otherwise Player i would be able to distinguish among the histories, since the player remembers all of the actions they played in the past). The game is perfect recall if all players have perfect recall.

2.2 The player's perspective: Tree-form decision processes

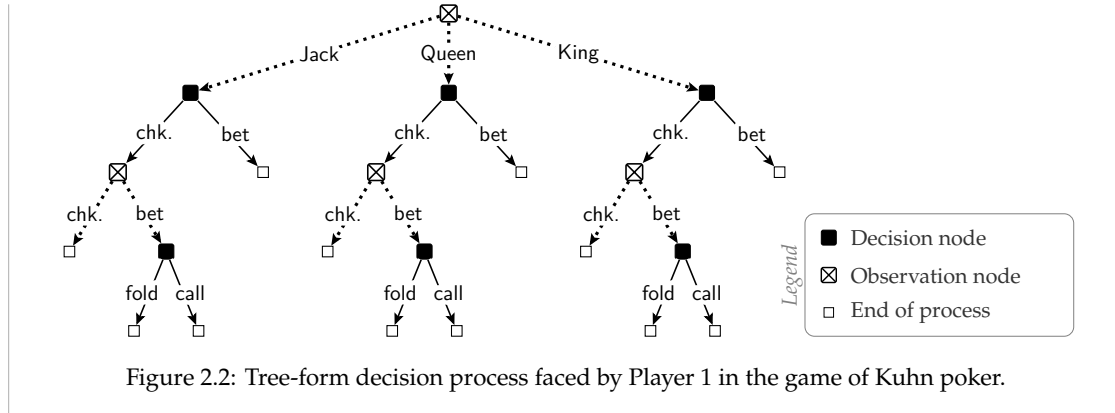
The game tree representation introduced in [Section 2.1](#) provides a description of the *global dynamics* of the game, without taking the side of any player in particular.

In this section, we will lay foundations and tools for operating with the *tree-form decision process* (TFDP) that an individual player faces in an imperfect-information extensive-form game. The TFDP provides a more natural formalism for defining player-specific quantities and procedures, such as *strategies* and *learning algorithms*, that inherently refer to the decision space that *one* player faces while playing the game. Most of the content of this dissertation, being generally concerned with how individual players can optimize or refine their strategies so that equilibrium is reached, will find its natural formalization in the language of TFDPs.

Example 2.3 (Player 1's decision process in Kuhn poker). As an example, consider Player 1 in Kuhn poker ([Example 2.1](#)). From the player's point of view, playing the game could be summarized as follows:

- As soon as the game starts, the player observes a private card that has been dealt to them; the set of possible signals is {Jack, Queen, King}.
- No matter the card observed, the player now needs to select one action from the set {check, bet}.
 - If the player bets, the player does not have a chance to act further
 - Otherwise, if the player checks, the player will then observe whether the opponent checks (at which point the interaction terminates) or bets. In the latter case, a new decision needs to be made, between folding the hand, or calling the bet. In either case, after the action has been selected, the interaction terminates.

By arranging the structure of decisions and observations along a tree as in [Figure 2.2](#), we obtain the *tree-form decision process* for Player 1.



The tree-form decision process lays out the player's opportunities to act. Unlike the game tree, in which each node belongs to one of many players, the tree-form decision process is a directed tree made of only two types of nodes: *decision nodes*, at which the player must act by picking an action from a set of legal actions, and *observation nodes*, at which the player does not act but rather observes a signal drawn from a set of possible signals. Furthermore, the information structure of the player, previously defined indirectly through information sets, is captured directly in the TFDP representation.

While the process that led us to [Figure 2.2](#) was heuristic and based on our intuitive understanding of Kuhn poker, in the next subsection we discuss how a TFDP can be extracted from a game tree in any perfect-recall imperfect-information extensive-form game.

2.2.1 Extracting a tree-form decision process from the game tree

In some cases, like in [Example 2.3](#), it is straightforward to compile the tree-form decision process faced by a player starting from our intuitive understanding of the game. In this subsection we discuss how the TFDP for the player can be constructed programmatically starting from the game tree when such an understanding is missing. We assume that an n -player imperfect-information extensive-form game with perfect recall and a player $i \in \llbracket n \rrbracket$ of interest, have been fixed.

The set of decision nodes \mathcal{J}_i of the player's TFDP coincides with the set of his or her information sets, that is, $\mathcal{J}_i = \mathcal{I}_i$. This is consistent with the fact that the player cannot condition their behavior on anything other than their information set, given that they cannot distinguish between histories in the same information set. Furthermore, the set of actions available at any decision node $j = I \in \mathcal{I}_i$ coincides with the set of actions \mathcal{A}_I available at any history in information set I .

Fix an information set I for the player, and for any history $h \in I$, imagine walking the path from the root of the game tree to h , keeping track of all the information sets and actions of Player i encountered along the path—let us call this the *trace* corresponding to history h . Because the player recalls their past actions and information sets, and yet all $h \in I$ are by definition indistinguishable to the player, it follows immediately that the traces of all histories $h \in I$ must

coincide. Hence, the notion of trace of I , defined as the trace of any $h \in I$, is well defined. We give two examples illustrating traces.

Example 2.4. Consider Kuhn poker (Figure 2.1) from the point of view of Player 1.

- The trace of any history in A is the sequence (A) .
- The trace of any history in E is the sequence $(B, \text{chk.}, E)$.

From the point of view of Player 2, the trace of any history in R is the sequence (R) .

Example 2.5. Consider the small game tree given in Figure 2.3.

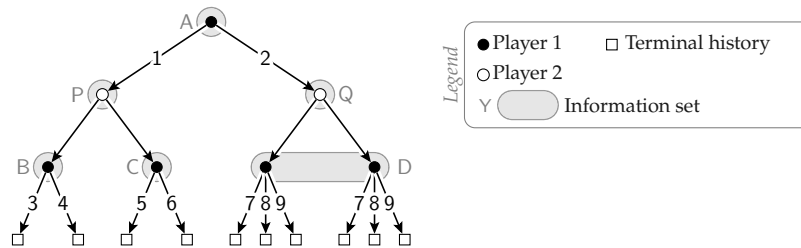


Figure 2.3: Small game tree used in the example.

Taking the side of Player 1, the trace of the only history in B is the sequence $(A, 1, B)$, the trace of any history in D is $(A, 2, D)$, and the trace of the only history in A is (A) . Taking the side of Player 2, the trace of the only history in P is (P) , and the trace of the only history in Q is (Q) .

Traces implicitly encode a notion of partial chronological ordering between information sets, of which the player has recall—see Definition 2.1. Hence, for the TFDP of Player i to be an accurate representation of the decision process the player faces while playing the game, it is necessary that *traces of the information sets are the same in the game tree and in the TFDP*. In other words, we require that decision points in the TFDP be structured so as to satisfy that the trace of any information set I matches the sequence of information sets and actions encountered from the root of the TFDP to decision node I .

Definition 2.2 (Tree-form decision process). Fix the game tree of an n -player imperfect information game, and a player $i \in \llbracket n \rrbracket$. A *tree-form decision process (TFDP)* for Player i is a directed rooted tree made of *decision, observation, and terminal nodes*, satisfying the following properties.

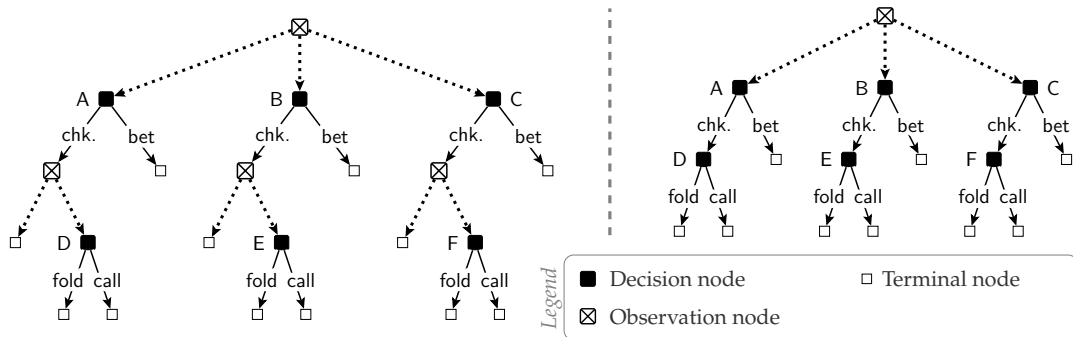
- The set of decision nodes \mathcal{J}_i of the TFDP is equal to the set \mathcal{I}_i of information sets.
- The set of actions available at each decision node $j = I \in \mathcal{I}_i$ (i.e., the set of outgoing

edges from the decision node) is equal to the set of actions \mathcal{A}_I available at any history $h \in I$ in the game tree.

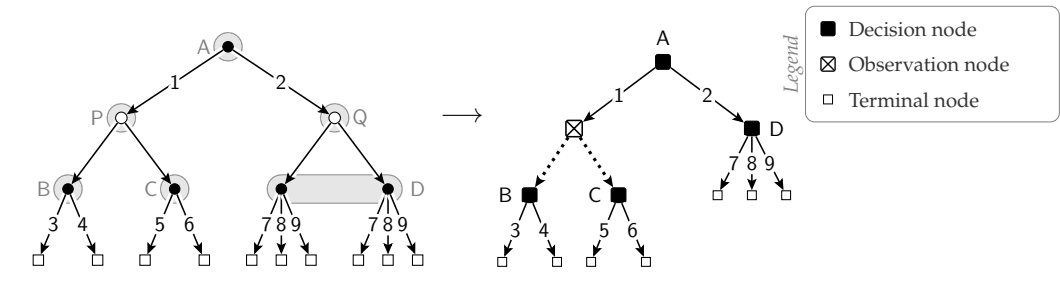
- Given any decision node $j = I \in \mathcal{I}_i$, the sequence of decision nodes and actions encountered from the root of the TFDP to j is equal to the trace of any history $h \in I$.

We remark that [Definition 2.2](#) leaves the labeling and structure of observation nodes unspecified. In fact, a player might have multiple TFDPs that satisfy the definition, and differ in how the observation nodes are placed. We illustrate this in the next example.

Example 2.6. The following are both valid TFDPs capturing Player 1's decision process when playing Kuhn poker.



Example 2.7. A valid TFDP representing the decision process of Player 1 in the small game of [Figure 2.3](#), reproduced below on the left, is shown below on the right.



2.2.2 Notation

Most of the algorithms and procedures we will introduce in this dissertation are best understood as operating on the tree-form decision process of a player. In this section we introduce additional notation and conventions related to such objects, which will be used throughout the document. A summary of the notation is given in [Table 2.1](#) and in [Appendix B](#).

Convention regarding players In this dissertation we use the general convention of subscripting quantities relative to a Player i with a lowercase i whenever important. This is the case, for example, of the information sets of the players, which we have denoted $\mathcal{I}_1, \dots, \mathcal{I}_n$ in the past. However, as most of the algorithms we will discuss in this dissertation assume that a TFDP for a generic Player i has been fixed, it will be typical to omit the lowercase i from most of the discussion. The notation summarized in Table 2.1 has the player omitted.

Decision and observation nodes, transition function:

- We denote the set of decision nodes in the TFDP as \mathcal{J} , and the set of observation nodes as \mathcal{K} . At each decision node $j \in \mathcal{J}$, the player selects an action from the set \mathcal{A}_j of available actions. At each observation node $k \in \mathcal{K}$, the player observes a signal s_k from the environment out of a set of possible signals \mathcal{S}_k .
- We denote ρ the transition function of the process. Picking action $a \in \mathcal{A}_j$ at decision node $j \in \mathcal{J}$ results in the process transitioning to $\rho(j, a) \in \mathcal{J} \cup \mathcal{K} \cup \{\perp\}$, where \perp denotes the end of the decision process. Similarly, the process transitions to $\rho(k, s) \in \mathcal{J} \cup \mathcal{K} \cup \{\perp\}$ after the player observes signal $s \in \mathcal{S}_k$ at observation node $k \in \mathcal{K}$.

Sequences, which identify actions at decision nodes, and will be key in defining *sequence-form strategies* in the next section:

- A pair (j, a) where $j \in \mathcal{J}$ and $a \in \mathcal{A}_j$ is called a *non-empty sequence*. The set of all non-empty sequences is denoted as $\Sigma^* := \{(j, a) : j \in \mathcal{J}, a \in \mathcal{A}_j\}$. For notational convenience, we will often denote an element (j, a) in Σ as ja without using parentheses, especially when used as a subscript.
- The symbol \emptyset denotes a special sequence called the *empty sequence*. The set of all sequences, including the empty one, is denoted Σ .
- Given a decision node $j \in \mathcal{J}$, we denote by p_j its *parent sequence*, defined as the last sequence (that is, decision point-action pair) encountered on the path from the root of the decision process to j . If the player does not act before j (that is, j is the root of the process or only observation nodes are encountered on the path from the root to j), we let $p_j = \emptyset$.
- Given a sequence $\sigma \in \Sigma$, we denote with \mathcal{C}_σ the set of decision nodes j whose parent sequence is σ : $\mathcal{C}_\sigma := \{j \in \mathcal{J} : p_j = \sigma\}$.

Subtrees and descendency relationships Finally, we introduce the following symbols to establish descendency relationships:

- Given two decision nodes j, j' , we write $j \prec j'$ (or equivalently $j' \succ j$) to mean that j is an ancestor of j' in the TFDP and that $j \neq j'$. The symbol $j \preceq j'$ (or equivalently $j' \succeq j$) means that $j \prec j'$ or $j = j'$.
- Given two sequences $ja, j'a'$, we write $ja \preceq j'a'$ (or equivalently $j'a' \succeq ja$) to mean that (unique) path from the root to action a' at j' passes through action a at j .

- The overloaded notation $\sigma \succcurlyeq j$ (or equivalently $j \preccurlyeq \sigma$), defined for any $j \in \mathcal{J}$, and sequence $\sigma = (j', a') \in \Sigma^*$, denotes that $j' \succcurlyeq j$.
- Finally, we let $\Sigma_j := \{\sigma \in \Sigma^* : \sigma \succcurlyeq j\} \subseteq \Sigma^*$ be the set of sequences at or below a given $j \in \mathcal{J}$.

Symbol	Description
\mathcal{J}	Set of decision nodes
\mathcal{A}_j	Set of legal actions at decision node $j \in \mathcal{J}$
\mathcal{K}	Set of observation nodes
\mathcal{S}_k	Set of possible signals at observation node $k \in \mathcal{K}$
ρ	Transition function: <ul style="list-style-type: none"> • given $j \in \mathcal{J}$ and $a \in \mathcal{A}_j$, $\rho(j, a)$ returns the next point $v \in \mathcal{J} \cup \mathcal{K}$ in the decision tree that is reached after selecting legal action a in j, or \perp if the decision process ends; • given $k \in \mathcal{K}$ and $s \in \mathcal{S}_k$, $\rho(k, s)$ returns the next point $v \in \mathcal{J} \cup \mathcal{K}$ in the decision tree that is reached after observing signal s in k, or \perp if the decision process ends
Σ^*	Set of non-empty sequences, defined as $\Sigma^* := \{(j, a) : j \in \mathcal{J}, a \in \mathcal{A}_j\}$
Σ	Set of sequences, defined as $\Sigma := \Sigma^* \cup \{\emptyset\}$ where the special element \emptyset is called the <i>empty sequence</i>
p_j	Parent sequence of decision node $j \in \mathcal{J}$, defined as the last sequence (decision node-action pair) on the path from the root of the TFDP to decision node j ; if the player does not act before j , $p_j = \emptyset$
\mathcal{C}_σ	Decision nodes $j \in \mathcal{J}$ with parent sequence $\sigma \in \Sigma$: $\mathcal{C}_\sigma := \{j \in \mathcal{J} : p_j = \sigma\}$
$j' \preccurlyeq j$	$j' \in \mathcal{J}$ is on the path from the root to $j \in \mathcal{J}$
$j'a' \preccurlyeq ja$	Action a' at j' is on the path from the root to action a at $j \in \mathcal{J}$
$\sigma \succcurlyeq j$	Shorthand for $\sigma = j'a'$ with $j' \succcurlyeq j$
$\Sigma_{\succcurlyeq j}$	Sequences at or below j : $\Sigma_{\succcurlyeq j} := \{\sigma \in \Sigma^* : \sigma \succcurlyeq j\}$

Table 2.1: Summary of basic notation for TFDPs. In cases where it is important to specify the player to which the different quantities belong, a subscript with the player will be added.

2.3 Strategies and sequence-form representation

Consider the tree-form decision process faced by a player in an imperfect-information extensive-form game. Conceptually, a strategy for a player corresponds to a choice of distribution over the set of actions \mathcal{A}_j at each decision node $j \in \mathcal{J}$. So, perhaps the most intuitive representation of a strategy, called a *behavioral strategy*, is as a vector $\mathbf{x} \in \mathbb{R}_{\geq 0}^\Sigma$ indexed over sequences assigning to each action a at decision node j the probability of picking that action at that decision node. The set of all possible behavioral strategies is clearly convex, as it is the Cartesian product of probability

simplexes—one per each decision node. However, that representation has a major drawback: the probability of reaching a particular terminal state in the decision process is the product of all actions on the path from the root to the terminal state. This makes many expressions of interest that depend on the probability of reaching terminal states (including crucially the expected utility in the game) non-convex.

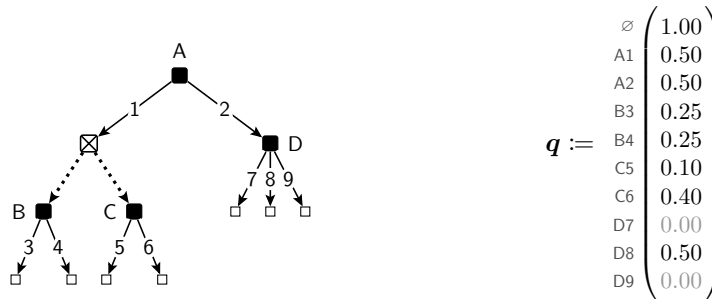
2.3.1 Sequence-form representation of strategies

The *sequence-form representation* (Romanovskii, 1962; Koller, Megiddo, and von Stengel, 1996; von Stengel, 1996) soundly resolves the issue of non-convexity. In the sequence-form representation, a strategy is a vector $\mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma}$ whose entries are indexed by Σ . However, the entry $x[ja]$ contains the *product* of the probabilities of all actions at all decision nodes on the path from the root of the process to action a at decision node j . In order to be a valid sequence-form strategy, the entries in \mathbf{x} must therefore satisfy the following probability-flow-conservation constraints:

$$\mathbf{x}[\emptyset] = 1, \quad \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] = \mathbf{x}[p_j] \quad \forall j \in \mathcal{J}. \quad (2.1)$$

Conversely, it is easy to see that any \mathbf{x} that satisfies the above constraints is the sequence-form representation of at least one behavioral strategy.

Example 2.8. Consider the TFDP introduced in Example 2.7, reproduced below.



Let us consider the mixed sequence-form strategy $\mathbf{q} \in \mathcal{Q}$ defined alongside the TFDP above. We have that $q[A1] = q[A2] = 0.5$, and therefore Player 1 will select between actions 1 and 2 at decision node A uniformly at random.

Suppose Player 1 selects action 1. If Player 1 reached decision node B, she would select actions 3 and 4 with probability $0.25/0.5 = 0.5$ each. On the other hand, if Player 1 reached decision node C, she would choose action 5 with probability $0.1/0.5 = 0.2$, and action 6 with probability $0.4/0.5 = 0.8$.

Analogously, if Player 1 played action 2 at decision node A, upon reaching decision node D she would play action 8 with probability $0.5/0.5 = 1$.

In general, the probability of playing action a at a generic decision node j can be obtained by dividing $q[ja]$ by $q[p_j]$.

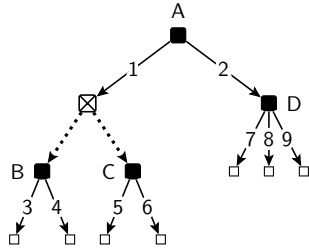
The set of all sequence-form strategies will be denoted with the symbol \mathcal{Q} . Since (2.1) defines linear constraints, \mathcal{Q} is a convex polytope.

Definition 2.3. The *polytope of sequence-form strategies* of a TFDP is the convex polytope

$$\mathcal{Q} := \left\{ \mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma} : \mathbf{x}[\emptyset] = 1, \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] = \mathbf{x}[p_j] \quad \forall j \in \mathcal{J} \right\}.$$

Example 2.9. Consider the tree-form decision process faced by Player 1 in the small game of Example 2.5, which is reproduced in Figure 2.4 (Left).

The decision process has four decision nodes $\mathcal{J} = \{A, B, C, D\}$ and nine sequences including the empty sequence \emptyset . For decision node D, the parent sequence is $p_D = A2$; for B and C it is $p_B = p_C = A1$; for A it is the empty sequence $p_A = \emptyset$. The constraints that define the sequence-form polytope (Definition 2.3) are shown in Figure 2.4 (Right).



Sequence-form constraints:

$$\left\{ \begin{array}{l} \mathbf{x}[\emptyset] = 1, \\ \mathbf{x}[A1] + \mathbf{x}[A2] = \mathbf{x}[\emptyset], \\ \mathbf{x}[B3] + \mathbf{x}[B4] = \mathbf{x}[A1], \\ \mathbf{x}[C5] + \mathbf{x}[C6] = \mathbf{x}[A1], \\ \mathbf{x}[D7] + \mathbf{x}[D8] + \mathbf{x}[D9] = \mathbf{x}[A2]. \end{array} \right.$$

Figure 2.4: (Left) Tree-form decision process considered in the example. (Right) The constraints that define the sequence-form polytope \mathcal{Q} for Player 1 (besides nonnegativity) in the TFDP shown on the left.

This dissertation will almost exclusively work with strategies represented in sequence form.

The polytope of sequence-form strategies possesses a strong combinatorial structure that enables speeding up several common optimization procedures and will be crucial in developing efficient algorithms to converge to equilibrium.

Sometimes, we will find it important to consider partial strategies that only specify behavior at a decision node j and all of its descendants $j' \succ j$. We make that formal through the following definition.

Definition 2.4. The set of sequence-form strategies for the subtree^[2.b] rooted at j , denoted $\mathcal{Q}_{\succ j}$, is the set of all vectors $\mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma_{\succ j}}$ such that probability-mass-conservation constraints hold at decision node j and all of its descendants $j' \succ j$, specifically

$$\mathcal{Q}_{\succ j} := \left\{ \mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma_{\succ j}} : \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] = 1, \quad \text{and} \quad \sum_{a \in \mathcal{A}_{j'}} \mathbf{x}[j'a] = \mathbf{x}[pj'] \quad \forall j' \succ j \right\}. \quad (2.2)$$

2.3.2 Deterministic sequence-form strategies and Kuhn's theorem

Out of the polytope of all possible strategies in the decision process, *deterministic* strategies are extremely important. Deterministic strategies are strategies that select exactly one action at each decision node, without ever randomizing the choice. In other words, deterministic strategies assign probability either 0 or 1 to each action at each decision node. Hence, when using the sequence-form representation, the set of all deterministic strategies—denoted Π —corresponds to the subset of \mathcal{Q} whose components are 0 or 1.

Definition 2.5. The set of *deterministic* sequence-form strategies is the set

$$\Pi := \mathcal{Q} \cap \{0, 1\}^{\Sigma}.$$

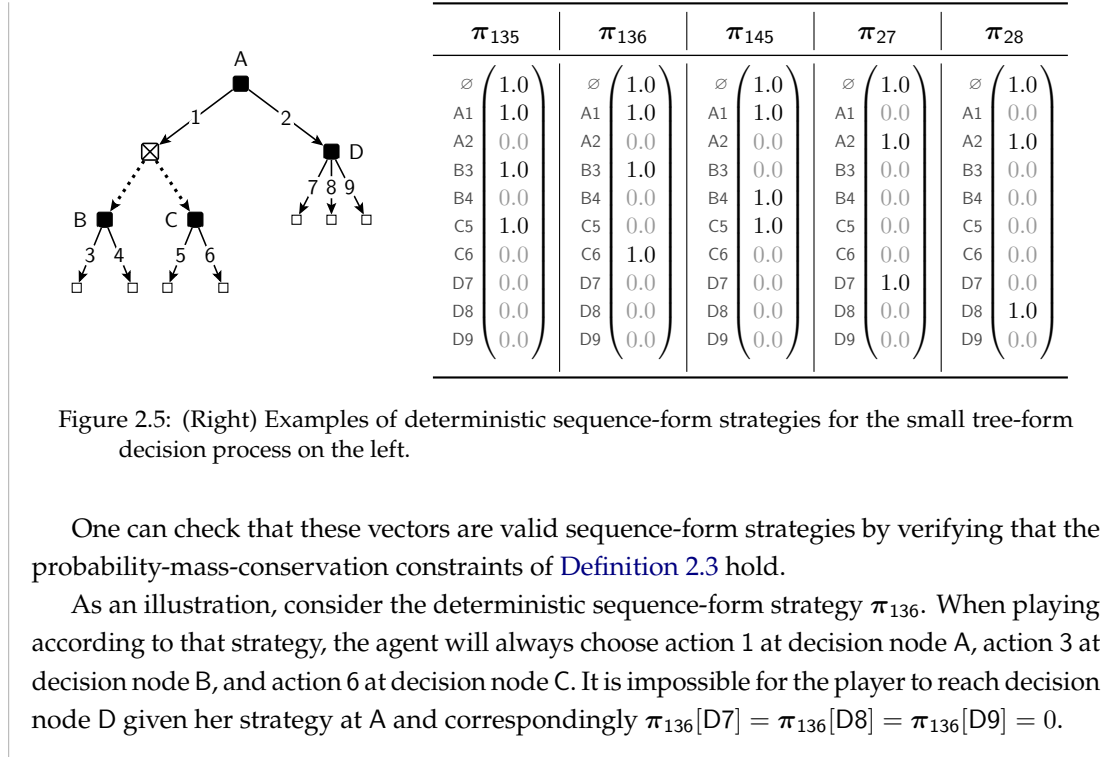
Similarly, the set of *deterministic sequence-form strategies for the subtree rooted at j* is

$$\Pi_{\succ j} := \mathcal{Q}_{\succ j} \cap \{0, 1\}^{\Sigma_{\succ j}}.$$

We provide examples of deterministic sequence-form strategies next.

Example 2.10 (Deterministic sequence-form strategies). Continuing [Example 2.7](#), in [Figure 2.5](#) (Right) we provide five deterministic sequence-form strategies $\pi_{135}, \pi_{136}, \pi_{145}, \pi_{27}, \pi_{28} \in \Pi$.

^[2.b]The term “subtree” does not refer to a subtree of the game tree, but rather to a subtree of the partially ordered set (\mathcal{J}, \prec) . In other words, the term *subtree* here refers to the fact that the quantities are specified only at decision node j and all of its descendants.



One can check that these vectors are valid sequence-form strategies by verifying that the probability-mass-conservation constraints of [Definition 2.3](#) hold.

As an illustration, consider the deterministic sequence-form strategy π_{136} . When playing according to that strategy, the agent will always choose action 1 at decision node A, action 3 at decision node B, and action 6 at decision node C. It is impossible for the player to reach decision node D given her strategy at A and correspondingly $\pi_{136}[D7] = \pi_{136}[D8] = \pi_{136}[D9] = 0$.

Kuhn's theorem (Kuhn, 1953) establishes the connection between the sequence-form strategy polytope and the set of deterministic sequence-form strategies.

Theorem 2.1 (Kuhn's theorem). The sequence-form polytope is the convex hull of the set of deterministic sequence-form strategies, that is,

$$\mathcal{Q} = \text{co } \Pi, \quad \text{and} \quad \mathcal{Q}_{\neq j} = \text{co } \Pi_{\neq j} \quad \forall j \in \mathcal{J}.$$

Chapter 3

No-regret learning in games

One of the key focuses of this dissertation is providing positive, constructive answers to the following fundamental question:

Can a player that repeatedly plays an imperfect-information extensive-form game follow rules to refine their strategy after each match, so as to guarantee mastering the game in the long run?

Throughout the different chapters, we will investigate different angles of the question: How does one define “mastering” the game? What kind of feedback is collected by each player after each match? How fast can the agent master the game?

To attach a quantitative measure to the goal of learning, we will operate within the framework of *regret minimization*. Regret minimization is founded on the idea that learning players should keep under control (specifically, keep as low as possible) their *regret*—the difference between the reward they accumulated through the actions they played, and the reward they would have accumulated in hindsight had they consistently modified their behavior according to some strategy transformation function. The size of the set of transformation functions considered by each learning player determines a natural notion of rationality of that player.

An extremely important feature of regret minimization is that it connects a *local* notion of optimality, the minimization of the regret incurred by each learning player, to a *global* notion of optimality, convergence to game-theoretic equilibrium. This connection informs a methodology for computing a variety of equilibria that is core to this dissertation. As of today, algorithms based on learning dynamics are by far the most scalable techniques for computing many notions of game-theoretic equilibrium in large games, and have played a central role in most recent game-playing AI breakthroughs. Compared to other traditional optimization techniques for equilibrium computation, which tend to define a more centralized notion of update, learning dynamics have the advantage that the optimization is decentralized (each player updates their own strategy incrementally), and uncoupled (each player only uses feedback about their own

utility function).

3.1 Hindsight rationality and Φ -regret

No-regret learning algorithms make the objective of “learning to play the game” concrete by means of the quantitative metric called *regret*. Intuitively, as the name suggests, regret tracks the difference between the utility that the algorithm accumulated over time, and the utility that it could have cumulated in hindsight by employing a different strategy. More specifically, given a generic player whose set of strategies is \mathcal{X} , we consider a necessary condition for saying that the player has “learnt” to play the game when looking back at the history of play, the player cannot think of any transformation $\phi : \mathcal{X} \rightarrow \mathcal{X}$ of their strategies that when applied at the whole history of play would have given strictly better utility to the player. The size of the set Φ of transformations $\phi : \mathcal{X} \rightarrow \mathcal{X}$ considered by the player defines a natural notion of how “*hindsight-rational*” the agent is. As we will see in [Section 3.1.2](#), higher hindsight rationality is tied to the ability to recover more sophisticated game-theoretic notion from no-regret strategies.

3.1.1 Definition of Φ -regret for a player in the repeated game

In order to formally define Φ -regret, consider a generic n -player imperfect-information extensive-form game, and let u_1, \dots, u_n and $\mathcal{Q}_1, \dots, \mathcal{Q}_n$ denote the expected utility functions and sequence-form strategy spaces of the players, respectively. Furthermore, let Φ be a compact set of functions $\phi : \mathcal{Q}_i \rightarrow \mathcal{Q}_i$ for a generic player i in the game. Suppose that at all times (repetition of the game) $t = 1, 2, \dots$, Player i plays according to some strategy $\mathbf{x}_i^{(t)}$, while all other players play according to strategies $(\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_{i-1}^{(t)}, \mathbf{x}_{i+1}^{(t)}, \dots, \mathbf{x}_n^{(t)}) =: \mathbf{x}_{-i}^{(t)}$. Then, for any time horizon T , the Φ -regret of Player i up to time T is defined as the difference between what Player i could have accumulated in hindsight had they consistently modified their strategy according to the best $\phi \in \Phi$, and the expected utility that they actually accumulated; in symbols

$$\Phi\text{-Reg}_i(\{\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}\}_{t=1}^T) := \max_{\phi \in \Phi} \left\{ \sum_{t=1}^T u_i(\hat{\phi}(\mathbf{x}_i^{(t)}), \mathbf{x}_{-i}^{(t)}) - u_i(\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}) \right\}.$$

It is clear from the definition that regret can grow at most linearly as a function of the time horizon T . When the player plays in a way that regret is guaranteed to grow sublinearly in T , we say that the player is a *no- Φ -regret learner*. As we show in the next subsection, sublinear growth of regret is enough to guarantee ergodic convergence to game-theoretic equilibrium.

3.1.2 Relationship between Φ -regret and game-theoretic equilibrium

An important feature of the framework of Φ -regret minimization is its ability to connect the *individual* notion of Φ -regret with *global* notions of game-theoretic optimality. The specific notion of game-theoretic equilibrium that can be recovered from no- Φ -regret learning players depends on the class of deviations Φ with respect to which the agents seek to minimize regret. As the set Φ grows, so does intuitively the rationality of the agents, and more sophisticated notions of equilibrium are approached. Below, we mention without proof a few landmark results showing that some of the best-studied notions of equilibrium can indeed be reached via learning dynamics.

Constant transformations (External regret) In this case, we are only requiring that the player does not regret substituting *all* of the strategies they played with a *unique, constant* strategy \hat{x} in hindsight. Despite the seemingly restrictive notion of rationality, regret minimization with respect to constant transformation is extremely powerful.

Connection 3.1 (D. Foster and Vohra, 1997). When all players in a multiplayer game play so that their Φ -regret with respect to the of constant transformations grows sublinearly in the number of repetitions of the game T , their average joint strategy^[3.a] converges to the set of *coarse correlated equilibria* of the game as $T \rightarrow \infty$.

Connection 3.2 (Folklore). When the two players in a *two-player zero-sum* game play so that their Φ -regret with respect to the set of constant transformations grows sublinearly in the number of repetitions of the game T , the average strategies $(\bar{x}_1^{(T)}, \bar{x}_2^{(T)}) := (\frac{1}{T} \sum_{t=1}^T x_1^{(t)}, \frac{1}{T} \sum_{t=1}^T x_2^{(t)})$ converge to the set of *Nash equilibria* of the game as $T \rightarrow \infty$.

Even more, we will show in [Section 3.2.3](#) that the minimization of regret with respect to constant transformations forms the backbone of learning with respect to any richer set of deviations. Φ -regret measured with respect to constant transformations is often called *external regret* or *static regret* in the literature.

Mass-transferring deviations (Internal regret) These deviations apply only to nonsequential games, where each player's strategy space is the probability simplex over their finite set of actions \mathcal{A} . In this context, the set of mass-transferring deviations is the set of transformation functions that intuitively transfer the probability mass given to an action a to another action b ; formally, $\Phi := \{\phi_{a \rightarrow b} : a, b \in \mathcal{A}\}$, where, for all $k \in \mathcal{A}$,

^[3.a]That is, at all times T , the correlated distribution of play $\mu^{(T)}$ that selects $(x_1^{(t)}, \dots, x_n^{(t)})$ with probability $1/T$.

$$\phi_{a \rightarrow b}(\mathbf{x})[k] := \begin{cases} \mathbf{x}[k] & \text{if } k \notin \{a, b\} \\ 0 & \text{if } k = a \\ \mathbf{x}[a] + \mathbf{x}[b] & \text{if } k = b. \end{cases}$$

Φ -regret measured with respect to mass-transferring transformations is often called *internal regret* in the literature. It can be easily shown that an internal-regret-minimizing agent is automatically external-regret-minimizing; so, [Connections 3.1](#) and [3.2](#) apply to internal-regret-minimizing agents too. In addition, internal-regret-minimizing agents recover correlated equilibria of the game, as the following celebrated result states.

Connection 3.3 (Hart and Mas-Colell, 2000). When all players in a multiplayer *nonsequential* game play so that their internal regret grows sublinearly as a function of the number of repetitions of the game T , their average joint strategy^[3.a] converges to the set of *correlated equilibria* of the game.

Trigger deviation functions (Trigger regret) In [Chapter 8](#) of this dissertation we will introduce a generalization of the mass-transferring deviations mentioned above, called *trigger deviation functions*. These deviation functions coincide with mass-transferring deviations in nonsequential games, and are well-defined in any extensive-form game. In this case, we will show how one can construct learning players that are able to update their strategies efficiently while guaranteeing that their Φ -regret bounded with a polynomial dependence on the size of the game tree. Most importantly, we will show that trigger deviation functions are related to solution concept called “extensive-form correlated equilibrium”, introduced by von Stengel and Forges (2008).

Connection 3.4 ([Chapter 8](#)). When all players in a multiplayer extensive-form game play so that their Φ -regret relative to the set of all trigger deviation functions grows sublinearly in the number of repetitions of the game T , their average joint strategy^[3.a] converges to the set of *extensive-form correlated equilibria* of the game.

Φ -regret measured with respect to trigger deviation functions takes the name of *trigger regret*.

3.1.3 Feedback available to the learning player

The previous subsections justify keeping regret as small as possible (and, in particular, at most sublinear) as valuable goal with important connections to the computation of game-theoretic equilibrium. In order for the player to learn and satisfy the sublinear regret requirement, it is

clearly important to define what kind of input (that is, feedback) the learning player has at his or her own disposal. In this dissertation we will consider three models of feedback that each learning player might use to progressively refine their strategies.

Gradient feedback In this model, after the t -th match (repetition of the game) is completed, the learning player receives as input a vector, which represents the gradient $\nabla_{\mathbf{x}_i} u_i(\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)})$ of the player's expected utility evaluated in the current strategy of each player.

Trajectory bandit feedback In this feedback model, it is assumed that the game is simulated by having the players sample actions according to their current strategies. The learning player observes the realized trajectory of play (the sequence of actions and observations from the root of the player's tree-form decision problem down to a terminal node), as well as the realized utility.

Bandit optimization feedback In this feedback model, the learning player only observes the realized utility, but not the realized trajectory of play. One way to think about this feedback model is that at the beginning of each match, each player writes down their strategy, and sends it to a third party. The third party simulates the game on behalf of the players, and informs every player of their expected utility. It might come as a surprise to the reader that even from this extremely limited feedback, learning and convergence to equilibrium are possible.

The majority of this dissertation will consider learning algorithms that operate with gradient feedback, for the important reason that learning under trajectory bandit and bandit optimization feedback can be reduced to it. Specifically, it is known that the feedback of the latter two models can be used to construct an *unbiased estimator* of the gradient of the utility function, making it feasible to retrofit a learning algorithm designed for gradient feedback into one for the trajectory bandit or bandit optimization feedback.

3.2 Mathematical abstraction of a predictive no- Φ -regret algorithm

We have seen in [Section 3.1.1](#) that a desirable goal for a generic player i in a repeated game is to ensure that the Φ -regret

$$\Phi\text{-Reg}_i\left(\{\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}\}_{t=1}^T\right) := \max_{\hat{\phi} \in \Phi} \left\{ \sum_{t=1}^T u_i\left(\hat{\phi}(\mathbf{x}_i^{(t)}), \mathbf{x}_{-i}^{(t)}\right) - u_i\left(\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}\right) \right\} \quad (3.1)$$

grows sublinearly as a function of the number of repetitions of the game T , no matter the strategies $\mathbf{x}_{-i}^{(t)}$ of the opponents. By construction of the sequence form, each player's utility is linear the

player's strategy, and in particular

$$u_i(\cdot, \mathbf{x}_{-i}^{(t)}) = \langle \nabla_{\mathbf{x}_i} u_i(\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_n^{(t)}), \cdot \rangle.$$

Hence, we can rewrite (3.1) to express the goal of Player i as ensuring that the quantity

$$\Phi\text{-Reg}_i(\{\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}\}_{t=1}^T) := \max_{\hat{\phi} \in \Phi} \left\{ \sum_{t=1}^T \langle \nabla_{\mathbf{x}_i} u_i(\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_n^{(t)}), \hat{\phi}(\mathbf{x}_i^{(t)}) - \mathbf{x}_i^{(t)} \rangle \right\}, \quad (3.2)$$

grows sublinearly in the number of repetitions T , no matter the strategies $\mathbf{x}_{-i}^{(t)}$ played by the opponents. The mathematical abstraction of a *no- Φ -regret algorithm* generalizes (3.2), by asking that

$$\Phi\text{-Reg}_i(\{\mathbf{x}_i^{(t)}, \mathbf{u}_i^{(t)}\}_{t=1}^T) := \max_{\hat{\phi} \in \Phi} \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\phi}(\mathbf{x}_i^{(t)}) - \mathbf{x}_i^{(t)} \rangle,$$

no matter the sequence of (potentially adversarially chosen) *linear utility vectors* $\mathbf{u}^{(t)}$, each of which is assumed to be drawn from some bounded set. Moving away from the concrete choice of linear utility vector $\nabla_{\mathbf{x}_i} u_i(\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_n^{(t)})$ to a generic $\mathbf{u}^{(t)}$ without assumptions other than bounded norm comes with advantages and disadvantages.

- Decoupling $\mathbf{u}^{(t)}$ from the strategies of the opponents, and requiring that the no- Φ -regret algorithm for Player i be able to guarantee sublinear Φ -regret no matter the sequence of $\mathbf{u}^{(t)}$, removes Player i 's opponents from the picture. In other words, learning defined in this abstracted way is properly a per-player endeavor.
- Another advantage is in that the generality afforded by the $\mathbf{u}^{(t)}$, which are now not constrained to be in the form $\mathbf{u}^{(t)} = \nabla_{\mathbf{x}_i} u_i(\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_n^{(t)})$, enables the use of no- Φ -regret algorithms for tasks other than learning in games. In this sense, one can think of a no- Φ -regret algorithm as an online optimization algorithm (Zinkevich, Bowling, Johanson, and Piccione, 2007).
- On the flip side, the generality comes at the cost of losing the ability to exploit any special structure enjoyed by utility vectors of the form $\mathbf{u}^{(t)} = \nabla_{\mathbf{x}_i} u_i(\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_n^{(t)})$, as the no- Φ -regret algorithm has in general no assumption about the nature of the utility vectors it could receive next.

The last point is not just of theoretical interest. When no assumptions are available about the utility vectors $\mathbf{u}^{(t)}$ (other than bounded norm), a lower bound of $\Omega_T(\sqrt{T})$ on the regret is known (Shalev-Shwartz, 2012). Yet, the utilities $\mathbf{u}^{(t)} = \nabla_{\mathbf{x}_i} u_i(\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_n^{(t)})$ that arise while learning games are structured and often have nice properties—for example changing slowly over time. It is then natural to wonder what improved guarantees can be achieved in games specifically,

and consequently how fast convergence to equilibrium can be guaranteed. This fundamental question was first formulated and addressed by Daskalakis, Deckelbaum, and Kim (2011) within the context of *zero-sum games*. Since then, there has been a considerable interest in extending their guarantee to more general settings (S. Rakhlin and Sridharan, 2013; Syrgkanis, Agarwal, Luo, and Schapire, 2015; D. J. Foster, Li, Lykouris, Sridharan, and Tardos, 2016; Chen and Peng, 2020; Daskalakis and Golowich, 2022; Piliouras, Sim, and Skoulakis, 2022). In particular, Daskalakis, Fishelson, and Golowich (2021) established that when all players in a general *nonsequential* games employ a variant of *multiplicative weights update* (MWU) (see Section 3.3.1), the regret of each player grows *nearly-optimally* as $\mathcal{O}_T(\log^4 T)$ after T repetitions of the game, leading to an *exponential improvement* over the guarantees obtained using traditional techniques within the no-regret framework. In Chapters 6 and 7 of this dissertation, we will look into this question, extending polylogarithmic guarantees to the significantly more challenging setting of imperfect-information extensive-form games, and further improving on the bounds known in the literature, establishing $\mathcal{O}_T(\log T)$ guarantees.

Predictions The preceding discussion about the advantages and disadvantages of abstracting learning in games via the framework of no- Φ -regret algorithms begs the natural question as to what is the best compromise, or middle ground, between the appealing decoupled formulation and the ability to not lose track of the positive properties of game-induced utilities. In this thesis we adopt the middle ground of *learning with predictions*, pioneered in the works of Chiang, Yang, C.-J. Lee, Mahdavi, C.-J. Lu, R. Jin, and Zhu (2012) and S. Rakhlin and Sridharan (2013).

The framework of learning with predictions does not introduce any assumption on the utility vectors $u^{(t)}$, but rather introduces the possibility that the no- Φ -regret algorithm is given a prediction of the next utility function before the strategy is output. In this model the prediction can either be received by an oracle as a generic input to the algorithm, or more typically it is assumed that it is constructed by the learning algorithm itself. In the latter case, a standard choice is to use as prediction the last-received utility vector, as we will recall in Section 3.2.1. In particular, this latter choice will be used in Chapters 6 and 7 and will lead to $\tilde{\mathcal{O}}_T(1/T)$ convergence to equilibria. We formalize this choice in what we call the *canonical optimistic learning setup* (COLS) in Section 3.2.1.

Definition of no- Φ -regret algorithm We are now ready to formalize in the concept of a *no- Φ -regret algorithm* in a definition.

Definition 3.1 (No- Φ -regret algorithm). Given a convex set \mathcal{X} and a set Φ of linear transformations $\phi : \mathcal{X} \rightarrow \mathcal{X}$, a *No- Φ -regret algorithm for the set \mathcal{X}* is a model for a decision maker that repeatedly interacts with a black-box environment. At each time t , the algorithm interacts with the environment through two operations:

- $\text{NextStrategy}(\mathbf{m}^{(t)})$ informs the algorithm of the prediction vector $\mathbf{m}^{(t)}$. The algorithm will output the next strategy $\mathbf{x}^{(t)} \in \mathcal{X}$;
- $\text{ObserveUtility}(\mathbf{u}^{(t)})$ provides the environment's feedback to the no-regret algorithm, in the form of a linear utility function $\mathcal{X} \ni \mathbf{x} \mapsto \langle \mathbf{u}^{(t)}, \mathbf{x} \rangle$. The utility vector $\mathbf{u}^{(t)}$ can depend adversarially on the outputs $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}$ if the no-regret algorithm is deterministic (*i.e.*, does not use randomness internally^a).

The Φ -regret cumulated up to any time T compared to a transformation $\hat{\phi} \in \Phi$ is defined as the quantity

$$\Phi\text{-Reg}^{(T)}(\hat{\phi}) := \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\phi}(\mathbf{x}^{(t)}) - \mathbf{x}^{(t)} \rangle, \quad (3.3)$$

whereas the Φ -regret cumulated by to time T by the algorithm is the maximum Φ -regret cumulated compared to any of the transformations in Φ ,

$$\Phi\text{-Reg}^{(T)} := \max_{\hat{\phi} \in \Phi} \Phi\text{-Reg}^{(T)}(\hat{\phi}). \quad (3.4)$$

^aWhen randomness is involved, the utility vector cannot depend adversarially on $\mathbf{x}^{(t)}$ or guaranteeing sublinear regret would be impossible. Rather, $\mathbf{u}^{(t)}$ must be conditionally independent on $\mathbf{x}^{(t)}$, given all past random outcomes.

Calls to NextStrategy and ObserveUtility keep alternating to each other: first, the no-regret algorithm will output a point $\mathbf{x}^{(1)}$, then it will receive feedback $\mathbf{u}^{(1)}$ from the environment, then it will output a new point $\mathbf{x}^{(2)}$, and so on. The decision-making encoded by the no-regret algorithm is *online*, in the sense that at each time t , the output of the no-regret algorithm can depend on the prior outputs $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t-1)}$ and corresponding observed utility vectors $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(t-1)}$, but no information about future utilities is available.

3.2.1 The canonical optimistic learning setup (COLS) for games

As mentioned in the discussion preceding [Definition 3.1](#), a no- Φ -regret algorithm is an abstract learning algorithm that outputs strategies (taking into account a prediction vector $\mathbf{m}^{(t)}$), and receives as feedback a utility vector $\mathbf{u}^{(t)}$, guaranteeing that the $\mathbf{x}^{(t)}$ accumulate sublinear Φ -regret with respect to the $\mathbf{u}^{(t)}$ over time. We now reconnect the definition to the context of learning in games, by specifying how the $\mathbf{m}^{(t)}$ and $\mathbf{u}^{(t)}$ are defined. In particular, we will refer to these specific choices as defining the *canonical optimistic learning setup (COLS)*.

In the COLS, at all times $t \in \{1, 2, \dots\}$ each player $i \in \llbracket n \rrbracket$ picks mixed strategies $\mathbf{x}_i^{(t)}$ according to some no- Φ -regret learning algorithm $\mathcal{R}_i.\text{NextStrategy}(\mathbf{m}_i^{(t)})$, where the prediction vector $\mathbf{m}_i^{(t)}$ is defined as the previous loss $\mathbf{m}_i^{(t)} := \mathbf{u}_i^{(t-1)}$ if $t \geq 2$, and $\mathbf{m}_i^{(1)} := \mathbf{0}$ otherwise. Then, all players receive as feedback the utility vectors $\mathbf{u}_i^{(t)} := \nabla_{\mathbf{x}_i} u_i(\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_m^{(t)})$, that is, the gradient of their

own utility evaluated in the mixed strategies last output by the learning players. In this setup, the Φ -regret accumulated by each player $i \in \llbracket n \rrbracket$ is equal to the quantity $\Phi\text{-Reg}_i(\{\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}\}_{t=1}^T)$ defined in (3.1), enabling the wealth of connections between Φ -regret and equilibrium computation laid out in Section 3.1.2.

3.2.2 The important special case of external regret minimization

The special case where Φ is chosen to be the set of constant transformations is so important that it warrants its own definition and notation.

Definition 3.2 (No-regret algorithm). Given a set \mathcal{X} , an *no-external-regret algorithm* for \mathcal{X} , or simply “*no-regret algorithm for \mathcal{X}* ”, is a no- Φ^{const} -regret algorithm, where Φ^{const} is the set of all *constant transformations*

$$\Phi^{\text{const}} := \{\phi_{\hat{\mathbf{x}}} : \hat{\mathbf{x}} \in \mathcal{X}\}, \quad \text{where } \phi_{\hat{\mathbf{x}}} : \mathcal{X} \ni \mathbf{x} \mapsto \hat{\mathbf{x}}.$$

Its corresponding Φ^{const} -regret (Equations (3.3) and (3.4)) is called “*external regret*” or simply “*regret*”. We will often indicate external regret with the symbol $\text{Reg}^{(T)}$ rather than $\Phi^{\text{const}}\text{-Reg}$. In particular, we will let

$$\text{Reg}^{(T)}(\hat{\mathbf{x}}) := \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle, \quad \text{Reg}^{(T)} := \max_{\hat{\mathbf{x}} \in \mathcal{X}} \text{Reg}^{(T)}(\hat{\mathbf{x}}). \quad (3.5)$$

3.2.3 Reducing Φ -regret minimization to external regret minimization

As mentioned, a no-external-regret algorithm is a particular case of a no- Φ -regret algorithm. Then, the problem of constructing a no-external-regret algorithm for a set \mathcal{X} cannot be harder than the problem of constructing a no- Φ -regret algorithm for \mathcal{X} for a richer set of transformation functions Φ . It might then seem surprising that there exists a construction that reduces Φ -regret minimization to regret minimization. More precisely, a result by G. J. Gordon, A. Greenwald, and Marks (2008) gives a way to construct a no- Φ -regret algorithm for a generic set \mathcal{X} , starting from any no-external-regret algorithm *for the set of functions* Φ . We present the result next, as it will be an important component in the construction of learning algorithms that guarantee convergence to the set of extensive-form correlated equilibria (Chapter 8).

Theorem 3.1 (G. J. Gordon, A. Greenwald, and Marks, 2008). Let \mathcal{R} be a deterministic no-regret algorithm for the set of transformations Φ whose (external) cumulative regret $\text{Reg}^{(T)}$ grows sublinearly in T , and assume that every $\phi \in \Phi$ admits a fixed point $\phi(\mathbf{x}) = \mathbf{x} \in \mathcal{X}$. Then, a

no- Φ -regret algorithm \mathcal{R}_Φ can be constructed starting from \mathcal{R} as follows:

- Each call to $\mathcal{R}_\Phi.\text{NextStrategy}$ first calls NextStrategy on \mathcal{R} to obtain the next transformation $\phi^{(t)}$. Then, a fixed point $\mathbf{x}^{(t)} = \phi^{(t)}(\mathbf{x}^{(t)})$ is computed and output.
- Each call to $\mathcal{R}_\Phi.\text{ObserveUtility}(\mathbf{u}^{(t)})$ with linear utility vector $\mathbf{u}^{(t)}$ first constructs the linear utility vector $\mathbf{L}^{(t)}$ corresponding to the linear map $\phi \mapsto \langle \mathbf{u}^{(t)}, \phi(\mathbf{x}^{(t)}) \rangle$, where $\mathbf{x}^{(t)}$ is the last-output strategy. Then, it passes $\mathbf{L}^{(t)}$ to \mathcal{R} by executing $\mathcal{R}.\text{ObserveUtility}(\mathbf{L}^{(t)})$.^b

Furthermore, the Φ -regret $\Phi\text{-Reg}^{(T)}$ cumulated up to time T by \mathcal{R}_Φ we have just defined is exactly equal to the (external) cumulative regret $\text{Reg}^{(T)}$ cumulated by \mathcal{R} :

$$\Phi\text{-Reg}^{(T)}(\hat{\phi}) = \text{Reg}^{(T)}(\hat{\phi}) \quad \forall \hat{\phi} \in \Phi, \quad T = 1, 2, \dots$$

So, because the regret cumulated by \mathcal{R} grows sublinearly by hypothesis, then so does the Φ -regret cumulated by \mathcal{R}_Φ .

^bOn the surface, it might look like $\mathbf{L}^{(t)}$ is independent on the last output $\phi^{(t)}$ of the no-regret algorithm \mathcal{R} , and thus, that it trivially satisfies the requirements of [Definition 3.2](#). However, that is not true: $\mathbf{x}^{(t)}$ is a fixed point of $\phi^{(t)}$, and since $\mathbf{x}^{(t)}$ enters into the definition of $\mathbf{L}^{(t)}$, if \mathcal{R} picks $\phi^{(t)}$ randomly, it might very well be that $\mathbf{L}^{(t)}$ is not conditionally independent on $\phi^{(t)}$. We sidestep this issue by requiring that \mathcal{R} is deterministic (cf. Footnote [a](#)).

Proof. We propose an independent proof of the above statement given the brevity and elegance of the argument. By construction, the algorithm \mathcal{R} outputs transformations $\phi^{(1)}, \phi^{(2)}, \dots \in \Phi$ and receives the linear utilities $\phi \mapsto \langle \mathbf{L}^{(t)}, \phi \rangle := \langle \mathbf{u}^{(t)}, \phi(\mathbf{x}^{(t)}) \rangle$. Hence, its cumulative external regret $\text{Reg}^{(T)}$ is by definition

$$\text{Reg}^{(T)}(\hat{\phi}) = \sum_{t=1}^T \left\langle \mathbf{u}^{(t)}, \hat{\phi}(\mathbf{x}^{(t)}) - \phi^{(t)}(\mathbf{x}^{(t)}) \right\rangle.$$

Since by construction $\mathbf{x}^{(t)}$ is a fixed point of $\phi^{(t)}$, then $\phi^{(t)}(\mathbf{x}^{(t)}) = \mathbf{x}^{(t)}$, and therefore

$$\text{Reg}^{(T)}(\hat{\phi}) = \sum_{t=1}^T \left\langle \mathbf{u}^{(t)}, \hat{\phi}(\mathbf{x}^{(t)}) - \mathbf{x}^{(t)} \right\rangle \stackrel{(3.3)}{=} \Phi\text{-Reg}^{(T)}(\hat{\phi}),$$

as we wanted to show. □

3.2.4 Degrees of predictivity

In this dissertation we propose a systematic classification of predictive regret minimization algorithms on the basis of how effectively they are able to take into account predictions. In particular, we introduce the following nomenclature.

Definition 3.3. A no- Φ -regret algorithm for a set \mathcal{X} is said to be:

- **Weakly predictive** relative to learning rate $\eta > 0$ and norm $\|\cdot\|$, if its Φ -regret satisfies a relation of the form

$$\Phi\text{-Reg}^{(T)} = \mathcal{O}_T \left(\frac{1}{\eta} + \eta \sum_{t=1}^T \|\mathbf{u}^{(t)} - \mathbf{m}^{(t)}\|_*^2 \right);$$

- **RVU-Predictive**^[3.b] relative to learning rate $\eta > 0$ and norm $\|\cdot\|$, if its Φ -regret satisfies a relation of the form

$$\Phi\text{-Reg}^{(T)} = \mathcal{O}_T \left(\frac{1}{\eta} + \eta \sum_{t=1}^T \|\mathbf{u}^{(t)} - \mathbf{m}^{(t)}\|_*^2 - \frac{1}{\eta} \sum_{t=2}^T \|\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}\|^2 \right); \quad (3.6)$$

- **Strongly predictive** relative to learning rate $\eta > 0$ and norm $\|\cdot\|$, if its Φ -regret satisfies a relation of the form

$$\max\{0, \Phi\text{-Reg}^{(T)}\} = \mathcal{O}_T \left(\frac{1}{\eta} + \eta \sum_{t=1}^T \|\mathbf{u}^{(t)} - \mathbf{m}^{(t)}\|_*^2 - \frac{1}{\eta} \sum_{t=2}^T \|\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}\|^2 \right).$$

The following chain of implications follows directly from the definition above:

$$(\text{strongly predictive}) \implies (\text{RVU-predictive}) \implies (\text{weakly predictive}).$$

3.3 No-external-regret algorithms for probability simplexes

In this section we review a few no-external-regret algorithms for probability simplexes. Several more are known in the literature.

3.3.1 Multiplicative weights update (MWU) and its predictive variant (OMWU)

The *predictive multiplicative weights update* algorithm is arguably the best studied no-external-regret algorithm with probability simplex, due to its strong properties that will recall in this subsection. We remark that in the literature, the predictive multiplicative weights update algorithm is often given under the assumption that the prediction $\mathbf{m}^{(t)}$ is set to $\mathbf{u}^{(t-1)}$ at all times t , as is standard within the COLS, and is known commonly under the alternative names *optimistic multiplicative weights update*, *optimistic randomized weighted majority*, and *optimistic hedge*. In this dissertation we

^[3.b]The term RVU stands for *Regret bounded by Variation of Utilities (RVU)* and was introduced by Syrgkanis, Agarwal, Luo, and Schapire (2015) to describe the functional form (3.6).

present the algorithm in its general form, that is, with no assumptions on $\mathbf{m}^{(t)}$, but still refer to it as OMWU to avoid confusion due to how widespread the acronym is. At all times t , OMWU produces as strategy the distribution $\mathbf{x}^{(t-1)} \in \Delta^d$ as specified in [Algorithm 3.1](#).

Algorithm 3.1: Predictive multiplicative weights update algorithm (OMWU)

Data: Learning rates $\eta^{(t)} > 0$

- 1 $\mathbf{u}^{(0)}, \mathbf{m}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^d; \mathbf{x}^{(0)} \leftarrow \frac{1}{d} \mathbf{1} \in \Delta^d$

- 2 **function** NextStrategy($\mathbf{m}^{(t)} \in \mathbb{R}^d$)
 - [▷ set $\mathbf{m}^{(t)} = \mathbf{0}$ for non-predictive variant]
 - 3 $\mathbf{w}^{(t)} \leftarrow \mathbf{u}^{(t-1)} - \mathbf{m}^{(t-1)} + \mathbf{m}^{(t)}$
 - 4 **for** $k \in \llbracket d \rrbracket$ **do**
 - 5 $\mathbf{x}^{(t)}[k] \leftarrow \frac{\mathbf{x}^{(t-1)}[k] \cdot \exp\{\eta^{(t)} \mathbf{w}^{(t)}[k]\}}{\sum_{k' \in \llbracket d \rrbracket} \mathbf{x}^{(t-1)}[k'] \cdot \exp\{\eta^{(t)} \mathbf{w}^{(t)}[k']\}}$
 - 6 **return** $\mathbf{x}^{(t)}$

- 7 **function** ObserveUtility($\mathbf{u}^{(t)}$)
 - [▷ No operation until the next call of NextStrategy]

The nonpredictive version of OMWU, called *multiplicative weights update (MWU)* (or alternatively *randomized weighted majority and hedge*), can be obtained from OMWU as the special case in which $\mathbf{m}^{(t)} = \mathbf{0}$ at all t . We mention the following well-known upper bound for the regret cumulated by OMWU.

Theorem 3.2 (Syrkkanis, Agarwal, Luo, and Schapire (2015)). After any T iterations, the regret cumulated by the OMWU algorithm ([Algorithm 3.1](#)) set up with constant learning rate $\eta^{(t)} = \eta > 0$, satisfies the RVU-predictive regret bound

$$\text{Reg}^{(T)} \leq \frac{\log d}{\eta} + \eta \sum_{t=1}^T \|\mathbf{u}^{(t)} - \mathbf{m}^{(t)}\|_{\infty}^2 - \frac{1}{4\eta} \sum_{t=2}^T \|\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}\|_1^2.$$

Furthermore, each iteration runs in $\mathcal{O}(d)$ time.

In particular, [Theorem 3.2](#) immediately implies that as long as the losses and predictions have bounded norm, the choice of learning rate $\eta^{(t)} = 1/\sqrt{T}$ guarantees $\text{Reg}^{(T)} = \mathcal{O}_T(1/\sqrt{T})$.

In nonsequential (that is, normal-form) games, where each player $i \in \llbracket n \rrbracket$ has a single decision point with a set \mathcal{A}_i of actions, OMWU can be used to learn Nash and coarse correlated equilibria ([Section 5.2.2](#)). Specifically, when each player i learns within the COLS using OMWU with the same, constant learning rate $\eta_i^{(t)} := \eta$ as their learning algorithm \mathcal{R}_i , the following strong properties are known to hold. For simplicity we assume without loss of generality that all utilities in the game

are in the range $[0, 1]$ (if not, rescaling all utilities to satisfy the condition leaves the equilibria unchanged).

Theorem 3.3 ($\mathcal{O}_T(T^{1/4})$ per-player regret; Syrgkanis, Agarwal, Luo, and Schapire, 2015). For all T , if $\eta = \frac{T^{-1/4}}{\sqrt{n-1}}$, the regret of each player $i \in \llbracket n \rrbracket$ is bounded as

$$\text{Reg}_i^{(T)} \leq (4 + \log |\mathcal{A}_i|) \sqrt{n-1} \cdot T^{1/4}.$$

Theorem 3.4 (Near-optimal per-player regret; Daskalakis, Fishelson, and Golowich, 2021). There exist universal constants $C, C' > 1$ so that, for all T , if $\eta \leq \frac{1}{Cn \log^4 T}$, the regret of each player $i \in \llbracket n \rrbracket$ is bounded as

$$\text{Reg}_i^{(T)} \leq \frac{\log |\mathcal{A}_i|}{\eta} + C' \log T.$$

Theorem 3.5 (Optimal regret sum; Syrgkanis, Agarwal, Luo, and Schapire, 2015). If $\eta \leq \frac{1}{\sqrt{8(n-1)}}$, at all times T the sum of the players' regrets satisfies

$$\sum_{i \in \llbracket n \rrbracket} \text{Reg}_i^{(T)} \leq \frac{n}{\eta} \max_{i \in \llbracket n \rrbracket} \log |\mathcal{A}_i|.$$

We remark that while [Theorem 3.4](#) provides a substantially better bound than [Theorem 3.3](#) asymptotically, for moderate values of T , [Theorem 3.3](#) provides a better numerical bound.

When Γ is a *two-player zero-sum* game, the following also holds when the two players learn within the COLS using OMWU.

Theorem 3.6 (Last-iterate convergence). There exists a certain schedule of learning rates $\eta_i^{(t)}$ such that the players' strategies $(\mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)})$ converge to a Nash equilibrium of the game (Hsieh, Antonakopoulos, and Mertikopoulos, 2021). Furthermore, if Γ has a unique Nash equilibrium $(\mathbf{x}_1^*, \mathbf{x}_2^*)$ and each player uses any constant learning rate $\eta_i^{(t)} := \eta \leq \frac{1}{8}$, at all times t the strategy profile $(\mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)})$ satisfies

$$\text{KL}(\mathbf{x}_1^* \parallel \mathbf{x}_1^{(t)}) + \text{KL}(\mathbf{x}_2^* \parallel \mathbf{x}_2^{(t)}) \leq C(1 + C')^{-t},$$

where the constants C, C' only depend on the game, and $\text{KL}(\cdot \parallel \cdot)$ denotes the KL-divergence between two distributions (C. Wei, C. Lee, M. Zhang, and Luo, 2021).

3.3.2 Regret matching (RM), regret matching⁺ (RM⁺), and variants

Regret matching (RM, Hart and Mas-Colell, 2000) is a simple no-external-regret algorithm for probability simplexes that has been widely used for computational game solving due to its good practical performance and lack of hyperparameters. Regret matching⁺ (RM⁺, Tammelin, 2014; Tammelin, Burch, Johanson, and Bowling, 2015) is a modification of RM which has been repeatedly observed to perform better in practice, although no theoretical justification for such an improved performance is known. Both of the algorithm predate the introduction of predictions as a concept in online optimization, so their formulation—given in Section 4.3.1—explicitly ignores the prediction vector supplied to NextStrategy.

Algorithm 3.2: Regret matching (RM)	Algorithm 3.3: Regret matching ⁺ (RM ⁺)
1 $\mathbf{r}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^d, \mathbf{x}^{(0)} \leftarrow \mathbf{1}/d \in \Delta^d$	1 $\mathbf{z}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^d, \mathbf{x}^{(0)} \leftarrow \mathbf{1}/d \in \Delta^d$
2 function NextStrategy($_$)	2 function NextStrategy($_$)
3 $\boldsymbol{\theta}^{(t)} \leftarrow [\mathbf{r}^{(t-1)}]^+$	3 $\boldsymbol{\theta}^{(t)} \leftarrow [\mathbf{z}^{(t-1)}]^+$
4 if $\boldsymbol{\theta}^{(t)} \neq \mathbf{0}$ return $\mathbf{x}^{(t)} \leftarrow \boldsymbol{\theta}^{(t)} / \ \boldsymbol{\theta}^{(t)}\ _1$	4 if $\boldsymbol{\theta}^{(t)} \neq \mathbf{0}$ return $\mathbf{x}^{(t)} \leftarrow \boldsymbol{\theta}^{(t)} / \ \boldsymbol{\theta}^{(t)}\ _1$
5 else return $\mathbf{x}^{(t)} \leftarrow$ any point in Δ^d	5 else return $\mathbf{x}^{(t)} \leftarrow$ any point in Δ^d
6 function ObserveUtility($\mathbf{u}^{(t)}$)	6 function ObserveUtility($\mathbf{u}^{(t)}$)
7 $\mathbf{r}^{(t)} \leftarrow \mathbf{r}^{(t-1)} + \mathbf{u}^{(t)} - \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \mathbf{1}$	7 $\mathbf{z}^{(t)} \leftarrow [\mathbf{z}^{(t-1)} + \mathbf{u}^{(t)} - \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \mathbf{1}]^+$

Predictive variants of RM and RM⁺ will be given in Chapter 4. The predictive variants of RM and RM⁺ retain all advantages of the original, nonpredictive algorithm (including the lack of hyperparameters), while at the same time capitalizing from the benefit of (weak) predictivity, further increasing the practical performance of these no-external-regret algorithms.

Regret matching and regret matching⁺ satisfy the following (nonpredictive) regret bound.

Theorem 3.7 (Farina, Kroer, and Sandholm, 2021b). The external regret cumulated up to any time T by the RM and RM⁺ algorithm satisfies, for all $\eta > 0$, the bound

$$\text{Reg}^{(T)} \leq \sqrt{2 \sum_{t=1}^T \|\mathbf{u}^{(t)} - \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \mathbf{1}\|_2^2}.$$

In particular, Theorem 3.7 shows that whenever the norm of the utility vectors is bounded by a time-independent quantity, the external regret incurred by RM and RM⁺ is upper bounded as $\text{Reg}^{(T)} = \mathcal{O}_T(\sqrt{T})$, a sublinear quantity.

Variants of RM have been proposed in the literature, including discounted regret matching (Discounted RM) and linear regret matching (Brown and Sandholm, 2019). These have been

observed to perform better in practice, though they are less well studied, and no predictive versions are known. We recall Discounted RM and Linear RM in [Algorithms 3.4](#) and [3.5](#) respectively.

Algorithm 3.4: Discounted regret matching (Discounted RM)	Algorithm 3.5: Linear regret matching (Linear RM)
<p>Data: $\alpha := 3/2, \beta := 0$ discounting parameters</p> <p>1 $\mathbf{r}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^d, \mathbf{x}^{(0)} \leftarrow \mathbf{1}/d \in \Delta^d$</p> <hr style="border: 0.5px solid black;"/> <p>2 function NextStrategy($_$)</p> <p>3 $\boldsymbol{\theta}^{(t)} \leftarrow [\mathbf{r}^{(t-1)}]^+$</p> <p>4 if $\boldsymbol{\theta}^{(t)} \neq \mathbf{0}$ return $\mathbf{x}^{(t)} \leftarrow \boldsymbol{\theta}^{(t)} / \ \boldsymbol{\theta}^{(t)}\ _1$</p> <p>5 else return $\mathbf{x}^{(t)} \leftarrow$ any point in Δ^d</p> <hr style="border: 0.5px solid black;"/> <p>6 function ObserveUtility($\mathbf{u}^{(t)} \in \mathbb{R}^d$)</p> <p>7 for $k \in \llbracket d \rrbracket$ do</p> <p>8 if $\mathbf{r}^{(t-1)}[k] > 0$ then</p> <p>9 $\mathbf{r}^{(t)}[k] \leftarrow \mathbf{r}^{(t-1)}[k] \cdot t^\alpha / (t^\alpha + 1)$</p> <p>10 else</p> <p>11 $\mathbf{r}^{(t)}[k] \leftarrow \mathbf{r}^{(t-1)}[k] \cdot t^\beta / (t^\beta + 1)$</p> <p>12 $\mathbf{r}^{(t)} \leftarrow \mathbf{r}^{(t)} + \mathbf{u}^{(t)} - \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \mathbf{1}$</p>	<p>1 $\mathbf{r}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^d, \mathbf{x}^{(0)} \leftarrow \mathbf{1}/d \in \Delta^d$</p> <hr style="border: 0.5px solid black;"/> <p>2 function NextStrategy($_$)</p> <p>3 $\boldsymbol{\theta}^{(t)} \leftarrow [\mathbf{r}^{(t-1)}]^+$</p> <p>4 if $\boldsymbol{\theta}^{(t)} \neq \mathbf{0}$ return $\mathbf{x}^{(t)} \leftarrow \boldsymbol{\theta}^{(t)} / \ \boldsymbol{\theta}^{(t)}\ _1$</p> <p>5 else return $\mathbf{x}^{(t)} \leftarrow$ any point in Δ^d</p> <hr style="border: 0.5px solid black;"/> <p>6 function ObserveUtility($\mathbf{u}^{(t)} \in \mathbb{R}^d$)</p> <p>7 $\mathbf{r}^{(t)} \leftarrow \mathbf{r}^{(t-1)} + t \cdot (\mathbf{u}^{(t)} - \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \mathbf{1})$</p>

Part II

Computation of coarse-correlated and Nash equilibria

Chapter 4

Composability of learning dynamics and predictive counterfactual regret minimization

4.1 Contributions and related work

In this chapter we will show that predictive no-external-regret learning algorithms lend themselves to a rather natural *compositional* calculus. In particular, we will show that for many composite strategy sets obtained via convexity-preserving operations on simpler sets (*e.g.*, Cartesian product, convex hull, intersection, and more), a predictive no-regret learning algorithm can be obtained by combining, as black boxes, any predictive no-regret algorithms for the simpler sets. We call these composition rules *regret circuits*. This approach has parallels with the calculus of convex sets and functions found in books such as Boyd and Vandenberghe (2004). It likewise is reminiscent of *disciplined convex programming* (Grant, Boyd, and Ye, 2006), which emphasizes the solving of convex programs via composition of simple convex functions and sets.

Applications to sequence-form polytopes An especially notable application of our framework is the construction of a weakly-predictive no-external-regret algorithm for sequence-form strategy polytopes (Section 4.3). In that case, our compositional calculus will show how *any* weakly-predictive no-external-regret algorithm for a single-decision (*i.e.*, nonsequential) problems can be promoted to a weakly-predictive no-external-regret algorithm for tree-form problems. The resulting algorithm is a generalization of the popular algorithm known as *counterfactual regret minimization* (CFR) (Zinkevich, Bowling, Johanson, and Piccione, 2007). CFR and its newer variants (Lanctot, Waugh, Zinkevich, and Bowling, 2009; Brown and Sandholm, 2015; Tammelin,

Burch, Johanson, and Bowling, 2015; Brown, Kroer, and Sandholm, 2017; Brown and Sandholm, 2017a; Brown and Sandholm, 2019; Brown and Sandholm, 2019), have been a central component in several recent milestones in solving imperfect-information games. Bowling, Burch, Johanson, and Tammelin (2015) used CFR⁺ to near-optimally solve heads-up limit Texas hold'em. Brown and Sandholm (2017c) used CFR variants, along with other scalability techniques such as real-time endgame solving (Ganzfried and Sandholm, 2015; Burch, Johanson, and Bowling, 2014; Moravcik, Schmid, Ha, Hladik, and Gaukrodger, 2016; Brown and Sandholm, 2017b) and automated action abstraction Brown and Sandholm, 2014, to create *Libratus*, an AI that beat top human specialist professionals at the larger game of heads-up no-limit Texas hold'em. Moravčík, Schmid, Burch, Viliam Lisý, Morrill, Bard, Davis, Waugh, Johanson, and Bowling (2017) also used CFR variants and endgame solving to beat professional human players at that game. CFR was originally proposed as an algorithm for approximating Nash equilibria in two-player zero-sum game, with a rather lengthy and complex proof. By using our regret circuit formalism, we will construct an algorithm that specializes into CFR when used in self-play in two-player zero-sum games, but is otherwise a no-external-regret algorithm in general. In other words, we move away from the understanding of CFR as a method operating on a two-player game tree, and instead embrace the one-sided, TFDP-based point of view, whereby CFR is a learning algorithm for a generic player's tree-form decision problem in a game. This point of view makes it apparent that CFR can be used beyond two-player zero-sum games, for all applications in which no-external-regret is useful (Section 3.1.2). All this comes with the benefit of a radically simpler proof of correctness, and enables using predictions within the CFR framework in a principled way. As we show in Section 4.3.2, this leads to some of the fastest learning algorithms known today for equilibrium computation in large extensive-form games, when paired with predictive no-regret algorithms for single-decision problems, such as ref PRM⁺.

Other applications throughout the dissertation However, the applications of our compositional calculus go well beyond constructing algorithms for sequence-form strategy spaces. The framework laid out in this chapter will lead to practical algorithms when constructing no- Φ -regret learning dynamics leading to extensive-form correlated equilibria (Chapter 8), and social-welfare-maximizing equilibria (Chapter 10).

Algorithms constructed via the regret circuits compositional calculus tend to perform better in practice than those constructed using other approaches, for example those based on online convex optimization techniques (Chapter 5). In fact, methods based on composition have several positive features that are hard to mirror in other, more monolithic approaches to constructing no-external-regret algorithms. For one, by treating every no-regret algorithm that appears in the compositional construction as an independent black box, the compositional approach enables one to select the best individual algorithm for each of them. Second, our framework is amenable to pruning or warm-starting techniques in different parts of the composition, and substituting






one or more parts of the composition with an approximation. Finally, it is immediately clear in what cases the no-external-regret algorithms that are being composed can be run in distributed and parallel environments.

4.2 Regret circuits

In this section we begin developing a set of rules for how no-external-regret algorithms can be constructed for composite convex sets. Specifically, given no-regret algorithms for convex sets (say, \mathcal{X} and \mathcal{Y}) we show how to compose these no-regret algorithms for the following standard convexity-preserving operations performed on the sets: convex hull, Cartesian product, affine transformation. In [Section 4.2.5](#), we will also introduce and give a regret circuit for a special operation, called *scaled extension*, which will become important later on in the dissertation, specifically in [Chapter 10](#).

4.2.1 Pictorial depiction of regret circuits

Regret circuits define the ways in which the inputs (predictions and utility vectors) and outputs (strategies) of no-regret algorithms can be combined to define a composite no-regret algorithm. To aid intuition and memory, we will pictorially visualize regret circuits as circuits in which components are no-regret-algorithms, and wires carry utility vectors, predictions, and decisions. A simple example of what such a representation will look like in this dissertation is given to the right of [Algorithm 4.1](#) on [page 52](#).

- a no-external-regret algorithm is drawn as a box with the name of the algorithm placed at the center, such as . We will use the convention of calling the no-regret algorithms that are being combined as $\mathcal{R}_{\mathcal{X}}, \mathcal{R}_{\mathcal{Y}}, et\ cetera$. The subscript of \mathcal{R} , say \mathcal{X} or \mathcal{Y} , corresponds to the set from which the algorithm draws its strategies.
- Input wires to the no-regret algorithms carry prediction and utility vectors the algorithms receive at the generic time t . They are dashed and colored dark red, like this: .
- Output wires from no-regret algorithms carry the strategy output at the generic time t . They are solid lines colored dark blue, like this: .
- The node \oplus denotes an *adder*, that is a node that outputs the sum of all its inputs.
- The node  is used to denote an operation, other than addition, that manipulates its input(s) into its (single) output.
- The black dot \bullet denotes a connection point between wires: all wires leaving the connection point carry the same signal (utility, prediction, or strategy) as the only wire entering the connection point.
- The boundary of the regret circuits is marked by a gray rectangle, such as .

4.2.2 Cartesian product

Let $\mathcal{X} \subseteq \mathbb{R}^{d_x}$ and $\mathcal{Y} \subseteq \mathbb{R}^{d_y}$ be two sets, and let $\mathcal{R}_\mathcal{X}$ and $\mathcal{R}_\mathcal{Y}$ be no-regret algorithms for \mathcal{X} and \mathcal{Y} respectively. In this section we show that it is possible to minimize regret on $\mathcal{X} \times \mathcal{Y}$ by simply minimizing it on \mathcal{X} and \mathcal{Y} independently—via $\mathcal{R}_\mathcal{X}$ and $\mathcal{R}_\mathcal{Y}$ —and then combining the decisions.

More precisely, we can combine $\mathcal{R}_\mathcal{X}$ and $\mathcal{R}_\mathcal{Y}$ to form a no-external-regret algorithm for the Cartesian product $\mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^{d_x+d_y}$ by means of the following natural procedure:

- To output the next strategy, given a prediction^[4.a] vector $\mathbf{m}^{(t)} := (\mathbf{m}_x^{(t)}, \mathbf{m}_y^{(t)}) \in \mathbb{R}^{d_x+d_y}$, we feed $\mathbf{m}_x^{(t)}$ and $\mathbf{m}_y^{(t)}$ to \mathcal{X} and \mathcal{Y} respectively, and receive their next strategies $\mathbf{x}^{(t)} \in \mathcal{X}$, $\mathbf{y}^{(t)} \in \mathcal{Y}$. We then proceed to outputting the strategy $(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \in \mathcal{X} \times \mathcal{Y}$.
- Given the utility vector $\mathbf{u}^{(t)} := (\mathbf{u}_x^{(t)}, \mathbf{u}_y^{(t)}) \in \mathbb{R}^{d_x+d_y}$, we forward $\mathbf{u}_x^{(t)}$ to $\mathcal{R}_\mathcal{X}$ and $\mathbf{u}_y^{(t)}$ to $\mathcal{R}_\mathcal{Y}$.

The algorithm we just described is summarized with pseudocode in [Algorithm 4.1](#) and depicted pictorially contextually to its right.

Algorithm 4.1: Regret circuit for $\mathcal{X} \times \mathcal{Y}$

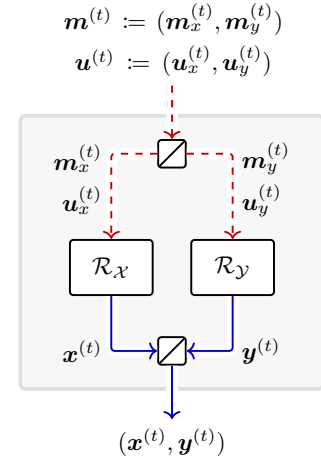
```

1 function NextStrategy( $\mathbf{m}^{(t)} := (\mathbf{m}_x^{(t)}, \mathbf{m}_y^{(t)}) \in \mathbb{R}^{d_x+d_y}$ )
  | [ $\triangleright$  Set  $\mathbf{m}^{(t)} = \mathbf{0}$  for non-predictive version]
2    $\mathbf{x}^{(t)} \leftarrow \mathcal{R}_\mathcal{X}.\text{NextStrategy}(\mathbf{m}_x^{(t)})$ 
3    $\mathbf{y}^{(t)} \leftarrow \mathcal{R}_\mathcal{Y}.\text{NextStrategy}(\mathbf{m}_y^{(t)})$ 
4   return  $(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \in \mathcal{X} \times \mathcal{Y}$ 


---


5 function ObserveUtility( $\mathbf{u}^{(t)} := (\mathbf{u}_x^{(t)}, \mathbf{u}_y^{(t)}) \in \mathbb{R}^{d_x+d_y}$ )
6    $\mathcal{R}_\mathcal{X}.\text{ObserveUtility}(\mathbf{u}_x^{(t)})$ 
7    $\mathcal{R}_\mathcal{Y}.\text{ObserveUtility}(\mathbf{u}_y^{(t)})$ 

```



Regret analysis We now turn our attention to the analysis of the regret cumulated by the circuit. By construction, at all times t the no-regret algorithms $\mathcal{R}_\mathcal{X}$ and $\mathcal{R}_\mathcal{Y}$ produce iterates $\mathbf{x}^{(t)}$ and $\mathbf{y}^{(t)}$, respectively, and observe utility vectors $\mathbf{u}_\mathcal{X}^{(t)}$ and $\mathbf{u}_\mathcal{Y}^{(t)}$, respectively. Hence, the regret accumulated by $\mathcal{R}_\mathcal{X}$ and $\mathcal{R}_\mathcal{Y}$ up to any T relative to any comparators $\hat{\mathbf{x}} \in \mathcal{X}$, $\hat{\mathbf{y}} \in \mathcal{Y}$ is given by

$$\text{Reg}_\mathcal{X}^{(T)}(\hat{\mathbf{x}}) := \sum_{t=1}^T \langle \mathbf{u}_x^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle, \quad \text{Reg}_\mathcal{Y}^{(T)}(\hat{\mathbf{y}}) := \sum_{t=1}^T \langle \mathbf{u}_y^{(t)}, \hat{\mathbf{y}} - \mathbf{y}^{(t)} \rangle. \quad (4.1)$$

^[4.a]In this dissertation predictions of the next utility vectors are always assumed to be given as input, with the understanding that the non-predictive version of the algorithm can be recovered by setting the prediction to be $\mathbf{0}$, the zero vector.

On the other hand, the regret circuit produces strategies $(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \in \mathcal{X} \times \mathcal{Y}$ and received utility vectors $(\mathbf{u}_x^t, \mathbf{u}_y^t)$ at all times t . Hence, its accumulated regret compared to any $(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \in \mathcal{X} \times \mathcal{Y}$ satisfies

$$\text{Reg}_{\mathcal{X} \times \mathcal{Y}}^{(T)}(\hat{\mathbf{x}}, \hat{\mathbf{y}}) := \sum_{t=1}^T \left\langle \begin{pmatrix} \mathbf{u}_x^{(t)} \\ \mathbf{u}_y^{(t)} \end{pmatrix}, \begin{pmatrix} \hat{\mathbf{x}} \\ \hat{\mathbf{y}} \end{pmatrix} - \begin{pmatrix} \mathbf{x}^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\rangle = \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) + \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}), \quad (4.2)$$

where we used (4.1) in the last step. By taking a maximum over all $(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \in \mathcal{X} \times \mathcal{Y}$ in (4.2) we further obtain

$$\text{Reg}_{\mathcal{X} \times \mathcal{Y}}^{(T)} = \text{Reg}_{\mathcal{X}}^{(T)} + \text{Reg}_{\mathcal{Y}}^{(T)}. \quad (4.3)$$

In summary, collecting (4.2) and (4.3) we have just proven the following characterization of the regret of Algorithm 4.1.

Theorem 4.1. The regret circuit of Algorithm 4.1 provides a no-regret algorithm for the Cartesian product $\mathcal{X} \times \mathcal{Y}$ whose regret $\text{Reg}_{\mathcal{X} \times \mathcal{Y}}^{(T)}$ accumulated up to any time T is related to the regrets $\text{Reg}_{\mathcal{X}}^{(T)}$ and $\text{Reg}_{\mathcal{Y}}^{(T)}$ accumulated by $\mathcal{R}_{\mathcal{X}}$ and $\mathcal{R}_{\mathcal{Y}}$, respectively, according to the equalities

- i. $\text{Reg}_{\mathcal{X} \times \mathcal{Y}}^{(T)}(\hat{\mathbf{x}}, \hat{\mathbf{y}}) = \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) + \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}})$, for all $\hat{\mathbf{x}} \in \mathcal{X}, \hat{\mathbf{y}} \in \mathcal{Y}$; and
- ii. $\text{Reg}_{\mathcal{X} \times \mathcal{Y}}^{(T)} = \text{Reg}_{\mathcal{X}}^{(T)} + \text{Reg}_{\mathcal{Y}}^{(T)}$.

So, in particular, $\text{Reg}_{\mathcal{X} \times \mathcal{Y}}^{(T)}$ grows sublinearly in T since $\text{Reg}_{\mathcal{X}}^{(T)}$ and $\text{Reg}_{\mathcal{Y}}^{(T)}$ are sublinear by the hypothesis that $\mathcal{R}_{\mathcal{X}}$ and $\mathcal{R}_{\mathcal{Y}}$ are no-regret algorithms.

Extension to multiple sets We conclude the section with a couple of remarks on how one can adapt the construction we have been describing thus far to Cartesian products of more than two sets, say the Cartesian product $\mathcal{Z} := \mathcal{X}_1 \times \mathcal{X}_2 \times \dots \times \mathcal{X}_n$ given access to no-regret algorithms $\mathcal{R}_{\mathcal{X}_i}$ for the individual sets. Two avenues are possible.

- First, one can observe that $\mathcal{X}_1 \times \mathcal{X}_2 \times \dots \times \mathcal{X}_n = (((\mathcal{X}_1 \times \mathcal{X}_2) \times \mathcal{X}_3) \times \dots) \times \mathcal{X}_n$. Hence, it is possible to construct a composite no-regret algorithm for \mathcal{Z} by composing Algorithm 4.1 $n - 1$ times: first, combining $\mathcal{R}_{\mathcal{X}_1}$ and $\mathcal{R}_{\mathcal{X}_2}$ to obtain algorithm $\mathcal{R}_{\mathcal{X}_1 \times \mathcal{X}_2}$, then combining the algorithm we just obtained with $\mathcal{R}_{\mathcal{X}_3}$, and so on.
- Alternatively, we point out that the idea behind Algorithm 4.1 and its corresponding analysis generalize straightforwardly to multiple sets, by defining a regret circuit that is able to output strategies on $\mathcal{X}_1 \times \dots \times \mathcal{X}_n$ by combining the individual strategies returned by the individual algorithms $\mathcal{R}_{\mathcal{X}_1}, \dots, \mathcal{R}_{\mathcal{X}_n}$.

4.2.3 Convex hull

We now show how to combine \mathcal{R}_X and \mathcal{R}_Y for two sets $X \subseteq \mathbb{R}^d$ and $Y \subseteq \mathbb{R}^d$ to construct a no-external-regret algorithm for the convex hull

$$\text{co}\{X, Y\} := \left\{ \lambda \mathbf{x} + (1 - \lambda) \mathbf{y} : \mathbf{x} \in X, \mathbf{y} \in Y, \lambda \in [0, 1] \right\}.$$

The construction is more involved than in the case of Cartesian products we saw in [Section 4.2.2](#), but some of the ideas carry over. Like in the Cartesian product construction, we will use the no-regret algorithms \mathcal{R}_X and \mathcal{R}_Y to propose candidate points $\mathbf{x}^{(t)}, \mathbf{y}^{(t)}$. However, in this case, we cannot simply return the pair $(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})$. Rather, we will use a third no-regret algorithm \mathcal{R}_{Δ^2} for the 2-simplex Δ^2 to decide how to “mix” $\mathbf{x}^{(t)}$ and $\mathbf{y}^{(t)}$. More precisely, we will do the following.

- To output the next strategy given the prediction vector $\mathbf{m}^{(t)} \in \mathbb{R}^d$, we ask \mathcal{R}_X and \mathcal{R}_Y for their next strategies—call them $\mathbf{x}^{(t)}$ and $\mathbf{y}^{(t)}$ —upon supplying $\mathbf{m}^{(t)}$ to both. Then, we will ask \mathcal{R}_{Δ^2} for its next strategy $\lambda^{(t)} = (\lambda_1^{(t)}, \lambda_2^{(t)}) \in \Delta^2$ given prediction

$$\mathbf{m}_{\Delta^2}^{(t)} := \begin{pmatrix} \langle \mathbf{m}^{(t)}, \mathbf{x}^{(t)} \rangle \\ \langle \mathbf{m}^{(t)}, \mathbf{y}^{(t)} \rangle \end{pmatrix},$$

and return the convex combination $\lambda_1^{(t)} \mathbf{x}^{(t)} + \lambda_2^{(t)} \mathbf{y}^{(t)} \in \text{co}\{X, Y\}$.

- When at time t we receive the utility vector $\mathbf{u}^{(t)} \in \mathbb{R}^d$, we forward $\mathbf{u}^{(t)}$ to \mathcal{R}_X and \mathcal{R}_Y . Then, we forward to \mathcal{R}_{Δ^2} the utility vector

$$\mathbf{u}_{\Delta^2}^{(t)} := \begin{pmatrix} \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \\ \langle \mathbf{u}^{(t)}, \mathbf{y}^{(t)} \rangle \end{pmatrix}. \quad (4.4)$$

The algorithm we just described is reported in pseudocode in [Algorithm 4.2](#) and pictorially contextually to the right.

Regret analysis By construction, the no-regret algorithms \mathcal{R}_X and \mathcal{R}_Y output strategies $\mathbf{x}^{(t)}, \mathbf{y}^{(t)}$ and observe utilities $\mathbf{u}^{(t)}$ at all time t . Hence, their regret cumulated up to time T relative to any comparators $\hat{\mathbf{x}} \in X, \hat{\mathbf{y}} \in Y$ is given by

$$\text{Reg}_X^{(T)}(\hat{\mathbf{x}}) := \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle \quad (4.5)$$

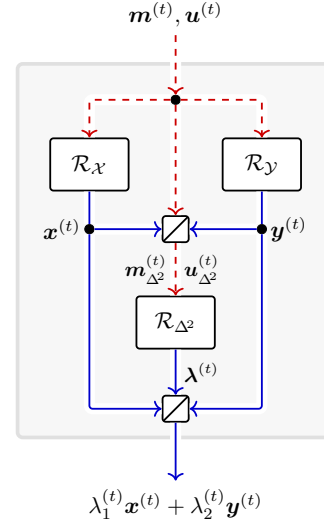
$$\text{Reg}_Y^{(T)}(\hat{\mathbf{y}}) := \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\mathbf{y}} - \mathbf{y}^{(t)} \rangle. \quad (4.6)$$

Algorithm 4.2: Regret circuit for $\text{co}\{\mathcal{X}, \mathcal{Y}\}$

```

1 function NextStrategy( $\mathbf{m}^{(t)} \in \mathbb{R}^d$ )
  [ $\triangleright$  Set  $\mathbf{m}^{(t)} = \mathbf{0}$  for non-predictive version]
2    $\mathbf{x}^{(t)} \in \mathcal{X} \leftarrow \mathcal{R}_{\mathcal{X}}.\text{NextStrategy}(\mathbf{m}^{(t)})$ 
3    $\mathbf{y}^{(t)} \in \mathcal{Y} \leftarrow \mathcal{R}_{\mathcal{Y}}.\text{NextStrategy}(\mathbf{m}^{(t)})$ 
4    $\mathbf{m}_{\Delta^2}^{(t)} \leftarrow \begin{pmatrix} \langle \mathbf{m}^{(t)}, \mathbf{x}^{(t)} \rangle \\ \langle \mathbf{m}^{(t)}, \mathbf{y}^{(t)} \rangle \end{pmatrix}$ 
5    $\boldsymbol{\lambda}^{(t)} := (\lambda_1^{(t)}, \lambda_2^{(t)}) \in \Delta^2 \leftarrow \mathcal{R}_{\Delta^2}.\text{NextStrategy}(\mathbf{m}_{\Delta^2}^{(t)})$ 
6   return  $\lambda_1^{(t)} \mathbf{x}^{(t)} + \lambda_2^{(t)} \mathbf{y}^{(t)} \in \text{co}\{\mathcal{X}, \mathcal{Y}\}$ 
7 function ObserveUtility( $\mathbf{u}^{(t)}$ )
8    $\mathcal{R}_{\mathcal{X}}.\text{ObserveUtility}(\mathbf{u}^{(t)})$ 
9    $\mathcal{R}_{\mathcal{Y}}.\text{ObserveUtility}(\mathbf{u}^{(t)})$ 
10   $\mathcal{R}_{\Delta^2}.\text{ObserveUtility}\left(\mathbf{u}_{\Delta^2}^{(t)} := \begin{pmatrix} \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \\ \langle \mathbf{u}^{(t)}, \mathbf{y}^{(t)} \rangle \end{pmatrix}\right)$ 

```



Similarly, the no-regret algorithm \mathcal{R}_{Δ^2} outputs strategies $\boldsymbol{\lambda}^{(t)} = (\lambda_1^{(t)}, \lambda_2^{(t)})$ at all times t and observes utility vectors $\mathbf{u}_{\Delta^2}^{(t)}$, defined in (4.4), at all times t . So, the regret cumulated by \mathcal{R}_{Δ^2} compared to any $\hat{\boldsymbol{\lambda}} \in \Delta^2$ is given by

$$\begin{aligned} \text{Reg}_{\Delta^2}^{(T)}(\hat{\boldsymbol{\lambda}}) &:= \sum_{t=1}^T \langle \mathbf{u}_{\Delta^2}^{(t)}, \hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}^{(t)} \rangle \\ &= \left(\sum_{t=1}^T \hat{\lambda}_1 \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle + \hat{\lambda}_2 \langle \mathbf{u}^{(t)}, \mathbf{y}^{(t)} \rangle \right) - \left(\sum_{t=1}^T \lambda_1^{(t)} \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle - \lambda_2^{(t)} \langle \mathbf{u}^{(t)}, \mathbf{y}^{(t)} \rangle \right). \end{aligned}$$

The following theorem connects these three quantities, showing that when all of them are sublinear, Algorithm 4.2 indeed yields a no-regret algorithm.

Theorem 4.2. The regret circuit of Algorithm 4.2 provides a no-regret algorithm for the convex hull $\text{co}\{\mathcal{X}, \mathcal{Y}\}$ whose regret $\text{Reg}_{\text{co}\{\mathcal{X}, \mathcal{Y}\}}^{(T)}$ up to any time T , is related to the regrets $\text{Reg}_{\mathcal{X}}^{(T)}$, $\text{Reg}_{\mathcal{Y}}^{(T)}$, and $\text{Reg}_{\Delta^2}^{(T)}$ accumulated by $\mathcal{R}_{\mathcal{X}}$, $\mathcal{R}_{\mathcal{Y}}$, and \mathcal{R}_{Δ^2} , respectively, according to

- i. for all comparators $\hat{\mathbf{x}} \in \mathcal{X}$, $\hat{\mathbf{y}} \in \mathcal{Y}$, and $(\hat{\lambda}_1, \hat{\lambda}_2) \in \Delta^2$,

$$\text{Reg}_{\text{co}\{\mathcal{X}, \mathcal{Y}\}}^{(T)}(\hat{\lambda}_1 \hat{\mathbf{x}} + \hat{\lambda}_2 \hat{\mathbf{y}}) = \text{Reg}_{\Delta^2}^{(T)}(\hat{\lambda}_1, \hat{\lambda}_2) + \hat{\lambda}_1 \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) + \hat{\lambda}_2 \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}),$$

- ii. $\text{Reg}_{\text{co}\{\mathcal{X}, \mathcal{Y}\}}^{(T)} \leq \text{Reg}_{\Delta^2}^{(T)} + \max\{\text{Reg}_{\mathcal{X}}^{(T)}, \text{Reg}_{\mathcal{Y}}^{(T)}\}$.

So, in particular, $\text{Reg}_{\text{co}\{\mathcal{X}, \mathcal{Y}\}}^{(T)}$ grows sublinearly in T since $\text{Reg}_{\mathcal{X}}^{(T)}$, $\text{Reg}_{\mathcal{Y}}^{(T)}$, and $\text{Reg}_{\Delta^2}^{(T)}$ are sublinear by the hypothesis that $\mathcal{R}_{\mathcal{X}}$, $\mathcal{R}_{\mathcal{Y}}$, and \mathcal{R}_{Δ^2} are no-regret algorithms.

Proof. We start by proving the first point. Starting from the definition of regret, we have

$$\begin{aligned} \text{Reg}_{\text{co}\{\mathcal{X}, \mathcal{Y}\}}^{(T)}(\hat{\lambda}_1 \hat{\mathbf{x}} + \hat{\lambda}_2 \hat{\mathbf{y}}) &:= \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\lambda}_1 \hat{\mathbf{x}} + \hat{\lambda}_2 \hat{\mathbf{y}} \rangle - \langle \mathbf{u}^{(t)}, \lambda_1^{(t)} \mathbf{x}^{(t)} + \lambda_2^{(t)} \mathbf{y}^{(t)} \rangle \\ &= \hat{\lambda}_1 \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\mathbf{x}} \rangle + \hat{\lambda}_2 \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\mathbf{y}} \rangle - \left(\sum_{t=1}^T \lambda_1^{(t)} \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle + \lambda_2^{(t)} \langle \mathbf{u}^{(t)}, \mathbf{y}^{(t)} \rangle \right) \end{aligned} \quad (4.7)$$

for all comparators $\hat{\mathbf{x}} \in \mathcal{X}$, $\hat{\mathbf{y}} \in \mathcal{Y}$, and $(\hat{\lambda}_1, \hat{\lambda}_2) \in \Delta^2$. From (4.5) and (4.6) we can relate the first two sums to the regrets accumulated by $\mathcal{R}_{\mathcal{X}}$ and $\mathcal{R}_{\mathcal{Y}}$, respectively, according to

$$\sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\mathbf{x}} \rangle = \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) + \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle, \quad (4.8)$$

$$\sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\mathbf{y}} \rangle = \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}) + \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \mathbf{y}^{(t)} \rangle. \quad (4.9)$$

Plugging Equations (4.8) and (4.9) into (4.7), we obtain

$$\begin{aligned} \text{Reg}_{\text{co}\{\mathcal{X}, \mathcal{Y}\}}^{(T)}(\hat{\lambda}_1 \hat{\mathbf{x}} + \hat{\lambda}_2 \hat{\mathbf{y}}) &= \hat{\lambda}_1 \left(\text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) + \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \right) + \hat{\lambda}_2 \left(\text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}) + \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \mathbf{y}^{(t)} \rangle \right) \\ &\quad - \left(\sum_{t=1}^T \lambda_1^{(t)} \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle + \lambda_2^{(t)} \langle \mathbf{u}^{(t)}, \mathbf{y}^{(t)} \rangle \right) \\ &= \hat{\lambda}_1 \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) + \hat{\lambda}_2 \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}) + \text{Reg}_{\Delta^2}^{(T)}(\hat{\lambda}_1, \hat{\lambda}_2). \end{aligned} \quad (4.10)$$

This concludes the proof of the first point. To show the second point, it is enough to note that for any $(\hat{\lambda}_1, \hat{\lambda}_2) \in \Delta^2$,

$$\hat{\lambda}_1 \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) + \hat{\lambda}_2 \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}) \leq \max \left\{ \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}), \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}) \right\}. \quad (4.11)$$

Hence,

$$\begin{aligned} \text{Reg}^{(T)} &:= \max_{\substack{(\hat{\lambda}_1, \hat{\lambda}_2) \in \Delta^2 \\ \hat{\mathbf{x}} \in \mathcal{X}, \hat{\mathbf{y}} \in \mathcal{Y}}} \text{Reg}^{(T)}(\hat{\lambda}_1 \hat{\mathbf{x}} + \hat{\lambda}_2 \hat{\mathbf{y}}) \\ &= \max_{\substack{(\hat{\lambda}_1, \hat{\lambda}_2) \in \Delta^2 \\ \hat{\mathbf{x}} \in \mathcal{X}, \hat{\mathbf{y}} \in \mathcal{Y}}} \left\{ \text{Reg}_{\Delta^2}^{(T)}(\hat{\lambda}_1, \hat{\lambda}_2) + \hat{\lambda}_1 \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) + \hat{\lambda}_2 \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}) \right\} \quad (\text{From (4.10)}) \\ &\leq \max_{\substack{(\hat{\lambda}_1, \hat{\lambda}_2) \in \Delta^2 \\ \hat{\mathbf{x}} \in \mathcal{X}, \hat{\mathbf{y}} \in \mathcal{Y}}} \left\{ \text{Reg}_{\Delta^2}^{(T)}(\hat{\lambda}_1, \hat{\lambda}_2) + \max \left\{ \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}), \text{Reg}_{\mathcal{Y}}^{(T)}(\hat{\mathbf{y}}) \right\} \right\} \quad (\text{From (4.11)}) \end{aligned}$$

$$= \text{Reg}_{\Delta^2}^{(T)} + \max\{\text{Reg}_{\mathcal{X}}^{(T)}, \text{Reg}_{\mathcal{Y}}^{(T)}\}$$

as we wanted to prove. □

Extension to multiple sets Of course, the construction shown in [Algorithm 4.2](#) can be readily extended to handle the convex hull $\text{co}\{\mathcal{X}_1, \dots, \mathcal{X}_n\}$ of n sets as follows.

- At each time instant t , we let m no-regret algorithms $\mathcal{R}_{\mathcal{X}_1}, \dots, \mathcal{R}_{\mathcal{X}_m}$ (for $\mathcal{X}_1, \dots, \mathcal{X}_m$, respectively) output their next strategies $\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_m^{(t)}$. Those strategies are combined according to the convex combination coefficients $\boldsymbol{\lambda}^{(t)}$ output by the additional no-regret algorithm \mathcal{R}_{Δ^m} for the m -simplex Δ^m to form the convex combination strategy $\lambda_1^{(t)} \mathbf{x}_1^{(t)} + \dots + \lambda_m^{(t)} \mathbf{x}_m^{(t)} \in \text{co}\{\mathcal{X}_1, \dots, \mathcal{X}_m\}$.
- The utility vector $\mathbf{u}^{(t)}$ is fed into all the no-external-regret algorithms $\mathcal{R}_{\mathcal{X}_i}$ ($i = 1, \dots, m$). Then, the utility vector $\mathbf{u}_{\Delta^m}^{(t)}$, defined as

$$\mathbf{u}_{\Delta^m}^{(t)} = \begin{pmatrix} \langle \mathbf{u}^{(t)}, \mathbf{x}_1^{(t)} \rangle \\ \vdots \\ \langle \mathbf{u}^{(t)}, \mathbf{x}_m^{(t)} \rangle \end{pmatrix},$$

is input into a no-regret algorithm for the m -simplex Δ^m .

With the same argument used earlier, it follows that the regret bound in that case is

$$\text{Reg}_{\text{co}\{\mathcal{X}_1, \dots, \mathcal{X}_m\}}^{(T)} \leq \text{Reg}_{\Delta^m}^{(T)} + \max\{\text{Reg}_{\mathcal{X}_1}^T, \dots, \text{Reg}_{\mathcal{X}_m}^T\}.$$

4.2.4 Affine transformation and Minkowski sum

Let $H : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}, \mathbf{x} \mapsto \mathbf{A}\mathbf{x} + \mathbf{b}$ be an affine map between two Euclidean spaces, and let $\mathcal{X} \subseteq \mathbb{R}^d$ be a strategy set for which a no-regret algorithm $\mathcal{R}_{\mathcal{X}}$ is given. In this section, we will show how $\mathcal{R}_{\mathcal{X}}$ can be adapted into a no-regret algorithm for the strategy set

$$H(\mathcal{X}) := \{\mathbf{A}\mathbf{x} + \mathbf{b} : \mathbf{x} \in \mathcal{X}\} \subseteq \mathbb{R}^{d'},$$

the image of \mathcal{X} under the affine map H . In this case we use this rather natural procedure:

- Given a prediction vector $\mathbf{m}^{(t)} \in \mathbb{R}^d$, we feed $\mathbf{A}^\top \mathbf{m}^{(t)}$ to $\mathcal{R}_{\mathcal{X}}$, receive its next strategy $\mathbf{x}^{(t)} \in \mathcal{X}$, and output $H(\mathbf{x}^{(t)}) = \mathbf{A}\mathbf{x}^{(t)} + \mathbf{b}$.
- Given a utility vector $\mathbf{u}^{(t)} \in \mathbb{R}^d$, we feed $\mathbf{A}^\top \mathbf{u}^{(t)}$ to $\mathcal{R}_{\mathcal{X}}$.

The construction is summarized by the circuit in [Figure 4.1](#).

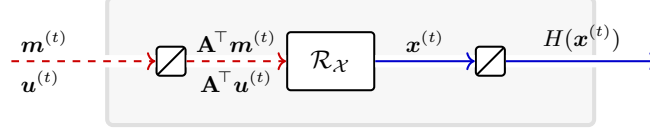


Figure 4.1: Regret circuit for the image $H(\mathcal{X})$ of \mathcal{X} under the affine transformation H .

Regret analysis We now analyze the regret circuit we have just described, seeking to relate the regret

$$\text{Reg}_{H(\mathcal{X})}^{(T)}(\hat{z}) := \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{z} - H(\mathbf{x}^{(t)}) \rangle$$

accumulated by the circuit relative to a generic comparator $\hat{z} \in H(\mathcal{X})$, to that accumulated by $\mathcal{R}_{\mathcal{X}}$. By construction, at all times t the no-regret algorithm $\mathcal{R}_{\mathcal{X}}$ produces iterates $\mathbf{x}^{(t)} \in \mathcal{X}$ and receives the utility vector $\mathbf{A}^\top \mathbf{u}^{(t)}$. Hence, the regret accumulated by $\mathcal{R}_{\mathcal{X}}$ up to any time T is given by

$$\begin{aligned} \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) &:= \sum_{t=1}^T \langle \mathbf{A}^\top \mathbf{u}^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle = \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \mathbf{A}\hat{\mathbf{x}} - \mathbf{A}\mathbf{x}^{(t)} \rangle \\ &= \sum_{t=1}^T \langle \mathbf{u}^{(t)}, (\mathbf{A}\hat{\mathbf{x}} + \mathbf{b}) - (\mathbf{A}\mathbf{x}^{(t)} + \mathbf{b}) \rangle = \text{Reg}_{H(\mathcal{X})}^{(T)}(H(\hat{\mathbf{x}})). \end{aligned}$$

Since every $\hat{\mathbf{y}} \in H(\mathcal{X})$ can be written as $\hat{\mathbf{y}} = H(\hat{\mathbf{x}})$ for some $\hat{\mathbf{x}} \in \mathcal{X}$, we can further conclude that $\text{Reg}_{\mathcal{X}}^{(T)} = \text{Reg}_{H(\mathcal{X})}^{(T)}$. In conclusion, we have shown the following.

Theorem 4.3. The regret circuit depicted in Figure 4.1 provides a no-regret algorithm for the image $H(\mathcal{X})$ of \mathcal{X} under affine transformation H , whose regret $\text{Reg}_{H(\mathcal{X})}^{(T)}$ accumulated up to any time T is related to the regret $\text{Reg}_{\mathcal{X}}^{(T)}$ accumulated by $\mathcal{R}_{\mathcal{X}}$ according to

- i. $\text{Reg}_{H(\mathcal{X})}^{(T)}(H(\hat{\mathbf{x}})) = \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}})$ for all comparators $\hat{\mathbf{x}} \in \mathcal{X}$; and
- ii. $\text{Reg}_{H(\mathcal{X})}^{(T)} = \text{Reg}_{\mathcal{X}}^{(T)}$.

So, in particular, $\text{Reg}_{H(\mathcal{X})}^{(T)}$ grows sublinearly in T since $\text{Reg}_{\mathcal{X}}^{(T)}$ is sublinear by the hypothesis that $\mathcal{R}_{\mathcal{X}}$ is a no-regret algorithms.

Application: Minkowski sum As a straightforward application, the above construction can be employed to construct a no-regret algorithm for the Minkowski sum $\mathcal{X} + \mathcal{Y} := \{\mathbf{x} + \mathbf{y} : \mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}\}$ of two sets. Indeed, note that $\mathcal{X} + \mathcal{Y} = \sigma(\mathcal{X} \times \mathcal{Y})$, where $\sigma : (\mathbf{x}, \mathbf{y}) \mapsto \mathbf{x} + \mathbf{y}$ is a linear map. Hence, we can combine the construction for affine transformations presented in this

section, together with the construction of the Cartesian product (Section 4.2.2, Algorithm 4.1). The resulting regret circuit is in Figure 4.2.

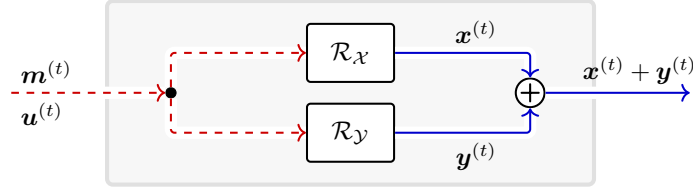


Figure 4.2: Regret circuit for the Minkowski sum $\mathcal{X} + \mathcal{Y}$.

4.2.5 Scaled extension

In this section we introduce the last regret circuit that will be used in the rest of the dissertation. It refers to a particular convexity-preserving operation, *scaled extension* (defined below), which was introduced in (Farina, Ling, Fang, and Sandholm, 2019b) to capture certain correlated strategy spaces. We will make extensive use of the scaled extension operation in Chapters 9 and 10.

Definition We now introduce the definition of the scaled extension operation.

Definition 4.1. Let \mathcal{X} and \mathcal{Y} be nonempty sets, and let $h : \mathcal{X} \rightarrow \mathbb{R}_+$ be a nonnegative affine real function. The *scaled extension* of \mathcal{X} with \mathcal{Y} via h is defined as the set

$$\mathcal{X} \overset{h}{\triangleleft} \mathcal{Y} := \{(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in \mathcal{X}, \mathbf{y} \in h(\mathbf{x})\mathcal{Y}\}.$$

As we show next, the operation above preserves convexity, non-emptiness, and compactness of the sets.

Lemma 4.1. Let \mathcal{X}, \mathcal{Y} and h be as in Definition 4.1. If \mathcal{X} and \mathcal{Y} are nonempty, compact, or convex, then $\mathcal{X} \overset{h}{\triangleleft} \mathcal{Y}$ is nonempty, compact, and convex, respectively.

Proof. Let $\mathcal{Z} := \mathcal{X} \overset{h}{\triangleleft} \mathcal{Y}$. We break the proof into three parts:

- *Non-emptiness.* Since \mathcal{X} and \mathcal{Y} are nonempty by hypothesis, let \mathbf{x} and \mathbf{y} be arbitrary points in \mathcal{X} and \mathcal{Y} . The element $(\mathbf{x}, h(\mathbf{x})\mathbf{y})$ belongs to \mathcal{Z} and therefore \mathcal{Z} is nonempty.
- *Compactness.* We now prove that \mathcal{Z} is a compact set via the Heine-Borel theorem, by proving that it is bounded and closed. First, we argue that \mathcal{Z} is bounded. Indeed, note that h is affine and therefore continuous, and since \mathcal{X} is compact we conclude by Weierstrass' theorem that $h^* := \max_{\mathbf{x} \in \mathcal{X}} h(\mathbf{x})$ exists and is finite. Hence, any $(\mathbf{x}, h(\mathbf{x})\mathbf{y})$ satisfies

$$\|(\mathbf{x}, h(\mathbf{x})\mathbf{y})\|_1 \leq \|\mathbf{x}\|_1 + h^*\|\mathbf{y}\|_1 < \infty,$$

since both \mathcal{X} and \mathcal{Y} are compact and hence bounded, showing that \mathcal{Z} is bounded. Now, we argue that \mathcal{Z} is (sequentially) closed. Indeed, let $\{z_i\} \rightarrow \bar{z}$ be a convergent sequence such that $z_i \in \mathcal{Z}$ for $i = 1, 2, \dots$; we will prove that $\bar{z} \in \mathcal{Z}$. By definition of \mathcal{Z} , for all i it must be $z_i = (\mathbf{x}_i, h(\mathbf{x}_i)\mathbf{y}_i)$ for some $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{X} \times \mathcal{Y}$. Since $\{z_i\}$ converges, then the sequences $\{\mathbf{x}_i\}$ and $\{h(\mathbf{x}_i)\mathbf{y}_i\}$ must also converge. Let $\bar{\mathbf{x}} := \lim \mathbf{x}_i$; by closedness of \mathcal{X} , it must be $\bar{\mathbf{x}} \in \mathcal{X}$. Furthermore, by continuity of h , $\lim h(\mathbf{x}_i) = h(\bar{\mathbf{x}})$. Now, using the (sequential) compactness of \mathcal{Y} , we can assume without loss of generality that $\{\mathbf{y}_i\}$ converges^a; let $\bar{\mathbf{y}} := \lim \mathbf{y}_i$. By the usual properties of limits, $\bar{z} = \lim z_i = \lim (\mathbf{x}_i, h(\mathbf{x}_i)\mathbf{y}_i) = (\bar{\mathbf{x}}, h(\bar{\mathbf{x}})\bar{\mathbf{y}})$, and since $\bar{\mathbf{x}} \in \mathcal{X}$, $\bar{\mathbf{y}} \in \mathcal{Y}$ we have $\bar{z} \in \mathcal{Z}$ and \mathcal{Z} is sequentially closed.

- *Convexity.* Let z, z' be any two points in \mathcal{Z} , and let $\lambda \in (0, 1)$. By definition of \mathcal{Z} , there must exist $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ and $\mathbf{y}, \mathbf{y}' \in \mathcal{Y}$ such that $z = (\mathbf{x}, h(\mathbf{x})\mathbf{y})$ and $z' = (\mathbf{x}', h(\mathbf{x}')\mathbf{y}')$. If $h(\mathbf{x}) = h(\mathbf{x}') = 0$, then the affinity of h implies $h(\lambda\mathbf{x} + (1-\lambda)\mathbf{x}') = 0$ and therefore

$$\begin{aligned} \lambda z + (1-\lambda)z' &= (\lambda\mathbf{x} + (1-\lambda)\mathbf{x}', \mathbf{0}) \\ &= (\lambda\mathbf{x} + (1-\lambda)\mathbf{x}', h(\lambda\mathbf{x} + (1-\lambda)\mathbf{x}')\mathbf{y}) \in \mathcal{Z}. \end{aligned}$$

On the other hand, if at least one of $h(\mathbf{x})$ and $h(\mathbf{x}')$ is strictly positive, then the quantity $s := \lambda h(\mathbf{x}) + (1-\lambda)h(\mathbf{x}')$ is also strictly positive and we can write

$$\begin{aligned} \lambda z + (1-\lambda)z' &= (\lambda\mathbf{x} + (1-\lambda)\mathbf{x}', \lambda h(\mathbf{x})\mathbf{y} + (1-\lambda)h(\mathbf{x}')\mathbf{y}') \\ &= (\lambda\mathbf{x} + (1-\lambda)\mathbf{x}', s \left[\frac{\lambda h(\mathbf{x})}{s}\mathbf{y} + \frac{(1-\lambda)h(\mathbf{x}')}{s}\mathbf{y}' \right]) \\ &= (\lambda\mathbf{x} + (1-\lambda)\mathbf{x}', h(\lambda\mathbf{x} + (1-\lambda)\mathbf{x}') \left[\frac{\lambda h(\mathbf{x})}{s}\mathbf{y} + \frac{(1-\lambda)h(\mathbf{x}')}{s}\mathbf{y}' \right]), \end{aligned}$$

where the last equality follows from the definition of s together with the affinity of h . The quantity in the square bracket is a convex combination of the vectors \mathbf{y} and \mathbf{y}' , and it is therefore equal to some $\mathbf{y}'' \in \mathcal{Y}$ by convexity of \mathcal{Y} . Hence,

$$\lambda z + (1-\lambda)z' = (\lambda\mathbf{x} + (1-\lambda)\mathbf{x}', h(\lambda\mathbf{x} + (1-\lambda)\mathbf{x}')\mathbf{y}'') \in \mathcal{Z},$$

as we wanted to show. □

^aOr else, extract a convergent subsequence.

Regret circuit We now show that it is always possible to construct a no-regret-algorithm for $\mathcal{Z} = \mathcal{X} \triangleleft^h \mathcal{Y}$, where $h(\mathbf{x}) = \langle \mathbf{a}, \mathbf{x} \rangle + b$ is a nonnegative affine function, starting from no-regret

algorithms $\mathcal{R}_X, \mathcal{R}_Y$ for $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}^{d'}$, respectively. In particular, we consider the following algorithm.

- To output the next strategy, given prediction vector $\mathbf{m}^{(t)} := (\mathbf{m}_x^{(t)}, \mathbf{m}_y^{(t)}) \in \mathbb{R}^d \times \mathbb{R}^{d'}$, we feed vector $\mathbf{m}_y^{(t)}$ to \mathcal{R}_Y , and receive its next strategy $\mathbf{y}^{(t)} \in \mathcal{Y}$. Then, we construct the prediction vector

$$\tilde{\mathbf{m}}_x^{(t)} := \mathbf{m}_x^{(t)} + \langle \mathbf{m}_y^{(t)}, \mathbf{y}^{(t)} \rangle \mathbf{a},$$

feed it to \mathcal{R}_X , and receive its next strategy $\mathbf{x}^{(t)} \in \mathcal{X}$. Finally, we return the point $(\mathbf{x}^{(t)}, h(\mathbf{x}^{(t)})\mathbf{y}^{(t)}) \in \mathcal{Z}$.

- When at time t we receive the utility vector $\mathbf{u}^{(t)} := (\mathbf{u}_x^{(t)}, \mathbf{u}_y^{(t)}) \in \mathbb{R}^d \times \mathbb{R}^{d'}$, we forward $\mathbf{u}_y^{(t)}$ to \mathcal{R}_Y , and forward the utility vector

$$\tilde{\mathbf{u}}_x^{(t)} := \mathbf{u}_x^{(t)} + \langle \mathbf{u}_y^{(t)}, \mathbf{y}^{(t)} \rangle \mathbf{a} \tag{4.12}$$

to \mathcal{R}_X .

The regret circuit is summarized pictorially in Figure 4.3.

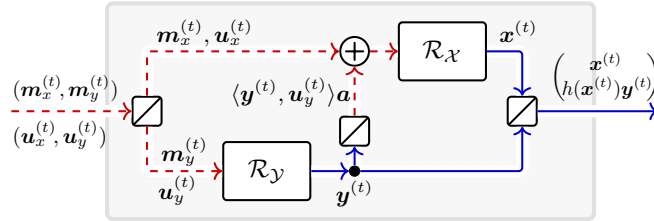


Figure 4.3: Regret circuit for the scaled extension $\mathcal{X} \stackrel{h}{\triangleleft} \mathcal{Y}$.

Regret analysis We now relate the regret accumulated by the construction we just described up to any time T compared to a generic point $(\hat{\mathbf{x}}, h(\hat{\mathbf{x}})\hat{\mathbf{y}}) \in \mathcal{Z}$, that is the quantity

$$\text{Reg}_{\mathcal{X} \stackrel{h}{\triangleleft} \mathcal{Y}}^{(T)}(\hat{\mathbf{x}}, h(\hat{\mathbf{x}})\hat{\mathbf{y}}) := \sum_{t=1}^T \left\langle \mathbf{u}^{(t)}, \begin{pmatrix} \hat{\mathbf{x}} \\ h(\hat{\mathbf{x}})\hat{\mathbf{y}} \end{pmatrix} - \begin{pmatrix} \mathbf{x}^{(t)} \\ h(\mathbf{x}^{(t)})\mathbf{y}^{(t)} \end{pmatrix} \right\rangle, \tag{4.13}$$

to the regrets $\text{Reg}_{\mathcal{X}}^{(T)}, \text{Reg}_{\mathcal{Y}}^{(T)}$ accumulated by \mathcal{R}_X and \mathcal{R}_Y respectively. By construction, at all times t the no-regret algorithm \mathcal{R}_Y outputs strategies $\mathbf{y}^{(t)} \in \mathcal{Y}$ and receives utilities $\mathbf{u}_y^{(t)}$. Hence, the regret accumulated by \mathcal{R}_Y up to time T compared to any $\hat{\mathbf{y}} \in \mathcal{Y}$ is

$$\text{Reg}_Y^{(T)}(\hat{\mathbf{y}}) := \sum_{t=1}^T \langle \mathbf{u}_Y^{(t)}, \hat{\mathbf{y}} - \mathbf{y}^{(t)} \rangle. \quad (4.14)$$

The no-regret algorithm \mathcal{R}_X outputs strategies $\mathbf{x}^{(t)} \in X$ and receives utilities $\tilde{\mathbf{u}}_X^{(t)}$ defined in (4.12); hence its regret compared to any $\hat{\mathbf{x}} \in X$ is

$$\text{Reg}_X^{(T)}(\hat{\mathbf{x}}) := \sum_{t=1}^T \langle \tilde{\mathbf{u}}_X^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle. \quad (4.15)$$

The connection between the three regret quantities defined above is given in the next theorem.

Theorem 4.4. The regret circuit of Figure 4.3 provides a no-regret algorithm for the scaled extension $X \overset{h}{\triangleleft} Y$ whose regret $\text{Reg}_{X \overset{h}{\triangleleft} Y}^{(T)}$ up to any time T , is related to the regrets $\text{Reg}_X^{(T)}$ and $\text{Reg}_Y^{(T)}$ accumulated by \mathcal{R}_X and \mathcal{R}_Y , respectively, according to

- i. $\text{Reg}_{X \overset{h}{\triangleleft} Y}^{(T)}(\hat{\mathbf{x}}, h(\hat{\mathbf{x}})\hat{\mathbf{y}}) = \text{Reg}_Y^{(T)}(\hat{\mathbf{y}}) + h(\hat{\mathbf{x}}) \text{Reg}_X^{(T)}(\hat{\mathbf{x}})$ for all $\hat{\mathbf{x}} \in X$, $\hat{\mathbf{y}} \in Y$; and
- ii. $\text{Reg}_{X \overset{h}{\triangleleft} Y}^{(T)} \leq \text{Reg}_Y^{(T)} + h^* \cdot \text{Reg}_X^{(T)}$, where $h^* := \max_{\mathbf{x} \in X} h(\mathbf{x})$.^[4.b]

So, in particular, $\text{Reg}_{X \overset{h}{\triangleleft} Y}^{(T)}$ grows sublinearly in T since $\text{Reg}_X^{(T)}$ and $\text{Reg}_Y^{(T)}$ are sublinear by the hypothesis that \mathcal{R}_X and \mathcal{R}_Y are no-regret algorithms.

Proof. By expanding the definition of $\tilde{\mathbf{u}}_X^{(t)}$ given in (4.12) in the definition (4.15) of regret for \mathcal{R}_X , we obtain

$$\begin{aligned} \text{Reg}_X^{(T)}(\hat{\mathbf{x}}) &= \left(\sum_{t=1}^T \langle \mathbf{u}_X^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle \right) + \left(\sum_{t=1}^T \langle \mathbf{u}_Y^{(t)}, \mathbf{y}^{(t)} \rangle \right) \langle \mathbf{a}, \hat{\mathbf{x}} \rangle - \sum_{t=1}^T \langle \mathbf{u}_Y^{(t)}, \mathbf{y}^{(t)} \rangle \cdot \langle \mathbf{a}, \mathbf{x}^{(t)} \rangle \\ &= \left(\sum_{t=1}^T \langle \mathbf{u}_X^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle \right) + \left(\sum_{t=1}^T \langle \mathbf{u}_Y^{(t)}, \mathbf{y}^{(t)} \rangle \right) (\langle \mathbf{a}, \hat{\mathbf{x}} \rangle + b) \\ &\quad - \sum_{t=1}^T \langle \mathbf{u}_Y^{(t)}, \mathbf{y}^{(t)} \rangle \cdot (\langle \mathbf{a}, \mathbf{x}^{(t)} \rangle + b) \\ &= \left(\sum_{t=1}^T \langle \mathbf{u}_X^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle \right) + \left(\sum_{t=1}^T \langle \mathbf{u}_Y^{(t)}, \mathbf{y}^{(t)} \rangle \right) h(\hat{\mathbf{x}}) - \sum_{t=1}^T \langle \mathbf{u}_Y^{(t)}, \mathbf{y}^{(t)} \rangle \cdot h(\mathbf{x}^{(t)}). \end{aligned}$$

Using (4.14), we can rewrite the middle sum as

^[4.b]Since h is affine and X is compact, h^* exists and is finite by Weierstrass' theorem.

$$\sum_{t=1}^T \langle \mathbf{u}_y^{(t)}, \mathbf{y}^{(t)} \rangle = -\text{Reg}_y^{(T)}(\hat{\mathbf{y}}) + \sum_{t=1}^T \langle \mathbf{u}_y^{(t)}, \hat{\mathbf{y}} \rangle,$$

and hence

$$\begin{aligned} \text{Reg}_{\mathcal{X}}^{(T)}(\hat{\mathbf{x}}) &= \left(\sum_{t=1}^T \langle \mathbf{u}_x^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle \right) + \left(-\text{Reg}_y^{(T)}(\hat{\mathbf{y}}) + \sum_{t=1}^T \langle \mathbf{u}_y^{(t)}, \hat{\mathbf{y}} \rangle \right) h(\hat{\mathbf{x}}) \\ &\quad - \sum_{t=1}^T \langle \mathbf{u}_y^{(t)}, \mathbf{y}^{(t)} \rangle \cdot h(\mathbf{x}^{(t)}) \\ &= -h(\hat{\mathbf{x}}) \text{Reg}_y^{(T)}(\hat{\mathbf{y}}) + \sum_{t=1}^T \left\langle \begin{pmatrix} \mathbf{u}_x^{(t)} \\ \mathbf{u}_y^{(t)} \end{pmatrix}, \begin{pmatrix} \hat{\mathbf{x}} \\ h(\hat{\mathbf{x}})\hat{\mathbf{y}} \end{pmatrix} - \begin{pmatrix} \mathbf{x}^{(t)} \\ h(\mathbf{x}^{(t)})\mathbf{y}^{(t)} \end{pmatrix} \right\rangle \\ &= -h(\hat{\mathbf{x}}) \text{Reg}_y^{(T)}(\hat{\mathbf{y}}) + \text{Reg}_{\mathcal{X}_h}^{(T)}(\hat{\mathbf{x}}, h(\hat{\mathbf{x}})\hat{\mathbf{y}}), \end{aligned}$$

where we used (4.13) in the last equality. Rearranging yields the result. \square

4.3 Predictive counterfactual regret minimization paradigm

The set of sequence-form strategies can be decomposed in a bottom-up fashion by using the convex hulls and Cartesian products operations. We illustrate this intuitively by means of a small example.

Example 4.1. Consider the small tree-form decision problem of Figure 4.4, which was originally introduced in Example 2.7. As our construction is bottom-up, we will denote with the symbol Q_v , with $v \in \mathcal{J} \cup \mathcal{K}$, the partial sequence-form strategy space corresponding to the subtree rooted at v .

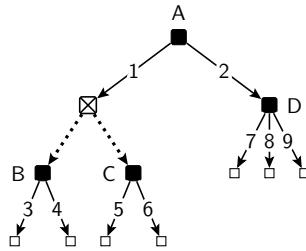


Figure 4.4: Sequential decision-making problem used in the example.

We can characterize the set of sequence-form strategies as follows:

- At the terminal decision nodes $j = B, C, D$, Q_j is a probability simplex. Specifically, $Q_B = Q_C = \Delta^2$ and $Q_D = \Delta^3$.
- At the (only) observation node k , a strategy for the subtree rooted at k must provide independent strategies for the subtrees rooted at B and C. So,

$$Q_k = Q_B \times Q_C$$

is the Cartesian product of the strategy spaces for the subtrees rooted B and C.

- At the decision node A, we need to first pick a probability distribution of play for the two actions (sequences 1 and 2). Let's call the probabilities assigned to those actions as λ_1 and λ_2 , respectively. Clearly, $(\lambda_1, \lambda_2) \in \Delta^2$. Once the probabilities of sequences 1 and 2 are chosen, strategies for the subtrees rooted at the observation node k and D. Since sequence-form strategies associate to each sequence σ the product of the probabilities of all actions on the path from the root of the TFDP to σ , the set of all valid sequence-form strategies for the subtree rooted in A (that is, the whole TFDP) is

$$\begin{aligned} Q_A &= \left\{ (\lambda_1, \lambda_2, \lambda_1 \mathbf{x}_k, \lambda_2 \mathbf{x}_D) : (\lambda_1, \lambda_2) \in \Delta^2, \mathbf{x}_k \in Q_k, \mathbf{x}_D \in Q_D \right\} \\ &= \text{co} \left\{ \left(\begin{array}{c} 1 \\ 0 \\ Q_k \\ \mathbf{0} \end{array} \right), \left(\begin{array}{c} 0 \\ 1 \\ \mathbf{0} \\ Q_{\neq D} \end{array} \right) \right\}. \end{aligned}$$

The above approach can be used in any TFDP. In particular, we always have that the sequence-form strategy space of a subtree rooted at an observation node is the Cartesian product of the sequence-form strategy spaces of the children subtrees. Furthermore, the sequence-form strategy space of a subtree rooted at a decision point is the convex hull of the sequence-form strategy spaces of the children subtrees, augmented with the indicators of the actions. These inductive rules are summarized in [Table 4.1](#).

By applying the regret circuits described in [Sections 4.2.2](#) and [4.2.3](#), we can then construct a no-regret algorithms for any sequence-form strategy space. The resulting no-regret algorithm is called CFR, whose pseudocode is given in [Algorithms 4.3](#) and [4.4](#). In a nutshell, CFR decomposes the problem of minimizing regret on the whole tree-form decision process into local regret minimization problems at each of the individual decision nodes $j \in \mathcal{J}$. Any no-regret algorithm \mathcal{R}_j for simplex domains can be used to solve the local regret minimization problems. Popular options are the regret matching (RM) algorithm, and the regret matching⁺ (RM⁺) algorithm discussed in [Section 3.3.2](#).

By combining the regret guarantees analyzed for the Cartesian product and convex hull regret

Algorithm 4.3: Predictive CFR (weakly-predictive no-external-regret algorithm for sequence-form strategy polytope)

Data: $\{\mathcal{R}_j : j \in \mathcal{J}\}$: no-regret algorithms for $\Delta^{\mathcal{A}_j}$; one for each decision node $j \in \mathcal{J}$.

```

1 function NextStrategy( $\mathbf{m}^{(t)}$ )
  [ $\triangleright$  Compute the expected prediction for each subtree rooted at each  $v \in \mathcal{J} \cup \mathcal{K}$ ]
2   $M^{(t)} \leftarrow$  empty dictionary          [ $\triangleright$  Maps keys  $\mathcal{J} \cup \mathcal{K} \cup \{\perp\}$  to real numbers]
3   $M^{(t)}[\perp] \leftarrow 0$ 
4  for each node in the tree  $v \in \mathcal{J} \cup \mathcal{K}$  in bottom-up order in the TFDP do
5    if  $v \in \mathcal{J}$  then
6      Let  $j = v$ 
7      [ $\triangleright$  At each decision node  $j \in \mathcal{J}$ , we now construct the counterfactual utility vector  $\mathbf{m}_j^{(t)}$ ]
8       $\mathbf{m}_j^{(t)} \leftarrow \mathbf{0} \in \mathbb{R}^{\mathcal{A}_j}$ 
9      for each action  $a \in \mathcal{A}_j$  do
10       |  $\mathbf{m}_j^{(t)}[a] \leftarrow \mathbf{m}^{(t)}[ja] + M^{(t)}[\rho(j, a)]$ 
11       |-----|
12       [ $\triangleright$  Query each of the  $\mathcal{R}_j$  for their strategy local at each decision node]
13        $\mathbf{b}_j^{(t)} \in \Delta^{\mathcal{A}_j} \leftarrow \mathcal{R}_j.\text{NextStrategy}(\mathbf{m}_j^{(t)})$ 
14       |-----|
15       [ $\triangleright$  Update expected values using the new local strategy]
16        $M^{(t)}[j] \leftarrow \sum_{a \in \mathcal{A}_j} \mathbf{b}_j^{(t)}[a] \cdot \mathbf{m}_j^{(t)}[a]$ 
17     else
18       Let  $k = v$ 
19        $M^{(t)}[k] \leftarrow \sum_{s \in \mathcal{S}_k} M^{(t)}[\rho(k, s)]$ 
20
21   [ $\triangleright$  Construct the sequence-form representation of the strategy that plays according to the
22   distribution  $\mathbf{b}_j^{(t)}$  at each decision node  $j \in \mathcal{J}$ ]
23    $\mathbf{x}^{(t)} = \mathbf{0} \in \mathbb{R}^\Sigma$ 
24   for each decision node  $j \in \mathcal{J}$  in top-down order in the TFDP do
25     for each action  $a \in \mathcal{A}_j$  do
26       if  $p_j = \emptyset$  then
27         |  $\mathbf{x}^{(t)}[ja] \leftarrow \mathbf{b}_j^{(t)}[a]$ 
28       else
29         |  $\mathbf{x}^{(t)}[ja] \leftarrow \mathbf{x}^{(t)}[p_j] \cdot \mathbf{b}_j^{(t)}[a]$ 
30   return  $\mathbf{x}^{(t)}$ 

```

(The algorithm continues in [Algorithm 4.4](#))

Algorithm 4.4: (Continued) Predictive CFR (weakly-predictive no-external-regret algorithm for sequence-form strategy polytope)

```

1 function ObserveUtility( $\mathbf{u}^{(t)} \in \mathbb{R}^\Sigma$ )
   | [ $\triangleright$  Compute the expected utility for each subtree rooted at each  $v \in \mathcal{J} \cup \mathcal{K}$ ]
2    $V^{(t)} \leftarrow$  empty dictionary;           [ $\triangleright$  Maps keys  $\mathcal{J} \cup \mathcal{K} \cup \{\perp\}$  to real numbers]
3    $V^{(t)}[\perp] \leftarrow 0$ ;
4   for each node in the tree  $v \in \mathcal{J} \cup \mathcal{K}$  in bottom-up order in the TFDP do
5     | if  $v \in \mathcal{J}$  then
6     |   | Let  $j = v$ ;
7     |   |  $V^{(t)}[j] \leftarrow \sum_{a \in \mathcal{A}_j} \mathbf{b}_j^{(t)}[a] \cdot (\mathbf{u}^{(t)}[ja] + V^{(t)}[\rho(j, a)]);$ 
8     |   | else
9     |   |   | Let  $k = v$ ;
10    |   |   |  $V^{(t)}[k] \leftarrow \sum_{s \in \mathcal{S}_k} V^{(t)}[\rho(k, s)];$ 
11    |   | [ $\triangleright$  At each decision node  $j \in \mathcal{J}$ , we now construct the counterfactual utility vector  $\mathbf{u}_j^{(t)}$ ]
12    |   | for each decision node  $j \in \mathcal{J}$  do
13    |   |   |  $\mathbf{u}_j^{(t)} \leftarrow \mathbf{0} \in \mathbb{R}^{\mathcal{A}_j}$ ;
14    |   |   | for each action  $a \in \mathcal{A}_j$  do
15    |   |   |   |  $\mathbf{u}_j^{(t)}[a] \leftarrow \mathbf{u}^{(t)}[ja] + V^{(t)}[\rho(j, a)];$ 
16    |   |   |  $\mathcal{R}_j.$ ObserveUtility( $\mathbf{u}_j^{(t)}$ );

```

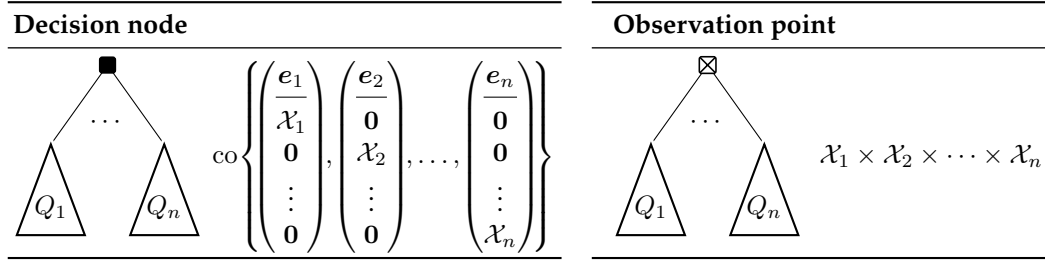


Table 4.1: Bottom-up construction rules for sequence-form strategy spaces. e_i denotes the i -th indicator vector, that is, the vector whose entries are all 0 except for the entry in position i , which is set to 1.

circuits (Theorems 4.1 and 4.2), it is immediate that the regret cumulated by the CFR algorithm satisfies the following bound.

Proposition 4.1. Let $\text{Reg}_j^{(T)}$ ($j \in \mathcal{J}$) denote the regret cumulated up to time T by each of the no-regret algorithms \mathcal{R}_j . Furthermore, let $\mathbf{b}_j \in \Delta^{A_j}$ be arbitrary strategies for the local decision nodes j , and let $\hat{\mathbf{x}} \in \mathcal{Q}$ be the sequence-form strategy corresponding to the behavioral strategy $\{\mathbf{b}_j : j \in \mathcal{J}\}$. Then, the regret $\text{Reg}^{(T)}$ cumulated by CFR (Algorithms 4.3 and 4.4) up to time T satisfies the equality

$$\text{Reg}^{(T)}(\hat{\mathbf{x}}) = \sum_{j \in \mathcal{J}} \hat{\mathbf{x}}[p_j] \cdot \text{Reg}_j^{(T)}(\hat{\mathbf{b}}_j).$$

In particular,

$$\text{Reg}^{(T)} \leq \sum_{j \in \mathcal{J}} \max\{0, \text{Reg}_j^{(T)}\}.$$

4.3.1 Predictive RM (PRM) and predictive RM^+ (PRM^+) algorithms

As mentioned in Section 3.3.2, variants of the RM algorithm such as RM^+ and Discounted RM have emerged as the preferred no-regret-algorithms for use in game theoretic settings, due to their strong performance in practice and their lack of hyperparameters. However, those algorithms do not support prediction, and therefore do not lead to an interesting predictive variant of CFR.

Motivated by the desire to combine the benefits of RM-based methods with those afforded by predictivity, in this section we introduce the first predictive versions of RM and RM^+ . The resulting algorithms, coined respectively *predictive RM (PRM)* and *predictive RM^+ (PRM^+)*, are given in Algorithms 4.5 and 4.6. The algorithms coincide with RM and RM^+ given in Algorithms 3.2

Algorithm 4.5: Predictive regret matching (PRM)	Algorithm 4.6: Predictive regret matching ⁺ (PRM ⁺)
<pre> 1 $\mathbf{r}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^d, \mathbf{x}^{(0)} \leftarrow \mathbf{1}/d \in \Delta^d$ 2 function NextStrategy($\mathbf{m}^{(t)}$) [\triangleright Set $\mathbf{m}^{(t)} = \mathbf{0}$ for non-predictive version] 3 $\boldsymbol{\theta}^{(t)} \leftarrow [\mathbf{r}^{(t-1)} + \mathbf{m}^{(t)} - \langle \mathbf{m}^{(t)}, \mathbf{x}^{(t-1)} \rangle \mathbf{1}]^+$ 4 if $\boldsymbol{\theta}^{(t)} \neq \mathbf{0}$ return $\mathbf{x}^{(t)} \leftarrow \boldsymbol{\theta}^{(t)} / \ \boldsymbol{\theta}^{(t)}\ _1$ 5 else return $\mathbf{x}^{(t)} \leftarrow$ any point in Δ^d 6 function ObserveUtility($\mathbf{u}^{(t)}$) 7 $\mathbf{r}^{(t)} \leftarrow \mathbf{r}^{(t-1)} + \mathbf{u}^{(t)} - \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \mathbf{1}$ </pre>	<pre> 1 $\mathbf{z}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^d, \mathbf{x}^{(0)} \leftarrow \mathbf{1}/d \in \Delta^d$ 2 function NextStrategy($\mathbf{m}^{(t)}$) [\triangleright Set $\mathbf{m}^{(t)} = \mathbf{0}$ for non-predictive version] 3 $\boldsymbol{\theta}^{(t)} \leftarrow [\mathbf{z}^{(t-1)} + \mathbf{m}^{(t)} - \langle \mathbf{m}^{(t)}, \mathbf{x}^{(t-1)} \rangle \mathbf{1}]^+$ 4 if $\boldsymbol{\theta}^{(t)} \neq \mathbf{0}$ return $\mathbf{x}^{(t)} \leftarrow \boldsymbol{\theta}^{(t)} / \ \boldsymbol{\theta}^{(t)}\ _1$ 5 else return $\mathbf{x}^{(t)} \leftarrow$ any point in Δ^d 6 function ObserveUtility($\mathbf{u}^{(t)}$) 7 $\mathbf{z}^{(t)} \leftarrow [\mathbf{z}^{(t-1)} + \mathbf{u}^{(t)} - \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \mathbf{1}]^+$ </pre>

and 3.3 when the prediction vector $\mathbf{m}^{(t)}$ is set to $\mathbf{0}$.

Theorem 4.5 (Farina, Kroer, and Sandholm, 2021b). PRM and PRM⁺ are no-external-regret algorithms for the domain Δ^d . For all $\hat{\mathbf{x}} \in \Delta^d$ and all times T satisfy the weakly-predictive regret bound

$$\begin{aligned}
R^T(\hat{\mathbf{x}}) &\leq \frac{1}{\eta} + \eta \sum_{t=1}^T \left\| \mathbf{u}^{(t)} - \mathbf{m}^{(t)} - \langle \mathbf{u}^{(t)} - \mathbf{m}^{(t)}, \mathbf{x}^{(t)} \rangle \mathbf{1} \right\|_2^2 \\
&\leq \frac{1}{\eta} + 2\eta(d+1) \sum_{t=1}^T \left\| \mathbf{u}^{(t)} - \mathbf{m}^{(t)} \right\|_2^2.
\end{aligned}$$

4.3.2 Experimental evaluation

We empirically investigate the instantiation of the predictive CFR algorithm described in Section 4.3 together with the weakly-predictive no-external-regret algorithm for probability simplexes PRM⁺ described in Section 4.3.1.

In the experiments, we apply two heuristics that usually lead to better practical performance: we average the sequence-form strategies $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)}$ using the formula $\bar{\mathbf{x}}^{(t)} := \left(1 - \frac{6t}{(t+1)(t+2)}\right) \bar{\mathbf{x}}^{(t-1)} + \frac{6t}{(t+1)(t+2)} \mathbf{x}^{(t)}$, and we use the *alternating updates* scheme. We call this algorithm *Predictive CFR⁺* (PCFR⁺). We compare PCFR⁺ to the prior state-of-the-art CFR variants: CFR⁺ (Tammelin, 2014), *Discounted CFR* (DCFR) with its recommended parameters (Brown and Sandholm, 2019), and *Linear CFR* (LCFR) (Brown and Sandholm, 2019).

We conduct the experiments on common parameteric benchmark games. Each game is identified with an alphabetical mnemonic that uniquely denotes the game and parameters. A full description of the game instances, as well as their dimensions, is available in Appendix A. Results are shown in Figure 4.5.

The x-axis shows the number of iterations of each algorithm. Every algorithm pays almost exactly the same cost per iteration, since the predictions require only one additional thresholding

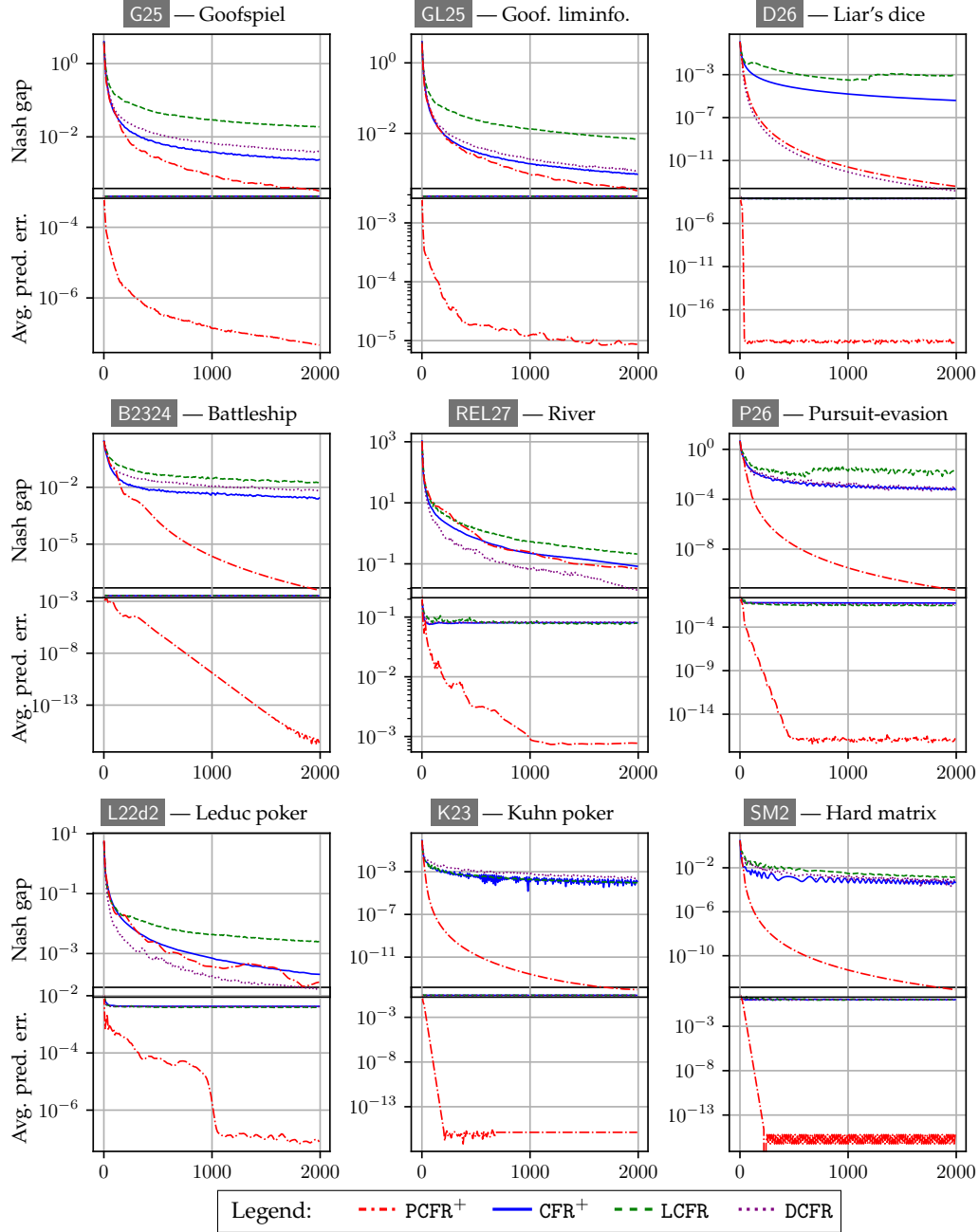


Figure 4.5: Performance of PCFR⁺, CFR⁺, DCFR, and LCFR on nine games. In all plots, the x axis is the number of iterations of each algorithm. For each game, the top plot shows that the Nash gap on the y axis (on a log scale), the bottom plot shows and the average prediction error (on a log scale).

step in PCFR⁺. For each game, the top plot shows on the y-axis the Nash gap, while the bottom plot shows the accuracy in our predictions of the regret vector, measured as the average ℓ_2 norm of the difference between the actual loss $\mathbf{u}^{(t)}$ received and its prediction $\mathbf{m}^{(t)}$ across all no-regret-algorithms at all decision points in the game. For all non-predictive algorithms (CFR⁺, LCFR, and DCFR), we let $\mathbf{m}^{(t)} = \mathbf{0}$. For our predictive algorithm, we set $\mathbf{m}^{(t)} = \mathbf{u}^{(t-1)}$ at all times $t \geq 2$ and $\mathbf{m}^{(1)} = \mathbf{0}$, in accordance with the canonical optimistic learning setup (Section 3.2.1). Both y-axes are in log scale. On **B2324** Battleship and **P26** Pursuit-evasion, PCFR⁺ is faster than the other algorithms by 3-6 orders of magnitude already after 500 iterations, and around 10 orders of magnitude after 2000 iterations. On **G25** Goofspiel, PCFR⁺ is also significantly faster than the other algorithms, by 0.5-1 order of magnitude. On **L22d2** Leduc poker as well as the **REL27** River endgame, the predictions in PCFR⁺ do not seem to help as much as in other games. On the River endgame, the performance is essentially the same as that of CFR⁺. On **L22d2** Leduc poker, it leads to a small speedup over CFR⁺. On both of those games, DCFR is fastest. In contrast, DCFR actually performs worse than CFR⁺ in the non-poker experiments, though it is sometimes on par with CFR⁺.

Finally, PRM⁺ converges very rapidly on the **SM2** game, a 2-by-2 matrix nonsequential game that was identified as a hard instance for RM-based methods (Farina, Kroer, and Sandholm, 2019b). By using optimism, we seem to be able to sidestep the issue.

We observe that, as expected, the convergence rate of PCFR⁺ is closely related to how good the predictions $\mathbf{m}^{(t)}$ of $\mathbf{u}^{(t)}$ are. On **B2324** Battleship and **P26** Pursuit-evasion, the predictions become extremely accurate very rapidly, and PCFR⁺ converges at an extremely fast rate. On **G25** Goofspiel, the predictions are fairly accurate (the error is of the order 10^{-5}) and PCFR⁺ is still significantly faster than the other algorithms. On the **REL27** River endgame, the average prediction error is of the order 10^{-3} , and PCFR⁺ performs on par with CFR⁺, and slower than DCFR. Overall, the experiments show that PCFR⁺ tend to be superior to CFR⁺ and DCFR on non-poker games thanks to its ability to incorporate predictions, whereas on poker games DCFR is the fastest.

Chapter 5

Notions of distance for sequence-form strategies, and prox methods

We have seen in [Chapter 4](#) that an effective way of constructing no-external-regret dynamics for sequence-form strategy polytopes is through regret circuits. While algorithms constructed through regret circuits are currently the practical state of the art for finding equilibria in large games, in this chapter we look into techniques that belong to the important literature of (online) convex optimization, which enjoy strong theoretical properties that are not known to apply to algorithms constructed through regret circuits. The investigation of these optimization techniques will naturally expose us to a series of fundamental questions about the metric structure and properties of sequence-form strategy polytopes. We touch on some of these questions below.

Unlike regret circuits, which guarantee feasibility of the strategies produced by exploiting the known combinatorial structure of the strategy set, most convex optimization methods maintain feasibility of the iterates by *projecting* onto the feasible sets. Typically, much flexibility is allowed in the notion of distance—called the *distance-generating function (DGF)*—that is used to define the projection (or, more precisely, *proximal*) steps. This immediately leads to the following natural question, which serves as the central theme of this chapter.

What are suitable notions of distance (distance-generating functions) for sequence-form strategies in imperfect-information extensive-form games, leading to efficient proximal steps?

In principle, Euclidean distance can be used as a notion of distance between sequence-form strategies. However, its efficacy is limited in practice by the fact that projecting points onto the polytope of sequence-form strategies according to the Euclidean distance is prohibitively expensive in practice (Gilpin, Peña, and Sandholm, 2012).

5.1 Contributions and related work

In this chapter we provide several contributions in the direction of better understanding which notions of distance best serve sequence-form strategy polytopes. Most importantly, in [Section 5.4.2](#) we provide the first DGF that enables, at the same time, (i) projections that can be computed in linear-time in the number of sequences, and (ii) polynomial diameter of the set of the sequence-form strategies, a key quantity that affects the runtime of convex optimization algorithms.

This chapter is structured as follows. In [Section 5.2.1](#) we recall the properties that DGFs need to satisfy for a generic convex feasible set, and introduce proximal setups. In [Section 5.2.2](#) we recall four convex optimization algorithms (two of which define no-external-regret algorithms, and two of which do not), elucidating their dependence on the choice of DGFs. In [Section 5.3](#) we investigate projections onto sequence-form strategy polytopes with respect to the Euclidean norm (and small variations of it, which will be important in [Chapter 6](#)). In particular, we will show that exact projections can be computed inductively on the structure of the TFDP, albeit at a superlinear cost in the number of sequences. Such a cost is prohibitive in large games. Motivated by the desire of overcoming the superlinear dependence, in [Section 5.4](#), we move away from Euclidean distance to focus on notions of distance that enable linear-time (in the number of sequences) projections onto the sequence-form strategy polytope. Specifically, in [Section 5.4.1](#) we introduce the class of *dilated* DGFs, a class of DGFs specifically designed for sequence-form strategy polytopes originally introduced by Hoda, Gilpin, Peña, and Sandholm (2010). Dilated DGFs enable fast projections, but the diameter they induce currently exhibits an unfavorable, exponential dependence on the dimension of the sequence-form strategy polytope. Finally, in [Section 5.4.2](#) we introduce a new DGF, which we coin *dilatable global entropy DGF*. Dilatable global entropy retains the appealing linear-time guarantees on projections, while overcoming—for the first time—the exponential diameter in favor of only a polynomial one. It is currently the theoretical state-of-the-art notion of distance for strategies in imperfect-information extensive-form games.

Prior work The framework of dilated DGFs was introduced by Hoda, Gilpin, Peña, and Sandholm (2010). Hoda, Gilpin, Peña, and Sandholm (2010) also introduce a notion of a “nice” DGF. Their definition is similar to the one used in this chapter, but only states that certain operations should be “easily computable”. In contrast, we attach a concrete meaning to that statement: we take it to mean linear time in the dimension of the domain.

Kroer, Waugh, Kılınç-Karzan, and Sandholm (2020) and Kroer, Farina, and Sandholm (2018b) note the drawback in the current analysis of dilated DGFs regarding the exponential dependence of the DGF weights in the depth of the decision space. Farina, Kroer, and Sandholm (2019b) extend the analysis of the strong-convexity modulus of φ with respect to the ℓ_2 norm on \mathcal{Q} , but their weighting scheme is still exponential in the worst case.

The predictive versions of OMD and FTRL presented in this chapter as applications of proximal

setups can be traced back to the works by Chiang, Yang, C.-J. Lee, Mahdavi, C.-J. Lu, R. Jin, and Zhu (2012), A. Rakhlin and Sridharan (2013), S. Rakhlin and Sridharan (2013), and Syrgkanis, Agarwal, Luo, and Schapire (2015). The mirror prox algorithm was introduced by Nemirovski (2004). The excessive gap technique was introduced by Nesterov (2005a).

Finally, we remark that the question as to what DGF achieves desirable properties has been explored in the literature for several classes of related decision sets as well, including i) the simplex case, where both the negative entropy DGF and the Euclidean DGF are known to have good properties (Held, Wolfe, and Crowder, 1974; Beck and Teboulle, 2003; Condat, 2016), and ii) the case of positive semidefinite matrices with a trace constraint, where the matrix entropy performs well (Aharon Ben-Tal and Nemirovski, 2005).

5.2 Preliminaries

In this section we recall key concepts and applications related to proximal setups.

5.2.1 Distance-generating functions and proximal setups

Let \mathcal{X} be a convex and compact set. A *distance-generating function* for \mathcal{X} is defined as follows.

Definition 5.1 (Distance-generating function). A *distance-generating function* (DGF) φ for a compact and convex set $\mathcal{X} \subseteq \mathbb{R}^d$ is a function $\varphi : \mathcal{X} \rightarrow \mathbb{R}$ such that:

- it is continuous on \mathcal{X} and differentiable in the relative interior of \mathcal{X} ;
- it is strongly convex in the relative interior of \mathcal{X} with respect to some norm $\|\cdot\|$, that is, there exists a constant $\mu > 0$ such that

$$\langle \nabla\varphi(\mathbf{x}) - \nabla\varphi(\mathbf{x}'), \mathbf{x} - \mathbf{x}' \rangle \geq \mu \|\mathbf{x} - \mathbf{x}'\|^2 \quad \forall \mathbf{x}, \mathbf{x}' \in \text{relint } \mathcal{X}.$$

For twice-differentiable φ , the strong convexity condition is automatically verified as long as

$$\langle \mathbf{m}, \nabla^2\varphi(\mathbf{x})\mathbf{m} \rangle \geq \mu \|\mathbf{m}\|^2, \quad \forall \mathbf{x} \in \text{relint } \mathcal{X}, \mathbf{m} \in \mathbb{R}^n. \quad (5.1)$$

Once a distance-generating function φ for \mathcal{X} has been picked, several important tools can be defined, which collectively form a *proximal setup* for \mathcal{X} :

- The *Bregman divergence* $D_\varphi : \mathcal{X} \times \text{relint } \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$ associated with φ yields a notion of distance between points defined as^[5.a]

^[5.a]A Bregman divergence need not be symmetric and thus might not be a metric.

$$D_\varphi(\mathbf{x} \parallel \mathbf{x}') := \varphi(\mathbf{x}) - \varphi(\mathbf{x}') - \langle \nabla \varphi(\mathbf{x}'), \mathbf{x} - \mathbf{x}' \rangle \quad \forall \mathbf{x} \in \mathcal{X}, \mathbf{x}' \in \text{relint } \mathcal{X}. \quad (5.2)$$

- The φ -diameter of \mathcal{X} is

$$\Omega_{\varphi, \mathcal{X}} := \max_{\mathbf{x}, \mathbf{x}' \in \mathcal{X}} D_\varphi(\mathbf{x} \parallel \mathbf{x}') \leq \max_{\mathbf{x} \in \mathcal{X}} \varphi(\mathbf{x}) - \min_{\mathbf{x} \in \mathcal{X}} \varphi(\mathbf{x}) \quad (5.3)$$

- Finally, we denote the largest possible value of the ℓ_1 norm on \mathcal{X} with the symbol

$$M_{\mathcal{X}} := \max_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x}\|_1.$$

While not a part of the assumptions on the DGF φ , it is typically assumed that φ allows one to efficiently compute the following two quantities, which come up at every iteration of most convex optimization algorithms:

- the *gradient* $\nabla \varphi(\mathbf{x})$ of φ at any point $\mathbf{x} \in \text{relint } \mathcal{X}$;
- the gradient of the convex conjugate φ^* of φ at any point $\mathbf{g} \in \mathbb{R}^d$:

$$\nabla \varphi^*(\mathbf{g}) = \arg \max_{\hat{\mathbf{x}} \in \mathcal{X}} \{\langle \mathbf{g}, \hat{\mathbf{x}} \rangle - \varphi(\hat{\mathbf{x}})\}.$$

The gradient of the convex conjugate can be intuitively thought of as a linear maximization problem over \mathcal{X} (i.e., the *support function* of \mathcal{X} , which is a non-smooth convex optimization problem), *smoothed* by the regularizer φ . For that reason, in this dissertation we shall refer to $\nabla \varphi^*(\mathbf{g})$ either symbolically, or occasionally as the *smoothed support function*.

Because the above two quantities arise so frequently in optimization methods, it is important that the chosen distance-generating function allow for efficient computation of them. In particular, in this dissertation we are concerned with “nice” DGFs that enable linear-time (in the dimension φ) exact computation of those two quantities.

Definition 5.2. A distance-generating function φ is said to be “nice” if $\varphi(\mathbf{x})$, $\nabla \varphi(\mathbf{x})$ and $\nabla \varphi^*(\mathbf{g})$ can be computed exactly in linear time in the dimension of the domain of φ .

Finally, we mention a closely related operation that comes up often in optimization methods: the *proximal operator* (or *prox operator* for short), defined as

$$\text{prox}_\varphi(\mathbf{g} \parallel \tilde{\mathbf{x}}) := \arg \min_{\mathbf{x} \in \mathcal{X}} \{\langle \mathbf{g}, \mathbf{x} \rangle + D_\varphi(\mathbf{x} \parallel \tilde{\mathbf{x}})\} = \nabla \varphi^*(-\mathbf{g} + \nabla \varphi(\tilde{\mathbf{x}})) \in \mathcal{X} \quad (5.4)$$

for any $\tilde{\mathbf{x}} \in \mathcal{X}$ and $\mathbf{g} \in \mathbb{R}^d$. In light of (5.4), the prox operator can be implemented efficiently provided that $\nabla \varphi$ and $\nabla \varphi^*$ can. So, prox operators can be computed exactly in linear time in the

dimension φ for “nice” DGFs.

5.2.2 Applications

Proximal setups are ubiquitous in convex optimization. In this section, we discuss four important convex optimization methods, all of which require a proximal setup.

The first two methods—online mirror descent (OMD) and follow-the-regularized-leader (FTRL)—are the premier algorithms in online convex optimization. Given any convex and compact set \mathcal{X} , OMD and FTRL define predictive no-external-regret algorithms with rather strong properties. By having access to good proximal setups (for example, enjoying linear-time projections) for sequence-form strategy polytopes $\mathcal{X} = \mathcal{Q}$, we will be able to efficiently instantiate OMD and FTRL in imperfect-information extensive-form games.

The other two methods—excessive gap technique and mirror prox—are the two premier accelerated bilinear saddle-point algorithm. They are offline optimization methods designed to solve problems of the form $\max_{\mathbf{x} \in \mathcal{X}} \min_{\mathbf{y} \in \mathcal{Y}} \mathbf{x}^\top \mathbf{A} \mathbf{y}$, which are frequent in games. The most famous example is perhaps the computation of a Nash equilibrium in a two-player zero-sum imperfect-information extensive-form game, in which case the sets \mathcal{X} and \mathcal{Y} are the sequence-form polytopes of the players, and \mathbf{A} is the payoff matrix of the game. This again underlines the importance of researching proximal setups for sequence-form polytopes with strong properties, which will be our focus in the next two sections.

5.2.3 Online mirror descent and follow-the-regularized-leader

Follow-the-regularized-leader (FTRL, Shalev-Shwartz and Singer, 2007) and *online mirror descent (OMD)* are the two best-known (and studied) algorithms in online convex optimization. While their *predictive* variants are relatively new and can be traced back to the works by Chiang, Yang, C.-J. Lee, Mahdavi, C.-J. Lu, R. Jin, and Zhu (2012), A. Rakhlin and Sridharan (2013), S. Rakhlin and Sridharan (2013), and Syrgkanis, Agarwal, Luo, and Schapire (2015), in line with the rest of the dissertation we use the name FTRL and OMD to refer to the predictive versions, with the usual understanding that non-predictive variants of FTRL and OMD algorithms correspond to predictive FTRL and predictive OMD when the prediction $\mathbf{m}^{(t)}$ is set to the $\mathbf{0}$ vector at all t .

Algorithms 5.1 and 5.2 give pseudocode for FTRL and OMD. In both algorithm, $\eta > 0$ is an arbitrary step size parameter, $\mathcal{X} \subseteq \mathbb{R}^d$ is a convex and compact set, and $\varphi : \mathcal{X} \rightarrow \mathbb{R}_{\geq 0}$ is a 1-strongly convex distance-generating function with respect to some norm $\|\cdot\|$. We also recall that the symbol $D_\varphi(\cdot \|\cdot)$ used in OMD denotes the *Bregman divergence* associated with φ , a standard surrogate notion of distance in convex optimization which was defined in (5.2).

Predictive FTRL and predictive OMD satisfy the following regret bound.

Algorithm 5.1: Predictive FTRL

```

1  $\mathbf{U}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^d$ 
2 function NextStrategy( $\mathbf{m}^{(t)} \in \mathbb{R}^d$ )
   | [ $\triangleright$  Set  $\mathbf{m}^{(t)} = \mathbf{0}$  for non-predictive version]
3    $\mathbf{x}^{(t)} \leftarrow \arg \min_{\hat{\mathbf{x}} \in \mathcal{X}} \left\{ \langle \mathbf{U}^{(t-1)} + \mathbf{m}^{(t)}, \hat{\mathbf{x}} \rangle + \frac{1}{\eta} \varphi(\hat{\mathbf{x}}) \right\}$ 
4   return  $\mathbf{x}^{(t)}$ 
5 function ObserveUtility( $\mathbf{u}^{(t)} \in \mathbb{R}^d$ )
6    $\mathbf{U}^{(t)} \leftarrow \mathbf{U}^{(t-1)} + \mathbf{u}^{(t)}$ 

```

Algorithm 5.2: Predictive OMD

```

1  $\mathbf{z}^{(0)} \in \mathcal{X}$  such that  $\nabla \varphi(\mathbf{z}^{(0)}) = \mathbf{0}$ 
2 function NextStrategy( $\mathbf{m}^{(t)} \in \mathbb{R}^d$ )
   | [ $\triangleright$  Set  $\mathbf{m}^{(t)} = \mathbf{0}$  for non-predictive version]
3    $\mathbf{x}^{(t)} \leftarrow \arg \min_{\hat{\mathbf{x}} \in \mathcal{X}} \left\{ \langle \mathbf{m}^{(t)}, \hat{\mathbf{x}} \rangle + \frac{1}{\eta} D_\varphi(\hat{\mathbf{x}} \parallel \mathbf{z}^{(t-1)}) \right\}$ 
4   return  $\mathbf{x}^{(t)}$ 
5 function ObserveUtility( $\mathbf{u}^{(t)} \in \mathbb{R}^d$ )
6    $\mathbf{z}^{(t)} \leftarrow \arg \min_{\hat{\mathbf{z}} \in \mathcal{X}} \left\{ \langle \mathbf{u}^{(t)}, \hat{\mathbf{z}} \rangle + \frac{1}{\eta} D_\varphi(\hat{\mathbf{z}} \parallel \mathbf{z}^{(t-1)}) \right\}$ 

```

Proposition 5.1 (Syrkkanis, Agarwal, Luo, and Schapire, 2015). Let $\Omega_{\varphi, \mathcal{X}}$ denote the diameter of φ over \mathcal{X} , as defined in Equation (5.3). At all times T , the regret cumulated by predictive FTRL (Algorithm 5.1) and predictive OMD (Algorithm 5.2) compared to *any* strategy $\hat{\mathbf{x}} \in \mathcal{X}$ is bounded as

$$\text{Reg}^{(T)} \leq \frac{\Omega_{\varphi, \mathcal{X}}}{\eta} + \eta \sum_{t=1}^T \|\ell^t - \mathbf{m}^{(t)}\|_*^2 - \frac{1}{c\eta} \sum_{t=2}^T \|\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}\|^2,$$

where $c = 4$ for FTRL and $c = 8$ for OMD, and where $\|\cdot\|_*$ denotes the dual of the norm $\|\cdot\|$ with respect to which φ is 1-strongly convex.

We remark that the bound is RVU-predictive (see Section 3.2.4), a stronger guarantee compared to CFR's weak-predictivity (Chapter 4). In particular, predictive OMD and predictive FTRL guarantee $\mathcal{O}_T(1/T)$ convergence to Nash equilibrium, and $\mathcal{O}_T(T^{1/4})$ per-player external regret when used in self-play, in turn implying $\mathcal{O}_T(T^{-3/4})$ convergence to a coarse-correlated equilibrium in general imperfect-information extensive-form games.

5.2.4 Bilinear saddle points: Excessive gap technique and mirror prox

Another important class of convex optimization methods with applications to game solving is methods for solving *bilinear saddle-point problems* (BSPPs), whose general form is

$$\min_{\mathbf{x} \in \mathcal{X}} \max_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{x}, \mathbf{A}\mathbf{y} \rangle, \tag{5.5}$$

where $\mathcal{X} \subseteq \mathbb{R}^{d_x}$, $\mathcal{Y} \subseteq \mathbb{R}^{d_y}$ are convex and compact sets, and $\mathbf{A} \in \mathbb{R}^{d_x \times d_y}$. This captures, among other, the problem of computing a Nash equilibrium in a two-player zero-sum imperfect-information extensive-form game.

We will now present the EGT and mirror prox algorithms for solving BSPPs. These algorithms depend on two proximal setups: one for \mathcal{X} and one for \mathcal{Y} , denoted φ_x and φ_y , respectively. Let $\|\cdot\|_x$ and $\|\cdot\|_y$ be the norms associated with the strong convexity of φ_x and φ_y in the given proximal setup. The convergence rate then depends on the following *operator norm* of the payoff matrix \mathbf{A} :

$$\|\mathbf{A}\| := \max\{\langle \mathbf{x}, \mathbf{A}\mathbf{y} \rangle : \|\mathbf{x}\|_x \leq 1, \|\mathbf{y}\|_y \leq 1\}.$$

The magnitude of $\|\mathbf{A}\|$ is the primary way in which the norm matters: if both φ_x and φ_y are strongly convex with respect to the ℓ_2 norm, then $\|\mathbf{A}\|$ can be on the order of $\sqrt{d_x d_y}$, whereas if both are with respect to the ℓ_1 norm, then $\|\mathbf{A}\|$ is simply equal to its largest entry.

Excessive gap technique (EGT) algorithm The *excessive gap technique (EGT)* is a first-order method introduced by Nesterov (2005b), and one of the primary applications is to solve BSPPs such as Equation (5.5). EGT assumes access to a proximal setup for \mathcal{X} and \mathcal{Y} , with one-strongly-convex DGFs φ_x, φ_y , and constructs smoothed approximations of the optimization problems faced by the max and min players. Based on this setup, we formally state the EGT of Nesterov (2005a) in Algorithm 5.3. EGT alternately takes steps focused on decreasing one or the other smoothing parameter. These steps are called ShrinkX and ShrinkY in Algorithm 5.3.

Algorithm 5.3: Excessive gap technique (EGT) algorithm

<pre> 1 function Initialize() 2 $t \leftarrow 0$ 3 $\mu_x^{(0)} \leftarrow \ \mathbf{A}\ , \mu_y^{(0)} \leftarrow \ \mathbf{A}\$ 4 $\hat{\mathbf{x}} \leftarrow \arg \min_{\hat{\mathbf{x}} \in \mathcal{X}} \varphi_x(\hat{\mathbf{x}})$ 5 $\mathbf{y}^{(0)} \leftarrow \nabla \varphi_y^*(\mathbf{A}^\top \hat{\mathbf{x}} / \mu_y^{(0)})$ 6 $\mathbf{x}^{(0)} \leftarrow \text{prox}_{\varphi_x} \left(\frac{1}{\mu_x^{(0)}} \mathbf{A}\mathbf{y}^{(0)} \parallel \hat{\mathbf{x}} \right)$ 7 function Iterate() 8 $t \leftarrow t + 1, \tau \leftarrow 2/(t + 2)$ 9 if t is even then ShrinkX() 10 else ShrinkY() </pre>	<pre> 11 function ShrinkX() 12 $\bar{\mathbf{x}} \leftarrow -\nabla \varphi_x^*(-\mathbf{A}\mathbf{y}^{t-1} / \mu_x^{t-1})$ 13 $\hat{\mathbf{x}} \leftarrow (1 - \tau)\mathbf{x}^{t-1} + \tau\bar{\mathbf{x}}$ 14 $\bar{\mathbf{y}} \leftarrow \nabla \varphi_y^*(\mathbf{A}^\top \hat{\mathbf{x}} / \mu_y^{t-1})$ 15 $\tilde{\mathbf{x}} \leftarrow \text{prox}_{\varphi_x} \left(\frac{\tau}{(1-\tau)\mu_x^{t-1}} \mathbf{A}\bar{\mathbf{y}} \parallel \hat{\mathbf{x}} \right)$ 16 $\mathbf{x}^t \leftarrow (1 - \tau)\mathbf{x}^{t-1} + \tau\tilde{\mathbf{x}}$ 17 $\mathbf{y}^t \leftarrow (1 - \tau)\mathbf{y}^{t-1} + \tau\bar{\mathbf{y}}$ 18 $\mu_x^t \leftarrow (1 - \tau)\mu_x^{t-1}$ </pre>	<pre> 19 function ShrinkY() 20 $\bar{\mathbf{y}} \leftarrow \nabla \varphi_y^*(\mathbf{A}^\top \mathbf{x}^{t-1} / \mu_y^{t-1})$ 21 $\hat{\mathbf{y}} \leftarrow (1 - \tau)\mathbf{y}^{t-1} + \tau\bar{\mathbf{y}}$ 22 $\bar{\mathbf{x}} \leftarrow -\nabla \varphi_x^*(-\mathbf{A}\hat{\mathbf{y}} / \mu_x^{t-1})$ 23 $\tilde{\mathbf{y}} \leftarrow \text{prox}_{\varphi_y} \left(\frac{-\tau}{(1-\tau)\mu_y^{t-1}} \mathbf{A}^\top \bar{\mathbf{x}} \parallel \hat{\mathbf{y}} \right)$ 24 $\mathbf{y}^t \leftarrow (1 - \tau)\mathbf{y}^{t-1} + \tau\tilde{\mathbf{y}}$ 25 $\mathbf{x}^t \leftarrow (1 - \tau)\mathbf{x}^{t-1} + \tau\bar{\mathbf{x}}$ 26 $\mu_y^t \leftarrow (1 - \tau)\mu_y^{t-1}$ </pre>
--	--	---

Algorithm 5.3 shows how initial points are selected and the alternating steps and stepsizes are computed. Nesterov (2005a) proves that the EGT algorithm converges at a rate of $\mathcal{O}_T(1/T)$.

Theorem 5.1 (Nesterov, 2005a, Theorem 6.3). At every iteration $t \geq 1$ of the EGT algorithm, the solution $(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})$ satisfies $\mathbf{x}^{(t)} \in \mathcal{X}, \mathbf{y}^{(t)} \in \mathcal{Y}$, and

$$\max_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{x}^{(t)}, \mathbf{A}\mathbf{y} \rangle - \min_{\hat{\mathbf{x}} \in \mathcal{X}} \langle \hat{\mathbf{x}}, \mathbf{A}\mathbf{y}^{(t)} \rangle \leq \frac{4\|\mathbf{A}\| \sqrt{\Omega_{\varphi_x, \mathcal{X}} \cdot \Omega_{\varphi_y, \mathcal{X}}}}{t + 1}.$$

Mirror prox (MPROX) algorithm The *mirror prox (MPROX)* algorithm was introduced by Nemirovski (2004). Rather than construct smoothed approximations, mirror prox directly uses the DGFs to take first-order steps. Hence, the MPROX algorithm is best understood as an algorithm that operates on the product space $\mathcal{X} \times \mathcal{Y}$ directly. As such, in most analyses of the MPROX algorithm, a single 1-strongly convex DGF for the product space $\mathcal{X} \times \mathcal{Y}$ is required. To better align with the setup used for EGT, we will define the DGF for the product space $\mathcal{X} \times \mathcal{Y}$ starting from proximal setups for both \mathcal{X} and \mathcal{Y} , with 1-strongly convex DGFs φ_x, φ_y with respect to norms $\|\cdot\|_x$ and $\|\cdot\|_y$, respectively. With this setup, it is immediate to see that the function

$$\varphi : \mathcal{X} \times \mathcal{Y} \ni (\mathbf{x}, \mathbf{y}) \mapsto \varphi_x(\mathbf{x}) + \varphi_y(\mathbf{y})$$

is a DGF for the product space $\mathcal{X} \times \mathcal{Y}$, which is strongly convex with modulus one with respect to the norm $\|(\mathbf{x}, \mathbf{y})\| := \sqrt{\|\mathbf{x}\|_x^2 + \|\mathbf{y}\|_y^2}$. Furthermore, each proximal step taken with respect to d can be expressed as two independent proximal steps with respect to φ_x and φ_y :

$$\begin{aligned} \text{prox}_{\varphi} \left(\begin{pmatrix} \mathbf{g}_x \\ \mathbf{g}_y \end{pmatrix} \parallel \begin{pmatrix} \tilde{\mathbf{x}} \\ \tilde{\mathbf{y}} \end{pmatrix} \right) &= \arg \min_{(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}} \left\{ \left\langle \begin{pmatrix} \mathbf{g}_x \\ \mathbf{g}_y \end{pmatrix}, \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \right\rangle + D_{\varphi} \left(\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \parallel \begin{pmatrix} \tilde{\mathbf{x}} \\ \tilde{\mathbf{y}} \end{pmatrix} \right) \right\} \\ &= \begin{pmatrix} \arg \min_{\mathbf{x} \in \mathcal{X}} \{ \langle \mathbf{g}_x, \mathbf{x} \rangle + \varphi_x(\mathbf{x}) - \langle \nabla \varphi_x(\tilde{\mathbf{x}}), \mathbf{x} \rangle \} \\ \arg \min_{\mathbf{y} \in \mathcal{Y}} \{ \langle \mathbf{g}_y, \mathbf{y} \rangle + \varphi_y(\mathbf{y}) - \langle \nabla \varphi_y(\tilde{\mathbf{y}}), \mathbf{y} \rangle \} \end{pmatrix} \\ &= \begin{pmatrix} \text{prox}_{\varphi_x}(\mathbf{g}_x \parallel \tilde{\mathbf{x}}) \\ \text{prox}_{\varphi_y}(\mathbf{g}_y \parallel \tilde{\mathbf{y}}) \end{pmatrix}. \end{aligned}$$

Similarly, the d -diameter of the product space $\mathcal{X} \times \mathcal{Y}$ is equal to the sum of diameters of \mathcal{X} and \mathcal{Y} in their respective proximal setups. Finally, we note that the function

$$F : \mathcal{X} \times \mathcal{Y} \ni (\mathbf{x}, \mathbf{y}) \mapsto \begin{pmatrix} \mathbf{A}\mathbf{y} \\ -\mathbf{A}^{\top}\mathbf{x} \end{pmatrix},$$

critical in the analysis of MPROX (Ahron Ben-Tal and Nemirovski, 2001), satisfies

$$\begin{aligned} \left\| F \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} - F \begin{pmatrix} \mathbf{x}' \\ \mathbf{y}' \end{pmatrix} \right\|_* &= \sqrt{\|\mathbf{A}(\mathbf{y} - \mathbf{y}')\|_{x^*}^2 + \|\mathbf{A}^{\top}(\mathbf{x} - \mathbf{x}')\|_{y^*}^2} \\ &\leq \sqrt{\left[\max_{\|\tilde{\mathbf{x}}\|_x \leq 1} \langle \tilde{\mathbf{x}}, \mathbf{A}(\mathbf{y} - \mathbf{y}') \rangle \right]^2 + \left[\max_{\|\tilde{\mathbf{y}}\|_y \leq 1} \langle \mathbf{x} - \mathbf{x}', \mathbf{A}\tilde{\mathbf{y}} \rangle \right]^2} \\ &\leq \sqrt{\|\mathbf{A}\|^2 \cdot \|\mathbf{y} - \mathbf{y}'\|_y^2 + \|\mathbf{A}\|^2 \cdot \|\mathbf{x} - \mathbf{x}'\|_x^2} = \|\mathbf{A}\| \cdot \left\| \begin{pmatrix} \mathbf{x} - \mathbf{x}' \\ \mathbf{y} - \mathbf{y}' \end{pmatrix} \right\|, \end{aligned}$$

that is, it is $\|\mathbf{A}\|$ -Lipschitz with respect to the norm $\|\cdot\|$ on $\mathcal{X} \times \mathcal{Y}$.

Algorithm 5.4 shows the sequence of steps taken in every iteration of the MPROX algorithm.

Algorithm 5.4: Mirror Prox (MPROX) algorithm

<pre> 1 function Initialize() 2 $t \leftarrow 0$ 3 $\mathbf{z}_x^{(0)} \leftarrow \arg \min_{\hat{\mathbf{x}} \in \mathcal{X}} \varphi_x(\hat{\mathbf{x}})$ 4 $\mathbf{z}_y^{(0)} \leftarrow \arg \min_{\hat{\mathbf{y}} \in \mathcal{Y}} \varphi_y(\hat{\mathbf{y}})$ </pre>	<pre> 5 function lterate() 6 $t \leftarrow t + 1$ 7 $\mathbf{w}_x^t \leftarrow \text{prox}_{\varphi_x}(\eta^t \mathbf{A} \mathbf{z}_y^{t-1} \parallel \mathbf{z}_x^t)$ 8 $\mathbf{w}_y^t \leftarrow \text{prox}_{\varphi_y}(-\eta^t \mathbf{A}^\top \mathbf{z}_x^{t-1} \parallel \mathbf{z}_y^t)$ 9 $\mathbf{z}_x^{t+1} \leftarrow \text{prox}_{\varphi_x}(\eta^t \mathbf{A} \mathbf{w}_y^t \parallel \mathbf{z}_x^t)$ 10 $\mathbf{z}_y^{t+1} \leftarrow \text{prox}_{\varphi_y}(-\eta^t \mathbf{A}^\top \mathbf{w}_x^t \parallel \mathbf{z}_y^t)$ 11 $\mathbf{x}^{(t)} \leftarrow [\sum_{\tau=1}^t \eta^\tau]^{-1} \sum_{\tau=1}^t \eta^\tau \mathbf{w}_x^\tau$ 12 $\mathbf{y}^{(t)} \leftarrow [\sum_{\tau=1}^t \eta^\tau]^{-1} \sum_{\tau=1}^t \eta^\tau \mathbf{w}_y^\tau$ </pre>	<p>Note: $\{\eta^t\}$ is a sequence of step-size parameters. A well-known and theoretically-sound choice for η^t is $\eta^t := \frac{1}{\ A\ }$ for all $t = 0, 1, \dots$ (see also Theorem 5.2).</p>
---	--	---

Compared to EGT, mirror prox has a somewhat simpler structure: it simply takes repeated extrapolated proximal steps. First, a proximal step in the descent direction is taken for both x and y . Then, the gradient at those new points is used to take a proximal step starting from the previous iterate (this is the extrapolation part: a step is taken starting from the previous iterate, but with the extrapolated gradient). Finally, the *average* strategy is output.

As we recall in the next theorem, like EGT the MPROX algorithm converges at rate $\mathcal{O}_T(1/T)$.

Theorem 5.2 (Ahron Ben-Tal and Nemirovski, 2001, Theorem 5.5.1). Suppose the stepsize in Algorithm 5.4 is set as $\eta_t = 1/\|A\|$. Then we have

$$\max_{\hat{\mathbf{y}} \in \mathcal{Y}} \langle \mathbf{x}^{(t)}, \mathbf{A} \hat{\mathbf{y}} \rangle - \min_{\hat{\mathbf{x}} \in \mathcal{X}} \langle \hat{\mathbf{x}}, \mathbf{A} \mathbf{y}^{(t)} \rangle \leq \frac{\|A\|(\Omega_{\varphi_x, \mathcal{X}} + \Omega_{\varphi_y, \mathcal{Y}})}{2t}.$$

5.3 Euclidean distance-generating function

We begin our investigation of distance-generating functions for sequence-form strategy spaces starting from the prototypical notion: the Euclidean distance-generating function. Fix any tree-form decision process, and let \mathcal{Q} be the corresponding sequence-form strategy polytope. The Euclidean distance-generating function is the function

$$\varphi(\mathbf{x}) := \frac{1}{2} \|\mathbf{x}\|_2^2 = \frac{1}{2} \sum_{\sigma \in \Sigma} x[\sigma]^2, \quad \mathbf{x} \in \mathcal{Q}.$$

In this case, using the observation that $\nabla \varphi(\mathbf{x}') = \mathbf{x}'$ for all $\mathbf{x}' \in \mathcal{Q}$, it is easy to see that the Bregman divergence associated with φ reduces to (half of) the squared Euclidean distance between two points,

$$D_\varphi(\mathbf{x} \parallel \mathbf{x}') := \frac{1}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{x}'\|_2^2 - \langle \mathbf{x}', \mathbf{x} - \mathbf{x}' \rangle = \frac{1}{2}\|\mathbf{x} - \mathbf{x}'\|_2^2.$$

Similarly, the gradient of the convex conjugate of φ in a generic point $\mathbf{g} \in \mathbb{R}^\Sigma$ reduces to the usual Euclidean projection,

$$\begin{aligned} \nabla\varphi^*(\mathbf{g}) &= \arg \max_{\hat{\mathbf{x}} \in \mathcal{Q}} \{ \langle \mathbf{g}, \hat{\mathbf{x}} \rangle - \varphi(\hat{\mathbf{x}}) \} = \arg \min_{\hat{\mathbf{x}} \in \mathcal{Q}} \left\{ -\langle \mathbf{g}, \hat{\mathbf{x}} \rangle + \frac{1}{2}\|\hat{\mathbf{x}}\|_2^2 \right\} \\ &= \arg \min_{\hat{\mathbf{x}} \in \mathcal{Q}} \frac{1}{2}\|\mathbf{g} - \hat{\mathbf{x}}\|_2^2. \end{aligned}$$

The computation of the projection $\arg \min_{\hat{\mathbf{x}} \in \mathcal{Q}} \|\mathbf{g} - \hat{\mathbf{x}}\|_2^2$ is not known to be possible in linear time in the number of sequences $|\Sigma|$. This means the Euclidean DGF is not known to be “nice” in the sense [Definition 5.2](#), and in many cases it might provide a less appealing proximal setup than, for example, the dilatable global entropy DGF which will be introduced later in the chapter. However, there are good reasons to study the properties of the Euclidean DGF. For one, out of the DGFs that have been proposed for imperfect-information extensive-form games it is the only one with Lipschitz-continuous gradients, which can be fundamental in certain contexts (see, *e.g.*, [Anagnostides, Panageas, Farina, and Sandholm \(2022\)](#)). Furthermore, some of the techniques we will develop when studying algorithms for Euclidean projection will prove important in other chapters (specifically, [Chapter 6](#)).

5.3.1 Exact Euclidean projection algorithm

In this section we give a combinatorial algorithm for computing exactly the Euclidean projection of a generic point $\mathbf{g} \in \mathbb{R}^\Sigma$. Our algorithm formalizes and extends an idea by [Gilpin \(2009\)](#). In fact, we give an algorithm that can work with any positive definite diagonal inner product norm, that is a DGF of the form

$$\varphi_{\mathbf{w}}(\mathbf{x}) := \langle \mathbf{x}, \text{diag}(\mathbf{w})\mathbf{x} \rangle = \frac{1}{2} \sum_{\sigma \in \Sigma} \left(\frac{\mathbf{x}[\sigma]}{\mathbf{w}[\sigma]} \right)^2,$$

for some given positive vector $\mathbf{w} \in \mathbb{R}_{>0}^\Sigma$. It is immediate to see that the Euclidean DGF corresponds to the special case $\varphi_{\mathbf{1}}$ in which $\mathbf{w} = \mathbf{1} \in \mathbb{R}_{>0}^\Sigma$ is the vector of all ones. In [Chapter 6](#) we will benefit from considering projections with respect to DGFs $\varphi_{\mathbf{w}}$ in which $\mathbf{w} \neq \mathbf{1}$.

The gradient of the convex conjugate of $\varphi_{\mathbf{w}}$ corresponds to the optimization problem

$$\nabla\varphi_{\mathbf{w}}^*(\mathbf{g}) = \arg \min_{\mathbf{x} \in \mathcal{Q}} \left\{ -\langle \mathbf{g}, \mathbf{x} \rangle + \frac{1}{2} \sum_{\sigma \in \Sigma} \left(\frac{\mathbf{x}[\sigma]}{\mathbf{w}[\sigma]} \right)^2 \right\} \quad (\mathbf{g} \in \mathbb{R}^\Sigma). \quad (5.6)$$

We will show how one can solve the projection problem [\(5.6\)](#) inductively, in a bottom-up fashion,

over the structure of the tree-form decision process. To do so, at every decision node $j \in \mathcal{J}$ we define the *value function*

$$V_{\succcurlyeq j}(t) : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}, \quad V_{\succcurlyeq j}(t) := \min_{\hat{\mathbf{x}} \in t \cdot \mathcal{Q}_{\succcurlyeq j}} \left\{ - \sum_{\sigma \in \Sigma_{\succcurlyeq j}} g[\sigma] \hat{\mathbf{x}}[\sigma] + \frac{1}{2} \sum_{\sigma \in \Sigma_{\succcurlyeq j}} \left(\frac{\hat{\mathbf{x}}[\sigma]}{\mathbf{w}[\sigma]} \right)^2 \right\}. \quad (5.7)$$

We will be particularly interested in the derivatives of $V_{\succcurlyeq j}(t)$, which we will denote as^[5.b]

$$\lambda_{\succcurlyeq j}(t) := \frac{d}{dt} V_{\succcurlyeq j}(t).$$

As we will show, the function $\lambda_{\succcurlyeq j}(t)$ is extremely structured, in that it is a strictly monotonically increasing piecewise-linear (SMPL) function, as defined next.

Definition 5.3 (SMPL and quasi-SMPL function). Given an interval $I \subseteq \mathbb{R}$ and a function $f : I \rightarrow \mathbb{R}$, we say that f is *SMPL* if it is strictly monotonically increasing and piecewise-linear on I .

A *quasi-SMPL* function is a function $f : \mathbb{R} \rightarrow [0, +\infty)$ of the form $f(x) = [g(x)]^+$ where $g(x) : \mathbb{R} \rightarrow \mathbb{R}$ is SMPL and $[\cdot]^+ := \max\{0, \cdot\}$.

The piecewise-linear nature of $\lambda_{\succcurlyeq j}(t)$ lends itself nicely to combinatorial algorithms, in that the function can be represented in memory implicitly through the set of *breakpoints* at which the slope of the function changes. We call this a *standard representation* for the function, as introduced next.

Definition 5.4. Given a SMPL or quasi-SMPL function f , a *standard representation* for it is an expression of the form

$$f(x) = \bar{\alpha} + \alpha_0 x + \sum_{s=1}^S \alpha_s [x - \beta_s]^+,$$

valid for all x in the domain of f , where $S \in \mathbb{N}_{\geq 0}$ and $\beta_1 < \dots < \beta_S$. The size of the standard representation is defined as the natural number S .

We review properties and manipulations of SMPL functions in an appendix to this chapter, [Section 5.A](#). With the above definitions we are ready to state the following result, which is central in our analysis.

^[5.b]For $t = 0$ we define $\lambda_{\succcurlyeq j}(t)$ in the usual way as $\lambda_{\succcurlyeq j}(0) = \lim_{t \rightarrow 0^+} \frac{V_{\succcurlyeq j}(t) - V_{\succcurlyeq j}(0)}{t} = \lim_{t \rightarrow 0^+} \frac{V_{\succcurlyeq j}(t)}{t}$.

Lemma 5.1. For any decision node $j \in \mathcal{J}$, the function $\lambda_{\succcurlyeq j}(t)$ is SMPL and has a standard representation of size $|\mathcal{Q}_{\succcurlyeq j}|$, which can be computed in polynomial time in $|\Sigma_{\succcurlyeq j}|$.

Proof. The result is known when j is a terminal decision node, i.e., $\mathcal{Q}_{\succcurlyeq j}$ is a simplex. Fix a particular $j \in \mathcal{J}$, and assume by induction that the result holds for all $j' \succ j$.

By definition, the value function decomposes recursively as

$$\begin{aligned} V_{\succcurlyeq j}(t) &= \min_{\mathbf{x}_\bullet \in t \Delta^{A_j}} \left\{ \left(- \sum_{a \in A_j} g[ja] \mathbf{x}_\bullet[a] + \frac{1}{2} \sum_{a \in A_j} \left(\frac{\mathbf{x}_\bullet[a]}{\mathbf{w}[ja]} \right)^2 \right) \right. \\ &\quad \left. + \sum_{a \in A_j} \sum_{j' \in \mathcal{C}_{ja}} \min_{\mathbf{x}_{j'} \in \mathbf{x}_\bullet[a] \mathcal{Q}_{\succcurlyeq j'}} \left\{ - \sum_{\sigma \in \Sigma_{\succcurlyeq j'}} g[\sigma] \mathbf{x}_{j'}[\sigma] + \sum_{\sigma \in \Sigma_{\succcurlyeq j'}} \left(\frac{\mathbf{x}_{j'}[\sigma]}{\mathbf{w}[\sigma]} \right)^2 \right\} \right\} \\ &= \min_{\mathbf{x}_\bullet \in t \Delta^{A_j}} \left\{ - \sum_{a \in A_j} g[ja] \mathbf{x}_\bullet[a] + \frac{1}{2} \sum_{a \in A_j} \left(\frac{\mathbf{x}_\bullet[a]}{\mathbf{w}[ja]} \right)^2 + \sum_{a \in A_j} \sum_{j' \in \mathcal{C}_{ja}} V_{\succcurlyeq j'}(\mathbf{x}_\bullet[a]) \right\}. \end{aligned} \quad (5.8)$$

Consider the KKT conditions for \mathbf{x}_\bullet in Equation (5.8):

$$\begin{aligned} -g[ja] + \frac{\mathbf{x}_\bullet[a]}{\mathbf{w}[ja]^2} + \sum_{j' \in \mathcal{C}_{ja}} \lambda_{\succcurlyeq j'}(\mathbf{x}_\bullet[a]) &= \lambda_\bullet + \boldsymbol{\mu}[a] && \forall a \in A_j && \text{(Stationarity)} \\ \mathbf{x}_\bullet &\in t \cdot \Delta^{A_j} && && \text{(Primal feasibility)} \\ \lambda_\bullet \in \mathbb{R}, \boldsymbol{\mu} &\in \mathbb{R}_{\geq 0}^d && && \text{(Dual feasibility)} \\ \boldsymbol{\mu}[a] \mathbf{x}_\bullet[a] &= 0 && \forall a \in A_j && \text{(Compl. slackness)} \end{aligned}$$

Solving for $\mathbf{x}_\bullet[a]$ in the stationarity condition, and using the conditions $\mathbf{x}_\bullet[a] \boldsymbol{\mu}[a] = 0$ and $\boldsymbol{\mu}[a] \geq 0$, it follows that for all $a \in A_j$

$$\mathbf{x}_\bullet[a] = \mathbf{w}[ja]^2 \left[\lambda_\bullet + g[ja] - \sum_{j' \in \mathcal{C}_{ja}} \lambda_{\succcurlyeq j'}(\mathbf{x}_\bullet[a]) \right]^+. \quad (5.9)$$

Strict monotonicity and piecewise-linearity of $\mathbf{x}_\bullet[a]$ as a function of λ_\bullet . Given the properties of SMPL functions, it is immediate to see that $\mathbf{x}_\bullet[a]$ is unique as a function of λ_\bullet . Indeed, note that (5.9) can be rewritten as

$$\mathbf{x}_\bullet[a] = \left[\mathbf{w}[ja]^2 \lambda_\bullet - \mathbf{w}[ja]^2 \left(-\mathbf{g}[ja] + \sum_{j' \in \mathcal{C}_{ja}} \lambda_{\succ j'}(\mathbf{x}_\bullet[a]) \right) \right]^+,$$

which is a fixed-point problem of the form studied in [Lemma 5.8](#) in the appendix for $y = \mathbf{w}[ja]^2 \lambda_\bullet$ and function f_a defined as

$$f_a(\mathbf{x}_\bullet[a]) = \mathbf{w}[ja]^2 \left(-\mathbf{g}[ja] + \sum_{j' \in \mathcal{C}_{ja}} \lambda_{\succ j'}(\mathbf{x}_\bullet[a]) \right),$$

which is clearly SMPL by inductive hypothesis. Hence, the unique solution to the previous fixed-point equation is given by the quasi-SMPL function

$$g_a : \lambda_\bullet \mapsto \frac{1}{\mathbf{w}[ja]^2} \left[(\mathbf{x}_\bullet[a] + f_a)^{-1}(\lambda_\bullet) \right]^+,$$

a standard representation of which can be computed in time $\mathcal{O}(|\mathcal{Q}_{\succ j}|)$ by combining the results of [Lemmas 5.3, 5.5](#) and [5.6](#) given that a standard representation of each $\lambda_{\succ j'}(t)$ ($j' \in \mathcal{C}_{ja}$) of size $|\Sigma_{\succ j'}|$ is available by inductive hypothesis.

Strict monotonicity and piecewise-linearity of λ_\bullet as a function of t . At this stage, we know that given any value of the dual variable λ_\bullet , the unique value of the coordinate $\mathbf{x}_\bullet[a]$ that solves the KKT system can be computed using the quasi-SMPL function g_a . In turn, this means that we can remove the primal variables \mathbf{x}_\bullet from the KKT system, leaving us a system in λ_\bullet and t only. We now show that the solution λ_\bullet^* of that system is a SMPL function of $t \in [0, +\infty)$.

Indeed, the value $\lambda_\bullet^*(t)$ that solves the KKT system has to satisfy the primal feasibility condition

$$t = \sum_{a \in \mathcal{A}_j} \mathbf{x}_\bullet[a] = \sum_{a \in \mathcal{A}_j} g_a(\lambda_\bullet).$$

Fix any $t > 0$. The right-hand side of the equation is a sum of quasi-SMPL functions. Hence, from [Lemma 5.4](#), we have that the right-hand side has a standard representation of size at most $|\mathcal{A}_j| + \sum_{a \in \mathcal{A}_j} \sum_{j' \in \mathcal{C}_{ja}} |\Sigma_{\succ j'}| = |\Sigma_{\succ j}|$ can be computed in time $\mathcal{O}(|\Sigma_{\succ j}| \log |\mathcal{A}_j|)$. Furthermore, from [Lemma 5.7](#), we have that the λ_\bullet^* that satisfies the equation is unique, and in fact that the mapping $(0, +\infty) \ni t \mapsto \lambda_\bullet^*(t)$ is SMPL with standard representation of size at most $|\Sigma_{\succ j}|$.

Relating λ_\bullet and $\lambda_{\succ j}(t)$. Since $\lambda_\bullet^*(t)$ is the coefficient on t in the Lagrangian relaxation of [\(5.8\)](#), it is a subgradient of $V_{\succ j}(t)$, and since there is a unique solution, we get that it is the derivative,

that is, $\lambda_{\succ j}^*(t) = \lambda_{\succ j}(t)$ for all $t \in (0, +\infty)$. To conclude the proof by induction, we then need to analyze the case $t = 0$, which has so far been excluded. When $t = 0$, the feasible set $t\mathcal{Q}_{\succ j}$ is the singleton $\{\mathbf{0}\}$, and $V_{\succ j}(0) = 0$. Since $V_{\succ j}(t)$ is continuous on $[0, +\infty)$, and since $\lim_{t \rightarrow 0^+} \lambda_{\succ j}(t) = \lim_{t \rightarrow 0^+} \lambda_{\bullet}^*(t)$ exists since $\lambda_{\bullet}^*(t)$ is piecewise-linear, then by the mean value theorem,

$$\lambda_{\succ j}(0) = \lim_{t \rightarrow 0^+} \lambda_{\bullet}^*(t),$$

that is, the continuous extension of λ_{\bullet}^* must be (right) derivative of $V_{\succ j}(t)$ in 0. As extending continuously $\lambda_{\bullet}^*(t)$ clearly does not alter its being SMPL nor its standard representation, we conclude the proof of the inductive case. \square

5.4 Distance-generating functions with linear-time projections

Most convex optimization methods maintain feasibility by projecting onto the feasible set at each iteration. Hence, designing notions of distance that enable efficient projections is an important step in scaling those methods.

In this section, we focus on notions of distance that enable projections in linear time in the dimension of the dimension of the sequence-form strategy polytope. Remarkably, in [Section 5.4.2](#) we introduce the first such notion of distance whose linear-time projection property does *not* come at the cost of an exponential diameter.

5.4.1 Dilated distance-generating functions

Dilated distance-generating functions are a general framework for constructing “nice” DGFs (in the sense of [Definition 5.2](#)) for (the relative interior of the) sequence-form polytopes (Hoda, Gilpin, Peña, and Sandholm, 2010). Specifically, a dilated DGF for a sequence-form polytope is constructed by taking a weighted sum over suitable *local* regularizers $\varphi_{\emptyset} : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ and $\varphi_j : \Delta^{A_j} \rightarrow \mathbb{R}_{\geq 0}$ for all decision nodes $j \in \mathcal{J}$, and is of the form

$$\varphi : \mathcal{Q} \ni \mathbf{x} \mapsto \alpha_{\emptyset} \varphi_{\emptyset}(\mathbf{x}[\emptyset]) + \sum_{j \in \mathcal{J}} \alpha_j \varphi_j^{\square}(\mathbf{x}[p_j], (\mathbf{x}[ja])_{a \in A_j}), \quad (5.10)$$

where

$$\varphi_j^{\square} : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}^{A_j} \rightarrow \mathbb{R}, \quad \varphi_j^{\square}(p, \mathbf{z}) := \begin{cases} 0 & \text{if } p = 0 \\ p \cdot \varphi_j\left(\frac{\mathbf{z}}{p}\right) & \text{otherwise.} \end{cases} \quad (5.11)$$

Each local function $\varphi_j : \Delta^{A_j} \rightarrow \mathbb{R}$ is assumed to be continuously differentiable and strongly

convex modulus one on the relative interior of the probability simplex Δ^{A_j} . By dividing $(\mathbf{x}[ja])_{a \in A_j}$ by $\mathbf{x}[p_j]$ in (5.11), we renormalize $(\mathbf{x}[ja])_{a \in A_j}$ to the simplex, measure the DGF there, and then scale that value back by $\mathbf{x}[p_j]$. Finally, the weight α_j is a flexible weight term that can be chosen to ensure good properties. Hoda, Gilpin, Peña, and Sandholm (2010) showed that if each local DGF φ_j is strongly convex, then the dilated DGF φ is also strongly convex (although they do not give an explicit modulus), and they show that the associated smoothed support function can easily be computed, provided that the smoothed support function for each φ_j can easily be computed. For dilated DGFs, the strongest general result on the strong-convexity modulus comes from Farina, Kroer, and Sandholm (2019b), where the authors show that if each local DGF φ_j is strongly convex modulus one with respect to the ℓ_2 norm, and the weights of the decision nodes are set recursively according to $\alpha_j = 2 + 2 \max_{a \in A_j} \sum_{j' \in \mathcal{C}_{j_a}} \alpha_{j'}$ (so, in particular decision nodes without descendants have a weight of 2), then φ is strongly convex modulus one with respect to the ℓ_2 norm on \mathcal{Q} .

The local DGFs must be chosen so that they are compatible with the relative interior of the simplex. For a given simplex Δ^k , these are usually chosen either as:

- the (negative) entropy DGF, $\log k + \sum_{i=1}^k \mathbf{y}[i] \log \mathbf{y}[i]$, where we let $\mathbf{y}[i] \log \mathbf{y}[i] = 0$ whenever $\mathbf{y}[i] = 0$; or
- the Euclidean DGF, $\frac{1}{2} \sum_{i=1}^k (\mathbf{y}[i] - 1/k)^2$. These are both 1-strongly convex on $\text{relint } \Delta^k$ (for entropy with respect to the ℓ_1 norm and for Euclidean with respect to the ℓ_2 norm), and their associated smoothed support functions can be computed in $\mathcal{O}(k)$ time (see, e.g., Ahron Ben-Tal and Nemirovski (2001) and Condat (2016)).

5.4.1.1 “Nice”ness of dilated distance-generating functions

One of the most important properties of dilated DGFs is that they lead to a “nice” DGF as long as each local convex conjugate gradient $\nabla \varphi_j^*$ can be computed in time linear in $|A_j|$.^[5.c] Specifically, the gradient of a dilated DGF and of its convex conjugate can be computed exactly in closed form by combining the gradients of each φ_j and their convex conjugates, as shown in Algorithm 5.5.

5.4.1.2 Dilated entropy distance-generating function

As mentioned, the dilated entropy DGF is the instantiation of the general dilated DGF framework of Section 5.4.1 with the particular choice of using the (negative) entropy function at each decision node. In particular, for any choice of weights $\alpha_\emptyset, \alpha_j > 0$ it is the regularizer of the form^[5.d]

^[5.c]In particular, this makes the dilated entropy and dilated Euclidean DGFs “nice” DGFs.

^[5.d]In this dissertation, we let $0 \log(0) = 0 \log(0/0) = 0$. Since the dilated entropy DGF is a Legendre function, it is guaranteed that all iterates and prox-steps will remain in the relative interior of the optimization domain at all times, thus avoiding the non-differentiability issue of the entropy function at the boundary of \mathcal{Q} .

Algorithm 5.5: Gradient and smoothed support function for general dilated DGFs.

<pre> 1 function Gradient($x \in \text{relint } Q$) 2 $g \leftarrow \mathbf{0} \in \mathbb{R}^\Sigma$ 3 for $j \in \mathcal{J}$ in bottom-up order do 4 $b \in \Delta^{\mathcal{A}_j} \leftarrow \left(\frac{x[ja]}{x[p_j]} \right)_{a \in \mathcal{A}_j}$ 5 for $a \in \mathcal{A}_j$ do 6 $g[ja] \leftarrow g[ja] + \alpha_j \nabla \varphi_j(b)$ 7 $g[p_j] \leftarrow g[p_j] + \alpha_j (\varphi_j(b) - \langle \nabla \varphi_j(b), b \rangle)$ 8 $g[\emptyset] \leftarrow g[\emptyset] + \alpha_\emptyset \nabla \varphi_\emptyset(x[\emptyset])$ 9 return g </pre>	<pre> 10 function ConjugateGradient($g \in \mathbb{R}^\Sigma$) 11 $z \leftarrow \mathbf{0} \in \mathbb{R}^\Sigma$ 12 $z[\emptyset] \leftarrow 1$ 13 for $j \in \mathcal{J}$ in bottom-up order do 14 $(z[ja])_{a \in \mathcal{A}_j} \leftarrow \nabla \varphi_j^*(g[ja])_{a \in \mathcal{A}_j}$ 15 $g[p_j] \leftarrow g[p_j] - \varphi_j((z[ja])_{a \in \mathcal{A}_j}) + \sum_{a \in \mathcal{A}_j} g[ja] z[ja]$ 16 for $j \in \mathcal{J}$ in top-down order do 17 for $a \in \mathcal{A}_j$ do 18 $z[ja] \leftarrow z[p_j] \cdot z[ja]$ 19 return $z \in \mathcal{Q}$ </pre>
---	---

$$\mathcal{Q} \ni \mathbf{x} \mapsto \alpha_\emptyset \mathbf{x}[\emptyset] \log \mathbf{x}[\emptyset] + \sum_{j \in \mathcal{J}} \alpha_j \left(\mathbf{x}[p_j] \log |\mathcal{A}_j| + \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] \log \frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} \right). \quad (5.12)$$

We will now briefly review existing results specific to the dilated entropy DGF, for which stronger results are known than for the general class of dilated DGFs. A central result in the present paper is to show that there exist DGFs for sequence-form polytopes which are better than the dilated entropy DGF, but that these DGFs can be partially recast in a dilated form, in order to enable efficient computation of the smoothed support function.

First, as a direct consequence of the more general discussion in [Section 5.4.1](#) and [Algorithm 5.5](#), the dilated entropy DGF is a “nice” DGF (in the precise sense of [Definition 5.2](#)) no matter the choice of weights α . In particular, in the case of the negative entropy functions $\varphi_j(\mathbf{x}) := \log |\mathcal{A}_j| + \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] \log \mathbf{x}[ja]$, one has

$$(\nabla \varphi_j(\mathbf{x}))[a] = 1 + \log \mathbf{x}[a], \quad (\nabla \varphi_j^*(\mathbf{g}))[a] = \frac{e^{g_a}}{\sum_{a' \in \mathcal{A}_j} e^{g_{a'}}}$$

for all $a \in \mathcal{A}_j$, $\mathbf{x} \in \text{relint } \Delta^{\mathcal{A}_j}$, and $\mathbf{g} \in \mathbb{R}^{\mathcal{A}_j}$. By plugging the above expression in the template of [Algorithm 5.5](#) we obtain linear-time exact algorithms to compute $\nabla \varphi$ and $\nabla \varphi^*$.

Kroer, Waugh, Kılınç-Karzan, and Sandholm (2020) show that the dilated entropy DGF is strongly convex modulus $1/M_{\mathcal{Q}}$ with respect to the ℓ_1 norm, when the weights α are chosen as in the following definition.

Definition 5.5 (Kroer et al. dilated entropy DGF). Define the DGF weights β_j recursively as

$$\beta_\emptyset := 2 + 2 \sum_{j \in \mathcal{C}_\emptyset} \beta_j, \quad \beta_j := 2 + 2 \max_{a \in \mathcal{A}_j} \sum_{j' \in \mathcal{C}_{ja}} \beta_{j'} \quad \forall j \in \mathcal{J}.$$

The resulting instantiation of the dilated entropy DGF is the *Kroer et al. dilated entropy DGF*

$$\kappa : \mathbf{x} \mapsto \beta_{\emptyset} \mathbf{x}[\emptyset] \log \mathbf{x}[\emptyset] + \sum_{j \in \mathcal{J}} \beta_j \left(\mathbf{x}[p_j] \log |\mathcal{A}_j| + \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] \log \frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} \right).$$

Example 5.1. Consider the small tree-form decision process introduced in [Example 2.7](#), reproduced below. In this case, the weights β in the construction of the Kroer et al.'s dilated entropy DGF φ are computed as follows: $\beta_D = \beta_C = \beta_B = 2$; $\beta_A = 2 + 2 \max\{\beta_B + \beta_C, \beta_D\} = 10$; $\beta_{\emptyset} = 2 + 2\beta_A = 22$. Correspondingly, the DGF φ is the function

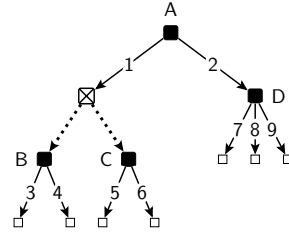
$$\kappa : \mathbf{x} \mapsto 22 \mathbf{x}[\emptyset] \log \mathbf{x}[\emptyset]$$

$$+ 10 \left(\mathbf{x}[\emptyset] \log(2) + \mathbf{x}[A1] \log \frac{\mathbf{x}[A1]}{\mathbf{x}[\emptyset]} + \mathbf{x}[A2] \log \frac{\mathbf{x}[A2]}{\mathbf{x}[\emptyset]} \right)$$

$$+ 2 \left(\mathbf{x}[A1] \log(2) + \mathbf{x}[B3] \log \frac{\mathbf{x}[B3]}{\mathbf{x}[A1]} + \mathbf{x}[B4] \log \frac{\mathbf{x}[B4]}{\mathbf{x}[A1]} \right)$$

$$+ 2 \left(\mathbf{x}[A1] \log(2) + \mathbf{x}[C5] \log \frac{\mathbf{x}[C5]}{\mathbf{x}[A1]} + \mathbf{x}[C6] \log \frac{\mathbf{x}[C6]}{\mathbf{x}[A1]} \right)$$

$$+ 2 \left(\mathbf{x}[A2] \log(2) + \mathbf{x}[D7] \log \frac{\mathbf{x}[D7]}{\mathbf{x}[A2]} + \mathbf{x}[D8] \log \frac{\mathbf{x}[D8]}{\mathbf{x}[A2]} + \mathbf{x}[D9] \log \frac{\mathbf{x}[D9]}{\mathbf{x}[A2]} \right).$$



We remark that, on the surface, the strong convexity modulus of $\frac{1}{M_Q}$ with respect to the ℓ_1 norm might appear less appealing than the modulus 1 obtained by using the ℓ_2 norm. However, recall that the norm that is used to measure strong convexity affects the value of the operator norm of \mathbf{A} , which is significantly smaller under the $\ell_1 - \ell_\infty$ operator norm (where it is equal to $\max_{r,c} |\mathbf{A}[r,c]|$) than the $\ell_2 - \ell_2$ operator norm for strong convexity with respect to the ℓ_2 norm.

Unfortunately, due to the exponential nature of the weights defined in [Definition 5.5](#), the only known bound on the diameter of \mathcal{Q} under the Kroer et al. dilated entropy DGF is exponential.

Proposition 5.2 (Kroer, Waugh, Kılınç-Karzan, and Sandholm, 2020). The diameter of \mathcal{Q} under the Kroer et al. dilated entropy DGF κ satisfies

$$\Omega_{\kappa, \mathcal{Q}} \leq 2^{2^{\mathcal{Q}}+2} M_{\mathcal{Q}}^2 \max_{j' \in \mathcal{J}} \log |\mathcal{A}_{j'}|.$$

In fact, this drawback is actually shared by all known analyses of dilated DGFs, not just dilated entropy (Farina, Kroer, and Sandholm, 2019b).

5.4.2 Dilatable global entropy distance-generating function

As mentioned at the end of the previous subsection, one drawback of both the general and entropy-specific dilated DGFs developed in the past is that they have an exponential dependence on the depth of the sequence-form polytope. In particular, note that the factor of 2 in the recursive definition of the weights means that the factor β_j for some root decision node is growing at least on the order of $2^{2^{\mathfrak{D}_Q}}$, where \mathfrak{D}_Q is the depth of the tree-form decision process. For some sequence-form polytopes this might be acceptable: if the tree-form decision process is reasonably balanced, then the number of decision nodes is also exponential in depth. However, for other sequence-form polytopes this is unacceptable: the most extreme case would be a single line of decision nodes, where the number of decision nodes is linear in \mathfrak{D}_Q , but the β_j at the root is exponentially large. This exponential dependence on depth also enters the convergence rate of the optimization methods, since it effectively acts as a scalar on the polytope diameter Ω induced by \mathfrak{D}_Q . Motivated by the need to soundly resolve that drawback, in this subsection we introduce the first “nice” DGF (in the sense of [Definition 5.2](#)) with guaranteed polynomially-small diameter for any decision node.

Definition 5.6 (Dilatable global entropy). The *dilatable global entropy distance generating function* ψ is the function $\psi : \mathcal{Q} \rightarrow \mathbb{R}_{\geq 0}$ defined as

$$\psi : \mathcal{Q} \ni \mathbf{x} \mapsto w_{\emptyset} \mathbf{x}[\emptyset] \log(\mathbf{x}[\emptyset]) + \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} w_{ja} \mathbf{x}[ja] \log \mathbf{x}[ja] + \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log |\mathcal{A}_j|,$$

where each $\gamma_j \geq 1$ ($j \in \mathcal{J}$) is defined recursively as

$$\gamma_{\emptyset} = 1 + \sum_{j \in \mathcal{C}_{\emptyset}} \gamma_j, \quad \gamma_j := 1 + \max_{a \in \mathcal{A}_j} \left\{ \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} \right\} \quad \forall j \in \mathcal{J}, \quad (5.13)$$

and each $w_{\sigma} \geq 1$ ($\sigma \in \Sigma$) is defined recursively as

$$w_{\emptyset} := \gamma_{\emptyset} - \sum_{j \in \mathcal{C}_{\emptyset}} \gamma_j,$$

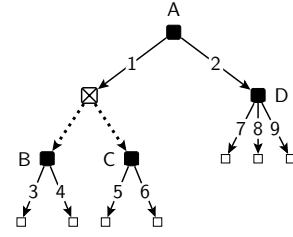
$$w_{ja} := \gamma_j - \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} = 1 + \max_{a' \in \mathcal{A}_j} \left\{ \sum_{j' \in \mathcal{C}_{ja'}} \gamma_{j'} \right\} - \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} \quad \forall ja \in \Sigma.$$

The weights γ_j defined in (5.13) are very similar to the ones given for the dilated DGFs in the previous section, except that the whole expression is smaller by a factor of two. Avoiding this factor of two is crucial, because it allows us to avoid the exponential dependence on depth. Here,

it is easy to see that γ_j is upper bounded by the number of decision nodes in the subtree rooted at j , so γ_j is at most polynomial in the size of the sequential decision problem. In fact, it is not hard to show that if j is the sole root decision node, then γ_j is equal to $\max_{\mathbf{x} \in \mathcal{Q}} \|\mathbf{x}\|_1$.

Example 5.2. We continue the example started in Example 5.1, this time looking at the *dilatable* global entropy function. In this case, the weights γ in the construction of the dilatable global entropy DGF ψ are computed as follows: $\gamma_D = \gamma_C = \gamma_B = 1$; $\gamma_A = 1 + \max\{\gamma_B + \gamma_C, \gamma_D\} = 3$; $\gamma_\emptyset = 1 + \gamma_A = 4$. Correspondingly, all weights w_{ja} are equal to 1 except for $w_{A2} = \gamma_A - \gamma_D = 2$. Correspondingly, the DGF φ is the function

$$\begin{aligned} \psi(\mathbf{x}) = & \mathbf{x}[\emptyset] \log \mathbf{x}[\emptyset] \\ & + \mathbf{x}[A1] \log \mathbf{x}[A1] + 2\mathbf{x}[A2] \log \mathbf{x}[A2] \\ & + \mathbf{x}[B3] \log \mathbf{x}[B3] + \mathbf{x}[B4] \log \mathbf{x}[B4] \\ & + \mathbf{x}[C5] \log \mathbf{x}[C5] + \mathbf{x}[C6] \log \mathbf{x}[C6] \\ & + \mathbf{x}[D7] \log \mathbf{x}[D7] + \mathbf{x}[D8] \log \mathbf{x}[D8] + \mathbf{x}[D9] \log \mathbf{x}[D9] \\ & + 3\mathbf{x}[\emptyset] \log(2) + \mathbf{x}[A1] \log(2) + \mathbf{x}[A2] \log(2) \\ & + \mathbf{x}[A2] \log(2). \end{aligned}$$



Small TFDP considered in this example.

5.4.2.1 Dilatability property

The adjective *dilatable* comes from the key property that the dilatable global entropy is equal to a specific dilated entropy regularizer $\tilde{\psi}$, on the sequence-form strategy space \mathcal{Q} . More precisely, consider the dilated entropy DGF defined as

$$\tilde{\psi} : \mathcal{Q} \ni \mathbf{x} \mapsto \gamma_\emptyset \mathbf{x}[\emptyset] \log \mathbf{x}[\emptyset] + \sum_{j \in \mathcal{J}} \gamma_j \left(\mathbf{x}[p_j] \log |\mathcal{A}_j| + \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] \log \frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} \right).$$

Then, we have the following.

Theorem 5.3. The dilatable global entropy DGF and the dilated entropy DGF coincide on the polytope of sequence-form strategies \mathcal{Q} , that is, $\psi(\mathbf{x}) = \tilde{\psi}(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{Q}$.

Proof. We start by expanding the definition of $\tilde{\psi}(\mathbf{x})$ for $\mathbf{x} \in \mathcal{Q}$:

$$\tilde{\psi}(\mathbf{x}) := \gamma_\emptyset \mathbf{x}[\emptyset] \log \mathbf{x}[\emptyset] + \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \gamma_j \mathbf{x}[ja] \log \frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} + \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log |\mathcal{A}_j|$$

$$= \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \gamma_j \mathbf{x}[ja] \log \mathbf{x}[ja] - \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \gamma_j \mathbf{x}[ja] \log \mathbf{x}[p_j] + \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log |\mathcal{A}_j|, \quad (5.14)$$

where in the second equality we have used the fact that $\mathbf{x}[\emptyset] = 1$, as well as the properties of logarithms. Given the assumption $\mathbf{x} \in \mathcal{Q}$, it holds that $\sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] = \mathbf{x}[p_j]$ for all $j \in \mathcal{J}$ and so we can simplify the middle summation in (5.14) and obtain

$$\tilde{\psi}(\mathbf{x}) = \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \gamma_j \mathbf{x}[ja] \log \mathbf{x}[ja] - \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log \mathbf{x}[p_j] + \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log |\mathcal{A}_j|.$$

The middle summation multiplies the weights γ_j by a quantity that depends on the parent sequence p_j of j . It can therefore be rewritten equivalently by summing over nonterminal sequences and multiplying by the weight of the children decision nodes, as follows.

$$\begin{aligned} \tilde{\psi}(\mathbf{x}) &= \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \gamma_j \mathbf{x}[ja] \log \mathbf{x}[ja] - \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \sum_{j' \in \mathcal{C}_{j_a}} \gamma_{j'} \mathbf{x}[ja] \log \mathbf{x}[ja] + \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log |\mathcal{A}_j| \\ &= \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \gamma_j \mathbf{x}[ja] \log \mathbf{x}[ja] - \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \left(\sum_{j' \in \mathcal{C}_{j_a}} \gamma_{j'} \right) \mathbf{x}[ja] \log \mathbf{x}[ja] + \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log |\mathcal{A}_j| \\ &= \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} w_{j_a} \mathbf{x}[ja] \log \mathbf{x}[ja] + \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log |\mathcal{A}_j| \\ &= \psi(\mathbf{x}), \end{aligned}$$

as we wanted to show. \square

The equality established in [Theorem 5.3](#) does not hold outside the sequence-form polytope—a set with no interior. Because of that, it is not surprising that in general $\nabla \psi(\mathbf{x}) \neq \nabla \tilde{\psi}(\mathbf{x})$ even at points \mathbf{x} in the relative interior of the sequence-form polytope. We will return to the difference between the gradients of ψ and $\tilde{\psi}$ in [Section 5.4.2.5](#), where we show that the mismatch between $\nabla \psi(\mathbf{x})$ and $\nabla \tilde{\psi}(\mathbf{x})$ is orthogonal to the sequence-form polytope when $\mathbf{x} \in \mathcal{Q}$.

5.4.2.2 “Nice”ness of the dilatable global entropy DGF

We now show that our dilatable global entropy regularizer is “nice” in the sense of [Definition 5.2](#), that is, its gradient and the gradient of its convex conjugate can be computed exactly in linear time in $|\Sigma|$.

- The gradient of ψ can be trivially computed in closed form and linear time in $|\Sigma|$ starting from [Definition 5.6](#) as

$$\left(\nabla \psi(\mathbf{x}) \right) [\sigma] = (1 + \log \mathbf{x}[\sigma]) w_\sigma + \sum_{j \in \mathcal{C}_\sigma} \gamma_j \log |\mathcal{A}_j| \quad \forall \sigma \in \Sigma, \mathbf{x} \in \text{relint } \mathcal{Q}.$$

- Using the dilatability property, we have that the gradient of the convex conjugate satisfies

$$\nabla\psi^*(\mathbf{g}) = \arg \max_{\mathbf{x} \in \mathcal{Q}} \{\langle \mathbf{g}, \mathbf{x} \rangle - \psi(\mathbf{x})\} = \arg \max_{\mathbf{x} \in \mathcal{Q}} \{\langle \mathbf{g}, \mathbf{x} \rangle - \tilde{\psi}(\mathbf{x})\} = \nabla\tilde{\psi}^*(\mathbf{g}), \quad (5.15)$$

where we used the dilatability property ([Theorem 5.3](#)) in the second equality. Therefore, since $\tilde{\psi}$ is a dilated DGF and its smoothed support function can be computed in linear time, the smoothed support function of ψ can be computed in linear time in $|\Sigma|$.

An immediate consequence of the niceness established in the previous bullet point is that proximal operators of ψ can be computed efficiently, as

$$\text{prox}_{\psi}(\mathbf{g} \parallel \mathbf{c}) = \nabla\psi^*(-\mathbf{g} + \nabla\psi(\mathbf{c})) = \nabla\tilde{\psi}^*(-\mathbf{g} + \nabla\psi(\mathbf{c})),$$

where the last equality is a special case of [\(5.15\)](#).

5.4.2.3 Strong convexity of the dilatable global entropy DGF

On the other hand, we now show that ψ has the advantage of a better strong convex modulus, compared to the existing dilated entropy DGFs.

Theorem 5.4. The dilatable global entropy function $\psi : \mathcal{Q} \rightarrow \mathbb{R}_{\geq 0}$ is a DGF for the sequence-form polytope \mathcal{Q} , 1-strongly convex on $\text{relint } \mathcal{Q}$ with respect to the ℓ_2 norm.

Proof. The function ψ is twice-differentiable on $(0, 1)^\Sigma \supseteq \text{relint } \mathcal{Q}$. Using [\(5.1\)](#) we conclude that ψ is 1-strongly convex, since the Hessian is

$$\nabla^2\psi(\mathbf{x}) = \text{diag}\left(\left\{\frac{w_\sigma}{\mathbf{x}[\sigma]}\right\}_{\sigma \in \Sigma}\right) \succcurlyeq I,$$

where we used the inequalities $0 \leq \mathbf{x}[\sigma] \leq 1$ and $w_\sigma \geq 1$. □

Theorem 5.5. The dilatable global entropy function ψ is strongly convex modulus $1/M_{\mathcal{Q}}$ with respect to the ℓ_1 norm on $\text{relint } \mathcal{Q}$.

Proof. Using the second-order definition of strong convexity, we wish to show that the inequality $\langle \mathbf{m}, \nabla^2\psi(\mathbf{x})\mathbf{m} \rangle \geq \frac{1}{M_{\mathcal{Q}}} \|\mathbf{m}\|_1^2$ holds for any $\mathbf{m} \in \mathbb{R}^\Sigma$. Expanding the Hessian matrix and using the fact that $w_\sigma \geq 1$ for all $\sigma \in \Sigma$ gives

$$\langle \mathbf{m}, \nabla^2 \psi(\mathbf{x}) \mathbf{m} \rangle = \left\langle \mathbf{m}, \text{diag} \left(\left\{ \frac{w_\sigma}{\mathbf{x}[\sigma]} \right\}_{\sigma \in \Sigma} \right) \mathbf{m} \right\rangle \geq \sum_{\sigma \in \Sigma} \frac{\mathbf{m}[\sigma]^2}{\mathbf{x}[\sigma]}. \quad (5.16)$$

On the other hand, by expanding the definition of $\|\mathbf{m}\|_1^2$ and applying the Cauchy-Schwarz inequality, we have

$$\begin{aligned} \|\mathbf{m}\|_1^2 &= \left(\sum_{\sigma \in \Sigma} |\mathbf{m}[\sigma]| \right)^2 = \left(\sum_{\sigma \in \Sigma} \frac{|\mathbf{m}[\sigma]|}{\sqrt{\mathbf{x}[\sigma]}} \sqrt{\mathbf{x}[\sigma]} \right)^2 \\ &\leq \left(\sum_{\sigma \in \Sigma} \frac{\mathbf{m}[\sigma]^2}{\mathbf{x}[\sigma]} \right) \left(\sum_{\sigma \in \Sigma} \mathbf{x}[\sigma] \right) = \left(\sum_{\sigma \in \Sigma} \frac{\mathbf{m}[\sigma]^2}{\mathbf{x}[\sigma]} \right) \|\mathbf{x}\|_1 \leq \left(\sum_{\sigma \in \Sigma} \frac{\mathbf{m}[\sigma]^2}{\mathbf{x}[\sigma]} \right) M_Q. \end{aligned}$$

Substituting (5.16) into the last inequality yields a proof of the desired strong convexity modulus $1/M_Q$. \square

5.4.2.4 Diameter of the dilatable global entropy DGF

The properties above immediately imply that our dilatable global entropy DGF satisfies all the requirements for a prox setup on the polytope of sequence-form strategies \mathcal{Q} . Here we complete the analysis by giving bounds on the diameter induced by ψ .

Theorem 5.6. The ψ -diameter $\Omega_{\psi, \mathcal{Q}}$ of \mathcal{Q} is at most $M_Q^2 \max_{j' \in \mathcal{J}} \log |\mathcal{A}_{j'}|$.

Proof. By the definition of the polytope diameter and the fact that we chose our DGFs such that $\min_{\mathbf{x} \in \mathcal{A}} \psi(\mathbf{x}) = 0$, we have

$$\begin{aligned} \Omega_{\psi, \mathcal{Q}} &\leq \max_{\mathbf{x} \in \mathcal{Q}} \psi(\mathbf{x}) \leq \max_{\mathbf{x} \in \mathcal{Q}} \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \log |\mathcal{A}_j| \leq \max_{\mathbf{x} \in \mathcal{Q}} \max_{j' \in \mathcal{J}} \log |\mathcal{A}_{j'}| \sum_{j \in \mathcal{J}} \gamma_j \mathbf{x}[p_j] \\ &\leq M_Q \max_{\mathbf{x} \in \mathcal{Q}} \max_{j' \in \mathcal{J}} \log |\mathcal{A}_{j'}| \sum_{j \in \mathcal{J}} \mathbf{x}[p_j] \\ &\leq M_Q^2 \max_{j' \in \mathcal{J}} \log |\mathcal{A}_{j'}|, \end{aligned}$$

where the second inequality is by noting that $\log \mathbf{x}[\sigma] \leq 0$ since $\mathbf{x}[\sigma] \leq 1$ for all $\sigma \in \Sigma$, the fourth inequality is by noting that γ_j is largest at root decision nodes, where it is at most M_Q , and the fifth inequality upper bounds $\sum_{j \in \mathcal{J}} \mathbf{x}[p_j]$ by M_Q . \square

In particular, [Theorem 5.6](#) shows that the dilatable global entropy DGF improves the polytope diameter of dilated entropy ([Proposition 5.2](#)) by a factor of $2^{\mathcal{D}_Q+2}$, achieving a polytope diameter with no exponential dependence on the depth \mathcal{D}_Q of the sequence-form polytope.

Example 5.3. Let \mathcal{Q} be the sequence-form polytope corresponding to an optimal stopping problem: an agent has a sequence of k decision nodes, and at each decision node they can either choose to stop the game, in which case a payoff is reached, or they can choose to continue the game, in which case they continue to the next decision node. For such a problem, \mathcal{Q} has depth k , which is also the size of \mathcal{Q} as measured by the ℓ_1 norm, i.e. $M_{\mathcal{Q}} = k$. In this case, the diameter associated to ψ is $k^2 \log 2$, whereas the dilated entropy DGF of Kroer, Waugh, Kılınç-Karzan, and Sandholm (2020) gives diameter $2^{k+2} k^2 \log 2$.

Example 5.3 shows the most extreme type of decision problem in terms of our improvement over existing results. Another example of a polytope that partially embeds this structure is the decision spaces of a no-limit poker game. In those games, a player may raise by any amount between the minimum raise size and the size of their stack. Because of this, there exists a very deep decision path where players take turns raising by the minimum amount. However, the average depth is much smaller than this.

Summing up our results on the dilatable global entropy, we have shown that it enjoys the same fast smoothed-support-function computation as the dilated entropy DGF, while having a better way to achieve strong convexity modulus $1/M_{\mathcal{Q}}$. In particular, the existing dilated entropy setup requires the weight parameters β to grow exponentially in the depth of the sequence-form polytope, whereas we have only a linear growth in those weights. More concretely, this means that the largest weights $\max_{j \in \mathcal{J}} \beta_j$ in the dilated entropy DGF are larger than the largest weights $\max_{j \in \mathcal{J}} \gamma_j$ in the dilatable global entropy DGF by a factor of more than $2^{\mathfrak{D}_{\mathcal{Q}}}$. This in turn allowed us to achieve a better polytope diameter by a factor of $2^{\mathfrak{D}_{\mathcal{Q}}+2}$ while retaining the same strong convexity modulus.

5.4.2.5 Gradient properties of the dilatable global entropy DGF

The previous subsections establish that the dilated global entropy DGF $\tilde{\psi}$ is dilatable (*i.e.*, it coincides with a dilated entropy DGF, $\tilde{\psi}$, on all of their domain \mathcal{Q}) and nice (*i.e.*, gradients, smoothed support functions, and proximal operators can be computed efficiently). Furthermore, we have seen that $\tilde{\psi}$ has very appealing metric properties, including a *diagonal* Hessian matrix that renders establishing strong convexity rather trivial, especially as compared to the analyses of dilated entropy DGFs in the prior literature (Kroer, Waugh, Kılınç-Karzan, and Sandholm, 2020).

In this subsection we go one step further: by studying the relationship between the gradients of ψ and $\tilde{\psi}$, we will establish that the two enjoy the same strong convexity parameter. In particular, we will make use of the following property, which we prove at the end of the section.

Lemma 5.2. At any point $\mathbf{x} \in \text{relint } \mathcal{Q}$, the gradients $\nabla\psi(\mathbf{x})$ and $\nabla\tilde{\psi}(\mathbf{x})$ differ only along orthogonal directions to the affine hull $\text{aff } \mathcal{Q}$ of \mathcal{Q} , that is,

$$\langle \nabla\psi(\mathbf{x}) - \nabla\tilde{\psi}(\mathbf{x}), \mathbf{y} - \mathbf{z} \rangle = 0 \quad \forall \mathbf{y}, \mathbf{z} \in \mathcal{Q}.$$

Lemma 5.2 immediately implies that for any two points $\mathbf{x}, \mathbf{x}' \in \text{relint } \mathcal{Q}$

$$\langle \nabla\tilde{\psi}(\mathbf{x}) - \nabla\tilde{\psi}(\mathbf{x}'), \mathbf{x} - \mathbf{x}' \rangle = \langle \nabla\psi(\mathbf{x}) - \nabla\psi(\mathbf{x}'), \mathbf{x} - \mathbf{x}' \rangle,$$

that is, the strong convexity properties of ψ established in Section 5.4.2.3 apply to $\tilde{\psi}$ as well. In particular, this shows that the specific dilated entropy DGE $\tilde{\psi}$ is 1-strongly convex on $\text{relint } \mathcal{Q}$ with respect to the ℓ_2 norm, and $1/M_{\mathcal{Q}}$ -strongly convex with respect to the ℓ_1 norm. Additionally, the analysis of the diameter of ψ transfers directly to $\tilde{\psi}$ by dilatability (Section 5.4.2.1). Combined, these results enable us to introduce a 1-strongly convex *dilated entropy* DGF (and not just any nice DGF) $\tilde{\psi}$ that for the first time has worst-case polynomially-bounded diameter.

We remark that it would be hard to establish such a result without first introducing our dilated global regularizer, as analyzing the product $\langle \nabla\tilde{\psi}(\mathbf{x}) - \nabla\tilde{\psi}(\mathbf{x}'), \mathbf{x} - \mathbf{x}' \rangle$ for a dilated entropy DGF $\tilde{\psi}$ is notoriously involved. Instead, the good metric properties of ψ —especially the diagonal Hessian matrix—make the analysis of the strong convexity and diameter properties trivial.

In fact, every strong convexity proof for an entropy DGF that we are aware of uses the Hessian-based condition (Ahron Ben-Tal and Nemirovski, 2001; Kroer, Waugh, Kılınç-Karzan, and Sandholm, 2020). If we had used this definition directly on the dilated DGF φ , then the lower bound from Kroer, Waugh, Kılınç-Karzan, and Sandholm (2020) implies that we would get an exponential dependence on the depth. This is because the Hessian condition is a sufficient but not necessary condition for strong convexity: it asks for the condition to hold along directions orthogonal to the affine hull of the sequence-form strategy space, which is not necessary. Arguably, the key insight for the dilatable global entropy is that it allows us to use the Hessian-based sufficient condition for proving strong convexity in a way that avoids paying the cost of maintaining strong convexity along irrelevant orthogonal directions. The fact that it also greatly simplifies the Hessian-based analysis is an added bonus.

Another consequence of Lemma 5.2 is that the proximal steps induced by ψ and $\tilde{\psi}$ coincide. Indeed, for all centers $\mathbf{y} \in \text{relint } \mathcal{Q}$ and gradients \mathbf{g} , one has

$$\begin{aligned} \arg \min_{\mathbf{y} \in \mathcal{Q}} \{ \langle \mathbf{g}, \mathbf{y} \rangle + D_{\psi}(\mathbf{y} \| \mathbf{x}) \} \\ = \arg \min_{\mathbf{y} \in \mathcal{Q}} \{ \langle \mathbf{g}, \mathbf{y} \rangle + \psi(\mathbf{y}) - \nabla\psi(\mathbf{x})^{\top}(\mathbf{y} - \mathbf{x}) \} \end{aligned}$$

$$\begin{aligned}
&= \arg \min_{\mathbf{y} \in \mathcal{Q}} \{ \langle \mathbf{g}, \mathbf{y} \rangle + \tilde{\psi}(\mathbf{y}) - \nabla \psi(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) \} && \text{(By dilatibility, Theorem 5.3)} \\
&= \arg \min_{\mathbf{y} \in \mathcal{Q}} \{ \langle \mathbf{g}, \mathbf{y} \rangle + \tilde{\psi}(\mathbf{y}) - \nabla \tilde{\psi}(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) \} && \text{(Lemma 5.2)} \\
&= \arg \min_{\mathbf{y} \in \mathcal{Q}} \{ \langle \mathbf{g}, \mathbf{y} \rangle + D_{\tilde{\psi}}(\mathbf{y} \parallel \mathbf{x}) \}.
\end{aligned}$$

Hence, mirror descent, online mirror descent, follow-the-regularizer-leader, mirror prox, and EGT (among others) all produce the same iterates when instantiated with $\tilde{\psi}$ or ψ . It is conceivable that some methods—for examples methods that require taking norms of gradients, or methods that rely on norms induced by the Hessian matrix of the regularizer (e.g., Abernethy and A. Rakhlin (2009))—would in general not be equivalent when set up using $\tilde{\psi}$ or ψ .

Proof of Lemma 5.2. We will prove the theorem by structural induction on the structure of the sequence-form strategy space. Specifically, we seek to prove that at all decision nodes $j \in \mathcal{J}$, one has

$$\sum_{j' \succ j} \sum_{a' \in \mathcal{A}_{j'}} \frac{\partial(\psi - \tilde{\psi})}{\partial \mathbf{x}[j'a']}(\mathbf{x}) \mathbf{y}[j'a'] = \gamma_j \mathbf{y}[p_j] \log \mathbf{x}[p_j]. \quad (5.17)$$

Since \mathbf{y} is arbitrary, the equality above will immediately imply that for any j such that $p_j = \emptyset$

$$\sum_{j' \succ j} \sum_{a' \in \mathcal{A}_{j'}} \frac{\partial(\psi - \tilde{\psi})}{\partial \mathbf{x}[j'a']}(\mathbf{x}) (\mathbf{y}[j'a'] - \mathbf{z}[j'a']) = \gamma_j (\mathbf{y}[\emptyset] - \mathbf{z}[\emptyset]) \log \mathbf{x}[\emptyset] = 0, \quad (5.18)$$

where the last equality follows from the fact that $\mathbf{y}[\emptyset] = \mathbf{z}[\emptyset] = 1$ since \mathbf{y}, \mathbf{z} are sequence-form strategies. Now, since the set of sequences can be partitioned as $\Sigma = \{\emptyset\} \sqcup \bigsqcup_{j \in \mathcal{J}: p_j = \emptyset} \{j'a' : j' \succ j, a' \in \mathcal{A}_{j'}\}$, and furthermore $\mathbf{y}[\emptyset] = \mathbf{z}[\emptyset] = 1$, it is evident that (5.18) considered for the set $\{j \in \mathcal{J} : p_j = \emptyset\}$ immediately implies the statement.

In order to prove (5.17) we will use structural induction over the structure of the sequence-form decision problem.

- **Base case:** terminal decision nodes j . Consider a terminal decision node j , that is one for which $\mathcal{C}_{ja} = \{\}$ for all $a \in \mathcal{A}_j$. Then, for all $a \in \mathcal{A}_j$ we obtain from direct inspection that

$$\frac{\partial \psi}{\partial \mathbf{x}[ja]}(\mathbf{x}) = w_{ja}(1 + \log \mathbf{x}[ja]); \quad \frac{\partial \tilde{\psi}}{\partial \mathbf{x}[ja]}(\mathbf{x}) = \gamma_j \left(1 + \log \frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} \right).$$

Hence,

$$\begin{aligned} \sum_{a \in \mathcal{A}_j} \frac{\partial(\psi - \tilde{\psi})}{\partial \mathbf{x}[ja]}(\mathbf{x}) \mathbf{y}[ja] &= \sum_{a \in \mathcal{A}_j} \left(w_{ja} - \gamma_j + (w_{ja} - \gamma_j) \log \mathbf{x}[ja] + \gamma_j \log \mathbf{x}[p_j] \right) \mathbf{y}[ja] \\ &= \gamma_j \mathbf{y}[p_j] \log \mathbf{x}[p_j], \end{aligned}$$

where we used the fact that by definition $\gamma_j = w_{ja} = 1$, as well as the fact that \mathbf{y} is a sequence-form strategy and therefore $\sum_{a \in \mathcal{A}_j} \mathbf{y}[ja] = \mathbf{y}[p_j]$. Since the only $j' \succ j$ is j itself, the proof of the base case is complete.

- **Inductive step.** Consider a nonterminal j . Again by direct inspection, we obtain

$$\begin{aligned} \frac{\partial \psi}{\partial \mathbf{x}[ja]}(\mathbf{x}) &= w_{ja}(1 + \log \mathbf{x}[ja]) + \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} \log |\mathcal{A}_{j'}|; \text{ and} \\ \frac{\partial \tilde{\psi}}{\partial \mathbf{x}[ja]}(\mathbf{x}) &= \gamma_j \left(1 + \log \frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} \right) - \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} + \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} \log |\mathcal{A}_{j'}| \\ &= w_{ja} + \gamma_j \log \mathbf{x}[ja] - \gamma_j \log \mathbf{x}[p_j] + \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} \log |\mathcal{A}_{j'}|, \end{aligned}$$

where the last equality uses the definition of $w_{ja} := \gamma_j - \sum_{j' \in \mathcal{A}_j} \gamma_{j'}$.

Now, the set of descendant decision nodes $\{j' \in \mathcal{J} : j' \succ j\}$ can be partitioned as

$$\{j\} \sqcup \bigsqcup_{a \in \mathcal{A}_j} \bigsqcup_{j' \in \mathcal{C}_{ja}} \{j'' \in \mathcal{J} : j'' \succ j'\}.$$

Correspondingly, using the inductive hypothesis we have

$$\begin{aligned} \sum_{j' \succ j} \sum_{a' \in \mathcal{A}_{j'}} \frac{\partial(\psi - \tilde{\psi})}{\partial \mathbf{x}[j'a']}(\mathbf{x}) \mathbf{y}[j'a'] &= \left(\sum_{a \in \mathcal{A}_j} \frac{\partial(\psi - \tilde{\psi})}{\partial \mathbf{x}[ja]}(\mathbf{x}) \mathbf{y}[ja] \right) + \sum_{a \in \mathcal{A}_j} \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} \mathbf{y}[ja] \log \mathbf{x}[ja] \\ &= \left(\sum_{a \in \mathcal{A}_j} \left((w_{ja} - \gamma_j) \log \mathbf{x}[ja] + \gamma_j \log \mathbf{x}[p_j] \right) \mathbf{y}[ja] \right) \\ &\quad + \sum_{a \in \mathcal{A}_j} \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} \mathbf{y}[ja] \log \mathbf{x}[ja] \\ &= \sum_{a \in \mathcal{A}_j} \left(\left(w_{ja} - \gamma_j + \sum_{j' \in \mathcal{C}_{ja}} \gamma_{j'} \right) \log \mathbf{x}[ja] + \gamma_j \log \mathbf{x}[p_j] \right) \mathbf{y}[ja] \end{aligned}$$

$$= \sum_{a \in \mathcal{A}_j} \gamma_j \mathbf{y}[ja] \log \mathbf{x}[p_j] = \gamma_j \mathbf{y}[p_j] \log \mathbf{x}[p_j],$$

where we again used the definition of $w_{ja} := \gamma_j - \sum_{j' \in \mathcal{A}_j} \gamma_{j'}$, as well as the fact that \mathbf{y} is a sequence-form strategy and therefore $\sum_{a \in \mathcal{A}_j} \mathbf{y}[ja] = \mathbf{y}[p_j]$.

This concludes the induction proof. \square

5.5 Experimental evaluation

In this section we compare the numerical performance of the dilated entropy and dilatable global entropy DGFs for computing Nash equilibria in two-player zero-sum imperfect-information extensive-form games.

Our experiments will be shown on nine different games, which span a variety of poker games, other recreational games, as well as a pursuit-evasion game played on a graph. All games are standard benchmarks in the computational game theory literature, and a full description of the games is given in [Appendix A](#). In [Table 5.1\(a\)](#) we summarize the game instances we use, as well as some of their key dimensions: the number of decision points $|\mathcal{J}_1|, |\mathcal{J}_2|$ for Player 1 and 2, respectively, the number of sequences $|\Sigma_1|, |\Sigma_2|$, and the number of terminal nodes (leaves).

Game instance	Decision Points $ \mathcal{J}_1 + \mathcal{J}_2 $	Sequences $ \Sigma_1 + \Sigma_2 $	Leaves $ \mathcal{Z} $	Weights β		Weights γ	
				Avg	Max	Avg	Max
K23 Kuhn poker	12	26	30	8.857	38	2.286	7
L2232 Leduc poker (3 ranks)	288	674	1116	11.766	680	2.117	43
L22d2 Leduc poker (13 ranks)	5148	12 014	98 956	12.057	12 320	2.131	703
G24 Goofspiel	34 952	42 658	13 824	6.912	23 440	1.698	917
B2323 Battleship (3 turns)	81 027	327 070	552 132	3.294	2890	1.242	99
B2324 Battleship (4 turns)	1 050 723	3 236 158	3 487 428	3.753	27 470	1.331	483
D26 Liar's dice	24 576	49 142	147 420	15.556	65 540	2.043	1399
P24 Pursuit-evasion (4 turns)	382	2086	15 898	8.286	60	1.943	5
P25 Pursuit-evasion (6 turns)	11 888	69 029	118 514	19.424	250	2.508	7

(A) — Game instances and sizes

(B)

(C)

Table 5.1: (A) Various measures of the size of each of the games that we test algorithms on. (B), (C) The magnitude of the dilated entropy DGF and dilatable global entropy DGF weights.

Our experiments will show performance on three algorithms. First, we will plot the performance for both the EGT and MPROX algorithms, with stepsizes and smoothing chosen according to the theoretical values dictated by [Theorems 5.1](#) and [5.2](#), with one minor variation: for each DGF, we do not scale by M_Q , which is required in order to achieve strong convexity with respect to the

ℓ_1 norm. This is done because scaling by this last factor leads to extremely slow performance for all of the DGFs. Second, we will also show results on a tweaked variant of EGT called ‘EGT/AS’, which implements several heuristics that typically lead to better performance in practice, as seen in Hoda, Gilpin, Peña, and Sandholm, 2010; Kroer, Waugh, Kılınç-Karzan, and Sandholm, 2020; Kroer, Farina, and Sandholm, 2018b. These heuristic are:

1. *μ balancing*: At each iteration, we take a step on the player i whose smoothing parameter μ_i is larger.
2. *Aggressive stepsizing*: The original stepsize of EGT at iteration t is $\tau = 2/(3 + t)$, which is typically too conservative in practice. Instead, EGT/AS maintains some current value τ , initially set at $\tau = 0.5$. EGT/AS then repeatedly attempts to take steps with the current τ , and after every step checks whether the invariant condition of EGT still holds. If not, then we undo the step, decrease τ , and repeat the process.
3. *Initial μ fitting*: The initial EGT values for μ_x, μ_y are much too conservative. Instead, At the beginning of the algorithm we perform a search over initial values for $\mu_x = \mu_y$. The search starts at the candidate value $\mu = 10^{-6}$ and stops as soon as the choice of $\mu_x = \mu_y = \mu$ yields an excessive gap value above 0.1. If the current choice does not, μ is incremented by 20% and the fitting continues.

For all parameters above, we use the same values as in Kroer, Farina, and Sandholm (2018b), even though those values were tuned for the dilated entropy DGF, rather than dilatable global entropy.

In the presentation of the numerical performance, we will generally plot the number of iterations of the FOM on the x-axis, rather than plot wall-clock time. Since we hold the algorithmic setup fixed in each plot, apart from the DGF, this gives a fair representation of performance, since they all use the same set of operations (in particular the same number of gradient computations, which is typically the most expensive operation). For EGT/AS, we will instead plot the number of gradient computations on the x-axis, since the number of gradient computations can vary for each DGF, depending on the amount of backtracking incurred.

We will focus on comparing our new dilatable global entropy for the sequence-form polytope (Definition 5.6) to the prior state-of-the-art dilated entropy DGF (Definition 5.5) from Kroer, Waugh, Kılınç-Karzan, and Sandholm, 2020.

Before we study the numerical performance, we look at the size of the DGF weights β and γ for each of the games. Table 5.1 column (b) shows the average and maximum size of the dilated entropy, and Table 5.1 column (c) shows the corresponding values for the dilatable global entropy DGF. We see that the dilatable global entropy DGF requires vastly less weight, especially in terms of the maximal weights near the root of each decision space.

5.5.1 MPROX and EGT without aggressive stepsizing

First, we study the theoretically-correct way to use the DGFs. In particular, we instantiate both EGT and mirror prox with the stepsizes and DGFs as specified in [Theorems 5.1](#) and [5.2](#), for the dilatable global entropy and dilated entropy. The results for EGT are shown in [Figures 5.1](#) and [5.2](#). Across both algorithms and all nine games, we see that our new dilatable global entropy DGF performs better, sometimes by over an order of magnitude (*e.g.*, in liar’s dice and pursuit evasion (6 turns)). This is in line with the fact that our new DGF has a better strong convexity modulus, which allows for a much smaller amount of smoothing, while still guaranteeing correctness. This in turns allows the algorithms to safely take larger steps, thereby progressing faster.

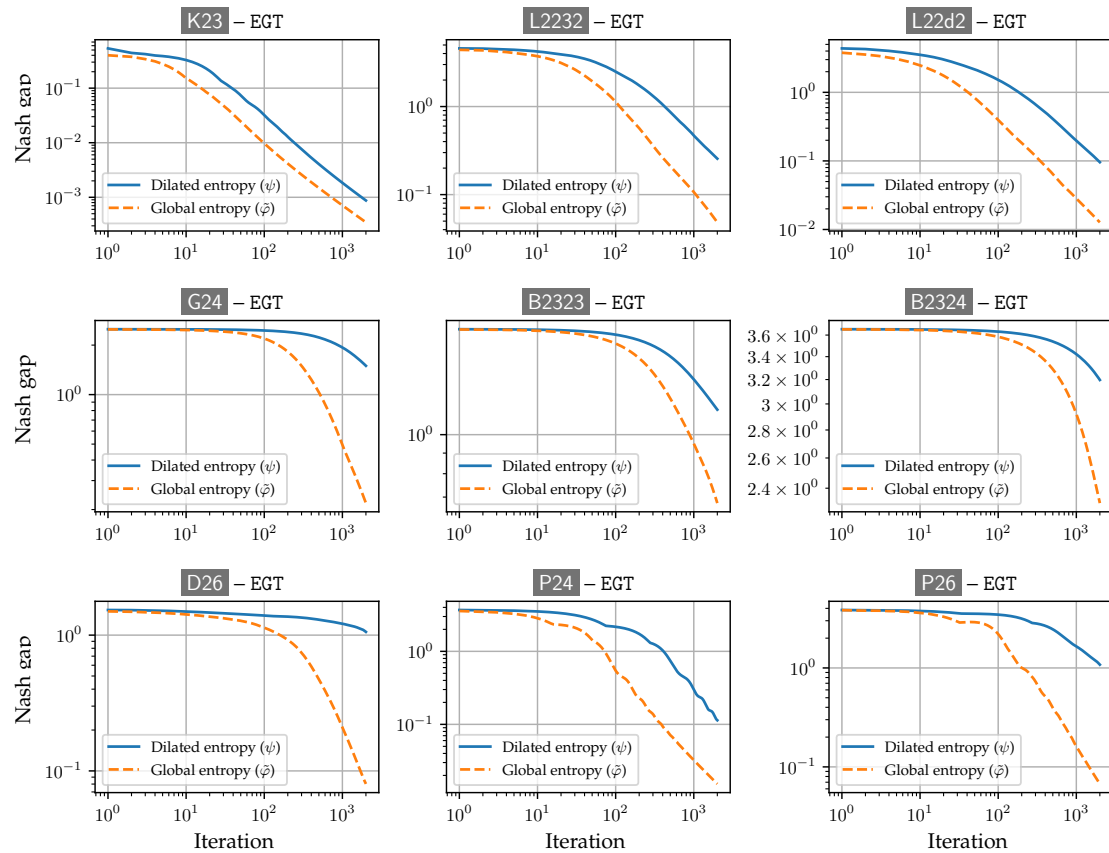


Figure 5.1: Performance of the EGT algorithm instantiated with the two entropy DGFs across nine games. The x-axis shows the number of EGT iterations, and the y-axis shows the distance to Nash equilibrium.

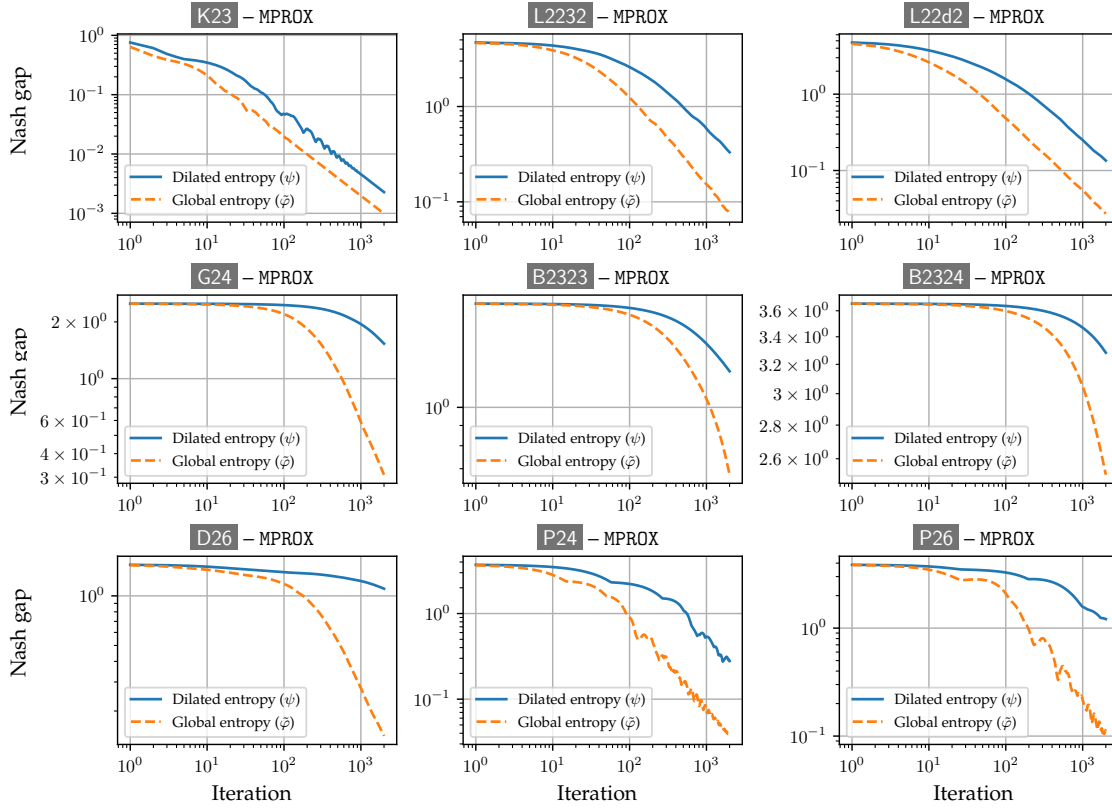


Figure 5.2: Performance of the MPROX algorithm instantiated with the two entropy DGFs across nine games.

5.5.2 EGT with aggressive stepsizing

Secondly, we investigate the numerical performance of the two entropy DGFs in the EGT/AS algorithm in Figure 5.3.

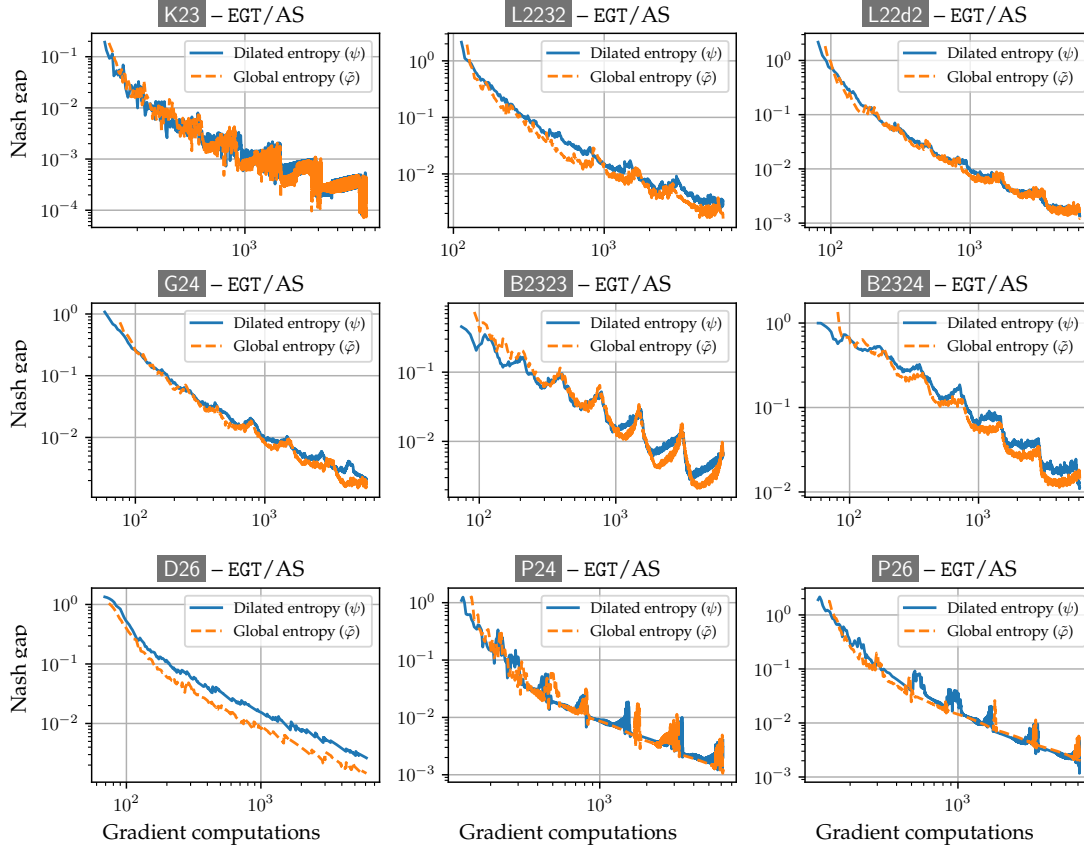


Figure 5.3: Performance of the EGT/AS algorithm instantiated with the two entropy DGFs, as well as aggressive stepsizing, μ balancing, and initial μ fitting.

Here we see a smaller performance improvement. For most of the games, we get a small factor of improvement for the first 100 or so iterations, but then the performance is similar thereafter. For liar's Dice there is a persistent improvement to using dilatable global entropy across all iterations.

5.A Appendix: Properties of SMPL functions

In this appendix we mention several basic results about strictly monotonically increasing piecewise-linear (SMPL) and quasi-SMPL functions. Proofs of elementary results are omitted.

Lemma 5.3. Let $f : I \rightarrow \mathbb{R}$ be SMPL, and consider a standard representation of f of size S . Then, for any $b \in \mathbb{R}$ and $k \geq 0$, a standard representation for the SMPL function $I \ni x \mapsto b + kf(x)$ can be computed in $\mathcal{O}(S + 1)$ time.

Lemma 5.4. The sum $f_1 + \cdots + f_n$ of n SMPL (resp., quasi-SMPL) functions $f_i : I \rightarrow \mathbb{R}$ is a SMPL (resp., quasi-SMPL) function $I \rightarrow \mathbb{R}$. Furthermore, if each f_i admits a standard representation of size S_i , then a standard representation of size at most $S_1 + \cdots + S_n$ for their sum can be computed in $\mathcal{O}(S_1 + \cdots + S_n + 1) \log n$ time.

Lemma 5.5. Let $f : \mathbb{R} \rightarrow \mathbb{R}$ be SMPL, and consider a standard representation of f of size S . Then, for any $\beta \in \mathbb{R}$, a standard representation of size at most S for the quasi-SMPL function $I \ni x \mapsto [f(x) - \beta]^+$ can be computed in $\mathcal{O}(S + 1)$ time.

Lemma 5.6. The inverse $f^{-1} : \text{range}(f) \rightarrow \mathbb{R}$ of a SMPL function $f : I \rightarrow \mathbb{R}$ is SMPL. Furthermore, if f admits a standard representation of size S , then a standard representation for f^{-1} of size at most S can be computed in $\mathcal{O}(S + 1)$ time.

Lemma 5.7. Let $f : \mathbb{R} \rightarrow [0, +\infty)$ be quasi-SMPL. The inverse $f^{-1} : (0, +\infty) \rightarrow \mathbb{R}$ of f restricted^a to domain $(0, +\infty)$ is SMPL. Furthermore, if f admits a standard representation of size S , then a standard representation of size at most S for f^{-1} can be computed in $\mathcal{O}(S + 1)$ time.

^aWe restrict the domain to $(0, +\infty)$ because $f^{-1}(0)$ may be multivalued.

Proof. We have $f(x) = [g(x)]^+$ where g is SMPL. It follows that the function $\bar{g} : I \rightarrow \mathbb{R}$ defined as $\bar{g}(x) = g(x)$ for the interval $I = \{x : g(x) > 0\}$ is SMPL as well. For any x such that $f(x) > 0$ we have $x \in I$, and thus $f^{-1} = g^{-1}$, and it follows from [Lemma 5.6](#) that f^{-1} is SMPL. \square

Lemma 5.8. Let $f : [0, +\infty) \rightarrow \mathbb{R}$ be a SMPL function, and consider the function g that maps y to the unique solution to the equation $x = [y - f(x)]^+$. Then, g is quasi-SMPL and satisfies $g(y) = [(x + f)^{-1}(y)]^+$, where $(x + f)^{-1}$ denotes the inverse of the SMPL function $x \mapsto x + f([x]^+)$.

Proof. For any $y \in \mathbb{R}$, the function $h_y : x \mapsto x - [y - f(x)]^+$ is clearly SMPL on $[0, +\infty)$. Furthermore, $h_y(0) \leq 0$ and $h_y(+\infty) = +\infty$, implying that $h_y(x) = 0$ has a unique solution. We now show that $g(y) = [(x+f)^{-1}(y)]^+$ is that solution, that is, it satisfies $g(y) = [y - f(g(y))]^+$ for all $y \in \mathbb{R}$. Fix any $y \in \mathbb{R}$ and let

$$\bar{g} := (x + f)^{-1}(y) \iff \bar{g} + f([\bar{g}]^+) = y \iff \bar{g} = y - f([\bar{g}]^+) \quad (5.19)$$

There are two cases:

- If $\bar{g} \geq 0$, then $g(y) = [\bar{g}]^+ = \bar{g}$, and so we have

$$g(y) = [\bar{g}]^+ = [y - f([\bar{g}]^+)]^+ = [y - f(g(y))]^+,$$

as we wanted to show.

- Otherwise, $\bar{g} < 0$ and $g(y) = 0$. From (5.19), the condition $\bar{g} < 0$ implies $y < f([\bar{g}]^+) = f(0)$. So, it is indeed the case that

$$0 = g(y) = [y - f(0)]^+ = [y - f(g(0))]^+,$$

as we wanted to show.

Finally, we note that the function $(x + f)^{-1} : \mathbb{R} \rightarrow \mathbb{R}$ is SMPL due to [Lemma 5.6](#), implying that $g(y)$ is quasi-SMPL. \square

Chapter 6

Strongly predictive learning dynamics, and $\mathcal{O}_T(\log T/T)$ convergence in self play

As we have seen in the past chapters, it is possible to devise learning algorithms with guaranteed external regret of order $\mathcal{O}_T(\sqrt{T})$ in any imperfect-information extensive-form game, *without any assumption on the sequence of utility vectors observed by the learning algorithm*. Furthermore, when all players employ any such algorithm (with potentially different players opting for different options), the average distribution of play converges to the set of coarse correlated equilibria.

The combination of the two previous facts is remarkable: equilibrium can be reached at the approximation rate of $\mathcal{O}_T(1/\sqrt{T})$ even when playing against adversarial unknown players. However, this begs a natural question:

Is a regret rate better than $\mathcal{O}_T(\sqrt{T})$ achievable in less adversarial (more predictable) environments, including the setting of training agents using self play?

What are the optimal performance guarantees we can obtain when learning agents are competing against each other?

6.1 Related work

The fundamental question above was first formulated and addressed by Daskalakis, Deckelbaum, and Kim (2011) within the context of nonsequential, zero-sum games. Since then, there has been a considerable interest in extending their guarantee to more general settings (S. Rakhlin and Sridharan, 2013; Syrgkanis, Agarwal, Luo, and Schapire, 2015; D. J. Foster, Li, Lykouris, Sridharan,

and Tardos, 2016; Chen and Peng, 2020; Daskalakis and Golowich, 2022; Piliouras, Sim, and Skoulakis, 2021). In particular, Daskalakis, Fishelson, and Golowich (2021) recently established that when all players in a general *normal-form game* employ the *optimistic* variant of *multiplicative weights update* (MWU) (reviewed in Section 3.3.1), the regret of each player grows *nearly-optimally* as $\mathcal{O}_T(\log^4 T)$ after T repetitions of the game, leading to an *exponential improvement* over the guarantees obtained using traditional techniques within the no-regret framework. However, while normal-form games are a common way to represent strategic interactions in theory, most settings of practical significance inevitably involve more complex strategy spaces. For those settings, any faithful approximation of the game using the normal form is typically inefficient, requiring an action space that is exponential in the natural parameters of the problem, thereby limiting the practical implications of those prior results.

6.2 Contributions

As mentioned, a positive answer for the fundamental question expressed above was known for nonsequential games but not for imperfect-information extensive-form games, which are significantly more involved. In this chapter we answer the question in the positive for any imperfect-information extensive-form games, no matter the number of players. Specifically, in this chapter we will show the first uncoupled no-regret algorithms that, if used in self play, guarantee cumulating $\mathcal{O}_T(\log T)$ regret in the worst case, respectively. This method improves over the prior state of the art, which achieved $\mathcal{O}_T(\log^4 T)$ (albeit with better dependence on the size of the game) and was based on the significantly different technique of Kernelized multiplicative weights, described in Chapter 7.

One remarkable property about the algorithm we describe in this section, which we call *log-regularized lifted optimistic FTRL* (LRL-OFTRL), is its wide applicability even beyond imperfect-information extensive-form games. Indeed, like the OMD and FTRL algorithms we discussed in Chapter 5, LRL-OFTRL can be used to define no-external-regret dynamics that apply to any convex and compact strategy set \mathcal{X} , not just sequence-form strategy spaces. However, LRL-OFTRL differs from OMD and FTRL in at least two significant ways. First, LRL-OFTRL defines *strongly predictive* dynamics. This property will be fundamental in establishing logarithmic regret bounds in self play. Second, LRL-OFTRL does not enable flexibility in the proximal setup used to instantiate the algorithm, and rather requires that a logarithmic regularizer be used. This inflexibility should be naturally met with suspicion, as the logarithmic DGF is not “nice” in the sense of Definition 5.2, and for general convex and compact strategy sets an exact algorithm for projecting with respect to the logarithmic DGF might not even exist. Luckily, we show that under mild assumptions—which we show are satisfied in the sequence-form strategy polytope setting—an approximate projection can be computed with an extremely favorable $\log \log(1/\epsilon)$ dependence on the desired approximation level $\epsilon > 0$.

6.3 Setup and algorithm pseudocode

Due to the applicability of LRL-OFTRL to any convex and compact strategy set, we present LRL-OFTRL in full generality, letting $\mathcal{X} \subseteq \mathbb{R}^d$ denote such a strategy set. Without loss of generality, we will further assume that $\mathcal{X} \subseteq [0, +\infty)^d$ (otherwise, it suffices to first shift the set), and that there is no index $r \in \llbracket d \rrbracket$ such that $x[r] = 0$ for all $\mathbf{x} \in \mathcal{X}$ (if not, dropping the identically-zero dimension would not alter regret).

The pseudocode for LRL-OFTRL is simple, and is given in [Algorithm 6.1](#).

Algorithm 6.1: Log-Regularized Lifted Optimistic FTRL (LRL-OFTRL)

Data:

- $\mathcal{X} \subseteq \mathbb{R}_{\geq 0}^d$ convex and compact strategy set
- $\eta \in \mathbb{R}_{> 0}$ learning rate.

```

1  $\tilde{\mathbf{U}}^{(1)} \leftarrow \mathbf{0} \in \mathbb{R}^{d+1}$ 
2 function NextStrategy( $\mathbf{m}^{(t)} \in \mathbb{R}^\Sigma$ )
3    $\tilde{\mathbf{m}}^{(t)} \leftarrow \begin{pmatrix} -\langle \mathbf{m}^{(t)}, \mathbf{x}^{(t)} \rangle \\ \mathbf{m}^{(t)} \end{pmatrix}$  [> Lifting]
4    $\begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \leftarrow \arg \max_{(\lambda, \mathbf{y}) \in \tilde{\mathcal{X}}} \left\{ \eta \langle \tilde{\mathbf{U}}^{(t)} + \tilde{\mathbf{m}}^{(t)}, \begin{pmatrix} \lambda \\ \mathbf{y} \end{pmatrix} \rangle + \log \lambda + \sum_{r=1}^d \log \mathbf{y}[r] \right\}$  [> FTRL step]
5   return  $\mathbf{x}^{(t)} := \frac{\mathbf{y}^{(t)}}{\lambda^{(t)}} \in \mathcal{X}$  [> Normalization]


---


6 function ObserveUtility( $\mathbf{u}^{(t)} \in \mathbb{R}^\Sigma$ )
7    $\tilde{\mathbf{u}}^{(t)} \leftarrow \begin{pmatrix} -\langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle \\ \mathbf{u}^{(t)} \end{pmatrix}$  [> Lifting]
8    $\tilde{\mathbf{U}}^{(t+1)} \leftarrow \tilde{\mathbf{U}}^{(t)} + \tilde{\mathbf{u}}^{(t)}$ 

```

It is easy to see that internally LRL-OFTRL builds on the follow-the-regularized-leader (FTRL) algorithm, which was introduced in [Section 5.2.3](#), but with some important twists.

1. *Lifting.* First, the FTRL steps are performed not on \mathcal{X} , but rather on the *lifted* strategy set $\tilde{\mathcal{X}}$, defined as

$$\tilde{\mathcal{X}} := \{(\lambda, \mathbf{y}) : \lambda \in [0, 1], \mathbf{y} \in \lambda \mathcal{X}\} \subseteq \mathbb{R}^{d+1}. \quad (6.1)$$

(See [Figure 6.1](#) for geometric intuition on the relationship between \mathcal{X} and $\tilde{\mathcal{X}}$). The utility vectors $\mathbf{u}^{(t)} \in \mathbb{R}^d$ passed to LRL-OFTRL as feedback will also be lifted to vectors $\tilde{\mathbf{u}}^{(t)} \in \mathbb{R}^{d+1}$, defined on [Line 7](#); this ensures that $\tilde{\mathbf{u}}^{(t)}$ is orthogonal to the vector $(1, \mathbf{x}^{(t)})$.

2. *Logarithmic regularization.* Second, FTRL is instantiated with the logarithmic DGF for \mathbb{R}^{d+1} , that is, the function

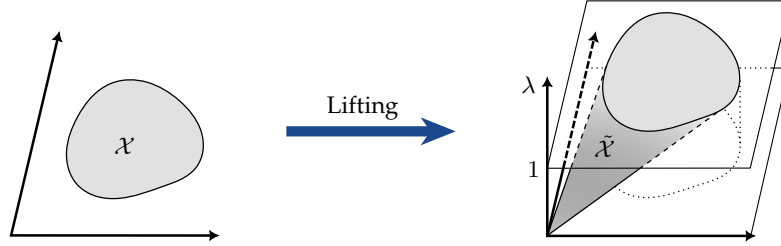


Figure 6.1: The lifting operation performed by the LRL-OFTRL algorithm on the strategy set.

$$\zeta(\lambda, \mathbf{y}) := -\log \lambda - \sum_{r=1}^d \log \mathbf{y}[r], \quad \forall (\lambda, \mathbf{y}) \in \mathbb{R}_{>0}^{d+1}.$$

We discuss how the optimization problem defined in [Line 4](#) can be solved efficiently in [Section 6.6](#). Below we point out that [Line 4](#) is indeed well-defined.

Proposition 6.1. For any $\eta \geq 0$ and at all times $t \in \mathbb{N}_{\geq 0}$, the FTRL optimization problem on [Line 4](#) of [Algorithm 6.1](#) admits a unique optimal solution $(\lambda^{(t)}, \mathbf{y}^{(t)}) \in \tilde{\mathcal{X}} \cap \mathbb{R}_{>0}^{d+1}$.

3. *Normalization.* Third, given the iterate $(\lambda^{(t)}, \mathbf{y}^{(t)})$ output by the FTRL step at time t , our no-regret algorithm over \mathcal{X} selects the next strategy $\mathbf{x}^{(t)} := \mathbf{y}^{(t)} / \lambda^{(t)}$ ([Line 5](#)); this is indeed a valid strategy in \mathcal{X} by definition of $\tilde{\mathcal{X}}$ in [\(6.1\)](#), as well as the fact that $\lambda^{(t)} > 0$ as asserted in [Proposition 6.1](#).

6.4 Regret analysis

In this section, we study the regret of LRL-OFTRL under the idealized assumption that the optimization problem on [Line 4](#) (FTRL step) is solved exactly at each time t . In [Section 6.6](#) we will relax that assumption, and study the regret of LRL-OFTRL when the solution to [Line 4](#) is approximated using variants of Newton's method.

Connection between regret on \mathcal{X} and on $\tilde{\mathcal{X}}$ Our ultimate goal will be that of analyzing the external regret

$$\text{Reg}^{(T)} := \max_{\hat{\mathbf{x}} \in \mathcal{X}} \left\{ \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \hat{\mathbf{x}} - \mathbf{x}^{(t)} \rangle \right\}$$

accumulated by LRL-OFTRL up to any time $T \in \mathbb{N}_{\geq 1}$. To do so, it will be particularly useful to relate it to the regret $\tilde{\text{Reg}}^{(T)}$ incurred with respect to the lifted utilities $\tilde{\mathbf{u}}^{(t)}$ by the lifted iterates

$(\lambda^{(t)}, \mathbf{y}^{(t)}) \in \tilde{\mathcal{X}}$ (produced by [Line 4](#)) in the lifted space $\tilde{\mathcal{X}}$, that is,

$$\tilde{\text{Reg}}^{(T)} := \max_{(\lambda^*, \mathbf{y}^*) \in \tilde{\mathcal{X}}} \sum_{t=1}^T \left\langle \tilde{\mathbf{u}}^{(t)}, \begin{pmatrix} \lambda^* \\ \mathbf{y}^* \end{pmatrix} - \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\rangle.$$

As the following theorem clarifies, there is a strong connection between $\tilde{\text{Reg}}^{(T)}$ and $\text{Reg}^{(T)}$.

Theorem 6.1. For any time $T \in \mathbb{N}_{\geq 0}$ it holds that $\tilde{\text{Reg}}^{(T)} = \max\{0, \text{Reg}^{(T)}\}$. In particular, it follows that $\tilde{\text{Reg}}^{(T)} \geq 0$ and $\text{Reg}^{(T)} \leq \tilde{\text{Reg}}^{(T)}$ for any $T \in \mathbb{N}_{\geq 0}$.

Proof. First, by definition of $\tilde{\mathbf{u}}^{(t)}$ in [Line 7](#), it follows that for any t ,

$$\left\langle \tilde{\mathbf{u}}^{(t)}, \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\rangle = \left\langle \tilde{\mathbf{u}}^{(t)}, \begin{pmatrix} 1 \\ \mathbf{x}^{(t)} \end{pmatrix} \right\rangle = 0.$$

As a result, we have that $\max\{0, \text{Reg}^{(T)}\}$ is equal to

$$\begin{aligned} \max \left\{ 0, \max_{\mathbf{x}^* \in \mathcal{X}} \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \mathbf{x}^* - \mathbf{x}^{(t)} \rangle \right\} &= \max \left\{ 0, \max_{\mathbf{x}^* \in \mathcal{X}} \sum_{t=1}^T \left\langle \tilde{\mathbf{u}}^{(t)}, \begin{pmatrix} 1 \\ \mathbf{x}^* \end{pmatrix} - \begin{pmatrix} 1 \\ \mathbf{x}^{(t)} \end{pmatrix} \right\rangle \right\} \\ &= \max \left\{ 0, \max_{\mathbf{x}^* \in \mathcal{X}} \sum_{t=1}^T \left\langle \tilde{\mathbf{u}}^{(t)}, \begin{pmatrix} 1 \\ \mathbf{x}^* \end{pmatrix} \right\rangle \right\} \\ &= \max_{(\lambda^*, \mathbf{y}^*) \in \tilde{\mathcal{X}}} \sum_{t=1}^T \left\langle \tilde{\mathbf{u}}^{(t)}, \begin{pmatrix} \lambda^* \\ \mathbf{y}^* \end{pmatrix} \right\rangle \\ &= \max_{(\lambda^*, \mathbf{y}^*) \in \tilde{\mathcal{X}}} \sum_{t=1}^T \left\langle \tilde{\mathbf{u}}^{(t)}, \begin{pmatrix} \lambda^* \\ \mathbf{y}^* \end{pmatrix} - \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\rangle = \tilde{\text{Reg}}^{(T)}, \end{aligned}$$

as we wanted to show. □

The nonnegativity of $\tilde{\text{Reg}}^{(T)}$ will be a crucial property in establishing [Theorem 6.2](#). Further, [Theorem 6.1](#) implies that a guarantee over the lifted space can be automatically translated to a regret bound over the original space \mathcal{X} .

Local norms induced by the logarithmic DGF Given any vector $(\lambda, \mathbf{y}) \in \tilde{\mathcal{X}} \cap \mathbb{R}_{>0}^{d+1}$, the Hessian matrix of ζ at (λ, \mathbf{y}) induces a *local* primal-dual norm pair centered at that point, defined as

$$\left\| \begin{pmatrix} a \\ \mathbf{z} \end{pmatrix} \right\|_{(\lambda, \mathbf{y})} := \sqrt{\left(\frac{a}{\lambda}\right)^2 + \sum_{r=1}^d \left(\frac{\mathbf{z}[r]}{\mathbf{y}[r]}\right)^2}, \quad \left\| \begin{pmatrix} a \\ \mathbf{z} \end{pmatrix} \right\|_{*, (\lambda, \mathbf{y})} := \sqrt{(a\lambda)^2 + \sum_{r=1}^d (\mathbf{z}[r]\mathbf{y}[r])^2}$$

for any $(a, \mathbf{z}) \in \mathbb{R}^{d+1}$. (It is a well-known fact that $\|\cdot\|_{*,(\lambda,\mathbf{y})}$ is the dual norm of $\|\cdot\|_{(\lambda,\mathbf{y})}$, and *vice versa*.) To simplify notation, we will use the streamlined symbols

$$\|\cdot\|_t := \|\cdot\|_{(\lambda^{(t)}, \mathbf{y}^{(t)})} \quad \text{and} \quad \|\cdot\|_{*,t} := \|\cdot\|_{*,(\lambda^{(t)}, \mathbf{y}^{(t)})} \quad (6.2)$$

to denote the local norms centered at the iterate $(\lambda^{(t)}, \mathbf{y}^{(t)})$ produced by FTRL at time t (Line 4). In the next proposition we establish a refined regret bound in terms of this primal-dual norm pair.

Proposition 6.2 (Regret of FTRL in local norms). Let $\tilde{\text{Reg}}^{(T)}$ be the regret cumulated up to time T by the internal FTRL algorithm. If $\|\mathbf{u}^{(t)}\|_\infty \|\mathbf{x}\|_1 \leq 1$ at all times $t \in \llbracket T \rrbracket$, then for any time horizon $T \in \mathbb{N}_{\geq 0}$ and learning rate $\eta \leq \frac{1}{50}$,

$$\tilde{\text{Reg}}^{(T)} \leq \frac{(d+1) \log T}{\eta} + 5\eta \sum_{t=1}^T \left\| \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \right\|_{*,t}^2 - \frac{1}{27\eta} \sum_{t=1}^{T-1} \left\| \begin{pmatrix} \lambda^{(t+1)} \\ \mathbf{y}^{(t+1)} \end{pmatrix} - \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\|_t^2.$$

Proposition 6.2 differs from prior analogous results in that the regularizer is not a *barrier* over the feasible set.

Multiplicative stability of the lifted iterates The iterates produced by FTRL satisfy a refined notion of stability, which we refer to as *multiplicative stability*.

Proposition 6.3. For any time $t \in \mathbb{N}_{\geq 0}$ and learning rate $\eta \leq \frac{1}{50}$, if $\|\mathbf{u}^{(t)}\|_\infty \|\mathbf{x}\|_1, \|\mathbf{m}^{(t)}\|_\infty \|\mathbf{x}\|_1 \leq 1$ for all $\mathbf{x} \in \mathcal{X}$,

$$\left\| \begin{pmatrix} \lambda^{(t+1)} \\ \mathbf{y}^{(t+1)} \end{pmatrix} - \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\|_t \leq 22\eta.$$

Intuitively, this property ensures that coordinates of successive iterates will have a small multiplicative deviation. We leverage this notion of stability to establish the following crucial lemma.

Lemma 6.1. For any time $t \in \mathbb{N}_{\geq 0}$ and learning rate $\eta \leq \frac{1}{50}$, if $\|\mathbf{u}^{(t)}\|_\infty \|\mathbf{x}\|_1 \leq 1$,

$$\|\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\|_1 \leq 4\|\mathcal{X}\|_1 \left\| \begin{pmatrix} \lambda^{(t+1)} \\ \mathbf{y}^{(t+1)} \end{pmatrix} - \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\|_t.$$

Establishing strong predictivity Combining this lemma with [Proposition 6.2](#) allows us to obtain an RVU bound for the original space \mathcal{X} , with no dependencies on local norms.

Corollary 6.1 (RVU bound in the original (unlifted) space). Fix any time $T \in \mathbb{N}_{\geq 0}$, and suppose that $\|\mathbf{u}^{(t)}\|_\infty \leq B$ for any $t \in \llbracket T \rrbracket$. If $\eta \leq \frac{1}{256B\|\mathcal{X}\|_1}$,

$$\begin{aligned} \tilde{\text{Reg}}^{(T)} \leq & 6B\|\mathcal{X}\|_1 + \frac{(d+1)\log T}{\eta} + 16\eta\|\mathcal{X}\|_1^2 \sum_{t=1}^{T-1} \|\mathbf{u}^{(t)} - \mathbf{m}^{(t)}\|_\infty^2 \\ & - \frac{1}{512\eta\|\mathcal{X}\|_1^2} \sum_{t=1}^{T-1} \|\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\|_1^2. \end{aligned}$$

6.5 Connecting strong predictivity and logarithmic regret

The variation in one player's utilities is related to the variation in the joint strategies based on the smoothness condition of the utility function, connecting the last two terms of the RVU bound. In particular, for the rest of the chapter, we will work under the following standard assumptions, which are always verified in (finite) normal-form and extensive-form games.

Assumption 6.1. The utility function $u_i(\mathbf{x}_1, \dots, \mathbf{x}_n)$ of any player $i \in \llbracket n \rrbracket$ satisfies the following properties:

1. (Concavity) $u_i(\mathbf{x}_i, \mathbf{x}_{-i})$ is *concave* in \mathbf{x}_i for $\mathbf{x}_{-i} = (\mathbf{x}_1, \dots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \dots, \mathbf{x}_n) \in \times_{j \neq i} \mathcal{X}_j$;
2. (Bounded gradients) for any $(\mathbf{x}_1, \dots, \mathbf{x}_n) \in \times_{j=1}^n \mathcal{X}_j$, $\|\nabla_{\mathbf{x}_i} u_i(\mathbf{x}_1, \dots, \mathbf{x}_n)\|_\infty \leq B$, for some parameter $B > 0$; and
3. (L -smoothness) there exists $L > 0$ so that for any two joint strategy profiles $\mathbf{x}, \mathbf{x}' \in \times_{j=1}^n \mathcal{X}_j$,

$$\|\nabla_{\mathbf{x}_i} u_i(\mathbf{x}) - \nabla_{\mathbf{x}_i} u_i(\mathbf{x}')\|_\infty \leq L \sum_{j \in \llbracket n \rrbracket} \|\mathbf{x}_j - \mathbf{x}'_j\|_1.$$

Given the assumption above, and by leveraging the nonnegativity of the regrets in the lifted space, we establish that the second-order path lengths of the dynamics up to time T are bounded by $\mathcal{O}_T(\log T)$:

Theorem 6.2. If all players follow LRL-OFTRL with learning rate

$$\eta \leq \min \left\{ \frac{1}{256B\|\mathcal{X}\|_1}, \frac{1}{128nL\|\mathcal{X}\|_1^2} \right\},$$

where $\|\mathcal{X}\|_1 := \max_{i \in \llbracket n \rrbracket} \|\mathcal{X}_i\|_1$, then

$$\sum_{i=1}^n \sum_{t=1}^{T-1} \|\mathbf{x}_i^{(t+1)} - \mathbf{x}_i^{(t)}\|_1^2 \leq 6144n\eta B\|\mathcal{X}\|_1^3 + 1024n(d+1)\|\mathcal{X}\|_1^2 \log T. \quad (6.3)$$

We next leverage [Theorem 6.2](#) to establish [Theorem 6.3](#).

Theorem 6.3. If all players use LRL-OFTRL within the COLS (that is, with $\mathbf{m}^{(t)} = \mathbf{u}^{(t-1)}$; cf. [Section 3.2.1](#)) with learning rate

$$\eta = \min \left\{ \frac{1}{256B\|\mathcal{X}\|_1}, \frac{1}{128nL\|\mathcal{X}\|_1^2} \right\},$$

then for any $T \in \mathbb{N}_{\geq 0}$ the regret $\text{Reg}_i^{(T)}$ of each player $i \in \llbracket n \rrbracket$ can be bounded as

$$\text{Reg}_i^{(T)} \leq 12B\|\mathcal{X}\|_1 + 256(d+1) \max\{nL\|\mathcal{X}\|_1^2, 2B\|\mathcal{X}\|_1\} \log T. \quad (6.4)$$

Furthermore, the algorithm can be adaptive so that if player i is instead facing adversarial utilities, then $\text{Reg}_i^{(T)} = \mathcal{O}_T(\sqrt{T})$.

For clarity, below we cast [\(6.4\)](#) of [Theorem 6.3](#) in normal-form games with utilities normalized in the range $[-1, 1]$, in which case we can take $B = 1$, $L = 1$ and $\|\mathcal{X}\|_1 = 1$.

Corollary 6.2 (Normal-form games). Suppose that all players in a normal-form game with $n \geq 2$ follow LRL-OFTRL within the COLS with learning rate $\eta = \frac{1}{128n}$. Then, for any $T \in \mathbb{N}_{\geq 0}$ and player $i \in \llbracket n \rrbracket$,

$$\text{Reg}_i^{(T)} \leq 12 + 256n(d+1) \log T.$$

6.6 Implementation and iteration complexity

In this section, we discuss the implementation and iteration complexity of LRL-OFTRL. The main difficulty in the implementation is the computation of the solution to the strictly concave *nonsmooth*

constrained optimization problem in [Line 4](#). We start by studying how the guarantees laid out in [Theorem 6.3](#) are affected when the exact solution to the FTRL problem ([Line 5](#)) in [Algorithm 6.1](#) is replaced with an approximation. Specifically, suppose that at all times t the solution to the FTRL step ([Line 4](#)) in [Algorithm 6.1](#) is only *approximately* solved within tolerance $\epsilon^{(t)}$, in the sense that

$$\left\| \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} - \begin{pmatrix} \lambda_\star^{(t)} \\ \mathbf{y}_\star^{(t)} \end{pmatrix} \right\|_{(\lambda_\star^{(t)}, \mathbf{y}_\star^{(t)})} \leq \epsilon^{(t)}, \quad (6.5)$$

where $(\lambda^{(t)}, \mathbf{y}^{(t)}) \in \mathbb{R}_{>0}^{d+1}$ and

$$\begin{pmatrix} \lambda_\star^{(t)} \\ \mathbf{y}_\star^{(t)} \end{pmatrix} := \arg \max_{(\lambda, \mathbf{y}) \in \tilde{\mathcal{X}}} \left\{ \eta \left\langle \tilde{\mathbf{U}}^{(t)} + \tilde{\mathbf{m}}^{(t)}, \begin{pmatrix} \lambda \\ \mathbf{y} \end{pmatrix} \right\rangle + \log \lambda + \sum_{r=1}^d \log \mathbf{y}[r] \right\}.$$

Then, it can be proven directly from the definition of regret that the guarantees given in [Corollary 6.1](#) deteriorate by an additive factor proportional to the sum of the tolerances $\sum_{t=1}^T \epsilon^{(t)}$. As an immediate corollary, when $\epsilon^{(t)} := \epsilon := 1/T$, the conclusion of [Theorem 6.3](#) applies even when the solution to the optimization problem on [Line 4](#) is only approximated up to ϵ tolerance. Therefore, to complete our construction, it suffices to show that it is indeed possible to efficiently compute approximate solutions to the FTRL step. In the remainder of this section, we show that this is indeed the case assuming access to two different types of oracles. It should be stressed that optimizing over a general convex set introduces several challenges not present under simplex domains, inevitably leading to an increased per-iteration complexity compared to algorithms designed specifically for normal-form games—such as OMWU.

6.6.1 Local proximal oracle

First, we will assume access to a *proximal oracle* in local norm for the set $\tilde{\mathcal{X}}$, that is, access to a function that is able to compute the solution to the (positive-definite) quadratic optimization problem

$$\Pi_{\tilde{\mathbf{w}}}(\tilde{\mathbf{g}}) := \arg \min_{\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}} \left\{ \langle \tilde{\mathbf{g}}, \tilde{\mathbf{x}} \rangle + \frac{1}{2} \|\tilde{\mathbf{x}} - \tilde{\mathbf{w}}\|_{\tilde{\mathbf{w}}}^2 \right\} = \arg \min_{\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}} \left\{ \langle \tilde{\mathbf{g}}, \tilde{\mathbf{x}} \rangle + \frac{1}{2} \sum_{r=1}^{d+1} \left(\frac{\tilde{\mathbf{x}}[r]}{\tilde{\mathbf{w}}[r]} - 1 \right)^2 \right\} \quad (6.6)$$

for arbitrary centers $\tilde{\mathbf{w}} \in \mathbb{R}_{>0}^{d+1}$ and gradients $\tilde{\mathbf{g}} \in \mathbb{R}^{d+1}$. For certain sets $\mathcal{X} \subseteq \mathbb{R}^d$, exact proximal oracles with polynomial complexity in the dimension d can be given. In particular, this is the case for sequence-form strategy polytopes, where $\Pi_{\tilde{\mathbf{w}}}(\tilde{\mathbf{g}})$ can be computed using the algorithm described in [Section 5.3.1](#). Specifically, the following holds.

Proposition 6.4. Let $\mathcal{X} = \mathcal{Q}$ be the polytope of sequence-form strategies for a player in an imperfect-information extensive-form game. Then, the local proximal oracle $\Pi_{\tilde{\mathbf{w}}}(\tilde{\mathbf{g}})$ defined in (6.6) can be implemented exactly in time polynomial in the number of sequences $|\Sigma|$ for any $\tilde{\mathbf{w}} \in \mathbb{R}_{>0}^{|\Sigma|+1}$ and $\tilde{\mathbf{g}} \in \mathbb{R}^{|\Sigma|+1}$.

When an efficient, exact local proximal oracle is available, the projection $(\lambda_\star^{(t)}, \mathbf{y}_\star^{(t)}) \in \tilde{\mathcal{X}}$ can be computed up to any precision $\epsilon > 0$ using $\log \log(1/\epsilon)$ calls to the oracle, by using the *proximal Newton algorithm* (Tran-Dinh, Kyrillidis, and Cevher, 2015). More precisely, Tran-Dinh, Kyrillidis, and Cevher (2015) studied the following composite minimization problem:

$$\min_{\tilde{\mathbf{x}} \in \mathbb{R}^{d+1}} \{F(\tilde{\mathbf{x}}) := f(\tilde{\mathbf{x}}) + g(\tilde{\mathbf{x}})\}, \quad (6.7)$$

where f is a (standard) self-concordant and convex function, and $g : \mathbb{R}^{d+1} \rightarrow \mathbb{R} \cup \{+\infty\}$ is a proper, closed and convex function. In our setting, we will let g be defined as

$$g(\tilde{\mathbf{x}}) := \begin{cases} 0 & \text{if } \tilde{\mathbf{x}} \in \tilde{\mathcal{X}}, \\ +\infty & \text{otherwise.} \end{cases}$$

Further, for a given time $t \in \mathbb{N}_{\geq 0}$, we let

$$f : \tilde{\mathbf{x}} \mapsto -\eta \langle \tilde{\mathbf{U}}^{(t)} + \tilde{\mathbf{m}}^{(t)}, \tilde{\mathbf{x}} \rangle - \sum_{r=1}^{d+1} \log \tilde{\mathbf{x}}[r]$$

and

$$\tilde{\mathbf{s}}_k := \arg \min_{\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}} \left\{ f(\tilde{\mathbf{x}}_k) + (\nabla f(\tilde{\mathbf{x}}_k))^\top (\tilde{\mathbf{x}} - \tilde{\mathbf{x}}_k) + \frac{1}{2} (\tilde{\mathbf{x}} - \tilde{\mathbf{x}}_k)^\top \nabla^2 f(\tilde{\mathbf{x}}_k) (\tilde{\mathbf{x}} - \tilde{\mathbf{x}}_k) \right\}, \quad (6.8)$$

for some $\tilde{\mathbf{x}}_k \in \mathbb{R}_{>0}^{d+1}$. The optimization problem (6.8) can be solved directly through the local proximal oracle. The proximal Newton method given in Algorithm 6.2.

The algorithm proceeds in two phases. In the first phase we perform *damped steps* of proximal Newton until we reach the region of quadratic convergence. Afterwards, we perform *full steps* of proximal Newton until the desired precision $\epsilon > 0$ has been reached. The following is known.

Theorem 6.4 (Tran-Dinh, Kyrillidis, and Cevher, 2015, Theorem 9). Algorithm 6.2 returns $\tilde{\mathbf{x}}_K \in \mathbb{R}_{>0}^{d+1}$ such that $\|\tilde{\mathbf{x}}_K - \tilde{\mathbf{x}}^*\|_{\tilde{\mathbf{x}}^*} \leq 2\epsilon$ after at most

Algorithm 6.2: Proximal Newton method (Tran-Dinh, Kyrillidis, and Cevher, 2015)

Data: Initial point $\tilde{\mathbf{x}}_0$; Precision $\epsilon > 0$; Constant $\sigma := 0.2$

```

1 for  $k = 1, \dots, K$  do
2   Obtain the proximal Newton direction  $\tilde{\mathbf{d}}_k \leftarrow \tilde{\mathbf{s}}_k - \tilde{\mathbf{x}}_k$ , where  $\tilde{\mathbf{s}}_k$  is defined in (6.8)
3   Set  $\lambda_k \leftarrow \|\tilde{\mathbf{d}}_k\|_{\tilde{\mathbf{x}}_k}$ 
4   if  $\lambda_k > 0.2$  then
5      $\tilde{\mathbf{x}}_{k+1} \leftarrow \tilde{\mathbf{x}}_k + \alpha_k \tilde{\mathbf{d}}_k$ , where  $\alpha_k := (1 + \lambda_k)^{-1}$  [▷ Damped Step]
6   else if  $\lambda_k > \epsilon$  then
7      $\tilde{\mathbf{x}}_{k+1} \leftarrow \tilde{\mathbf{x}}_k + \tilde{\mathbf{d}}_k$  [▷ Full Step]
8   else
9     return  $\tilde{\mathbf{x}}_k$ 

```

$$K = \left\lceil \frac{f(\tilde{\mathbf{x}}_0) - f(\tilde{\mathbf{x}}^*)}{0.017} \right\rceil + \left\lceil 1.5 \ln \ln \left(\frac{0.28}{\epsilon} \right) \right\rceil + 2$$

iterations, for any $\epsilon > 0$, where $\tilde{\mathbf{x}}^* = \arg \min_{\tilde{\mathbf{x}}} F(\tilde{\mathbf{x}})$, for the composite function F defined in (6.7).

Corollary 6.3. Given any $\epsilon > 0$, it is possible to compute $(\lambda^{(t)}, \mathbf{y}^{(t)}) \in \tilde{\mathcal{X}} \cap \mathbb{R}_{>0}^{d+1}$ such that (6.5) holds for $\epsilon^{(t)} = \epsilon$ using $\mathcal{O}(\log \log(1/\epsilon))$ operations and $\mathcal{O}(\log \log(1/\epsilon))$ calls to the proximal oracle defined in Equation (6.6).

Proof. The corollary follows from Theorem 6.4 by initializing Algorithm 6.2 at every iteration $t \geq 2$ with $\tilde{\mathbf{x}}_0 := \tilde{\mathbf{x}}^{(t-1)} = (\lambda^{(t-1)}, \mathbf{y}^{(t-1)})$. Then, as long as $\epsilon^{(t-1)}$ is sufficiently small, the number of iterations predicted by Theorem 6.4 will be bounded by $\mathcal{O}(\log \log(1/\epsilon))$, as claimed. \square

6.6.2 Linear maximization oracle

Moreover, we consider having access to a weaker *linear maximization oracle* (LMO) for the set \mathcal{X} :

$$\mathcal{L}_{\mathcal{X}}(\mathbf{u}) := \arg \max_{\mathbf{x} \in \mathcal{X}} \langle \mathbf{x}, \mathbf{u} \rangle. \quad (6.9)$$

Such an oracle is more realistic in many settings (Jaggi, 2013), and it is particularly natural in the context of games, where it can be thought of as a *best response* oracle. We point out that an LMO for \mathcal{X} automatically implies an LMO for $\tilde{\mathcal{X}}$. The following guarantee follows readily by applying the *Frank-Wolfe (projected) Newton method* (Liu, Cevher, and Tran-Dinh, 2020, Algorithms 1 and 2).

Theorem 6.5 (Frank-Wolfe Newton). Given any $\epsilon > 0$, it is possible to compute $(\lambda^{(t)}, \mathbf{y}^{(t)}) \in \tilde{\mathcal{X}} \cap \mathbb{R}_{>0}^{d+1}$ such that (6.5) holds for $\epsilon^{(t)} = \epsilon$ using $\mathcal{O}(\text{poly}(1/\epsilon))$ operations and $\mathcal{O}(\text{poly}(1/\epsilon))$ calls to the LMO oracle defined in Equation (6.9).

6.7 Experimental evaluation

Finally, we support the theory developed in this chapter by conducting experiments on four established benchmark games: **K23** and **K33**, respectively 2-player and 3-player *Kuhn poker* (Kuhn, 1950); **GL23** 2-player limited-information *Goofspiel* (Ross, 1971); and **S1a22**, the baseline version of (2-player) *Sheriff* (Farina, Ling, Fang, and Sandholm, 2019a). As usual, a full description of the game instances used is available in [Appendix A](#). Of these games, only 2-player Kuhn poker (**K23**) is a zero-sum game.

The experiments are meant to verify the logarithmic per-player regret bound established in [Theorem 6.3](#). Our findings are summarized in [Figure 6.2](#), where we remark that the x-axis is on a logarithmic scale. All plots refer to the choice of learning rate $\eta := 0.5$, which was selected after a very mild tuning process.

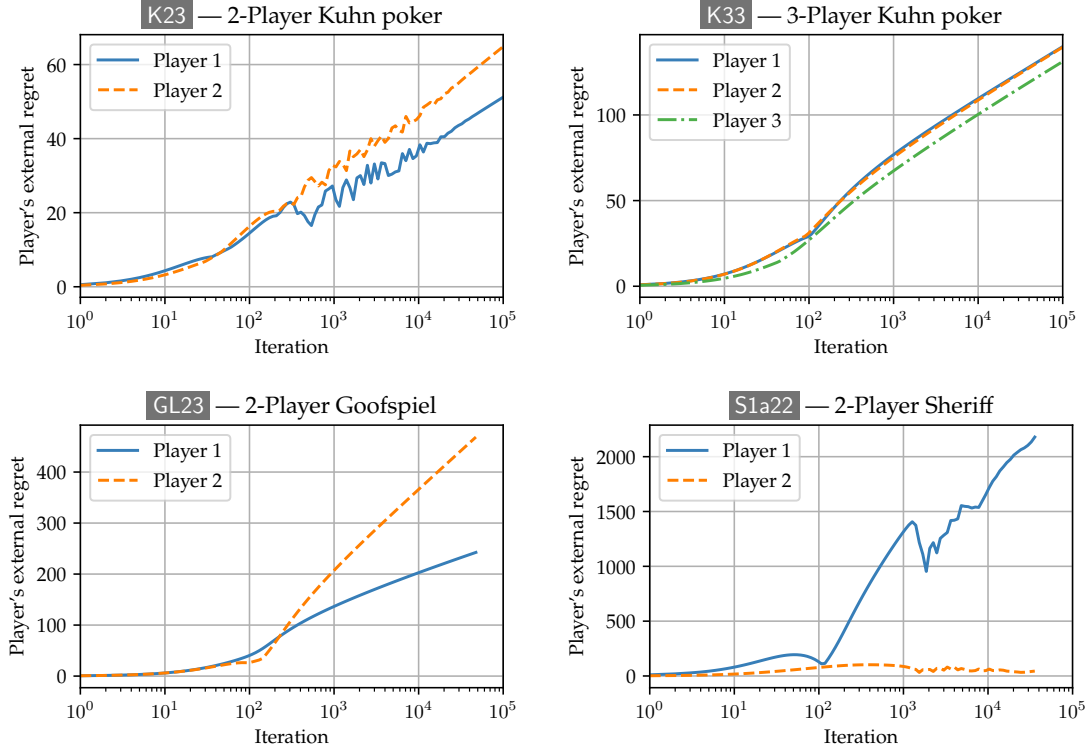


Figure 6.2: The regret of the players when they follow our learning dynamics, LRL-OFTRL. The x -axis indexes the iteration, while the y -axis the regret. The scale on the x -axis is *logarithmic*. We observe that the regret of each player grows as $\mathcal{O}_T(\log T)$, verifying [Theorem 6.3](#).

We confirm that each player’s regret indeed appears to have a linear trend in the semi-logarithmic plot, which corresponds to a logarithmic rate of growth as a function of the number of iterations T .



6.A Appendix: Proof details

6.A.1 Proof of [Proposition 6.1](#)

Proposition 6.1 (Restated). For any $\eta \geq 0$ and at all times $t \in \mathbb{N}_{\geq 0}$, the FTRL optimization problem on [Line 4](#) of [Algorithm 6.1](#) admits a unique optimal solution $(\lambda^{(t)}, \mathbf{y}^{(t)}) \in \tilde{\mathcal{X}} \cap \mathbb{R}_{>0}^{d+1}$.

Proof. Uniqueness follows immediately from strict convexity. In the rest of the proof we focus on the existence part.

We start by showing that there exists a point $\tilde{x} \in \tilde{\mathcal{X}}$ whose coordinates are all strictly positive. By hypothesis (see [Section 6.3](#)), for every coordinate $r \in \llbracket d \rrbracket$, there exists a point \mathbf{x}_r such that $\mathbf{x}_r[r] > 0$. Hence, by convexity of $\mathcal{X} \subseteq [0, +\infty)^d$ and by definition of $\tilde{\mathcal{X}}$, the point

$$(1, \mathbf{x}^\circ) := \left(1, \frac{1}{d} \sum_{r=1}^d \mathbf{x}_r \right).$$

is such that $(1, \mathbf{x}^\circ) \in \tilde{\mathcal{X}} \cap \mathbb{R}_{>0}^d$.

Let now M be the ℓ_∞ norm of the linear part in the FTRL step ([Line 4](#) of [Algorithm 6.1](#)). Then, a *lower bound* on the optimal value v^* of objective is obtained by plugging in the point $(1, \mathbf{x}^\circ)$ at least

$$v^* \geq -M(1 + \|\mathcal{X}\|_1) + \sum_{r=1}^d \log \mathbf{x}^\circ[r]. \quad (6.10)$$

Let now

$$m := \exp \left\{ -(2M + d)(1 + \|\mathcal{X}\|_1) + \sum_{r=1}^d \log \mathbf{x}^\circ[r] \right\} > 0. \quad (6.11)$$

We will show that any point $(\lambda, \mathbf{y}) \notin [m, +\infty) \cap \tilde{\mathcal{X}}$ cannot be optimal for the FTRL objective. Indeed, take a point $(\lambda, \mathbf{y}) \notin [m, +\infty) \cap \tilde{\mathcal{X}}$. Then, at least one coordinate of (λ, \mathbf{y}) is strictly less than m . If $\lambda < m$, then the objective value at (λ, \mathbf{y}) is at most

$$\begin{aligned} M\lambda + M\|\mathcal{X}\|_1 + \log \lambda + \sum_{r=1}^d \log \mathbf{y}[r] &\leq M(1 + \|\mathcal{X}\|_1) + \log m + \sum_{r=1}^d \log \|\mathcal{X}\|_1 \\ &\leq M(1 + \|\mathcal{X}\|_1) + \log m + d\|\mathcal{X}\|_1 \\ &< (M + d)(1 + \|\mathcal{X}\|_1) + \log m \\ &= -M(1 + \|\mathcal{X}\|_1) + \sum_{r=1}^d \log \mathbf{x}^\circ[r] && \text{(from (6.11))} \\ &\leq v^*, && \text{(from (6.10))} \end{aligned}$$

where the first inequality follows from upper bounding any coordinate of \mathbf{y} with $\|\mathcal{X}\|_1$, and the second inequality follows from using the inequality $\log z \leq z$, valid for all $z \in (0, +\infty)$. Similarly, if $\mathbf{y}[s] < m$ for some $s \in \llbracket d \rrbracket$, then we can upper bound the objective value at (λ, \mathbf{y}) as

$$\begin{aligned}
 M + M\|\mathcal{X}\|_1 + \log 1 + \sum_{r=1}^d \log \mathbf{y}[r] &\leq M(1 + \|\mathcal{X}\|_1) + \log m + \sum_{r=1}^d \log \|\mathcal{X}\|_1 \\
 &\leq M(1 + \|\mathcal{X}\|_1) + (d-1)(\|\mathcal{X}\|_1 - 1) + \log m \\
 &< (M+d)(1 + \|\mathcal{X}\|_1) + \log m \leq v^*.
 \end{aligned}$$

So, in either case, we see that no optimal point can have any coordinate strictly less than m . Consequently, the maximizer of the FTRL step lies in the set $\mathcal{S} := [m, +\infty)^{d+1} \cap \tilde{\mathcal{X}}$. Since both $[m, +\infty)^{d+1}$ and $\tilde{\mathcal{X}}$ are closed, and since $\tilde{\mathcal{X}}$ is bounded by hypothesis, the set \mathcal{S} is compact. Furthermore, note that \mathcal{S} is nonempty, as $(1, \mathbf{x}^\circ) \in \mathcal{S}$, as for any $s \in [d]$

$$\begin{aligned}
 \log m &= -(2M+d)(1 + \|\mathcal{X}\|_1) + \sum_{r=1}^d \log \mathbf{x}^\circ[r] \\
 &\leq -(2M+d)(1 + \|\mathcal{X}\|_1) + \log \mathbf{x}^\circ[s] + (d-1) \log \|\mathcal{X}\|_1 \\
 &\leq -(2M+d)(1 + \|\mathcal{X}\|_1) + \log \mathbf{x}^\circ[s] + (d-1)(\|\mathcal{X}\|_1 - 1) \\
 &\leq \log \mathbf{x}^\circ[s],
 \end{aligned}$$

implying that $(1, \mathbf{x}^\circ) \in [m, +\infty)^{d+1}$. Since \mathcal{S} is compact and nonempty and the objective function is continuous, the optimization problem attains an optimal solution on \mathcal{S} by virtue of Weierstrass' theorem. \square

6.A.2 Proof of Proposition 6.2

For notational convenience, we define the log-regularizer $\psi : \tilde{\mathcal{X}} \rightarrow \mathbb{R}_{\geq 0}$ as

$$\psi(\mathbf{x}) := -\frac{1}{\eta} \sum_{r=1}^{d+1} \log \tilde{\mathbf{x}}[r],$$

and its induced Bregman divergence

$$D_\psi(\mathbf{x} \parallel \tilde{\mathbf{a}}) := \frac{1}{\eta} \sum_{r=1}^{d+1} h\left(\frac{\tilde{\mathbf{x}}[r]}{\tilde{\mathbf{a}}[r]}\right), \quad \text{where } h(a) = a - 1 - \ln(a).$$

Moreover, we define

$$\tilde{\mathbf{x}}^{(t)} = \arg \max_{\mathbf{x} \in \tilde{\mathcal{X}}} -F_t(\mathbf{x}) = \arg \min_{\mathbf{x} \in \tilde{\mathcal{X}}} F_t(\mathbf{x}), \quad \text{where } F_t(\mathbf{x}) = -\langle \tilde{\mathbf{U}}^{(t)} + \tilde{\mathbf{m}}^{(t)}, \mathbf{x} \rangle + \psi(\mathbf{x}). \quad (6.12)$$

We note that F_t is a convex function for each t and $\tilde{\mathbf{x}}^{(t)}$ is exactly equal to $\begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix}$ computed by [Algorithm 6.1](#). Further, we define an auxiliary sequence $\{\tilde{\mathbf{a}}^{(t)}\}_{t=1,2,\dots}$ defined as follows.

$$\tilde{\mathbf{a}}^{(t)} = \arg \max_{\mathbf{x} \in \tilde{\mathcal{X}}} -G_t(\mathbf{x}) = \arg \min_{\mathbf{x} \in \tilde{\mathcal{X}}} G_t(\mathbf{x}), \quad \text{where } G_t(\mathbf{x}) = -\langle \tilde{\mathbf{U}}^{(t)}, \mathbf{x} \rangle + \psi(\mathbf{x}). \quad (6.13)$$

Similarly, G_t is a convex function for each t . We also recall the primal and dual norm notation:

$$\|\tilde{\mathbf{a}}\|_t = \sum_{r=1}^{d+1} \left(\frac{\tilde{\mathbf{a}}[r]}{\tilde{\mathbf{x}}^{(t)}[r]} \right)^2, \quad \|\tilde{\mathbf{a}}\|_{*,t} = \sum_{r=1}^{d+1} \left(\tilde{\mathbf{x}}^{(t)}[r] \tilde{\mathbf{a}}[r] \right)^2.$$

Finally, for a $(d+1) \times (d+1)$ positive definite matrix \mathbf{M} , we use $\|\tilde{\mathbf{a}}\|_{\mathbf{M}}$ to denote the induced quadratic norm $\sqrt{\tilde{\mathbf{a}}^\top \mathbf{M} \tilde{\mathbf{a}}}$.

In the proof of [Proposition 6.2](#), we will make use of the following lemmas.

Lemma 6.2. The update rule (6.12) ensures the following for any $\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}$:

$$\begin{aligned} \sum_{t=1}^T \langle \tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} \rangle &\leq \psi(\tilde{\mathbf{x}}) - \psi(\tilde{\mathbf{x}}^{(1)}) + \sum_{t=1}^T \langle \tilde{\mathbf{a}}^{(t+1)} - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \rangle \\ &\quad - \sum_{t=1}^T \left(D_\psi(\tilde{\mathbf{x}}^{(t)} \parallel \tilde{\mathbf{a}}^{(t)}) + D_\psi(\tilde{\mathbf{a}}^{(t+1)} \parallel \tilde{\mathbf{x}}^{(t)}) \right). \end{aligned}$$

Proof. First note that for any convex function $F : \tilde{\mathcal{X}} \rightarrow \mathbb{R}$ and a minimizer \mathbf{x}^* , we have for any $\mathbf{x} \in \tilde{\mathcal{X}}$:

$$F(\mathbf{x}^*) = F(\mathbf{x}) - \langle \nabla F(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^* \rangle - D_F(\mathbf{x} \parallel \mathbf{x}^*) \leq F(\mathbf{x}) - D_F(\mathbf{x} \parallel \mathbf{x}^*),$$

where D_F is the Bregman Divergence induced by F and the inequality is by the first-order optimality. Using this fact and the optimality of $\tilde{\mathbf{a}}^{(t)}$, we have

$$\begin{aligned} G_t(\tilde{\mathbf{a}}^{(t)}) &\leq G_t(\tilde{\mathbf{x}}^{(t)}) - D_\psi(\tilde{\mathbf{x}}^{(t)} \parallel \tilde{\mathbf{a}}^{(t)}) \\ &= F_t(\tilde{\mathbf{x}}^{(t)}) + \langle \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{m}}^{(t)} \rangle - D_\psi(\tilde{\mathbf{x}}^{(t)} \parallel \tilde{\mathbf{a}}^{(t)}) \end{aligned}$$

Similarly, using the optimality of $\tilde{\mathbf{x}}^{(t)}$, we have

$$F_t(\tilde{\mathbf{x}}^{(t)}) \leq F_t(\tilde{\mathbf{a}}^{(t+1)}) - D_\psi(\tilde{\mathbf{a}}^{(t+1)} \parallel \tilde{\mathbf{x}}^{(t)})$$

$$= G_{t+1}(\tilde{\mathbf{a}}^{(t+1)}) + \left\langle \tilde{\mathbf{a}}^{(t+1)}, \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \right\rangle - D_\psi(\tilde{\mathbf{a}}^{(t+1)} \parallel \tilde{\mathbf{x}}^{(t)})$$

Combining the inequalities and summing over t , we have

$$\begin{aligned} G_1(\tilde{\mathbf{a}}^{(1)}) &\leq G_{T+1}(\tilde{\mathbf{a}}^{(T+1)}) + \sum_{t=1}^T \left(\left\langle \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} \right\rangle + \left\langle \tilde{\mathbf{a}}^{(t+1)} - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \right\rangle \right) \\ &\quad + \sum_{t=1}^T \left(-D_\psi(\tilde{\mathbf{x}}^{(t)} \parallel \tilde{\mathbf{a}}^{(t)}) - D_\psi(\tilde{\mathbf{a}}^{(t+1)} \parallel \tilde{\mathbf{x}}^{(t)}) \right). \end{aligned}$$

Observe that $G_1(\tilde{\mathbf{a}}^{(1)}) = \psi(\tilde{\mathbf{x}}^{(1)})$ and $G_{T+1}(\tilde{\mathbf{a}}^{(T+1)}) \leq -\langle \tilde{\mathbf{x}}, \tilde{\mathbf{U}}^{(T+1)} \rangle + \psi(\tilde{\mathbf{x}})$. Rearranging then proves the lemma. \square

Lemma 6.3. If $\eta \leq \frac{1}{50}$ and $\|\mathbf{u}^{(t)}\|_\infty \|\mathbf{x}\|_1, \|\mathbf{m}^{(t)}\|_\infty \|\mathbf{x}\|_1 \leq 1$ for all $\mathbf{x} \in \mathcal{X}$, then we have

$$\left\| \tilde{\mathbf{a}}^{(t+1)} - \tilde{\mathbf{x}}^{(t)} \right\|_t \leq 5\eta \left\| \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \right\|_{*,t} \leq 10\sqrt{2}\eta \leq 15\eta, \quad (6.14)$$

$$\left\| \tilde{\mathbf{x}}^{(t+1)} - \tilde{\mathbf{x}}^{(t)} \right\|_t \leq 5\eta \left\| 2\tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \right\|_{*,t} \leq 15\sqrt{2}\eta \leq 22\eta. \quad (6.15)$$

Proof. The second part of both inequalities is clear by definitions:

$$\begin{aligned} &\left\| \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \right\|_{*,t}^2 \\ &= \left(\lambda^{(t)} \left(\langle \mathbf{x}^{(t)}, \mathbf{u}^{(t)} \rangle - \langle \mathbf{x}^{(t-1)}, \mathbf{m}^{(t)} \rangle \right) \right)^2 + \sum_{r=1}^d \left(\mathbf{y}^{(t)}[r] \left(\mathbf{u}^{(t)}[r] - \mathbf{m}^{(t)}[r] \right) \right)^2 \\ &\leq 4(\lambda^{(t)})^2 + \frac{4}{\|\mathcal{X}\|_1^2} \sum_{r=1}^d \left(\mathbf{y}^{(t)}[r] \right)^2 \leq 8, \end{aligned}$$

where we use $\langle \mathbf{x}, \mathbf{u}^{(t)} \rangle, \langle \mathbf{x}, \mathbf{m}^{(t)} \rangle \leq 1$ for any \mathbf{x} by the hypotheses (using Cauchy-Schwarz), and similarly $|\mathbf{u}^{(\tau)}[r]|, |\mathbf{m}^{(\tau)}[r]| \leq \frac{1}{\|\mathcal{X}\|_1}$ for any time τ and any coordinate r by the assumption. Analogously,

$$\begin{aligned} &\left\| 2\tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \right\|_{*,t}^2 \\ &= \left(\lambda^{(t)} \left(2\langle \mathbf{x}^{(t)}, \mathbf{u}^{(t)} \rangle - \langle \mathbf{x}^{(t-1)}, \mathbf{u}^{(t-1)} \rangle \right) \right)^2 + \sum_{r=1}^d \left(\mathbf{y}^{(t)}[r] \left(2\mathbf{u}^{(t)}[r] - \mathbf{u}^{(t-1)}[r] \right) \right)^2 \end{aligned}$$

$$\leq 9(\lambda^{(t)})^2 + \frac{9}{\|\mathcal{X}\|_1^2} \sum_{r=1}^d \left(\mathbf{y}^{(t)}[r] \right)^2 \leq 18.$$

To prove the first inequality in Eq. (6.14), let $\mathcal{E}_t = \left\{ \mathbf{x} : \|\mathbf{x} - \tilde{\mathbf{x}}^{(t)}\|_t \leq 5\eta \|\tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)}\|_{*,t} \right\}$.

Noticing that $\tilde{\mathbf{a}}^{(t+1)}$ is the minimizer of the convex function G_{t+1} , to show $\tilde{\mathbf{a}}^{(t+1)} \in \mathcal{E}_t$, it suffices to show that for all $\tilde{\mathbf{x}}$ on the boundary of \mathcal{E}_t , we have $G_{t+1}(\tilde{\mathbf{x}}) \geq G_{t+1}(\tilde{\mathbf{x}}^{(t)})$. Indeed, using Taylor's theorem, for any such $\tilde{\mathbf{x}}$, there is a point $\boldsymbol{\xi}$ on the line segment between $\tilde{\mathbf{x}}^{(t)}$ and $\tilde{\mathbf{x}}$ such that

$$\begin{aligned} G_{t+1}(\tilde{\mathbf{x}}) &= G_{t+1}(\tilde{\mathbf{x}}^{(t)}) + \left\langle \nabla G_{t+1}(\tilde{\mathbf{x}}^{(t)}), \tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)} \right\rangle + \frac{1}{2} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_{\nabla^2 G_{t+1}(\boldsymbol{\xi})}^2 \\ &= G_{t+1}(\tilde{\mathbf{x}}^{(t)}) - \left\langle \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)}, \tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)} \right\rangle + \left\langle \nabla F_t(\tilde{\mathbf{x}}^{(t)}), \tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)} \right\rangle + \frac{1}{2} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_{\nabla^2 \psi(\boldsymbol{\xi})}^2 \\ &\geq G_{t+1}(\tilde{\mathbf{x}}^{(t)}) - \left\langle \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)}, \tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)} \right\rangle + \frac{1}{2} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_{\nabla^2 \psi(\boldsymbol{\xi})}^2 \\ &\hspace{20em} \text{(by the optimality of } \tilde{\mathbf{x}}^{(t)}) \\ &\geq G_{t+1}(\tilde{\mathbf{x}}^{(t)}) - \|\tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)}\|_{*,t} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_t + \frac{1}{2} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_{\nabla^2 \psi(\boldsymbol{\xi})}^2 \\ &\hspace{20em} \text{(by Hölder's inequality)} \\ &\geq G_{t+1}(\tilde{\mathbf{x}}^{(t)}) - \|\tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)}\|_{*,t} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_t + \frac{2}{9\eta} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_t^2 \quad (\star) \\ &= G_{t+1}(\tilde{\mathbf{x}}^{(t)}) + \frac{5}{9}\eta \|\tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)}\|_{*,t}^2 \quad \left(\|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_t = 5\eta \|\tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)}\|_{*,t} \right) \\ &\geq G_{t+1}(\tilde{\mathbf{x}}^{(t)}). \end{aligned}$$

Here, the inequality (\star) holds because Lemma 6.4 (together with the condition $\eta \leq \frac{1}{50}$) shows $\frac{1}{2}\tilde{\mathbf{x}}^{(t)}[i] \leq \tilde{\mathbf{x}}[i] \leq \frac{3}{2}\tilde{\mathbf{x}}^{(t)}[i]$, which implies $\frac{1}{2}\tilde{\mathbf{x}}^{(t)}[i] \leq \boldsymbol{\xi}[i] \leq \frac{3}{2}\tilde{\mathbf{x}}^{(t)}[i]$ as well, and thus $\nabla^2 \psi(\boldsymbol{\xi}) \succcurlyeq \frac{4}{9}\nabla^2 \psi(\tilde{\mathbf{x}}^{(t)})$. This finishes the proof for Eq. (6.14). The first inequality of Eq. (6.15) can be proven in the same manner. \square

A restatement of the second inequality of Lemma 6.3 was given earlier in this chapter, in Proposition 6.3.

Proposition 6.3 (Restated). For any time $t \in \mathbb{N}_{\geq 0}$ and learning rate $\eta \leq \frac{1}{50}$, if $\|\mathbf{u}^{(t)}\|_\infty \|\mathbf{x}\|_1, \|\mathbf{m}^{(t)}\|_\infty \|\mathbf{x}\|_1 \leq 1$ for all $\mathbf{x} \in \mathcal{X}$,

$$\left\| \begin{pmatrix} \lambda^{(t+1)} \\ \mathbf{y}^{(t+1)} \end{pmatrix} - \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\|_t \leq 22\eta.$$

Lemma 6.4. If $\tilde{\mathbf{x}}$ satisfies $\|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_t \leq \frac{1}{2}$, then $\frac{1}{2}\tilde{\mathbf{x}}^{(t)}[i] \leq \tilde{\mathbf{x}}[i] \leq \frac{3}{2}\tilde{\mathbf{x}}^{(t)}[i]$ for every coordinate i .

Proof. By definition, $\|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}\|_t \leq \frac{1}{2}$ implies for any i , $\frac{|\tilde{\mathbf{x}}[i] - \tilde{\mathbf{x}}^{(t)}[i]|}{\tilde{\mathbf{x}}^{(t)}[i]} \leq \frac{1}{2}$, and thus $\frac{1}{2}\tilde{\mathbf{x}}^{(t)}[i] \leq \tilde{\mathbf{x}}[i] \leq \frac{3}{2}\tilde{\mathbf{x}}^{(t)}[i]$. \square

Lemma 6.5. If $\eta \leq \frac{1}{50}$, then we have

$$\sum_{t=1}^T \left(D_\psi(\tilde{\mathbf{x}}^{(t)} \parallel \tilde{\mathbf{a}}^{(t)}) + D_\psi(\tilde{\mathbf{a}}^{(t+1)} \parallel \tilde{\mathbf{x}}^{(t)}) \right) \geq \frac{1}{27\eta} \sum_{t=1}^{T-1} \left\| \tilde{\mathbf{x}}^{(t+1)} - \tilde{\mathbf{x}}^{(t)} \right\|_t^2.$$

Proof. Recall $h(a) = a - 1 - \ln(a)$ and $D_\psi(\mathbf{x} \parallel \tilde{\mathbf{a}}) = \frac{1}{\eta} \sum_{i=1}^{d+1} h\left(\frac{\tilde{\mathbf{x}}[i]}{\tilde{\mathbf{a}}[i]}\right)$. We proceed as

$$\begin{aligned} & \sum_{t=1}^T \left(D_\psi(\tilde{\mathbf{x}}^{(t)} \parallel \tilde{\mathbf{a}}^{(t)}) + D_\psi(\tilde{\mathbf{a}}^{(t+1)} \parallel \tilde{\mathbf{x}}^{(t)}) \right) \\ & \geq \sum_{t=1}^{T-1} \left(D_\psi(\tilde{\mathbf{x}}^{(t+1)} \parallel \tilde{\mathbf{a}}^{(t+1)}) + D_\psi(\tilde{\mathbf{a}}^{(t+1)} \parallel \tilde{\mathbf{x}}^{(t)}) \right) \\ & = \frac{1}{\eta} \sum_{t=1}^{T-1} \sum_{i=1}^{d+1} \left(h\left(\frac{\tilde{\mathbf{x}}^{(t+1)}[i]}{\tilde{\mathbf{a}}^{(t+1)}[i]}\right) + h\left(\frac{\tilde{\mathbf{a}}^{(t+1)}[i]}{\tilde{\mathbf{x}}^{(t)}[i]}\right) \right) \\ & \geq \frac{1}{6\eta} \sum_{t=1}^{T-1} \sum_{i=1}^{d+1} \left(\frac{(\tilde{\mathbf{x}}^{(t+1)}[i] - \tilde{\mathbf{a}}^{(t+1)}[i])^2}{(\tilde{\mathbf{a}}^{(t+1)}[i])^2} + \frac{(\tilde{\mathbf{a}}^{(t+1)}[i] - \tilde{\mathbf{x}}^{(t)}[i])^2}{(\tilde{\mathbf{x}}^{(t)}[i])^2} \right) \\ & \hspace{15em} (h(y) \geq \frac{(y-1)^2}{6} \text{ for } y \in [\frac{1}{3}, 3]) \\ & \geq \frac{2}{27\eta} \sum_{t=1}^{T-1} \sum_{i=1}^{d+1} \left(\frac{(\tilde{\mathbf{x}}^{(t+1)}[i] - \tilde{\mathbf{a}}^{(t+1)}[i])^2}{(\tilde{\mathbf{x}}^{(t)}[i])^2} + \frac{(\tilde{\mathbf{a}}^{(t+1)}[i] - \tilde{\mathbf{x}}^{(t)}[i])^2}{(\tilde{\mathbf{x}}^{(t)}[i])^2} \right) \\ & \geq \frac{1}{27\eta} \sum_{t=1}^{T-1} \sum_{i=1}^{d+1} \left(\frac{(\tilde{\mathbf{x}}^{(t+1)}[i] - \tilde{\mathbf{x}}^{(t)}[i])^2}{(\tilde{\mathbf{x}}^{(t)}[i])^2} \right) = \frac{1}{27\eta} \sum_{t=1}^{T-1} \left\| \tilde{\mathbf{x}}^{(t+1)} - \tilde{\mathbf{x}}^{(t)} \right\|_t^2. \end{aligned}$$

Here, the third and the fourth inequalities hold because by [Lemma 6.3](#) and [Lemma 6.4](#), we have $\frac{1}{2} \leq \frac{\tilde{\mathbf{a}}^{(t+1)}[i]}{\tilde{\mathbf{x}}^{(t)}[i]} \leq \frac{3}{2}$ and $\frac{1}{2} \leq \frac{\tilde{\mathbf{x}}^{(t+1)}[i]}{\tilde{\mathbf{a}}^{(t+1)}[i]} \leq \frac{3}{2}$, and thus $\frac{1}{3} \leq \frac{\tilde{\mathbf{x}}^{(t+1)}[i]}{\tilde{\mathbf{a}}^{(t+1)}[i]} \leq 3$. \square

We are now ready to establish [Proposition 6.2](#).

Proposition 6.2 (Restated; Regret of FTRL in local norms). Let $\tilde{\text{Reg}}^{(T)}$ be the regret cumulated up to time T by the internal FTRL algorithm. If $\|\mathbf{u}^{(t)}\|_\infty \|\mathbf{x}\|_1 \leq 1$ at all times $t \in \llbracket T \rrbracket$, then for any time horizon $T \in \mathbb{N}_{\geq 0}$ and learning rate $\eta \leq \frac{1}{50}$,

$$\tilde{\text{Reg}}^{(T)} \leq \frac{(d+1) \log T}{\eta} + 5\eta \sum_{t=1}^T \left\| \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \right\|_{*,t}^2 - \frac{1}{27\eta} \sum_{t=1}^{T-1} \left\| \begin{pmatrix} \lambda^{(t+1)} \\ \mathbf{y}^{(t+1)} \end{pmatrix} - \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\|_t^2.$$

Proof. For any comparator $\tilde{\mathbf{x}} \in \tilde{\mathcal{X}}$, define $\tilde{\mathbf{x}}' = \frac{T-1}{T} \cdot \tilde{\mathbf{x}} + \frac{1}{T} \cdot \tilde{\mathbf{x}}^{(1)} \in \tilde{\mathcal{X}}$, where we recall $\tilde{\mathbf{x}}^{(1)} = \arg \min_{\mathbf{x} \in \tilde{\mathcal{X}}} F_1(\mathbf{x}) = \arg \min_{\mathbf{x} \in \tilde{\mathcal{X}}} \psi(\mathbf{x})$. Then, we have

$$\begin{aligned} \sum_{t=1}^T \langle \tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} \rangle &= \sum_{t=1}^T \langle \tilde{\mathbf{x}} - \tilde{\mathbf{x}}', \tilde{\mathbf{u}}^{(t)} \rangle + \sum_{t=1}^T \langle \tilde{\mathbf{x}}' - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} \rangle \\ &= \frac{1}{T} \sum_{t=1}^T \langle \tilde{\mathbf{x}} - \tilde{\mathbf{x}}^{(1)}, \tilde{\mathbf{u}}^{(t)} \rangle + \sum_{t=1}^T \langle \tilde{\mathbf{x}}' - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} \rangle \\ &\leq 4 + \sum_{t=1}^T \langle \tilde{\mathbf{x}}' - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} \rangle, \end{aligned}$$

where the last inequality follows from Cauchy-Schwarz together with the assumption that $\|\mathbf{u}^{(t)}\|_\infty \leq \frac{1}{\|\tilde{\mathcal{X}}\|_1}$.

Now, from [Lemma 6.2](#), the last term $\sum_{t=1}^T \langle \tilde{\mathbf{x}}' - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} \rangle$ (cumulative regret against $\tilde{\mathbf{x}}'$) is bounded by

$$\begin{aligned} \sum_{t=1}^T \langle \tilde{\mathbf{x}}' - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} \rangle &\leq \psi(\tilde{\mathbf{x}}') - \psi(\tilde{\mathbf{x}}^{(1)}) + \sum_{t=1}^T \langle \tilde{\mathbf{a}}^{(t+1)} - \tilde{\mathbf{x}}^{(t)}, \tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)} \rangle \\ &\quad - \sum_{t=1}^T \left(D_\psi(\tilde{\mathbf{x}}^{(t)} \parallel \tilde{\mathbf{a}}^{(t)}) + D_\psi(\tilde{\mathbf{a}}^{(t+1)} \parallel \tilde{\mathbf{x}}^{(t)}) \right). \end{aligned}$$

For the term $\psi(\tilde{\mathbf{x}}') - \psi(\tilde{\mathbf{x}}^{(1)})$, a direct calculation using definitions shows

$$\psi(\tilde{\mathbf{x}}') - \psi(\tilde{\mathbf{x}}^{(1)}) = \frac{1}{\eta} \sum_{i=1}^{d+1} \log \frac{\tilde{\mathbf{x}}^{(1)}[i]}{\tilde{\mathbf{x}}'[i]} \leq \frac{d+1}{\eta} \log T.$$

For the other terms, we apply [Lemma 6.3](#) and [Lemma 6.5](#), which completes the proof. \square

6.A.3 Proof of Corollary 6.1

Next, we establish an RVU bound in the original (unlifted) space, namely Corollary 6.1. To this end, we first proceed with the proof of Lemma 6.1, which boils down to the following simple claim.

Lemma 6.6. Let $(\lambda, \mathbf{y}), (\lambda', \mathbf{y}') \in \tilde{\mathcal{X}} \cap \mathbb{R}_{>0}^{d+1}$ be arbitrary points such that

$$\left\| \begin{pmatrix} \lambda' \\ \mathbf{y}' \end{pmatrix} - \begin{pmatrix} \lambda \\ \mathbf{y} \end{pmatrix} \right\|_{(\lambda, \mathbf{y})} \leq \frac{1}{2}.$$

Then,

$$\left\| \frac{\mathbf{y}}{\lambda} - \frac{\mathbf{y}'}{\lambda'} \right\|_1 \leq 4 \|\mathcal{X}\|_1 \cdot \left\| \begin{pmatrix} \lambda' \\ \mathbf{y}' \end{pmatrix} - \begin{pmatrix} \lambda \\ \mathbf{y} \end{pmatrix} \right\|_{(\lambda, \mathbf{y})}.$$

Proof. Let μ be defined as

$$\mu := \max \left\{ \left| \frac{\lambda'}{\lambda} - 1 \right|, \max_{r \in \llbracket d \rrbracket} \left| \frac{\mathbf{y}'[r]}{\mathbf{y}[r]} - 1 \right| \right\}. \quad (6.16)$$

By definition, $\left| \frac{\lambda'}{\lambda} - 1 \right| \leq \mu$, which in turn implies that

$$(1 - \mu)\lambda \leq \lambda' \leq (1 + \mu)\lambda. \quad (6.17)$$

Similarly, for any $r \in \llbracket d \rrbracket$,

$$(1 - \mu)\mathbf{y}[r] \leq \mathbf{y}'[r] \leq (1 + \mu)\mathbf{y}[r]. \quad (6.18)$$

As a result, combining (6.17) and (6.18) we get that for any $r \in \llbracket d \rrbracket$,

$$\frac{\mathbf{y}'[r]}{\lambda'} - \frac{\mathbf{y}[r]}{\lambda} \leq \left(\frac{1 + \mu}{1 - \mu} - 1 \right) \frac{\mathbf{y}[r]}{\lambda} \leq 4\mu \frac{\mathbf{y}[r]}{\lambda} = 4\mu \mathbf{x}[r],$$

since $\mu \leq \frac{1}{2}$. Similarly, by (6.17) and (6.18),

$$\frac{\mathbf{y}[r]}{\lambda} - \frac{\mathbf{y}'[r]}{\lambda'} \leq \left(1 - \frac{1 - \mu}{1 + \mu} \right) \frac{\mathbf{y}[r]}{\lambda} \leq 2\mu \frac{\mathbf{y}[r]}{\lambda} = 2\mu \mathbf{x}[r].$$

Thus, it follows that $\left| \frac{\mathbf{y}'[r]}{\lambda'} - \frac{\mathbf{y}[r]}{\lambda} \right| \leq 4\mu \mathbf{x}[r]$, in turn implying that

$$\|\mathbf{x}' - \mathbf{x}\|_1 = \sum_{r=1}^d \left| \frac{\mathbf{y}'[r]}{\lambda'} - \frac{\mathbf{y}[r]}{\lambda} \right| \leq 4\mu \sum_{r=1}^d \mathbf{x}[r] \leq 4\mu \|\mathcal{X}\|_1. \quad (6.19)$$

Moreover, by definition of (6.16),

$$(\mu)^2 \leq \left\| \begin{pmatrix} \lambda' \\ \mathbf{y}' \end{pmatrix} - \begin{pmatrix} \lambda \\ \mathbf{y} \end{pmatrix} \right\|_t^2.$$

Finally, combining this bound with (6.19) concludes the proof. \square

Lemma 6.1 (Restated). For any time $t \in \mathbb{N}_{\geq 0}$ and learning rate $\eta \leq \frac{1}{50}$, if $\|\mathbf{u}^{(t)}\|_\infty \|\mathbf{x}\|_1 \leq 1$,

$$\|\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\|_1 \leq 4\|\mathcal{X}\|_1 \left\| \begin{pmatrix} \lambda^{(t+1)} \\ \mathbf{y}^{(t+1)} \end{pmatrix} - \begin{pmatrix} \lambda^{(t)} \\ \mathbf{y}^{(t)} \end{pmatrix} \right\|_t.$$

Proof. Since $\eta \leq \frac{1}{50}$ by assumption, we have

$$\left\| \mathbf{x}^{(t+1)} - \mathbf{x}^{(t)} \right\|_t \leq 22\eta < \frac{1}{2}.$$

Hence, we are in the domain of applicability of Lemma 6.6, which immediately yields the statement. \square

Corollary 6.1 (Restated; RVU bound in the original (unlifted) space). Fix any time $T \in \mathbb{N}_{\geq 0}$, and suppose that $\|\mathbf{u}^{(t)}\|_\infty \leq B$ for any $t \in \llbracket T \rrbracket$. If $\eta \leq \frac{1}{256B\|\mathcal{X}\|_1}$,

$$\begin{aligned} \tilde{\text{Reg}}^{(T)} &\leq 6B\|\mathcal{X}\|_1 + \frac{(d+1)\log T}{\eta} + 16\eta\|\mathcal{X}\|_1^2 \sum_{t=1}^{T-1} \|\mathbf{u}^{(t)} - \mathbf{m}^{(t)}\|_\infty^2 \\ &\quad - \frac{1}{512\eta\|\mathcal{X}\|_1^2} \sum_{t=1}^{T-1} \|\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\|_1^2. \end{aligned}$$

Proof. At first, assume that $\|\mathbf{u}^{(t)}\|_\infty \leq 1/\|\mathcal{X}\|_1$. By definition of the induced dual local norm in (6.2),

$$\|\tilde{\mathbf{u}}^{(t)} - \tilde{\mathbf{m}}^{(t)}\|_{*,t}^2 \leq \left(\langle \mathbf{x}^{(t)}, \mathbf{u}^{(t)} \rangle - \langle \mathbf{x}^{(t)}, \mathbf{m}^{(t)} \rangle \right)^2 (\lambda^{(t)})^2 + \sum_{r=1}^d \mathbf{y}[r]^2 (\mathbf{u}^{(t)}[r] - \mathbf{m}^{(t)}[r])^2$$

$$\begin{aligned} &\leq \langle \mathbf{x}^{(t)}, \mathbf{u}^{(t)} - \mathbf{m}^{(t)} \rangle^2 + \sum_{r=1}^d \mathbf{x}[r]^2 (\mathbf{u}^{(t)}[r] - \mathbf{m}^{(t)}[r])^2 \\ &\leq 2\|\mathcal{X}\|_1^2 \cdot \|\mathbf{u}^{(t)} - \mathbf{m}^{(t)}\|_\infty^2, \end{aligned} \quad (6.20)$$

Combining with [Proposition 6.2](#) and [Lemma 6.1](#), we get that $\tilde{\text{Reg}}^{(T)}$ is upper bounded by

$$6 + \frac{(d+1)\log T}{\eta} + 16\eta\|\mathcal{X}\|_1^2 \sum_{t=1}^T \|\mathbf{u}^{(t)} - \mathbf{m}^{(t)}\|_\infty^2 - \frac{1}{512\eta\|\mathcal{X}\|_1^2} \sum_{t=1}^{T-1} \|\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\|_1^2.$$

Finally, we relax the assumption that $\|\mathbf{u}^{(t)}\|_\infty, \|\mathbf{m}^{(t)}\|_\infty \leq 1/\|\mathcal{X}\|_1$. In that case, one can reduce to the above analysis by first rescaling all utilities and predictions by the factor $1/(B\|\mathcal{X}\|_1)$ —which in turn is equivalent to rescaling the learning rate η by $1/(B\|\mathcal{X}\|_1)$. We then need to correct for the fact that the norm of the difference of utilities gets rescaled by a factor $1/(B\|\mathcal{X}\|_1)^2$, and that the regret $\tilde{\text{Reg}}^{(T)}$ with respect to the original utilities is a factor $B\|\mathcal{X}\|_1$ larger than the regret measured on the rescaled utilities. Taking these considerations into account leads to the statement. \square

6.A.4 Proof of Theorems 6.2 and 6.3

Finally, we are ready to establish [Theorem 6.3](#). To this end, the main ingredient is the bound on the second-order path lengths predicted by [Theorem 6.2](#), which is recalled below.

Theorem 6.2 (Restated). If all players follow LRL-OFTRL with learning rate

$$\eta \leq \min \left\{ \frac{1}{256B\|\mathcal{X}\|_1}, \frac{1}{128nL\|\mathcal{X}\|_1^2} \right\},$$

where $\|\mathcal{X}\|_1 := \max_{i \in \llbracket n \rrbracket} \|\mathcal{X}_i\|_1$, then

$$\sum_{i=1}^n \sum_{t=1}^{T-1} \|\mathbf{x}_i^{(t+1)} - \mathbf{x}_i^{(t)}\|_1^2 \leq 6144n\eta B\|\mathcal{X}\|_1^3 + 1024n(d+1)\|\mathcal{X}\|_1^2 \log T. \quad (6.3)$$

Proof. For any player $i \in \llbracket n \rrbracket$,

$$\left(\|\mathbf{u}_i^{(t+1)} - \mathbf{u}_i^{(t)}\|_\infty \right)^2 \leq \left(L \sum_{j=1}^n \|\mathbf{x}_j^{(t+1)} - \mathbf{x}_j^{(t)}\|_1 \right)^2 \leq L^2 n \sum_{j=1}^n \|\mathbf{x}_j^{(t+1)} - \mathbf{x}_j^{(t)}\|_1^2,$$

by Jensen's inequality. Hence, by [Corollary 6.1](#) the regret $\text{Reg}_i^{(T)}$ of each player $i \in \llbracket n \rrbracket$ can be

upper bounded by

$$6B\|\mathcal{X}\|_1 + \frac{(d+1)\log T}{\eta} + 16\eta\|\mathcal{X}\|_1^2 L^2 n \sum_{j=1}^n \sum_{t=1}^{T-1} \|\mathbf{x}_j^{(t+1)} - \mathbf{x}_j^{(t)}\|_1^2 - \frac{1}{512\eta\|\mathcal{X}\|_1^2} \sum_{t=1}^{T-1} \|\mathbf{x}_i^{(t+1)} - \mathbf{x}_i^{(t)}\|_1^2,$$

Summing over all players $i \in \llbracket n \rrbracket$, we have that

$$\begin{aligned} \sum_{i=1}^n \tilde{\text{Reg}}_i^{(T)} &\leq 6nB\|\mathcal{X}\|_1 + n \frac{(d+1)\log T}{\eta} \\ &\quad + \sum_{i=1}^n \left(16\eta\|\mathcal{X}\|_1^2 L^2 n^2 - \frac{1}{512\eta\|\mathcal{X}\|_1^2} \right) \sum_{t=1}^{T-1} \|\mathbf{x}_i^{(t+1)} - \mathbf{x}_i^{(t)}\|_1^2 \\ &\leq 6nB\|\mathcal{X}\|_1 + n \frac{(d+1)\log T}{\eta} - \frac{1}{1024\eta\|\mathcal{X}\|_1^2} \sum_{i=1}^n \sum_{t=1}^{T-1} \|\mathbf{x}_i^{(t+1)} - \mathbf{x}_i^{(t)}\|_1^2, \end{aligned}$$

since $\eta \leq \frac{1}{256nL\|\mathcal{X}\|_1^2}$. Finally, the theorem follows since $\sum_{i=1}^n \tilde{\text{Reg}}_i^{(T)} \geq 0$, which in turn follows directly from [Theorem 6.1](#). \square

Theorem 6.3 (Restated). If all players use LRL-OFTRL within the COLS (that is, with $\mathbf{m}^{(t)} = \mathbf{u}^{(t-1)}$; cf. [Section 3.2.1](#)) with learning rate

$$\eta = \min \left\{ \frac{1}{256B\|\mathcal{X}\|_1}, \frac{1}{128nL\|\mathcal{X}\|_1^2} \right\},$$

then for any $T \in \mathbb{N}_{\geq 0}$ the regret $\text{Reg}_i^{(T)}$ of each player $i \in \llbracket n \rrbracket$ can be bounded as

$$\text{Reg}_i^{(T)} \leq 12B\|\mathcal{X}\|_1 + 256(d+1) \max\{nL\|\mathcal{X}\|_1^2, 2B\|\mathcal{X}\|_1\} \log T. \quad (6.4)$$

Furthermore, the algorithm can be adaptive so that if player i is instead facing adversarial utilities, then $\text{Reg}_i^{(T)} = \mathcal{O}_T(\sqrt{T})$.

Proof. First of all, by [Assumption 6.1](#) we have that for any player $i \in \llbracket n \rrbracket$,

$$\|\mathbf{u}_i^{(t+1)} - \mathbf{u}_i^{(t)}\|_\infty^2 \leq \left(L \sum_{j=1}^n \|\mathbf{x}_j^{(t+1)} - \mathbf{x}_j^{(t)}\|_1 \right)^2 \leq L^2 n \sum_{j=1}^n \|\mathbf{x}_j^{(t+1)} - \mathbf{x}_j^{(t)}\|_1^2.$$

Hence, summing over all t ,

$$\begin{aligned} \sum_{t=1}^{T-1} \|\mathbf{u}_i^{(t+1)} - \mathbf{u}_i^{(t)}\|_\infty^2 &\leq L^2 n \sum_{t=1}^{T-1} \sum_{j=1}^n \|\mathbf{x}_j^{(t+1)} - \mathbf{x}_j^{(t)}\|_1^2 \\ &\leq 6144n^2 L^2 \eta B \|\mathcal{X}\|_1^3 + 1024n^2 L^2 (d+1) \|\mathcal{X}\|_1^2 \log T, \end{aligned}$$

where the last bound uses [Theorem 6.2](#). As a result, from [Corollary 6.1](#), if $\eta = \frac{1}{128nL\|\mathcal{X}\|_1^2}$,

$$\begin{aligned} \tilde{\text{Reg}}_i^{(T)} &\leq 6B\|\mathcal{X}\|_1 + \frac{(d+1)\log T}{\eta} + 16\eta\|\mathcal{X}\|_1^2 \sum_{t=1}^{T-1} \|\mathbf{u}_i^{(t+1)} - \mathbf{u}_i^{(t)}\|_\infty^2 \\ &\leq 12B\|\mathcal{X}\|_1 + 256(d+1)nL\|\mathcal{X}\|_1^2 \log T. \end{aligned}$$

Thus, the bound on $\text{Reg}_i^{(T)}$ follows directly since $\text{Reg}_i^{(T)} \leq \tilde{\text{Reg}}_i^{(T)}$ by [Theorem 6.1](#). The case where $\eta = \frac{1}{256B\|\mathcal{X}\|_1}$ is analogous.

Next, let us focus on the adversarial bound. Each player can simply check whether there exists a time $t \in \llbracket T \rrbracket$ such that

$$\sum_{\tau=1}^{(t-1)} \|\mathbf{u}_i^{(\tau+1)} - \mathbf{u}_i^{(\tau)}\|_\infty^2 > 6144n^2 L^2 \eta B \|\mathcal{X}\|_1^3 + 1024n^2 L^2 (d+1) \|\mathcal{X}\|_1^2 \log t. \quad (6.21)$$

In particular, we know from [Theorem 6.2](#) that when all players follow the prescribed protocol (6.21) will never be satisfied. On the other hand, if there exists time t so that (6.21) holds, then it suffices to switch to any no-regret learning algorithm tuned to face adversarial utilities. \square

Chapter 7

State-of-the-art regret dependence on game size via kernelization

7.1 Contributions and related work

In this chapter we establish a new technique for constructing no-external-regret dynamics for imperfect-information extensive-form games, which leads to the current state-of-the-art regret bounds in terms of dependence from the game dimensions, while at the same time enjoying polylogarithmic (albeit with a worse exponent than the LRL-OFTRL algorithm we presented in [Chapter 6](#)) regret in self-play, and polynomial-time iterations. However, in addition to the benefits above, I believe the technique is remarkable in that (i) it surprisingly shows that a long-held popular wisdom in the literature of imperfect-information extensive-form games is inaccurate, and (ii) reduces the gap between learning in sequential and nonsequential games, by showing that in some cases the former (which is substantially harder both conceptually and in a complexity-theoretic sense in general) can be efficiently reduced to the latter.

In order to understand our technique, it is beneficial to introduce the concept of *normal-form equivalent* of the tree-form decision problem faced by a player. In short, because the polytope of (sequence-form) strategies is the convex polytope spanned by the deterministic strategies, picking a (sequence-form) strategy for the game is equivalent to picking a convex combination of vertices. Such a process is no longer tree-form, but is rather a single decision with as many “actions” as deterministic strategies for the player. By doing away with the tree-form structure in favor of a single decision node, learning in the normal-form equivalent of a TFDP can then be achieved by using any learning algorithm for nonsequential games, opening the way to reaping the benefits of techniques for nonsequential games in imperfect-information extensive-form games too. Of course, the catch of the above approach is that it comes at the cost of an exponential blowup of the strategy space, as the flattened (normal-form equivalent) decision problem typically has

exponentially many actions—as such is the number of deterministic strategies in a TFDP. For this reason, the normal-form representation was viewed as impractical, historically leading the communities working on learning in normal-form and imperfect-information extensive-form games to follow separate tracks, with the latter community often having to catch up with advances (e.g., last-iterate convergence and predictive regret bounds) from the former, larger community.

We contradict popular belief and show that it is possible to work with the normal-form equivalent of a TFDP efficiently: we provide a kernel-based reduction that allows us to simulate predictive multiplicative weights update OMWU algorithm—arguably the premier learning algorithm for nonsequential games; cf. Section 3.3.1—on the normal-form representation, using only linear (in the TFDP size) time per iteration. Our algorithm, *Kernelized OMWU* (KOMWU), achieves all the guarantees provided by the various normal-form results mentioned previously, as well as any future results on OMWU for nonsequential games.

As an unexpected byproduct, KOMWU obtains new state-of-the-art regret bounds among all online learning algorithms for TFDPs (see also Table 7.1); we improve the dependence on the maximum ℓ_1 norm $\|\mathcal{Q}\|_1$ over the sequence-form polytope \mathcal{Q} from $\|\mathcal{Q}\|_1^2$ to $\|\mathcal{Q}\|_1$ (for the non-optimistic version we improve it from $\|\mathcal{Q}\|_1$ to $\sqrt{\|\mathcal{Q}\|_1}$).

Algorithm		Per-player regret bound	Last-iter. conv. [†]
CFR (regret matching / regret matching ⁺)	Chapter 4	$\mathcal{O}(\sqrt{A} \ \mathcal{Q}\ _1 T^{1/2})$	no
CFR (MWU)	Chapter 4	$\mathcal{O}(\sqrt{\log A} \ \mathcal{Q}\ _1 T^{1/2})$	no
FTRL/ OMD (dilated entropy)	Chapter 5	$\mathcal{O}(\sqrt{\log A} 2^{D/2} \ \mathcal{Q}\ _1 T^{1/2})$	no
FTRL/ OMD (dilatable global entropy)	Chapter 5	$\mathcal{O}(\sqrt{\log A} \ \mathcal{Q}\ _1 T^{1/2})$	no
Kernelized MWU	(this chapter)	$\mathcal{O}(\sqrt{\log A} \sqrt{\ \mathcal{Q}\ _1} T^{1/2})$	no
Predictive CFR	Chapter 4	$\mathcal{O}_T(\sqrt{T})$	no
Predictive FTRL/ OMD (dilated entropy)	Chapter 5	$\mathcal{O}(\sqrt{m} \log(A) 2^D \ \mathcal{Q}\ _1^2 T^{1/4})$	yes*
Predictive FTRL/ OMD (dilatable gl. ent.)	Chapter 5	$\mathcal{O}(\sqrt{m} \log(A) \ \mathcal{Q}\ _1^2 T^{1/4})$	no
LRL-OFTRL [‡]	Chapter 6	$\mathcal{O}(m \text{poly}(A) \ \mathcal{Q}\ _1^2 \log T)$	no
Kernelized OMWU	(this chapter)	$\mathcal{O}(m \log(A) \ \mathcal{Q}\ _1 \log^4(T))$	yes

Table 7.1: Properties of various no-regret algorithms for imperfect-information extensive-form games.

All algorithms take linear time to perform an iteration. The first set of rows are for non-predictive algorithms. The second set of rows are for predictive algorithms. The regret bounds are per player and apply to multiplayer general-sum games. They depend on the maximum number of actions A available at any decision node, the maximum ℓ_1 norm $\|\mathcal{Q}\|_1 = \max_{q \in \mathcal{Q}} \|q\|_1$ over the player’s sequence-form strategy polytope \mathcal{Q} , the depth D of the decision polytope, and the number of players m . Optimistic algorithms have better asymptotic regret, but worse dependence on the game constants m , A , and $\|\mathcal{Q}\|_1$. Note that our algorithms achieve better dependence on $\|\mathcal{Q}\|_1$ compared to all existing algorithms. [†]Last-iterate convergence results are for two-player zero-sum games, and some results rely on the assumption of a unique Nash equilibrium—see Section 7.5 for details. [‡]We remark that LRL-OFTRL enjoys polynomial-time iterations (ignoring a $\log \log T$ dependence), but *not* linear-time iterations unlike the other methods in the table. *See C.-W. Lee, Kroer, and Luo (2021).

Due to the connection between regret minimization and convergence to Nash equilibrium,

this also improves the state-of-the-art bounds for converging to a Nash equilibrium at either a rate of $1/\sqrt{T}$ or $1/T$ by the same factor. Moreover, KOMWU achieves last-iterate convergence, and as such it is the first algorithm to achieve linear-rate last-iterate convergence with a learning rate that does not become impractically-small as the game grows large (albeit under a restrictive uniqueness assumption).

More generally, we remark that KOMWU can simulate OMWU in general *0/1-polyhedral sets* (of which the decision sets for TFDPs are a special case): a decision set $\Omega \subseteq \mathbb{R}^d$ which is convex and polyhedral, and whose vertices are all contained in $\{0, 1\}^d$. KOMWU reduces the problem of running OMWU on the vertices of the polyhedral set to $d + 1$ evaluations of what we call the *0/1-polyhedral kernel*. Thus, given an efficient algorithm for performing these kernel evaluations, KOMWU enables one to get all the benefits of running MWU or OMWU on the simplex of vertices, while retaining the crucial property that each iteration of OMWU can be performed efficiently.

Regret minimization over 0/1 polyhedral sets, of which the sequence-form strategy polytope is an example, is closely related to online combinatorial optimization problems (Audibert, Bubeck, and Lugosi, 2014), where the decision maker (randomly) selects a 0/1 vertex in each round instead of a point in the convex hull of the set of vertices, and the regret is measured in expectation. We review the approaches most closely related to the use of MWU here. One approach similar to our KOMWU is to perform MWU over vertices (e.g., Cesa-Bianchi and Lugosi (2012)); the remaining problem is whether there is an efficient way to maintain and sample from the weights. Such efficient implementations have been shown in many instances such as paths (Takimoto and Warmuth, 2003), spanning trees (Koo, Globerson, Carreras Pérez, and Collins, 2007), and m -set (Warmuth and Kuzmin, 2008). The work by Takimoto and Warmuth (2003) is the closest to the kernel approach we adopted, where they show how to produce MWU iterates for paths in directed graphs. The kernelized method described in this chapter can be seen as a significant extension of their approach to general 0/1 polyhedral games, unifying many of the previous results listed above. This unification not only results in important applications to imperfect-information extensive-form games, but also leads to improvement to previously studied problems such as n -sets.

7.2 A natural reduction: Running OMWU on the vertices of the strategy set

Since the set of sequence-form strategies \mathcal{Q} in a TFDP is a convex polytope, the decision problem of picking a sequence-form strategy $x^{(t)} \in \mathcal{Q}$ can be equivalently thought of as the decision problem of picking a convex combination $\lambda^{(t)}$ of vertices of \mathcal{Q} —that is, of deterministic strategies $\pi \in \Pi$ (Definition 2.5).

This intuition is correct: as we show next, a learning algorithm \mathcal{R} for $\mathcal{Q} \subseteq \mathbb{R}^\Sigma$ can be

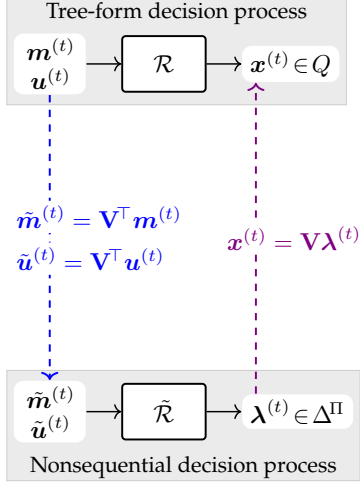


Figure 7.1: Overview of reduction template from \mathcal{Q} to Δ^Π . The matrix \mathbf{V} has the vertices Π of \mathcal{Q} as columns.

Algorithm 7.1: Vertex OMWU

Data: Learning rates $\eta^{(t)} > 0$

- 1 $\mathbf{u}^{(0)}, \mathbf{m}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^\Sigma; \boldsymbol{\lambda}^{(0)} \leftarrow \frac{1}{|\Pi|} \mathbf{1} \in \Delta^\Pi$ [\triangleright Initialization]
- 2 **function** NextStrategy($\mathbf{m}^{(t)} \in \mathbb{R}^\Sigma$) [\triangleright Set $\mathbf{m}^{(t)} = \mathbf{0}$ for non-predictive variant]
- 3 $\mathbf{w}^{(t)} \leftarrow \mathbf{u}^{(t-1)} - \mathbf{m}^{(t-1)} + \mathbf{m}^{(t)}$
- 4 **for** $\boldsymbol{\pi} \in \Pi$ **do** [\triangleright Run the OMWU update on $\boldsymbol{\lambda}$ using $\mathcal{A} = \Pi$]
- 5 $\lambda^{(t)}[\boldsymbol{\pi}] := \frac{\lambda^{(t-1)}[\boldsymbol{\pi}] \cdot \exp\{\eta^{(t)} \langle \mathbf{w}^{(t)}, \boldsymbol{\pi} \rangle\}}{\sum_{\boldsymbol{\pi}' \in \Pi} \lambda^{(t-1)}[\boldsymbol{\pi}'] \cdot \exp\{\eta^{(t)} \langle \mathbf{w}^{(t)}, \boldsymbol{\pi}' \rangle\}}$ (7.1)
- 6 [\triangleright Compute new convex combination of vertices]
- 7 $\mathcal{Q} \ni \mathbf{x}^{(t)} := \sum_{\boldsymbol{\pi} \in \Pi} \lambda^{(t)}[\boldsymbol{\pi}] \cdot \boldsymbol{\pi}$ (7.2)
- 8 **return** $\mathbf{x}^{(t)}$
- 8 **function** ObserveUtility($\mathbf{u}^{(t)} \in \mathbb{R}^\Sigma$)
- 9 $t \leftarrow t + 1$

constructed from any learning algorithm $\tilde{\mathcal{R}}$ for the set of *vertices* Π . Indeed, consider the following conceptual template:

- Each distribution $\boldsymbol{\lambda}^{(t)} \in \Delta^\Pi$ output by $\tilde{\mathcal{R}}$ defines the convex combination of vertices of \mathcal{Q} and is used to construct the sequence-form strategy

$$\mathcal{Q} \ni \mathbf{x}^{(t)} := \sum_{\boldsymbol{\pi} \in \Pi} \lambda^{(t)}[\boldsymbol{\pi}] \boldsymbol{\pi}.$$

- Each prediction $\mathbf{m}^{(t)} \in \mathbb{R}^\Sigma$ and utility vector $\mathbf{u}^{(t)}$ fed into \mathcal{R} is used to construct a prediction $\tilde{\mathbf{m}}$ and utility vector $\tilde{\mathbf{u}} \in \mathbb{R}^\Pi$ according to

$$\begin{aligned} \tilde{\mathbf{m}}^{(t)}[\boldsymbol{\pi}] &:= \langle \mathbf{m}^{(t)}, \boldsymbol{\pi} \rangle, \\ \tilde{\mathbf{u}}^{(t)}[\boldsymbol{\pi}] &:= \langle \mathbf{u}^{(t)}, \boldsymbol{\pi} \rangle \end{aligned} \quad \forall \boldsymbol{\pi} \in \Pi.$$

By letting \mathbf{V} denote the matrix whose columns are the deterministic strategies $\boldsymbol{\pi} \in \Pi$, we can rewrite the above template more concisely by saying that each output $\boldsymbol{\lambda}^{(t)} \in \Delta^\Pi$ of $\tilde{\mathcal{R}}$ is transformed into the output $\mathbf{x}^{(t)} := \mathbf{V}\boldsymbol{\lambda}^{(t)} \in \mathcal{Q}$ of \mathcal{R} , and each prediction and utility vector $\mathbf{m}^{(t)}, \mathbf{u}^{(t)} \in \mathbb{R}^\Sigma$ of \mathcal{R} are transformed into a prediction and utility vector $\tilde{\mathbf{m}}^{(t)} := \mathbf{V}^\top \mathbf{m}^{(t)}$ and $\tilde{\mathbf{u}}^{(t)} := \mathbf{V}^\top \mathbf{u}^{(t)}$ for $\tilde{\mathcal{R}}$, respectively (see also Figure 7.1). The correctness of the above template is given by the observation that, since $\tilde{\mathbf{u}}^{(t)} := \mathbf{V}^\top \mathbf{u}^{(t)}$ and $\mathbf{x}^{(t)} := \mathbf{V}\boldsymbol{\lambda}^{(t)}$ by construction, then the

following equality of regrets holds.

Lemma 7.1. The regret $\tilde{\text{Reg}}^{(T)} := \sum_{t=1}^T \langle \tilde{\mathbf{u}}^{(t)}, \boldsymbol{\lambda}^{(t)} \rangle$ cumulated by $\tilde{\mathcal{R}}$ is equal, at all times T , to the regret $\text{Reg}^{(T)} := \sum_{t=1}^T \langle \mathbf{u}^{(t)}, \mathbf{x}^{(t)} \rangle$.

In this chapter we are particularly interested in the algorithm obtained by using the above construction for the specific choice of OMWU (Section 3.3.1) as the algorithm $\tilde{\mathcal{R}}$. We coin *Vertex OMWU* the resulting learning algorithm \mathcal{R} . The algorithm is summarized in pseudocode in Algorithm 7.1.

7.3 Kernelized multiplicative weights

Kernelized OMWU (KOMWU) gives a way of efficiently simulating the Vertex OMWU algorithm given in Algorithm 7.1 and described in Section 7.2.^[7.a]

We start by defining the sequence-form feature map and associated kernel in Section 7.3.1, and move on to describe how the kernel enables efficient simulation of Vertex OMWU in Section 7.3.2.

7.3.1 The sequence-form kernel

As the name suggests, at its heart KOMWU uses a *kernel trick* to simulate the steps of Vertex OMWU without actually operating on the exponentially large probability simplex of deterministic strategies. Let $\sigma \in \pi$ be a shorthand notation for the set of sequences $\sigma \in \Sigma$ such that $\pi[\sigma] = 1$, that is, those sequences that are supported by the deterministic strategy π .

Definition 7.1 (Sequence-form feature map). The *sequence-form feature map* $\phi_{\mathcal{Q}} : \mathbb{R}^{\Sigma} \rightarrow \mathbb{R}^{\Pi}$ is the function such that

$$\phi_{\mathcal{Q}}(\mathbf{x})[\boldsymbol{\pi}] := \prod_{\sigma \in \boldsymbol{\pi}} \mathbf{x}[\sigma] \quad \forall \mathbf{x} \in \mathbb{R}^{\Sigma}, \boldsymbol{\pi} \in \Pi. \quad (7.3)$$

Definition 7.2. The *sequence-form kernel* $K_{\mathcal{Q}}$ is the function $K_{\mathcal{Q}} : \mathbb{R}^{\Sigma} \times \mathbb{R}^{\Sigma} \rightarrow \mathbb{R}$,

$$K_{\mathcal{Q}}(\mathbf{x}, \mathbf{y}) := \langle \phi_{\mathcal{Q}}(\mathbf{x}), \phi_{\mathcal{Q}}(\mathbf{y}) \rangle = \sum_{\boldsymbol{\pi} \in \Pi} \prod_{k \in \boldsymbol{\pi}} \mathbf{x}[\sigma] \mathbf{y}[\sigma]. \quad (7.4)$$

^[7.a]More generally, the idea underlying KOMWU applies to on polyhedral decision sets whose vertices have 0/1 integer coordinates, and leads to an efficient implementation of the Vertex MWU algorithm for such a decision set as long as a specific *polyhedral kernel* function can be evaluated efficiently. The reader interested in the more general result is welcome to read the section assuming that $\mathcal{Q} \subseteq \mathbb{R}^d$ is an arbitrary polytope with (possibly exponentially many) 0/1 integral vertices $\Pi := \{\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_{|\Pi|}\} \subseteq \{0, 1\}^d$.

We remark that the sequence-form feature map and kernel are defined over the entirety of \mathbb{R}^Σ , and therefore take as input generic points $\mathbf{x}, \mathbf{y} \in \mathbb{R}^\Sigma$ which might or might not be valid sequence-form strategies.

7.3.2 Using the kernel to simulate the Vertex OMWU algorithm

We show that Vertex OMWU can be simulated using $d + 1$ evaluation of the kernel $K_{\mathcal{Q}}$ at every iteration. The key observation is summarized in the next theorem, which shows that the iterates $\lambda^{(t)}$ produced by Vertex OMWU are highly structured, in the sense that they are always proportional to the feature mapping $\phi_{\mathcal{Q}}(\mathbf{b}^{(t)})$ for some $\mathbf{b}^{(t)} \in \mathbb{R}^\Sigma$.

Theorem 7.1. Consider the Vertex OMWU algorithm (Algorithm 7.2). At all times $t \in \mathbb{N}_{\geq 0}$, the vector $\mathbf{b}^{(t)} \in \mathbb{R}^\Sigma$ defined as

$$\mathbf{b}^{(t)}[\sigma] := \exp\left\{-\sum_{\tau=1}^t \eta^{(\tau)} \mathbf{w}^{(\tau)}[\sigma]\right\} \quad (7.5)$$

for all $k = 1, \dots, d$, is such that

$$\lambda^{(t)} = \frac{\phi_{\mathcal{Q}}(\mathbf{b}^{(t)})}{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})}. \quad (7.6)$$

Proof. By induction.

- At time $t = 0$, the vector $\mathbf{b}^{(0)}$ is $\mathbf{b}^{(0)} = \mathbf{1} \in \mathbb{R}^\Sigma$. By definition of the feature map (7.3), $\phi_{\mathcal{Q}}(\mathbf{1}) = \mathbf{1} \in \mathbb{R}^\Pi$. So,

$$K_{\mathcal{Q}}(\mathbf{b}^{(0)}, \mathbf{1}) = \sum_{\pi \in \Pi} 1 = |\Pi|$$

and hence the right-hand side of (7.6) is $\frac{1}{|\Pi|} \mathbf{1}$, which matches $\lambda^{(0)}$ produced by Vertex OMWU, as we wanted to show.

- Assume the statement holds up to some time $t - 1 \geq 0$. We will show that it holds at time t as well. Since π has integral 0/1 coordinates, we can write

$$\begin{aligned} \exp\{-\eta^{(t)} \langle \mathbf{w}^{(t)}, \pi \rangle\} &= \exp\left\{-\eta^{(t)} \sum_{\sigma \in \pi} \mathbf{w}^{(t)}[\sigma]\right\} \\ &= \prod_{\sigma \in \pi} \exp\{-\eta^{(t)} \mathbf{w}^{(t)}[\sigma]\}. \end{aligned} \quad (7.7)$$

From the inductive hypothesis and (7.3), for all $\pi \in \Pi$,

$$\lambda^{(t-1)}[\pi] = \frac{\phi_{\mathcal{Q}}(\mathbf{b}^{(t-1)})[\pi]}{K_{\mathcal{Q}}(\mathbf{b}^{(t-1)}, \mathbf{1})} = \frac{\prod_{\sigma \in \pi} \mathbf{b}^{(t-1)}[\sigma]}{K_{\mathcal{Q}}(\mathbf{b}^{(t-1)}, \mathbf{1})}. \quad (7.8)$$

Plugging (7.7) and (7.8) into (7.1), we complete the inductive step

$$\lambda^{(t)}[\pi] = \frac{\prod_{\sigma \in \pi} \mathbf{b}^{(t-1)}[\sigma] \exp\{-\eta^{(t)} \mathbf{w}^{(t)}[\sigma]\}}{\sum_{\pi \in \Pi} \prod_{\sigma \in \pi} \mathbf{b}^{(t-1)}[\sigma] \exp\{-\eta^{(t)} \mathbf{w}^{(t)}[\sigma]\}} = \frac{\phi_{\mathcal{Q}}(\mathbf{b}^{(t)})[\pi]}{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})}$$

for all $\pi \in \Pi$, where in the last step we used the fact that

$$\mathbf{b}^{(t)}[\sigma] = \mathbf{b}^{(t-1)}[\sigma] \exp\{-\eta^{(t)} \mathbf{w}^{(t)}[\sigma]\}$$

by (7.5). □

The structure of $\lambda^{(t)}$ uncovered by [Theorem 7.1](#) can be leveraged to compute the iterate $\mathbf{x}^{(t)}$ produced by Vertex OMWU, *i.e.*, the convex combination of the vertices (7.2), using $d+1$ evaluations of the kernel $K_{\mathcal{Q}}$. We do so by extending an idea of Takimoto and Warmuth (2003, Equation (5.2)).

Theorem 7.2. Let $\mathbf{b}^{(t)}$ be as in [Theorem 7.1](#). For each $\sigma \in \Sigma := \{\sigma_1, \dots, \sigma_{|\Sigma|}\}$, let $\bar{\mathbf{e}}_{\sigma} \in \mathbb{R}^{\Sigma}$ be defined as the indicator vector

$$\bar{\mathbf{e}}_{\sigma}[\sigma'] := \mathbb{1}_{\sigma \neq \sigma'} := \begin{cases} 0 & \text{if } \sigma' = \sigma \\ 1 & \text{if } \sigma' \neq \sigma. \end{cases} \quad (7.9)$$

Then, at all $t \geq 1$, the iterate $\mathbf{x}^{(t)} \in \mathcal{Q}$ produced by Vertex OMWU can be written as

$$\mathbf{x}^{(t)} = \left(1 - \frac{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{\mathbf{e}}_{\sigma_1})}{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})}, \dots, 1 - \frac{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{\mathbf{e}}_{\sigma_{|\Sigma|}})}{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})} \right). \quad (7.10)$$

Proof. The proof crucially relies on the observation that for all $\sigma \in \Sigma$, the feature map $\phi_{\mathcal{Q}}(\bar{\mathbf{e}}_{\sigma})$ satisfies

$$\phi_{\mathcal{Q}}(\bar{\mathbf{e}}_{\sigma})[\pi] = \prod_{\sigma' \in \pi} \bar{\mathbf{e}}_{\sigma}[\sigma'] = \prod_{\sigma' \in \pi} \mathbb{1}_{\sigma' \neq \sigma} = \mathbb{1}_{\sigma \notin \pi}, \quad \forall \pi \in \Pi.$$

Using the fact that $\phi_{\mathcal{Q}}(\mathbf{1}) = \mathbf{1}$, we conclude that

$$\phi_{\mathcal{Q}}(\mathbf{1})[\pi] - \phi_{\mathcal{Q}}(\bar{\mathbf{e}}_{\sigma})[\pi] = \mathbb{1}_{\sigma \in \pi}, \quad \forall \sigma \in \Sigma. \quad (7.11)$$

Therefore, for all $\sigma \in \Sigma$, we obtain

$$\begin{aligned} \mathbf{x}^{(t)}[\sigma] &\stackrel{(7.2)}{=} \sum_{\pi \in \Pi} \lambda^{(t)}[\pi] \cdot \pi[\sigma] = \sum_{\pi \in \Pi} \lambda^{(t)}[\pi] \cdot \mathbb{1}_{\sigma \in \pi} = \sum_{\pi \in \Pi} \lambda^{(t)}[\pi] \cdot (\phi_{\mathcal{Q}}(\mathbf{1})[\pi] - \phi_{\mathcal{Q}}(\bar{\mathbf{e}}_{\sigma})[\pi]) \\ &= \frac{\langle \phi_{\mathcal{Q}}(\mathbf{b}^{(t)}), \phi_{\mathcal{Q}}(\mathbf{1}) \rangle - \langle \phi_{\mathcal{Q}}(\mathbf{b}^{(t)}), \phi_{\mathcal{Q}}(\bar{\mathbf{e}}_{\sigma}) \rangle}{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})} = \frac{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1}) - K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{\mathbf{e}}_{\sigma})}{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})} \\ &= 1 - \frac{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{\mathbf{e}}_{\sigma})}{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})}, \end{aligned}$$

where the second equality follows from the integrality of $\pi \in \Pi := \mathcal{Q} \cap \{0, 1\}^{\Sigma}$, the third from (7.11), the fourth from [Theorem 7.1](#), and the fifth from the definition of $K_{\mathcal{Q}}$ (7.4). \square

Combined, [Theorems 7.1](#) and [7.2](#) suggest that by keeping track of the vectors $\mathbf{b}^{(t)}$ instead of $\lambda^{(t)}$, updating them using [Theorem 7.1](#) and reconstructing the iterates $\mathbf{x}^{(t)}$ using [Theorem 7.2](#), Vertex OMWU can be simulated efficiently. We call the resulting algorithm, given in [Algorithm 7.2](#), *Kernelized OMWU (KOMWU)*. Similarly, we call *Kernelized MWU* the non-optimistic version of KOMWU obtained as the special case in which $\mathbf{m}^{(t)} = \mathbf{0}$ at all t . In light of the preceding discussion, we have the following.

Theorem 7.3. Kernelized OMWU produces the same iterates $\mathbf{x}^{(t)}$ as Vertex OMWU when it receives the same sequence of predictions $\mathbf{m}^{(t)}$ and losses $\mathbf{u}^{(t)} \in \mathbb{R}^{\Sigma}$. Furthermore, each iteration of KOMWU runs in time proportional to the time required to compute the $|\Sigma| + 1$ kernel evaluations $\{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})\} \cup \{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{\mathbf{e}}_{\sigma}) : \sigma \in \Sigma\}$.

7.4 Efficient evaluation of the sequence-form kernel

To complete our construction of KOMWU, we now show that the sequence-form kernel can be evaluated efficiently. The central result of this section, [Theorem 7.5](#), shows that KOMWU can be implemented with linear-time iterations in the number of sequences Σ .

7.4.1 Worst-case linear complexity for a single evaluation

We start by verifying that the sequence-form kernel can be evaluated in linear time for any pair of points $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{\Sigma}$. To do so, we introduce a *partial kernel function* $K_j : \mathbb{R}^{\Sigma} \times \mathbb{R}^{\Sigma} \rightarrow \mathbb{R}$ for every decision node $j \in \mathcal{J}$,

$$K_j : \mathbb{R}^{\Sigma} \times \mathbb{R}^{\Sigma} \rightarrow \mathbb{R}, \quad K_j(\mathbf{x}, \mathbf{y}) := \sum_{\pi \in \Pi_{> j}} \prod_{\sigma \in \pi} x[\sigma] y[\sigma]. \quad (7.12)$$

Algorithm 7.2: Kernelized OMWU (KOMWU)

Data: Learning rates $\eta^{(t)} > 0$

1 $\mathbf{u}^{(0)}, \mathbf{m}^{(0)}, \mathbf{s}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^\Sigma$ [▷ Initialization]

2 **function** NextStrategy($\mathbf{m}^{(t)} \in \mathbb{R}^\Sigma$) [▷ Set $\mathbf{m}^{(t)} = \mathbf{0}$ for non-predictive variant]

[▷ Compute $\mathbf{b}^{(t)}$ according to [Theorem 7.1](#)]

3 $\mathbf{w}^{(t)} \leftarrow \mathbf{u}^{(t-1)} - \mathbf{m}^{(t-1)} + \mathbf{m}^{(t)}$

4 $\mathbf{s}^{(t)} \leftarrow \mathbf{s}^{(t-1)} + \eta^{(t)} \mathbf{w}^{(t)}$ [▷ $\mathbf{s}^{(t)} = \sum \eta^{(\tau)} \mathbf{w}^{(\tau)}$]

5 **for** $\sigma \in \Sigma$ **do**

6 $\mathbf{b}^{(t)}[\sigma] \leftarrow \exp\{-\mathbf{s}^{(t)}[\sigma]\}$ [▷ See (7.5)]

[▷ Produce iterate $\mathbf{x}^{(t)}$ according to [Theorem 7.2](#)]

7 $\mathbf{x}^{(t)} \leftarrow \mathbf{0} \in \mathbb{R}^\Sigma$

8 $\alpha \leftarrow K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})$ [▷ $K_{\mathcal{Q}}$ is defined in (7.4)]

9 **for** $\sigma \in \Sigma$ **do**

10 $\mathbf{x}^{(t)}[\sigma] \leftarrow 1 - \frac{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{\mathbf{e}}_\sigma)}{\alpha}$ [▷ See (7.10)]

11 **return** $\mathbf{x}^{(t)}$

12 **function** ObserveUtility($\mathbf{u}^{(t)} \in \mathbb{R}^\Sigma$)

13 $t \leftarrow t + 1$

Theorem 7.4. For any vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^\Sigma$, the two following recursive relationships hold:

$$K_{\mathcal{Q}}(\mathbf{x}, \mathbf{y}) = \mathbf{x}[\emptyset] \mathbf{y}[\emptyset] \prod_{j \in \mathcal{C}_\emptyset} K_j(\mathbf{x}, \mathbf{y}), \quad (7.13)$$

and, for all decision nodes $j \in \mathcal{J}$,

$$K_j(\mathbf{x}, \mathbf{y}) = \sum_{a \in \mathcal{A}_j} \left(\mathbf{x}[ja] \mathbf{y}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{y}) \right). \quad (7.14)$$

In particular, [Equations \(7.13\) and \(7.14\)](#) give a recursive algorithm to evaluate the polyhedral kernel $K_{\mathcal{Q}}$ associated with the sequence-form strategy space of any player i in an imperfect-information extensive-form game in linear time in the number of sequences $|\Sigma|$.

7.4.2 Batched computation and amortized complexity

[Theorem 7.4](#) shows that the kernel $K_{\mathcal{Q}}$ can be evaluated in linear time (in the number of sequences Σ) at any pair of points $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^\Sigma \times \mathbb{R}^\Sigma$. So, the KOMWU algorithm ([Algorithm 7.2](#)) can be trivially

implemented in quadratic $\mathcal{O}(|\Sigma|^2)$ time per iteration by directly evaluating the $|\Sigma| + 1$ kernel evaluations $\{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})\} \cup \{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{e}_{\sigma}) : \sigma \in \Sigma\}$ needed at each iteration, where $\bar{e}_{\sigma} \in \mathbb{R}^{\Sigma}$, defined in (7.9) for the general case, is the vector whose components are $\bar{e}_{\sigma}[\sigma'] := \mathbb{1}_{\sigma \neq \sigma'}$ for all $\sigma, \sigma' \in \Sigma$.

We will now refine that result by showing an implementation of KOMWU with *linear*-time (i.e., $\mathcal{O}(|\Sigma|)$) per-iteration complexity, by exploiting the structure of the particular set of kernel evaluations needed at every iteration. In particular, we rely on the following observation.

Proposition 7.1. For any player $i \in \llbracket m \rrbracket$, vector $\mathbf{x} \in \mathbb{R}_{>0}^{\Sigma}$, and sequence $ja \in \Sigma^*$,

$$\frac{1 - K_{\mathcal{Q}}(\mathbf{x}, \bar{e}_{ja})/K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1})}{1 - K_{\mathcal{Q}}(\mathbf{x}, \bar{e}_{p_j})/K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1})} = \frac{\mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{1})}{K_j(\mathbf{x}, \mathbf{1})}.$$

We defer the proof of [Proposition 7.1](#) as an appendix at the end of the chapter, and first discuss how the result informs an efficient batched computation of the kernel evaluations needed in KOMWU. Specifically, in light of [Proposition 7.1](#), we do the following to compute $\{K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{e}_{\sigma}) : \sigma \in \Sigma\}$ in cumulative $\mathcal{O}(|\Sigma|)$ time.

1. First, we compute the values $K_j(\mathbf{b}^{(t)}, \mathbf{1})$ for all $j \in \mathcal{J}$ in cumulative $\mathcal{O}(|\Sigma|)$ time by using (7.14).
2. Then, we compute the ratio $K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{e}_{\emptyset})/K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})$ by evaluating the two kernel separately using [Theorem 7.4](#), spending $\mathcal{O}(|\Sigma|)$ time.
3. At this point, we repeatedly use [Proposition 7.1](#) in a top-down fashion along the tree-form decision process of player i to compute the ratio $K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{e}_{ja})/K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})$ for each sequence $ja \in \Sigma^*$ given the value of the parent ratio $K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{e}_{p_j})/K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})$ and the partial kernel evaluations $\{K_j(\mathbf{b}^{(t)}, \mathbf{1}) : j \in \mathcal{J}\}$ from Step 1. For each $ja \in \Sigma^*$, [Proposition 7.1](#) gives a formula whose runtime is linear in the number of children decision nodes $|\mathcal{C}_{ja}|$ at that sequence. Therefore, the cumulative runtime required to compute all ratios $K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{e}_{ja})/K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})$ is $\mathcal{O}(|\Sigma|)$.
4. Finally, by multiplying the ratios computed in Step 3 by the value of $K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \mathbf{1})$ computed in Step 2, we can easily recover each $K_{\mathcal{Q}}(\mathbf{b}^{(t)}, \bar{e}_{\sigma})$ for every $\sigma \in \Sigma^*$.

Hence, we have just proved the following result.

Theorem 7.5. For each player i in a perfect-recall extensive-form game, the Kernelized OMWU algorithm can be implemented exactly, with a per-iteration complexity linear in the number of sequences $|\Sigma|$ of that player.

7.5 Regret bound of Kernelized OMWU

If the players in an imperfect-information extensive-form game run KOMWU, then we can combine [Theorem 7.3](#) with standard OMWU regret bounds given in [Section 3.3.1](#) to immediately get the following.

Theorem 7.6. In an imperfect-information extensive-form game, after any T iterations, KOMWU satisfies

1. A player i using KOMWU with $\eta^{(t)} := \eta = \sqrt{8 \log(A) \|\mathcal{Q}\|_1} / \sqrt{T}$ is guaranteed to incur regret at most $\text{Reg}^{(T)} = \mathcal{O}(\sqrt{\|\mathcal{Q}\|_1 \log(A) T})$.
2. For all T , if $\eta^{(t)} := \eta = \frac{T^{-1/4}}{\sqrt{m-1}}$, the regret of each player $i \in \llbracket m \rrbracket$ is bounded as $\text{Reg}^{(T)} \leq (4 + \|\mathcal{Q}\| \log A) \sqrt{m-1} \cdot T^{1/4}$.
3. There exist $C, C' > 0$ such that if all m players learn using KOMWU with constant learning rate $\eta^{(t)} := \eta \leq 1/(Cm \log^4 T)$, then each player is guaranteed to incur regret at most $\frac{\log(A) \|\mathcal{Q}\|_1}{\eta} + C' \log T$.
4. If all m player learn using KOMWU with $\eta^{(t)} := \eta \leq 1/\sqrt{8}(m-1)$, then the sum of regrets is at most $\sum_{i=1}^m \text{Reg}^{(T)} = \mathcal{O}(\max_{i=1}^m \{\|\mathcal{Q}\|_1 \log A_i\} \frac{m}{\eta})$.
5. For two-player zero-sum imperfect-information extensive-form games, if both players learn using KOMWU, then there exists a schedule of learning-rates $\eta^{(t)}$ such that the iterates converge to a Nash equilibrium. Furthermore, if the NFG representation of the imperfect-information extensive-form game has a unique Nash equilibrium and both players use learning rates $\eta^{(t)} = \eta \leq 1/8$, then the iterates converge to a Nash equilibrium at a linear rate $C(1 + C')^{-t}$, where C, C' are constants that depend on the game.

Prior to our result, the strongest regret bound for methods that take linear time per iteration was based on instantiating, *e.g.*, follow the regularized leader (FTRL) or its optimistic variant with the dilatable global entropy regularizer discussed in [Chapter 5 \(Section 5.4.2\)](#). For FTRL this yields a regret bound of the form $\mathcal{O}(\sqrt{\log(A) \|\mathcal{Q}\|_1^2 T})$. For optimistic FTRL this yields a regret bound of the form $\mathcal{O}(\log(A) \|\mathcal{Q}\|_1^2 \sqrt{m} T^{1/4})$, when every player in an m -player game uses that algorithm and appropriate learning rates.

Our algorithm improves the state-of-the-art rate in two ways. First, we improve the dependence on game constants by almost a square root factor, because our dependence on $\|\mathcal{Q}\|_1$ is smaller by a square root, compared to prior results. Secondly, in the multiplayer general-sum setting, every other method achieves regret that is on the order of $T^{1/4}$, whereas our method achieves

regret on the order of $\log^4(T)$. In the context of two-player zero-sum imperfect-information extensive-form games, the bound on the sum of regrets in [Theorem 7.6](#) guarantees convergence to a Nash equilibrium at a rate of $\mathcal{O}(\max_i \|\mathcal{Q}\|_1 \log A_i/T)$. This similarly improves the prior state of the art.

C.-W. Lee, Kroer, and Luo (2021) showed the first last-iterate results for imperfect-information extensive-form games using algorithms that require linear time per iteration. In particular, they show that the dilated entropy DGF combined with optimistic online mirror descent leads to last-iterate convergence at a linear rate. However, their result requires learning rates $\eta \leq 1/(8|\Sigma|)$. This learning rate is impractically small in practice. In contrast, our last-iterate linear-rate result for KOMWU allows learning rates of size $1/8$. That said, our result is not directly comparable to theirs. The existence of a unique Nash equilibrium in the imperfect-information extensive-form game representation is a necessary condition for uniqueness in the NFG representation. However, it is possible that the NFG has additional equilibria even when the imperfect-information extensive-form game does not. C. Wei, C. Lee, M. Zhang, and Luo (2021) conjecture that linear-rate convergence holds even without the assumption of a unique Nash equilibrium. If this conjecture turns out to be true for NFGs, then [Theorem 7.3](#) would immediately imply that KOMWU also has last-iterate linear-rate convergence without the uniqueness assumption.

7.6 Experimental evaluation

We complement the theoretical results provided in this chapter with a numerical investigation of KOMWU in Kuhn and Leduc poker (Kuhn, 1950; Southey, Bowling, Larson, Piccione, Burch, Billings, and Rayner, 2005), standard benchmark games from the extensive-form games literature. As usual, a full description of the game instances we use is available in [Appendix A](#).

Specifically, we investigate whether the strong theoretical guarantees enjoyed by KOMWU, which crucially rely on predictivity, are shared by non-predictive algorithm such as (vanilla) CFR and CFR instantiated with RM^+ at all decision points. Additionally, we investigate how KOMWU compares with DOMWU (C.-W. Lee, Kroer, and Luo, 2021).

Results are shown in [Figure 7.2](#) and [Figure 7.3](#).

For algorithms that require learning rates (KOMWU and DOMWU), we consider the four different choices of constants $\eta^{(t)} := \eta \in \{0.1, 1, 5, 10\}$. We remark that the payoff ranges of these games are not $[0, 1]$ (*i.e.*, the games have not been normalized). The payoff range of Kuhn poker is 6 for the 3-player variant and 8 for the 4-player variant. The payoff range of Leduc poker is 21 for the 3-player variant and 28 for the 4-player variant. So, a learning rate value of $\eta = 0.1$ corresponds to a significantly smaller learning rate in the normalized game where the payoffs have been shifted and rescaled to lie within $[0, 1]$.

In all games, we observe that the maximum per-player regret cumulated by KOMWU plateaus and remains constants, unlike the CFR variants. This behavior is consistent with the near-optimal

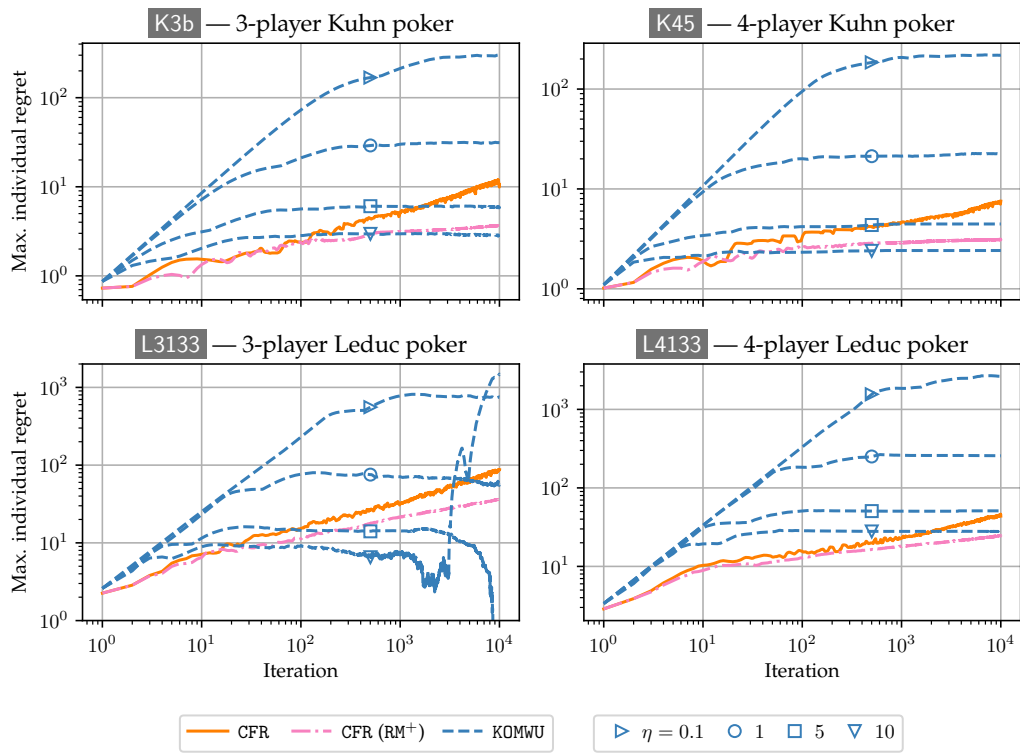


Figure 7.2: Experimental comparison of KOMWU with CFR.

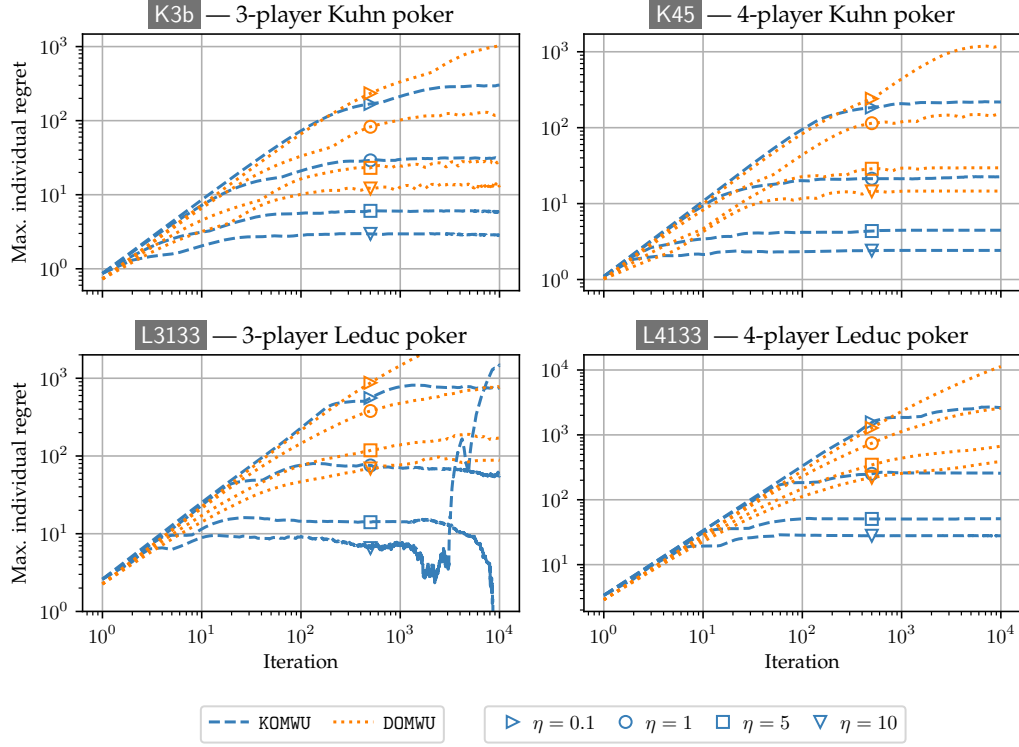


Figure 7.3: Experimental comparison of KOMWU with DOMWU for different choices of learning rates.

per-player regret guarantees of KOMWU (Theorem 7.6). In the 3-player variant of Leduc poker, we observe that the largest learning rate we use, $\eta = 10$, leads to divergent behavior of the learning dynamics.

7.A Appendix: Proof details

7.A.1 Proof of Theorem 7.4

Proof. In the proof of this result, we will make use of the following additional notation. Given any $\mathbf{x} \in \mathbb{R}^\Sigma$ and a $j \in \mathcal{J}$, we let $\mathbf{x}_{(j)} \in \mathbb{R}^{\Sigma_{\succ j}}$ denote the subvector obtained from \mathbf{x} by only considering sequences $\sigma \in \Sigma_{\succ j}$, that is, the vector whose entries are defined as $\mathbf{x}_{(j)}[\sigma] = \mathbf{x}[\sigma]$ for all $\sigma \in \Sigma_{\succ j}$.

- Proof of (7.13). Direct inspection of the definitions of Π and $\Pi_{\succ j}$ (given in Section 2.2.2), together with the observation that the $\{\Sigma_{\succ j} : j \in \mathcal{C}_\emptyset\}$ form a partition of Σ^* , reveals that

$$\Pi = \left\{ \boldsymbol{\pi} \in \{0, 1\}^\Sigma : \begin{array}{l} \textcircled{1} \boldsymbol{\pi}[\emptyset] = 1 \\ \textcircled{2} \boldsymbol{\pi}_{(j)} \in \Pi_{\succ j} \quad \forall j \in \mathcal{C}_\emptyset \end{array} \right\} \quad (7.15)$$

The observation above can be summarized informally into the statement that “ Π is equal, up to permutation of indices, to the Cartesian product $\times_{j \in \mathcal{C}_\emptyset} \Pi_{\succ j}$ ”. The idea for the proof is then to use that Cartesian product structure in the definition of 0/1-polyhedral kernel (7.4), as follows

$$\begin{aligned} K_{\mathcal{Q}}(\mathbf{x}, \mathbf{y}) &= \sum_{\boldsymbol{\pi} \in \Pi} \prod_{\sigma \in \boldsymbol{\pi}} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \\ &= \sum_{\boldsymbol{\pi} \in \Pi} \left(\mathbf{x}[\emptyset] \mathbf{y}[\emptyset] \prod_{j' \in \mathcal{C}_\emptyset} \prod_{\sigma \in \boldsymbol{\pi}_{(j')}} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \right) \\ &= \sum_{\boldsymbol{\pi}_{(j)} \in \Pi_{\succ j} \quad \forall j \in \mathcal{C}_\emptyset} \left(\mathbf{x}[\emptyset] \mathbf{y}[\emptyset] \prod_{j' \in \mathcal{C}_\emptyset} \prod_{\sigma \in \boldsymbol{\pi}_{(j')}} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \right) \\ &= \mathbf{x}[\emptyset] \mathbf{y}[\emptyset] \sum_{\boldsymbol{\pi}_{(j)} \in \Pi_{\succ j} \quad \forall j \in \mathcal{C}_\emptyset} \left(\prod_{j' \in \mathcal{C}_\emptyset} \prod_{\sigma \in \boldsymbol{\pi}_{(j')}} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \right) \\ &= \mathbf{x}[\emptyset] \mathbf{y}[\emptyset] \prod_{j \in \mathcal{C}_\emptyset} \sum_{\boldsymbol{\pi}_{(j)} \in \Pi_{\succ j}} \prod_{\sigma \in \boldsymbol{\pi}_{(j)}} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \\ &= \mathbf{x}[\emptyset] \mathbf{y}[\emptyset] \prod_{j \in \mathcal{C}_\emptyset} K_j(\mathbf{x}, \mathbf{y}), \end{aligned}$$

where the second equality follows from the fact that $\{\emptyset\} \cup \{\Sigma_{\succ j} : j \in \mathcal{C}_\emptyset\}$ form a partition of Σ , the third equality follows from (7.15), the fifth equality from the fact that each $\boldsymbol{\pi}_{(j)} \in \Pi_{\succ j}$ can be chosen independently, and the last equality from the definition of partial kernel function (7.12).

- Proof of (7.14). Similarly to what we did for (7.13), we start by giving an inductive characterization of $\Pi_{\succ j}$ as a function of the children $\Pi_{\succ j'}$ for $j' \in \cup_{a \in \mathcal{A}_j} \mathcal{C}_{ja}$. Specifically, direct inspection of the definitions of $\Pi_{\succ j}$, together with the observation that the $\{\Sigma_{\succ j'} : j' \in \cup_{a \in \mathcal{A}_j} \mathcal{C}_{ja}\}$ form a partition of $\Sigma_{\succ j}$, reveals that

$$\Pi_{\succ j} = \left\{ \boldsymbol{\pi} \in \{0, 1\}^{\Sigma_{\succ j}} : \begin{array}{l} \textcircled{1} \sum_{a \in \mathcal{A}_j} \boldsymbol{\pi}[ja] = 1 \\ \textcircled{2} \boldsymbol{\pi}_{(j')} \in \boldsymbol{\pi}[ja] \cdot \Pi_{\succ j'} \quad \forall a \in \mathcal{A}_j, j' \in \mathcal{C}_{ja} \end{array} \right\}. \quad (7.16)$$

From constraint ① together with the fact that $\pi[ja] \in \{0, 1\}$ for all $a \in \mathcal{A}_j$, we conclude that exactly one $a^* \in \mathcal{A}_j$ is such that $\pi[ja^*] = 1$, while $\pi[ja] = 0$ for all other $a \in \mathcal{A}_j$, $a \neq a^*$. So, we can rewrite (7.16) as

$$\Pi_{\succcurlyeq j} = \bigcup_{a^* \in \mathcal{A}_j} \left\{ \pi \in \{0, 1\}^{\Sigma_{\succcurlyeq j}} : \begin{array}{ll} \text{① } \pi[ja^*] = 1 & \\ \text{② } \pi[ja] = 0 & \forall a \in \mathcal{A}_j, a \neq a^* \\ \text{③ } \pi_{(j')} \in \Pi_{\succcurlyeq j'} & \forall j' \in \mathcal{C}_{ja^*} \\ \text{④ } \pi_{(j')} = \mathbf{0} & \forall j' \in \bigcup_{a \in \mathcal{A}_j, a \neq a^*} \mathcal{C}_{ja} \end{array} \right\}, \quad (7.17)$$

where the union is clearly disjoint. The above equality can be summarized informally into the statement that “ $\Pi_{\succcurlyeq j}$ is equal, up to permutation of indices, to a disjoint union over actions $a^* \in \mathcal{A}_j$ of Cartesian products $\times_{j' \in \mathcal{C}_{ja^*}} \Pi_{\succcurlyeq j'}$ ”. We can then use the same set of manipulations we already used in the proof of (7.13) to obtain

$$\begin{aligned} K_j(\mathbf{x}, \mathbf{y}) &= \sum_{\pi \in \Pi_{\succcurlyeq j}} \prod_{\sigma \in \pi} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \\ &= \sum_{\pi \in \Pi_{\succcurlyeq j}} \left(\mathbf{x}[ja^*] \mathbf{y}[ja^*] \prod_{j' \in \mathcal{C}_{ja^*}} \prod_{\sigma \in \pi_{(j')}} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \right) \\ &= \sum_{a^* \in \mathcal{A}_j} \sum_{\substack{\pi_{j'} \in \Pi_{\succcurlyeq j'} \\ \forall j' \in \mathcal{C}_{ja^*}}} \left(\mathbf{x}[ja^*] \mathbf{y}[ja^*] \prod_{j' \in \mathcal{C}_{ja^*}} \prod_{\sigma \in \pi_{(j')}} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \right) \\ &= \sum_{a^* \in \mathcal{A}_j} \left(\mathbf{x}[ja^*] \mathbf{y}[ja^*] \prod_{j' \in \mathcal{C}_{ja^*}} \sum_{\pi_{(j')} \in \Pi_{\succcurlyeq j'}} \prod_{\sigma \in \pi_{(j')}} \mathbf{x}[\sigma] \mathbf{y}[\sigma] \right) \\ &= \sum_{a \in \mathcal{A}_j} \left(\mathbf{x}[ja] \mathbf{y}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{y}) \right), \end{aligned}$$

where the second equality follows from the fact that the $\{\Sigma_{\succcurlyeq j'} : j' \in \bigcup_{a \in \mathcal{A}_j} \mathcal{C}_{ja}\}$ form a partition of $\Sigma_{\succcurlyeq j}$, third equality follows from (7.17), the fourth equality from the fact that each $\pi_{j'} \in \Pi_{\succcurlyeq j'}$ can be picked independently, and the last equality from the definition of partial kernel function (7.12) as well as renaming a^* into a . \square

7.A.2 Proof of Proposition 7.1

We give a self-contained proof of the result stated in Proposition 7.1, repeated below for convenience.

Proposition 7.1 (Restated). For any player $i \in \llbracket m \rrbracket$, vector $\mathbf{x} \in \mathbb{R}_{>0}^\Sigma$, and sequence $ja \in \Sigma^*$,

$$\frac{1 - K_{\mathcal{Q}}(\mathbf{x}, \bar{\mathbf{e}}_{ja})/K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1})}{1 - K_{\mathcal{Q}}(\mathbf{x}, \bar{\mathbf{e}}_{p_j})/K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1})} = \frac{\mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{1})}{K_j(\mathbf{x}, \mathbf{1})}.$$

Proof. Note that since $\mathbf{x} > \mathbf{0}$, clearly $K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1}), K_j(\mathbf{x}, \mathbf{1}) > 0$. Furthermore, from (7.11) we have that for all $\sigma \in \Sigma$

$$K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1}) - K_{\mathcal{Q}}(\mathbf{x}, \bar{\mathbf{e}}_\sigma) = \langle \phi_{\mathcal{Q}}(\mathbf{1}) - \phi_{\mathcal{Q}}(\bar{\mathbf{e}}_\sigma), \phi_{\mathcal{Q}}(\mathbf{x}) \rangle = \sum_{\substack{\pi \in \Pi \\ \pi[\sigma]=1}} \prod_{\sigma' \in \pi} \mathbf{x}[\sigma'] > 0. \quad (7.18)$$

The above inequality immediately implies that $0 < K_{\mathcal{Q}}(\mathbf{x}, \bar{\mathbf{e}}_{p_j})/K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1}) < 1$ and therefore all denominators in the statement are nonzero, making the statement well-formed.

In light of (7.18), we further have

$$\begin{aligned} \frac{1 - K_{\mathcal{Q}}(\mathbf{x}, \bar{\mathbf{e}}_{ja})/K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1})}{1 - K_{\mathcal{Q}}(\mathbf{x}, \bar{\mathbf{e}}_{p_j})/K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1})} &= \frac{\mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{1})}{K_j(\mathbf{x}, \mathbf{1})} \\ \iff \frac{K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1}) - K_{\mathcal{Q}}(\mathbf{x}, \bar{\mathbf{e}}_{ja})}{K_{\mathcal{Q}}(\mathbf{x}, \mathbf{1}) - K_{\mathcal{Q}}(\mathbf{x}, \bar{\mathbf{e}}_{p_j})} &= \frac{\mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{1})}{K_j(\mathbf{x}, \mathbf{1})} \\ \iff \frac{\sum_{\pi \in \Pi, \pi[ja]=1} \prod_{\sigma \in \pi} \mathbf{x}[\sigma]}{\sum_{\pi \in \Pi, \pi[p_j]=1} \prod_{\sigma \in \pi} \mathbf{x}[\sigma]} &= \frac{\mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{1})}{K_j(\mathbf{x}, \mathbf{1})}. \end{aligned} \quad (7.19)$$

We now prove (7.19). Let

$$\mathcal{A} := \{\pi \in \Pi : \pi[ja] = 1\}, \quad \mathcal{B} := \{\pi \in \Pi : \pi[p_j] = 1\}$$

be the domains of the summations. From the definition of Π (specifically, constraints ② in the definition of \mathcal{Q} , of which Π is a subset by definition; see Definition 2.5), it is clear that $\mathcal{A} \subseteq \mathcal{B}$. Furthermore, it is straightforward to check, using the definitions of Π_{\succ_j} , Π , and \mathcal{B} , that

$$\pi_{(j)} \in \Pi_{\succ_j} \quad \forall \pi \in \mathcal{B} \quad (7.20)$$

We now introduce the function $((\cdot \parallel \cdot)) : \mathcal{B} \times \Pi_{\succ_j} \rightarrow \mathcal{B}$ defined as follows. Given any $\pi \in \mathcal{B}$ and $\pi' \in \Pi_{\succ_j}$, $((\pi \parallel \pi'))$ is the vector obtained from π by replacing all sequences at or below decision node j with what is prescribed by π' ; formally,

$$((\pi \parallel \pi'))[\sigma] := \begin{cases} \pi'[\sigma] & \text{if } \sigma \in \Sigma_{\neq j} \\ \pi[\sigma] & \text{otherwise.} \end{cases} \quad \forall \pi \in \mathcal{B}, \pi' \in \Pi_{\neq j} \quad (7.21)$$

It is immediate to check that $((\pi \parallel \pi'))$ is indeed an element of \mathcal{B} . We now introduce the following result.

Lemma 7.2. There exists a set $\mathcal{P} \subseteq \mathcal{B}$ such that every $\pi'' \in \mathcal{B}$ can be uniquely written as $\pi'' = ((\pi \parallel \pi'))$ for some $\pi \in \mathcal{P}$ and $\pi' \in \Pi_{\neq j}$. Vice versa, given any $\pi \in \mathcal{P}$ and $\pi' \in \Pi_{\neq j}$, then $((\pi \parallel \pi')) \in \mathcal{B}$.

Proof. The second part of the statement is straightforward. We now prove the first part.

Fix any $\pi^* \in \Pi_{\neq j}$ and let $\mathcal{P} := \{((\pi \parallel \pi^*)) : \pi \in \mathcal{B}\}$. It is straightforward to verify that for any $\pi'' \in \mathcal{B}$, the choices $\pi := ((\pi'' \parallel \pi^*)) \in \mathcal{P}$ and $\pi' := \pi_{(j)} \in \Pi_{\neq j}$ satisfy the equality $((\pi \parallel \pi')) = \pi''$. So, every $\pi'' \in \mathcal{B}$ can be expressed in *at least one way* as $\pi'' = ((\pi \parallel \pi'))$ for some $\pi \in \mathcal{P}$ and $\pi' \in \Pi_{\neq j}$. We now show that the choice above is in fact the unique choice. First, it is clear from the definition of $((\cdot \parallel \cdot))$ that π' must satisfy $\pi' = \pi''_{(j)}$, and so it is uniquely determined. Suppose now that there exist $\pi, \tilde{\pi} \in \mathcal{P}$ such that $((\pi \parallel \pi')) = ((\tilde{\pi} \parallel \pi'))$. Then, π and $\tilde{\pi}$ must coincide on all $\sigma \in \Sigma \setminus \Sigma_{\neq j}$. However, since all elements of \mathcal{P} are of the form $((b \parallel \pi^*))$ for some $b \in \mathcal{B}$, then π and $\tilde{\pi}$ must also coincide on all $\sigma \in \Sigma_{\neq j}$. So, π and $\tilde{\pi}$ coincide on all coordinates $\sigma \in \Sigma$, and the statement follows. \square

Lemma 7.2 exposes a convenient combinatorial structure of the set \mathcal{B} . In particular, it enables us to rewrite the denominator on the left-hand side of (7.19) as follows

$$\begin{aligned} \sum_{\pi \in \mathcal{B}} \prod_{\sigma \in \pi} x[\sigma] &= \sum_{\pi' \in \mathcal{P}} \sum_{\pi'' \in \Pi_{\neq j}} \prod_{\sigma \in ((\pi' \parallel \pi''))} x[\sigma] \\ &= \sum_{\pi' \in \mathcal{P}} \sum_{\pi'' \in \Pi_{\neq j}} \left(\prod_{\substack{\sigma \in ((\pi' \parallel \pi'')) \\ \sigma \in \Sigma_{\neq j}}} x[\sigma] \right) \left(\prod_{\substack{\sigma \in ((\pi' \parallel \pi'')) \\ \sigma \notin \Sigma_{\neq j}}} x[\sigma] \right) \\ &= \sum_{\pi' \in \mathcal{P}} \sum_{\pi'' \in \Pi_{\neq j}} \left(\prod_{\sigma \in \pi''} x[\sigma] \right) \left(\prod_{\substack{\sigma \in \pi' \\ \sigma \notin \Sigma_{\neq j}}} x[\sigma] \right) \\ &= \left(\sum_{\pi'' \in \Pi_{\neq j}} \prod_{\sigma \in \pi''} x[\sigma] \right) \left(\sum_{\pi' \in \mathcal{P}} \prod_{\substack{\sigma \in \pi' \\ \sigma \notin \Sigma_{\neq j}}} x[\sigma] \right) \end{aligned}$$

$$= K_j(\mathbf{x}, \mathbf{1}) \cdot \left(\sum_{\pi' \in \mathcal{P}} \prod_{\substack{\sigma \in \pi' \\ \sigma \notin \Sigma_{\neq j}}} \mathbf{x}[\sigma] \right), \quad (7.22)$$

where we used (7.21) in the third equality.

We can use a similar technique to express the numerator of the left-hand side of (7.19). Let

$$\Pi_{i,ja} := \{\pi \in \Pi_{\neq j} : \pi[ja] = 1\}.$$

Using the constraints that define Π and the definition of \mathcal{A} , it follows immediately that for any $\pi \in \mathcal{A}$, $\pi_{(j)} \in \Pi_{i,ja}$. Furthermore, a direct consequence of Lemma 7.2 is the following:

Corollary 7.1. The same set $\mathcal{P} \subseteq \mathcal{B}$ introduced in Lemma 7.2 is such that every $\pi'' \in \mathcal{A}$ can be uniquely written as $\pi'' = ((\pi \parallel \pi'))$ for some $\pi \in \mathcal{P}$ and $\pi' \in \Pi_{i,ja}$.

Using Corollary 7.1 and following the same steps that led to (7.22), we express the numerator of the left-hand side of (7.19) as

$$\begin{aligned} \sum_{\pi \in \mathcal{A}} \prod_{\sigma \in \pi} \mathbf{x}[\sigma] &= \sum_{\pi' \in \mathcal{P}} \sum_{\pi'' \in \Pi_{i,ja}} \prod_{\sigma \in ((\pi' \parallel \pi''))} \mathbf{x}[\sigma] \\ &= \sum_{\pi' \in \mathcal{P}} \sum_{\pi'' \in \Pi_{i,ja}} \left(\prod_{\substack{\sigma \in ((\pi' \parallel \pi'')) \\ \sigma \in \Sigma_{\neq j}}} \mathbf{x}[\sigma] \right) \left(\prod_{\substack{\sigma \in ((\pi' \parallel \pi'')) \\ \sigma \notin \Sigma_{\neq j}}} \mathbf{x}[\sigma] \right) \\ &= \sum_{\pi' \in \mathcal{P}} \sum_{\pi'' \in \Pi_{i,ja}} \left(\prod_{\sigma \in \pi''} \mathbf{x}[\sigma] \right) \left(\prod_{\substack{\sigma \in \pi' \\ \sigma \notin \Sigma_{\neq j}}} \mathbf{x}[\sigma] \right) \\ &= \left(\sum_{\pi'' \in \Pi_{i,ja}} \prod_{\sigma \in \pi''} \mathbf{x}[\sigma] \right) \left(\sum_{\substack{\pi' \in \mathcal{P} \\ \sigma \in \pi' \\ \sigma \notin \Sigma_{\neq j}}} \prod \mathbf{x}[\sigma] \right). \end{aligned} \quad (7.23)$$

The statement then follows immediately if we can prove that

$$\sum_{\pi \in \Pi_{i,ja}} \prod_{\sigma \in \pi} \mathbf{x}[\sigma] = \mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{1}).$$

To do so, we use the same approach as in the proof of Theorem 7.4. In fact, we can directly use the inductive characterization of $\Pi_{\neq j}$ obtained in (7.17) to write

$$\Pi_{i,ja} = \left\{ \pi \in \{0, 1\}^{\Sigma_{\succ j}} : \begin{array}{ll} \textcircled{1} \pi[ja] = 1 & \\ \textcircled{2} \pi[ja'] = 0 & \forall a' \in \mathcal{A}_j, a' \neq a \\ \textcircled{3} \pi_{(j')} \in \Pi_{\succ j'} & \forall j' \in \mathcal{C}_{ja} \\ \textcircled{4} \pi_{(j')} = \mathbf{0} & \forall j' \in \cup_{a' \in \mathcal{A}_j, a' \neq a} \mathcal{C}_{ja'} \end{array} \right\},$$

which fundamentally uncovers the *Cartesian-product structure* of $\Pi_{i,ja}$. Using the same technique as [Theorem 7.4](#), we then have

$$\begin{aligned} \sum_{\pi \in \Pi_{i,ja}} \prod_{\sigma \in \pi} \mathbf{x}[\sigma] &= \sum_{\pi_{(j')} \in \Pi_{\succ j'} \forall j' \in \mathcal{C}_{ja}} \left(\mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} \prod_{\sigma \in \pi_{(j')}} \mathbf{x}[\sigma] \right) \\ &= \left(\mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} \sum_{\pi_{(j')} \in \Pi_{\succ j'}} \prod_{\sigma \in \pi_{(j')}} \mathbf{x}[\sigma] \right) \\ &= \left(\mathbf{x}[ja] \prod_{j' \in \mathcal{C}_{ja}} K_{j'}(\mathbf{x}, \mathbf{1}) \right), \end{aligned}$$

and the statement is proven. □

Part III

Computation of extensive-form correlated and team equilibria

Chapter 8

Uncoupled learning of extensive-form correlated equilibrium

In [Chapter 4](#) we have seen how one can construct simple, uncoupled no-*external*-regret dynamics for sequence-form strategy polytopes in any imperfect-information extensive-form game. Due to the connection between external regret and coarse correlated equilibria discussed in [Chapter 3](#), those dynamics can be used to compute normal-form coarse-correlated equilibria in general multiplayer games, as well as Nash equilibria in two-player zero-sum games.

However, a different notion of equilibrium, extensive-form correlated equilibrium (the counterpart of correlated equilibrium in normal-form games), cannot be obtained through no-external-regret dynamics, begging the following question.

Do uncoupled, polynomial-time learning dynamics leading to the set of extensive-form correlated equilibrium exist in any imperfect-information extensive-form game?

In this chapter, we settle it for the positive.

8.1 Contributions and related work

In this chapter we show that it is possible to construct simple, uncoupled learning dynamics that minimize a stronger notion of regret—that is, that guarantee a stronger notion of hindsight rationality—than external regret. In particular, we will define and study an instance of Φ -regret based on the notion of *trigger deviation functions*. As we will show, the minimization of Φ -regret

with respect to the set of trigger deviation functions leads to extensive-form correlated equilibrium (EFCE), the imperfect-information extensive-form game counterpart of what correlated equilibrium is for single-decision simultaneous-action games (von Stengel and Forges, 2008).

Related work The notion of trigger deviation functions which we introduce in this chapter builds on prior work on *extensive-form transformations* by G. J. Gordon, A. Greenwald, and Marks (2008). However, trigger deviation functions are simpler than the extensive-form transformations. Specifically, extensive-form transformations allow one to specify more than one trigger sequence (together with different continuation strategies, one for each specified trigger sequence), whereas our notion of trigger deviation functions only contemplates a single trigger sequence. Consequently, the set of all trigger deviation functions is significantly smaller, and simpler, than the set of all extensive-form transformations. The simpler structural properties of the set of trigger deviation functions, explored in Section 8.3.2, will enable us to construct an efficient no-regret algorithm for the convex hull of the set of all canonical trigger deviation matrices.

We also mention relevant literature concurrent and subsequent to the conference version of the paper on which this chapter is based (Celli, Marchesi, Farina, and Gatti, 2020). First, we acknowledge the thesis work by H. Zhang (2022) on computing certain refinements of EFCE via polynomial-time uncoupled learning dynamics, though their procedure could require up to exponential memory. In a later chapter of the thesis, the author notes that some of the dynamics introduced in the thesis can be modified to guarantee polynomial memory usage when convergence to the set of (unrefined) EFCE is sought. That work was conducted independently and concurrently with ours. In a recent paper, Morrill, D’Orazio, Sarfati, Lanctot, J. Wright, A. Greenwald, and Bowling (2020) conduct a study of different forms of correlation in extensive-form games, defining a taxonomy of solution concepts. Each of their solution concepts is attained by a particular set of no-regret learning dynamics, which is obtained by instantiating the phi-regret minimization framework (A. Greenwald and Jafari, 2003; Stoltz and Lugosi, 2007; G. J. Gordon, A. Greenwald, and Marks, 2008) with a suitably-defined deviation function. As part of their analysis, Morrill, D’Orazio, Sarfati, Lanctot, J. Wright, A. Greenwald, and Bowling (2020) investigate some properties of the well-established CFR regret minimization algorithm (Zinkevich, Bowling, Johanson, and Piccione, 2007) applied to n -player general-sum extensive-form games, establishing that it is hindsight-rational with respect to a specific set of deviation functions, which the authors coin *blind counterfactual deviations*. In subsequent recent work, Morrill, D’Orazio, Lanctot, J. R. Wright, Bowling, and A. R. Greenwald (2021) extend their prior work (Morrill, D’Orazio, Sarfati, Lanctot, J. Wright, A. Greenwald, and Bowling, 2020) by identifying a general class of deviations—called *behavioral deviations*—that induce equilibria that can be found through uncoupled no-regret learning dynamics. Behavioral deviations are defined as those specifying an action transformation independently at each decision node of the game. As the authors note, the deviation functions involved in the definition of EFCE do not fall under that category. A particular

class of behavioral deviation functions—called *causal partial sequence deviations*—induces solution concepts that are (subsets of) EFCEs. Thus, their result begets an alternative set of no-regret learning dynamics that converge to EFCE, based on a different set of deviation functions than those we use in this dissertation.

Organization This chapter is organized as follows. In [Section 8.2](#) we introduce the notion of trigger agents and trigger deviation functions, and formally establish the important connection between such notions and convergence to the set of EFCEs via no-regret dynamics. In [Section 8.3](#) we show how regret with respect to the set of all trigger deviation functions can be kept sublinear. We do so by first identifying a suitable characterization of the set of all trigger deviation functions. Such a characterization exhibits a strong combinatorial structure that naturally lends itself to being addressed via the regret decomposition techniques we saw in [Chapter 4](#). Pseudocode for the resulting learning dynamics, as well as a theoretical analysis, is given in [Section 8.3.3](#).

8.2 Extensive-form correlated equilibrium and its relation with Φ -regret

Extensive-form correlated equilibrium (EFCE) has been proposed by von Stengel and Forges (2008) as the natural counterpart to (normal-form) correlated equilibrium in sequential games. In an EFCE, before the beginning of the game an external mediator draws a recommended action for each of the possible decision nodes that players may encounter in the game, according to some probability distribution known to the player. These recommendations are not immediately revealed to each player. Instead, the mediator incrementally reveals relevant action recommendations as players reach new decision nodes. At any decision node, the acting player is free to deviate from the recommended action, but doing so comes at the cost of future recommendations, which are no longer issued if the player deviates. In an EFCE, the recommended behavior is incentive-compatible for each player, that is, no player is strictly better off ever deviating from any of the mediator’s recommended actions.

We make the definition of EFCE formal in the next subsection.

8.2.1 Trigger agents and trigger deviation functions

Multiple equivalent definitions of EFCE can be given; we will follow the equivalent formulation given by Farina, Ling, Fang, and Sandholm (2019a) based on the concept of *trigger agents* introduced by G. J. Gordon, A. Greenwald, and Marks (2008) and Dudik and G. J. Gordon (2009). Fix any n -player imperfect-information extensive-form game, and let $i \in \llbracket n \rrbracket$ be a player. Furthermore, let $\hat{\sigma} = (j, a) \in \Sigma_i^*$ be a non-empty sequence for Player i , and let $\hat{\pi} \in \Pi_{i, \succ j}$ be a strategy for the

subtree rooted at decision node j of the tree-form decision process for Player i . The $(\hat{\sigma}, \hat{\pi})$ -trigger agent is the agent that plays the game as Player i according to the following rules.

- If the trigger agent has never been recommended to play action a at decision node j , the trigger agent will follow whatever recommendation is issued by the mediator.
- When the trigger agent reaches decision node j and is recommended to play action a , we say that the trigger agent “gets triggered” by the *trigger sequence* $\hat{\sigma} = (j, a)$. This means that, from that point on, the trigger agent will disregard the recommendations and play according to the *continuation strategy* $\hat{\pi}$ from decision node j onward (that is, at j and all its descendant decision nodes).

More formally, a $(\hat{\sigma}, \hat{\pi})$ -trigger agent for Player i is a function that maps a recommended strategy to realized behavior, according to a *trigger deviation function*, which we now formally define.

Definition 8.1 (Trigger deviation function). Let $i \in \llbracket n \rrbracket$, $\hat{\sigma} = (j, a) \in \Sigma_i^*$, and $\hat{\pi} \in \Pi_{i, \succ j}$. We call “*trigger deviation function corresponding to trigger $\hat{\sigma}$ and continuation strategy $\hat{\pi}$ ””, any linear function $\phi_{\hat{\sigma} \rightarrow \hat{\pi}} : \mathbb{R}^{\Sigma_i} \rightarrow \mathbb{R}^{\Sigma_i}$ whose effect on sequence-form strategies is as follows:*

- all strategies $\pi \in \Pi_i$ that do not prescribe the sequence $\hat{\sigma}$ are left unmodified. In symbols,

$$\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\pi) = \pi \quad \forall \pi \in \Pi_i : \pi[\hat{\sigma}] = 0; \quad (8.1)$$

- all strategies $\pi \in \Pi_i$ that prescribe sequence $\hat{\sigma} = (j, a)$ are modified so that the behavior at j and all of its descendants is replaced with the behavior prescribed by the continuation strategy $\hat{\pi}$. In symbols,

$$\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\pi)[\sigma] = \begin{cases} \pi[\sigma] & \text{if } \sigma \not\succeq j \\ \hat{\pi}[\sigma] & \text{if } \sigma \succeq j \end{cases} \quad \forall \sigma \in \Sigma_i, \pi \in \Pi_i : \pi[\hat{\sigma}] = 1. \quad (8.2)$$

At this stage, it is technically unclear whether a linear function that satisfies [Definition 8.1](#) exists for all valid choices of $\hat{\sigma}$ and $\hat{\pi}$. We show that this is indeed the case in [Section 8.3.1](#), by exhibiting such a linear function via an explicit transformation matrix. Until then, we will take for granted the well-posedness of the definition of trigger deviations, and focus on their relationship with EFCE.

Having defined trigger deviation functions, we are now ready to formally define the concept of extensive-form correlated equilibrium (EFCE) as a distribution of recommended behavior, such that no player has incentive to unilaterally deviate from the recommendation according to any trigger deviation function. Formally, we have the following.

Definition 8.2 (ϵ -EFCE). Let Γ be an n -player imperfect information game, where each player $i \in \llbracket n \rrbracket$ has multilinear expected utility $u_i : \mathcal{Q}_1 \times \cdots \times \mathcal{Q}_{\llbracket n \rrbracket} \rightarrow \mathbb{R}$. A distribution $\mu \in \Delta^{\Pi_1 \times \cdots \times \Pi_{\llbracket n \rrbracket}}$ is an ϵ -EFCE if for all $i \in \llbracket n \rrbracket$, $\hat{\sigma} = (j, a) \in \Sigma_i^*$, and $\hat{\pi} \in \Pi_{i, \neq j}$,

$$\sum_{\pi_1 \in \Pi_1} \cdots \sum_{\pi_n \in \Pi_n} \mu[\pi_1, \dots, \pi_n] \left(u_i(\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\pi_i), \pi_{-i}) - u_i(\pi_i, \pi_{-i}) \right) \leq \epsilon.$$

In particular, an EFCE is defined as a 0-EFCE.

8.2.2 Convergence to the set of EFCEs via no- Φ -regret dynamics

We now show that the set of EFCEs can be approached by no- Φ -regret dynamics, where Φ is the set of all trigger deviation functions (Definition 8.1). This result can be thought of as the extension of celebrated connection between correlated equilibrium and internal regret (D. Foster and Vohra, 1997) to imperfect-information extensive-form games.

Theorem 8.1. Consider an n -player repeated imperfect-information extensive-form game, and denote with $\text{Reg}_i^{(T)}$ the Φ -regret of a generic player i , with respect to the set of all trigger deviation functions (Definition 8.1) for that player, that is,

$$\text{Reg}_i^{(T)} := \max_{\substack{\hat{\sigma}=(j,a) \in \Sigma_i^* \\ \hat{\pi} \in \Pi_{i, \neq j}}} \left\{ \sum_{t=1}^T \left(u_i(\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\mathbf{x}_i^{(t)}), \mathbf{x}_{-i}^{(t)}) - u_i(\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}) \right) \right\}.$$

Furthermore, for all $i \in \llbracket i \rrbracket$ and time t , let $\mu_i^{(t)} \in \Delta^{\Pi_i}$ be a distribution of strategies whose expectation is $\mathbf{x}_i^{(t)}$, that is, be such that

$$\mathbf{x}_i^{(t)} = \sum_{\pi_i \in \Pi_i} \mu_i^{(t)}[\pi_i] \pi_i.$$

Then, at all times T the average product distribution of play

$$\bar{\mu}^{(T)} := \frac{1}{T} \sum_{t=1}^T \mu_1^{(t)} \otimes \cdots \otimes \mu_n^{(t)}, \quad (8.3)$$

is an $\epsilon^{(T)}$ -EFCE, where

$$\epsilon^{(T)} := \frac{\max_{i \in \llbracket n \rrbracket} \text{Reg}_i^{(T)}}{T}.$$

Proof. Fix any player $i \in \llbracket n \rrbracket$, trigger sequence $\hat{\sigma} = (j, a) \in \Sigma_i^*$, and continuation strategy $\hat{\pi} \in \Pi_{i, \succ j}$. From the multilinearity of the expected utility function u_i as a function of sequence-form strategies, we obtain that

$$\begin{aligned} u_i(\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\mathbf{x}_i^{(t)}), \mathbf{x}_{-i}^{(t)}) &= u_i\left(\phi_{\hat{\sigma} \rightarrow \hat{\pi}}\left(\sum_{\pi_i \in \Pi_i} \mu_i^{(t)}[\pi_i] \pi_i\right), \sum_{\pi_{-i} \in \Pi_{-i}} \mu_{-i}^{(t)}[\pi_{-i}] \pi_{-i}\right) \\ &= u_i\left(\sum_{\pi_i \in \Pi_i} \mu_i^{(t)}[\pi_i] \phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\pi_i), \sum_{\pi_{-i} \in \Pi_{-i}} \mu_{-i}^{(t)}[\pi_{-i}] \pi_{-i}\right) \quad (8.4) \\ &= \sum_{\pi_1 \in \Pi_1} \cdots \sum_{\pi_n \in \Pi_n} \left(\prod_{r \in \llbracket n \rrbracket} \mu_r^{(t)}[\pi_r]\right) u_i(\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\pi_i), \pi_{-i}), \end{aligned}$$

where (8.4) follows from the linearity of $\phi_{\hat{\sigma} \rightarrow \hat{\pi}}$. Similarly, we have

$$u_i(\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}) = \sum_{\pi_1 \in \Pi_1} \cdots \sum_{\pi_n \in \Pi_n} \left(\prod_{r \in \llbracket n \rrbracket} \mu_r^{(t)}[\pi_r]\right) u_i(\pi_i, \pi_{-i}).$$

Hence, using the definition of Φ -regret for Player i , we can write

$$\begin{aligned} \frac{\text{Reg}_i^{(T)}}{T} &\geq \frac{1}{T} \sum_{t=1}^T \left(u_i(\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\mathbf{x}_i^{(t)}), \mathbf{x}_{-i}^{(t)}) - u_i(\mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}) \right) \\ &= \sum_{\pi_1 \in \Pi_1} \cdots \sum_{\pi_n \in \Pi_n} \left(\frac{1}{T} \sum_{t=1}^T \prod_{r \in \llbracket n \rrbracket} \mu_r^{(t)}[\pi_r] \right) \left(u_i(\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\pi_i), \pi_{-i}) - u_i(\pi_i, \pi_{-i}) \right) \\ &= \sum_{\pi_1 \in \Pi_1} \cdots \sum_{\pi_n \in \Pi_n} \bar{\mu}[\pi_1, \dots, \pi_n] \left(u_i(\phi_{\hat{\sigma} \rightarrow \hat{\pi}}(\pi_i), \pi_{-i}) - u_i(\pi_i, \pi_{-i}) \right). \end{aligned}$$

Rearranging, taking a maximum over players $i \in \llbracket n \rrbracket$, trigger sequences $\hat{\sigma} = (j, a) \in \Sigma_i^*$, and continuation strategies $\hat{\pi} \in \Pi_{i, \succ j}$, and applying Definition 8.2 yields the statement. \square

Theorem 8.1 relates an inherently *global* object—EFCE, a *correlated* distribution of play—to an inherently *local*, *per-player* goal—hindsight rationality for a learning agent with respect to a specific set of deviation functions. In light of this connection, in the rest of the chapter we focus on how any individual player can ensure their Φ -regret with respect to the set of trigger deviations grows sublinear in time. As a result, we will assume that a particular player and corresponding tree-form decision process has been isolated, and we will omit (and imply) all player subscripts from the corresponding TFDP quantities.

Definition 8.3. We refer to a no- Φ -regret algorithm for the set Φ of all trigger deviation functions as a *no-trigger-regret* algorithm.

The rest of the chapter will be concerned with constructing simple, efficient no-trigger-regret dynamics.

8.3 Construction of no-trigger-regret dynamics

For the rest of the chapter we assume that an arbitrary TFDP has been fixed, and omit any player indices.

In this section, we give an explicit construction of a no-trigger-regret algorithm.^[8.a] To that end, in [Section 8.3.1](#) we start by giving an explicit characterization of the set of all trigger deviation functions, showing that it admits a concise combinatorial description in terms of *canonical trigger deviation matrices*. Then, in [Section 8.3.2](#) we study the combinatorial structure of the convex hull of canonical trigger deviation matrices, showing that it can be described using simple convexity-preserving operations already studied in [Chapter 4](#).

8.3.1 Canonical trigger deviation matrices

In this subsection, we show that for any trigger sequence $\hat{\sigma} = (j, a) \in \Sigma^*$ and continuation strategy $\hat{\pi} \in \Pi_{\succ j}$, a particular trigger deviation function ([Definition 8.1](#)) can be written explicitly in matrix form, using a heavily structured object which we coin *canonical trigger deviation matrix*. Such a structural characterization will form the basis for constructing simple, efficient, uncoupled no-regret learning dynamics that converge to the set of EFCEs in any imperfect-information extensive-form game.

Definition 8.4 (Canonical trigger deviation matrix). Let $\hat{\sigma} = (j, a) \in \Sigma^*$ and $\mathbf{y} \in \mathbb{R}_{\geq 0}^{\Sigma_{\succ j}}$. We denote with $\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{y}} \in \mathbb{R}_{\geq 0}^{\Sigma \times \Sigma}$ the matrix whose entries are defined as

$$\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{y}}[\sigma_r, \sigma_c] := \begin{cases} 1 & \text{if } \sigma_c \not\succeq \hat{\sigma} \text{ and } \sigma_r = \sigma_c \\ \mathbf{y}[\sigma_r] & \text{if } \sigma_c = \hat{\sigma} \text{ and } \sigma_r \succ j \\ 0 & \text{otherwise} \end{cases} \quad \forall \sigma_r, \sigma_c \in \Sigma. \quad (8.5)$$

The matrix $\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{y}}$ is called the “*canonical trigger deviation matrix corresponding to trigger $\hat{\sigma}$ and continuation strategy $\hat{\pi}$* ”. Furthermore, the set of all canonical trigger deviation matrices is

^[8.a]In fact, this was the first efficient construction proposed in the literature.

denoted with the symbol

$$\Psi := \left\{ \mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} : \hat{\sigma} = (j, a) \in \Sigma^*, \hat{\pi} \in \Pi_{\succ j} \right\}.$$

Canonical trigger deviation matrices fully characterize the set of trigger deviation functions, as we show next.

Lemma 8.1. For any $\hat{\sigma} = (j, a) \in \Sigma^*$ and $\hat{\pi} \in \Pi_{\succ j}$, the linear function

$$\phi_{\hat{\sigma} \rightarrow \hat{\pi}} : \mathbf{x} \mapsto \mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} \mathbf{x},$$

where $\mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}}$ is as defined in [Definition 8.4](#), is a trigger deviation function corresponding to trigger $\hat{\sigma}$ and continuation strategy $\hat{\pi}$, in the sense of [Definition 8.1](#).

Proof. The proof just amounts to a simple application of several definitions. Let $\pi \in \Pi$ be an arbitrary sequence-form strategy. By expanding the matrix-vector multiplication $\mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} \pi$ using the definition [\(8.5\)](#), we obtain that for all $\sigma \in \Sigma$

$$(\mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} \pi)[\sigma] = \pi[\sigma] \mathbb{1}_{\sigma \not\succeq \hat{\sigma}} + \hat{\pi}[\sigma] \pi[\hat{\sigma}] \mathbb{1}_{\sigma \succ j}. \quad (8.6)$$

There are only two possibilities:

- If $\pi[\hat{\sigma}] = 0$, then [\(8.6\)](#) simplifies to

$$(\mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} \pi)[\sigma] = \begin{cases} \pi[\sigma] & \text{if } \sigma \not\succeq \hat{\sigma} \\ 0 & \text{otherwise.} \end{cases}$$

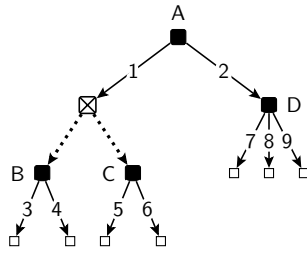
Since by case hypothesis the probability of the sequence of actions from the root of the game tree down to $\hat{\sigma}$ is zero, then necessarily the probability of any longer sequence of actions $\sigma \succ \hat{\sigma}$ must be zero as well, that is $\pi[\sigma] = 0$ for all $\sigma \succ \hat{\sigma}$. So, $\mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} \pi = \pi$ and [\(8.1\)](#) holds.

- Conversely, assume $\pi[\hat{\sigma}] = 1$. This means that at decision node $j \in \mathcal{J}$ action a is selected (with probability 1), and therefore $\pi[\sigma] = 0$ for all $\sigma = (j, a') : a' \in \mathcal{A}_j, a' \neq a$. This means that $\pi[\sigma] \mathbb{1}_{\sigma \not\succeq \hat{\sigma}} = \pi[\sigma] \mathbb{1}_{\sigma \not\succeq j}$ for all $\sigma \in \Sigma$. Substituting that equality into [\(8.6\)](#) gives [\(8.2\)](#), as we wanted to show. \square

As the combinatorial structure of canonical trigger deviation functions will be the crucial component in the construction of our simple no-regret dynamics converging to the set of EFCs, we now pause to provide a few examples. Specifically, in the following example we show three

canonical trigger deviation matrices, and illustrate how they modify sequence-form strategies in a simple extensive-form game.

Example 8.1. We build on the small extensive-form game and example sequence-form strategies defined in Example 2.7, which are reproduced below for convenience.



	π_{135}	π_{136}	π_{145}	π_{27}	π_{28}
\emptyset	$\begin{pmatrix} 1.0 \\ 1.0 \\ 0.0 \\ 1.0 \\ 1.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}$	$\begin{pmatrix} 1.0 \\ 1.0 \\ 0.0 \\ 1.0 \\ 0.0 \\ 1.0 \\ 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}$	$\begin{pmatrix} 1.0 \\ 1.0 \\ 0.0 \\ 0.0 \\ 1.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}$	$\begin{pmatrix} 1.0 \\ 0.0 \\ 1.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 1.0 \\ 0.0 \\ 0.0 \end{pmatrix}$	$\begin{pmatrix} 1.0 \\ 0.0 \\ 1.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 1.0 \\ 0.0 \end{pmatrix}$
A1					
A2					
B3					
B4					
C5					
C6					
D7					
D8					
D9					

We consider the following three examples of canonical trigger deviation matrices:

T_a	T_b	T_c
Trigger sequence: A1 Continuation: A2, D7	Trigger sequence: A2 Continuation: A1, B3, C5	Trigger sequence: B3 Continuation: B4
$\begin{matrix} & \emptyset & A1 & A2 & B3 & B4 & C5 & C6 & D7 & D8 & D9 \\ \emptyset & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ B3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ B4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ C5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ C6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ D7 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ D8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ D9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}$	$\begin{matrix} & \emptyset & A1 & A2 & B3 & B4 & C5 & C6 & D7 & D8 & D9 \\ \emptyset & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ B3 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ B4 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ C5 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ C6 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ D7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ D8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ D9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$	$\begin{matrix} & \emptyset & A1 & A2 & B3 & B4 & C5 & C6 & D7 & D8 & D9 \\ \emptyset & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ B3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ B4 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ C5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ C6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ D7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ D8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ D9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}$

Figure 8.1: Canonical trigger deviation matrices. Entries shaded with dark gray represent the entries of the matrix defined in the second case of Equation (8.5). Given trigger sequence $\hat{\sigma} \in \Sigma^*$, all indices (σ_r, σ_c) such that $\sigma_r, \sigma_c \succcurlyeq \hat{\sigma}$ are shaded with light gray.

First example ($T_{A1 \rightarrow \hat{\pi}_{A1}}$) First, let us consider the trigger deviation matrix corresponding to trigger sequence $\hat{\sigma} = (A, 1)$, and the continuation strategy $\hat{\pi}_{A1}$ that plays action 2 at decision node A, and subsequently action 7 at decision node D. The matrix $T_{A1 \rightarrow \hat{\pi}_{A1}}$ is reported

in Figure 8.1 (Left). In order to illustrate the effect of this linear mapping on sequence-form strategies, we provide some examples using the sequence-form strategy vectors defined in Figure 2.5. First, we observe that any sequence-form strategy choosing action A1 with probability 1 triggers a deviation which follows the continuation strategy $\hat{\pi}$. The deviation for those sequence-form strategies results in a final sequence-form strategy equal to π_{27} . For example, using some of the sequence form strategies of Figure 2.5, it can be easily verified (by working out the matrix-vector product) that:

$$\mathbf{T}_{A1 \rightarrow \hat{\pi}_{A1}} \pi_{135} = \mathbf{T}_{A1 \rightarrow \hat{\pi}_{A1}} \pi_{136} = \mathbf{T}_{A1 \rightarrow \hat{\pi}_{A1}} \pi_{145} = \pi_{27}.$$

On the other hand, sequence-form strategies that do not select sequence 1 are left unmodified by the linear mapping. For instance,

$$\mathbf{T}_{A1 \rightarrow \hat{\pi}_{A1}} \pi_{28} = \pi_{28} \quad \text{and} \quad \mathbf{T}_{A1 \rightarrow \hat{\pi}_{A1}} \pi_{27} = \pi_{27}.$$

Second example ($\mathbf{T}_{A2 \rightarrow \hat{\pi}_{A2}}$) Second, we examine the trigger deviation matrix $\mathbf{T}_{A2 \rightarrow \hat{\pi}_{A2}}$ for trigger sequence $\hat{\sigma} = (A, 2)$, where the continuation strategy $\hat{\pi}_{A2}$ is defined so that Player 1 plays action 1 at decision node A, sequence 3 at decision node B, and action 5 at decision node C. The corresponding matrix $\mathbf{T}_{A2 \rightarrow \hat{\pi}_{A2}}$ is reported in Figure 8.1 (Middle). As in the previous case, all sequence-form strategy vectors which put probability 1 on action 2 at decision node A are modified so that be strategy at A and its descendants B, C, D matches the continuation strategy. For example, we have that

$$\mathbf{T}_{A2 \rightarrow \hat{\pi}_{A2}} \pi_{27} = \mathbf{T}_{A2 \rightarrow \hat{\pi}_{A2}} \pi_{28} = \pi_{135}.$$

Furthermore, sequence-form strategies which do not put probability 1 on sequence A2 are left unchanged. So, for example,

$$\mathbf{T}_{A2 \rightarrow \hat{\pi}_{A2}} \pi_{136} = \pi_{136} \quad \text{and} \quad \mathbf{T}_{A2 \rightarrow \hat{\pi}_{A2}} \pi_{145} = \pi_{145}.$$

Third example ($\mathbf{T}_{B3 \rightarrow \hat{\pi}_{B3}}$) As a final example, Figure 8.1 (Right) shows the deviation matrix $\mathbf{T}_{B3 \rightarrow \hat{\pi}_{B3}}$ corresponding to trigger sequence $\hat{\sigma} = (B, 3)$ and continuation strategy $\hat{\pi}_{B3}$ selecting action 4 at decision node B. Here, we have that

$$\mathbf{T}_{B3 \rightarrow \hat{\pi}_{B3}} \pi_{135} = \mathbf{T}_{B3 \rightarrow \hat{\pi}_{B3}} \pi_{145} \quad \text{and} \quad \mathbf{T}_{B3 \rightarrow \hat{\pi}_{B3}} \pi_{145} = \pi_{145}.$$

8.3.2 Structural decomposition of canonical trigger deviation matrices

The characterization of trigger deviation functions through canonical trigger deviation matrices (Lemma 8.1) enables us to simplify the task of constructing no-trigger-regret dynamics to ensuring sublinear regret compared to canonical trigger deviation matrices. Therefore, for the rest of the chapter our objective will be to construct a no- Ψ -regret algorithm for any sequence-form polytope \mathcal{Q} . We will achieve that by actually considering a slightly more complicated problem: constructing a no- $(\text{co } \Psi)$ -regret algorithm, that is, tackling the *convex hull* of Ψ . It is clear, since $\text{co } \Psi \supseteq \Psi$, that any no- $(\text{co } \Psi)$ -regret algorithm is automatically a no- Ψ -regret algorithm as well.

From the general theory developed in Chapter 4, a no- $(\text{co } \Psi)$ -regret algorithm can be constructed from any no-external-regret algorithm $\tilde{\mathcal{R}}$ for the set $\text{co } \Psi$. In this section we show how the strong combinatorial structure of $\text{co } \Psi$ enables use of regret circuits to construct a no-external-regret algorithm for it.

The starting point of our approach is the observation that, because the convex hull operation is associative, the set

$$\text{co } \Psi = \text{co} \left\{ \mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} : \hat{\sigma} = (j, a) \in \Sigma^*, \hat{\pi} \in \Pi_{\neq j} \right\}$$

can be evaluated in two stages:

1. First, for each sequence $\hat{\sigma} = (j, a) \in \Sigma^*$ one can define the set

$$\Lambda_{\hat{\sigma}} := \text{co} \left\{ \mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} : \hat{\pi} \in \Pi_j \right\};$$

2. Then, one can take the convex hull of all $\Lambda_{\hat{\sigma}}$, that is,

$$\text{co } \Psi = \text{co} \left\{ \Lambda_{\hat{\sigma}} : \hat{\sigma} \in \Sigma^* \right\}. \quad (8.7)$$

Our construction of $\tilde{\mathcal{R}}$ will follow a similar structure. First, for each $\hat{\sigma} \in \Sigma^*$ we will construct a no-regret algorithm $\mathcal{R}_{\hat{\sigma}}$ for the set of deviations $\Lambda_{\hat{\sigma}}$. Then, we will combine all the no-regret algorithms $\mathcal{R}_{\hat{\sigma}}$ into a composite no-regret algorithm $\tilde{\mathcal{R}}$ for the set $\text{co } \Psi$.

No-external-regret algorithm for the inner sets $\Lambda_{\hat{\sigma}}$ For any $\hat{\sigma} = (j, a) \in \Sigma^*$, a no-regret algorithm for the set $\Lambda_{\hat{\sigma}}$ can be constructed starting from any no-regret algorithm for the set $\mathcal{Q}_{\neq j}$ by using one of the composition rules (regret circuits) seen in Chapter 4. The crucial insight lies in the observation that the mapping

$$H_{\hat{\sigma}} : \mathbb{R}^{\Sigma_{\neq j}} \ni \mathbf{y} \mapsto \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{y}}$$

is affine, since it is clear from [Definition 8.4](#) that the entries in $\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{y}} =: H_{\hat{\sigma}}(\mathbf{y})$ are either constants or linear combinations of entries in \mathbf{y} . By using the fact that affine images and convex hulls commute, we can therefore write

$$\Lambda_{\hat{\sigma}} := \text{co} \left\{ \mathbf{T}_{\hat{\sigma} \rightarrow \hat{\pi}} : \hat{\pi} \in \Pi_{\succ j} \right\} = \text{co} H_{\hat{\sigma}}(\Pi_{\succ j}) = H_{\hat{\sigma}}(\text{co} \Pi_{\succ j}) = H_{\hat{\sigma}}(\mathcal{Q}_{\succ j}).$$

So, we have just proved the following characterization of the set $\Lambda_{\hat{\sigma}}$.

Lemma 8.2. For all sequences $\hat{\sigma} = (j, a) \in \Sigma^*$, $\Lambda_{\hat{\sigma}}$ is the image of $\mathcal{Q}_{\succ j}$ under the affine mapping $H_{\hat{\sigma}}$, that is,

$$\Lambda_{\hat{\sigma}} = \left\{ \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}} : \mathbf{x}_{\hat{\sigma}} \in \mathcal{Q}_{\succ j} \right\}.$$

Let $\mathbf{A}_{\hat{\sigma}} \in \mathbb{R}^{(\Sigma \times \Sigma) \times \Sigma_{\succ j}}$, $\mathbf{B} \in \mathbb{R}^{\Sigma \times \Sigma}$ be such that the affine map $H_{\hat{\sigma}}$ has representation

$$H_{\hat{\sigma}}(\mathbf{y}) = \mathbf{A}_{\hat{\sigma}} \mathbf{y} + \mathbf{b} \quad \forall \mathbf{y} \in \mathbb{R}^{\Sigma_{\succ j}}.$$

By leveraging the regret circuit for affine transformations we developed in [Section 4.2.4](#), [Lemma 8.2](#) immediately implies that a no-external-regret algorithm $\mathcal{R}_{\hat{\sigma}}$ for $\Lambda_{\hat{\sigma}}$ can be constructed from any no-external-regret $\tilde{\mathcal{R}}_{\mathcal{Q}, \hat{\sigma}}$ for $\mathcal{Q}_{\succ j}$ (for example, CFR, [§4.3](#)), as done in [Algorithm 8.1](#).

Algorithm 8.1: No-external-regret algorithm $\mathcal{R}_{\hat{\sigma}}$ for set $\Lambda_{\hat{\sigma}} := \{\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}} : \mathbf{x}_{\hat{\sigma}} \in \mathcal{Q}_{\succ j}\}$

Data: • $\hat{\sigma} = (j, a) \in \Sigma^*$ trigger sequence
 • $\tilde{\mathcal{R}}_{\mathcal{Q}, \hat{\sigma}}$ no-external-regret algorithm for set $\mathcal{Q}_{\succ j}$ (e.g., CFR, [Section 4.3](#))

```

1 function NextElement( $\mathbf{M}^{(t)} \in \mathbb{R}^{\Sigma \times \Sigma}$ )
2    $\mathbf{x}_{\hat{\sigma}}^{(t)} \leftarrow \tilde{\mathcal{R}}_{\mathcal{Q}, \hat{\sigma}}. \text{NextElement}(\mathbf{A}_{\hat{\sigma}}^{\top} \mathbf{M}^{(t)})$ 
3   return  $\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^{(t)}}$ , represented in memory implicitly through the vector  $\mathbf{x}_{\hat{\sigma}}^{(t)}$ 

```

```

4 function ObserveUtility( $\mathbf{U}^{(t)} \in \mathbb{R}^{\Sigma \times \Sigma}$ )
5    $\tilde{\mathcal{R}}_{\mathcal{Q}, \hat{\sigma}}. \text{ObserveUtility}(\mathbf{A}_{\hat{\sigma}}^{\top} \mathbf{U}^{(t)})$ 

```

[Algorithm 8.1](#) can be instantiated with any no-regret algorithm $\tilde{\mathcal{R}}_{\mathcal{Q}, \hat{\sigma}}$ for the set of sequence-form strategies $\mathcal{Q}_{\succ j}$. The following proposition formalizes the cumulative regret guarantee when $\tilde{\mathcal{R}}_{\mathcal{Q}, \hat{\sigma}}$ is set to the CFR algorithm ([Section 4.3](#)), which so far has arguably been the most widely used no-regret algorithm for sequence-form strategy spaces.

Proposition 8.1. Let $\hat{\sigma} = (j, a) \in \Sigma^*$ be any trigger sequence. Consider the no-regret algorithm $\mathcal{R}_{\hat{\sigma}}$ ([Algorithm 8.1](#)), where $\tilde{\mathcal{R}}_{\mathcal{Q}, \hat{\sigma}}$ is set to be the CFR no-regret algorithm ([Section 4.3](#)) with RM

as the choice of local no-regret algorithm for each decision node. Then, upon observing any sequence of linear utility matrices $\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(T)} \in \mathbb{R}^{\Sigma \times \Sigma}$, the external regret cumulated by the elements $\mathbf{T}^{(1)} := \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^1}, \dots, \mathbf{T}^{(T)} := \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^T}$ output by $\mathcal{R}_{\hat{\sigma}}$ satisfies

$$\text{Reg}_{\hat{\sigma}}^{(T)} := \max_{\mathbf{T}^* \in \Lambda_{\hat{\sigma}}} \sum_{t=1}^T \langle \mathbf{U}^{(t)}, \mathbf{T}^* - \mathbf{T}^{(t)} \rangle \leq D |\Sigma_{\neq j}| \sqrt{T},$$

where D is the range of $\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(T)}$, i.e., any constant such that $\max_{\mathbf{T} \in \Lambda_{\hat{\sigma}}} \langle \mathbf{U}^{(t)}, \mathbf{T} \rangle \leq D$ for all $t = 1, \dots, T$. Furthermore, the NextElement and the ObserveUtility operations run in $\mathcal{O}(|\Sigma_{\neq j}|)$ time.

Proof. From [Theorem 4.3](#), the regret cumulated by $\mathcal{R}_{\hat{\sigma}}$ upon observing linear utility matrices $\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(t)}$ equals the regret cumulated by the CFR algorithm upon observing linear utility functions $\mathbf{A}_{\hat{\sigma}}^{\top} \mathbf{U}^{(t)}$. Furthermore, the range of $\mathbf{A}_{\hat{\sigma}}^{\top} \mathbf{U}^{(t)}$ satisfies the inequality

$$\max_{\mathbf{x} \in \mathcal{Q}_{\neq j}} \langle \mathbf{A}_{\hat{\sigma}}^{\top} \mathbf{U}^{(t)}, \mathbf{x} \rangle = \max_{\mathbf{x} \in \mathcal{Q}_{\neq j}} \langle \mathbf{U}^{(t)}, \mathbf{A}_{\hat{\sigma}} \mathbf{x} \rangle \leq D,$$

since $\mathbf{A}_{\hat{\sigma}} \mathbf{x} \in \Lambda_{\hat{\sigma}}$.

So, applying the regret bound of the CFR algorithm,

$$\text{Reg}_{\hat{\sigma}}^{(T)} \leq D \left(\sum_{j' \neq j} \sqrt{|\mathcal{A}_{j'}|} \right) \sqrt{T} \leq D \left(\sum_{j' \neq j} |\mathcal{A}_{j'}| \right) \sqrt{T} = D |\Sigma_{\neq j}| \sqrt{T},$$

completing the proof of the regret bound.

We now look at the complexity analysis. First, we observe that $H_{\hat{\sigma}}(\mathbf{y})$ maps each entry of \mathbf{y} to exactly one entry of the resulting canonical trigger deviation matrix $\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{y}}$. Hence, $\mathbf{A}_{\hat{\sigma}}$ has exactly $|\Sigma_{\neq j}|$ nonzero entries, and therefore the products $\mathbf{A}_{\hat{\sigma}}^{\top} \mathbf{M}^{(t)}$, $\mathbf{A}_{\hat{\sigma}}^{\top} \mathbf{U}^{(t)}$ require $\mathcal{O}(|\Sigma_{\neq j}|)$ operations each. Since CFR's NextElement and ObserveUtility operations both run in linear time in $|\Sigma_{\neq j}|$, we conclude that the NextElement of [Algorithm 8.1](#) required a total of $\mathcal{O}(|\Sigma_{\neq j}|)$ operation. \square

No-external-regret algorithm for the convex hull of the $\Lambda_{\hat{\sigma}}$ We have seen in [Section 4.2.3](#) that a no-regret algorithm for a composite set of the form $\text{co}\{\mathcal{X}_1, \dots, \mathcal{X}_m\}$ can be constructed by combining any individual no-regret algorithms for $\mathcal{X}_1, \dots, \mathcal{X}_m$ through the convex hull *regret circuit*.

Hence, we apply the construction described in [Algorithm 4.2](#) to obtain our no-regret algorithm \mathcal{R} for the set $\text{co } \Psi = \text{co}\{\Lambda_{\hat{\sigma}} : \hat{\sigma} \in \Sigma\}$ starting from the no-regret algorithms $\mathcal{R}_{\hat{\sigma}}$ ([Algorithm 8.1](#)),

one for each sequence $\hat{\sigma} \in \Sigma^*$, as well as any no-regret algorithm \mathcal{R}_Δ for the simplex Δ^{Σ^*} . Pseudocode is given in [Algorithm 8.2](#).

Algorithm 8.2: No-regret algorithm $\tilde{\mathcal{R}}$ for the set $\text{co } \Psi = \text{co}\{\Lambda_{\hat{\sigma}} : \hat{\sigma} \in \Sigma\}$

Data: • $\mathcal{R}_{\hat{\sigma}}$, one per $\hat{\sigma} \in \Sigma^*$: no-regret algorithm for $\Lambda_{\hat{\sigma}}$, defined in [Algorithm 8.1](#)
 • \mathcal{R}_Δ no-regret algorithm for Δ^{Σ^*} (e.g., regret matching (Hart and Mas-Colell, 2000))

```

1 function NextElement( $\mathbf{M}^{(t)} \in \mathbb{R}^{\Sigma \times \Sigma}$ )
2   for  $\hat{\sigma} \in \Sigma^*$  do
3      $\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^{(t)}} \leftarrow \mathcal{R}_{\hat{\sigma}}.\text{NextElement}()$  [▷ See Algorithm 8.1]
4      $\mathbf{m}_\Delta^{(t)} \leftarrow \left( \langle \mathbf{M}^{(t)}, \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^{(t)}} \rangle \right)_{\hat{\sigma} \in \Sigma^*}$ 
5      $\lambda^{(t)} \leftarrow \tilde{\mathcal{R}}_{\Delta^{\Sigma^*}}.\text{NextElement}(\mathbf{m}_\Delta^{(t)})$ 
6   return  $\sum_{\hat{\sigma} \in \Sigma^*} \lambda^{(t)}[\hat{\sigma}] \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^{(t)}}$ , represented in memory as list  $\{(\lambda^{(t)}[\hat{\sigma}], \mathbf{x}_{\hat{\sigma}}^{(t)})\}_{\hat{\sigma} \in \Sigma^*}$ 

```

```

7 function ObserveUtility( $\mathbf{U}^{(t)} \in \mathbb{R}^{\Sigma \times \Sigma}$ )
8   for  $\hat{\sigma} \in \Sigma^*$  do
9      $\mathcal{R}_{\hat{\sigma}}.\text{ObserveUtility}(\mathbf{U}^{(t)})$  [▷ See Algorithm 8.1]
10     $\mathbf{u}_\Delta^{(t)} \leftarrow \left( \langle \mathbf{U}^{(t)}, \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^{(t)}} \rangle \right)_{\hat{\sigma} \in \Sigma^*}$ 
11     $\tilde{\mathcal{R}}_\Delta.\text{ObserveUtility}(\mathbf{u}_\Delta^{(t)})$ 

```

Theorem 8.2. Consider the no-regret algorithm $\tilde{\mathcal{R}}$ ([Algorithm 8.2](#)), where \mathcal{R}_Δ is set to the regret matching algorithm, and $\mathcal{R}_{\hat{\sigma}}$ is instantiated as described in [Proposition 8.1](#). Upon observing any sequence of linear utility matrices $\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(T)} \in \mathbb{R}^{\Sigma \times \Sigma}$, the regret cumulated by the transformations $\mathbf{T}^{(1)}, \dots, \mathbf{T}^{(T)} \in \text{co } \Psi$ output by $\tilde{\mathcal{R}}$ satisfies

$$\text{Reg}_{\text{co } \Psi}^{(T)} := \max_{\mathbf{T}^* \in \text{co } \Psi} \sum_{t=1}^T \langle \mathbf{U}^{(t)}, \mathbf{T}^* - \mathbf{T}^{(t)} \rangle \leq 2D |\Sigma| \sqrt{T},$$

where D is any constant such that $\max_{\mathbf{T} \in \text{co } \Psi} \langle \mathbf{U}^{(t)}, \mathbf{T} \rangle \leq D$ for all $t = 1, \dots, T$. Furthermore, the NextElement and the ObserveUtility operations run in $\mathcal{O}(|\Sigma|^2)$ time.

Proof. At all times t , $\langle \mathbf{M}^{(t)}, \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^{(t)}} \rangle, \langle \mathbf{U}^{(t)}, \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}^{(t)}} \rangle \leq D$ is upper bounded by D . Hence, from the known regret bound of the regret matching algorithm (Hart and Mas-Colell, 2000; Zinkevich, Bowling, Johanson, and Piccione, 2007), the regret cumulated by \mathcal{R}_Δ after T iterations is upper bounded as

$$\text{Reg}_{\Delta}^{(T)} \leq D\sqrt{|\Sigma^*|}\sqrt{T} \leq D|\Sigma|\sqrt{T}.$$

On the other hand, the regret bound in [Proposition 8.1](#) shows that, for all $\hat{\sigma} = (j, a) \in \Sigma^*$, the regret $\text{Reg}_{\hat{\sigma}}^{(T)}$ cumulated by $\mathcal{R}_{\hat{\sigma}}$ is upper bounded as $\text{Reg}_{\hat{\sigma}}^{(T)} \leq D|\Sigma_{\neq j}|\sqrt{T}$. Applying [Theorem 4.2](#) together with the fact that $|\Sigma_{\neq j}| \leq |\Sigma|$ for all $j \in \mathcal{J}$, yields the regret bound in the statement.

Furthermore, the regret matching algorithm produces elements in $\mathcal{O}(|\Sigma^*|)$ time, while each iteration of the loop over Σ^* requires $\mathcal{O}(|\Sigma_{\neq j}|)$ time from [Proposition 8.1](#). \square

8.3.3 Complete algorithm and analysis

Having established the existence of an efficient no-external-regret algorithm for the set of canonical trigger deviation matrices $\text{co } \Psi$ ([Algorithm 8.2](#), $\tilde{\mathcal{R}}$), we can now invoke the reduction from Φ -regret to external regret seen in [Section 3.2.3](#) to establish no-trigger-regret dynamics.

Remark 8.1. To complete the construction, it is important to verify that for each canonical trigger deviation function $\mathbf{T}^{(t)}$ output by $\tilde{\mathcal{R}}$, a fixed point strategy $\mathbf{x}^{(t)} = \mathbf{T}^{(t)}\mathbf{x}^{(t)}$ exists and can be computed.

Existence is straightforward: as $\mathbf{x} \mapsto \mathbf{T}^{(t)}\mathbf{x}$ is a continuous function from \mathcal{Q} to itself, a fixed point must exist by Brouwer's fixed-point theorem.

Furthermore, since \mathcal{Q} is a polytope, we remark that a fixed point of each matrix $\mathbf{T}^{(t)}$ can be computed in polynomial time using linear programming. A more efficient algorithm is proposed in an appendix to this chapter, [Section 8.A](#).

The final algorithm is presented pictorially in [Figure 8.2](#) and in pseudocode in [Algorithm 8.3](#).

Algorithm 8.3: No-trigger-regret algorithm for sequence-form strategy polytope \mathcal{Q}

Data: $\tilde{\mathcal{R}}$ no-regret algorithm for Ψ , defined in [Algorithm 8.2](#)

```

1  $\mathbf{x}^{(0)} \leftarrow \mathbf{0} \in \mathbb{R}^{\Sigma}$ 
2 function NextElement( $\mathbf{m}^{(t)} \in \mathbb{R}^{\Sigma}$ )
3    $\mathbf{T}^{(t)} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}}^{(t)} \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}^{(t)}} \in \text{co } \Psi \leftarrow \tilde{\mathcal{R}}.\text{NextElement}(\mathbf{m}^{(t)} \otimes \mathbf{x}^{(t-1)})$  [▷ Algorithm 8.2]
4    $\mathbf{x}^{(t)} \in \mathcal{Q} \leftarrow \text{FixedPoint}(\mathbf{T}^{(t)})$  [▷ See Remark 8.1]
5   return  $\mathbf{x}^{(t)}$ 
6 function ObserveUtility( $\mathbf{u}^{(t)} \in \mathbb{R}^{\Sigma}$ )
7    $\tilde{\mathcal{R}}.\text{ObserveUtility}(\mathbf{u}^{(t)} \otimes \mathbf{x}^{(t)})$  [▷ See Algorithm 8.2]

```

From the guarantees of the different pieces combined, we obtain the following.

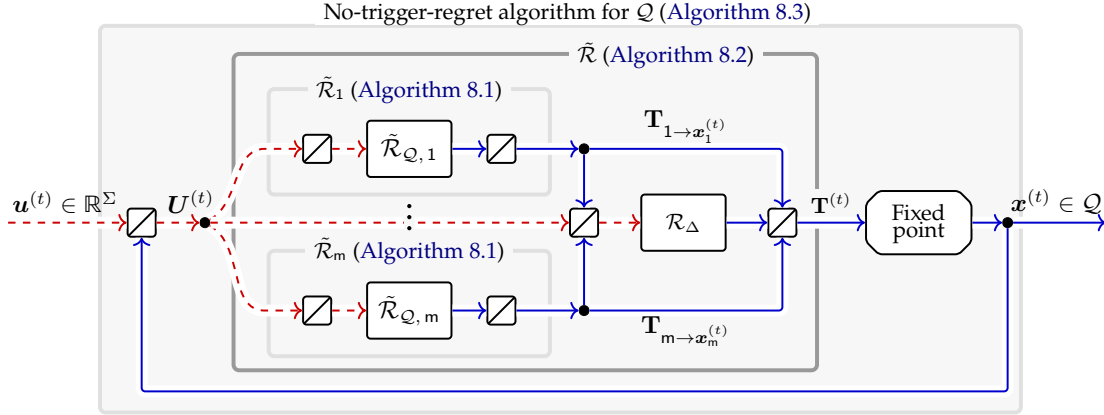


Figure 8.2: Pictorial depiction of our no-(co Ψ)-regret algorithm for the set of sequence-form strategies \mathcal{Q} . For notational convenience we let $\Sigma^* := \{1, \dots, m\}$.

Theorem 8.3. Algorithm \mathcal{R} , defined in Algorithm 8.3, is a no-(co Ψ)-regret algorithm for the set of sequence-form strategies \mathcal{Q} , whose cumulative co Ψ -regret upon observing any sequence of linear utility functions $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(T)} \in \mathbb{R}^\Sigma$ satisfies

$$\text{Reg}^{(T)} := \max_{\mathbf{T}^* \in \text{co } \Psi} \langle \mathbf{u}^{(t)}, \mathbf{T}^*(\mathbf{x}^{(t)}) - \mathbf{x}^{(t)} \rangle \leq 2D|\Sigma|\sqrt{T},$$

where D is any constant such that $\max_{\mathbf{x} \in \mathcal{Q}} \langle \mathbf{u}^{(t)}, \mathbf{x} \rangle \leq D$ for all $t = 1, \dots, T$. Furthermore, ObserveUtility requires time $\mathcal{O}(|\Sigma|^2)$, and NextElement requires $\mathcal{O}(|\Sigma|^2)$ operations, plus the time required to compute a fixed point of $\mathbf{T}^{(t)}$ (see Remark 8.1). In particular, if the algorithm described in the appendix (Section 8.A) is used to compute each fixed point, NextElement requires $\mathcal{O}(|\Sigma|^2 + \sum_{j \in \mathcal{J}} \text{FP}(|\mathcal{A}_j|))$ time at all t , where $\text{FP}(m)$ is the time required to find a fixed point of an $m \times m$ stochastic matrix.

8.4 Final remarks

We conclude by pointing out that the material in this chapter can be extended in several directions, though for space and organizational reasons we decided not to develop these extensions in detail in this dissertation.

For one, the whole machinery presented in this chapter applies to *extensive-form coarse correlated equilibria (EFCCE)* (Farina, Bianchi, and Sandholm, 2020), an intermediate solution concept that sits between EFCE and coarse-correlated equilibria. Since an EFCE is always an EFCCE, the algorithm we presented in this chapter can be used—without modifications—to find an EFCCE

as well. However, as shown by Anagnostides, Farina, Kroer, Celli, and Sandholm (2022), when an EFCCE is sought it is possible to slightly simplify the construction of the algorithm to obtain slightly faster per-iteration complexity.

Another direction is obtaining learning dynamics that can recover EFCE at the near-optimal rate $\tilde{O}_T(1/T)$, similarly to what was done in Chapter 6. This is possible, but is significantly harder due to the presence of the fixed point computation in EFCE (Anagnostides, Farina, Kroer, Celli, and Sandholm, 2022). We refer the reader interested in this direction to the work by Anagnostides, Daskalakis, Farina, Fishelson, Golowich, and Sandholm (2022) and Anagnostides, Farina, and Sandholm (2023).

8.A Appendix: Inductive computation of fixed points of trigger deviation matrices

As mentioned in Section 8.3.3, a fixed point of any transformation $\mathbf{T} \in \text{co } \Psi$ can be computed in a variety of ways, including linear programming. In this appendix, we show that the combinatorial structure of $\text{co } \Psi$ enables us to find a fixed point of any $\mathbf{T} \in \text{co } \Psi$ as the result of the solution of a sequence of fixed point computations for smaller stochastic matrices. This structural understanding will enable us to conclude that any transformation $\mathbf{T} \in \text{co } \Psi$ admits a fixed point $\mathbf{x} = \mathbf{T}\mathbf{x} \in \mathcal{Q}$ that can be computed in time quadratic in the number of sequences Σ .

As a key step in our algorithm, we will use the following well-known result about stationary distributions of stochastic matrices.

Fact 8.1. Any stochastic matrix $\mathbf{A} \in \mathbb{S}^d$ admits a fixed point $\mathbf{A}\mathbf{x} = \mathbf{x} \in \Delta^d$. Furthermore, such a fixed point can be computed in polynomial time in d .

Several specialized algorithms are known for computing fixed points of stochastic matrices (see, *e.g.*, Paige, Styan, and Wachter (1975) for a comparison of eight different methods). Since the particular choice of method is irrelevant, in this dissertation we will make the following assumption.

Assumption 8.1. Given any $m \in \mathbb{N}_{\geq 1}$, we assume access to an oracle for computing a fixed point of any $m \times m$ stochastic matrix \mathbf{A} . Furthermore, we assume that the oracle requires at most $\text{FP}(m)$ time in the worst case to compute any such fixed point.

Our algorithm for computing a fixed point of $\mathbf{T} \in \text{co } \Psi$ requires that the transformation \mathbf{T} be expressed as a convex combination of elements from the sets $\{\Lambda_{\hat{\sigma}}\}_{\hat{\sigma} \in \Sigma^*}$, that is, an expression of

the form

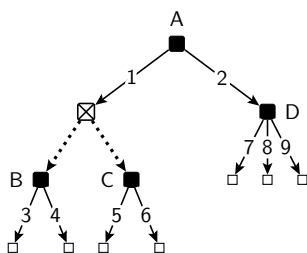
$$\mathbf{T} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}}, \quad \text{with } \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} = 1, \lambda_{\hat{\sigma}} \geq 0, \mathbf{x}_{\hat{\sigma}} \in \mathcal{Q}_{\succ j} \quad \forall \hat{\sigma} = (j, a) \in \Sigma^*, \quad (8.8)$$

in accordance with the characterization of $\text{co } \Psi$ established by (8.7) and Lemma 8.2. Note that our no-regret algorithm $\tilde{\mathcal{R}}$ for the set $\text{co } \Psi$ (Algorithm 8.2) already outputs transformations \mathbf{T} expressed in the form above.

Our algorithm operates *incrementally*, constructing a fixed point sequence-form strategy \mathbf{x} for \mathbf{T} decision node by decision node, in a top down fashion. To formalize this notion of top-down construction, we will make use of the two following definitions.

Definition 8.5. Let $J \subseteq \mathcal{J}$ be a subset of decision nodes. We say that J is a *trunk* of \mathcal{J} if, for every $j \in J$, all predecessors of j (that is, all $j' \in \mathcal{J}$ such that $j' \prec j$) are also in J .

Example 8.2. Consider again the small tree-form decision process of Example 2.7, reproduced below.



In this case, the sets $\{\}, \{A\}, \{A, B\}, \{A, C\}, \{A, D\}, \{A, B, D\}, \{A, C, D\}, \{A, B, C, D\}$ exhaust all the possible trunks.

Conversely, sets $\{B\}$ and $\{B, D\}$ are *not* trunks, because $A \prec B$ and yet A is not in the sets. Similarly, $\{C\}$ and $\{C, D\}$ are *not* trunks, since $A \prec C$ and yet A is not in the sets.

Definition 8.6. Let $J \subseteq \mathcal{J}$ be a trunk of \mathcal{J} (Definition 8.5), and $\mathbf{T} \in \text{co } \Psi$. We say that a vector $\mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma}$ is a J -*partial fixed point* of \mathbf{T} if it satisfies the sequence-form constraints at all $j \in J$, that is,

$$\mathbf{x}[\emptyset] = 1, \quad \mathbf{x}[p_j] = \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] \quad \forall j \in J, \quad (8.9)$$

and furthermore

$$(\mathbf{T}\mathbf{x})[\emptyset] = \mathbf{x}[\emptyset] = 1, \quad (\mathbf{T}\mathbf{x})[ja] = \mathbf{x}[ja] \quad \forall j \in J, a \in \mathcal{A}_j. \quad (8.10)$$

It follows from [Definition 8.6](#) that a \mathcal{J} -partial fixed point of \mathbf{T} is a vector $\mathbf{x} \in \mathcal{Q}$ such that $\mathbf{x} = \mathbf{T}\mathbf{x}$. The following simple lemma establishes a $\{\}$ -partial fixed point for any transformation $\mathbf{T} \in \text{co } \Psi$.

Lemma 8.3. Let $\mathbf{T} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}}$ be any transformation in $\text{co } \Psi$, expressed as in (8.8). Then, the vector $\mathbf{x}_0 \in \mathbb{R}_{\geq 0}^{\Sigma}$, whose entries are all zeros except for $\mathbf{x}_0[\emptyset] = 1$, is a $\{\}$ -partial fixed point of \mathbf{T} .

Proof. Condition (8.9) is straightforward. So, we focus on (8.10). Fix any $\hat{\sigma} = (j, a) \in \Sigma^*$. The definition of $\mathbf{T}_{\hat{\sigma} \rightarrow \hat{\mathbf{x}}}$, given in (8.5), implies that

$$\mathbf{T}_{\hat{\sigma} \rightarrow \hat{\mathbf{x}}}[\sigma_r, \emptyset] = \begin{cases} 1 & \text{if } \sigma_r = \emptyset \\ 0 & \text{otherwise} \end{cases} \quad \forall \sigma_r \in \Sigma.$$

Consequently, $\mathbf{T}_{\hat{\sigma} \rightarrow \hat{\mathbf{x}}}(\mathbf{x}_0) = \mathbf{T}_{\hat{\sigma} \rightarrow \hat{\mathbf{x}}} \mathbf{x}_0 = \mathbf{x}_0$ (from expanding the matrix-vector multiplication). So, $\mathbf{T}(\mathbf{x}_0) = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow \hat{\mathbf{x}}}(\mathbf{x}_0) = \mathbf{x}_0$ and in particular $\mathbf{T}(\mathbf{x}_0)[\emptyset] = \mathbf{x}_0[\emptyset] = 1$. So, (8.10) holds, as we wanted to show. \square

The key result that powers our algorithm to compute a fixed point of any $\mathbf{T} \in \text{co } \Psi$ is that a J -partial fixed point can be cheaply promoted to be a $(J \cup \{j^*\})$ -partial fixed point, where $j^* \in \mathcal{J} \setminus J$ is any decision node whose predecessors are all in J . [Algorithm 8.4](#) below gives an implementation of such a promotion: $\text{Extend}(\mathbf{T}, J, j^*, \mathbf{x})$ starts with a J -partial fixed point \mathbf{x} of \mathbf{T} , and modifies all entries $\mathbf{x}[j^*a]$, $a \in \mathcal{A}_{j^*}$, so that \mathbf{x} becomes a $(J \cup \{j^*\})$ -partial fixed point. Therefore, at a conceptual level, one can repeatedly invoke Extend , growing the trunk J one decision node at a time until $J = \mathcal{J}$, starting from the $\{\}$ -partial fixed point \mathbf{x}_0 described in [Lemma 8.3](#).

Algorithm 8.4: Extend($\mathbf{T}, J, j^*, \mathbf{x}$)

Input : • $\mathbf{T} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}} \in \text{co } \Psi$, represented as list $\{(\lambda_{\hat{\sigma}}, \mathbf{x}_{\hat{\sigma}})\}_{\hat{\sigma} \in \Sigma^*}$

- $J \subseteq \mathcal{J}$ trunk
- $j^* \in \mathcal{J}$ decision node not in J such that its immediate predecessor is in J
- $\mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma}$ J -partial fixed point of \mathbf{T}

Output : • $\mathbf{x}' \in \mathbb{R}_{\geq 0}^{\Sigma}$ $(J \cup \{j^*\})$ -partial fixed point of \mathbf{T}

1 $\sigma_p \leftarrow p_{j^*}$

2 Let $\mathbf{r} \in \mathbb{R}_{\geq 0}^{\mathcal{A}_{j^*}}$ be the vector whose entries are defined, for all $a \in \mathcal{A}_{j^*}$, as

$$\mathbf{r}[a] := \sum_{j' \preceq \sigma_p} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[j^*a] \mathbf{x}[j'a']$$

3 Let $\mathbf{W} \in \mathbb{R}_{\geq 0}^{\mathcal{A}_{j^*} \times \mathcal{A}_{j^*}}$ be the matrix whose entries are defined, for all $a_r, a_c \in \mathcal{A}_{j^*}$, as

$$\mathbf{W}[a_r, a_c] := \mathbf{r}[a_r] + \left(\lambda_{j^*a_c} \mathbf{x}_{j^*a_c}[j^*a_r] + \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^*a_c} \lambda_{\hat{\sigma}} \right) \mathbb{1}_{a_r = a_c} \right) \mathbf{x}[\sigma_p]$$

4 **if** $\mathbf{x}[\sigma_p] = 0$ **then**

5 | $\mathbf{w} \leftarrow \mathbf{0} \in \mathbb{R}_{\geq 0}^{\mathcal{A}_{j^*}}$

6 **else**

7 | $\mathbf{b} \in \Delta^{\mathcal{A}_{j^*}} \leftarrow$ fixed point of stochastic matrix $\frac{1}{\mathbf{x}[\sigma_p]} \mathbf{W}$

8 | $\mathbf{w} \leftarrow \mathbf{x}[\sigma_p] \mathbf{b}$

9 $\mathbf{x}' \leftarrow \mathbf{x}$

10 **for** $a \in \mathcal{A}_{j^*}$ **do**

11 | $\mathbf{x}'[j^*a] \leftarrow \mathbf{w}[j^*a]$

12 **return** \mathbf{x}'

Before giving a proof of correctness and an analysis of the complexity of Extend, we illustrate an application of the algorithm in the simple extensive-form game of [Example 2.5](#).

Example 8.3.

Consider the simple extensive-form game of [Example 2.5](#), and recall the three deviation matrices $\mathbf{T}_{A_1 \rightarrow \hat{\pi}_{A_1}}, \mathbf{T}_{A_2 \rightarrow \hat{\pi}_{A_2}}, \mathbf{T}_{B_3 \rightarrow \hat{\pi}_{B_3}}$ considered in [Example 8.1](#). We will illustrate two applications of Extend, with respect to the transformation

$$\mathbf{T} := \frac{1}{2} \mathbf{T}_{A_1 \rightarrow \hat{\pi}_{A_1}} + \frac{1}{3} \mathbf{T}_{A_2 \rightarrow \hat{\pi}_{A_2}} + \frac{1}{6} \mathbf{T}_{B_3 \rightarrow \hat{\pi}_{B_3}} \in \text{co } \Psi.$$

- In the first application, consider the trunk $J = \{\}$, decision node $j^* = A$, and the $\{\}$ -partial fixed point described in [Lemma 8.3](#), that is, the vector \mathbf{x} whose components are all 0 except for the entry corresponding to the empty sequence \emptyset , which is set to 1.

In this case, $\sigma_p = p_{j^*}$ (Line 1 of Algorithm 8.4) is the empty sequence. Since no decision node j' can possibly satisfy $j' \preceq \sigma_p$, the vector \mathbf{r} defined on Line 2 is the zero vector. Consequently, the matrix \mathbf{W} defined on Line 3 is

$$\mathbf{W} = \begin{matrix} & \begin{matrix} 1 & 2 \end{matrix} \\ \begin{pmatrix} 1/2 & 1/3 \\ 1/2 & 2/3 \end{pmatrix} & \begin{matrix} 1 \\ 2 \end{matrix} \end{matrix}$$

which is a stochastic matrix. A fixed point for \mathbf{W} is given by the vector $\mathbf{b} := (2/5, 3/5) \in \Delta^{\{2,3\}}$. So, the vector \mathbf{x}' returned by Extend is given by

$$\mathbf{x}'[\emptyset] = 1, \quad \mathbf{x}'[A1] = \frac{2}{5}, \quad \mathbf{x}'[A2] = \frac{3}{5}$$

and zero entries everywhere else. Direct computation reveals that \mathbf{x}' is indeed a $\{A\}$ -partial fixed point of \mathbf{T} .

- In the second application of Extend, we start from the $\{A\}$ -partial fixed point that we computed in the previous bullet point, and extend it to a $\{A, D\}$ -partial fixed point. Here, $j^* = D$, and so $\sigma_p = A2$). The only $j' \preceq \sigma_p$ is A , and so the vector \mathbf{r} defined on Line 2 is

$$\mathbf{r}[7] = \frac{1}{5}, \quad \mathbf{r}[8] = 0, \quad \mathbf{r}[9] = 0.$$

Consequently, the matrix \mathbf{W} defined on Line 3 is

$$\mathbf{W} = \begin{matrix} & \begin{matrix} 7 & 8 & 9 \end{matrix} \\ \begin{pmatrix} 3/5 & 1/5 & 1/5 \\ 0 & 2/5 & 0 \\ 0 & 0 & 2/5 \end{pmatrix} & \begin{matrix} 7 \\ 8 \\ 9 \end{matrix} \end{matrix}$$

As expected, $\mathbf{W} \in \frac{3}{5} \mathbb{S}^{\{D7, D8, D9\}} = \mathbf{x}[A2] \mathbb{S}^{\{D7, D8, D9\}}$. A fixed point for $\frac{1}{\mathbf{x}[A2]} \mathbf{W} = \frac{5}{3} \mathbf{W}$ is given by the vector $\mathbf{b} := (1, 0, 0)$. So, the vector \mathbf{x}' returned by Extend is given by

$$\mathbf{x}'[\emptyset] = 1, \quad \mathbf{x}'[A1] = \frac{2}{5}, \quad \mathbf{x}'[A2] = \frac{3}{5}, \quad \mathbf{x}'[D7] = \frac{3}{5}, \quad \mathbf{x}'[D8] = 0, \quad \mathbf{x}'[D9] = 0,$$

and zero entries everywhere else. Once again, direct computation reveals that \mathbf{x}' is indeed a $\{A, D\}$ -partial fixed point of \mathbf{T} .

Proposition 8.2. Let $\mathbf{T} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow x_{\hat{\sigma}}}$ be a linear transformation in $\text{co } \Psi$ expressed as in (8.8), $\mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma}$ be a J -partial fixed point of \mathbf{T} , and $j^* \in \mathcal{J}$ be a decision node not in J such that its immediate predecessor is in J . Then, $\text{Extend}(\mathbf{T}, J, j^*, \mathbf{x})$, given in Algorithm 8.4, computes a $(J \cup \{j^*\})$ -partial fixed point of \mathbf{T} in time upper bounded by $\mathcal{O}(|\Sigma| |\mathcal{A}_{j^*}| + \text{FP}(|\mathcal{A}_{j^*}|))$.

The proof of Proposition 8.2 is deferred until the end of the section. Proposition 8.2 immediately implies that a fixed point for $\mathbf{T} \in \text{co } \Psi$ can be computed by repeatedly invoking Extend to grow the trunk J one decision node at a time, until $J = \mathcal{J}$, starting from the $\{\}$ -partial fixed point $\mathbf{x}_0 \in \mathbb{R}_{\geq 0}^{\Sigma}$ introduced in Lemma 8.3. This leads to Algorithm 8.5, whose correctness and computational complexity is a straightforward corollary of Proposition 8.2.

Algorithm 8.5: FixedPoint(\mathbf{T})

Input : $\mathbf{T} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow x_{\hat{\sigma}}} \in \text{co } \Psi$ transformation
Output : $\mathbf{x} \in \mathcal{Q}$ such that $\mathbf{x} = \mathbf{T}\mathbf{x}$

- 1 $\mathbf{x} \leftarrow \mathbf{0} \in \mathbb{R}^{\Sigma}$, $\mathbf{x}[\emptyset] \leftarrow 1$
- 2 $J \leftarrow \{\}$
- 3 **for** $j \in \mathcal{J}$ in top-down order^[8.b] **do**
- 4 $\mathbf{x} \leftarrow \text{Extend}(\mathbf{T}, J, j, \mathbf{x})$ [▷ See Algorithm 8.4]
- 5 $J \leftarrow J \cup \{j\}$
- 6 **return** \mathbf{x}

Corollary 8.1. Let $\mathbf{T} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow x_{\hat{\sigma}}}$ be a transformation in $\text{co } \Psi$ expressed as in (8.8). Then, Algorithm 8.5 computes a fixed point $\mathcal{Q} \ni \mathbf{x} = \mathbf{T}\mathbf{x}$ in time upper bounded as $\mathcal{O}(|\Sigma|^2 + \sum_{j \in \mathcal{J}} \text{FP}(|\mathcal{A}_j|))$.

Proof of Proposition 8.2

In order to prove correctness of Extend in Proposition 8.2, we will find useful the following technical lemma.

Lemma 8.4. Let $\mathbf{T} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow x_{\hat{\sigma}}}$ be any linear transformation in $\text{co } \Psi$ expressed as in (8.8). Then, for all $\sigma \in \Sigma$,

^[8.b]That is, according to a pre-order tree traversal: if $j \prec j'$, then j appears before j' in the iteration order.

$$(\mathbf{T}\mathbf{x})[\sigma] = \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma} \lambda_{\hat{\sigma}}\right) \mathbf{x}[\sigma] + \sum_{j' \preceq \sigma} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[\sigma] \mathbf{x}[j'a'].$$

Proof. Fix any trigger sequence $\hat{\sigma} = j'a' \in \Sigma^*$. By expanding the matrix-vector multiplication between $\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}}$ (Definition 8.4) and \mathbf{x} , we have that for all $\sigma \in \Sigma$,

$$\mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}}(\mathbf{x})[\sigma] = \mathbf{x}[\sigma] \mathbb{1}_{\sigma \neq \hat{\sigma}} + \mathbf{x}_{\hat{\sigma}}[\sigma] \mathbf{x}[\hat{\sigma}] \mathbb{1}_{\sigma \succ j'}.$$
 (8.11)

Therefore, for all $\sigma \in \Sigma$,

$$\begin{aligned} (\mathbf{T}\mathbf{x})[\sigma] &= \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}}(\mathbf{x})[\sigma] = \sum_{\hat{\sigma} = j'a' \in \Sigma^*} \lambda_{\hat{\sigma}} (\mathbf{x}[\sigma] \mathbb{1}_{\sigma \neq \hat{\sigma}} + \mathbf{x}_{\hat{\sigma}}[\sigma] \mathbf{x}[\hat{\sigma}] \mathbb{1}_{\sigma \succ j'}) \\ &= \left(\sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \neq \sigma} \lambda_{\hat{\sigma}} \right) \mathbf{x}[\sigma] + \sum_{j' \preceq \sigma} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[\sigma] \mathbf{x}[j'a'] \\ &= \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma} \lambda_{\hat{\sigma}}\right) \mathbf{x}[\sigma] + \sum_{j' \preceq \sigma} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[\sigma] \mathbf{x}[j'a'], \end{aligned}$$

as we wanted to show. \square

We are now ready to prove Proposition 8.2, which is restated below for the reader's convenience.

Proposition 8.2 (Restated). Let $\mathbf{T} = \sum_{\hat{\sigma} \in \Sigma^*} \lambda_{\hat{\sigma}} \mathbf{T}_{\hat{\sigma} \rightarrow \mathbf{x}_{\hat{\sigma}}}$ be a linear transformation in $\text{co } \Psi$ expressed as in (8.8), $\mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma}$ be a J -partial fixed point of \mathbf{T} , and $j^* \in \mathcal{J}$ be a decision node not in J such that its immediate predecessor is in J . Then, $\text{Extend}(\mathbf{T}, J, j^*, \mathbf{x})$, given in Algorithm 8.4, computes a $(J \cup \{j^*\})$ -partial fixed point of \mathbf{T} in time upper bounded by $\mathcal{O}(|\Sigma| |\mathcal{A}_{j^*}| + \text{FP}(|\mathcal{A}_{j^*}|))$.

Proof. We break the proof into four parts. In the first part, we analyze the sum of the entries of vector \mathbf{r} defined in Line 2 of Algorithm 8.4. In the second part, we prove that $\frac{1}{\mathbf{x}[\sigma_p]} \mathbf{W} \in \mathbb{S}^{\mathcal{A}_{j^*}}$, as stated in Line 3. In the third part, we show that the output \mathbf{x}' of the algorithm is indeed a $(J \cup \{j^*\})$ -partial fixed point of \mathbf{T} . Finally, in the fourth part we analyze the computational complexity of the algorithm.

Part 1: Sum of the entries of \mathbf{r} In this first part of the proof, we study the sum of the entries of the vector \mathbf{r} defined on Line 2 in Algorithm 8.4. By hypothesis, the immediate predecessor of j^* is in J . So, because \mathbf{x} is a J -partial fixed point, the sequence $\sigma_p := p_{j^*}$ satisfies $(\mathbf{T}\mathbf{x})[\sigma_p] = \mathbf{x}[\sigma_p]$.

Hence, expanding the term $(\mathbf{T}\mathbf{x})[\sigma_p]$ using [Lemma 8.4](#), we conclude that

$$\left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma_p} \lambda_{\hat{\sigma}}\right) \mathbf{x}[\sigma_p] + \sum_{j' \preceq \sigma_p} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[\sigma_p] \mathbf{x}[j'a'] = \mathbf{x}[\sigma_p].$$

By rearranging terms, we have

$$\left(\sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma_p} \lambda_{\hat{\sigma}}\right) \mathbf{x}[\sigma_p] = \sum_{j' \preceq \sigma_p} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[\sigma_p] \mathbf{x}[j'a']. \quad (8.12)$$

On the other hand, since $\mathbf{x}_{j'a'} \in \mathcal{Q}_{\neq j'}$ for all $j' \preceq \sigma_p, a' \in \mathcal{A}_{j'}$, the sequence-form (probability-mass-conservation) constraints [\(2.2\)](#) imply that

$$\mathbf{x}_{j'a'}[\sigma_p] = \sum_{a \in \mathcal{A}_{j^*}} \mathbf{x}_{j'a'}[j^*a]. \quad \forall j' \preceq \sigma_p, a' \in \mathcal{A}_{j'}.$$

Hence, plugging the previous equality into [\(8.12\)](#), we obtain

$$\begin{aligned} \left(\sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma_p} \lambda_{\hat{\sigma}}\right) \mathbf{x}[\sigma_p] &= \sum_{j' \preceq \sigma_p} \sum_{a' \in \mathcal{A}_{j'}} \sum_{a \in \mathcal{A}_{j^*}} \lambda_{j'a'} \mathbf{x}_{j'a'}[j^*a] \mathbf{x}[j'a'] \\ &= \sum_{a \in \mathcal{A}_{j^*}} \left(\sum_{j' \preceq \sigma_p} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[j^*a] \mathbf{x}[j'a']\right) \\ &= \sum_{a \in \mathcal{A}_{j^*}} \mathbf{r}[a], \end{aligned}$$

where the last equality follows from recognizing the definition of \mathbf{r} on [Line 2](#) of [Algorithm 8.4](#). So, in conclusion,

$$\sum_{a \in \mathcal{A}_{j^*}} \mathbf{r}[a] = \left(\sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma_p} \lambda_{\hat{\sigma}}\right) \mathbf{x}[\sigma_p]. \quad (8.13)$$

Part 2: \mathbf{W} belongs to $\mathbf{x}[\sigma_p] \cdot \mathbb{S}^{\mathcal{A}_{j^*}}$ In this second part of the proof, we will prove that all columns of the nonnegative matrix \mathbf{W} , defined on [Line 3](#) of [Algorithm 8.4](#), sum to the same value $\mathbf{x}[\sigma_p]$. Fix any $a_c \in \mathcal{A}_{j^*}$. Using the definition of \mathbf{W} , the sum of the entries in the column of \mathbf{W} corresponding to action a_c is

$$\begin{aligned}
\sum_{a_r \in \mathcal{A}_{j^*}} \mathbf{W}[a_r, a_c] &= \sum_{a_r \in \mathcal{A}_{j^*}} \mathbf{r}[a_r] + \lambda_{j^* a_c} \mathbf{x}_{j^* a_c} [j^* a_r] \mathbf{x}[\sigma_p] \\
&\quad + \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^* a_c} \lambda_{\hat{\sigma}} \right) \mathbb{1}_{a_r = a_c} \mathbf{x}[\sigma_p] \\
&= \left(\sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma_p} \lambda_{\hat{\sigma}} \right) \mathbf{x}[\sigma_p] + \mathbf{x}[\sigma_p] \lambda_{j^* a_c} \left(\sum_{a_r \in \mathcal{A}_{j^*}} \mathbf{x}_{j^* a_c} [j^* a_r] \right) \\
&\quad + \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^* a_c} \lambda_{\hat{\sigma}} \right) \mathbf{x}[\sigma_p] \\
&= \left(\sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma_p} \lambda_{\hat{\sigma}} \right) \mathbf{x}[\sigma_p] + \mathbf{x}[\sigma_p] \lambda_{j^* a_c} + \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^* a_c} \lambda_{\hat{\sigma}} \right) \mathbf{x}[\sigma_p],
\end{aligned}$$

where we used (8.13) in the second equality, and the fact that $\sum_{a_r \in \mathcal{A}_{j^*}} \mathbf{x}_{j^* a_c} [j^* a_r] = 1$ since $\mathbf{x}_{j^* a_c} \in \mathcal{Q}_{\neq j^*}$ (Definition 2.4), in the third. Using the fact that the set of all predecessors of sequence (j^*, a_c) is the union between all predecessors of the parent sequence σ_p and $\{(j^*, a_c)\}$ itself, after rearranging terms we can write

$$\begin{aligned}
\sum_{a_r \in \mathcal{A}_{j^*}} \mathbf{W}[a_r, a_c] &= \left(\sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma_p} \lambda_{\hat{\sigma}} \right) \mathbf{x}[\sigma_p] + \mathbf{x}[\sigma_p] \lambda_{j^* a_c} + \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^* a_c} \lambda_{\hat{\sigma}} \right) \mathbf{x}[\sigma_p] \\
&= \mathbf{x}[\sigma_p] \left(1 + \lambda_{(j^*, a_c)} + \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq \sigma_p} \lambda_{\hat{\sigma}} - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^* a_c} \lambda_{\hat{\sigma}} \right) \\
&= \mathbf{x}[\sigma_p].
\end{aligned}$$

So, all columns of the nonnegative matrix \mathbf{W} sum to $\mathbf{x}[\sigma_p]$, and therefore $\mathbf{W} \in \mathbf{x}[\sigma_p] \cdot \mathbb{S}^{\mathcal{A}_{j^*}}$.

Part 3: \mathbf{x}' is a $(J \cup \{j^*\})$ -partial fixed point of \mathbf{T} We start by arguing that \mathbf{x}' satisfies the sequence-form constraints (8.9) for all $j \in J \cup \{j^*\}$. The crucial observation is that Algorithm 8.4 only modifies the indices corresponding to sequences (j^*, a) for $a \in \mathcal{A}_{j^*}$ and keeps all other entries unmodified. In particular,

$$\mathbf{x}'[ja] = \mathbf{x}[ja] \quad \forall j \in J, a \in \mathcal{A}_j. \quad (8.14)$$

Furthermore, because J is a trunk, the above equation implies that

$$\mathbf{x}'[p_j] = \mathbf{x}[p_j] \quad \forall j \in J.$$

Hence, using the hypothesis that \mathbf{x} is a J -partial fixed point of \mathbf{T} at the beginning of the call, we immediately conclude that the constraints (8.9) corresponding to $j \in J$ hold for vector \mathbf{x}' , and the only condition that remains to be verified is that

$$\mathbf{x}'[\sigma_p] = \sum_{a \in \mathcal{A}_{j^*}} \mathbf{x}'[(j^*, a)]. \quad (8.15)$$

If $\mathbf{x}[\sigma_p] = 0$, then all entries $\mathbf{x}'[(j^*, a)]$ are set to 0 (Line 5) and so (8.15) is trivially satisfied. On the other hand, if $\mathbf{x}[\sigma_p] \neq 0$, then $\mathbf{x}'[j^*a]$ is set to the value $\mathbf{x}[\sigma_p] \mathbf{b}[a]$ (Line 8), and since \mathbf{b} belongs to the simplex $\Delta^{\mathcal{A}_{j^*}}$, (8.15) holds in this case too. So, \mathbf{x}' satisfies (8.9) for all $j \in J \cup \{j^*\}$ as we wanted to show.

We now turn our attention to conditions (8.10). From Lemma 8.4 it follows that $(\mathbf{T}\mathbf{x})[\sigma]$ only depends on the values of $\mathbf{x}[j'a']$ for $j' \preceq \sigma, a' \in \mathcal{A}_{j'}$. So, from (8.14) it follows that

$$\mathbf{T}(\mathbf{x}')[ja] = \mathbf{x}[ja] = \mathbf{x}'[ja] \quad \forall j \in J, a \in \mathcal{A}_j,$$

and the only condition that remains to be verified is that

$$\mathbf{T}(\mathbf{x}')[j^*a^*] = \mathbf{x}'[j^*a^*] \quad \forall a^* \in \mathcal{A}_{j^*}. \quad (8.16)$$

Fix any $a^* \in \mathcal{A}_{j^*}$. We break the analysis into two cases.

- If $\mathbf{x}[\sigma_p] = 0$, then $\mathbf{w} = \mathbf{0}$ (Line 5) and therefore $\mathbf{x}'[j^*a^*] = 0$. Hence, to show that (8.16) holds, we need to show that $\mathbf{T}(\mathbf{x}')[j^*a^*] = 0$. To show that, we start from applying Lemma 8.4:

$$\mathbf{T}(\mathbf{x}')[j^*a^*] = \sum_{j' \preceq j^*a^*} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[j^*a^*] \mathbf{x}'[j'a'].$$

Now, using the fact that $\{j' \in \mathcal{J} : j' \preceq [j^*a^*]\}$ is equal to the disjoint union $\{j' \in \mathcal{J} : j' \preceq \sigma_p\} \cup \{j^*\}$, and that $\mathbf{x}'[(j^*, a')] = 0$ for all $a' \in \mathcal{A}_{j^*}$, we have

$$\mathbf{T}(\mathbf{x}')[j^*a^*] = \sum_{j' \preceq \sigma_p} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[j^*a^*] \mathbf{x}'[j'a']. \quad (8.17)$$

Since $\mathbf{x}_{j'a'} \in \mathcal{Q}_{\succ j'}$ is a nonnegative vector, from Definition 2.4 it follows that

$$\mathbf{x}_{j'a'}[\sigma_p] = \sum_{a \in \mathcal{A}_{j^*}} \mathbf{x}_{j'a'}[j^*a] \geq \mathbf{x}_{j'a'}[j^*a^*]. \quad (8.18)$$

Hence, substituting (8.18) into (8.17),

$$\begin{aligned}\mathbf{T}(\mathbf{x}')[j^*a^*] &\leq \sum_{j' \preceq \sigma_p} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[\sigma_p] \mathbf{x}'[j'a'] \\ &= \mathbf{T}(\mathbf{x}')[\sigma_p] = \mathbf{x}'[\sigma_p] = 0,\end{aligned}$$

where the first equality follows again from [Lemma 8.4](#), and the second equality follows from the inductive hypothesis that \mathbf{x}' is a J -partial fixed point of \mathbf{T} . Since \mathbf{x}' is a nonnegative vector and \mathbf{T} maps nonnegative vectors to nonnegative vectors, we conclude that $\mathbf{T}(\mathbf{x}')[j^*a^*] = 0$ as we wanted to show.

- If $\mathbf{x}[\sigma_p] \neq 0$, then \mathbf{b} is a fixed point of the stochastic matrix $\frac{1}{\mathbf{x}[\sigma_p]} \mathbf{W}$, and therefore it satisfies

$$\sum_{a_c \in \mathcal{A}_{j^*}} \mathbf{W}[a^*, a_c] \mathbf{b}[a_c] = \mathbf{x}[\sigma_p] \mathbf{b}[a^*].$$

Hence, by using the fact that $\mathbf{x}'[j^*a^*] = \mathbf{x}[\sigma_p] \mathbf{b}[a^*]$ ([Line 11](#)), we can write

$$\mathbf{x}'[j^*a^*] = \sum_{a_c \in \mathcal{A}_{j^*}} \mathbf{W}[a^*, a_c] \mathbf{b}[a_c].$$

By expanding the definition of $\mathbf{W}[a^*, a_c]$ ([Line 3](#)) on the right-hand side

$$\begin{aligned}\mathbf{x}'[j^*a^*] &= \sum_{a_c \in \mathcal{A}_{j^*}} \left[\mathbf{r}[a^*] + \left(\lambda_{j^*a_c} \mathbf{x}_{j^*a_c}[j^*a^*] + \left(1 - \sum_{\substack{\hat{\sigma} \in \Sigma^* \\ \hat{\sigma} \preceq j^*a^*}} \lambda_{\hat{\sigma}} \right) \mathbb{1}_{a^*=a_c} \right) \mathbf{x}[\sigma_p] \right] \mathbf{b}[a_c] \\ &= \mathbf{r}[a^*] + \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^*a^*} \lambda_{\hat{\sigma}} \right) \mathbf{x}'[j^*a^*] + \sum_{a_c \in \mathcal{A}_{j^*}} \lambda_{j^*a_c} \mathbf{x}_{j^*a_c}[j^*a^*] \mathbf{x}'[j^*a_c],\end{aligned}$$

where in the second equality we used the fact that $\mathbf{b} \in \Delta^{\mathcal{A}_{j^*}}$, and the fact that $\mathbf{x}'[j^*a] = \mathbf{x}[\sigma_p] \mathbf{b}[a]$ for all $a \in \mathcal{A}_{j^*}$ ([Line 11](#)). Expanding the definition of \mathbf{r} ([Line 2](#)),

$$\begin{aligned}\mathbf{x}'[j^*a^*] &= \sum_{j' \preceq \sigma_p} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[j^*a^*] \mathbf{x}[j'a'] + \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^*a^*} \lambda_{\hat{\sigma}} \right) \mathbf{x}'[j^*a^*] \\ &\quad + \sum_{a_c \in \mathcal{A}_{j^*}} \lambda_{j^*a_c} \mathbf{x}_{j^*a_c}[j^*a^*] \mathbf{x}'[(j^*, a_c)] \\ &= \left(1 - \sum_{\hat{\sigma} \in \Sigma^*, \hat{\sigma} \preceq j^*a^*} \lambda_{\hat{\sigma}} \right) \mathbf{x}'[j^*a^*] + \sum_{j' \preceq j^*a^*} \sum_{a' \in \mathcal{A}_{j'}} \lambda_{j'a'} \mathbf{x}_{j'a'}[j^*a^*] \mathbf{x}'[j'a']\end{aligned}$$

$$= \mathbf{T}(\mathbf{x}')[j^* a^*],$$

where we used the fact that the set $\{j' \in \mathcal{J} : j' \preceq j^* a^*\}$ is equal to the disjoint union $\{j' \in \mathcal{J} : j' \preceq \sigma_p\} \sqcup \{j^*\}$ in the second equality, and [Lemma 8.4](#) in the third equality.

Part 4: Complexity analysis In this part, we bound the number of operations required by [Algorithm 8.4](#).

- **Line 2:** each entry $r[a]$ can be trivially computed in $\mathcal{O}(|\Sigma|)$ time by traversing all predecessors of j^* . So, the vector r requires $\mathcal{O}(|\Sigma| |\mathcal{A}_{j^*}|)$ operations to be computed.
- **Line 3:** if $a_r = a_c$, then the number of operations required to compute $\mathbf{W}[a_r, a_c]$ is dominated by the computation of $\sum_{\tilde{\sigma} \preceq (j^*, a_c)} \lambda_{\tilde{\sigma}}$, which requires $\mathcal{O}(|\Sigma|)$ operations. Otherwise, if $a_r \neq a_c$, the computation of $\mathbf{W}[a_r, a_c]$ can be carried out in a constant number of operations. Hence, the computation of $\mathbf{W}[a_r, a_c]$ for all $a_r, a_c \in \mathcal{A}_{j^*}$ requires $\mathcal{O}(|\Sigma| |\mathcal{A}_{j^*}| + |\mathcal{A}_{j^*}|^2)$ time. Since $|\mathcal{A}_{j^*}| \leq |\Sigma|$, the total number of operations required to compute all entries of \mathbf{W} is $\mathcal{O}(|\Sigma| |\mathcal{A}_{j^*}|)$.
- **Lines 4 to 8:** if $x[\sigma_p] = 0$, then the computation of w requires $\mathcal{O}(|\mathcal{A}_{j^*}|)$ operations. If, on the other hand, $x[\sigma_p] \neq 0$, then the computation of w requires $\mathcal{O}(\text{FP}(|\mathcal{A}_{j^*}|) + |\mathcal{A}_{j^*}|)$ operation. Since clearly any fixed point oracle for a square matrix of order $|\mathcal{A}_{j^*}|$ needs to spend time at least $\Omega(|\mathcal{A}_{j^*}|)$ time writing the output, $\mathcal{O}(\text{FP}(|\mathcal{A}_{j^*}|) + |\mathcal{A}_{j^*}|) = \mathcal{O}(\text{FP}(|\mathcal{A}_{j^*}|))$. So, no matter the value of $x[\sigma_p]$, the number of iterations is bounded by $\mathcal{O}(\text{FP}(|\mathcal{A}_{j^*}|))$.
- **Line 11:** finally, the algorithm spends $\mathcal{O}(|\mathcal{A}_{j^*}|)$ operations to fill out the entries of \mathbf{x} .

Summing the number of operations of each of the different steps of the algorithm, we conclude that each call to $\text{Extend}(\mathbf{T}, J, j^*, \mathbf{x})$ requires at most $\mathcal{O}(|\Sigma| |\mathcal{A}_{j^*}| + \text{FP}(|\mathcal{A}_{j^*}|))$ operations. \square

Chapter 9

Geometry of correlated strategies, and positive complexity results for optimal EFCE

We have shown in [Chapter 8](#) that *one* extensive-form correlated equilibrium can be obtained by following simple, uncoupled no-trigger-regret dynamics. In this chapter we instead focus on the computation of an *optimal* (for example, social-welfare-maximizing) EFCE. Because an EFCE is by definition a correlated distribution over the Cartesian product of the strategies of the players, in order to find one such optimal equilibrium point it is sufficient to have a concise characterization of all such correlated distributions. That will be exactly the focus of this chapter:

Are there polynomially-sized descriptions of the set of all correlated strategies of players in an imperfect-information extensive-form game?

9.1 Contributions

In this chapter, we greatly expand the set of games in which we prove the question can be answered for the positive, obtaining the first *positive* complexity results around optimal correlated equilibria in more than a decade. Specifically, we show the following:

- In two-player games, correlated strategies have a polynomial description provided that a certain condition, which we coin *triangle-freeness*, is satisfied. We then show that the condition holds, for example, when all chance moves are public, that is, both players observe all chance moves. This result immediately implies that a social-welfare-maximizing EFCE can be computed in polynomial time in the game tree size in these games. Furthermore, the

techniques that we introduce to establish the result might be of independent interest.

- As a byproduct of the proof technique, we show that in triangle-free games the polytope of correlated strategies can be obtained via composition of the scaled extension operation that we introduced in [Chapter 4](#). This will set the stage for constructing a state-of-the-art learning algorithm for computing optimal EFCE in [Chapter 10](#).
- In [Section 9.5](#) we mention that in games that are not triangle-free, we recently established a generalization of the result, proving that the polytope of correlated strategies can be expressed as a linear transformation of a set obtained via a chain of scaled extension operations. While the length of the chain is in the worst case exponential in the size of the game tree—this is unavoidable unless $P = NP$ —we show that it can be bounded by a function of a parameter of the game that intuitively measures the amount of asymmetric private information among the players, thereby leading to state-of-the-art parameterized complexity results.
- Finally, in [Section 9.6](#) we put to use the characterization of the polytope of correlated strategies we develop in the chapter to empirically investigate the set of expected utilities that can be reached by EFCEs across nine games. In all cases, we observe that both the shape of the polytope and the range of reachable payoffs are highly nontrivial. This empirically suggests that being able to search and optimize over the space of EFCE (rather than computing a single EFCE without *a priori* guarantees over the social welfare, as in [Chapter 8](#)) is important.

9.2 Preliminaries, notation, and prior work

In this section, we define some notation to describe correlated strategies. We will focus on the case of two correlating players, as that reduces the notational burden while exposing the complexity in its generality. Extending the notation and concepts to more than two players is mechanical. We will denote quantities that belong to the two correlating players using subscripts 1 and 2, respectively.

A *correlated distribution of play* is a probability distribution over deterministic sequence-form strategies $\Pi_1 \times \Pi_2$ of the players. As summarized in [Table 9.1](#), given two decision nodes $j_1 \in \mathcal{J}_1, j_2 \in \mathcal{J}_2$, we say that j_1 and j_2 are *connected*, and write $j_1 \rightleftharpoons j_2$, if there exists at least one possible trajectory in the decision process is consistent with Player 1 acting at j_1 and Player 2 acting at j_2 . Equivalently, in game tree terms, $j_1 \rightleftharpoons j_2$ if and only if there exist nodes u_1 in information set j_1 and u_2 in information set j_2 , such that there exists a path from the root of the game tree to u_2 that passes through u_1 , or *vice versa*. With the notation of connected decision nodes, the notion of *relevant sequence pairs* can be introduced. We say that $(\sigma_1, \sigma_2) \in \Sigma_1 \times \Sigma_2$ form a relevant sequence pair—denoted $\sigma_1 \bowtie \sigma_2$ —if at least one of the sequences is the empty sequence,

Symbol	Description
$j_1 \rightleftharpoons j_2$	Connected decision nodes: if $j_1 \rightleftharpoons j_2$, it is possible that the trajectory in the decision process goes through both $j_1 \in \mathcal{J}_1$ and $j_2 \in \mathcal{J}_2$
$\sigma_1 \bowtie \sigma_2$	Relevant sequence pair: $\sigma_1 \bowtie \sigma_2$ if at least one between σ_1 and σ_2 is the empty sequence \emptyset , or if $\sigma_1 = (j_1, a_1), \sigma_2 = (j_2, a_2)$ with $j_1 \rightleftharpoons j_2$
$\Sigma_1 \bowtie \Sigma_2$	Set of all relevant sequence pairs $\{(\sigma_1, \sigma_2) \in \Sigma_1 \times \Sigma_2 : \sigma_1 \bowtie \sigma_2\}$
$j_1 \bowtie \sigma_2$	Shorthand for: $(j_1, a_1) \bowtie \sigma_2$ for all $a_1 \in A_{j_1}$
$\sigma_1 \bowtie j_2$	Shorthand for: $\sigma_1 \bowtie (j_2, a_2)$ for all $a_2 \in A_{j_2}$

Table 9.1: Additional notation used when dealing with correlation of strategy spaces.

or if $\sigma_1 = (j_1, a_1), \sigma_2 = (j_2, a_2)$ with $j_1 \rightleftharpoons j_2$. Finally, we denote with the symbol $\Sigma_1 \bowtie \Sigma_2$ the set of all relevant sequence pairs, and use the notation $j \bowtie \sigma$ to mean that $(j, a) \bowtie \sigma$ for all $a \in \mathcal{A}_j$.

9.2.1 Polytope of correlation plans Ξ

Consider now a particular terminal state of the game. Let σ_1, σ_2 be the last sequences that were encountered by the two players. Clearly, $\sigma_1 \bowtie \sigma_2$ because the underlying decision nodes must be connected. In order to express the expected utility of such terminal state, it is important to measure the probability according to which strategies consistent with σ_1 and σ_2 were selected. Let μ be the correlated distribution over deterministic sequence-form strategies for the agents. Such a probability is given by

$$\sum_{\substack{\pi \in \Pi_1 \\ \pi[\sigma_1]=1}} \sum_{\substack{\pi' \in \Pi_2 \\ \pi'[\sigma_2]=1}} \mu[\pi, \pi']. \quad (9.1)$$

Next, we extend (9.1) to any pair of relevant sequences.

Definition 9.1. Let $\mu \in \Delta^{\Pi_1 \times \Pi_2}$ be a correlated distribution over deterministic strategies for the correlating agents. We let f denote the function mapping μ to the vector

$$f(\mu)[\sigma_1, \sigma_2] := \sum_{\substack{\pi \in \Pi_1 \\ \pi[\sigma_1]=1}} \sum_{\substack{\pi' \in \Pi_2 \\ \pi'[\sigma_2]=1}} \mu[\pi, \pi'] \quad \forall (\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2. \quad (9.2)$$

The function f just introduced defines a convenient mapping between a correlated distribution of play μ , an exponentially large object, to the vector $f(\mu)$, which only has a polynomial number of entries in the sizes of the correlated strategy spaces. We call $f(\mu)$ the *correlation plan* equivalent to μ . The set of all legal correlation plan is introduced next.

Definition 9.2. The *polytope of correlation plans* Ξ is the image of f (Definition 9.1) as μ varies over the set of all possible correlated distributions over $\Pi_1 \times \Pi_2$, that is,

$$\Xi := \text{Im } f = \left\{ f(\mu) : \mu \in \Delta^{\Pi_1 \times \Pi_2} \right\}.$$

Since the function f (Definition 9.1) is linear and the probability simplex is a convex polytope, it immediately follows that Ξ is indeed a convex polytope, as stated in the following lemma.

Lemma 9.1. The polytope of correlation plans Ξ is a convex polytope, and a subset of $\mathbb{R}_{\geq 0}^{\Sigma_1 \times \Sigma_2}$.

Lemma 9.1 means good news: just like in the non-correlated case (Section 2.3), we can represent strategies as elements of low-dimensional (more precisely: with dimension bounded as a polynomial in the size of the input decision spaces) convex polytope. However, unlike the non-correlated case, in general the constraints that define Ξ are *unknown*. Furthermore, even if the constraints were known, a polynomial number of constraints might not be enough to describe Ξ . A simple computational complexity argument in the work by von Stengel and Forges (2008) shows that this hurdle cannot be avoided, unless $P = NP$.

Remark 9.1. Since f sums up distinct entries from the distribution μ , all entries of $f(\mu)$ belong to $[0, 1]$. Hence, $\Xi \subseteq [0, 1]^{\Sigma_1 \times \Sigma_2}$.

9.2.2 Optimal EFCE as a linear program

We now show that the concise representation defined by Ξ enables to express the set of all EFCEs as a polytope whose number of constraints is polynomial whenever Ξ can be expressed via polynomially many constraints. In order to elucidate the connection between the polytope of all EFCEs and the polytope of correlation plans Ξ , we go back to the very definition of EFCE, which was given in Section 8.2. According to the definition, a correlated strategy $\mu \in \Delta^{\Pi_1 \times \Pi_2}$ is an EFCE when for any player $i \in \{1, 2\}$, there is no trigger sequence $\hat{\sigma} = (j, a) \in \Sigma_i^*$ and continuation strategy (deterministic or mixed) $x_{\hat{\sigma}} \in \mathcal{Q}_{i, \succ j}$ such that the $(\hat{\sigma}, x_{\hat{\sigma}})$ -trigger agent for Player i attains more value in expectation than the player that always play according to the strategies sampled from μ . As we show in Section 9.B, given any $i \in \{1, 2\}$, $\hat{\sigma} = (j, a) \in \Sigma_i^*$, and continuation strategy $x_{\hat{\sigma}} \in \mathcal{Q}_{i, \succ j}$, the difference in expected utility obtained by the $(\hat{\sigma}, x_{\hat{\sigma}})$ -trigger agent is a bilinear function $\xi^\top \mathbf{A}_{i, \hat{\sigma}} x_{\hat{\sigma}}$ of the correlated plan $\xi \in \Xi$ that corresponds to μ , and $x_{\hat{\sigma}}$. Hence, by just using the definition of EFCE we get to the following intermediate result.

Lemma 9.2. An EFCE ξ that maximizes the linear^a objective $c^\top \xi$ is the solution to the following nonlinear optimization problem:

$$\left\{ \begin{array}{l} \max_{\xi} \quad c^\top \xi \\ \text{s.t.} \quad \textcircled{1} \quad \xi \in \Xi \\ \quad \quad \textcircled{2} \quad \max_{\mathbf{x}_{\hat{\sigma}} \in \mathcal{Q}_{i, \succ j}} \xi^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{\hat{\sigma}} \leq 0 \quad \forall i \in \{1, 2\}, \hat{\sigma} = (j, a) \in \Sigma_i^*. \end{array} \right.$$

^aWe remark that by the very construction of correlation plans, the expected utility of a player that plays according to the strategy sampled from ξ is a linear function of ξ . Hence, linear objectives in ξ are in particular able to capture any linear combination of the utilities of the players, including the sum of the utilities (that is, the social welfare).

We now show that the constraints $\textcircled{2}$ can be rewritten as linear constraints, by means of linear programming duality. Indeed, letting $\mathbf{F}_{i, \succ j}, \mathbf{f}_{i, \succ j}$ denote the sequence-form constraints that define $\mathcal{Q}_{i, \succ j}$, that is, $\mathcal{Q}_{i, \succ j} = \{\mathbf{x}_{\hat{\sigma}} : \mathbf{F}_{i, \succ j} \mathbf{x}_{\hat{\sigma}} = \mathbf{f}_{i, \succ j}, \mathbf{x}_{\hat{\sigma}} \geq \mathbf{0}\}$, we have that the linear programs

$$\left\{ \begin{array}{l} \max_{\mathbf{x}_{\hat{\sigma}}} \quad \xi^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{\hat{\sigma}} \\ \text{s.t.} \quad \mathbf{F}_{i, \succ j} \mathbf{x}_{\hat{\sigma}} = \mathbf{f}_{i, \succ j} \\ \quad \quad \mathbf{x}_{\hat{\sigma}} \geq \mathbf{0}, \end{array} \right. \quad \text{and} \quad \left\{ \begin{array}{l} \min_{\mathbf{v}_{i, \hat{\sigma}}} \quad \mathbf{f}_{i, \succ j}^\top \mathbf{v}_{i, \hat{\sigma}} \\ \text{s.t.} \quad \mathbf{F}_{i, \succ j}^\top \mathbf{v}_{i, \hat{\sigma}} \geq \mathbf{A}_{i, \hat{\sigma}}^\top \xi \\ \quad \quad \mathbf{v}_{i, \hat{\sigma}} \in \mathbb{R}^{\mathcal{J}_{i, \succ j}} \end{array} \right.$$

are dual to each other for all $i \in \{1, 2\}$ and trigger sequences $\hat{\sigma} = (j, a) \in \Sigma_i^*$, and therefore have the same value. So, it follows that constraint $\textcircled{2}$ in the formulation of Lemma 9.2 holds if and only if there exists $\mathbf{v}_{i, \hat{\sigma}} \in \mathbb{R}^{\mathcal{J}_{i, \succ j}}$ such that

$$\mathbf{F}_{i, \succ j}^\top \mathbf{v}_{i, \hat{\sigma}} \geq \mathbf{A}_{i, \hat{\sigma}}^\top \xi \quad \text{and} \quad \mathbf{f}_{i, \succ j}^\top \mathbf{v}_{i, \hat{\sigma}} \leq 0.$$

Hence, we have the following.

Proposition 9.1. An EFCE ξ that maximizes the linear objective $c^\top \xi$ is the solution to the following nonlinear optimization problem:

$$\left\{ \begin{array}{l} \max_{\xi} \quad c^\top \xi \\ \text{s.t.} \quad \textcircled{1} \quad \xi \in \Xi \\ \quad \quad \textcircled{2} \quad \mathbf{f}_{i, \succ j}^\top \mathbf{v}_{i, \hat{\sigma}} \leq 0 \quad \forall i \in \{1, 2\}, \hat{\sigma} = (j, a) \in \Sigma_i^* \\ \quad \quad \textcircled{3} \quad \mathbf{F}_{i, \succ j}^\top \mathbf{v}_{i, \hat{\sigma}} \geq \mathbf{A}_{i, \hat{\sigma}}^\top \xi \quad \forall i \in \{1, 2\}, \hat{\sigma} = (j, a) \in \Sigma_i^* \\ \quad \quad \textcircled{4} \quad \mathbf{v}_{i, \hat{\sigma}} \in \mathbb{R}^{\mathcal{J}_{i, \succ j}} \quad \forall i \in \{1, 2\}, \hat{\sigma} = (j, a) \in \Sigma_i^*. \end{array} \right.$$

9.2.3 The von Stengel-Forges polytope \mathcal{V}

We now introduce a second important polytope. It lives in the same Euclidean space as Ξ but unlike Ξ it is defined as the intersection of a polynomial number of linear constraints. This polytope was introduced without name by von Stengel and Forges (2008)—we will refer to it as the *von Stengel-Forges polytope*.

Definition 9.3 (von Stengel-Forges polytope). The *von Stengel-Forges polytope* is the convex polytope of non-negative vectors $\mathbf{v} \in \mathbb{R}_{\geq 0}^{\Sigma_1 \times \Sigma_2}$ indexed over relevant sequence pairs, and defined as

$$\mathcal{V} := \left\{ \mathbf{v} \in \mathbb{R}_{\geq 0}^{\Sigma_1 \times \Sigma_2} : \begin{array}{l} \textcircled{1} \quad \mathbf{v}[\emptyset, \emptyset] = 1 \\ \textcircled{2} \quad \sum_{a \in A_{j_1}} \mathbf{v}[(j_1, a), \sigma_2] = \mathbf{v}[p_{j_1}, \sigma_2] \quad \forall j_1 \in \mathcal{J}_1, \sigma_2 \in \Sigma_2 : j_1 \bowtie \sigma_2 \\ \textcircled{3} \quad \sum_{a \in A_{j_2}} \mathbf{v}[\sigma_1, (j_2, a)] = \mathbf{v}[\sigma_1, p_{j_2}] \quad \forall j_2 \in \mathcal{J}_2, \sigma_1 \in \Sigma_1 : \sigma_1 \bowtie j_2 \end{array} \right\}.$$

We remark that, unlike the sequence-form strategy polytope, the constraints that define the von Stengel-Forges polytope \mathcal{V} (Definition 9.3) do not exhibit a natural hierarchical structure: the same entry in a vector $\mathbf{v} \in \mathcal{V}$ can appear in multiple constraints, and furthermore the constraints will in general form cycles. This makes the problem of decomposing the structure of \mathcal{V} as we did in Section 2.3 significantly more challenging.

9.2.4 von Stengel and Forges (2008)'s result for two-player games without chance

The following important inclusion was shown by von Stengel and Forges (2008), and holds in general—that is, no matter the game.

Lemma 9.3. The von Stengel-Forges polytope is always a superset of the polytope of correlation plans. In symbols, $\Xi \subseteq \mathcal{V}$.

The reverse inclusion might or might not include. In their seminal paper, von Stengel and Forges (2008) show that the reverse inclusion holds in all two-player games without chance moves. From that statement, it follows that $\Xi = \mathcal{V}$, and since \mathcal{V} is a convex polytope defined by a polynomial number of linear constraints, optimization over the set of all EFCEs can be carried out in polynomial time. For more than a decade, von Stengel and Forges (2008)'s result remained our best understanding of the classes of games in which $\mathcal{V} = \Xi$. In this chapter we greatly extend the set of games for which the relationship holds.

9.3 Characterization of the relationship between Ξ and \mathcal{V}

We now give a complete characterization as to when that inclusion $\mathcal{V} \subseteq \Xi$ holds, by connecting it with the integrality of the vertices of the von Stengel-Forges polytope. In doing so, we will find the following lemma useful.

Lemma 9.4. Let $\mathbf{1}_{(\pi_1, \pi_2)} \in \Delta^{\Pi_1 \times \Pi_2}$ denote the distribution over $\Pi_1 \times \Pi_2$ that assigns mass 1 to the pair (π_1, π_2) , and mass 0 to any other pair of reduced-normal-form plans. Then,

$$\Xi = \text{co}\{f(\mathbf{1}_{(\pi_1, \pi_2)}) : \pi_1 \in \Pi_1, \pi_2 \in \Pi_2\}.$$

Proof. The “deterministic” distributions $\mathbf{1}_{(\pi_1, \pi_2)}$ are the vertices of $\Delta^{\Pi_1 \times \Pi_2}$, so, in particular,

$$\Delta^{\Pi_1 \times \Pi_2} = \text{co}\{\mathbf{1}_{(\pi_1, \pi_2)} : \pi_1 \in \Pi_1, \pi_2 \in \Pi_2\}.$$

Since by definition $\Xi = \text{Im } f$, and f is a linear function, the images (under f) of the $\mathbf{1}_{(\pi_1, \pi_2)}$ are a convex basis for Ξ , which is exactly the statement. \square

We are now ready to state our characterization.

Theorem 9.1. Let Γ be a two-player perfect-recall imperfect-information extensive-form game, let \mathcal{V} be its von Stengel-Forges polytope, and let Ξ be its polytope of correlation plans. Then, $\Xi = \mathcal{V}$ if and only if all vertices of \mathcal{V} have integer $\{0, 1\}$ coordinates.

Proof. We prove the two implications separately.

(\Rightarrow) We start by proving that if $\Xi = \mathcal{V}$, then all vertices of \mathcal{V} have integer $\{0, 1\}$ coordinates. Since $\mathcal{V} = \Xi$ by hypothesis, from Lemma 9.4 we can write

$$\mathcal{V} = \text{co}\{f(\mathbf{1}_{(\pi_1, \pi_2)}) : \pi_1 \in \Pi_1, \pi_2 \in \Pi_2\}.$$

So, to prove this direction it is enough to show that $f(\mathbf{1}_{(\pi_1, \pi_2)})$ has integer $\{0, 1\}$ coordinates for all $(\pi_1, \pi_2) \in \Pi_1 \times \Pi_2$. To see that, we use the definition (9.2): each entry in $f(\mathbf{1}_{(\pi_1, \pi_2)})$ is the sum of distinct entries of $\mathbf{1}_{(\pi_1, \pi_2)}$. Given that by definition $\mathbf{1}_{(\pi_1, \pi_2)}$ has exactly one entry with value 1 and $|\Pi_1 \times \Pi_2| - 1$ entries with value 0, we conclude that all coordinates of $f(\mathbf{1}_{(\pi_1, \pi_2)})$ are in $\{0, 1\}$.

(\Leftarrow) We now show that if all vertices of \mathcal{V} have integer $\{0, 1\}$ coordinates, then $\mathcal{V} \subseteq \Xi$. This is enough, since the reverse inclusion, $\mathcal{V} \supseteq \Xi$, is trivial and already known von Stengel

and Forges, 2008. Let $\{v_1, \dots, v_n\}$ be the vertices of \mathcal{V} . To conclude that $\mathcal{V} \subseteq \Xi$, we will prove that $v_i \in \Xi$ for all $i = 1, \dots, n$. This is sufficient since \mathcal{V} and Ξ are convex.

Let $v \in \{v_1, \dots, v_n\}$ be any vertex of \mathcal{V} . By hypothesis, $v[\sigma_1, \sigma_2] \in \{0, 1\}$ for all $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$. Because v satisfies the von Stengel-Forges constraints and furthermore v has $\{0, 1\}$ entries by hypothesis, the two vectors q_1, q_2 defined according to $q_1[\sigma_1] = v[\sigma_1, \emptyset]$ ($\sigma_1 \in \Sigma_1$) and $q_2[\sigma_2] = v[\emptyset, \sigma_2]$ ($\sigma_2 \in \Sigma_2$) are *pure* sequence-form strategies. Now, let π_1^* and π_2^* be the reduced-normal form plans corresponding to q_1 and q_2 , respectively. We will show that $v = f(\mathbf{1}_{(\pi_1^*, \pi_2^*)})$, which will immediately imply that $v \in \Xi$ using Lemma 9.4.

Since $\mathbf{1}_{(\pi_1^*, \pi_2^*)}$ has exactly one positive entry with value 1 in the position corresponding to (π_1^*, π_2^*) , by definition of the linear map f , for any $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$,

$$f(\mathbf{1}_{(\pi_1^*, \pi_2^*)})[\sigma_1, \sigma_2] = \mathbb{1}_{\sigma_1 \in \Pi_1(\pi_1^*)} \cdot \mathbb{1}_{\sigma_2 \in \Pi_2(\pi_2^*)}.$$

So, using the known properties of pure sequence-form strategies, we obtain

$$f(\mathbf{1}_{(\pi_1^*, \pi_2^*)})[\sigma_1, \sigma_2] = q_1[\sigma_1] \cdot q_2[\sigma_2] = v[\sigma_1, \emptyset] \cdot v[\emptyset, \sigma_2] = v[\sigma_1, \sigma_2],$$

where the last equality follows from Lemma 9.10. Since the equality holds for any $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$, we have that $v = f(\mathbf{1}_{(\pi_1^*, \pi_2^*)})$. \square

9.4 Scaled-extension-based structural decomposition for \mathcal{V}

Theorem 9.1 shows that in all games for which the von Stengel-Forges polytope has integral vertices, $\mathcal{V} = \Xi$, and hence computing an optimal EFCE can be done in polynomial time. In this section we will show that in games for which the von Stengel-Forges polytope \mathcal{V} can be expressed as a composition of scaled-extension operations (Definition 4.1) on probability simplexes, the vertices of \mathcal{V} are integral. We later isolate a condition, which we coin *scaled extension*, which implies the existence of such a scaled-extension-based decomposition. Finally, we will show that all two-player games with public chance moves are automatically triangle-free (and not all triangle-free have public chance moves), thereby greatly extending the set of games for which the equality $\mathcal{V} = \Xi$ is known to hold. The chain of implications that lead to result are summarized in Figure 9.1.

Beyond enabling the proof that $\mathcal{V} = \Xi$, we also remark that the existence of a scaled-extension-based decomposition enables the construction of scalable no-external-regret algorithms for \mathcal{V} using the regret circuits construction for scaled extension we presented in Section 4.2.5. We will leverage this observation in Chapter 10.

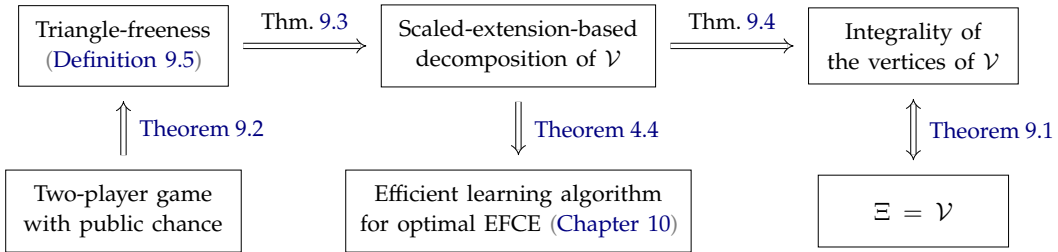


Figure 9.1: Overview of the connections among this chapter’s results.

9.4.1 Examples and intuition

We give three examples of increasing complexity that capture the main intuition behind our structural decomposition routine, and will set the stage for the rest of the theory to come.

Example 9.1. We consider as examples the three small imperfect-information extensive-form games whose game trees are depicted in the top row of Figure 9.2.

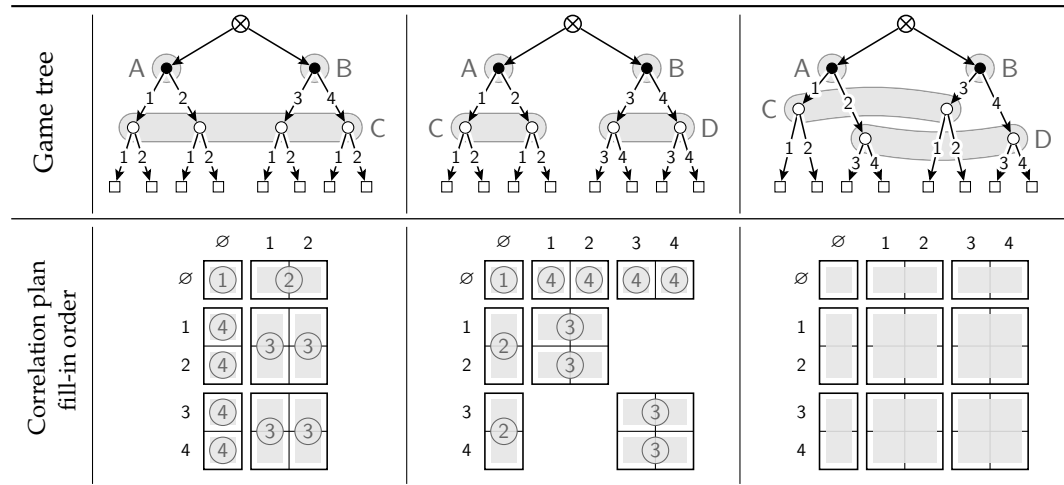


Figure 9.2: Three examples of extensive-form games with increasingly complex information partitions. As usual, the crossed nodes belong to the chance player, the black round nodes belong to Player 1, the white round nodes belong to Player 2, the gray round sets define information sets, and the white squares denote terminal nodes (payoffs are omitted as they are irrelevant). The numbers along the edges define the action names; for clarity we assign unique action names at each information set.

First example Consider the leftmost game in Figure 9.2. The game starts with a chance node,

where two outcomes (say, heads or tails) are possible. After observing the outcome of the chance node, Player 1 chooses between two actions (say, the “left” and the “right” action). The choice as to whether to play the left or the right action can be different based on the observed chance outcome. After Player 1 has played their action, Player 2 has to pick whether to play their left or right action—however, Player 2 does not observe the chance outcome nor Player 1’s action. The chance outcome is not observed by Player 2, so, this is not a public-chance game.

The only information set C for Player 2 is connected to both information sets (denoted A and B in Figure 9.2) of Player 1, so, all sequence pairs $(\sigma_1, \sigma_2) \in \Sigma_1 \times \Sigma_2$ are relevant. Since Player 2 only has one information set, it is easy to incrementally generate the von Stengel-Forges polytope. First, the fixed value 1 is assigned to $v[\emptyset, \emptyset]$ (step ① in the fill-in order). Then, this value is split arbitrarily into the two (non-negative) entries $v[\emptyset, C1], v[\emptyset, C2]$ so that $v[\emptyset, C1] + v[\emptyset, C2] = v[\emptyset, \emptyset]$ in accordance with the von Stengel-Forges constraints. This operation can be expressed using scaled extension as

$$\{(v[\emptyset, \emptyset], v[\emptyset, C1], v[\emptyset, C2])\} = \{1\} \overset{h}{\triangleleft} \Delta^2,$$

where h is the identity function (step ② in the fill-in order).

Then, $v[\emptyset, C1]$ is further split into $v[A1, C1] + v[A2, C1] = v[\emptyset, C1]$ and $v[B3, C1] + v[B4, C1] = v[\emptyset, C1]$, while $v[\emptyset, C2]$ is split into $v[A1, C2] + v[A2, C2] = v[\emptyset, C2]$ and $v[B3, C2] + v[B4, C2] = v[\emptyset, C2]$ (step ③ of the fill-in order). These operations can be expressed as scaled extensions with Δ^2 .

Now that the eight entries $v[\sigma_1, \sigma_2]$ for $\sigma_1 \in \{A1, A2, B3, B4\}, \sigma_2 \in \{C1, C2\}$ have been filled out, we fill in $v[\sigma_1, \emptyset]$ for all $\sigma_1 \in \{A1, A2, B3, B4\}$ in accordance with the von Stengel-Forges constraint $v[\sigma_1, \emptyset] = v[\sigma_1, C1] + v[\sigma_1, C2]$ (step ④). In this step, we are not splitting any values, but rather we are summing already-filled-in entries in v to form new entries. Specifically, we can extend the set of partially-filled-in vectors

$$v = (v[\emptyset, \emptyset], v[\emptyset, C1], v[\emptyset, C2], v[A1, C1], v[A2, C1], \\ v[B3, C1], v[B4, C1], v[A1, C2], v[A2, C2], v[B3, C2], v[B4, C2])$$

with the new entry $v[A1, \emptyset]$ by using the scaled extension operation $\{v\} \overset{h}{\triangleleft} \{1\}$ where h is the (linear) function that extracts the sum $v[\sigma_1, C1] + v[\sigma_1, C2]$ from v . By doing so, we have incrementally filled in all entries in v . Furthermore, by construction, we have that all von Stengel-Forges constraints $v[\sigma_1, \emptyset] = v[\sigma_1, C1] + v[\sigma_1, C2]$ ($\sigma_1 \in \{\emptyset, A1, A2, B3, B4\}$) and $v[\emptyset, \sigma_2] = v[A1, \sigma_2] + v[A2, \sigma_2] = v[B3, \sigma_2] + v[B4, \sigma_2]$ ($\sigma_2 \in \{C1, C2\}$) must hold. So,

the only two von Stengel-Forges constraints that we have ignored and might potentially be violated are $v[\emptyset, \emptyset] = v[A1, \emptyset] + v[A2, \emptyset]$ and $v[\emptyset, \emptyset] = v[B3, \emptyset] + v[B4, \emptyset]$. This concern is quickly resolved by noting that those constraints are implied by the other ones that we satisfy. In particular, by construction we have

$$\begin{aligned} v[A1, \emptyset] + v[A2, \emptyset] &= (v[A1, C1] + v[A1, C2]) + (v[A2, C1] + v[A2, C2]) \\ &= (v[A1, C1] + v[A2, C1]) + (v[A1, C2] + v[A2, C2]) \\ &= v[\emptyset, C1] + v[\emptyset, C2] \\ &= v[\emptyset, \emptyset], \end{aligned}$$

and an analogous statement holds for $v[B3, \emptyset] + v[B4, \emptyset]$. So, all constraints hold and the scaled-extension-based decomposition is finished.

Remark 9.2. An approach that would start by splitting $v[\emptyset, \emptyset]$ into $v[A1, \emptyset] + v[A2, \emptyset] = v[\emptyset, \emptyset]$ and $v[B3, \emptyset] + v[B4, \emptyset] = v[\emptyset, \emptyset]$, thereby inverting the order of fill-in steps ④ and ②, would fail. Indeed, after filling $v[\sigma_1, \sigma_2]$ for all $\sigma_1 \in \{A1, A2, B3, B4\}, \sigma_2 \in \{C1, C2\}$, there would be no clear way of guaranteeing that $v[A1, C1] + v[A2, C1] = v[B3, C1] + v[B4, C1]$ ($= v[\emptyset, C1]$).

Second example We now consider a variation of the game from the first example, where Player 2 observes the chance outcome but not the actions selected by Player 1. This game, shown in the middle column of Figure 9.2, has *public* chance moves, because the chance outcome is observed by all players. In this game, not all pairs of information sets are connected. In fact, only (A, C) and (B, D) are connected information set pairs. Correspondingly, entries such as $v[A1, D3]$, $v[B4, C2]$, and $v[A2, D4]$ are not defined in the correlation plans for the game. This observation is crucial, and will set apart this example from the next one. To fill in any correlation plan, we can start by splitting $v[\emptyset, \emptyset]$ into $v[A1, \emptyset] + v[A2, \emptyset] = v[\emptyset, \emptyset]$ and $v[B3, \emptyset] + v[B4, \emptyset] = v[\emptyset, \emptyset]$ (fill-in step ② in the figure). Both operations can be expressed as a scaled extension of partially-filled-in vectors with Δ^2 , scaled by the affine function that extracts $v[\emptyset, \emptyset] = 1$ from the partially-filled-in correlation plans. Then, we further split those values into entries $v[\sigma_1, C1] + v[\sigma_1, C2] = v[\sigma_1, \emptyset]$ for $\sigma_1 \in \{A1, A2\}$ in accordance with the von Stengel-Forges constraint. Similarly, we will in $v[\sigma_1, D3], v[\sigma_1, D4]$ for $\sigma_1 \in \{B3, B4\}$ in accordance with the constraint $v[\sigma_1, D3] + v[\sigma_1, D4] = v[\sigma_1, \emptyset]$ for $\sigma_1 \in \{A1, A2\}$ (fill-in step ③). Finally, we recover the values of $v[\emptyset, \sigma_2]$ for $\sigma_2 \in \{C1, C2, D3, D4\}$ with a scaled extension with the singleton set $\{1\}$ as discussed in the previous example. Again, it can be

checked that despite the fact that we ignored the constraints $v[\emptyset, C1] + v[\emptyset, C2] = v[\emptyset, \emptyset]$ and $v[\emptyset, D3] + v[\emptyset, D4] = v[\emptyset, \emptyset]$, those constraints are automatically satisfied by construction. In this case, we were able to sidestep the issue raised in [Remark 9.2](#) because of the particular connection between the information sets.

Third example Finally, we propose a third example in the third column of [Figure 9.2](#). It is a variation of the first example, where Player 2 now observes Player 1's action but *not* the chance outcome. The most significant difference with the second example is that the information structure of the game is now such that all pairs of information sets of the players are connected. Hence, the problem raised in [Remark 9.2](#) cannot be avoided. Our decomposition algorithm cannot handle this example.

9.4.2 Triangle-freeness

The third example in the previous section highlights an unfavorable situation in which our decomposition attempt based on incremental generation of the correlation plan. In order to codify all situations in which that issue does not arise, we introduce the concept of rank of an information set.

Definition 9.4. Let $i \in \{1, 2\}$ be a player, and let $-i$ denote the other player. Furthermore, let $j_i \in \mathcal{J}_i$ and $\sigma_{-i} \in \Sigma_{-i}$. The σ_{-i} -rank of j_i is the cardinality of the set

$$\{j_{-i} \in \mathcal{J}_{-i} : j_{-i} \rightleftharpoons j_i, p_{j_{-i}} = \sigma_{-i}\}.$$

The issue in [Remark 9.2](#) can be stated in terms of the ranks. Consider a relevant sequence pair $(\sigma_1, \sigma_2) \in \Sigma_1 \boxtimes \Sigma_2$ and two connected information sets $j_1 \rightleftharpoons j_2$ such that $p_{j_1} = \sigma_1, p_{j_2} = \sigma_2$. If the σ_1 -rank of j_2 and the σ_2 -rank of j_1 are both greater than 1, the issue cannot be avoided and the decomposition will fail. For example, in the third example, where our decomposition fails, all information sets have \emptyset -rank 2. We prove that such situations cannot occur, provided the game satisfies the following condition, which can be verified in polynomial time in the size of the EFG.

Definition 9.5 (Triangle-freeness). A two-player imperfect-information extensive-form game is *triangle-free* if, for any choice of two distinct information sets $j_1, j'_1 \in \mathcal{J}_1$ such that $p_{j_1} = p_{j'_1} = \sigma_1$ and two distinct information sets $j_2, j'_2 \in \mathcal{J}_2$ such that $p_{j_2} = p_{j'_2} = \sigma_2$, it is never the case that $j_1 \rightleftharpoons j_2 \wedge j'_1 \rightleftharpoons j'_2 \wedge j_1 \rightleftharpoons j'_2$.

Using the definition of triangle-freeness, we can verify that the issue exposed in [Remark 9.2](#) never occurs, as the next lemma clarifies.

Lemma 9.5. Consider a triangle-free game, let $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$, and let $j_1 \rightleftharpoons j_2$ be such that $p_{j_1} = \sigma_1, p_{j_2} = \sigma_2$. Then, at most one between the σ_1 -rank of j_2 and the σ_2 -rank of j_1 is strictly larger than 1.

Proof. The results follows almost immediately from the definition of triangle-freeness. We prove the statement by contradiction. Let $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$ be a relevant sequence pair, and let information sets $j_1 \in \mathcal{J}_1, j_2 \in \mathcal{J}_2$ be such that $p_{j_1} = \sigma_1, p_{j_2} = \sigma_2$. Furthermore, assume that the σ_1 -rank of j_2 is greater than 1, and at the same time the σ_2 -rank of j_1 is greater than 1. Since the σ_2 -rank of j_1 is greater than 1, there exists an information set $j'_2 \in \mathcal{J}_2, p_{j'_2} = \sigma_2$, distinct from j_2 , such that $j_1 \rightleftharpoons j'_2$. Similarly, because the σ_1 -rank of j_2 is greater than 1, there exists an information set $j'_1 \in \mathcal{J}_1, p_{j'_1} = \sigma_1$, distinct from j_1 , such that $j'_1 \rightleftharpoons j_2$. But then, we have found $j_1, j'_1 \in \mathcal{J}_1$ and $j'_2, j_2 \in \mathcal{J}_2$ such that $p_{j_1} = p_{j_2} = \sigma_1, p_{j'_2} = p_{j_2} = \sigma_2$ such that $j_1 \rightleftharpoons j'_2, j'_1 \rightleftharpoons j_2$, and $j_1 \rightleftharpoons j_2$. So, the game is *not* triangle-free, contradiction. \square

9.4.3 Two-player games with public chance moves are triangle-free

In [Theorem 9.2](#) we show that games with public chance (which includes games with no chance moves at all) always satisfy the triangle-freeness condition of [Definition 9.5](#).

Theorem 9.2. A two-player imperfect-information extensive-form game with public chance moves is triangle-free.

Proof. For contradiction, let j_1, j_2 be two distinct information sets for Player 1 such that $p_{j_1} = p_{j_2}$, let J_1, J_2 be two distinct information sets for Player 2 such that $p_{J_1} = p_{J_2}$, and assume that $j_1 \rightleftharpoons J_1, j_2 \rightleftharpoons J_2, j_1 \rightleftharpoons J_2$. By definition of connectedness, there exist nodes $u \in j_1, v \in J_1$ such that v is on the path from the root to u , or *vice versa*. Similarly, there exist nodes $u' \in j_2, v' \in J_2$ such that u' is on the path from the root to v' , or *vice versa*. Let w be the lowest common ancestor of u and u' . It is not possible that $w = u$ or $w = u'$, because otherwise the parent sequences of j_1 and j_2 would be different. So w must be a strict ancestor of both u and u' , and u and u' must be reached using *different* edges at w . Therefore, node w cannot belong to Player 1, or otherwise it again would not be true that $p_{j_1} = p_{j_2}$. So, there are only two possible cases: either w belongs to Player 2, or it belongs to the chance player. We break the analysis accordingly.

First case: w belongs to Player 2. From above, we know that u and u' are reached by following different branches at w . So, if both v and v' were strict descendants of w , they would need to be on two different branches of w (because they are connected to u and u' respectively), violating the condition $p_{J_1} = p_{J_2}$. So, at least one between v and v' is on the path from

the root to w (inclusive). But then either v is an ancestor of v' , or *vice versa*. Either case violates the hypothesis that $p_{J_1} = p_{J_2}$.

Second case: w belongs to the chance player. If any between v and v' is an ancestor of w , then necessarily either v is an ancestor of v' , or v' is an ancestor of v . Either case violates the condition $p_{J_1} = p_{J_2}$. So, both v and v' must be descendants of w . Because v is on the path from the root to u (or *vice versa*), and v' is on the path from the root to u' (or *vice versa*), then necessarily u, v and u', v' are on two different branches of the chance node w . To fix names, call a the action at w that must be taken to (eventually) reach u and v , and let b be the action that must be taken to (eventually) reach u' and v' . Now, we use the hypothesis that $j_1 \equiv J_2$, that is, there exists $u'' \in j_1, v'' \in J_2$ such that u'' is on the path from the root to v'' or *vice versa*. Assume that u'' is on the path from the root to v'' . Since u'' belongs to the same information set as u (that is, j_1), and since chance is public by hypothesis, then Player 1, when acting at u and u'' , must have observed action a at w . In other words, the path from the root to u'' must pass through action a at w . But then, using the fact that u'' is on the path from the root to v'' , this means that the path from the root to v'' passes through action a . However, the path from the root to v' passes through action b . Since chance is public, nodes v' and v'' cannot be in the same information set, because Player 2 is able to distinguish them by means of the observed chance outcome. We reached a contradiction. The symmetric case where v'' is on the path from the root to u'' is analogous. \square

However, not all triangle-free games must have public chance nodes. For example, the leftmost game in Figure 9.2 is triangle-free, but in that game the chance outcome is not public to Player 2. So, our results apply more broadly than games with public chance moves.

9.4.4 Computation of the decomposition

Our algorithm consists of a recursive function, `Decompose`. It takes three arguments:

- (i) a sequence pair $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$;
- (ii) a subset \mathcal{S} of the set of all relevant sequence pairs; and
- (iii) a set \mathcal{D} where only the entries indexed by the elements in \mathcal{S} have been filled in.

High-level overview The decomposition for the whole von Stengel-Forges polytope \mathcal{V} is computed by calling `Decompose` $((\emptyset, \emptyset), \{(\emptyset, \emptyset)\}, \{(1)\})$ —this corresponds to the starting situation in which only the entry $v[\emptyset, \emptyset]$ has been filled in (denoted as fill-in step ① in Figure 9.2). Each call to `Decompose` returns a pair $(\mathcal{S}', \mathcal{D}')$ of updated indices and partial vectors, to reflect the new entries that were filled in during the call.

`Decompose` $((\sigma_1, \sigma_2), \mathcal{S}, \mathcal{D})$ operates as follows (we denote with $-i$ the opponent for Player i):

1. Let $\mathcal{B}_i := \{I \in \mathcal{J}_i : I \bowtie \sigma_{-i}, p_I = \sigma_i\}$ for all $i \in \{1, 2\}$, and $\mathcal{J}^* \leftarrow \{\}$.
2. For each $(j_1, j_2) \in \mathcal{B}_1 \times \mathcal{B}_2$ such that $j_1 \rightleftharpoons j_2$, if the σ_2 -rank of I_1 is greater than or equal to the σ_1 -rank of j_2 , we update $\mathcal{J}^* \leftarrow \mathcal{J}^* \cup \{j_1\}$. Else, we update $\mathcal{J}^* \leftarrow \mathcal{J}^* \cup \{j_2\}$.
3. For each $i \in \{1, 2\}$ and $I \in \mathcal{B}_i$ such that the σ_{-i} -rank of I is 0, do $\mathcal{J}^* \leftarrow \mathcal{J}^* \cup \{I\}$.
4. For each $I \in \mathcal{J}^*$: (Below we assume that $I \in \mathcal{J}_1$, the other case is symmetrical)
 - (a) Fill in all entries $\{v[(I, a), \sigma_2] : a \in \mathcal{A}_I\}$ by splitting $v[\sigma_1, \sigma_2]$. This can be expressed using a scaled extension operation as $\mathcal{D} \leftarrow \mathcal{D} \stackrel{h}{\triangleleft} \Delta^{|\mathcal{A}_I|}$ where h extracts $v[\sigma_1, \sigma_2]$ from any partially-filled-in vector.
 - (b) Update $\mathcal{S} \leftarrow \mathcal{S} \cup \{(I, a), \sigma_2\}$ to reflect that the entries corresponding to $(I, a) \bowtie \sigma_2$ have been filled in.
 - (c) For each $a \in \mathcal{A}_I$ we assign $(\mathcal{S}, \mathcal{D}) \leftarrow \text{Decompose}((I, a), \sigma_2, \mathcal{S}, \mathcal{D})$. End for.
 - (d) Let $\mathcal{K} := \{J \in \mathcal{J}_2 : I \rightleftharpoons J\}$. For all $J \in \mathcal{J}_2$ such that $p_J \succcurlyeq (J', a')$ for some $J' \in \mathcal{K}, a' \in \mathcal{A}_{J'}$:
 - If $I \rightleftharpoons J$, then for all $a \in \mathcal{A}_J$ we fill in the sequence pair $v[\sigma_1, (J, a)]$ by assigning its value in accordance with the von Stengel-Forges constraint $v[\sigma_1, (J, a)] = \sum_{a^* \in \mathcal{A}_{I^*}} v[(I^*, a^*), (J, a)]$ via the scaled extension $\mathcal{D} \leftarrow \mathcal{D} \stackrel{h}{\triangleleft} \{1\}$ where the linear function h maps a partially-filled-in vector to the value of $\sum_{a^* \in \mathcal{A}_{I^*}} v[(I^*, a^*), (J, a)]$. Since this is done for all $a \in \mathcal{A}_J$, automatically $\sum_{a \in \mathcal{A}_J} v[\sigma_1, (J, a)] = v[\sigma_1, \sigma_2]$, and we can safely ignore the latter constraint.
 - Otherwise, we fill in the entries $\{v[\sigma_1, (J, a)] : a \in \mathcal{A}_J\}$, by splitting the value $v[\sigma_1, p_J]$. In this case, we let $\mathcal{D} \leftarrow \mathcal{D} \stackrel{h}{\triangleleft} \Delta^{|\mathcal{A}_J|}$ where h extracts the entry $v[\sigma_1, p_J]$ from a partially-filled-in vector in \mathcal{D} .
5. At this point, all the entries corresponding to indices $\tilde{\mathcal{S}} = \{(\sigma'_1, \sigma'_2) : \sigma'_1 \succcurlyeq \sigma_1, \sigma'_2 \succcurlyeq \sigma_2\}$ have been filled in, and we return $(\mathcal{S} \cup \tilde{\mathcal{S}}, \mathcal{D})$.

The above algorithm formalizes and generalizes the first two examples of [Figure 9.2](#). For example, step ② of the fill-in order in either example is captured in [Step 4\(a\)](#), while fill-in step ③ corresponds to [Step 4\(c\)](#). Finally, fill-in step ④ corresponds to [Step 4\(d\)](#). In the rest of the section, we formalize the construction and prove the correctness of the algorithm.

9.4.4.1 Two useful subroutines

We start by presenting two simple subroutines that capture [Step 4\(d\)](#) of [Section 9.4.4](#) (which correspond to fill-in step ④ in [Figure 9.2](#)). The two subroutines are symmetric and have the role of filling rows and columns of the correlation plans.

The following inductive contract will be important for the full algorithm.

Algorithm 9.1: FillOutRow $((\sigma_1, \sigma_2), j_1, \mathcal{S}, \mathcal{D})$

Preconditions: $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2, j_1 \in \mathcal{I}_1, p_{j_1} = \sigma_1, (\sigma_1, \sigma_2) \in \mathcal{S}$

```

1 for  $j_2$  such that  $p_{j_2} = \sigma_2$  and  $\sigma_1 \bowtie j_2$  do
2   if  $j_1 = j_2$  then
3     for  $\sigma'_2 \in \{(j_2, a) : a \in \mathcal{A}_{j_2}\}$  do
4       [ $\triangleright$  Fill  $(\sigma_1, \sigma'_2)$  by summing up all entries
5          $\{v[(j_1, a'), \sigma'_2] : a' \in \mathcal{A}_{j_1}\}$ ]
6        $\mathcal{S} \leftarrow \mathcal{S} \sqcup \{(\sigma_1, \sigma'_2)\}$ 
7        $\mathcal{D} \leftarrow \mathcal{D} \triangleleft^h \{1\}$  where
8          $h : v \mapsto \sum_{a' \in \mathcal{A}_{j_1}} v[(j_1, a'), \sigma'_2]$ 
9     else
10      [ $\triangleright$  Fill all  $\{v[\sigma_1, (j_2, a)] : a \in \mathcal{A}_{j_2}\}$  by
11        splitting  $v[\sigma_1, \sigma_2]$  accordance with the von
12        Stengel-Forges constraints]
13       $\mathcal{S} \leftarrow \mathcal{S} \sqcup \{(\sigma_1, (j_2, a)) : a \in \mathcal{A}_{j_2}\}$ 
14       $\mathcal{D} \leftarrow \mathcal{D} \triangleleft^h \Delta^{\mathcal{A}_{j_2}}$  where  $h : v \mapsto v[\sigma_1, \sigma_2]$ 
15    for  $\sigma'_2 \in \{(j_2, a) : a \in \mathcal{A}_{j_2}\}$  do
16      FillOutRow $((\sigma_1, \sigma'_2), j_1)$ 
17  return  $(\mathcal{S}, \mathcal{D})$ 

```

Algorithm 9.2: FillOutColumn $((\sigma_1, \sigma_2), j_2, \mathcal{S}, \mathcal{D})$

Preconditions: $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2, j_2 \in \mathcal{I}_2, p_{j_2} = \sigma_2, (\sigma_1, \sigma_2) \in \mathcal{S}$

```

1 for  $j_1$  such that  $p_{j_1} = \sigma_1$  and  $\sigma_2 \bowtie j_1$  do
2   if  $j_1 = j_2$  then
3     for  $\sigma'_1 \in \{(j_1, a) : a \in \mathcal{A}_{j_1}\}$  do
4       [ $\triangleright$  Fill  $(\sigma'_1, \sigma_2)$  by summing up all entries
5          $\{v[\sigma'_1, (j_2, a')] : a' \in \mathcal{A}_{j_2}\}$ ]
6        $\mathcal{S} \leftarrow \mathcal{S} \sqcup \{(\sigma'_1, \sigma_2)\}$ 
7        $\mathcal{D} \leftarrow \mathcal{D} \triangleleft^h \{1\}$  where
8          $h : v \mapsto \sum_{a' \in \mathcal{A}_{j_2}} v[\sigma'_1, (j_2, a')]$ 
9     else
10      [ $\triangleright$  Fill all  $\{v[(j_1, a), \sigma_2] : a \in \mathcal{A}_{j_1}\}$  by splitting
11         $v[\sigma_1, \sigma_2]$  accordance with the von
12        Stengel-Forges constraints]
13       $\mathcal{S} \leftarrow \mathcal{S} \sqcup \{(j_1, a), \sigma_2) : a \in \mathcal{A}_{j_1}\}$ 
14       $\mathcal{D} \leftarrow \mathcal{D} \triangleleft^h \Delta^{\mathcal{A}_{j_1}}$  where  $h : v \mapsto v[\sigma_1, \sigma_2]$ 
15    for  $\sigma'_1 \in \{(j_1, a) : a \in \mathcal{A}_{j_1}\}$  do
16      FillOutColumn $((\sigma'_1, \sigma_2), j_2)$ 
17  return  $(\mathcal{S}, \mathcal{D})$ 

```

Lemma 9.6 (Inductive contract for FillOutRow). Suppose that the following preconditions hold when FillOutRow $((\sigma_1, \sigma_2), j_1, \mathcal{S}, \mathcal{D})$ is called:

- (Pre1) $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$;
- (Pre2) $j_1 \in \mathcal{I}_1$ is such that $p_{j_1} = \sigma$;
- (Pre3) \mathcal{S} contains only relevant sequence pairs and \mathcal{D} consists of vectors indexed by exactly the indices in \mathcal{S} ;
- (Pre4) $(\sigma_1, \sigma_2) \in \mathcal{S}$, but $(\sigma_1, \sigma'_2) \notin \mathcal{S}$ for all $\sigma'_2 \succ \sigma_2$;
- (Pre5) For all $a \in j_1$ and $\sigma'_2 \succ \sigma_2$ such that $j_1 \bowtie \sigma'_2, ((j_1, a), \sigma'_2) \in \mathcal{S}$;
- (Pre6) If $j_1 \bowtie \sigma_2$, all $v \in \mathcal{D}$ satisfy the von Stengel-Forges constraint $v[\sigma_1, \sigma_2] = \sum_{a \in j_1} v[(j_1, a), \sigma_2]$;
- (Pre7) All $v \in \mathcal{D}$ satisfy the von Stengel-Forges constraints

$$v[(I, a), p_{j_2}] = \sum_{a' \in \mathcal{A}_{j_2}} v[(I, a), (j_2, a')]$$

for all $a \in j_1$, and $j_2 \in \mathcal{I}_2 : j_1 \bowtie j_2, p_{j_2} \succ \sigma_2$.

Then, the sets $(\mathcal{S}', \mathcal{D}')$ returned by the call are such that

- (Post1) \mathcal{S}' contains only relevant sequence pairs and \mathcal{D}' consists of vectors indexed by

exactly the indices in \mathcal{S}' ;

(Post2) $\mathcal{S}' = \mathcal{S} \sqcup \{(\sigma_1, \sigma'_2) : \sigma'_2 \succ \sigma_2, \sigma \bowtie \sigma'_2\}$;

(Post3) All $v \in \mathcal{D}'$ satisfy the von Stengel-Forges constraints

$$v[\sigma_1, p_{j_2}] = \sum_{a' \in \mathcal{A}_{j_2}} v[\sigma_1, (j_2, a')] \quad \forall j_2 \in \mathcal{I}_2 : \sigma \bowtie j_2, p_{j_2} \succ \sigma_2$$

and all von Stengel-Forges constraints

$$v[\sigma_1, \sigma'_2] = \sum_{a \in \mathcal{A}_{j_1}} v[(I, a), \sigma'_2] \quad \forall \sigma'_2 \in \Sigma_2 : \sigma'_2 \bowtie j_1, \sigma'_2 \succ \sigma_2.$$

Proof. By induction.

Base case The base case corresponds to $\sigma_2 \in \Sigma_2$ such that no information set $j_2 \in \mathcal{I}_2 : p_{j_2} = \sigma_2 \wedge \sigma_1 \bowtie j_2$ exists. In that case, [Algorithm 9.1](#) returns immediately, so (Post1) holds trivially from (Pre3). Since no j_2 such that $p_{j_2} = \sigma_2 \wedge \sigma_1 \bowtie j_2$ exists, no $\sigma'_2 \succ \sigma_2$ such that $\sigma_1 \bowtie \sigma'_2$ exists, so (Post2) holds as well. The first set of constraints of (Post3) is empty, and the second set reduces to (Pre6).

Inductive step Suppose that the inductive hypothesis holds when $\sigma'_2 \succ \sigma_2$. We will show that it holds when $\sigma'_2 = \sigma_2$ as well. In order to use the inductive hypothesis, we first need to check that the preconditions are preserved at the time of the recursive call on [Line 10](#). (Pre1) holds since $\sigma_1 \bowtie j_2$. (Pre2) holds trivially since σ does not chance. (Pre3) holds since we are updating \mathcal{S} and \mathcal{D} in tandem on lines 4, 5 and 7, 8. (Pre4) holds since by the time of the recursive call we have only filled in entries (σ_1, σ'_2) where σ'_2 is an immediate successor of σ_2 . (Pre5) at [Line 10](#) holds trivially, since it refers to a subset of the entries for which the condition held at the beginning of the call. (Pre6) holds because $j_1 \bowtie \sigma'_2 \iff j_1 \rightleftharpoons j_2$. Hence, if $j_1 \bowtie \sigma'_2$ then [Lines 4 and 5](#) must have run. (Pre7) at [Line 10](#) holds trivially, since it refers to a subset of the constraints for which the condition held at the beginning of the call. Using the inductive hypothesis, (Post1), (Post2), and the second set of constraints in (Post3) follow immediately. The only constraints that are left to be verified are

$$v[\sigma_1, \sigma_2] = \sum_{a' \in \mathcal{A}_{j_2}} v[\sigma_1, (j_2, a')] \quad \forall j_2 \in \mathcal{I}_2 : \sigma \bowtie j_2, p_{j_2} = \sigma_2. \quad (9.3)$$

That constraint is guaranteed by [Lines 7 and 8](#) for all $j_2 \neq j_1$. So, we need to verify that it holds for all those j_2 such that $p_{j_2} = \sigma_2, \sigma \bowtie j_2$ and $j_1 \rightleftharpoons j_2$. Let j_2 be one such

information set. Then, from Lines 4 and 5 we have that

$$\mathbf{v}[\sigma_1, (j_2, a)] = \sum_{a' \in \mathcal{A}_{j_1}} \mathbf{v}[(I, a'), (j_2, a)] \quad \forall a \in \mathcal{A}_{j_2}.$$

Summing the above equations across all $a \in \mathcal{A}_{j_2}$ and using (Pre7) yields

$$\begin{aligned} \sum_{a \in \mathcal{A}_{j_2}} \mathbf{v}[\sigma_1, (j_2, a)] &= \sum_{a \in \mathcal{A}_{j_2}} \sum_{a' \in \mathcal{A}_{j_1}} \mathbf{v}[(I, a'), (j_2, a)] \\ &= \sum_{a' \in \mathcal{A}_{j_1}} \sum_{a \in \mathcal{A}_{j_2}} \mathbf{v}[(I, a'), (j_2, a)] \\ &= \sum_{a' \in \mathcal{A}_{j_1}} \mathbf{v}[(I, a'), p_{j_2}] \\ &= \sum_{a' \in \mathcal{A}_{j_1}} \mathbf{v}[(I, a'), \sigma_2], \end{aligned}$$

where we used the hypothesis that $p_{j_2} = \sigma_2$ in the last equality. Finally, since $j_1 \rightleftharpoons j_2$ and $p_{j_2} = \sigma_2$, it must be $j_1 \bowtie \sigma_2$ and so, using (Pre6), we obtain that

$$\sum_{a \in \mathcal{A}_{j_2}} \mathbf{v}[\sigma_1, (j_2, a)] = \mathbf{v}[\sigma_1, \sigma_2],$$

completing the proof of [Equation \(9.3\)](#). So, (Post3) holds as well and the proof of the inductive step is complete. \square

The inductive contract for FillOutColumn is symmetric.

9.4.4.2 The full decomposition algorithm

We are now ready to present the full decomposition algorithm.

Algorithm 9.3: Decompose($(\sigma_1, \sigma_2), \mathcal{S}, \mathcal{D}$)

Preconditions: $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2, (\sigma_1, \sigma_2) \in \mathcal{S}$

```

1   $B \leftarrow \{\}$ 
2  for all  $i \in \{1, 2\}, I \in \mathcal{I}_i, p_I = \sigma_i, \sigma_{-i} \bowtie I$  do
3  |   if the  $\sigma_{-i}$ -rank of  $I$  is 0 then
4  |   |    $B \leftarrow B \sqcup I$ 
5  for  $(j_1, j_2) \in \mathcal{I}_1 \times \mathcal{I}_2$  such that  $p_{j_1} = \sigma_1, p_{j_2} = \sigma_2, j_1 \rightleftharpoons j_2$  do
6  |   if the  $\sigma_2$ -rank of  $j_1$  is  $\geq$  the  $\sigma_1$ -rank of  $j_2$  then
7  |   |    $B \leftarrow B \sqcup j_1$ 
8  |   else
9  |   |    $B \leftarrow B \sqcup j_2$ 
10 for  $I \in B$  do
11 |   if  $I \in \mathcal{I}_1$  then
12 |   |   [ $\triangleright$  Fill all  $\{v[(I, a), \sigma_2] : a \in \mathcal{A}_I\}$  by splitting  $v[\sigma_1, \sigma_2]$  accordance with the von Stengel-Forges constraints]
13 |   |    $\mathcal{S} \leftarrow \mathcal{S} \sqcup \{(I, a), \sigma_2\} : a \in \mathcal{A}_I\}$ 
14 |   |    $\mathcal{D} \leftarrow \mathcal{D} \stackrel{h}{\triangleleft} \Delta^{\mathcal{A}_I}$  where  $h : v \mapsto v[\sigma_1, \sigma_2]$ 
15 |   |   for  $\sigma'_1 \in \{(I, a) : a \in \mathcal{A}_I\}$  do [ $\triangleright$  Recursive call]
16 |   |   |   Decompose( $(\sigma'_1, \sigma_2), \mathcal{S}, \mathcal{D}$ )
17 |   |   [ $\triangleright$  Fill a portion of the row for  $\sigma_1$ ]
18 |   |   for  $j_2 \in \mathcal{I}_2 : \sigma_1 \bowtie j_2, p_{j_2} = \sigma_2$  do
19 |   |   |   for  $\sigma'_2 \in \{(j_2, a') : a' \in \mathcal{A}_{j_2}\}$  do
20 |   |   |   |   [ $\triangleright$  Fill  $(\sigma_1, \sigma'_2)$  by summing up all entries  $\{v[(I, a'), \sigma'_2] : a' \in \mathcal{A}_I\}$ ]
21 |   |   |   |    $\mathcal{S} \leftarrow \mathcal{S} \sqcup \{(\sigma_1, \sigma'_2)\}$ 
22 |   |   |   |    $\mathcal{D} \leftarrow \mathcal{D} \stackrel{h}{\triangleleft} \{1\}$  where  $h : v \mapsto \sum_{a' \in \mathcal{A}_I} v[(I, a'), \sigma'_2]$ 
23 |   |   |   |   FillOutRow( $(\sigma_1, \sigma'_2), I$ )
24 |   |   else
25 |   |   |   [ $\triangleright$  Fill all  $\{v[\sigma_1, (I, a)] : a \in \mathcal{A}_I\}$  by splitting  $v[\sigma_1, \sigma_2]$  accordance with the von Stengel-Forges constraints]
26 |   |   |    $\mathcal{S} \leftarrow \mathcal{S} \sqcup \{(\sigma_1, (I, a)) : a \in \mathcal{A}_I\}$ 
27 |   |   |    $\mathcal{D} \leftarrow \mathcal{D} \stackrel{h}{\triangleleft} \Delta^{\mathcal{A}_I}$  where  $h : v \mapsto v[\sigma_1, \sigma_2]$ 
28 |   |   |   for  $\sigma'_2 \in \{(I, a) : a \in \mathcal{A}_I\}$  do [ $\triangleright$  Recursive call]
29 |   |   |   |   Decompose( $(\sigma_1, \sigma'_2), \mathcal{S}, \mathcal{D}$ )
30 |   |   |   [ $\triangleright$  Fill a portion of the column for  $\sigma_2$ ]
31 |   |   |   for  $j_1 \in \mathcal{I}_1 : \sigma_2 \bowtie j_1, p_{j_1} = \sigma_1$  do
32 |   |   |   |   for  $\sigma'_1 \in \{(j_1, a') : a' \in \mathcal{A}_{j_1}\}$  do
33 |   |   |   |   |   [ $\triangleright$  Fill  $(\sigma'_1, \sigma_2)$  by summing up all entries  $\{v[\sigma'_1, (I, a')] : a' \in \mathcal{A}_I\}$ ]
34 |   |   |   |   |    $\mathcal{S} \leftarrow \mathcal{S} \sqcup \{(\sigma'_1, \sigma_2)\}$ 
35 |   |   |   |   |    $\mathcal{D} \leftarrow \mathcal{D} \stackrel{h}{\triangleleft} \{1\}$  where  $h : v \mapsto \sum_{a' \in \mathcal{A}_I} v[\sigma'_1, (I, a')]$ 
36 |   |   |   |   |   FillOutColumn( $(\sigma'_1, \sigma_2), I$ )
37 return  $(\mathcal{S}, \mathcal{D})$ 

```

Lemma 9.7 (Inductive contract for Decompose). Assume that at the beginning of each call to $\text{Decompose}((\sigma_1, \sigma_2), \mathcal{S}, \mathcal{D})$ the following conditions hold

- (Pre1) \mathcal{S} contains only relevant sequence pairs and \mathcal{D} consists of vectors indexed by exactly the indices in \mathcal{S} .
- (Pre2) \mathcal{S} does not contain any relevant sequence pairs which are descendants of (σ_1, σ_2) , with the only exception of (σ_1, σ_2) itself. In formulas,

$$\mathcal{S} \cap \{(\sigma'_1, \sigma'_2) \in \Sigma_1 \times \Sigma_2 : \sigma'_1 \succ \sigma_1, \sigma'_2 \succ \sigma_2\} = \{(\sigma_1, \sigma_2)\}.$$

Then, at the end of the call, the returned sets $(\mathcal{S}', \mathcal{D}')$ are such that

- (Post1) \mathcal{S}' contains only relevant sequence pairs and \mathcal{D}' consists of vectors v indexed by exactly the indices in \mathcal{S}' .
- (Post2) The call has filled in exactly all relevant sequence pair indices that are descendants of (σ_1, σ_2) (except for (σ_1, σ_2) itself, which was already filled in). In formulas,

$$\mathcal{S}' = \mathcal{S} \sqcup \{(\sigma'_1, \sigma'_2) \in \Sigma_1 \times \Sigma_2 : \sigma'_1 \succ \sigma_1, \sigma'_2 \succ \sigma_2, (\sigma'_1, \sigma'_2) \neq (\sigma_1, \sigma_2), \sigma'_1 \bowtie \sigma'_2\}.$$

- (Post3) \mathcal{D}' satisfies the subset of von Stengel-Forges constraints

$$\begin{aligned} \sum_{a \in A_I} v[(I, a), \sigma'_2] &= v[p_I, \sigma'_2] \quad \forall \sigma'_2 \succ \sigma_2, I \in \mathcal{I}_1 \text{ s.t. } \sigma'_2 \bowtie I, p_I \succ \sigma_1 \\ \sum_{a \in A_J} v[\sigma'_1, (J, a)] &= v[\sigma'_1, p_J] \quad \forall \sigma'_1 \succ \sigma_1, J \in \mathcal{I}_2 \text{ s.t. } \sigma'_1 \bowtie J, p_J \succ \sigma_2. \end{aligned}$$

Proof. By induction.

Base case The base case is any (σ_1, σ_2) such that there is no $\sigma'_1 \succ \sigma_1, \sigma'_2 \succ \sigma_2, \sigma'_1 \bowtie \sigma'_2$. In that case, the set B is empty, so the algorithm terminates immediately without modifying the sets \mathcal{S} and \mathcal{D} . Consequently, (Post1) and (Post2) hold trivially from (Pre1) and (Pre2). (Post3) reduces to an empty set of constraints, so (Post3) holds as well.

Inductive step In order to use the inductive hypothesis, we will need to prove that the preconditions for Decompose hold on Lines 15 and 25. We will focus on Line 15 ($I \in \mathcal{I}_1$), as the analysis for the other case ($I \in \mathcal{I}_2$) is symmetric. (Pre1) clearly holds, since we always update \mathcal{S} and \mathcal{D} in tandem. Since all iterations of the **for** loop on Line 10 touch different information sets, at the time of the recursive call on Line 15, and given (Post2) for all previous recursive calls, the only relevant sequence pairs (σ''_1, σ''_2) such that $\sigma''_1 \succ \sigma'_1, \sigma''_2 \succ \sigma_2$ that have been filled are the ones on Lines 12 and 13. So, (Pre2) holds.

We now check that the preconditions for FillOutRow hold at Line 20. (Pre1), (Pre2), (Pre3), and (Pre4) are trivial. (Pre5) and (Pre7) are guaranteed by (Post2) and (Post3) of Decompose applied to Line 15. (Pre6) holds because of Lines 18 and 19.

Using the inductive contracts of FillOutRow, FillOutColumn and Decompose for the recursive calls, we now show that all postconditions hold at the end of the call. (Post1) is trivial since we always update S and \mathcal{D} together. (Post2) holds by keeping track of what entries are filled in Lines 12, 13, 18, 19, 22, 23, 28, 29, as well as those filled in the calls to FillOutRow, FillOutColumn and Decompose, as regulated by postcondition (Post2) in the inductive contracts of the functions. In order to verify (Post3), we need to verify that the constraints that are not already guaranteed by the recursive calls hold. In particular, we need to verify that

$$\begin{aligned} \textcircled{A} \quad & \sum_{a \in A_I} v[(I, a), \sigma_2] = v[\sigma_1, \sigma_2] \quad \forall I \in \mathcal{I}_1 \text{ s.t. } \sigma_2 \bowtie I, p_I = \sigma_1, I \notin B \\ \textcircled{B} \quad & \sum_{a \in A_J} v[\sigma_1, (J, a)] = v[\sigma_1, \sigma_2] \quad \forall J \in \mathcal{I}_2 \text{ s.t. } \sigma_1 \bowtie J, p_J = \sigma_2, J \notin B. \end{aligned}$$

We will show that constraints \textcircled{A} hold; the proof for \textcircled{B} is symmetric. Using [Lemma 9.5](#) together with the definition of B (Lines 1-9), any information set $I \in \mathcal{I}_i : p_I = \sigma_i, \sigma_{-i} \bowtie I$ that is not in B must have σ_{-i} -rank *exactly* 1. Let $I \in \mathcal{I}_1$ be such that $\sigma_2 \bowtie I, p_I = \sigma_1, I \notin B$, as required in \textcircled{A} . Since the σ_2 -rank of I is 1, let J be the only information set in \mathcal{I}_2 such that $I \rightleftharpoons J, p_J = \sigma_2$. Note that $J \in B$. The entries $v[(I, a), \sigma_2] : a \in A_I$ were filled in Lines 28 and 29 when the **for** loop picked up $J \in B$. So, in particular,

$$v[(I, a), \sigma_2] = \sum_{a' \in A_J} v[(I, a), (J, a')] \quad \forall a \in A_I.$$

Summing the above equations across $a \in A_I$, we obtain

$$\begin{aligned} \sum_{a \in A_I} v[(I, a), \sigma_2] &= \sum_{a \in A_I} \sum_{a' \in A_J} v[(I, a), (J, a')] \\ &= \sum_{a' \in A_J} \sum_{a \in A_I} v[(I, a), (J, a')] \\ &= \sum_{a' \in A_J} v[\sigma_1, (J, a')] \\ &= v[\sigma_1, \sigma_2], \end{aligned}$$

where the last equation follows from the way the entries $v[\sigma_1, (J, a')] : a' \in A_J$ were filled in (Lines 22 and 23). This shows that the set of constraints \textcircled{A} hold. \square

We are now ready to conclude the proof of correctness for the decomposition algorithm.

Theorem 9.3. The von Stengel-Forges polytope \mathcal{V} of a two-player perfect-recall triangle-free EFG can be expressed via a sequence of scaled extensions with simplexes and singleton sets:

$$\mathcal{V} = \{1\} \overset{h_1}{\triangleleft} \mathcal{X}_1 \overset{h_2}{\triangleleft} \mathcal{X}_2 \overset{h_3}{\triangleleft} \cdots \overset{h_n}{\triangleleft} \mathcal{X}_n, \quad (9.4)$$

where, for $i = 1, \dots, n$, either $\mathcal{X}_i = \Delta^{s_i}$ for some simplex dimension $s_i \in \mathbb{N}$, or $\mathcal{X}_i = \{1\}$, and h_i is a linear function. Furthermore, an exact algorithm exists to compute such expression in linear time in the dimensionality of \mathcal{V} , and so, in time at most quadratic in the size of the game.

Proof. The correctness of the algorithm follow from (Post3) in the inductive contract. Every time the set of partially-filled-in vectors \mathcal{D} gets extended, it is extended with either the singleton set $\{1\}$ or a simplex. In either case the nonnegative affine functions h used are linear. So, the decomposition structure is as in the statement. Finally, since the overhead of each call (on top of the recursive calls) is linear in the number of relevant sequence pairs $(\sigma, \tau) \in \Sigma_1 \boxtimes \Sigma_2$ that are filled, and each relevant sequence pair is filled only once, the complexity of the algorithm is linear in the number of relevant sequence pairs. \square

9.4.5 Integrality of the vertices of \mathcal{V} in triangle-free games

The scaled-based decomposition of \mathcal{V} can be used to conclude the integrality of the vertices of \mathcal{V} , by leveraging the following analytical result about the scaled extension operation.

Lemma 9.8. Let \mathcal{X}, \mathcal{Y} , and h be as in [Definition 4.1](#). If \mathcal{X} is a convex polytope with vertices $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, and \mathcal{Y} is a convex polytope with vertices $\{\mathbf{y}_1, \dots, \mathbf{y}_m\}$, then $\mathcal{X} \overset{h}{\triangleleft} \mathcal{Y}$ is a convex polytope whose vertices are a nonempty subset of $\{(\mathbf{x}_i, h(\mathbf{x}_i)\mathbf{y}_j) : i \in \{1, \dots, n\}, j \in \{1, \dots, m\}\}$.

Proof. Take any point $z \in \mathcal{X} \overset{h}{\triangleleft} \mathcal{Y}$. By definition of scaled, extension, there exist $\mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$ such that $z = (\mathbf{x}, h(\mathbf{x})\mathbf{y})$. Since $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ are the vertices of \mathcal{X} , \mathbf{x} can be written as a convex combination $\mathbf{x} = \sum_{i=1}^n \lambda_i \mathbf{x}_i$ where $(\lambda_1, \dots, \lambda_n) \in \Delta^n$. Similarly, $\mathbf{y} = \sum_{j=1}^m \mu_j \mathbf{y}_j$ for some $(\mu_1, \dots, \mu_m) \in \Delta^m$. Hence, using the hypothesis that h is affine, we can write

$$z = (\mathbf{x}, h(\mathbf{x})\mathbf{y}) = \left(\sum_{i=1}^n \lambda_i \mathbf{x}_i, h \left(\sum_{i=1}^n \lambda_i \mathbf{x}_i \right) \sum_{j=1}^m \mu_j \mathbf{y}_j \right)$$

$$= \left(\sum_{i=1}^n \lambda_i \mathbf{x}_i, \left(\sum_{i=1}^n \lambda_i h(\mathbf{x}_i) \right) \sum_{j=1}^m \mu_j \mathbf{y}_j \right) = \sum_{i=1}^n \sum_{j=1}^m \lambda_i \mu_j (\mathbf{x}_i, h(\mathbf{x}_i) \mathbf{y}_j).$$

Since $\lambda_i \mu_j \geq 0$ for all $i \in \llbracket n \rrbracket, j \in \llbracket m \rrbracket$ and

$$\sum_{i=1}^n \sum_{j=1}^m \lambda_i \mu_j = \left(\sum_{i=1}^n \lambda_i \right) \left(\sum_{j=1}^m \mu_j \right) = 1,$$

we conclude that $\mathbf{z} \in \text{co}\{(\mathbf{x}_i, h(\mathbf{x}_i) \mathbf{y}_j) : i \in \llbracket n \rrbracket, j \in \llbracket m \rrbracket\}$. On the other hand, $(\mathbf{x}_i, h(\mathbf{x}_i) \mathbf{y}_j) \in \mathcal{X} \overset{h}{\triangleleft} \mathcal{Y}$, so

$$\mathcal{X} \overset{h}{\triangleleft} \mathcal{Y} = \text{co} \left\{ (\mathbf{x}_i, h(\mathbf{x}_i) \mathbf{y}_j) : i \in \llbracket n \rrbracket, j \in \llbracket m \rrbracket \right\}.$$

Since the vertices of a (nonempty) polytope are a (nonempty) subset of any convex basis for the polytope, the vertices of $\mathcal{X} \overset{h}{\triangleleft} \mathcal{Y}$ must be a nonempty subset of $\{(\mathbf{x}_i, h(\mathbf{x}_i) \mathbf{y}_j) : i \in \llbracket n \rrbracket, j \in \llbracket m \rrbracket\}$, which is the statement. \square

In particular, by applying [Lemma 9.8](#) inductively on the structure of the scaled-extension-based structural decomposition of \mathcal{V} , we obtain the following theorem.

Theorem 9.4. Let \mathcal{V} be the von Stengel-Forges polytope of a two-player triangle-free game ([Definition 9.5](#)). All vertices of \mathcal{V} have integer $\{0, 1\}$ coordinates.

Proof. We prove the statement by induction over the scaled-extension-based decomposition

$$\mathcal{V} = \{1\} \overset{h_1}{\triangleleft} \mathcal{X}_1 \overset{h_2}{\triangleleft} \dots \overset{h_n}{\triangleleft} \mathcal{X}_n.$$

In particular, we will show that for all $k = 0, \dots, n$, the coordinates of the vertices of the polytope

$$\mathcal{V}_k = \{1\} \overset{h_1}{\triangleleft} \dots \overset{h_k}{\triangleleft} \mathcal{X}_k$$

constructed by considering only the first k scaled extensions in the decomposition are all integer. Since $\mathcal{V} \subseteq [0, 1]^{\Sigma_1 \boxtimes \Sigma_2}$ ([Remark 9.1](#)), this immediately implies that each coordinate is in $\{0, 1\}$.

Base case: $k = 0$. In this case, $\mathcal{V}_0 = \{1\}$. The only vertex is $\{1\}$, which is integer. So, base case trivially holds.

Inductive step Suppose that the polytope \mathcal{V}_k ($k < n$) has integer vertices. We will show that the same holds for \mathcal{V}_{k+1} . Clearly, $\mathcal{V}_{k+1} = \mathcal{V}_k \stackrel{h_{k+1}}{\triangleleft} \mathcal{X}_{k+1}$. From the properties of the structural decomposition, we know that \mathcal{K}_{k+1} is either the singleton $\{1\}$, or a probability simplex $\Delta^{s_{k+1}}$ for some appropriate dimension s_{k+1} . We break the analysis accordingly.

- If $\mathcal{X}_{k+1} = \{1\}$, the scaled extension represents filling in a linearly-dependent entry in $v \in \mathcal{V}$ by summing already-filled-in entries. So, h_{k+1} takes a partially-filled-in vector from \mathcal{V}_k and sums up some of its coordinates. Let v_1, \dots, v_n be the vertices of \mathcal{V}_k . Using Lemma 9.8, the vertices of \mathcal{V}_{k+1} are a subset of

$$\{(v_i, h(v_i) \cdot 1) : i = 1, \dots, n\}. \quad (9.5)$$

Since by inductive hypothesis v_i have integer coordinates, and h sums up some of them, $h(v)_i$ is integer for all $i = 1, \dots, n$. So, all of the vectors in (9.5) have integer coordinates, and in particular this must be true of the vertices of \mathcal{V}_{k+1} .

- If $\mathcal{X}_{k+1} = \Delta^{s_{k+1}}$, the scaled extension represents the operation of partitioning an already-filled-in entry $v[\sigma, \tau]$ of \mathcal{V}_k into s_i non-negative real values. The affine function h_{k+1} extracts the entry $v[\sigma, \tau]$ from each vector $v \in \mathcal{V}_k$. Let v_1, \dots, v_n be the vertices of \mathcal{V}_k . The vertices of $\Delta^{s_{k+1}}$ are the canonical basis vectors $\{e_1, \dots, e_{s_{k+1}}\}$. From Lemma 9.8, the vertices of \mathcal{V}_{k+1} are a subset of

$$\begin{aligned} & \{(v_i, h(v_i)e_j) : i = 1, \dots, n, j = 1, \dots, s_{k+1}\} \\ & = \{(v_i, v_i[\sigma, \tau]e_j) : i = 1, \dots, n, j = 1, \dots, s_{k+1}\}. \end{aligned} \quad (9.6)$$

Since by inductive hypothesis the vertices v_i have integer coordinates, $v_i[\sigma, \tau]$ is an integer. Since the canonical basis vector only have entries in $\{0, 1\}$, all of the vectors in (9.6) have integer coordinates. So, in particular, this must be true of the vertices of \mathcal{V}_{k+1} . \square

Finally, combining Theorem 9.4 and Theorem 9.1, we obtain the central theorem of this chapter.

Theorem 9.5. In a two-player perfect-recall imperfect-information extensive-form game that satisfies the triangle-freeness condition (Definition 9.5), the polytope of correlation plans coincides with the von Stengel-Forges polytope. Consequently, an optimal EFCE can be computed in polynomial time (in the size of the input imperfect-information extensive-form game) in two-player triangle-free games.

9.5 Beyond triangle-freeness

The result established in [Theorem 9.3](#) shows that, in certain games, the polytope of correlation plans can be expressed as a polynomially-sized chain of scaled extension operations. This characterization implies that an optimal EFCE can be found in polynomial time, and, as we will show in [Chapter 10](#), it also enables the construction of practical no-regret algorithms via the formalism of regret circuits ([Chapter 4](#)). In work following the introduction of the result, B. H. Zhang, Farina, Celli, and Sandholm (2022) showed that a characterization of Ξ based on scaled extension exists in any game, though it might not be of polynomial size.^[9.a] To establish that result, which will not be discussed in detail in this dissertation, we used techniques from the theory of tree decompositions to construct a polytope of dimension typically exponential in the input game tree size, whose vertices are guaranteed to map onto the vertices of the polytope of correlation plans via a linear projection. We remark that the length of the chain of scaled extension operations that define the polytope whose projection is Ξ is exponential in a parameter that intuitively represent the amount of uncommon information between the players, thereby yielding the current state-of-the-art parameterized complexity results for the computation of optimal EFCE in general imperfect-information extensive-form games.

We refer the interested reader to the paper by B. H. Zhang, Farina, Celli, and Sandholm (2022) for the result, and to prior work by B. H. Zhang, Farina, and Sandholm (2023) and B. H. Zhang and Sandholm (2022a) as a gentle introduction on how the machinery of tree decomposition can be applied in correlated strategy spaces, and how that relates to scaled extension.

9.6 Experimental investigation of utilities reached by EFCE

The characterization of the polytope of correlation plans given in this chapter enables visualizing, for the first time, the polytope of all expected utility profiles that can be reached by EFCEs. Indeed, recall that the expected utility for any player is a linear function of the correlation plan ξ . Hence, set of all utilities reachable by EFCE is a linear transformation of the polytope of EFCE defined by constraints ①-④ in [Proposition 9.1](#), and can thus be computed efficiently.

In the next two subsections, we use the characterization of the polytope of correlation plans to investigate empirically the set of expected utilities that can be reached by EFCE in game instances drawn from a variety of standard parametric classes, identified with an alphabetical mnemonic: **B** – battleship, **D** – liar’s dice, **K** – Kuhn poker, **L** – Leduc poker, **RS** – ridesharing game, **S** – sheriff, **T** – three-player tricks game, **TP** – double dummy bridge game. As in the other chapters, a full description of the games is available in [Appendix A](#).

^[9.a]As discussed, the exponential size cannot be avoided, unless $P = NP$. See also von Stengel and Forges (2008).

9.6.1 Two-player general-sum games

Figure 9.3 shows the polytopes of expected utilities that can be reached by EFCE in three two-player general-sum games. We observe that both the shape of the polytope and the range of reachable payoffs is highly nontrivial. This empirically suggests that being able to search and optimize over the space of EFCE (rather than computing a single EFCE without *a priori* guarantees over the social welfare, as in Chapter 8) is important.

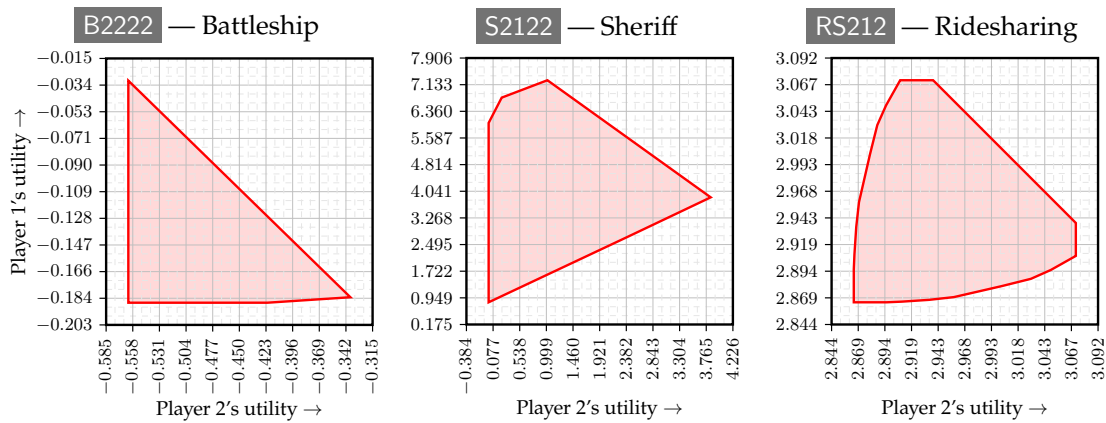


Figure 9.3: Polytope of expected utilities for the players that can be reached via extensive-form correlated equilibria in three standard two-player general-sum imperfect-information extensive-form games.

9.6.2 Three-player zero-sum games

Figure 9.3 shows the polytopes of expected utilities that can be reached by EFCE across six three-player zero-sum games. We remark that despite the appearance of curved boundaries, the set of payoffs reachable by EFCE, being the linear projection of a polytope with finitely many facets, must itself be a polytope defined by finitely many facets. The experimental data confirms the conclusion we drew in the case of two-player games (Figure 9.3): both the shape of the polytope and the range of reachable payoffs is highly nontrivial. Once again, this empirically suggests that being able to search and optimize over the space of EFCE (rather than computing a single EFCE without *a priori* guarantees over the social welfare, as in Chapter 8) is important.

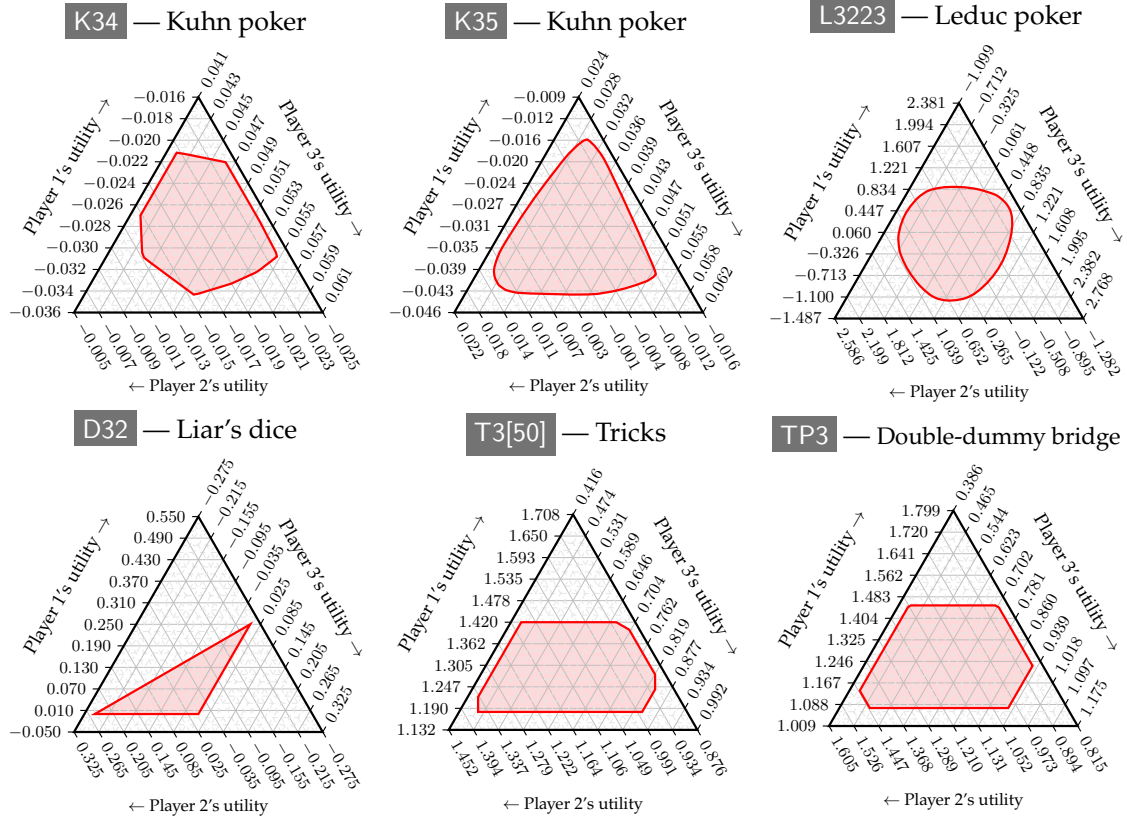


Figure 9.4: Polytope of expected utilities for the players that can be reached via extensive-form correlated equilibria in three standard three-player zero-sum imperfect-information extensive-form games.

9.A Appendix: Additional lemmas on the structure of \mathcal{V}

Lemma 9.9. Let $v \in \mathcal{V}$. For all $\sigma_1 \in \Sigma_1$ such that $v[\sigma_1, \emptyset] = 0$, $v[\sigma_1, \sigma_2] = 0$ for all $\sigma_2 \bowtie \sigma_1$. Similarly, for all $\sigma_2 \in \Sigma_2$ such that $v[\emptyset, \sigma_2] = 0$, $v[\sigma_1, \sigma_2] = 0$ for all $\sigma_1 \bowtie \sigma_2$.

Proof. We prove the theorem by induction on the *depth* of the sequences σ_1 and σ_2 . The depth $\text{depth}(\sigma)$ of a generic sequence $\sigma = (I, a) \in \Sigma_i$ of Player i is defined as the number of actions that Player i plays on the path from the root of the tree down to action a at information set I included. Conventionally, we let the depth of the empty sequence be 0.

Take $\sigma_1 \in \Sigma_1$ such that $v[\sigma_1, \emptyset] = 0$. For σ_2 of depth 0 (that is, $\sigma_2 = \emptyset$), clearly $v[\sigma_1, \sigma_2] = 0$. For the inductive step, suppose that $v[\sigma_1, \sigma_2] = 0$ for all $\sigma_2 \in \Sigma_2, \sigma_1 \bowtie \sigma_2$ such that $\text{depth}(\sigma_2) \leq d_2$. We will show that $v[\sigma_2, \sigma_2] = 0$ for $\text{depth}(\sigma_2) \leq d_2 + 1$. Indeed, let $(I, a') = \sigma_2 \bowtie \sigma_1$ of depth $d_2 + 1$. Since $v \in \mathcal{V}$, in particular the von Stengel-Forges constraint $\sum_{a \in A_I} v[\sigma_1, (I, a)] = v[\sigma_1, p_I]$ must hold. The depth of p_I is d_2 , so by the inductive hypothesis, it must be $v[\sigma_1, p_I] = 0$, and therefore $\sum_{a \in A_I} v[\sigma_1, (I, a)] = 0$. But all entries of v are nonnegative, so it must be $v[\sigma_1, (I, a)] = 0$ for all $a \in A_I$, and in particular for $(I, a') = \sigma_2$. This completes the proof by induction.

The proof for the second part is analogous. \square

Lemma 9.10. Let $v \in \mathcal{V}$ have integer $\{0, 1\}$ coordinates. Then, for all $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$, it holds that

$$v[\sigma_1, \sigma_2] = v[\sigma_1, \emptyset] \cdot v[\emptyset, \sigma_2].$$

Proof. We prove the theorem by induction on the depth of the sequences, similarly to [Lemma 9.9](#).

The base case for the induction proof corresponds to the case where σ_1 and σ_2 both have depth 0, that is, $\sigma_1 = \sigma_2 = \emptyset$. In that case, the theorem is clearly true, because $v[\emptyset, \emptyset] = 1$ as part of the von Stengel-Forges constraints ([Definition 9.3](#)).

Now, suppose that the statement holds as long as $\text{depth}(\sigma_1), \text{depth}(\sigma_2) \leq d$. We will show that the statement will hold for any $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$ such that $\text{depth}(\sigma_1), \text{depth}(\sigma_2) \leq d + 1$. Indeed, consider $(\sigma_1, \sigma_2) \in \Sigma_1 \bowtie \Sigma_2$ such that $\text{depth}(\sigma_1), \text{depth}(\sigma_2) \leq d + 1$. If any of the sequences is the empty sequence, the statements holds trivially, so assume that neither is the empty sequence and in particular $\sigma_1 = (I, a), \sigma_2 = (J, b)$. If $v[\sigma_1, \emptyset] = 0$, then from [Lemma 9.9](#) $v[\sigma_1, \sigma_2] = 0$ and the statement holds. Similarly, if $v[\emptyset, \sigma_2] = 0$, then $v[\sigma_1, \sigma_2] = 0$, and the statement holds. Hence, the only remaining case given the integrality assumption on the coordinates of v is $v[\sigma_1, \emptyset] = v[\emptyset, \sigma_2] = 1$.

From the von Stengel-Forges constraints,

$$v[p_I, \emptyset] = \sum_{a' \in A_I} v[(I, a'), \emptyset] = 1 + \sum_{a' \in A_I, a' \neq a} v[(I, a'), \emptyset] \geq 1.$$

Hence, because all entries of v are in $\{0, 1\}$, it must be $v[p_I, \emptyset] = 1$ and $v[(I, a'), \emptyset] = 0$ for all $a' \in A_I, a' \neq a$. With a similar argument we conclude that $v[\emptyset, p_J] = 1$ and $v[\emptyset, (J, b')] = 0$ for all $b' \in A_J, b \neq b'$. Using the inductive hypothesis,

$$v[p_I, p_J] = v[p_I, \emptyset] \cdot v[\emptyset, p_J] = 1.$$

Now, using the von Stengel-Forges constraints together with the equality $v[p_I, p_J] = 1$ we just proved, we conclude that

$$\sum_{a' \in A_I} \sum_{b' \in A_J} v[(I, a'), (J, b')] = 1. \quad (9.7)$$

On the other hand, since $v[(I, a'), \emptyset] = 0$ for all $a' \in A_I, a' \neq a$ and $v[\emptyset, (J, b')] = 0$ for all $b' \in A_J, b' \neq b$, from [Lemma 9.9](#) we have that

$$a' \neq a \vee b' \neq b \implies v[(I, a'), (J, b')] = 0. \quad (9.8)$$

From (9.8) and (9.7), we conclude that $v[(I, a), (J, b)] = v[\sigma_1, \sigma_2] = 1 = v[\sigma_1, \emptyset] \cdot v[\emptyset, \sigma_2]$, as we wanted to show. \square

9.B Appendix: Optimal EFCE as a linear program

In order to express the utility of a trigger agent, it is necessary to compute the probability of the game ending in each of the terminal states. To do that, for the purposes of this appendix we introduce the following additional notation:

- $\Pi_i(I)$, is the set of reduced-normal-form strategies that can lead to information set I (which belongs to Player i) assuming that the other player acts to do so as well. This is equivalent (assuming no zero-chance nodes or disconnected game trees) to saying that all reduced-normal-form strategies in $\Pi_i(I)$ have *some* action which belongs to information set I . Formally,

$$\Pi_i(I) := \{\pi \in \Pi_i : \pi[p_I] = 1\}.$$

- $\Pi_i(Ia)$ is the set of reduced normal form strategies which will lead to information set I and recommend the action a at I . This is equivalent to the set of reduced normal form strategies which contain a as part of their recommendation (this set is typically a subset of $\Pi_i(Ia)$). Formally,

$$\Pi_i(Ia) := \{\pi \in \Pi_i : \pi[Ia] = 1\}.$$

- $\sigma_i(z)$ is the last sequence belonging to Player i on the path from the roof of the game tree to the terminal state $z \in \mathcal{Z}$.
- $\Pi_i(z)$ is the set of reduced-normal-form strategies which can lead to the terminal state z (assuming the other player players to do so). This is equivalent to $\Pi_i(\sigma_i(z))$.

For simplicity, we focus on the two-player case; generalizing to the general case is straightforward. Consider the random variable $t_{\hat{\sigma}} : \Pi_1 \times \Pi_2 \times \Pi_1(\hat{I}) \rightarrow \mathcal{Z}$ that maps a triple of reduced-normal-form strategies $(\pi_1, \pi_2, \hat{\pi}_1)$ to the terminal state of the game that is reached when Player 1 is a $\hat{\sigma}$ -trigger agent and Player 2 follows all recommendations. We are interested in expressing the probability of terminating at each $z \in \mathcal{Z}$ for a $\hat{\sigma}$ -trigger agent, given the mediator's joint distribution μ over reduced normal form strategies and the trigger strategy $\hat{\mu}$ for the deviating player, which we will assume to be Player 1 without loss of generality. For each trigger $\hat{\sigma}$, the terminal leaves may be partitioned into the following 3 sets.

- $\mathcal{Z}_{\hat{\sigma}}$ (or equivalently $\mathcal{Z}_{\hat{I}, \hat{\sigma}}$) is the set of terminal nodes that are descendants of the trigger $\hat{\sigma} = (\hat{I}, \hat{\sigma})$. In order for the game to end in one of these terminal nodes, it is necessary that the recommendation device recommended to Player 1 the trigger sequence $\hat{\sigma}$, and therefore the agent must have deviated. Furthermore, Player 2 must have been recommended the terminal sequence $\sigma_2(z)$ corresponding to the terminal state, and finally $\hat{\pi}_1$ must be compatible with $\sigma_1(z)$. We can capture all these constraints concisely by saying that the sampled $(\pi_1, \pi_2, \hat{\pi}_1)$ must be such that $\pi_1 \in \Pi_1(\hat{\sigma})$, $\pi_2 \in \Pi_2(z)$ and $\hat{\pi}_1 \in \Pi_1(z)$. Therefore the probability that a $\hat{\sigma}$ trigger agent terminates at some $z \in \mathcal{Z}_{\hat{\sigma}}$ is given by,

$$\mathbb{P}_{\mu, \hat{\mu}}[t_{\hat{\sigma}} = z \in \mathcal{Z}_{\hat{\sigma}}] = \left(\sum_{\substack{\pi_1 \in \Pi_1(\hat{\sigma}) \\ \pi_2 \in \Pi_2(z)}} \mu(\pi_1, \pi_2) \right) \left(\sum_{\hat{\pi}_1 \in \Pi_1(z)} \hat{\mu}_1(\hat{\pi}_1) \right),$$

where the first term in the product is the probability that Player 2 plays to z and Player 1 gets triggered, and the second term is the probability that the deviation strategy from Player 1 upon getting triggered is one that reaches z .

- $\mathcal{Z}_{\hat{I}}$ is the set of terminal states that are descendant of any sequence in \hat{I} , *except* $\hat{\sigma}$. In order for the game to reach this terminal state, recommendations issued to Player 1 by the correlation device must have been such that Player 1 reached \hat{I} . There are two cases: either the correlation device recommended $\hat{\sigma}$ at \hat{I} , or it did not. In the former case, Player 1 started deviating (using the sampled reduced-normal-form plan $\hat{\pi}_1$); hence, in this case it must be $\hat{\pi}_1 \in \Pi_1(z)$. In the latter case, Player 1 does not deviate from the recommendation, and therefore it must be $\pi_1 \in \Pi_1(z)$. Either way, Player 2 must have been recommended the terminal sequence z corresponding to the terminal state z ; that is, $\pi_2 \in \Pi_2(z)$. Collecting all these constraints, it must be

$$(\pi_1, \pi_2, \hat{\pi}_1) \in \Pi_1(\hat{\sigma}) \times \Pi_2(z) \times \Pi_1(z) \cup \Pi_1(z) \times \Pi_2(z) \times \Pi_1(\hat{I}).$$

Using the fact that the two cases as to whether or not Player 1 was recommended $\hat{\sigma}$ or not at \hat{I} are disjoint, we can write

$$\mathbb{P}_{\mu, \hat{\mu}}[t_{\hat{\sigma}} = z \in \mathcal{Z}_{\hat{I}}] = \left(\sum_{\substack{\pi_1 \in \Pi_1(\hat{\sigma}) \\ \pi_2 \in \Pi_2(z)}} \mu(\pi_1, \pi_2) \right) \left(\sum_{\hat{\pi}_1 \in \Pi_1(z)} \hat{\mu}_1(\hat{\pi}_1) \right) + \left(\sum_{\substack{\pi_1 \in \Pi_1(z) \\ \pi_2 \in \Pi_2(z)}} \mu(\pi_1, \pi_2) \right).$$

The first term in the summation may be understood as the probability that the agent was triggered and its deviation was to play something other than $\hat{\sigma}$. The second term is that probability that the agent was not triggered and the game simply terminates at z based on μ .

- Finally, $\mathcal{Z}_{-\hat{I}}$ is the set of terminal nodes that are neither in $\mathcal{Z}_{\hat{\sigma}}$ nor in $\mathcal{Z}_{\hat{I}}$. If the game has ended in any terminal state that belongs to $\mathcal{Z}_{-\hat{I}}$, Player 1 has not deviated from the recommended strategy, since they have never even reached the trigger information set, \hat{I} . Hence, in this case it must be $(\pi_1, \pi_2) \in \Pi_1(z) \times \Pi_2(z)$. Hence,

$$\mathbb{P}_{\mu, \hat{\mu}}[t_{\hat{\sigma}} = z \in \mathcal{Z}_{-\hat{I}}] = \sum_{\substack{\pi_1 \in \Pi_1(z) \\ \pi_2 \in \Pi_2(z)}} \mu(\pi_1, \pi_2).$$

With the above, we can finally express the constraint that no deviation strategy $\hat{\mu}$ can lead to a higher utility for Player 1 than simply following each recommendation. Indeed, for all $\hat{\mu}$, the utility of the trigger agent is expressed as

$$\sum_{z \in \mathcal{Z}} u_1(z) \mathbb{P}_{\mu, \hat{\mu}}[t_{\hat{\sigma}} = z],$$

where the correct expression for $\mathbb{P}_{\mu, \hat{\mu}}[t_{\hat{\sigma}} = z]$ must be selected depending on whether $z \in \mathcal{Z}_{\hat{\sigma}}$, $z \in \mathcal{Z}_{\hat{I}}$ or $z \in \mathcal{Z}_{-\hat{I}}$. On the other hand, the utility of an agent that follows all recommendations is

$$\sum_{z \in \mathcal{Z}} u_1(z) \mathbb{P}_{\mu, \hat{\mu}}[\pi_1 \in \Pi_1(z), \pi_2 \in \Pi_2(z)] = \sum_{z \in \mathcal{Z}} \left(u_1(z) \sum_{\substack{\pi_1 \in \Pi_1(z) \\ \pi_2 \in \Pi_2(z)}} \mu(\pi_1, \pi_2) \right).$$

Therefore, following all recommendations is a best response for the $\hat{\sigma}$ -trigger agent if and only if μ is chosen so that

$$\sum_{z \in \mathcal{Z}} u_1(z) \left(\mathbb{P}_{\mu, \hat{\mu}}[t_{\hat{\sigma}} = z] - \sum_{\substack{\pi_1 \in \Pi_1(z) \\ \pi_2 \in \Pi_2(z)}} \mu(\pi_1, \pi_2) \right) \leq 0 \quad \forall \hat{\mu} \in \Delta^{\Pi_1(\hat{I})}. \quad (9.9)$$

The crucial observation is that all the probabilities $\mathbb{P}_{\mu, \hat{\mu}}[t = z]$ defined above can be expressed via the following quantities:

$$y_{1,\hat{\sigma}}(z) := \sum_{\hat{\pi}_1 \in \Pi_1(z)} \hat{\mu}_1(\hat{\pi}_1) \quad \forall z \in \mathcal{Z}; \quad \xi_1(\sigma_1; z) := \sum_{\substack{\pi_1 \in \Pi_1(\sigma_1) \\ \pi_2 \in \Pi_2(z)}} \mu(\pi_1, \pi_2) \quad \forall \sigma_1 \in \Sigma_1, z \in \mathcal{Z}.$$

For example, for all $z \in \mathcal{Z}_{\hat{f}}$ we can write

$$\mathbb{P}_{\mu, \hat{\mu}}[t_{\hat{\sigma}} = z] = \xi_1(\hat{\sigma}; z) y_{1,\hat{\sigma}}(z) + \xi_1(\sigma_1(z); z).$$

When deviations relative to Player 2 are brought into the picture, the following two sets of symmetric quantities also become relevant:

$$y_{2,\hat{\sigma}}(z) := \sum_{\hat{\pi}_2 \in \Pi_2(z)} \hat{\mu}_2(\hat{\pi}_2) \quad \forall z \in \mathcal{Z}; \quad \xi_2(\sigma_2; z) := \sum_{\substack{\pi_1 \in \Pi_1(z) \\ \pi_2 \in \Pi_2(\sigma_2)}} \mu(\pi_1, \pi_2) \quad \forall \sigma_2 \in \Sigma_2, z \in \mathcal{Z}.$$

It is now natural to perform a change of variables, and pick (correlated) distributions for the random variables $y_{1,\hat{\sigma}}(\cdot)$, $y_{2,\hat{\sigma}}(\cdot)$, $\xi_1(\cdot; \cdot)$ and $\xi_2(\cdot; \cdot)$ instead of μ , $\hat{\mu}_1$ and $\hat{\mu}_2$. Since there are only a polynomial number (in the game tree size) of combinations of arguments for these new random variables, this approach would allow one to remove the redundancy of realization-equivalent normal-form plans and focus on a polynomially-small search space. In the case of the random variables $y_{1,\hat{\sigma}}$ and $y_{2,\hat{\sigma}}$, it is clear that the change of variables is possible via the sequence form (von Stengel, 2002). Therefore, the only difficulty is in characterizing the space spanned by ξ_1 and ξ_2 as μ varies across the probability simplex, which has been the focus of the present chapter.

Chapter 10

Learning optimal extensive-form correlated equilibria

10.1 Contributions

In this chapter, we propose the first no-regret learning algorithm that guarantees convergence to an optimal EFCE. Specifically, we show that computing optimal EFCE in multiplayer imperfect-information extensive-form games can be cast as solving a bilinear saddle-point problem. This unlocks applying rich technology, both theoretical and experimental, developed so far for computing bilinear saddle points (for example, Nash equilibria in zero-sum games) for the more challenging—and much less understood—problem of computing optimal equilibria.

To establish the result, we build on the structural understanding of the polytope of correlation plans Ξ , which we developed in [Chapter 9](#), to cast the computation of optimal EFCE as a max-min equilibrium over two heavily structured domains: Ξ on the one side, and a Cartesian product of rescaled sequence-form strategy polytopes on the other. By then leveraging the machinery of regret circuits, together with the insight regarding the scaled-extension-based decomposition of Ξ , we are able to construct learning algorithms that are able to compute EFCE at significantly faster and at a significantly larger scale than the best prior approach based on linear programming ([Proposition 9.1](#)).

We experimentally evaluate our learning-based method on 23 game instances from 8 different classes of established parametric benchmark games. Our experiments once again confirm the conclusion that learning dynamics form the basis for some of the most scalable technique available today to compute equilibrium points in large games, and computation of optimal EFCE in imperfect-information extensive-form games is no exception.

Beyond computational advances, we remark that our bilinear formulation also allows us to obtain *last-iterate* guarantees to the optimal equilibria when using predictive online gradient

descent as the learning algorithm, instead of the time-average guarantees traditionally derived within the no-regret framework.

10.2 Optimal EFCE as a bilinear saddle-point problem

In this section we show that the problem of computing an optimal (for example, social welfare maximizing) EFCE can be converted to the problem of finding a Nash equilibrium in a two-player zero-sum game. This realization is important algorithmically, in that it will enable us to tap into the richness of approaches studied in [Part II](#) of this dissertation to compute an optimal EFCE at scale, for the first time.

Our approach relies on a Lagrangian relaxation of the optimization problem that was given in [Lemma 9.2](#), reproduced below for convenience:

$$\left\{ \begin{array}{ll} \max_{\boldsymbol{\xi}} & \mathbf{c}^\top \boldsymbol{\xi} \\ \text{s.t.} & \textcircled{1} \quad \boldsymbol{\xi} \in \Xi \\ & \textcircled{2} \quad \max_{\mathbf{x}_{\hat{\sigma}} \in \mathcal{Q}_{i, \neq j}} \boldsymbol{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{\hat{\sigma}} \leq 0 \quad \forall i \in \llbracket n \rrbracket, \hat{\sigma} = (j, a) \in \Sigma_i^*. \end{array} \right. \quad (10.1)$$

We note that the feasible set of the program above does not change if constraint $\textcircled{2}$ is replaced by

$$\max \left\{ 0, \max_{\mathbf{x}_{\hat{\sigma}} \in \mathcal{Q}_{i, \neq j}} \boldsymbol{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{\hat{\sigma}} \right\} \leq 0 \quad \forall i \in \llbracket n \rrbracket, \hat{\sigma} = (j, a) \in \Sigma_i^*.$$

Hence, using the fact that $\max\{0, \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})\} = \max_{\mathbf{x} \in \text{co}\{\{\mathbf{0}\}, \mathcal{X}\}} f(\mathbf{x})$ for any linear function f and set \mathcal{X} , the following program is equivalent to [\(10.1\)](#):

$$\left\{ \begin{array}{ll} \max_{\boldsymbol{\xi}} & \mathbf{c}^\top \boldsymbol{\xi} \\ \text{s.t.} & \textcircled{1} \quad \boldsymbol{\xi} \in \Xi \\ & \textcircled{2} \quad \max_{\mathbf{x}_{\hat{\sigma}} \in \text{co}\{\{\mathbf{0}\}, \mathcal{Q}_{i, \neq j}\}} \boldsymbol{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{\hat{\sigma}} \leq 0 \quad \forall i \in \llbracket n \rrbracket, \hat{\sigma} = (j, a) \in \Sigma_i^*. \end{array} \right. \quad (10.2)$$

Letting now $\mathcal{Q}_{i, \hat{\sigma}} := \text{co}\{\{\mathbf{0}\}, \mathcal{Q}_{i, \neq j}\}$ for all $i \in \llbracket n \rrbracket$ and $\hat{\sigma} = (j, a) \in \Sigma_i^*$, the Lagrangian relaxation of the above program is the saddle-point problem

$$\max_{\boldsymbol{\xi} \in \Xi} \min_{\substack{\lambda \in \mathbb{R}_{\geq 0}, \\ \mathbf{x}_{i, \hat{\sigma}} \in \mathcal{Q}_{i, \hat{\sigma}} \quad \forall i, \hat{\sigma}}} \mathbf{c}^\top \boldsymbol{\xi} - \lambda \sum_{i \in \llbracket n \rrbracket} \sum_{\hat{\sigma} \in \Sigma_i^*} \boldsymbol{\xi}^\top \mathbf{A}_{\hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}}. \quad (10.3)$$

We first point out that the above saddle-point optimization problem admits a solution $(\boldsymbol{\xi}^*, \mathbf{x}^*, \lambda^*)$.

Proposition 10.1. The problem (10.3) admits a finite saddle-point solution (ξ^*, x^*, λ^*) . Moreover, for all fixed $\lambda > \lambda^*$, the problems (10.3) and (10.1) have the same value and same set of optimal solutions.

Proof. Let v be the optimal value of (10.2), which, as we argued, is equal to the optimal value of (10.1). The Lagrangian of (10.2) is

$$\max_{\xi \in \Xi} \min_{\substack{\lambda_{i,\hat{\sigma}} \in \mathbb{R}_{\geq 0}, \\ \mathbf{x}_{i,\hat{\sigma}} \in Q_{i,\hat{\sigma}}, \\ \forall i \in [n], \hat{\sigma} \in \Sigma_i^*}} \mathbf{c}^\top \xi - \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \lambda_{i,\hat{\sigma}} \xi^\top \mathbf{A}_{\hat{\sigma}} \mathbf{x}_{i,\hat{\sigma}}.$$

Now, making the change of variables $\bar{\mathbf{x}}_{i,\hat{\sigma}} := \lambda_{i,\hat{\sigma}} \mathbf{x}_{i,\hat{\sigma}}$, the above problem is equivalent to

$$\max_{\xi \in \Xi} \min_{\substack{\bar{\mathbf{x}}_{i,\hat{\sigma}} \in \bar{Q}_{i,\hat{\sigma}}, \\ \forall i \in [n], \hat{\sigma} \in \Sigma_i^*}} \mathbf{c}^\top \xi - \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \xi^\top \mathbf{A}_{\hat{\sigma}} \bar{\mathbf{x}}_{i,\hat{\sigma}}. \quad (10.4)$$

where $\bar{Q}_{i,\hat{\sigma}}$ is the conic hull of $Q_{i,\hat{\sigma}}$: $\bar{Q}_{i,\hat{\sigma}} := [0, +\infty)Q_{i,\hat{\sigma}}$. The problem (10.4) is a bilinear saddle-point problem, where Ξ is compact and convex and $\bar{Q}_{i,\hat{\sigma}}$ is convex. Thus, Sion's minimax theorem Sion, 1958 applies, and we have that the value of (10.4) is equal to the value of the problem

$$\min_{\substack{\bar{\mathbf{x}}_{i,\hat{\sigma}} \in \bar{Q}_{i,\hat{\sigma}}, \\ \forall i \in [n], \hat{\sigma} \in \Sigma_i^*}} \max_{\xi \in \Xi} \mathbf{c}^\top \xi - \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \xi^\top \mathbf{A}_{\hat{\sigma}} \bar{\mathbf{x}}_{i,\hat{\sigma}}. \quad (10.5)$$

Since this is a linear program (by taking a dual of the inner maximization problem) with a finite value, its optimum value must be achieved by some $\bar{\mathbf{x}} := (\dots, \bar{\mathbf{x}}_{i,\hat{\sigma}}, \dots) := (\dots, \lambda_{i,\hat{\sigma}} \mathbf{x}_{i,\hat{\sigma}}, \dots)$. Let $\lambda^* := \max_{i,\hat{\sigma}} \lambda_{i,\hat{\sigma}}$. Using the fact that $\mathbf{0} \in Q_{i,\hat{\sigma}} \subseteq \bar{Q}_{i,\hat{\sigma}}$ and clearly $\xi^\top \mathbf{A}_{i,\hat{\sigma}} \mathbf{0} = 0$ for all μ , the profile

$$\bar{\mathbf{x}}' := (\dots, \lambda^* \bar{\mathbf{x}}_{i,\hat{\sigma}}, \dots) \quad \text{where} \quad \mathbf{x}'_{i,\hat{\sigma}} := \mathbf{0} + \frac{\lambda_{i,\hat{\sigma}}}{\lambda^*} (\mathbf{x}_{i,\hat{\sigma}} - \mathbf{0})$$

is also an optimal solution of (10.5). Therefore, for any $\lambda \geq \lambda^*$, $\mathbf{x}' := (\mathbf{x}'_1, \dots, \mathbf{x}'_n)$ is an optimal solution for the minimizer in (10.3) that achieves the value of (10.1), so (10.1) and (10.3) have the same value.

Now take $\lambda > \lambda^*$, and suppose for contradiction that (10.3) admits some optimal $\mu \in \Xi$ that is not optimal in (10.1). Then, either $\mathbf{c}^\top \mu < v$, or μ violates some constraint $\max_{\mathbf{x}_{i,\hat{\sigma}}} \mu^\top \mathbf{A}_{i,\hat{\sigma}} \mathbf{x}_{i,\hat{\sigma}} \leq 0$. The first case is impossible because then setting $\mathbf{x}_{i,\hat{\sigma}} = \mathbf{0}$ for all i yields value less than v in (10.3). In the second case, since we know that (10.3) and (10.1) have the same value when

$\lambda = \lambda^*$, we have

$$\mathbf{c}^\top \boldsymbol{\xi} - \lambda \max_{\mathbf{x} \in X} \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \boldsymbol{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}} < \mathbf{c}^\top \boldsymbol{\xi} - \lambda^* \max_{\mathbf{x} \in X} \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \boldsymbol{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}} \leq v. \quad \square$$

10.3 No-regret learning algorithm

Since (10.3) is a bilinear saddle point, its solution can be found by self play of no-external-regret algorithms. However, a challenge that arises in this approach is that the domain of the minimization is unbounded—the Lagrange multiplier is allowed to take any nonnegative value. Nevertheless, we show in the theorem below that it suffices to set the Lagrange multiplier to a fixed value (that may depend on the time horizon); appropriately setting that value will allow us to trade-off between the equilibrium gap and the optimality gap. Before we proceed, we remark that the minimization domain in (10.3) is the Cartesian product of $\mathcal{Q}_{i, \hat{\sigma}}$ over all possible $(i, \hat{\sigma})$. Hence, a no-regret algorithm for the minimization problem can be obtained by instantiating individual no-regret algorithm for each $\mathcal{Q}_{i, \hat{\sigma}}$ using the approach described in Section 4.2.2; this justifies the notation $\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)}$ used for the regret of the min deviator below.

Theorem 10.1. Suppose that the max deviator in the saddle-point problem (10.3) incurs regret $\text{Reg}_{\Xi}^{(T)}$ and the min deviator incurs regret $\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)}$ after $T \in \mathbb{N}$ repetitions, for a fixed $\lambda = \lambda(T) > 0$. Then, the average mediator strategy $\Xi \ni \bar{\boldsymbol{\xi}} := \frac{1}{T} \sum_{t=1}^T \boldsymbol{\xi}^{(t)}$ satisfies the following:

1. For any strategy $\boldsymbol{\xi}^* \in \Xi$ such that $\max_{i \in [n], \hat{\sigma} \in \Sigma_i^*} \max_{\mathbf{x}_{i, \hat{\sigma}}^* \in \mathcal{Q}_{i, \hat{\sigma}}} (\boldsymbol{\xi}^*)^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}}^* \leq 0$,

$$\mathbf{c}^\top \bar{\boldsymbol{\xi}} \geq \mathbf{c}^\top \boldsymbol{\xi}^* - \frac{1}{T} \left(\text{Reg}_{\Xi}^{(T)} + \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} \right);$$

2. The equilibrium gap of $\bar{\boldsymbol{\xi}}$ decays with a rate of λ^{-1} :

$$\max_{i \in [n], \hat{\sigma} \in \Sigma_i^*} \max_{\mathbf{x}_{i, \hat{\sigma}}^* \in \mathcal{Q}_{i, \hat{\sigma}}} \bar{\boldsymbol{\xi}}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}}^* \leq \frac{\max_{\boldsymbol{\xi}, \boldsymbol{\xi}' \in \Xi} \mathbf{c}^\top (\boldsymbol{\xi} - \boldsymbol{\xi}')}{\lambda} + \frac{1}{\lambda T} \left(\text{Reg}_{\Xi}^{(T)} + \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} \right).$$

Proof. Let $\bar{\boldsymbol{\xi}} \in \Xi$ be the average strategy of the mediator, and $\bar{\mathbf{x}}_i \in \mathcal{Q}_{i, \hat{\sigma}}$ be the average strategy of each deviator $i \in [n], \hat{\sigma} \in \Sigma_i^*$ over the T iterations. We first argue about the approximate optimality of $\bar{\boldsymbol{\xi}}$. In particular, we have that

$$\mathbf{c}^\top \bar{\xi} \geq \max_{\xi \in \Xi} \left\{ \mathbf{c}^\top \xi - \lambda \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \xi^\top \mathbf{A}_{i, \hat{\sigma}} \bar{\mathbf{x}}_i \right\} - \frac{1}{T} \left(\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} + \text{Reg}_{\Xi}^{(T)} \right) \quad (10.6)$$

$$\geq \mathbf{c}^\top \xi^* - \lambda \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} (\xi^*)^\top \mathbf{A}_{i, \hat{\sigma}} \bar{\mathbf{x}}_i - \frac{1}{T} \left(\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} + \text{Reg}_{\Xi}^{(T)} \right) \quad (10.7)$$

$$\geq \mathbf{c}^\top \xi^* - \lambda \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \max_{\mathbf{x}_i^* \in \mathcal{Q}_{i, \hat{\sigma}}} (\xi^*)^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_i^* - \frac{1}{T} \left(\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} + \text{Reg}_{\Xi}^{(T)} \right) \quad (10.8)$$

$$\geq \mathbf{c}^\top \xi^* - \frac{1}{T} \left(\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} + \text{Reg}_{\Xi}^{(T)} \right),$$

where:

- (10.6) follows from the fact that the Lagrangian of the problem,

$$\mathcal{L} : \Xi \times X \ni (\xi, (\mathbf{x}_{i, \hat{\sigma}})_{i=1}^n) \mapsto \mathbf{c}^\top \xi - \lambda \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \xi^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}},$$

satisfies

$$\max_{\xi^* \in \Xi} \mathcal{L}(\xi^*, (\bar{\mathbf{x}}_i)_{i=1}^n) - \min_{(\mathbf{x}_i^*)_{i=1}^n \in X} \mathcal{L}(\bar{\xi}, (\mathbf{x}_i^*)_{i=1}^n) \leq \frac{1}{T} \left(\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} + \text{Reg}_{\Xi}^{(T)} \right), \quad (10.9)$$

in turn implying (10.6) since

$$\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \max_{\mathbf{x}_i^* \in \mathcal{Q}_{i, \hat{\sigma}}} \bar{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_i^* \geq \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \bar{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{0} = 0;$$

- (10.7) uses the notation ξ^* to represent any equilibrium strategy optimizing the objective $\mathbf{c}^\top \xi$; and
- (10.8) follows from the fact that, by assumption, ξ^* satisfies the equilibrium constraint $\max_{\mathbf{x}_i^* \in \mathcal{Q}_{i, \hat{\sigma}}} (\xi^*)^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_i^* \leq 0$ for any Player $i \in [n]$, $\hat{\sigma} \in \Sigma_i^*$, as well as the nonnegativity of the Lagrange multiplier.

This establishes Item 1 of the statement.

Next, we analyze the equilibrium gap of $\bar{\xi}$. Consider any mediator strategy $\xi \in \Xi$ such that $\xi^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}} \leq 0$ for any $\mathbf{x}_{i, \hat{\sigma}} \in \mathcal{Q}_{i, \hat{\sigma}}$ and deviator $i \in [n]$, $\hat{\sigma} \in \Sigma_i^*$. By (10.9),

$$\begin{aligned} \mathbf{c}^\top \boldsymbol{\xi} - \lambda \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \boldsymbol{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \bar{\mathbf{x}}_i - \mathbf{c}^\top \bar{\boldsymbol{\xi}} + \lambda \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \max_{\mathbf{x}_{i, \hat{\sigma}}^* \in \mathcal{Q}_{i, \hat{\sigma}}} \bar{\boldsymbol{\xi}}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}}^* \\ \leq \frac{1}{T} \left(\text{Reg}_{\Xi}^{(T)} + \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} \right). \end{aligned} \quad (10.10)$$

But, by the equilibrium constraint for $\boldsymbol{\xi}$, it follows that $\boldsymbol{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}} \leq 0$ for any $\mathbf{x}_{i, \hat{\sigma}} \in \mathcal{Q}_{i, \hat{\sigma}}$ and deviator $i \in [n], \hat{\sigma} \in \Sigma_i^*$, in turn implying that $\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \boldsymbol{\xi}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}} \leq 0$. So, combining with (10.10),

$$\lambda \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \max_{\mathbf{x}_{i, \hat{\sigma}}^* \in \mathcal{Q}_{i, \hat{\sigma}}} \bar{\boldsymbol{\xi}}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}}^* \leq \mathbf{c}^\top \bar{\boldsymbol{\xi}} - \mathbf{c}^\top \boldsymbol{\xi} + \frac{1}{T} \left(\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} + \text{Reg}_{\Xi}^{(T)} \right). \quad (10.11)$$

Finally, given that $\max_{\mathbf{x}_{i', \hat{\sigma}}^* \in \mathcal{X}_{i'}} \bar{\boldsymbol{\xi}}^\top \mathbf{A}_{i', \hat{\sigma}} \mathbf{x}_{i', \hat{\sigma}}^* \geq \bar{\boldsymbol{\xi}}^\top \mathbf{A}_{i', \hat{\sigma}} \mathbf{d}_{i'} = 0$ for any deviator i' , it follows that

$$\sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \max_{\mathbf{x}_{i, \hat{\sigma}}^* \in \mathcal{Q}_{i, \hat{\sigma}}} \bar{\boldsymbol{\xi}}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}}^* \geq \max_{i \in [n], \hat{\sigma} \in \Sigma_i^*} \max_{\mathbf{x}_{i, \hat{\sigma}}^* \in \mathcal{Q}_{i, \hat{\sigma}}} \bar{\boldsymbol{\xi}}^\top \mathbf{A}_{i, \hat{\sigma}} \mathbf{x}_{i, \hat{\sigma}}^*,$$

and (10.11) implies Item 2 of the statement. \square

As a result, if we can simultaneously guarantee that $\lambda(T) \rightarrow +\infty$ and

$$\frac{1}{T} \left(\text{Reg}_{\Xi}^{(T)} + \sum_{i \in [n]} \sum_{\hat{\sigma} \in \Sigma_i^*} \text{Reg}_{\mathcal{Q}_{i, \hat{\sigma}}}^{(T)} \right) \rightarrow 0, \quad \text{as } T \rightarrow +\infty,$$

Theorem 10.1 guarantees that both the optimality gap (Item 1) and the equilibrium gap (Item 2) converge to 0. We show that this is indeed possible in the sequel (**Corollaries 10.1** and **10.2**), obtaining favorable rates of convergence as well.

It is important to stress that while there exists a bounded critical Lagrange multiplier for our problem (**Proposition 10.1**), thereby obviating the need for truncating its value, such a bound is not necessarily polynomial.

10.3.1 Construction of weakly-predictive algorithms via regret circuits (**Chapter 4**)

Next, we combine **Theorem 10.1** with suitable regret minimization algorithms in order to guarantee fast convergence to optimal equilibria. A particularly natural approach in this case is via the regret circuits introduced in **Chapter 4**. Indeed, the regret minimization problem over the domain of the minimization in the saddle-point problem is the Cartesian product of the sets $\mathcal{Q}_{i, \hat{\sigma}}$, each of which is the convex hull between the singleton set $\{0\}$ and the sequence-form

polytope $\mathcal{Q}_{i, \neq j}$, for which several no-regret algorithms are known (including those presented in this dissertation, in [Chapters 4 to 7](#)).

Similarly, as we discussed in [Chapter 9](#), the domain Ξ of the maximization problem—the polytope of correlation plans—can be expressed as (possibly a linear projection of an exponentially long) composition of scaled extension operations and probability simplexes. Since scaled extension admits a regret circuit, we can repeatedly use the circuit in combination with no-regret algorithms for probability simplexes (for example, Discounted RM or PRM⁺) to construct a parameter-free no-external-regret algorithm that is rather competitive in practice.

Given that the deviators’ observed utilities have range $\mathcal{O}_T(\lambda)$, the regret of each deviator under suitable choices of local algorithm (for example, Discounted RM or PRM⁺ as we use in the experiments) will grow as $\mathcal{O}_T(\lambda\sqrt{T})$. Selecting a bound of $\lambda := T^{1/4}$ on the Lagrange multiplier, so as to optimally trade off Items 1 and 2 of [Theorem 10.1](#), leads to the following conclusion.

Corollary 10.1. There exist regret minimization algorithms such that when employed in the saddle-point problem (10.3), the average strategy of the mediator $\bar{\xi} := \frac{1}{T} \sum_{t=1}^T \xi^{(t)}$ converges to the set of optimal equilibria at a rate of $T^{-1/4}$. The per-iteration complexity is polynomial in the number of scaled extensions required to represent \mathcal{X} (see [Chapter 9](#)).

10.3.2 Faster rates using RVU-predictive algorithms ([Chapter 5](#))

As already discussed in [Chapter 5](#), proximal algorithms such as predictive online mirror descent beget RVU-predictive algorithms, which typically imply faster convergence rate compared to only weakly-predictive algorithm such as those constructed in [Section 10.3.1](#). In particular, using predictive mirror descent we can guarantee that the sum of the agents’ regrets in the saddle-point problem (10.3) will now grow as $\mathcal{O}_T(\lambda)$, instead of the previous bound $\mathcal{O}_T(\lambda\sqrt{T})$. Thus, letting $\lambda = T^{1/2}$ leads to the following improved rate of convergence.

Corollary 10.2 (Improved rates using predictivity). There exist regret minimization algorithms that guarantee that the average strategy of the mediator $\bar{\xi} := \frac{1}{T} \sum_{t=1}^T \xi^{(t)}$ converges to the set of optimal equilibria at a rate of $T^{-1/2}$. The per-iteration complexity is analogous to [Corollary 10.1](#).

Proof sketch. By the RVU bound (Syrkkanis, Agarwal, Luo, and Schapire, 2015; S. Rakhlin and Sridharan, 2013), the sum of the agents’ regrets can be bounded as $\frac{\text{diam}_{\Xi}^2 + \text{diam}_{\mathcal{X}}^2}{\eta}$, for a sufficiently small $\eta = \mathcal{O}_T(1/\lambda)$, where diam_{Ξ} and $\text{diam}_{\mathcal{X}}$ denote the ℓ_2 -diameter of Ξ and \mathcal{X} , respectively. Thus, taking $\eta = \Theta(1/\lambda)$ to be sufficiently small implies the statement. \square

Improving the $T^{-1/2}$ rate of [Corollary 10.2](#) is an interesting direction for future research.

10.3.3 Remarks on last-iterate convergence

The results we have stated thus far apply for the *average* strategy produced by the no-regret algorithm—a typical feature of traditional guarantees in the no-regret framework. In contrast, there is a recent line of work that endeavors to recover *last-iterate* guarantees as well (Daskalakis and Panageas, 2019; Gorbunov, Loizou, and Gidel, 2022; Abe, Sakamoto, and Iwasaki, 2022; Cai, Oikonomou, and Zheng, 2022; Azizian, Iutzeler, Malick, and Mertikopoulos, 2021; C. Wei, C. Lee, M. Zhang, and Luo, 2021; C.-W. Lee, Kroer, and Luo, 2021; Golowich, Pattathil, and Daskalakis, 2020; Lin, Zhou, Mertikopoulos, and Jordan, 2020; Gorbunov, Berard, Gidel, and Loizou, 2022). Yet, despite many efforts, the known last-iterate guarantees of no-regret learning algorithms apply only for restricted classes of games, such as two-player zero-sum games. There is an inherent reason for the limited scope of those results: last-iterate convergence is inherently tied to Nash equilibria, which in turn are hard to compute in general games (Daskalakis, Goldberg, and Papadimitriou, 2009; Chen, Deng, and Teng, 2009)—let alone computing an optimal one (Gilboa and Zemel, 1989; Conitzer and Sandholm, 2008). Indeed, any given joint strategy profile of the deviators induces a product distribution, so ϵ -iterate convergence requires at the very least computing an ϵ -Nash equilibrium.

Proposition 10.2 (Informal). Any uncoupled learning dynamics without a mediator require superpolynomial time to guarantee ϵ -last-iterate convergence, for a sufficiently small $\epsilon = (1/\text{poly})$, even for two-deviator normal-form games, unless $\text{PPAD} \subseteq \text{P}$.

There are also unconditional exponential communication-complexity lower bounds for uncoupled methods (Babichenko and Rubinstein, 2022; Hirsch, Papadimitriou, and Vavasis, 1989; Roughgarden and Weinstein, 2016; Hart and Mansour, 2010), as well as other pertinent impossibility results (Hart and Mas-Colell, 2003; Milionis, Papadimitriou, Piliouras, and Spendlove, 2022) that document the inherent persistence of limit cycles in general-sum games. In contrast, an important advantage of our formulation using Ξ is that we can guarantee last-iterate convergence to bilinear saddle points, using the known bound of $\mathcal{O}_T(\lambda/\sqrt{T})$ for the last-iterate gap of predictive online gradient descent can be employed (Anagnostides, Panageas, Farina, and Sandholm, 2022).

Theorem 10.2 (Last-iterate convergence to bilinear saddle points). There exist regret minimization algorithms that guarantee that the last strategy of the mediator $\xi^{(T)}$ converges to the set of optimal equilibria at a rate of $T^{-1/4}$. The per-iteration complexity is analogous to [Corollaries 10.1 and 10.2](#).

As such, our mediator-augmented learning paradigm bypasses the hardness of [Proposition 10.2](#) since last-iterate convergence is no longer tied to convergence to Nash equilibria.

10.4 Experimental evaluation

We extensively evaluate the empirical performance of the regret-circuit-based learning algorithm given in Section 10.3.1 for computing optimal EFCE across 23 game instances and different objective functions including social welfare. The game instances we use are described in detail in Appendix A, and belong to following eight different classes of established parametric benchmark games, each identified with an alphabetical mnemonic: **B** – Battleship, **D** – Liar’s dice, **GL** – Goofspiel, **K** – Kuhn poker, **L** – Leduc poker, **RS** – ridesharing game, **S** – Sheriff, **TP** – double dummy bridge game.

For each of the 23 games, we compare the runtime required by a commercial linear programming method instantiated on the optimal EFCE LP given in Proposition 9.1 (and originally proposed by B. H. Zhang, Farina, Celli, and Sandholm, 2022) (‘LP’) and the runtime required by our regret-circuit-based learning algorithm defined in Section 10.3.1 (‘Ours’) for computing ϵ -optimal equilibrium points.

Game	Max. Pl.1’s utility		Max. Pl.2’s utility		Max. soc. welf.		Game	Max. Pl.1’s utility		Max. Pl.2’s utility		Max. Pl.3’s utility	
	LP	Ours	LP	Ours	LP	Ours		LP	Ours	LP	Ours	LP	Ours
B2222	0.00s	0.01s	0.00s	0.05s	0.00s	0.02s	D32	12.00s	0.40s	11.00s	0.35s	10.00s	0.66s
B2322	9.00s	1.23s	9.00s	4.63s	9.00s	1.60s	D33	timeout	timeout	timeout	timeout	timeout	timeout
B2323	3m 54s	48.40s	4m 9s	1m 27s	3m 40s	44.87s	GL33	0.00s	0.01s	0.00s	0.01s	0.00s	0.01s
B2324	timeout	9m 3s	timeout	10m 21s	timeout	10m 48s	K35	57.00s	0.55s	55.00s	1.03s	60.00s	1.26s
S2122	0.00s	0.01s	0.00s	0.02s	0.00s	0.02s	L3132	8m 18s	6.10s	8m 57s	7.65s	7m 35s	6.78s
S2123	1.00s	0.09s	1.00s	0.34s	1.00s	0.15s	L3133	21m 25s	6.84s	21m 43s	10.76s	19m 58s	10.28s
S2133	4.00s	0.52s	3.00s	1.31s	3.00s	0.49s	L3151	timeout	timeout	timeout	timeout	timeout	timeout
S2254	timeout	2m 10s	timeout	timeout	timeout	3m 34s	L3223	2m 2s	5.52s	1m 50s	6.46s	2m 0s	5.94s
S2264	timeout	timeout	timeout	timeout	timeout	timeout	L3523	timeout	timeout	timeout	timeout	timeout	timeout
RS212	0.00s	0.00s	0.00s	0.00s	0.00s	0.00s	TP3	timeout	13.46s	timeout	14.25s	timeout	14.48s
RS213	timeout	35.37s	timeout	32.49s	timeout	23.37s							
RS222	0.00s	0.00s	0.00s	0.00s	0.00s	0.00s							
RS223	timeout	timeout	timeout	timeout	timeout	timeout							

Table 10.1: Experimental comparison between our learning-based approach (‘Ours’, Section 10.3.1) and using the commercial LP solver Gurobi on the linear programming formulation of optimal EFCE (Proposition 9.1), both for computing an optimal EFCE within an optimality and feasibility tolerance set to 1% of the payoff range of the game.

Table 10.1 and Table 10.2 show experimental results for the case in which the threshold ϵ is set to be, respectively, 1% and 0.1% of the payoff range of the game. Each table is split into a left part, which contains two-player general-sum games, and a right part with three-player zero-sum games. For each game, three objective functions are considered:

- For the two-player general-sum games, we consider the three objectives of (i) maximizing Player 1’s utility, (ii) maximizing Player 2’s utility, and (iii) maximizing social welfare, that is, the sum of the two players’ utilities.
- For the three-player zero-sum games, we consider the three objectives corresponding to

maximizing Player 1, 2, and 3’s utility, respectively.

Each row corresponds to a game, whose identifier begins with the alphabetical mnemonic of the game class.

The regret-circuit-based construction detailed in [Section 10.3.1](#) allows a degree of flexibility as to what no-regret algorithms are used for each of the probability simplexes that make up Ξ and the $Q_{i,\hat{\sigma}}$. In the experimental tables, the column ‘Ours’ reports the minimum between the time required by the regret-circuit-based algorithm instantiated with Discounted RM and PRM⁺ at each decision point. For each run of the algorithms, the timeout was set at one hour.

We observe that our learning-based approach is faster—often by more than an order of magnitude—and more scalable than the linear program. This shows the promise of our computational approach.

Game	Max. Pl.1’s utility		Max. Pl.2’s utility		Max. soc. welf.		Game	Max. Pl.1’s utility		Max. Pl.2’s utility		Max. Pl.3’s utility	
	LP	Ours	LP	Ours	LP	Ours		LP	Ours	LP	Ours	LP	Ours
B2222	0.00s	0.16s	0.00s	0.10s	0.00s	0.03s	D32	14.00s	0.92s	13.00s	0.74s	12.00s	1.60s
B2322	9.00s	11.04s	9.00s	10.87s	9.00s	4.11s	D33	timeout	timeout	timeout	timeout	timeout	timeout
B2323	4m 12s	3m 41s	4m 9s	3m 35s	3m 40s	1m 28s	GL33	0.00s	0.06s	0.00s	0.05s	0.00s	0.05s
B2324	timeout	timeout	timeout	timeout	timeout	timeout	K35	1m 5s	2.99s	1m 3s	4.80s	1m 4s	4.66s
S2122	0.00s	0.04s	0.00s	0.07s	0.00s	0.07s	L3132	9m 41s	26.68s	10m 28s	27.57s	9m 54s	30.00s
S2123	2.00s	0.24s	1.00s	1.63s	1.00s	0.37s	L3133	26m 52s	22.31s	27m 22s	39.44s	25m 32s	40.75s
S2133	5.00s	1.12s	3.00s	4.80s	4.00s	0.95s	L3151	timeout	timeout	timeout	timeout	timeout	timeout
S2254	timeout	6m 24s	timeout	timeout	timeout	9m 2s	L3223	2m 38s	20.44s	2m 32s	28.27s	2m 31s	49.95s
S2264	timeout	timeout	timeout	timeout	timeout	timeout	L3523	timeout	timeout	timeout	timeout	timeout	timeout
RS212	0.00s	0.00s	0.00s	0.01s	0.00s	0.00s	TP3	timeout	26.28s	timeout	40.64s	timeout	41.96s
RS213	timeout	5m 1s	timeout	4m 19s	timeout	2m 28s							
RS222	0.00s	0.02s	0.00s	0.02s	0.00s	0.02s							
RS223	timeout	timeout	timeout	timeout	timeout	timeout							

Table 10.2: Experimental comparison between our learning-based approach (‘Ours’, [Section 10.3.1](#)) and using the commercial LP solver Gurobi on the linear programming formulation of optimal EFCE ([Proposition 9.1](#)), both for computing an optimal EFCE within an optimality and feasibility tolerance set to 0.1% of the payoff range of the game.

Chapter 11

Learning dynamics for team coordination and collusion

The positive results regarding the geometry of correlated strategies we investigated in [Chapter 9](#) also imply positive results about the computation of optimal team strategies in imperfect-information two-team zero-sum games, helping answer questions such as:

How should two players colluding against a third at a poker table play?

Or, how would the two defenders in Bridge (who are prohibited from communicating privately during the game) play optimally against the declarer?

Depending on the amount of communication allowed among the team members, we can identify at least three solution concepts.

- If the team members can freely (and privately) communicate during play, the team effectively becomes a single player. Hence, all the tools we have seen so far (for example, learning an optimal strategy using CFR or linear programming) directly apply. We will refer to this equilibrium as ‘Team Nash equilibrium’.
- If the team members cannot communicate at all, then the strategies of the team members should be picked as the pair of strategies that maximizes the expected utility of the team against a best-responding agent. This solution concept is called a *team maxmin equilibrium (TME)*. As we show in [Section 11.2](#), the minmax theorem does not hold in general for TME. So, perhaps, the term “equilibrium” should be used carefully when referring to TME.
- The third case is intermediate, and models the setting where the team members have an opportunity to discuss and agree on tactics before the game starts, but are otherwise unable

to communicate during the game, except through their publicly-observed actions. This models, for example, multiplayer poker in the presence of collusion, and the game of bridge. This solution concept is called a *team maxmin equilibrium with correlation device (TMECor)*.

TMECor can be thought of as follows. Before playing, the team members get together in secret, and discuss about tactics for the game. They come up with m possible plans, each of which specifies a deterministic strategy for each team member, and write them down in m separate envelopes. Then, just before the game starts, they together pick one of the m envelopes according to a shared probability distribution, and play according to the plan in the chosen envelope.

It is important to realize that the sampling of the envelope can happen even if the team members cannot communicate before the game starts, as long as they can agree on some shared signal, such as for example a common clock. With that signal as input, the team members could seed a random number generator and use that to agree on the same random envelope to use, without having communicated among each other.

Table 11.1 compares the properties of the three solution concepts defined above. Generally speaking, as the amount of allowed communication increases (from TME to TMECor to Team Nash), the utility of the team increases, while the complexity of computing the optimal strategy for the team decreases. In particular, as the first three rows of Table 11.1 show, the problem of computing a TME strategy for the team is significantly less well-behaved than TMECor.

For all these reasons, we believe that TMECor is often a superior solution concept for adversarial team games when no in-game private communication is allowed. It yields higher utility to the team, is computationally and game-theoretically better behaved, and can be implemented in many practical interaction.

	TME no communication ever	TMECor no communication during play	Team Nash Eq. private communication during play
Convex problem	✗	✓	✓
Bilinear saddle-point problem	✗	✓	✓
Low-dimensional strat. polytope	✗	✓	✓
Minmax theorem	✗	✓	✓
Team utility	low	higher	highest
Complexity	very hard	sometimes hard	polynomial

Table 11.1: Comparison between TME, TMECor and Team Nash equilibrium.

11.1 Contributions and related work

After demonstrating that TME does not satisfy the minmax theorem, in this chapter we show that *learning dynamics* for the teams can be devised to compute TMECor, by connecting TMECor computation with EFCE-related technology introduced in Chapters 9 and 10. As we show, this approach leads to theoretical and practical state-of-the-art learning algorithms for computing TMECor strategies in imperfect-information two-team zero-sum games, adding to the conclusion, already drawn several times in this dissertation, that uncoupled learning dynamics often lead to both the theoretical and practical state-of-the-art technique for computing equilibria in imperfect-information extensive-form games.

The study of TME dates back to at least the work by von Stengel and Koller (1997). The study of TMECor—the focus of this chapter—is more recent. Basilio, Celli, De Nittis, and Gatti (2017) show that TMECor can lead to significantly higher utility (up to a factor linear in the number of game tree leaves) than TME. The study of the computational aspects of TMECor was initiated by Celli and Gatti (2018). Until recently, the best techniques for solving adversarial team games in absence of communication were based on *column generation* (Farina, Celli, Gatti, and Sandholm, 2018; Farina, Celli, Gatti, and Sandholm, 2021; Y. Zhang, An, and Černỳ, 2021; B. H. Zhang, Farina, Celli, and Sandholm, 2022). Those techniques work well in some small and medium-sized games in practice, but generally have no or weak theoretical guarantees. More recently, other techniques emerged. B. H. Zhang and Sandholm (2022b) developed an algorithm for solving adversarial team games based on a novel *tree decomposition* of each player’s strategy space, and use it to devise a linear program. They show parameterized complexity bounds that scale with the amount of uncommon information among team members. Simultaneously, Carminati, Cacciamani, Ciccone, and Gatti (2022) developed an algorithm for converting the game into a strategically equivalent (but exponentially-larger) *two-player* game with perfect recall, inspired by prior research in the multiagent reinforcement learning community (e.g., Nayyar, Mahajan, and Teneketzis, 2013; Sokota, Lockhart, Timbers, Davoodi, D’Orazio, Burch, Schmid, Bowling, and Lanctot, 2021).

The technique we develop in this chapter is based on learning dynamics, and yields an algorithm significantly faster than all prior approaches.

11.2 Failure of minmax theorem for team maxmin equilibrium

As mentioned in the introduction, one of the major shortcomings of TME (at least from a game-theoretic point of view) is that it fails the minmax theorem. We demonstrate this by explicitly showing a game in which the maxmin and minmax values of the game are different.

Consider the matching pennies game of Figure 11.1 (Left), where a team \blacktriangle of two players (denoted $1\blacktriangle$ and $2\blacktriangle$ respectively) compete against an opponent. Each player has a penny. First, the opponent (denoted \blacktriangledown) secretly turns their penny heads (H) or tails (T). Then, Player $1\blacktriangle$ secretly

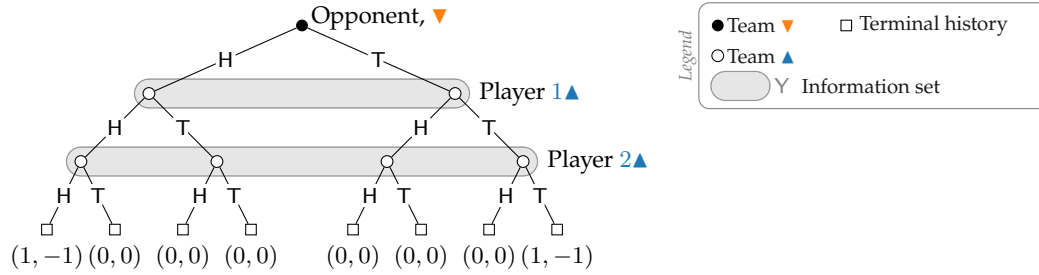


Figure 11.1: Matching pennies game.

turns their penny heads or tail. Finally, Player 2▲ turns their penny heads or tail. After all pennies have been turned, all of them are revealed. If all three pennies match (that is, they are all heads or they are all tails), the team wins a payoff of 1 and the opponent suffers a loss of -1 . Otherwise, nobody gains or loses anything.

Maxmin value We will use x, y, z to denote strategies for Players 1▲, 2▲, and ▼, respectively. The TME strategy for team ▲ is the solution to the *maxmin* problem

$$v_{\maxmin} := \begin{cases} \max_{x,y} & \begin{cases} \min_z & x_H \cdot y_H \cdot z_H + x_T \cdot y_T \cdot z_T \\ \text{s.t.} & \textcircled{1} z_H + z_T = 1 \\ & \textcircled{2} z_H, z_T \geq 0 \end{cases} \\ \text{s.t.} & \begin{cases} \textcircled{3} x_H + x_T = 1 \\ \textcircled{4} y_H + y_T = 1 \\ \textcircled{5} x_H, x_T, y_H, y_T \geq 0. \end{cases} \end{cases} \quad (\spadesuit)$$

Lemma 11.1. The solution to the maxmin problem defined in (\spadesuit) is $v_{\maxmin} = 1/4$.

Proof. The internal minimization problem in (\spadesuit) is minimizing a linear function over the 2-simplex. So, the solution to the internal minimization problem is the minimum of the function over the two vertices, that is, $\min\{x_H \cdot y_H, y_T \cdot y_T\}$. Hence, we can rewrite (\spadesuit) as

$$v_{\maxmin} = \begin{cases} \max_{x,y} & \min\{x_H \cdot y_H, y_T \cdot y_T\} \\ \text{s.t.} & \begin{cases} \textcircled{3} x_H + x_T = 1 \\ \textcircled{4} y_H + y_T = 1 \\ \textcircled{5} x_H, x_T, y_H, y_T \geq 0. \end{cases} \end{cases}$$

We can now perform the substitutions $x_T = 1 - x_H$ and $y_T = 1 - y_H$ (justified by ③ and ④), and get to

$$v_{\max\min} = \begin{cases} \max_{x,y} & \min\{x_H \cdot y_H, (1 - x_H) \cdot (1 - y_H)\} \\ \text{s.t.} & \textcircled{5} \ 0 \leq x_H \leq 1, \quad 0 \leq y_H \leq 1. \end{cases}$$

Not that the problem is completely symmetric: if (x_H^*, y_H^*) is an optimal solution, then $(1 - x_H^*, 1 - y_H^*)$ is also optimal. Hence, there exists at least one optimal solution in which

$$x_H \cdot y_H \leq (1 - x_H) \cdot (1 - y_H) \iff x_H + y_H \leq 1$$

and consequently

$$v_{\max\min} = \begin{cases} \max_{x,y} & x_H \cdot y_H \\ \text{s.t.} & \textcircled{6} \ x_H + y_H \leq 1 \\ & \textcircled{5} \ 0 \leq x_H \leq 1, \quad 0 \leq y_H \leq 1. \end{cases}$$

Since x_H is nonnegative, the objective is maximized when y_H is maximized, which happens for $y_H = 1 - x_H$. So,

$$v_{\max\min} = \begin{cases} \max_{x_H} & x_H \cdot (1 - x_H) \\ \text{s.t.} & \textcircled{5} \ 0 \leq x_H \leq 1, \end{cases}$$

which is the maximum of a one-dimensional parabola. Taking the gradient of the objective, it's immediate to check that the maximum is obtained when $x_H = 1/2$, and hence the statement of the lemma follows. \square

Minmax value Conversely, the TME strategy for the opponent is the solution to the *maxmin* problem

$$v_{\min\max} := \begin{cases} \min_z & \begin{cases} \max_{x,y} & x_H \cdot y_H \cdot z_H + x_T \cdot y_T \cdot z_T \\ \text{s.t.} & \textcircled{1} \ x_H + x_T = 1 \\ & \textcircled{2} \ y_H + y_T = 1 \\ & \textcircled{3} \ x_H, x_T, y_H, y_T \geq 0. \end{cases} \\ \text{s.t.} & \textcircled{4} \ z_H + z_T = 1 \\ & \textcircled{5} \ z_H, z_T \geq 0 \end{cases} \quad (\clubsuit)$$

Lemma 11.2. The solution to the minmax problem defined in (♣) is $v_{\min\max} = 1/2$.

Proof. The choices $(x_H, x_T, y_H, y_T) = (1, 0, 1, 0)$ and $(x_H, x_T, y_H, y_T) = (0, 1, 0, 1)$ are feasible for the internal maximization problem in (♣). Hence, we have

$$v_{\min\max} \geq \begin{cases} \min_z & \max\{z_H, z_T\} \\ \text{s.t.} & \textcircled{4} \ z_H + z_T = 1 \\ & \textcircled{5} \ z_H, z_T \geq 0 \end{cases} = \begin{cases} \min_z & \max\{z_H, 1 - z_H\} \\ \text{s.t.} & \textcircled{5} \ 0 \leq z_H \leq 1. \end{cases}$$

It's immediate to see that the solution to the one-dimensional optimization problem on the right-hand side is $1/2$, attained for $z_H = 1/2$. So $v_{\min\max} \geq 1/2$. To complete the proof, it is enough to show that there exists an assignment of z for which (♣) attains the value $1/2$.

To that end, consider the feasible assignment $z_H = z_T = 1/2$. Then, the objective of (♣) is

$$\begin{aligned} \begin{cases} \max_{x,y} & 1/2 \cdot x_H \cdot y_H + 1/2 \cdot x_T \cdot y_T \\ \text{s.t.} & \textcircled{1} \ x_H + x_T = 1 \\ & \textcircled{2} \ y_H + y_T = 1 \\ & \textcircled{3} \ x_H, x_T, y_H, y_T \geq 0. \end{cases} &= \begin{cases} \max_{x,y} & 1/2(x_H \cdot y_H + (1 - x_H)(1 - y_H)) \\ \text{s.t.} & \textcircled{3} \ 0 \leq x_H \leq 1, \ 0 \leq y_H \leq 1. \end{cases} \\ &= \begin{cases} \max_{x,y} & (x_H - 1/2)(y_H - 1/2) + 1/4 \\ \text{s.t.} & \textcircled{3} \ 0 \leq x_H \leq 1, \ 0 \leq y_H \leq 1. \end{cases} \\ &= 1/2, \end{aligned}$$

where the last equality follows from since the product in the objective is maximized for $x_H = y_H = 1$. □

Given that the maxmin and minmax values are different, we conclude that, unlike regular minmax equilibrium, TME does *not* satisfy the minmax theorem in general.

11.3 TMECOR as a bilinear saddle-point problem

In this section, we show how TMECOR in imperfect-information two-team zero-sum games can be cast as a bilinear saddle-point problem over polytopes embedded in Euclidean spaces of polynomial dimension in the game size. To simplify the exposition, we will assume that the teams—denoted \blacktriangle and \blacktriangledown respectively—each consist of two team members, denoted $1\blacktriangle, 2\blacktriangle$ and $1\blacktriangledown, 2\blacktriangledown$ respectively. The case where a team has less or more players extends directly; in the experimental evaluation we test on teams of varying size.

Similar to maxmin strategies in regular (non-team) two-player zero-sum games, a TMECor strategy for team (say, \blacktriangle) is a distribution $\mu_{\blacktriangle} \in \Delta^{\Pi_{1\blacktriangle} \times \Pi_{2\blacktriangle}}$ over tuples of deterministic strategies for the team members' (1 \blacktriangle and 2 \blacktriangle), such that it maximizes the team's expected utility against a best-responding, independent opposing team:

$$\max_{\mu_{\blacktriangle} \in \Delta^{\Pi_{1\blacktriangle} \times \Pi_{2\blacktriangle}}} \min_{\mu_{\blacktriangledown} \in \Delta^{\Pi_{1\blacktriangledown} \times \Pi_{2\blacktriangledown}}} \mathbb{E}_{(\pi_{1\blacktriangle}, \pi_{2\blacktriangle}) \sim \mu_{\blacktriangle}} \mathbb{E}_{(\pi_{1\blacktriangledown}, \pi_{2\blacktriangledown}) \sim \mu_{\blacktriangledown}} \left[u_{\blacktriangle}(\pi_{1\blacktriangle}, \pi_{2\blacktriangle}, \pi_{1\blacktriangledown}, \pi_{2\blacktriangledown}) \right], \quad (11.1)$$

where $u_{\blacktriangle} = u_{1\blacktriangle} + u_{2\blacktriangle}$ denotes the utility function of team \blacktriangle , defined as the sum of the utilities of its team members.

By definition of expectation, the objective function in (11.1) is bilinear in μ_{\blacktriangle} and μ_{\blacktriangledown} . This shows that the computation of a TMECor is a bilinear saddle-point problem over convex polytopes (and therefore in particular it can be converted into a linear program, which is itself a convex problem, cf. Table 11.1). Another consequence of the bilinear saddle-point structure is the fact that the minmax theorem must hold for TMECor strategies, unlike TME, as we have hinted in Section 11.2. However, as formulated, the optimization problem (11.1) has the major drawback of requiring optimization over exponentially-large domains, since so is typically the number of deterministic strategies of every player. We show in the next subsection that by operating an appropriate change of variables, we can retain the convexity-concavity and the bilinear saddle-point structure, while at the same time gaining that the domains of the maximizing and minimization problems belong to a space of polynomial dimension in the game tree size.

11.3.1 Realization vectors and low-dimensional parameterization

Letting $u_{\blacktriangle}(z) = u_{1\blacktriangle}(z) + u_{2\blacktriangle}(z)$ denote the utility received by team \blacktriangle at terminal node $z \in \mathcal{Z}$, $p_c(z)$ denote the product of the probability of all chance actions on the path from the root of the game tree to terminal node z , and $\sigma_i(z)$ denote the last sequence on the path from the root of the game tree to z , by definition we have that the utility $u_{\blacktriangle}(\pi_{1\blacktriangle}, \pi_{2\blacktriangle}, \pi_{1\blacktriangledown}, \pi_{2\blacktriangledown})$ can be written as

$$\sum_{z \in \mathcal{Z}} u_{\blacktriangle}(z) \cdot p_c(z) \cdot \underbrace{\pi_{1\blacktriangle}[\sigma_{1\blacktriangle}(z)] \cdot \pi_{2\blacktriangle}[\sigma_{2\blacktriangle}(z)]}_{\substack{\text{probability of all } \blacktriangle\text{'s} \\ \text{actions on the path} \\ \text{from root to } z}} \cdot \underbrace{\pi_{1\blacktriangledown}[\sigma_{1\blacktriangledown}(z)] \cdot \pi_{2\blacktriangledown}[\sigma_{2\blacktriangledown}(z)]}_{\substack{\text{probability of all } \blacktriangledown\text{'s} \\ \text{actions on the path} \\ \text{from root to } z}} \quad (11.2)$$

By using (11.2) together with the independence of μ_{\blacktriangle} and μ_{\blacktriangledown} , the expected utility

$$\bar{u}_{\blacktriangle}(\mu_{\blacktriangle}, \mu_{\blacktriangledown}) := \mathbb{E}_{(\pi_{1\blacktriangle}, \pi_{2\blacktriangle}) \sim \mu_{\blacktriangle}} \mathbb{E}_{(\pi_{1\blacktriangledown}, \pi_{2\blacktriangledown}) \sim \mu_{\blacktriangledown}} \left[u_{\blacktriangle}(\pi_{1\blacktriangle}, \pi_{2\blacktriangle}, \pi_{1\blacktriangledown}, \pi_{2\blacktriangledown}) \right]$$

satisfies the equation

$$\bar{u}_\Delta(\boldsymbol{\mu}_\Delta, \boldsymbol{\mu}_\nabla) = \sum_{z \in \mathcal{Z}} u_\Delta(z) \cdot p_c(z) \cdot \mathbb{E}_{\boldsymbol{\mu}_\Delta} \left[\boldsymbol{\pi}_{1\Delta}[\sigma_{1\Delta}(z)] \cdot \boldsymbol{\pi}_{2\Delta}[\sigma_{2\Delta}(z)] \right] \cdot \mathbb{E}_{\boldsymbol{\mu}_\nabla} \left[\boldsymbol{\pi}_{1\nabla}[\sigma_{1\nabla}(z)] \cdot \boldsymbol{\pi}_{2\nabla}[\sigma_{2\nabla}(z)] \right], \quad (11.3)$$

where for brevity we used $\mathbb{E}_{\boldsymbol{\mu}_\Delta}$ and $\mathbb{E}_{\boldsymbol{\mu}_\nabla}$ as shorthands for $\mathbb{E}_{(\boldsymbol{\pi}_{1\Delta}, \boldsymbol{\pi}_{2\Delta}) \sim \boldsymbol{\mu}_\Delta}$ and $\mathbb{E}_{(\boldsymbol{\pi}_{1\nabla}, \boldsymbol{\pi}_{2\nabla}) \sim \boldsymbol{\mu}_\nabla}$, respectively. The above formulation shows that it is not necessary to optimize over $\boldsymbol{\mu}_\Delta$ and $\boldsymbol{\mu}_\nabla$ directly, as the objective function only depends bilinearly on the aggregate quantities

$$\begin{aligned} \boldsymbol{\omega}_\Delta[z] &:= \mathbb{E}_{\boldsymbol{\mu}_\Delta} \left[\boldsymbol{\pi}_{1\Delta}[\sigma_{1\Delta}(z)] \cdot \boldsymbol{\pi}_{2\Delta}[\sigma_{2\Delta}(z)] \right] = \sum_{(\boldsymbol{\pi}_{1\Delta}, \boldsymbol{\pi}_{2\Delta})} \boldsymbol{\mu}_\Delta[\boldsymbol{\pi}_{1\Delta}, \boldsymbol{\pi}_{2\Delta}] \cdot \boldsymbol{\pi}_{1\Delta}[\sigma_{1\Delta}(z)] \cdot \boldsymbol{\pi}_{2\Delta}[\sigma_{2\Delta}(z)], \\ \boldsymbol{\omega}_\nabla[z] &:= \mathbb{E}_{\boldsymbol{\mu}_\nabla} \left[\boldsymbol{\pi}_{1\nabla}[\sigma_{1\nabla}(z)] \cdot \boldsymbol{\pi}_{2\nabla}[\sigma_{2\nabla}(z)] \right] = \sum_{(\boldsymbol{\pi}_{1\nabla}, \boldsymbol{\pi}_{2\nabla})} \boldsymbol{\mu}_\nabla[\boldsymbol{\pi}_{1\nabla}, \boldsymbol{\pi}_{2\nabla}] \cdot \boldsymbol{\pi}_{1\nabla}[\sigma_{1\nabla}(z)] \cdot \boldsymbol{\pi}_{2\nabla}[\sigma_{2\nabla}(z)] \end{aligned} \quad (11.4)$$

for $z \in \mathcal{Z}$. Hence, at least conceptually, we can operate a change of variables in the optimization problem, trading the distributions $\boldsymbol{\mu}_\Delta, \boldsymbol{\mu}_\nabla$ in (11.1), for the vectors $\boldsymbol{\omega}_\Delta, \boldsymbol{\omega}_\nabla$, called *realization vectors*. Doing so greatly reduces the number of variables in the optimization problem: the distributions $\boldsymbol{\mu}$ are over an exponential number of elements (the product of deterministic plans for each of the team's members), while $\boldsymbol{\omega}_\Delta, \boldsymbol{\omega}_\nabla$ only require $|\mathcal{Z}|$ scalar variables—a linear amount in the game tree size.

In order to perform the change of variable effectively, we need to study the new domain of the optimization problem. In particular, as we move away from $\boldsymbol{\mu}_i \in \Delta^{\Pi_{i1} \times \Pi_{i2}}$ and allow the team to specify a $\boldsymbol{\omega}_i$ instead ($i \in \{\Delta, \nabla\}$), we need to characterize the domain of $\boldsymbol{\omega}_i$ that can be induced from $\boldsymbol{\mu}_i$ according to (11.4). The key observation here is that the mapping from $\boldsymbol{\mu}_i$ to $\boldsymbol{\omega}_i$ is *linear*. Hence, since the set of feasible $\boldsymbol{\mu}_i$ is a convex polytope (the simplex $\Delta^{\Pi_{i1} \times \Pi_{i2}}$), we can use the following lemma to conclude that the set Ω_i of all feasible $\boldsymbol{\omega}_i$ is itself a convex polytope. Hence, a TMECOR can be expressed as a bilinear saddle-point problem

$$\max_{\boldsymbol{\omega}_\Delta \in \Omega_\Delta} \min_{\boldsymbol{\omega}_\nabla \in \Omega_\nabla} \sum_{z \in \mathcal{Z}} u_\Delta(z) \cdot p_c(z) \cdot \boldsymbol{\omega}_\Delta[z] \cdot \boldsymbol{\omega}_\nabla[z] =: \max_{\boldsymbol{\omega}_\Delta \in \Omega_\Delta} \min_{\boldsymbol{\omega}_\nabla \in \Omega_\nabla} \boldsymbol{\omega}_\Delta^\top \mathbf{U}_\Delta \boldsymbol{\omega}_\nabla, \quad (11.5)$$

where \mathbf{U}_Δ is an appropriate matrix that encodes the bilinear objective.

While the formulation (11.5) is reminiscent of the bilinear formulation of Nash equilibrium in two-player zero-sum games, several important differences exist. Most importantly, unfortunately, in general the convex polytopes Ω_Δ and Ω_∇ cannot be captured via a polynomial number of linear constraints, as it is known that the computation of TMECOR is computationally intractable in practice (Chu and Halpern, 2001). This makes the formulation (11.5) conceptually similar to that of EFCE (Chapters 9 and 10). In the next subsection we formalize this latter connection, and give new positive complexity results for TMECOR, as well as learning dynamics for teams inspired by the technique pioneered by Chapter 10.

11.3.2 Connection with correlation plans and triangle-freeness

For each team $i \in \{\blacktriangle, \blacktriangledown\}$, consider the game tree Γ_i obtained by converting all nodes of the opposing team into chance nodes. The polytope of correlation plans Ξ_i of Γ_i encode a *superset* of the information encoded by realization plans. Indeed, it is immediate to see from the definition of the polytope of correlation plans (Definitions 9.1 and 9.2) that the following holds.

Lemma 11.3. The polytope of valid realization vectors for each team $i \in \{\blacktriangle, \blacktriangledown\}$ can be obtained by considering a subset of the dimensions of the polytope of correlation plans Ξ_i of the team, specifically

$$\Omega_i = \left\{ \left(\xi[\sigma_{i1}(z), \sigma_{i2}(z)] \right)_{z \in \mathcal{Z}} : \xi \in \Xi_i \right\}. \quad (11.6)$$

As a direct consequence, we have the following.

Corollary 11.1. When the information structure of the members of a team $i \in \{\blacktriangle, \blacktriangledown\}$ is triangle-free (Section 9.4.2), then Ω_i can be defined using only polynomially many constraints in the size of the input game tree.

Hence, when both teams are triangle-free, or a triangle-free team plays against a single opponent (for which the set of correlation plans is simply the sequence-form polytope), a TMECor can be found in polynomial time.

As far as we are aware, this positive complexity result has not been noted before in the literature. As an example application, we remark that in many cases the **G** Goofspiel, **GL** limited-information Goofspiel, and **B** battleship games involve triangle freeness, and therefore a TMECor in those instances can be computed in polynomial time.

More generally, the connection between Ω_i and Ξ_i enables us to use the results discussed in Section 9.5 to characterize an upper bound on the number of constraints required to define Ω_i , and consequently arrive at parameterized complexity results for the computation of TMECor.

Corollary 11.2. For any team $i \in \{\blacktriangle, \blacktriangledown\}$, the set Ω_i of realization vectors is a linear transformation of a set that can be defined via a chain of scaled extension operations

$$\{1\} \overset{h_1}{\triangleleft} \mathcal{X}_1 \overset{h_2}{\triangleleft} \mathcal{X}_2 \overset{h_3}{\triangleleft} \cdots \overset{h_m}{\triangleleft} \mathcal{X}_m, \quad (11.7)$$

where, for $j = 1, \dots, m$, $\mathcal{X}_j = \Delta^{s_j}$ for some simplex dimension $s_j \in \mathbb{N}$ and h_j is a linear function.

Remark 11.1. In fact, we remark that since Ω_i only depends on a subset of Ξ_i , sharper bounds can be given on the number of simplexes m used in (11.7) and the total dimension $\sum_{j=1}^m s_j$ that what could be done directly using the results known for Ξ_i alone. We will not discuss such improvements in this dissertation, but refer the interested reader to the preprint by B. H. Zhang, Farina, and Sandholm (2023).

Following the same approach as Chapter 10, by leveraging the structural decomposition of $\Omega_{\blacktriangle}, \Omega_{\blacktriangledown}$ via scaled extension operations (Corollary 11.2) via the framework of regret circuits described in Chapter 4, we obtain no-external-regret learning dynamics. By using the folklore connection between no-external-regret algorithms and solutions to bilinear saddle-point problems, we can then use these learning dynamics to compute TMECor in games. We show in Section 11.4 that this approach indeed leads to the current practical state-of-the-art algorithm for computing TMECor.

11.4 Experimental evaluation

In this section we experimentally evaluate our no-external-regret dynamics for solving the TMECor bilinear saddle-point formulation (11.5). Following the approach laid out in Chapter 10, we construct the dynamics by applying the regret circuit of Section 4.2.5 to the scaled-extension-based decomposition of the sets (Corollary 11.2), employing both PRM^+ and Discounted RM as local no-external-regret minimizers for the probability simplexes that form the scaled-extension-decomposition of $\Xi_{\blacktriangle}, \Xi_{\blacktriangledown}$.

We compare our method against the prior state-of-the-art algorithms from three different set of techniques that have been used for this problem (see also Section 11.1):

- The tree-decomposition-based LP solver proposed by B. H. Zhang and Sandholm (2022b) (henceforth “ZS22”). We used the original implementation of the authors, which internally uses the barrier algorithm implemented by the commercial solver Gurobi. As recommended by the authors, we turned Gurobi’s presolver off to avoid numerical instability and increase speed. We allowed Gurobi to use up to four threads.
- The single-oracle algorithm of B. H. Zhang, Farina, Celli, and Sandholm (2022) (henceforth “ZFCS22”), which is an improved version of a prior single-oracle algorithm by Farina, Celli, Gatti, and Sandholm (2021). It represents currently the fastest single-oracle benchmark, and therefore it fairly represents the state of the literature based on such technique. ZFCS22 iteratively refines the strategy of each team by solving best-response problems using a tight integer program derived from the theory of extensive-form correlation (von Stengel and Forges, 2008; Farina, Kroer, and Sandholm, 2021b). We used the original code by the authors, which

was implemented for three-player games in which a team of two players faces an opponent. Like ZS22 and our LP-based solver, ZFCS22 uses the commercial solver Gurobi to solve linear and integer linear programs. We allowed Gurobi to use up to four threads.

- The coordinator-based representation of Carminati, Cacciamani, Ciccone, and Gatti (2022) (henceforth, “CCCG22”), which computes a TMECor by converting the problem into computing a Nash equilibrium of a strategically-equivalent by exponentially-sized imperfect-information extensive-form game, using a technique inspired by the work of Nayyar, Mahajan, and Teneketzis (2013).

All experiments were run on a 64-core AMD EPYC 7282 processor. Each algorithm was allocated a maximum of 4 threads, 60GBs of RAM, and a time limit of 6 hours. Experimental results are summarized in Table 11.2.

Game $\{\blacktriangledown\}$	Decompos. of Ω_{\blacktriangle}		Decompos. of $\Omega_{\blacktriangledown}$		CCCG22 Game size	This chapter		ZS22	ZFCS22		
	Simplexes	Dimension	Simplexes	Dimension		$\epsilon=10^{-3}$	$\epsilon=10^{-4}$	‡	$\epsilon=10^{-3}$	$\epsilon=10^{-4}$	
K33	{3}	487	918	37	36	4,108	0.00s	0.00s	0.01s	0.00s	0.00s
K34	{3}	2,100	6,711	49	48	66,349	0.00s	0.00s	0.02s	0.01s	0.02s
K36	{3}	54,255	336,944	73	72	7,002,763	0.03s	0.12s	1.22s	0.14s	0.14s
K38	{3}	1,783,926	15,564,765	97	96	488,157,721	4.73s	32.36s	3m 23s	0.23s	0.32s
K3c	{3}	—	—	—	—	—	oom	oom	oom	0.84s	1.39s
K45	{3,4}	26,566	124,875	4,621	15,415	—	0.03s	0.05s	0.79s	—	—
K45	{4}	998,471	4,658,070	121	120	202,660,366	1.59s	6.34s	3m 25s	—	—
L3133	{3}	23,983	49,005	685	684	1,691,158	0.02s	0.05s	0.50s	24.89s	45.96s
L3143	{3}	139,964	417,027	1,201	1,200	61,983,093	0.10s	0.48s	7.58s	2m 4s	6m 3s
L3151	{3}	150,707	496,196	1,501	1,500	—	0.18s	0.50s	9.30s	3.06s	13.98s
L3153	{3}	855,397	3,486,091	1,861	1,860	1,973,610,366	1.24s	4.94s	4m 24s	7m 23s	28m 13s
L3223	{3}	32,750	45,913	2,437	2,436	538,111	0.03s	0.08s	0.27s	13.48s	18.53s
L3523	{3}	2,911,352	4,183,685	220,705	220,704	222,239,487	11.26s	24.86s	2m 26s	> 6h	> 6h
L4133	{3,4}	79,351	158,058	75,157	155,475	277,714,570	0.21s	0.92s	7.30s	—	—
D33	{3}	91,858	215,967	1,522	1,521	—	0.11s	0.40s	2.10s	11.05s	11.05s
D34	{3}	4,043,377	13,749,608	16,381	16,380	—	22.54s	1m 28s	8m 29s	3h 19m	3h 19m
D43	{2,4}	514,120	1,217,310	486,442	1,155,144	—	2.31s	4.70s	1m 32s	—	—
D62	{2,4,6}	254,758	457,795	218,570	389,995	—	1.72s	4.26s	16.55s	—	—
D62	{4,6}	991,861	2,029,546	46,236	60,717	—	3.80s	11.09s	1m 35s	—	—
D62	{6}	3,158,364	7,395,885	5,551	5,550	—	30.20s	1m 11s	8m 53s	—	—

Table 11.2: Runtime of our learning algorithm (column ‘This chapter’), compared to prior state-of-the-art algorithms based on linear programming (‘ZS22’, B. H. Zhang and Sandholm, 2022b) and column generation (‘ZFCS22’, B. H. Zhang, Farina, Celli, and Sandholm, 2022) respectively, on several standard parametric benchmark games. ‘—’: Missing or unknown value.

Column ‘Game’ indicates the game, and the set of players on Team \blacktriangledown . The games belong to

different classes of standard parametric benchmark games, each identified with an alphabetical mnemonic whose starting letters and numbers identify the game class and number of players: **K3** **K4** 3- and 4-player Kuhn poker; **L3** **L4** 3- and 4-player Leduc poker; **D3** **D4** **D6** 3-, 4-, and 6-player liar’s dice. Depending on the game instance, we test with different time sizes, ranging from all-versus-one scenarios, to balanced splits. As usual, we refer the reader to [Appendix A](#) for a description of each game class and their parameters. Columns ‘**Decompos. of Ω_{\blacktriangle}** ’ and ‘**Decompos. of $\Omega_{\blacktriangledown}$** ’ report the total number of simplexes and dimension of the scaled-extension decomposition of Ω_{\blacktriangle} , $\Omega_{\blacktriangledown}$ (see [Remark 11.1](#)) for each game. Column ‘**CCCG22**’ indicates the number of histories in the equivalent game produced by Carminati, Cacciamani, Ciccone, and Gatti (2022)’s algorithm. Column ‘**This chapter**’ reports the time to convergence of the *best* variant of our learning dynamics to an average team exploitability of less than ϵ times the range of payoffs of the game. Column ‘**ZS22**’ reports the time it took ZS22 to compute an equilibrium strategy for team \blacktriangle , to Gurobi’s default precision. Finally, column ‘**ZFCS22**’ reports the time it took ZFCS22 to compute an equilibrium strategy for team \blacktriangle with exploitability of less than ϵ times the range of payoffs of the game. The missing values in that column are due to the fact that the implementation of ZFCS22 by the original authors only supported 3-player games.

Overall, our algorithms based on the team belief DAG are generally 2-3 orders of magnitude faster than ZS22. In games with high parameters (e.g., **K38** and **K3c**), on the other hand, ZFCS22 is significantly more scalable, as it avoids the exponential dependence in the parameters at the cost of requiring the solution to integer programs, for which runtime guarantees are hard to give. Compared to the converted game of Carminati, Cacciamani, Ciccone, and Gatti (2022), our team belief DAG is much smaller, often by orders of magnitude, which allows our algorithms to similarly be faster by orders of magnitude. Since Carminati, Cacciamani, Ciccone, and Gatti (2022) do not give detailed timing results for their implementation for most of the games they tested, we have not included a runtime comparison. However, they reported a runtime of approximately 3 minutes to achieve an exploitability of 0.021 in **L3133**, whereas our algorithm took 0.02 seconds to achieve a lower exploitability of 0.001—a difference of about four orders of magnitude.

Part IV

Beyond perfect rationality

Chapter 12

Positive complexity results for trembling-hand perfect equilibria

The Nash equilibrium prescribes optimal strategies against perfectly rational opponents. However, it is known that it has serious shortcomings when used to prescribe strategies to be deployed against imperfect opponents who may make mistakes. This is problematic when operationalizing equilibria in the real world among imperfect players. In this chapter we will provide new positive complexity results for the computation of a *refinement* of the Nash equilibrium—called an *extensive-form perfect equilibrium (EFPE)*—that guarantees rational strategies even upon mistakes of the opponent.

Undesirable behavior can be prescribed by the Nash equilibrium already in the simple and well-understood case of *two-player zero-sum* games, as we illustrate with the following example.

Example 12.1. As an example, consider the perfect-information game in [Figure 12.1](#).

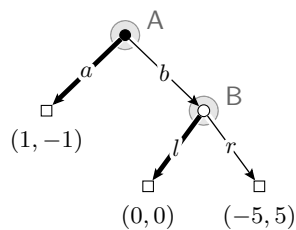


Figure 12.1: Perfect-information game in which a sequentially-irrational Nash equilibrium is highlighted.

The bold lines show one of the Nash equilibria of the game. It does not matter whether the white player acting at B chooses action l or action r because he never gets to action if

the black player acting at A plays rationally. So, in Nash equilibrium, the white player can choose action l . However, if the black player makes a mistake and chooses action b , it would be better for the white player to choose action r (thus getting a payoff of 5 instead of 0). So, in that part of the game where the black player has made the mistake, the white player's Nash equilibrium strategy is not rational. In game-theoretic terms, it is not *sequentially rational*.

While in the particular example of Figure 12.1 the issue could be resolved by using an equilibrium refinement called *subgame perfect* Nash equilibrium, that solution concept is not particularly useful in imperfect-information extensive-form games, where few subgames (nodes of the game tree that are alone in their information set) typically exist.

Example 12.2. For example, consider the example in Figure 12.2.

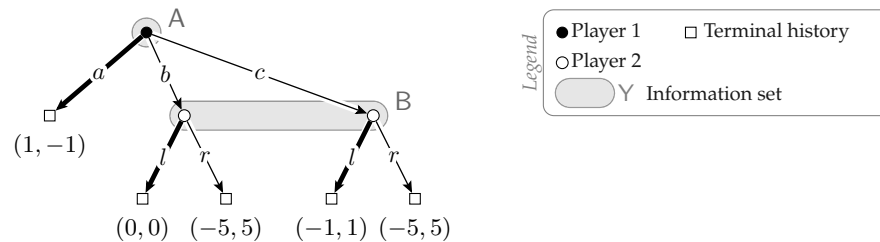


Figure 12.2: Small *imperfect*-information game in which a sequentially-irrational Nash equilibrium is highlighted.

The white player acting at information set B does not have any subgame, and therefore the highlighted sequentially-irrational Nash equilibrium is subgame perfect.

Another refinement of Nash equilibrium is *undominated Nash equilibrium (UNE)*, that is, Nash equilibrium where the pure strategies in the support of the equilibrium do not include strongly dominated strategies. UNE would remove the unreasonable Nash equilibria in the games of both Figures 12.1 and 12.2, but there are other games where UNEs can be sequentially irrational, and in general undomination and sequential rationality are incomparable in the sense that neither implies the other (Miltersen and Sørensen, 2006). We included an example of a situation where an undominated equilibrium is sequentially irrational in an appendix to this chapter, Section 12.A.

Trembling-hand refinements are the standard solution proposed in the literature to avoid the issue of sequential irrationality. One of these, *extensive-form perfect equilibrium (EFPE)* will then be the focus of this chapter. We review trembling-hand refinements, and in particular EFPE, in the next section.

12.1 Preliminaries on trembling-hand refinements

The issue of sequential irrationality stems from the fact that some parts of the game tree are unreachable at equilibrium. For those excluded parts of the game tree, any strategy can be picked without affecting the equilibrium. The idea behind trembling-hand refinements can be conceptualized easily. To avoid sequential irrationality, trembling-hand refinements force all players to explore the whole game tree. They do so by requiring the players to *tremble*, that is, play all actions at all decision points with some strictly positive lower bound probability controlled by a hyperparameter $\epsilon > 0$. A trembling-hand refinement is then any limit point of such Nash equilibria as $\epsilon \rightarrow 0^+$.

Different equilibrium notions differ as to how the lower bounds are set as a function of ϵ . In this chapter we are interested in extensive-form perfect equilibria (EFPE), due to Selten (1975), which define perhaps the conceptually simplest trembling scheme. Specifically, in an EFPE, the trembles are *behavioral*: given $\epsilon > 0$, the perturbed game simply mandates that every action at every decision point must be picked with probability at least ϵ .

Definition 12.1 (Extensive-form perfect equilibrium, EFPE). Given a game Γ , an EFPE is a limit point, as $\epsilon \rightarrow 0^+$, of Nash equilibria of the perturbed $\Gamma(\epsilon)$ in which each player is constrained to play only strategies that put probability at least ϵ on every action.

It is well-known (e.g., Kreps and Wilson, 1982) that every game has at least one EFPE, and that EFPEs are sequentially rational.

Other notions of trembling-hand refinements exist, most notably the *quasi-perfect equilibrium* (QPE) due to van Damme (1984). The definition of QPE is significantly more complicated, and will be given in Definition 13.5 (in Chapter 13), together with its one-sided counterpart called *one-sided QPE* (Farina and Sandholm, 2021a).

12.2 Contributions and related results

The main contribution of this chapter will be to establish the following result:

Finding an EFPE in a two-player game is not harder than finding a Nash equilibrium.

Specifically, in zero-sum games, an EFPE can be found in polynomial time in the size of the input game, and in general-sum games it can be computed using a path-following algorithm with polynomial-time steps.

The result closes a problem left explicitly open by Miltersen and Sørensen (2006), and complements a similar result that was known for the quasi-perfect equilibrium (see Table 12.1).

Solution concept	Two-player general-sum games	Two-player zero-sum games
Nash equilibrium (NE) (Nash, 1950)	PPAD-complete (Daskalakis, Goldberg, and Papadimitriou, 2009)	FP (Romanovskii, 1962) (von Stengel, 1996)
Quasi-perfect equilibrium (QPE) (van Damme, 1984)	PPAD-complete (Miltersen and Sørensen, 2010)	FP (Miltersen and Sørensen, 2010)
Extensive-form perfect eq. (EFPE) (Selten, 1975)	PPAD-complete (this chapter)	FP (this chapter)

Table 12.1: Complexity of computing trembling-hand equilibrium refinements in two-player games.

Building on some of the ideas and techniques introduced in this chapter, in [Chapter 13](#) we will propose the first *scalable* algorithm for computing exact QPE, EFPE, and other refinements of the Nash equilibrium in two-player zero-sum games.

12.3 Positive complexity results for two-player EFPE

In two-player games, a Nash equilibrium can be expressed as a linear complementarity problem (LCP). In this section, we show that the same holds even when each strategy is required to select every action with probability at least ϵ as required to compute the EFPE defined in [Definition 12.1](#).

12.3.1 Behavioral perturbation matrices

The constraint that each action in a strategy $\mathbf{x}_i \in \mathcal{Q}_i$ for Player $i \in \{1, 2\}$ must be selected with probability is at least ϵ is captured in the sequence-form representation by requiring that

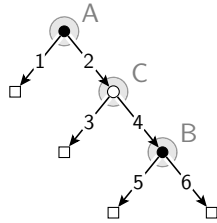
$$\mathbf{x}_i[ja] \geq \epsilon \cdot \mathbf{x}_i[p_j] \quad \forall ja \in \Sigma_i^*.$$

By including the dummy constraint $\mathbf{x}_i[\emptyset] \geq 0$ so as to make the matrix square, we can collect the linear constraints above in matrix form as $\mathbf{R}_i(\epsilon) \mathbf{x}_i \geq \mathbf{0}$, where the matrix $\mathbf{R}_i(\epsilon) \in \mathbb{R}^{\Sigma_i \times \Sigma_i}$ is defined by

$$\mathbf{R}_i(\epsilon)[\sigma, \sigma'] := \begin{cases} 1 & \text{if } \sigma = \sigma' \\ -\epsilon & \text{if } \sigma = ja \text{ and } \sigma' = p_j \\ 0 & \text{otherwise,} \end{cases} \quad \forall \sigma, \sigma' \in \Sigma_i.$$

We call $\mathbf{R}_i(\epsilon)$ the *behavioral perturbation matrix* of Player i . We illustrate these matrices with an example.

Example 12.3. Consider the small game below. The matrices $\mathbf{R}_1(\epsilon)$ and $\mathbf{R}_2(\epsilon)$ in this case are as follows:



$$\mathbf{R}_1(\epsilon) = \begin{pmatrix} \emptyset & A1 & A2 & B5 & B6 \\ 1 & 0 & 0 & 0 & 0 \\ -\epsilon & 1 & 0 & 0 & 0 \\ -\epsilon & 0 & 1 & 0 & 0 \\ 0 & 0 & -\epsilon & 1 & 0 \\ 0 & 0 & -\epsilon & 0 & 1 \end{pmatrix} \begin{matrix} \emptyset \\ A1 \\ A2 \\ B5 \\ B6 \end{matrix} \quad \mathbf{R}_2(\epsilon) = \begin{pmatrix} \emptyset & C3 & C4 \\ 1 & 0 & 0 \\ -\epsilon & 1 & 0 \\ -\epsilon & 0 & 1 \end{pmatrix} \begin{matrix} \emptyset \\ C3. \\ C4 \end{matrix}$$

Assuming that the rows and columns of behavioral perturbation matrices are ordered in accordance with the partial order \preceq of sequences, we remark that behavioral perturbation matrices are lower triangular square matrices having only 0 or $-\epsilon$ as entries. We now show that they are always invertible and that their inverse is a matrix of polynomials in ϵ .

Lemma 12.1. Let $\mathbf{R}_i(\epsilon)$ be a $n \times n$ behavioral perturbation matrix. Then $\mathbf{R}_i(\epsilon)$ is invertible. Furthermore, assuming that the rows and columns of behavioral perturbation matrices are ordered in accordance with the partial order \preceq of sequences, its inverse is

$$\mathbf{R}_i(\epsilon)^{-1} = \mathbf{I} + \epsilon \mathbf{E}(\epsilon),$$

where \mathbf{I} is the identity matrix, and $\mathbf{E}(\epsilon)$ is a lower triangular matrix whose entries are polynomials in ϵ having non-negative integer coefficients.

Proof. By induction on the submatrix formed by the first k rows and k columns. When $k = 1$, the statement is trivial. Now, suppose the theorem holds for the $k \times k$ top-left submatrix of $\mathbf{R}_i(\epsilon)$, denoted $\mathbf{R}_i^k(\epsilon)$; we will show that it holds for the $(k + 1) \times (k + 1)$ top-left submatrix $mR_i^{k+1}(\epsilon)$ of $\mathbf{R}_i(\epsilon)$. By definition of $\mathbf{R}_i(\epsilon)$, we can write

$$\mathbf{R}_i^{k+1}(\epsilon) = \left(\begin{array}{c|c} \mathbf{R}_i^k(\epsilon) & 0 \\ \hline \mathbf{b}(\epsilon)^\top & 1 \end{array} \right), \quad \text{where } \mathbf{b}(\epsilon)^\top = (0, \dots, 0, -\epsilon, 0, \dots, 0).$$

It is immediate to check by direct inspection that the matrix

$$\mathbf{R}_i^{k+1}(\epsilon)^{-1} = \left(\begin{array}{c|c} \mathbf{R}_i^k(\epsilon)^{-1} & 0 \\ \hline -\mathbf{b}(\epsilon)^\top \mathbf{R}_i^k(\epsilon)^{-1} & 1 \end{array} \right)$$

is the inverse of $\mathbf{R}_i^{k+1}(\epsilon)$. Using the inductive hypothesis, we have $\mathbf{R}_i^k(\epsilon)^{-1} = \mathbf{I} + \epsilon \mathbf{E}^k(\epsilon)$, and therefore

$$\mathbf{R}_i^{k+1}(\epsilon)^{-1} = \mathbf{I} + \epsilon \left(\begin{array}{c|c} \mathbf{E}^k(\epsilon) & 0 \\ \hline (-\mathbf{b}(\epsilon)/\epsilon)^\top \mathbf{R}_i^k(\epsilon)^{-1} & 0 \end{array} \right).$$

Since $(-\mathbf{b}(\epsilon)/\epsilon)^\top = (0, \dots, 0, 1, 0, \dots, 0)$ is a non-negative real vector, the row vector $\mathbf{b}'(\epsilon) := (-\mathbf{b}(\epsilon)/\epsilon)^\top \mathbf{R}_i^k(\epsilon)^{-1}$ has entries that are polynomials with non-negative integer coefficients, and so

$$\mathbf{R}_i^{k+1}(\epsilon)^{-1} = \mathbf{I} + \epsilon \left(\begin{array}{c|c} \mathbf{E}^k(\epsilon) & 0 \\ \hline \mathbf{b}'(\epsilon) & 0 \end{array} \right) =: \mathbf{I} + \epsilon \mathbf{E}^{k+1}(\epsilon),$$

where $\mathbf{E}^{k+1}(\epsilon)$ is a lower triangular matrix whose entries are polynomials in ϵ having non-negative integer coefficients, as we wanted to show. \square

Example 12.4. For the matrix $\mathbf{R}_1(\epsilon)$ of Example 12.3, we have:

$$\mathbf{R}_1(\epsilon)^{-1} = \begin{pmatrix} \emptyset & A1 & A2 & B5 & B6 \\ 1 & 0 & 0 & 0 & 0 \\ \epsilon & 1 & 0 & 0 & 0 \\ \epsilon & 0 & 1 & 0 & 0 \\ \epsilon^2 & 0 & \epsilon & 1 & 0 \\ \epsilon^2 & 0 & \epsilon & 0 & 1 \end{pmatrix} \begin{matrix} \emptyset \\ A1 \\ A2 \\ B5 \\ B6 \end{matrix}.$$

12.3.2 EFPE as a trembling linear complementarity problem (LCP)

In this subsection, we show that the problem of finding a Nash equilibrium of a two-player game $\Gamma(\epsilon)$ subject to the constraint that each player pick each action with lower bound probability ϵ can be expressed as a linear complementarity program whose entries are polynomials in ϵ . In what follows, we let $\mathbf{F}_i \in \mathbb{R}^{\mathcal{J}_i \times \Sigma_i}$ and $\mathbf{f}_i \in \mathbb{R}^{\mathcal{J}_i}$ define the sequence-form polytope

$$\mathcal{Q}_i = \{\mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma_i} : \mathbf{F}_i \mathbf{x} = \mathbf{f}_i, \mathbf{x} \geq \mathbf{0}\}$$

of Player $i \in \{1, 2\}$, and \mathbf{U}_i define the payoff matrix of Player i in sequence form.

Lemma 12.2. An EFPE is a limit point as $\epsilon \rightarrow 0^+$ of any solution of the perturbed standard-form LCP

$$P(\epsilon) : \begin{cases} \text{find } z, w \\ \text{s.t. } z^\top w = 0 \\ w = M(\epsilon)z + b \\ z, w \geq 0 \end{cases}$$

where, using the shorthand $\underline{\mathbf{R}}_1 \equiv \mathbf{R}_1(\epsilon)$, $\underline{\mathbf{R}}_2 \equiv \mathbf{R}_2(\epsilon)$ for brevity,

$$z = \begin{pmatrix} \tilde{x}_1 \\ \tilde{x}_2 \\ v_1^+ \\ v_1^- \\ v_2^+ \\ v_2^- \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 0 \\ f_1 \\ -f_1 \\ f_2 \\ -f_2 \end{pmatrix},$$

$$M(\epsilon) = \begin{pmatrix} 0 & -\underline{\mathbf{R}}_1^{-\top} \mathbf{U}_1 \underline{\mathbf{R}}_2^{-1} & \underline{\mathbf{R}}_1^{-\top} \mathbf{F}_1^\top & -\underline{\mathbf{R}}_1^{-\top} \mathbf{F}_1^\top & 0 & 0 \\ -\underline{\mathbf{R}}_2^{-\top} \mathbf{U}_2^\top \underline{\mathbf{R}}_1^{-1} & 0 & 0 & 0 & \underline{\mathbf{R}}_2^{-\top} \mathbf{F}_2^\top & -\underline{\mathbf{R}}_2^{-\top} \mathbf{F}_2^\top \\ -\mathbf{F}_1 \underline{\mathbf{R}}_1^{-1} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{F}_1 \underline{\mathbf{R}}_1^{-1} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\mathbf{F}_2 \underline{\mathbf{R}}_2^{-1} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{F}_2 \underline{\mathbf{R}}_2^{-1} & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Proof. The problem of finding the best response of Player i in the perturbed game $\Gamma(\epsilon)$ where every action is selected with probability at least ϵ is a linear problem

$$BR_i(\epsilon) : \begin{cases} \max_{x_i} & x_i^\top \mathbf{U}_i x_{-i} \\ \text{s.t.} & \mathbf{F}_i x_i = f_i \\ & \mathbf{R}_i(\epsilon) x_i \geq 0 \end{cases}$$

Notice that $\mathbf{R}_i(\epsilon)$ is invertible ([Lemma 12.1](#)), hence by changing variable $\tilde{x}_i := \mathbf{R}_i(\epsilon) x_i$, we find the equivalent problem:

$$BR_i(\epsilon) : \begin{cases} \max_{\tilde{x}_i} & \tilde{x}_i^\top \mathbf{R}_i(\epsilon)^{-\top} \mathbf{U}_i \mathbf{R}_{-i}(\epsilon)^{-1} \tilde{x}_{-i} \\ \text{s.t.} & \textcircled{1} \mathbf{F}_i \mathbf{R}_i(\epsilon)^{-1} \tilde{x}_i = f_i \\ & \textcircled{2} \tilde{x}_i \geq 0. \end{cases}$$

Taking the dual:

$$\overline{\text{BR}}_i(\epsilon) : \begin{cases} \min_{v_i} & \mathbf{f}_i^\top v_i \\ \text{s.t.} & \textcircled{3} \quad \mathbf{R}_i(\epsilon)^{-\top} \mathbf{F}_i^\top v_i \geq \mathbf{R}_i(\epsilon)^{-\top} \mathbf{U}_i \mathbf{R}_{-i}(\epsilon)^{-1} \tilde{\mathbf{x}}_{-i} \\ & \textcircled{4} \quad v_i \text{ free in sign.} \end{cases}$$

Finally, complementarity slackness requires that

$$\textcircled{5} \quad \tilde{\mathbf{x}}_i^\top (\mathbf{R}_i(\epsilon)^{-\top} \mathbf{F}_i^\top v_i - \mathbf{R}_i(\epsilon)^{-\top} \mathbf{U}_i \mathbf{R}_{-i}(\epsilon)^{-1} \tilde{\mathbf{x}}_{-i}) = 0.$$

Solving problem $\text{BR}_i(\epsilon)$ or $\overline{\text{BR}}_i(\epsilon)$ is equivalent to solving the feasibility problem defined by constraints $\textcircled{1}$ to $\textcircled{5}$. It is now easy to see that we can cast the problem of satisfying conditions $\textcircled{1}$ to $\textcircled{5}$ for both players as a standard-form LCP whose parameters are as defined in this lemma. \square

In particular, we show that a polynomial (in the game size) number of bits is sufficient to represent the polynomial entries of the LCP introduced in [Lemma 12.2](#).

Lemma 12.3. Consider the LCP formulation of [Lemma 12.2](#), where ϵ is treated as a symbolic variable, so that the entries of $\mathbf{M}(\epsilon)$ are polynomials in ϵ . A number of bits polynomial in the input game size is sufficient to store all coefficients appearing in $P(\epsilon)$.

Proof. Consider the LCP formulation of [Lemma 12.2](#). We begin by showing that each coefficient appearing in $P(\epsilon)$ requires a polynomial amount of memory to be store. This property trivially holds for vector \mathbf{b} . On the other hand, all numbers appearing in matrix $\mathbf{M}(\epsilon)$ are either zeros, or they are obtained by multiplying two or more of the following matrices together: $\mathbf{R}_1(\epsilon)^{-\top}, \mathbf{R}_2(\epsilon)^{-\top}, \mathbf{U}_1, \mathbf{U}_2^\top, \mathbf{F}_1^\top, \mathbf{F}_2^\top$. Hence, as long as each of the coefficients appearing in the above-mentioned matrices requires a polynomial number of bits in the input game size, the property is true. This is clearly true for $\mathbf{U}_1, \mathbf{U}_2^\top, \mathbf{F}_1, \mathbf{F}_2^\top$, so we are left with the task of proving this property for $\mathbf{R}_1(\epsilon)^{-1}$ and $\mathbf{R}_2(\epsilon)^{-1}$. However, since $\det \mathbf{R}_1(\epsilon) = 1$ (indeed, notice that $\mathbf{R}_1(\epsilon)$ is lower triangular), using the adjoint matrix theorem and the Leibniz formula for the determinant, we conclude that each entry in $\mathbf{R}_1(\epsilon)^{-1}$ is obtained as a sum of $|\Sigma_1|!$ terms, each of which is a product of $|\Sigma_1|$ entries of $\mathbf{R}_1(\epsilon)$, since $|\Sigma_1|$ is the dimension of $\mathbf{R}_1(\epsilon)$ (the same holds for $\mathbf{R}_2(\epsilon)$). Therefore, the property holds, showing that each coefficient in $\mathbf{M}(\epsilon)$ and \mathbf{b} requires a polynomial amount of memory to be stored.

We now show that the maximum degree appearing in $\mathbf{M}(\epsilon)$ is $2|\Sigma_1|$. This is a consequence of the observation above: since each entry in $\mathbf{R}_1(\epsilon)^{-1}$ is obtained as sum of $|\Sigma_1|!$ terms, each of which is a product of $|\Sigma_1|$ entries of $\mathbf{R}_1(\epsilon)$, the maximum degree appearing in $\mathbf{R}_1(\epsilon)^{-1}$ is $|\Sigma_1|$. Now, since each element of $\mathbf{M}(\epsilon)$ is obtained from the product of at most two matrices dependent on ϵ , the maximum degree appearing in $\mathbf{M}(\epsilon)$ (and therefore in $P(\epsilon)$) is $2|\Sigma_1|$.

Thus, we have a polynomial amount of coefficients to store, each of which requires a

polynomial amount of memory. The required space is therefore polynomial. \square

Finally, we show that for ϵ sufficiently small, the LCP of [Lemma 12.2](#) is feasible.

12.3.3 Existence of a negligible positive perturbation (NPP)

Before turning our attention to the *computational* aspects of finding an EFPE, we introduce some general concepts, pertaining to perturbed linear optimization problems. While we target the development of these concepts with our specific use-case in mind, it should be noted that this section's definitions and lemmas are of broader interest, being applicable to any linear program (LP) or LCP. We recall that a *basis* \mathcal{B} for a standard-form LP with constraints $\mathbf{M}\mathbf{x} = \mathbf{b}$ or a standard-form LCP with linear equality constraints $\mathbf{w} = \mathbf{M}\mathbf{z} + \mathbf{b}$ is a set of linearly independent columns of \mathbf{M} such that the associated solution (called *basic solution*) is feasible.

Definition 12.2 (Negligible positive perturbation, NPP). Let $P(\epsilon)$ be an LCP dependent on some perturbation ϵ . The value $\epsilon^* > 0$ is a *negligible positive perturbation* (NPP) if any optimal basis \mathcal{B} for $P(\epsilon^*)$ is optimal for $P(\epsilon)$, for all $\epsilon \in (0, \epsilon^*]$.

Definition 12.3 (Optimality certificate for a basis). Given an LCP $P(\epsilon)$, and a basis \mathcal{B} for it, we call the finite-dimensional column vector $\gamma_{\mathcal{B}}(\epsilon)$ an *optimality certificate* for \mathcal{B} if for all $\epsilon > 0$

$$\gamma_{\mathcal{B}}(\epsilon) \geq 0 \iff \mathcal{B} \text{ is optimal for } P(\epsilon).$$

Lemma 12.4. In the case of a perturbed LCP in standard form

$$P(\epsilon) : \begin{cases} \text{find} & \mathbf{z}, \mathbf{w} \\ \text{s.t.} & \textcircled{1} \quad \mathbf{z}^\top \mathbf{w} = 0 \\ & \textcircled{2} \quad \mathbf{w} = \mathbf{M}(\epsilon)\mathbf{z} + \mathbf{b}(\epsilon) \\ & \textcircled{3} \quad \mathbf{z}, \mathbf{w} \geq \mathbf{0} \end{cases}$$

an optimality certificate for the complementary basis \mathcal{B} is

$$\gamma_{\mathcal{B}}(\epsilon) = \mathbf{B}(\epsilon)^{-1} \mathbf{b}(\epsilon)$$

where \mathbf{B} is the basis matrix corresponding to \mathcal{B} .

Proof. Since the basis \mathcal{B} is complementary by hypothesis, constraint ① is always satisfied. Constraint ② is satisfied by the definition of $\mathbf{B}(\epsilon)$. Constraint ③ is satisfied if and only if $\mathbf{B}(\epsilon)^{-1}\mathbf{b}(\epsilon) = \gamma_{\mathcal{B}}(\epsilon) \geq \mathbf{0}$. \square

Finally, before proceeding, we introduce three lemmas that come in handy when dealing with optimality certificates. Indeed, it is often the case that $\gamma_{\mathcal{B}}(\epsilon)$ has polynomial or rational functions (with respect to ϵ) as entries.

Lemma 12.5. Let $p(\epsilon) := a_0 + a_1\epsilon^1 + \cdots + a_n\epsilon^n$ be a real polynomial such that $a_0 \neq 0$, and let $\mu := \max_i |a_i|$. Then $p(\epsilon)$ has the same sign of a_0 for all $0 \leq \epsilon \leq \epsilon^*$, where

$$\epsilon^* := |a_0|/(\mu + |a_0|).$$

Proof. We prove that when $a_0 > 0$, $p(\epsilon)$ is positive for all $0 \leq \epsilon \leq \epsilon^*$. Indeed,

$$p(\epsilon) = a_0 + a_1\epsilon^1 + \cdots + a_n\epsilon^n > a_0 - \mu\epsilon \sum_{i=0}^{\infty} \epsilon^i = a_0 - \frac{\mu\epsilon}{1-\epsilon}.$$

Since $\epsilon \leq \epsilon^* := a_0/(\mu + a_0)$ we have

$$p(\epsilon) > a_0 - \frac{\mu a_0}{\mu + a_0 - a_0} = 0.$$

The case $a_0 < 0$ is symmetric, and in that case $p(\epsilon)$ is negative for all $0 \leq \epsilon \leq \epsilon^*$. \square

As a corollary, we extend the result of [Lemma 12.5](#) to rational functions.

Lemma 12.6. Let

$$p(\epsilon) := \frac{a_0 + a_1\epsilon^1 + \cdots + a_n\epsilon^n}{b_0 + b_1\epsilon^1 + \cdots + b_m\epsilon^m}$$

be a rational function such that $a_0, b_0 \neq 0$, and let $\mu_a := \max_i |a_i|$, $\mu_b := \max_i |b_i|$. Then $p(\epsilon)$ has the same sign of a_0/b_0 for all $0 \leq \epsilon \leq \epsilon^*$, where

$$\epsilon^* := \min \left\{ \frac{|a_0|}{\mu_a + |a_0|}, \frac{|b_0|}{\mu_b + |b_0|} \right\}.$$

Proof. The proof follows immediately by applying [Lemma 12.5](#) to the numerator and the denominator of $p(\epsilon)$. \square

Lemma 12.7. Let

$$p(\epsilon) := \frac{a_0 + a_1\epsilon^1 + \cdots + a_n\epsilon^n}{b_0 + b_1\epsilon^1 + \cdots + b_m\epsilon^m}$$

be a rational function with integer coefficients, where the denominator is not identically zero; let $\mu_a := \max_i |a_i|$, $\mu_b := \max_i |b_i|$, $\mu := \max\{\mu_a, \mu_b\}$ and $\epsilon^* := 1/(2\mu)$. Then exactly one of the following holds:

- $p(\epsilon^*) = 0$ for all $0 < \epsilon \leq \epsilon^*$,
- $p(\epsilon^*) > 0$ for all $0 < \epsilon \leq \epsilon^*$,
- $p(\epsilon^*) < 0$ for all $0 < \epsilon \leq \epsilon^*$.

Proof. If the numerator of $p(\epsilon)$ is identically zero, the thesis follows trivially, as $p(\epsilon) = 0$ for all ϵ , while the denominator is never zero for all $0 < \epsilon \leq \epsilon^*$ due to [Lemma 12.5](#). If, on the other hand, the numerator of $p(\epsilon)$ is not identically zero, there exist j and x , both non-negative, such that p can be written as

$$p(\epsilon) = \frac{\epsilon^q(a_q + a_{q+1}\epsilon + \cdots + a_n\epsilon^{n-q})}{\epsilon^r(b_r + b_{r+1}\epsilon^1 + \cdots + b_n\epsilon^{n-r})},$$

with $a_q, b_r \neq 0$. Since a_q and b_r are integer, $|a_q|, |b_r| \geq 1$ and we have

$$\epsilon^* = \frac{1}{2\mu} \leq \min \left\{ \frac{|a_q|}{\mu_a + |a_q|}, \frac{|b_r|}{\mu_b + |b_r|} \right\}.$$

Using [Lemma 12.6](#) we conclude that the sign of $p(\epsilon)$ is constant, and equal to that of a_q/b_r , for all $0 < \epsilon \leq \epsilon^*$. \square

Theorem 12.1. Given a (general-sum) two-player game Γ with $\nu := \max_{i \in \{1,2\}, j \in \mathcal{J}_i} |\mathcal{A}_j|$, the problem $P(\epsilon)$ of computing any EFPE for Γ admits an NPP $\epsilon^* \leq 1/\nu$ that can be computed from Γ in polynomial time. In particular, $\epsilon^* = 1/V^*$, where the integer value V^* can be represented in memory with a number of bits polynomial in the input game size.

Proof. We illustrate the steps that lead to the determination of such V^* . The central idea is as follows: we want to determine ϵ^* so that, whatever the feasible base \mathcal{B} for $P(\epsilon^*)$ may be, the optimality certificate for ϵ^* is positive for all $\epsilon \in (0, \epsilon^*]$. Indeed, it is immediate to see that such ϵ^* is necessarily an NPP.

- **Optimality certificate.** We begin by studying the optimality certificate for the LCP $P(\epsilon)$, that is, by [Lemma 12.4](#),

$$\mathbf{B}(\epsilon)^{-1}\mathbf{b}(\epsilon) \geq \mathbf{0},$$

where $\mathbf{B}(\epsilon)$ is base matrix corresponding to the feasible base \mathcal{B} found by Lemke's algorithm. Introducing $\mathbf{C}(\epsilon) = \text{cof } \mathbf{B}(\epsilon)$, the cofactor matrix of matrix $\mathbf{B}(\epsilon)$, and leveraging the well-known identity $\mathbf{B}(\epsilon)^{-1} = \mathbf{C}(\epsilon)^\top / \det \mathbf{B}(\epsilon)$, we can rewrite the optimality certificate above as

$$\frac{\mathbf{C}(\epsilon)^\top \mathbf{b}(\epsilon)}{\det \mathbf{B}(\epsilon)} \geq \mathbf{0}.$$

The vectorial condition above is equivalent to a system of m scalar conditions, each of the form

$$g_k(\epsilon) = \frac{\mathbf{c}_k(\epsilon)^\top \mathbf{b}(\epsilon)}{\det \mathbf{B}(\epsilon)} \geq 0,$$

where $\mathbf{c}_k(\epsilon)$ is the k -th row of $\mathbf{C}(\epsilon)^\top$. Evidently, $g_k(\epsilon)$ is a rational function in ϵ , having only integer coefficients, for all $k = 1, \dots, m$.

- **Denominator coefficients.** We now give an upper bound on the coefficients of the denominator of $g_k(\epsilon)$, that is $\det \mathbf{B}(\epsilon)$. Let V_B be the largest coefficient that could potentially appear in $\mathbf{B}(\epsilon)$ and $\mathbf{b}(\epsilon)$, and let d be the largest polynomial degree appearing in $\mathbf{B}(\epsilon)$. Notice that $d \in \mathcal{O}(\text{poly}(m))$. By using Hadamard's inequality, we can write

$$\text{coeff}(\det \mathbf{B}(\epsilon)) \leq m^{m/2} V_B^m \text{coeff}((1 + \epsilon + \dots + \epsilon^d)^m),$$

where $\text{coeff}(\cdot)$ is the largest coefficient of its polynomial argument. Since $\text{coeff}((1 + \epsilon + \dots + \epsilon^d)^m) \leq d^m$, we have

$$\text{coeff}(\det \mathbf{B}(\epsilon)) \leq V_D := m^{m/2} (dV_B)^m.$$

Notice that this bound is valid for all possible basis matrices $\mathbf{B}(\epsilon)$. Furthermore, notice that

$$\log V_D = m/2 \log m + m \log d + m \log V_B,$$

and by [Lemma 12.3](#) we conclude that V_D requires a number of bits polynomial in the input game size in order to be stored in memory.

- **Numerator coefficients.** Since the elements of $c_i(\epsilon)$ are cofactors for $\mathbf{B}(\epsilon)$, they are upper-bounded by $\det \mathbf{B}(\epsilon)$, which in turn is upper-bounded by V_D . Therefore,

$$\text{coeff}(c_i(\epsilon)^\top \mathbf{b}(\epsilon)) \leq V_N := V_B V_D.$$

Again, it is worthwhile to notice that this bound is valid for all possible basis matrices $\mathbf{B}(\epsilon)$.

- **Wrapping up.** Define $V^* = 2 \max\{V_N, V_D\} = 2V_B V_D$. We now argue that $\epsilon^* = 1/V^*$ is an NPP for $P(\epsilon)$. Indeed, let \mathcal{B}^* be a feasible basis (since $\epsilon^* \leq 1/m$, \mathcal{B} always exists—see [Lemma 12.9](#)) for $P(\epsilon^*)$, and let $B(\epsilon^*)$ be the corresponding base matrix. Being \mathcal{B} feasible for $P(\epsilon^*)$, each row g_k in the optimality certificate is non-negative when evaluated at ϵ^* for all k . Therefore, we know from [Lemma 12.7](#) that $g_k(\epsilon) \geq 0$ for all $\epsilon \in (0, 1/V^*] = (0, \epsilon^*]$. Hence, the optimality certificate for \mathcal{B} is non-negative for all $0 < \epsilon \leq \epsilon^*$, which is equivalent to say that ϵ^* is an NPP. Finally, note that $V^* = 2V_B V_D$ be stored in memory with a number of bits polynomial in the game size. This completes the proof. \square

12.3.4 Computation of extensive-form perfect equilibria

We finally delve into the computational details of finding extensive-form perfect equilibria. The central result of this section ([Theorem 12.1](#)) roughly states that the EFPE LCP ([Lemma 12.2](#)) always admits a “small” NPP. Leveraging this fact, we quickly derive a path-following algorithm for the computation of EFPE in general-sum games in which each pivoting step has a polynomial-time cost ([Theorem 12.2](#)), and a polynomial-time algorithm for the zero-sum counterpart ([Theorem 12.3](#)). These two algorithms put the two search problems in PPAD and FP classes, respectively.

To establish the result, we will make use of Lemke’s algorithm (Lemke, 1970), a classic iterative algorithm that is able to solve a linear complementarity problem, provided it satisfies the following conditions.

Lemma 12.8 (Theorem 4.1, Koller, Megiddo, and von Stengel, 1996). Consider a linear complementarity problem in standard form

$$\left\{ \begin{array}{ll} \text{find} & \mathbf{z}, \mathbf{w} \\ \text{s.t.} & \textcircled{1} \quad \mathbf{z}^\top \mathbf{w} = 0 \\ & \textcircled{2} \quad \mathbf{w} = \mathbf{M}\mathbf{z} + \mathbf{b} \\ & \textcircled{3} \quad \mathbf{z}, \mathbf{w} \geq \mathbf{0}. \end{array} \right.$$

If the two conditions

(a) $\mathbf{z}^\top \mathbf{M}\mathbf{z} \geq \mathbf{0}$ for all $\mathbf{z} \geq \mathbf{0}$, and

(b) $\mathbf{z} \geq \mathbf{0}$, $\mathbf{M}\mathbf{z} \geq \mathbf{0}$, $\mathbf{z}^\top \mathbf{M}\mathbf{z} = 0 \implies \mathbf{z}^\top \mathbf{b} \geq 0$,

are satisfied, then Lemke's algorithm computes a solution of the LCP.

We start by observing that, as long as the perturbation ϵ is "reasonably small", the LCP defined in [Lemma 12.2](#) always admits a solution.

Lemma 12.9. If $0 < \epsilon \leq 1/\nu$, where $\nu := \max_{i \in \{1,2\}, j \in \mathcal{J}_i} |\mathcal{A}_j|$ is the maximum number of actions available at any information set of the game. Lemke's algorithm always finds a solution for $P(\epsilon)$.

Proof. We follow the same proof structure as that in Koller, Megiddo, and von Stengel, 1996, Section 4. In particular, we prove that if $\mathbf{U}_1, \mathbf{U}_2 < \mathbf{0}$, then conditions (a) and (b) of [Lemma 12.8](#) hold for all problems $P(\epsilon)$ defined in [Lemma 12.2](#). Notice that we can always assume $\mathbf{U}_1, \mathbf{U}_2 < \mathbf{0}$ without loss of generality, as we can apply an offset to the payoff matrices leaving the game unaltered.

- **Condition (a).** We need to show that when $\mathbf{U}_1, \mathbf{U}_2 < \mathbf{0}$, then $\mathbf{z}^\top \mathbf{M}(\epsilon)\mathbf{z} \geq 0$ for all $\mathbf{z} \geq \mathbf{0}$. We have:

$$\mathbf{z}^\top \mathbf{M}(\epsilon)\mathbf{z} = \tilde{\mathbf{x}}_1^\top \mathbf{R}_1(\epsilon)^{-\top} (-\mathbf{U}_1 - \mathbf{U}_2) \mathbf{R}_2(\epsilon)^{-1} \tilde{\mathbf{x}}_2.$$

Substituting $\mathbf{U} = -\mathbf{U}_1 - \mathbf{U}_2 > \mathbf{0}$ and using [Lemma 12.1](#):

$$\begin{aligned} \mathbf{z}^\top \mathbf{M}(\epsilon)\mathbf{z} &= \tilde{\mathbf{x}}_1^\top (\mathbf{I} + \epsilon \mathbf{E}_1(\epsilon))^\top \mathbf{U} (\mathbf{I} + \epsilon \mathbf{E}_2(\epsilon)) \tilde{\mathbf{x}}_2 \\ &\geq \tilde{\mathbf{x}}_1^\top \mathbf{U} \tilde{\mathbf{x}}_2. \end{aligned} \tag{12.1}$$

When $\mathbf{z} \geq \mathbf{0}$, then $\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2 \geq \mathbf{0}$ and we conclude that $\tilde{\mathbf{x}}_1^\top \mathbf{U} \tilde{\mathbf{x}}_2 \geq 0$, which implies the thesis.

- **Condition (b).** We already proved ([Equation \(12.1\)](#)) that

$$\mathbf{z}^\top \mathbf{M}(\epsilon)\mathbf{z} \geq \tilde{\mathbf{x}}_1^\top \mathbf{U} \tilde{\mathbf{x}}_2,$$

where $\mathbf{U} > \mathbf{0}$. In order for $\mathbf{z}^\top \mathbf{M}(\epsilon) \mathbf{z}$ to be zero given $\mathbf{z} \geq \mathbf{0}$, it is necessary that $\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2 = \mathbf{0}$. Defining $\mathbf{v}_1 = \mathbf{v}_1^+ - \mathbf{v}_1^-$ and $\mathbf{v}_2 = \mathbf{v}_2^+ - \mathbf{v}_2^-$, we have

$$\mathbf{M}(\epsilon) \mathbf{z} = \begin{pmatrix} \mathbf{R}_1(\epsilon)^{-\top} \mathbf{F}_1^\top \mathbf{v}_1 \\ \mathbf{R}_2(\epsilon)^{-\top} \mathbf{F}_2^\top \mathbf{v}_2 \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \quad \mathbf{z}^\top \mathbf{b} = \mathbf{b}^\top \mathbf{z} = \mathbf{f}_1^\top \mathbf{v}_1 + \mathbf{f}_2^\top \mathbf{v}_2.$$

Hence, in order to complete the proof, it suffices to show that

$$\mathbf{R}_i(\epsilon)^{-\top} \mathbf{F}_i^\top \mathbf{v}_i \geq \mathbf{0} \implies \mathbf{f}_i^\top \mathbf{v}_i \geq 0 \quad (i \in \{1, 2\}).$$

To this end, we consider the following linear optimization problem $Y_i(\epsilon)$, and its dual $\bar{Y}_i(\epsilon)$:

$$Y_i(\epsilon) : \begin{cases} \max_{\tilde{\mathbf{x}}_i} & 0 \\ \text{s.t.} & \mathbf{F}_i \mathbf{R}_i(\epsilon)^{-1} \tilde{\mathbf{x}}_i = \mathbf{f}_i, \\ & \tilde{\mathbf{x}}_i \geq \mathbf{0} \end{cases}$$

$$\bar{Y}_i(\epsilon) : \begin{cases} \min_{\mathbf{v}_i} & \mathbf{f}_i^\top \mathbf{v}_i \\ \text{s.t.} & \mathbf{R}_i(\epsilon)^{-\top} \mathbf{F}_i^\top \mathbf{v}_i \geq \mathbf{0} \end{cases}.$$

Notice that $Y_i(\epsilon)$ is feasible since $\epsilon \leq 1/\nu$ by hypothesis (indeed, the strategy that picks every action uniformly at random satisfies that each action is selected with probability $\geq 1/\nu$). By the strong duality theorem, we conclude that whenever the constraint of the dual problem is satisfied, the objective value is non-negative, that is $\mathbf{R}_i(\epsilon)^{-\top} \mathbf{F}_i^\top \mathbf{v}_i \geq \mathbf{0} \implies \mathbf{f}_i^\top \mathbf{v}_i \geq 0$ as we aimed to show. \square

We remark that when ϵ is a given value, the task of finding an NE of $P(\epsilon)$ has a powerful interpretation. Indeed, it captures the situation in which the actions of a player are subject to execution uncertainty and therefore a player cannot perfectly control their actions.

Theorem 12.2. The problem of computing an EFPE of a general-sum two-player game Γ is PPAD-complete.

Proof. Let $\epsilon^* = 1/V^*$ be an NPP as defined in [Theorem 12.1](#), and let \mathcal{B} be a feasible base for the (numerical) problem $P(\epsilon^*)$, found using Lemke's algorithm. Since ϵ^* is an NPP, the pair of strategies $(\mathbf{x}_1^*, \mathbf{x}_2^*)$ corresponding to \mathcal{B} retain their feasibility with respect to the LCP $P(\epsilon)$ as

$\epsilon \rightarrow 0^+$, meaning that (x_1^*, x_2^*) is in fact an EFPE.

Furthermore, given that V^* requires a number of bits polynomial in the input game size, each iteration of Lemke's algorithm takes time polynomial in the game size. This proves that the algorithm described is a path-following algorithm requiring a polynomial-time cost at each step, and therefore the problem of finding an EFPE in two-player games is in the PPAD class. The hardness easily follows from the fact that EFPE is a refinement of Nash equilibrium, an EFPE always exists, and finding a Nash is PPAD-complete. Therefore, if finding an EFPE were not PPAD-hard, then one could use the EFPE-finding algorithm with the aim of finding an NE and therefore not even finding an NE would be PPAD-hard. This concludes the proof. \square

We remark that the proof of theorem above also applies for an arbitrary ϵ (potentially non-NPP), showing that finding an NE for any $\epsilon < \max_{i \in \{1,2\}, j \in \mathcal{J}_i} |A_j|$ is in the PPAD class. We summarize the procedure to find an EFPE of general-sum games in [Algorithm 12.1](#).

Algorithm 12.1: Find-EFPE

- 1 Compute ϵ^* from Γ as in the proof of [Theorem 12.1](#)
 - 2 Determine a basis \mathcal{B} for the numerical LCP $P(\epsilon^*)$
 - 3 Let $\mathbf{B}(\epsilon)$ be the base matrix corresponding to \mathcal{B} in $P(\epsilon)$, as ϵ varies
 - [▷ Since $\mathbf{B}(\epsilon)^{-1}\mathbf{b}$ is a rational bounded function in a neighborhood of 0, $\mathbf{B}(0)^{-1}\mathbf{b}$ exists]
 - 4 $(\tilde{x}_1, \tilde{x}_2, \mathbf{v}_1^+, \mathbf{v}_1^-, \mathbf{v}_2^+, \mathbf{v}_2^-)^\top = \mathbf{B}(0)^{-1}\mathbf{b}$
 - [▷ Note that $\mathbf{R}_1(0) = \mathbf{R}_2(0) = \mathbf{I}$, so $\tilde{x}_1 = x_1, \tilde{x}_2 = x_2$]
 - 5 **return** the pair of strategies $(\tilde{x}_1, \tilde{x}_2)$
-

12.3.5 Polynomial-time computation in zero-sum games

We now show that [Algorithm 12.1](#) requires polynomial time when the game is zero sum.

Theorem 12.3. The problem of computing an EFPE of a zero-sum two-player game Γ can be solved in polynomial time in the size of the input game.

Proof. Like in [Theorem 12.2](#), we can easily extract an EFPE by looking at the feasible matrix \mathcal{B} which solves the (numerical) LCP $P(\epsilon^*)$. However, in the zero-sum setting, we do not need to use Lemke's algorithm. Indeed, notice that in zero-sum two-player games, matrix $\mathbf{M}(\epsilon)$ as defined in [Lemma 12.2](#) is such that $\mathbf{M}(\epsilon) + \mathbf{M}(\epsilon)^\top = \mathbf{0}$ for all ϵ , because $\mathbf{U}_2 = -\mathbf{U}_1^\top$. Therefore, the complementarity condition can be rewritten as

$$0 = \mathbf{z}^\top (\mathbf{M}(\epsilon)\mathbf{z} + \mathbf{b}(\epsilon)) = \frac{1}{2} \mathbf{z}^\top (\mathbf{M}(\epsilon) + \mathbf{M}(\epsilon)^\top) \mathbf{z} + \mathbf{z}^\top \mathbf{b}(\epsilon) = \mathbf{z}^\top \mathbf{b}(\epsilon),$$

a linear condition instead of a quadratic one. This shows that when the game is zero-sum, the LCP is actually an LP. As such, a basis for the LCP of Algorithm 12.1 can be computed in polynomial time, leading to an overall polynomial time algorithm. This completes the proof. \square

We remark that the technique used to establish the above result yields a polynomial-time method for computing EFPE. However, the method scales unfavorably in practice due to the need to operate with extremely small rational numbers. A significantly more scalable approach will be presented in Chapter 13.



12.A Appendix: Undomination does not prevent sequential irrationality

One might believe that the problem of sequential irrationality is that of picking dominated strategies. So, one might be inclined to look into the problem of finding a Nash equilibrium whose support does not include any (weakly) dominated strategy (the concept is not immediately well defined, but for the purposes of this discussion let's restrict ourselves to Nash equilibria in deterministic strategies).

Example 12.5. Figure 12.3 shows a modification of the classic *Guess-the-Ace* game introduced by Miltersen and Sørensen (2006).

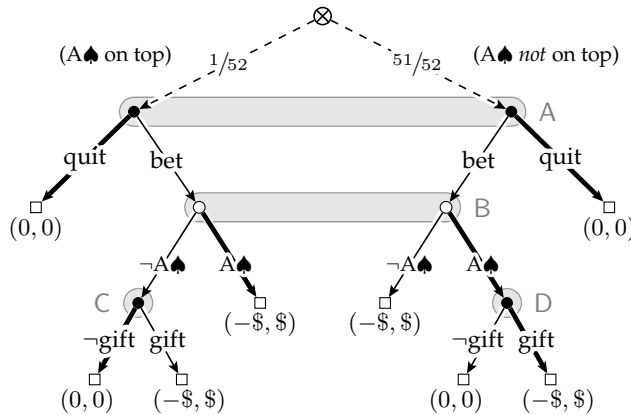


Figure 12.3: Modified *Guess-the-Ace* games. The highlighted equilibrium is undominated, and yet sequentially irrational.

In regular *Guess-the-Ace*, at the start a standard 52-card deck is perfectly shuffled, face down, by a dealer. Then, Player 1 can decide whether to immediately end the game, at which point no money is transferred between the players, or offer \$1000 to Player 2 if they can correctly guess whether the top card of the shuffled deck is the ace of spaces or not. If Player 2 guesses correctly, the \$1000 get transferred from Player 1 to Player 2; if not, no money is transferred.

In Figure 12.3, again due to Miltersen and Sørensen (2006), the *Guess-the-Ace* game is slightly modified in that, when Player 2 guesses wrong, Player 1 can decide whether they still want to give \$1000 to Player 2 out of the kindness of their heart or not. By introducing that possibility, action ' $\neg A_{\spadesuit}$ ' is not strictly dominating anymore, because Player 2 might still hope that the second gift of \$1000 is given only when the insensible guess ' A_{\spadesuit} ' is made.

The example above implies the following general principle.

Observation 12.1. Undomination does not prevent a player from playing risky actions, “hoping” for an opponent’s mistake.

Chapter 13

Computing exact trembling-hand refinements in two-player zero-sum games at scale

In this chapter we develop the first technique for computing exact trembling-hand equilibrium refinements—including extensive-form perfect (Chapter 12) and quasi-perfect equilibria—in large two-player zero-sum games. For the first time, it enables computing exact, refined, sequentially-rational Nash equilibrium strategies in games with up to hundreds of millions of terminal states, 4-5 orders of magnitude larger than what prior techniques could handle.

13.1 Related work

An alternative algorithm to compute an exact quasi-perfect equilibrium (QPE) is a simplex algorithm variant that deals symbolically with the perturbation using the lexico-minimum ratio test (Miltersen and Sørensen, 2010). It was not known if, in practice, it can scale up to large instances. Our experiments show that it does not. While in principle also an extensive-form perfect equilibrium (EFPE) can be computed using a simplex algorithm that deals with the perturbation symbolically, it is not even clear whether it can run in polynomial space. In summary, although there is agreement that Nash equilibrium refinements can play an important role even in two-player zero-sum games, prior algorithms do not scale in practice.

Some algorithms have been proposed for computing approximate trembling-hand equilibria resorting to regret-minimization techniques (Farina, Kroer, and Sandholm, 2017) or to smoothing methods paired with bilinear saddle-point techniques (Kroer, Farina, and Sandholm, 2017). Those algorithms do not provide any guarantee of finding or approximating actual QPEs or EFPEs.

Rather, they provide approximate solutions to approximate solution concepts.

Finally, we mention *proper equilibria* (PEs), proposed by Myerson (1978). PEs form a non-empty subset of EFPEs, with the additional requirement that the worse an action is for a player, the lower the agent's tremble probability on that action must be. Therefore, the trembles are a function of the strategies of the players. This potentially complicates equilibrium finding. It is unknown whether PEs can be efficiently found in imperfect-information extensive-form games, and the techniques in this chapter do not apply to PE.

13.2 Refined strategies as solutions to trembling linear programs

In this chapter we will focus on three notions of trembling-hand refinements: extensive-form perfect equilibria (EFPE, already seen in Chapter 12), quasi-perfect equilibrium (QPE, defined in Section 13.2.2), and one-sided quasi-perfect equilibrium (OS-QPE, defined in Section 13.2.3). As we will show, in two-player zero-sum games all these equilibria are similar, in the sense that they can be expressed as limit points, as $\epsilon \rightarrow 0^+$, of sequences of optimal solutions to appropriate linear programs (LPs) whose entries are polynomials of $\epsilon > 0$. In this section, we formalize this type of problems into a general framework and develop a series of results that apply not only to QPEs, OS-QPEs, and EFPEs, but to any other computational problem with that general structure as well.

Definition 13.1 (Trembling linear program). A *trembling linear program* (TLP) is a linear program parametric in $\epsilon > 0$,

$$P(\epsilon) : \begin{cases} \max & c(\epsilon)^\top x \\ \text{s.t.} & \mathbf{A}(\epsilon) x = \mathbf{b}(\epsilon) \\ & x \geq \mathbf{0}, \end{cases}$$

where $\mathbf{A}(\epsilon)$, $\mathbf{b}(\epsilon)$, $c(\epsilon)$ are polynomials in the variable ϵ with rational coefficients. Furthermore, we require that the set of all feasible solutions for $P(\epsilon)$ be non-empty for all sufficiently small ϵ , and that the set of all feasible solutions for $P(0)$ be non-empty and bounded.

In the case of QPEs (Section 13.2.2) the perturbation variable ϵ only appears in \mathbf{b} , whereas in the case of EFPEs (Section 13.2.1) the perturbation ϵ only appears in \mathbf{A} .^[13.a]

^[13.a]In principle, one could also study the case where ϵ appears only in c . To our knowledge, that does not have applications to games, and has not been studied in that context. However, it has been studied, for example, for *lexicographic multi-objective* LPs Isermann, 1982. In particular, given the set of objectives $c_1^\top x, c_2^\top x, \dots, c_n^\top x$, one can define a new objective $(c_1 + \epsilon c_2 + \dots + \epsilon^{n-1} c_n)^\top x$. That formulation is fundamentally equivalent to the *grossone*-based approach Cococcioni, Pappalardo, and Sergeev, 2018. Even for the *grossone*-based approach, our TLP-based approach overcomes some of the key limitations of the work by Cococcioni, Pappalardo, and Sergeev (2018). In particular, our algorithm for finding a TLP limit solution (Section 13.4) is purely numerical (yet provably correct)—thus avoiding the time

We now formalize the concept of *limit solution* for a TLP:

Definition 13.2 (Limit solution of a TLP). A limit solution of a TLP $P(\epsilon)$ is a limit point of optimal solutions for $P(\epsilon)$ as $\epsilon \rightarrow 0^+$.

In the rest of the section, we show that EFPE, QPE, and OS-QPE strategies can all be expressed as limit solutions of appropriately defined trembling LPs. To do so, we will assume that a two-player zero-sum game has been fixed. Analogously to the previous chapter, for each player $i \in \{1, 2\}$, we let $\mathbf{F}_i \in \mathbb{R}^{\mathcal{J}_i \times \Sigma_i}$ and $\mathbf{f}_i \in \mathbb{R}^{\mathcal{J}_i}$ define the player's sequence-form polytope

$$\mathcal{Q}_i = \{\mathbf{x} \in \mathbb{R}_{\geq 0}^{\Sigma_i} : \mathbf{F}_i \mathbf{x} = \mathbf{f}_i, \mathbf{x} \geq \mathbf{0}\}.$$

Furthermore, we let \mathbf{U}_i be the payoff matrix of Player i .

13.2.1 Extensive-form perfect equilibria as trembling linear programs

As defined in [Definition 12.1](#) in [Chapter 12](#), an EFPE is a limit point, as $\epsilon \rightarrow 0^+$, of Nash equilibria of the restricted game in which players need to select each action with probability at least ϵ . Using the fact that in two-player zero-sum games Nash equilibrium strategies are solutions to a linear program, we obtain the following result.

Proposition 13.1. Any limit point of solutions to the trembling linear program

$$\left\{ \begin{array}{l} \max_{\tilde{\mathbf{x}}_i, \mathbf{v}_{-i}} \quad \mathbf{f}_{-i}^\top \mathbf{v}_{-i} \\ \text{s.t.} \quad \textcircled{1} \quad \mathbf{R}_{-i}^{-\top}(\epsilon) \mathbf{F}_{-i}^\top \mathbf{v}_{-i} \leq \mathbf{R}_{-i}^{-\top}(\epsilon) \mathbf{U}_i^\top \mathbf{R}_i^{-1}(\epsilon) \tilde{\mathbf{x}}_i \\ \quad \quad \quad \textcircled{2} \quad \mathbf{F}_i \mathbf{R}_i^{-1}(\epsilon) \tilde{\mathbf{x}}_i = \mathbf{f}_i \\ \quad \quad \quad \textcircled{3} \quad \tilde{\mathbf{x}}_i \geq \mathbf{0}. \end{array} \right.$$

as the trembling magnitude $\epsilon \rightarrow 0^+$ is an extensive-form perfect equilibrium strategy for Player $i \in \{1, 2\}$.

Proof. A Nash equilibrium strategy for Player $i \in \{1, 2\}$ in a two-player zero-sum game is a max-min strategy. Incorporating the constraint that the strategy \mathbf{x}_i pick each action with probability at least ϵ , captured via the behavioral perturbation matrix $\mathbf{R}_i(\epsilon)$ as $\mathbf{R}_i(\epsilon) \mathbf{x}_i \geq \mathbf{0}$, we obtain the optimization problem

and space penalty for storing the symbolic powers of the *grossone* unit—and not tied to the use of the simplex algorithm, thereby guaranteeing a polynomial run time.

$$\left\{ \begin{array}{l} \max_{\mathbf{x}_i} \left\{ \begin{array}{l} \min_{\mathbf{x}_{-i}} \mathbf{x}_i^\top \mathbf{U}_i \mathbf{x}_{-i} \\ \text{s.t.} \quad \textcircled{1} \quad \mathbf{F}_{-i} \mathbf{x}_{-i} = \mathbf{f}_{-i} \\ \quad \quad \textcircled{2} \quad \mathbf{R}_{-i}(\epsilon) \mathbf{x}_{-i} \geq \mathbf{0} \end{array} \right. \\ \text{s.t.} \quad \textcircled{3} \quad \mathbf{F}_i \mathbf{x}_i = \mathbf{f}_i \\ \quad \quad \textcircled{4} \quad \mathbf{R}_i(\epsilon) \mathbf{x}_i \geq \mathbf{0}. \end{array} \right. \quad (13.1)$$

Since the perturbation matrices \mathbf{R}_i and \mathbf{R}_{-i} are invertible (Lemma 12.1), we can substitute $\tilde{\mathbf{x}}_i = \mathbf{R}_i(\epsilon)\mathbf{x}_i$, $\tilde{\mathbf{x}}_{-i} = \mathbf{R}_{-i}(\epsilon)\mathbf{x}_{-i}$, obtaining

$$\left\{ \begin{array}{l} \max_{\tilde{\mathbf{x}}_i} \left\{ \begin{array}{l} \min_{\tilde{\mathbf{x}}_{-i}} \tilde{\mathbf{x}}_i^\top \mathbf{R}_i^{-\top}(\epsilon) \mathbf{U}_i \mathbf{R}_{-i}^{-1}(\epsilon) \tilde{\mathbf{x}}_{-i} \\ \text{s.t.} \quad \textcircled{1} \quad \mathbf{F}_{-i} \mathbf{R}_{-i}^{-1}(\epsilon) \tilde{\mathbf{x}}_{-i} = \mathbf{f}_{-i} \\ \quad \quad \textcircled{2} \quad \tilde{\mathbf{x}}_{-i} \geq \mathbf{0} \end{array} \right. \\ \text{s.t.} \quad \textcircled{3} \quad \mathbf{F}_i \mathbf{R}_i^{-1}(\epsilon) \tilde{\mathbf{x}}_i = \mathbf{f}_i \\ \quad \quad \textcircled{4} \quad \tilde{\mathbf{x}}_i \geq \mathbf{0}. \end{array} \right.$$

Taking the dual of the inner problem introducing the vector of dual variables \mathbf{v}_{-i} for constraint $\textcircled{1}$, we obtain the LP in the statement. \square

13.2.2 Quasi-perfect equilibria: Definition and formulation

Quasi-perfection, introduced by van Damme (1984), is significantly more intricate to define than extensive-form perfection. Instead of giving an explicit lower bound on the probability with which each action needs to be selected, the definition of a quasi-perfect equilibrium (QPE) relies on a refined notion of best response. We now give one of the multiple known equivalent definitions, and we present it for the special case of two-player games only. Several equivalent definitions that apply to more general games can be found in the original work by van Damme, as well as in the work by Miltersen and Sørensen (2010) and Gatti, Gilli, and Marchesi (2020).

Definition 13.3 (*I*-local purification). Let $i \in \{1, 2\}$ be a player, \mathbf{x} be a strategy for Player i , and let $I \in \mathcal{I}_i$ be an information set. We say that a strategy \mathbf{x}' for Player i is an *I*-local purification of \mathbf{x} if \mathbf{x}' is deterministic at any information set $I' \succcurlyeq I$, and coincides with \mathbf{x} at any other information set. When \mathbf{x}' is an *I*-local purification of \mathbf{x} , we further say that

- \mathbf{x}' is ϵ -consistent with \mathbf{x} if, for all $I' \succcurlyeq I$, \mathbf{x}' assigns probability 1 only to actions that have probability $\geq \epsilon$ in \mathbf{x} ;
- $\boldsymbol{\pi}'$ is optimal against a given strategy of the opponent if no other *I*-local purification of \mathbf{x} achieves (strictly) higher expected utility against said strategy of the opponent.

Definition 13.4 (ϵ -quasi-perfect best response). A strategy x_i is an ϵ -quasi-perfect best response to the opponent strategy x_{-i} if

- (i) x assigns strictly positive probability to all actions of Player i ; and
- (ii) for all information sets $I \in \mathcal{I}_i$ of Player i , every ϵ -consistent I -local purifications of x_i (Definition 13.3) is optimal for x_{-i} .

A strategy profile (x_1, x_2) where each strategy is an ϵ -quasi-perfect best response to the opponent's strategy is called an ϵ -quasi-perfect strategy profile.

Definition 13.5 (Quasi-perfect equilibrium). A quasi-perfect equilibrium is any limit point of ϵ -quasi-perfect strategy profiles as $\epsilon \rightarrow 0^+$.

It is known since the work by Miltersen and Sørensen (2010) that some QPEs (we call them *Miltersen-Sørensen QPEs*) can be computed in any two-player game as the limit point of Nash equilibria of perturbed games $\Gamma(\epsilon)$, akin to EFPE. The subtlety is that while in EFPE each perturbed game $\Gamma(\epsilon)$ mandates a lower bound of ϵ on the probability of playing each action, in the case of a Miltersen-Sørensen QPE the lower bounds are given on the probability of each *sequence* of actions. Specifically, for any $\epsilon \geq 0$ and for each player $i \in \{1, 2\}$, let $\ell_i : \epsilon \rightarrow \mathbb{R}_{>0}^{\Sigma_i}$ denote the vector whose entries are defined as

$$\ell_i(\epsilon)[\sigma] = \epsilon^{\text{depth}(\sigma)} \quad \forall \sigma \in \Sigma_i, \tag{13.2}$$

where $\text{depth}(\sigma)$ denotes the number of Player i 's actions on the path from the root of the player's tree-form decision process down to the sequence σ (the empty sequence has a depth of 0). Miltersen and Sørensen (2010) prove that any limit point of the solution to the perturbed optimization problem

$$\begin{aligned} \max_{\substack{\mathbf{F}_1 \mathbf{x}_1 = \mathbf{f}_1 \\ \mathbf{x}_1 \geq \ell_1(\epsilon)}} \min_{\substack{\mathbf{F}_2 \mathbf{x}_2 = \mathbf{f}_2 \\ \mathbf{x}_2 \geq \ell_2(\epsilon)}} \mathbf{x}_1^\top \mathbf{U}_1 \mathbf{x}_2. \end{aligned} \tag{13.3}$$

is a Miltersen-Sørensen QPE.^[13.b]

^[13.b]Recently, Gatti, Gilli, and Marchesi (2020) took this construction further, and showed that *any* QPE can be expressed as a limit point of solutions to (13.3), as long as more general vectors of polynomials ℓ_1, ℓ_2 are used than in (13.2). In this chapter, we will solely focus on Miltersen-Sørensen-style perturbation as defined in (13.2).

Proposition 13.2. Any limit point of solutions to the trembling linear program

$$\left\{ \begin{array}{l} \max_{\tilde{\mathbf{x}}_i, \mathbf{v}_{-i}, \mathbf{z}_{-i}} \quad \mathbf{f}_{-i}^\top \mathbf{v}_{-i} + \mathbf{l}_{-i}^\top(\epsilon) \mathbf{z}_{-i} \\ \text{s.t.} \quad \textcircled{1} \quad \mathbf{F}_{-i}^\top \mathbf{v}_{-i} - \mathbf{U}_i^\top \tilde{\mathbf{x}}_i \leq \mathbf{U}_i^\top \mathbf{l}_i(\epsilon) - \mathbf{z}_{-i} \\ \quad \quad \textcircled{2} \quad \mathbf{F}_i \tilde{\mathbf{x}}_i = \mathbf{f}_i - \mathbf{F}_i \mathbf{l}_i(\epsilon) \\ \quad \quad \textcircled{3} \quad \tilde{\mathbf{x}}_i, \mathbf{z}_{-i} \geq \mathbf{0} \end{array} \right.$$

as the trembling magnitude $\epsilon \rightarrow 0^+$ is a Miltersen-Sørensen QPE strategy for Player $i \in \{1, 2\}$.

Proof. Similarly to the proof of [Proposition 13.1](#), we start from the problem of finding a maxmin strategy for Player i in the game subject to the perturbation constraints,

$$\left\{ \begin{array}{l} \max_{\mathbf{x}_i} \quad \left\{ \begin{array}{l} \min_{\mathbf{x}_{-i}} \quad \mathbf{x}_i^\top \mathbf{U}_i \mathbf{x}_{-i} \\ \text{s.t.} \quad \textcircled{1} \quad \mathbf{F}_{-i} \mathbf{x}_{-i} = \mathbf{f}_{-i} \\ \quad \quad \textcircled{2} \quad \mathbf{x}_{-i} \geq \mathbf{l}_{-i}(\epsilon) \end{array} \right. \\ \text{s.t.} \quad \textcircled{3} \quad \mathbf{F}_i \mathbf{x}_i = \mathbf{f}_i \\ \quad \quad \textcircled{4} \quad \mathbf{x}_i \geq \mathbf{l}_i(\epsilon). \end{array} \right. \quad (13.4)$$

Taking the dual of the inner minimization problem, we obtain

$$\left\{ \begin{array}{l} \max_{\mathbf{x}_i} \quad \left\{ \begin{array}{l} \max_{\mathbf{v}_{-i}, \mathbf{z}_{-i}} \quad \mathbf{f}_{-i}^\top \mathbf{v}_{-i} + \mathbf{l}_{-i}(\epsilon)^\top \mathbf{z}_{-i} \\ \text{s.t.} \quad \textcircled{1} \quad \mathbf{F}_{-i}^\top \mathbf{v}_{-i} + \mathbf{z}_{-i} \leq \mathbf{U}_i^\top \mathbf{x}_i \\ \quad \quad \textcircled{2} \quad \mathbf{z}_{-i} \geq \mathbf{0} \end{array} \right. \\ \text{s.t.} \quad \textcircled{3} \quad \mathbf{F}_i \mathbf{x}_i = \mathbf{f}_i \\ \quad \quad \textcircled{4} \quad \mathbf{x}_i \geq \mathbf{l}_i(\epsilon). \end{array} \right.$$

Finally, by substituting $\tilde{\mathbf{x}}_i = \mathbf{x}_i - \mathbf{l}_i(\epsilon)$, $\tilde{\mathbf{x}}_{-i} = \mathbf{x}_{-i} - \mathbf{l}_{-i}(\epsilon)$ we obtain the formulation in the statement. \square

13.2.3 One-sided quasi-perfect equilibrium: Definition and formulation

Both EFPE and QPE define “two sided” notions of refinements, in the sense that both players are trembling. That two-sidedness comes at a computational cost: both the domain of the maximization and minimization problem in the saddle-point formulations (for example, [Equation \(13.3\)](#) in the case of QPE) of the refinements are perturbed, making the computation of a limit point computationally expensive. Yet, in many strategic interactions of interest, a player might be concerned about being able to capitalize on the opponent’s mistakes, but not about

making mistakes of her own. After all, in the age of machines, that player might well be a bot interacting strategically (for example, playing a poker tournament) against imperfect opponents. In that situation, the player in question might therefore seek, in the interest of lowering the computational requirement of computing a robust strategy, to find equilibrium points that are robust to perturbations *of the opponent's strategy only*, thereby breaking the two-sidedness of all known trembling-hand equilibrium refinements.

We introduce a *one-sided* trembling-hand refinement, which we coin *one-sided quasi-perfect equilibrium* (OS-QPE, Farina and Sandholm, 2021a). Because of the asymmetric role of the players, from now on we stop referring to the players as Player 1 and 2, and adopt the terms *machine player* and *imperfect player* to highlight their asymmetric role. The machine player is assumed to never make mistakes: lower bounds on the probability of play (the “trembling hands”) are only introduced for the imperfect player. Accordingly, from now on we will drop subscripts 1 and 2 to denote quantities that belong to the players, and will use m and h for quantities belonging to the machine and the imperfect player, respectively.

Definition 13.6 (One-sided quasi-perfect equilibrium). We call a strategy profile (π_m, π_h) a *one-sided ϵ -quasi-perfect strategy profile* if π_h is an ϵ -quasi-perfect best response (Definition 13.4) to π_m , and π_m is a best response to π_h . We say that (π_m, π_h) is an *one-sided quasi-perfect equilibrium* if it is the limit point of one-sided ϵ -quasi-perfect strategy profiles, as $\epsilon \rightarrow 0^+$.

In two-player zero-sum games, a OS-QPE is any limit point as $\epsilon \rightarrow 0^+$ of solutions of the bilinear saddle-point problem

$$\max_{\substack{\mathbf{F}_m \mathbf{x}_m = \mathbf{f}_m \\ \mathbf{x}_m \geq \mathbf{0}}} \min_{\substack{\mathbf{F}_h \mathbf{x}_h = \mathbf{f}_h \\ \mathbf{x}_h \geq \boldsymbol{\ell}_h(\epsilon)}} \mathbf{x}_m^\top \mathbf{U}_m \mathbf{x}_h \tag{13.5}$$

where $\boldsymbol{\ell}_h(\epsilon)$ is as in Equation (13.2). Then, for any $\epsilon > 0$ for which the domain of the minimization problem is nonempty, any solution to (13.5) is a one-sided ϵ -quasi-perfect strategy profile.

Following steps similar to those we took for QPE in Proposition 13.2, we obtain the following trembling LP formulation.

Proposition 13.3. Any limit point of solutions to the trembling linear program

$$P(\epsilon) := \begin{cases} \arg \max_{\mathbf{x}_m} (\mathbf{U}_m \boldsymbol{\ell}_h(\epsilon))^\top \mathbf{x}_m + (\mathbf{f}_h - \mathbf{F}_h \boldsymbol{\ell}_h(\epsilon))^\top \mathbf{v} \\ \text{s.t. } \textcircled{1} \mathbf{U}_m^\top \mathbf{x}_m - \mathbf{F}_h \mathbf{v} \geq \mathbf{0} \\ \textcircled{2} \mathbf{F}_m \mathbf{x}_m = \mathbf{f}_m \\ \textcircled{3} \mathbf{x}_m \geq \mathbf{0}, \mathbf{v} \text{ free.} \end{cases}$$

as the trembling magnitude $\epsilon \rightarrow 0^+$ is a one-sided quasi-perfect equilibrium strategy for the machine player.

13.2.4 Formulations with sparsified payoff matrices

As we discuss in [Section 13.4](#), our algorithm for solving trembling linear programs relies on solving the linear program $P(\epsilon)$ for different numerical instantiations of the value of $\epsilon > 0$. Since the solution of the linear programs is the bottleneck of our algorithm, generally speaking the sparser the formulation of the linear programs $P(\epsilon)$, the better. The use of *sparsified* payoff matrices was recently shown to help speed up the solution of linear programs representing Nash equilibrium computations (B. H. Zhang and Sandholm, 2020). Farina and Sandholm (2022) show games with a strong combinatorial structure such as poker can be sparsified efficiently, and propose a more general framework than that of B. H. Zhang and Sandholm (2020). In particular, a *sparsification* of Player i 's payoff matrix \mathbf{U}_i is a decomposition of the form

$$\mathbf{U}_i = \hat{\mathbf{U}}_i + \mathbf{S}_i \mathbf{K}^{-1} \mathbf{V}_i^\top, \quad (13.6)$$

such that the combined number of nonzeros in $\hat{\mathbf{U}}_i$, $\mathbf{K}\mathbf{S}_i$, and \mathbf{V}_i is significantly smaller than the number of nonzeros in \mathbf{U}_i , and \mathbf{K} is a square invertible matrix.

All formulations of refinements ([Propositions 13.1 to 13.3](#)) can benefit from a sparsified payoff matrix directly. As an illustration, we consider the case of OS-QPE, but the technique applies to all other equilibria. By plugging in (13.6) into the constraints of the formulation of [Proposition 13.3](#), we can express OS-QPE as limit solutions of the trembling LP

$$P(\epsilon) := \begin{cases} \arg \max_{\mathbf{x}_m} (\mathbf{U}_m \boldsymbol{\ell}_h(\epsilon))^\top \mathbf{x}_m + (\mathbf{f}_h - \mathbf{F}_h \boldsymbol{\ell}_h(\epsilon))^\top \mathbf{v} \\ \text{s.t. } \textcircled{1} (\hat{\mathbf{U}}_m + \mathbf{S}_m \mathbf{K}^{-1} \mathbf{V}_m^\top)^\top \mathbf{x}_m - \mathbf{F}_h \mathbf{v} \geq \mathbf{0} \\ \textcircled{2} \mathbf{F}_m \mathbf{x}_m = \mathbf{f}_m \\ \textcircled{3} \mathbf{x}_m \geq \mathbf{0}, \mathbf{v} \text{ free.} \end{cases}$$

Hence, by introducing the variable

$$\mathbf{y}_m := \mathbf{K}^{-\top} \mathbf{S}_m^\top \mathbf{x}_m \iff \mathbf{K}^\top \mathbf{y}_m = \mathbf{S}_m^\top \mathbf{x}_m,$$

we can rewrite the formulation into sparsified form as

$$P(\epsilon) := \begin{cases} \arg \max_{\mathbf{x}_m} (\mathbf{U}_m \boldsymbol{\ell}_h(\epsilon))^\top \mathbf{x}_m + (\mathbf{f}_h - \mathbf{F}_h \boldsymbol{\ell}_h(\epsilon))^\top \mathbf{v} \\ \text{s.t. } \begin{array}{l} \textcircled{*} \mathbf{S}_m^\top \mathbf{x}_m - \mathbf{K}^\top \mathbf{y}_m = \mathbf{0} \\ \textcircled{1} \hat{\mathbf{U}}_m^\top \mathbf{x}_m - \mathbf{F}_h \mathbf{v} + \mathbf{V}_m \mathbf{y}_m \geq \mathbf{0} \\ \textcircled{2} \mathbf{F}_m \mathbf{x}_m = \mathbf{f}_m \\ \textcircled{3} \mathbf{x}_m \geq \mathbf{0}, \mathbf{v} \text{ free,} \end{array} \end{cases}$$

where the dense \mathbf{U}_m matrix in constraint $\textcircled{1}$ has been replaced with a much smaller combined number of nonzeros due to the presence of \mathbf{K} , \mathbf{V}_m , and \mathbf{S}_m in the sparsified formulation. Due to the reduced number of nonzero entries in the LP, solving the LP typically requires less computational resources (time and memory).

We will use the sparsified formulation of EFPE, QPE, and OS-QPE when testing on real poker endgames in [Section 13.5](#).

13.3 Basis stability

We now introduce the concept of *basis stability* for TLPs. As we will show, there is a tight connection between a stable basis and a TLP limit solution. In particular, given a stable basis, one can find a TLP limit solution in polynomial time.

First, recall that a *basis* of an LP is a subset of the program's variables such that the submatrix \mathbf{B} obtained considering the columns of \mathbf{A} corresponding to variables in the basis is square and invertible. Now we define our notion of basis stability.

Definition 13.7 (Stable basis). Let $P(\epsilon)$ be a TLP. The LP basis \mathcal{B} is called *stable* if there exists $\bar{\epsilon} > 0$ such that \mathcal{B} is optimal for $P(\epsilon)$ for all $\epsilon : 0 < \epsilon \leq \bar{\epsilon}$.

[Theorem 13.1](#) states a connection between stable bases and TLP limit solutions. Informally, [Theorem 13.1](#) guarantees that once we know a stable basis \mathcal{B} , we can recover a limit solution of the TLP by simply taking the limit of the solutions to the underlying perturbed LP induced by \mathcal{B} .

Theorem 13.1. Let $P(\epsilon)$ be a TLP, and let \mathcal{B} be a stable basis for P , optimal for all $\epsilon : 0 < \epsilon \leq \bar{\epsilon}$. Furthermore, let $\mathbf{x}(\epsilon) := \mathbf{B}(\epsilon)^{-1}(\epsilon) \mathbf{b}(\epsilon)$ be the optimal basic solution of $P(\epsilon)$ corresponding to \mathcal{B} . Then,

1. $\tilde{\mathbf{x}} = \lim_{\epsilon \rightarrow 0^+} \mathbf{x}(\epsilon)$ exists, and
2. $\tilde{\mathbf{x}}$ is a limit solution to the TLP P .

Proof. The fact that \tilde{x} is a solution to P follows directly from [Definition 13.2](#). Therefore, it is enough to show the existence of \tilde{x} .

For all $\epsilon \geq 0$, let $\mathbf{B}(\epsilon)$ be the basis matrix in $P(\epsilon)$, corresponding to the given basis \mathcal{B} . By hypothesis, \mathcal{B} is optimal for $P(\epsilon)$ for all $\epsilon : 0 < \epsilon \leq \bar{\epsilon}$; hence, $\mathbf{B}(\epsilon)$ is invertible for all $\epsilon : 0 < \epsilon \leq \bar{\epsilon}$ and we conclude that $\det \mathbf{B}(\epsilon)$ is not identically zero over that range. This implies that

$$\mathbf{x}(\epsilon) = \mathbf{B}^{-1}(\epsilon) \mathbf{b}(\epsilon)$$

is well defined for all $0 < \epsilon \leq \bar{\epsilon}$ and is a vector of rational functions. This, together with the boundedness assumption of the feasible set of $P(0)$, is enough to conclude that $\lim_{\epsilon \rightarrow 0^+} \mathbf{x}(\epsilon)$ exists. \square

The matrix of (symbolic) rational functions $\mathbf{B}^{-1}(\epsilon)$ in [Theorem 13.1](#) can be computed in polynomial time in the size of the TLP starting from $\mathbf{B}(\epsilon)$. Hence, we can compute all the rational function entries of $\mathbf{x}(\epsilon)$ in polynomial time, and therefore also $\lim_{\epsilon \rightarrow 0^+} \mathbf{x}$. Thus we have the following result.

Theorem 13.2. Let P be a TLP, and let \mathcal{B} be a stable basis for P . A limit solution to P can be computed in polynomial time in the size of the input TLP.

13.3.1 Analytic basis stability condition and existence of stable bases

[Theorem 13.2](#) above shows that the problem of solving a TLP is not harder than the problem of finding a stable basis for it. In this subsection, we focus on this latter problem, showing that a stable basis always exists and can be computed in polynomial time given access to an efficient LP oracle.

Given a TLP and a stable basis \mathcal{B} for it, let $\mathbf{B}(\epsilon)$ be the basis matrix corresponding to \mathcal{B} in the underlying perturbed LP $P(\epsilon)$. From the theory of LPs, we know that \mathcal{B} is optimal for $P(\epsilon)$ if and only if (see, for instance, the book by Dantzig and Thapa ([2006](#))):

- it is primal-feasible, that is,

$$\mathbf{B}^{-1}(\epsilon) \mathbf{b}(\epsilon) \geq \mathbf{0}.$$

(In practice, sometimes we have non-negativity constraint on only some of the variables, but not all. In this case, we can simply check that $\mathbf{B}^{-1}(\epsilon) \mathbf{b}(\epsilon)$ is non-negative only for what concerns the relevant entries.)

- the reduced costs of all nonbasic columns are nonpositive, that is,

$$\mathbf{c}_{\bar{\mathcal{B}}}^\top - \mathbf{c}_{\mathcal{B}}^\top \mathbf{B}^{-1}(\epsilon) \bar{\mathbf{B}}(\epsilon) \leq \mathbf{0},$$

where $\mathbf{c}_{\mathcal{B}}$ is the part of \mathbf{c} corresponding to the basic variables, $\mathbf{c}_{\bar{\mathcal{B}}}$ is the part of \mathbf{c} corresponding to the nonbasic variables, and $\bar{\mathbf{B}}(\epsilon)$ is the matrix formed by all nonbasic columns.

Therefore, we can define the following analytical notion of a *stability certificate*. It collects the conditions above into a vector $\mathbf{t}_{\mathcal{B}}(\epsilon)$, which is nonnegative if and only if \mathcal{B} is an optimal basis for the LP $P(\epsilon)$. Therefore, by the definition of basis stability, a basis is stable if $\mathbf{t}_{\mathcal{B}}(\epsilon)$ is nonnegative for all sufficiently small values of ϵ . Formally, we have the following.

Theorem 13.3. Given a TLP $P(\epsilon)$, a basis \mathcal{B} is stable if and only if there exists $\bar{\epsilon} > 0$ such that

$$\mathbf{t}_{\mathcal{B}}(\epsilon) := \begin{pmatrix} \mathbf{B}^{-1}(\epsilon) \mathbf{b}(\epsilon) \\ \bar{\mathbf{B}}^\top(\epsilon) \mathbf{B}^{-1}(\epsilon) \mathbf{c}_{\mathcal{B}} - \mathbf{c}_{\bar{\mathcal{B}}} \end{pmatrix} \geq \mathbf{0} \quad \forall \epsilon : 0 \leq \epsilon \leq \bar{\epsilon}.$$

The vector $\mathbf{t}_{\mathcal{B}}(\epsilon)$ is called the *stability certificate* for \mathcal{B} .

13.3.2 Existence of stable bases

We use [Theorem 13.3](#) as an important building block to prove the following. The proof is presented in the appendix.

Theorem 13.4. Given as input a TLP $P(\epsilon)$, there exists $\epsilon^* > 0$ such that any optimal basis for the numerical LP $P(\epsilon^*)$ is stable. Furthermore, such a value ϵ^* can be computed in polynomial time in the input size.^a

^aWe assume that a polynomial of degree d requires $\Omega(d)$ space to represent in the input. If this were not the case, evaluating a polynomial in an integer n would not be an efficient operation, since it requires $\Omega(d \log n)$ bits to represent the output.

Before showing the proof of [Theorem 13.4](#) we recall a couple of simple facts from [Chapter 12](#).

Lemma 12.6 (Restated). Let

$$p(\epsilon) := \frac{a_0 + a_1 \epsilon^1 + \dots + a_n \epsilon^n}{b_0 + b_1 \epsilon^1 + \dots + b_m \epsilon^m}$$

be a rational function such that $a_0, b_0 \neq 0$, and let $\mu_a := \max_i |a_i|$, $\mu_b := \max_i |b_i|$. Then $p(\epsilon)$ has the same sign of a_0/b_0 for all $0 \leq \epsilon \leq \epsilon^*$, where

$$\epsilon^* := \min \left\{ \frac{|a_0|}{\mu + |a_0|}, \frac{|b_0|}{\mu + |b_0|} \right\}.$$

Lemma 12.7 (Restated). Let

$$p(\epsilon) := \frac{a_0 + a_1\epsilon^1 + \cdots + a_n\epsilon^n}{b_0 + b_1\epsilon^1 + \cdots + b_m\epsilon^m}$$

be a rational function with integer coefficients, where the denominator is not identically zero; let $\mu_a := \max_i |a_i|$, $\mu_b := \max_i |b_i|$, $\mu := \max\{\mu_a, \mu_b\}$ and $\epsilon^* := 1/(2\mu)$. Then exactly one of the following holds:

- $p(\epsilon^*) = 0$ for all $0 < \epsilon \leq \epsilon^*$,
- $p(\epsilon^*) > 0$ for all $0 < \epsilon \leq \epsilon^*$,
- $p(\epsilon^*) < 0$ for all $0 < \epsilon \leq \epsilon^*$.

We now proceed with the proof of [Theorem 13.4](#).

Proof of Theorem 13.4. Let \mathcal{B}^* be the set of all bases for $P(\epsilon)$ that are optimal for at least one $\epsilon \in \mathbb{R}_{>0}$. For any $\mathcal{B} \in \mathcal{B}^*$, we let $t_{\mathcal{B}}(\delta)$ be the stability certificate for \mathcal{B} ([Theorem 13.3](#)). All entries of $t_{\mathcal{B}}(\delta)$ are rational functions in δ ; hence, by [Lemma 12.6](#), we can find a value $\delta_{\mathcal{B}}^* > 0$, such that all entries of $t_{\mathcal{B}}(\delta)$ keep the same sign on the domain $0 < \delta \leq \delta_{\mathcal{B}}^*$. We now introduce the function $f : \mathcal{B}^* \rightarrow \mathbb{R}_{>0}$ that maps every $\mathcal{B} \in \mathcal{B}^*$ to the corresponding value of $\delta_{\mathcal{B}}^*$. Since \mathcal{B}^* is finite, $\min f$ exists and is (strictly) positive; this means that any optimal basis for $P(\min f)$ is optimal for all $P(\epsilon)$ where $0 < \epsilon \leq \min f$, and is therefore stable.

In light of the above, we only need to prove that we can compute a lower bound for $\min f$ in polynomial time. We will assume without loss of generality that the objective function is not perturbed. Furthermore, we will assume without loss of generality that $\mathbf{A}(\epsilon)$, $\mathbf{b}(\epsilon)$ and $\mathbf{c}(\epsilon)$ only contain integer entries (if not, it is enough to multiply all the entries in the LP by the least common multiple of all denominators to satisfy this assumptions). As long as we can prove that the maximum coefficient appearing in $t_{\mathcal{B}}$ is polynomially large (in the size of the input TLP), the result follows from the bound in [Lemma 12.6](#).

The entries of the stability certificate are obtained by composing sums and products of entries from three vectors: the LP matrix $\mathbf{A}(\epsilon)$, the inverse of the basis matrix $\mathbf{B}^{-1}(\epsilon)$, the vector of constants $\mathbf{b}(\epsilon)$ and the objective function coefficients \mathbf{c} . Let M be the largest coefficient that appears in $\mathbf{A}(\epsilon)$, $\mathbf{b}(\epsilon)$ and \mathbf{c} , and let m be the largest polynomial degree appearing in $\mathbf{A}(\epsilon)$ and $\mathbf{b}(\epsilon)$. We now study the magnitude of the maximum coefficient and the maximum polynomial degree that can appear in $\mathbf{B}(\epsilon)^{-1}$.

Introducing $\mathbf{C}(\epsilon) = \text{cof } \mathbf{B}(\epsilon)$, the cofactor matrix of $\mathbf{B}(\epsilon)$, we can write the well-known identity

$$\mathbf{B}^{-1}(\epsilon) = \frac{\mathbf{C}(\epsilon)^\top}{\det \mathbf{B}(\epsilon)}$$

Denominator. We now give an upper bound on the coefficients of the denominator of the entries in $\mathbf{B}^{-1}(\epsilon)$. By using Hadamard's inequality, we can write

$$\text{coeff}(\det \mathbf{B}(\epsilon)) \leq n^{n/2} M^n \text{coeff}((1 + \epsilon + \dots + \epsilon^{m_A})^n),$$

where $\text{coeff}(\cdot)$ is the largest coefficient of its polynomial argument. Since

$$\text{coeff}((1 + \epsilon + \dots + \epsilon^{m_A})^n) \leq (m_A + 1)^n,$$

we have

$$\text{coeff}(\det \mathbf{B}(\epsilon)) \leq n^{n/2} ((m_A + 1)M)^n, \quad \deg(\det \mathbf{B}(\epsilon)) \leq n \cdot m_A.$$

Notice that this bound is valid for all possible basis matrices $\mathbf{B}(\epsilon)$.

Numerator. It is easy to see that the bounds on $\text{coeff}(\det \mathbf{B}(\epsilon))$ hold for the cofactor matrix as well:

$$\text{coeff}(\det \mathbf{C}(\epsilon)^\top) \leq n^{n/2} ((m_A + 1)M)^n, \quad \deg(\det \mathbf{C}(\epsilon)^\top) \leq n \cdot m_A.$$

Again, it is worthwhile to notice that this bound is valid for all possible basis matrices $\mathbf{B}(\epsilon)$.

Stability certificate. We have

$$\begin{aligned} \text{coeff}(\mathbf{C}^\top \mathbf{b}(\epsilon)) &\leq \text{coeff}(\bar{\mathbf{B}}^\top(\epsilon) \mathbf{C}(\epsilon)^\top \mathbf{c}_B) \leq n^{n/2} ((m_A + 1)M)^n \cdot m_A \cdot M_c M, \\ \text{coeff}(\det \mathbf{B}(\epsilon) \mathbf{c}_{\bar{B}}) &\leq n^{n/2} ((m_A + 1)M)^n \cdot M_c. \end{aligned}$$

Hence,

$$\begin{aligned} \text{coeff}(\mathbf{t}_B(\epsilon)) &\leq n^{n/2} ((m_A + 1)M)^n \cdot M_c (m_A M + 1) \\ &\leq n^{n/2} ((m_A + 1)M)^{n+1} \cdot M_c. \end{aligned}$$

Therefore, all coefficients involved require a polynomial number of bits to be represented, concluding the proof. \square

The existence of polynomial-time algorithms for solving LPs (Karmarkar, 1984), as well as for finding optimal basic solutions (Megiddo, 1991), taken together with [Theorems 13.2](#) and [13.4](#), immediately imply the following corollary.

Corollary 13.1. A limit solution to a TLP can be found in polynomial time by means of the following algorithm.

Algorithm 13.1: Naïve algorithm for finding a limit solution to a TLP $P(\epsilon)$.

- 1 Compute the value ϵ^* as described in the proof of [Theorem 13.4](#);
 - 2 Extract an optimal basis \mathcal{B} for $P(\epsilon^*)$;
 - 3 Extract the (symbolic) basis matrix $\mathbf{B}(\epsilon)$ corresponding to \mathcal{B} ;
 - 4 Compute the symbolic vector $\mathbf{x}(\epsilon) = \mathbf{B}^{-1}(\epsilon) \mathbf{b}(\epsilon)$;
 - 5 **return** $\tilde{\mathbf{x}} = \lim_{\epsilon \rightarrow 0^+} \mathbf{x}(\epsilon)$; [▷ See also [Theorem 13.2](#)]
-

When this algorithm is applied to the EFPE or QPE TLPs, it essentially specializes into the algorithm proposed by Miltersen and Sørensen (2010) for QPE and the one proposed in [Chapter 12](#) for EFPE. In practice, as we show in the experiments later in this dissertation, this algorithm is extremely inefficient, because it involves finding an exact solution to an LP whose numerical constants require a large number of bits. Therefore, we devote the next section to developing a practical algorithm for finding limit solutions to general TLPs.

13.4 A practical algorithm for finding a TLP limit solution

We now develop a practical algorithm for finding a limit solution in a TLP $P(\epsilon)$. It avoids the pessimistically small numerical perturbation ϵ^* of [Theorem 13.4](#) by using an efficient stability-checking oracle for checking if a basis is stable or not. It enables an iterative algorithm that repeatedly picks a numerical perturbation ξ , computes an optimal basis for the perturbed LP $P(\xi)$, and queries the basis-stability oracle. If the basis is not stable, the algorithm concludes that the perturbation value ξ was too optimistic, and a new iteration is performed with a smaller perturbation (for example, $\xi/2$). On the other hand, if the basis is stable, the algorithm takes the limit of the LP solution and returns it as the limit solution of the TLP (by [Theorem 13.1](#), this is guaranteed to provide a limit solution). Termination of the algorithm is guaranteed by the following observation.

Observation 13.1. Let ϵ^* be as in [Theorem 13.4](#). Any value of ξ in the range $(0, \epsilon^*]$ guarantees termination of the algorithm. Indeed, by [Theorem 13.4](#), any optimal basis for $P(\xi)$ is stable

and makes our iterative algorithm terminate. Furthermore, if after every negative stability test the value of ξ is reduced by a constant multiplicative factor (e.g., halved), then since ϵ^* only has a polynomial number of bits, the algorithm terminates after trying at most a polynomial number of different values for ξ .

This algorithm is summarized pictorially in Figure 13.1.

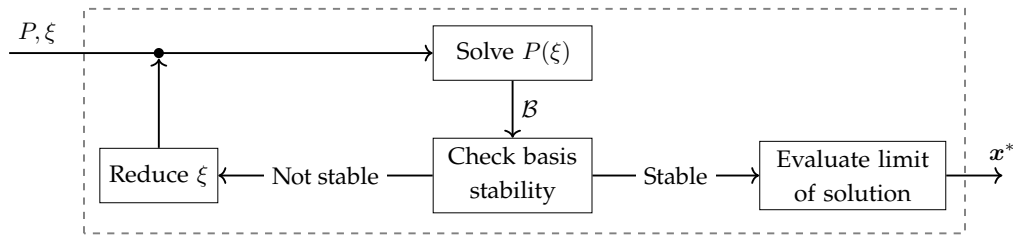


Figure 13.1: High-level overview of the steps of our practical algorithm for finding a TLP limit solution.

Two practical considerations contribute to making our algorithm appealing. First, as long as ξ is sufficiently large, we can use any finite-precision LP solver to solve the numerically-perturbed linear program. However, as soon as ξ becomes small, a finite-precision solver is doomed to fail because of numerical instability, and an implementation of the simplex algorithm supporting arbitrary-precision rational numbers is required. Second, the optimal basis found in the previous iteration can be used to *warm-start* the next iteration. This greatly reduces the runtime of the method, as iterations of the arbitrary-precision simplex method are extremely expensive.

In the next section, we formally state the purpose of the basis-stability oracle.

13.4.1 Basis-stability oracle

We now formally state the purpose of the basis-stability oracle.

Definition 13.8. Given a TLP $P(\epsilon)$ and a basis \mathcal{B} optimal for some numeric instance $P(\bar{\epsilon})$, a *basis-stability oracle* determines whether \mathcal{B} is stable or not.

It is not necessary—and in general not true—that the inverse of $\mathbf{B}(0)$ exist. We start from the simpler case in which $\mathbf{B}^{-1}(0)$ exists (thus ruling out the possibility that the stability certificate $t_{\mathcal{B}}$ is not defined in 0) and later move to the general case.

However, the existence of $\mathbf{B}^{-1}(0)$ allows further numerical optimizations making the overall algorithm fast. In what follows we separately analyze these two cases (singular versus non-singular $\mathbf{B}(0)$).

13.4.2 Oracle for non-singular basis matrices

If $\mathbf{B}(0)$ is non-singular, then $\mathbf{B}^{-1}(\epsilon) \mathbf{b}(\epsilon)$ and $\bar{\mathbf{B}}^\top(\epsilon) \mathbf{B}^{-\top}(\epsilon) \mathbf{c}_B - \mathbf{c}_{\bar{B}}$ are analytic functions of ϵ at $\epsilon = 0$. Thus, $\mathbf{t}_B(\epsilon)$ is analytic at $\epsilon = 0$. In other words, each entry $t_i(\epsilon)$ of $\mathbf{t}_B(\epsilon)$ is equal to its Taylor expansion

$$t_i(\epsilon) = \alpha_{i0} + \frac{\alpha_{i1}}{1!} \epsilon + \frac{\alpha_{i2}}{2!} \epsilon^2 + \frac{\alpha_{i3}}{3!} \epsilon^3 + \dots$$

where $\alpha_{ij} = (d^j t_i(\epsilon)/d\epsilon^j)(0)$ is the j -th derivative of $t_i(\epsilon)$ evaluated at $\epsilon = 0$.^[13.c] The sign of $t_i(\epsilon)$ in positive proximity^[13.d] of 0 is the same as the first (*i.e.*, relative to the lowest degree monomial) non-zero coefficient of the expansion of $t_i(\epsilon)$ around 0. In other words, there exists a $\bar{\epsilon} > 0$ such that $t_i(\epsilon)$ has the same sign as the first non-zero derivative of t_i evaluated in 0 for all $0 < \epsilon < \bar{\epsilon}$. If all derivatives are 0, then we conclude that $t_i(\epsilon)$ is identically zero around $\epsilon = 0$.

This suggests a simple algorithm for determining whether B is stable: we compute its stability certificate $\mathbf{t}_B(\epsilon)$ and repeatedly differentiate each row until we either determine the sign of that row in positive proximity of 0 or we establish that the row is identically zero. If all the rows happen to be non-negative in positive proximity of 0, then the basis is stable; otherwise, it is not. In order to make the algorithm fast, we need to be able to quickly evaluate $\mathbf{t}_B(\epsilon)$ and its derivatives at 0. This fundamentally reduces to our ability to efficiently compute a Taylor expansion of a function of the form $\mathbf{B}^{-1}(\epsilon) \mathbf{H}(\epsilon)$ around $\epsilon = 0$, where \mathbf{H} is a matrix or vector whose entries are polynomial in ϵ . This part of the algorithm assumes that a sparse LU factorization of the numerical basis matrix $\mathbf{B}(0)$ is available; one is easy to compute in polynomial time. Below, we will break the presentation of the algorithm into multiple steps. Since the algorithm described below can be applied to any square matrix $\mathbf{M}(\epsilon)$ —not only to a basis matrix $\mathbf{B}(\epsilon)$ —with polynomial entries and with nonzero determinant at $\epsilon = 0$, we will use the symbol $\mathbf{M}(\epsilon)$ in place of $\mathbf{B}(\epsilon)$.

Derivatives of $\mathbf{M}^{-1}(\epsilon) \mathbf{H}$. We start by showing how to efficiently and inductively evaluate derivatives of $\mathbf{M}^{-1}(\epsilon) \mathbf{H}$ in 0, where \mathbf{H} is a *constant* matrix or vector. We start with a simple lemma.

Lemma 13.1. For all $n \geq 1$,

$$\sum_{i=0}^n \binom{n}{i} \frac{d^i \mathbf{M}(\epsilon)}{d\epsilon^i} \frac{d^{n-i} \mathbf{M}^{-1}(\epsilon)}{d\epsilon^{n-i}} = \mathbf{0}.$$

^[13.c]Throughout this dissertation, we define the zeroth derivative $d^0 f/d\epsilon^0$ of f to be f itself.

^[13.d]We say that a property parametrized by ϵ is true in *positive proximity* of 0 to mean that there exists a $\bar{\epsilon} > 0$ such that the property holds for all $\epsilon : 0 < \epsilon < \bar{\epsilon}$. We say that the property is true in *proximity* of 0 if there exists a $\bar{\epsilon} > 0$ such that the property holds for all $\epsilon : 0 < |\epsilon| < \bar{\epsilon}$.

Proof. The statement is equivalent to the expansion of the identity

$$\frac{d^n}{d\epsilon^n}(\mathbf{M}(\epsilon) \mathbf{M}^{-1}(\epsilon)) = \mathbf{0},$$

which holds true for all $n \geq 1$, by means of the product rule of derivatives. □

Lemma 13.1 implies that

$$\mathbf{M}(0) \frac{d^n \mathbf{M}^{-1}}{d\epsilon^n}(0) = - \sum_{i=1}^n \binom{n}{i} \frac{d^i \mathbf{M}}{d\epsilon^i}(0) \frac{d^{n-i} \mathbf{M}^{-1}}{d\epsilon^{n-i}}(0).$$

Multiplying by \mathbf{H} and introducing the symbol $\mathbf{D}_n := \frac{d^n \mathbf{M}^{-1}}{d\epsilon^n}(0) \mathbf{H}$, we obtain

$$\mathbf{M}(0) \mathbf{D}_n = - \sum_{i=1}^n \binom{n}{i} \frac{d^i \mathbf{M}}{d\epsilon^i}(0) \mathbf{D}_{n-i}.$$

The right hand side is relatively inexpensive to compute, especially when n is small. Indeed, computing $(d^i \mathbf{M}/d\epsilon^i)(0)$ amounts to extracting the coefficients of the monomials of degree i of the polynomial entries in \mathbf{M} . This can be done extremely efficiently by reading directly from the perturbed LP constraint matrix \mathbf{A} . Therefore, if we inductively assume knowledge of $\mathbf{D}_0, \mathbf{D}_1, \dots, \mathbf{D}_{n-1}$, we can easily compute \mathbf{D}_n using the precomputed LU factorization of $\mathbf{M}(0)$.

Derivatives of $\mathbf{M}^{-1}(\epsilon) \mathbf{H}(\epsilon)$. We now turn our attention to the computation of the derivatives of $\mathbf{M}^{-1}(\epsilon) \mathbf{H}(\epsilon)$, where $\mathbf{H}(\epsilon)$ can be any matrix or vector with polynomial entries. This case is particularly relevant, as it applies to both the primal-feasibility conditions and the reduced costs. We introduce the formal symbol $\langle i, j \rangle$ defined over pairs $(i, j) \in \mathbb{N}^2$ as

$$\langle i, j \rangle := \frac{d^i \mathbf{M}^{-1}}{d\epsilon^i}(0) \frac{d^j \mathbf{H}}{d\epsilon^j}(0).$$

By means of the product rule, we have that

$$\frac{d^n (\mathbf{M}^{-1} \mathbf{H})}{d\epsilon^n}(0) = \sum_{i=0}^n \binom{n}{i} \langle i, n-i \rangle. \tag{13.7}$$

From the previous section, we know how to compute $\langle i+1, j \rangle$ having access to $\langle 0, j \rangle, \langle 1, j \rangle, \dots, \langle i, j \rangle$. On the other hand, $\langle 0, j \rangle = \mathbf{M}(0)^{-1} d^j/d\epsilon^j \mathbf{H}(0)$ is easy to compute having access to the LU factorization of $\mathbf{M}(0)$. Therefore, Equation (13.7) gives an efficient way of expanding $\mathbf{M}^{-1}(\epsilon) \mathbf{H}(\epsilon)$ into its power series. Finally, we address the problem of determining, row by row in the derivative vector in the Taylor series, when it is safe to stop after observing only zero-valued derivatives for some row for a number of iterations (*i.e.*, a number of terms in the Taylor series).

Lemma 13.2. Consider a TLP $\epsilon \mapsto P(\epsilon)$ where $P(\epsilon)$ has n rows and let m be the maximum degree appearing in the polynomial functions defining P . Given any basis \mathcal{B} , if the first $2nm + 1$ derivatives of the i -th entry of the stability certificate $t_{\mathcal{B}}(\epsilon)$ are all zero, the entry is identically zero.

Since $2nm + 1$ is a polynomial number in the input size, we conclude that the overall algorithm runs in polynomial time, since it terminates in a polynomial number of steps and each step takes polynomial time. There is a more convenient way of determining whether a given row is 0. It is sufficient to pick a random number $\tilde{\epsilon}$ (for example in $(0, 1)$), and evaluate the rational function t_i at $\tilde{\epsilon}$: if $t_i(\tilde{\epsilon}) = 0$, then t_i is identically zero with probability 1 because of the fundamental theorem of algebra. This is the variant that we use in the experiments later in this dissertation.

Finally, in some cases we can take theoretically sound shortcuts to further enhance the speed of the algorithm. For example, in the formulation of QPE given in [Section 13.2.2](#), the LP constraint matrix \mathbf{A} is constant, meaning that the stability certificate \mathbf{b} has polynomial entries (as opposed to *ratios* of polynomials entries). In this case, we can avoid computing all the derivatives of \mathbf{B}^{-1} , with large practical savings of time and space.

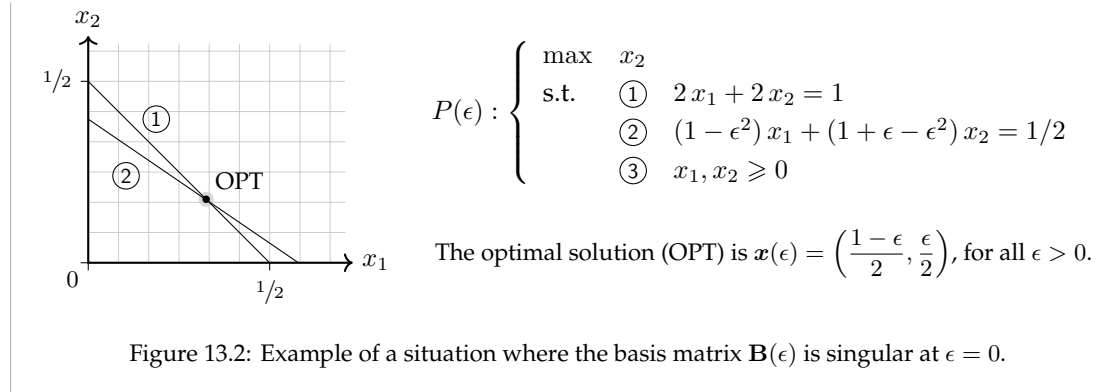
13.4.3 Oracle for singular basis matrices

We now show how to deal with a singular $\mathbf{B}(0)$. An example of this phenomenon is shown in the next example.

Example 13.1. Consider the TLP of [Figure 13.2](#); there, the (only) basis matrix

$$\mathbf{B}(\epsilon) = \begin{pmatrix} 2 & 2 \\ 1 - \epsilon^2 & 1 + \epsilon - \epsilon^2 \end{pmatrix}$$

is not invertible when $\epsilon = 0$.



The core idea of our method is to replace the computation of the Taylor expansion of the stability certificate around $\epsilon = 0$ with a Laurent expansion, that is, a power series in ϵ where negative exponents are allowed. Lemma 13.3 provides the key result that enables this process.

Lemma 13.3. Let $\mathbf{M}(\epsilon)$ be a square matrix with polynomial entries, not all of which are identically zero. Then there exist $k \in \mathbb{N}^+$ and matrices $\tilde{\mathbf{M}}(\epsilon)$ and $\mathbf{T}(\epsilon)$ that have polynomials as entries, with nonsingular $\tilde{\mathbf{M}}(0)$, such that

$$\mathbf{M}(\epsilon) = \epsilon^k \mathbf{T}^{-1}(\epsilon) \tilde{\mathbf{M}}(\epsilon), \tag{13.8}$$

in proximity of $\epsilon = 0$.

Proof. We prove the lemma by induction on the number of roots in 0 of $\det \mathbf{M}(\epsilon)$. This number corresponds to the maximum integer $d \geq 0$ such that ϵ^d is a divisor of $\det \mathbf{M}(\epsilon)$.

Base case. When $d = 0$, $\det \mathbf{M}(0) \neq 0$, and therefore $\mathbf{M}(0)$ is nonsingular. The result holds trivially by letting $\tilde{\mathbf{M}}(\epsilon) = \mathbf{M}(\epsilon)$ and $\mathbf{T}(\epsilon) = \mathbf{I}$ be the identity function for all ϵ .

Inductive step. Suppose the results holds for all matrices $\mathbf{M}(\epsilon)$ whose determinants have $d \leq \bar{d} - 1$ roots in 0, with $\bar{d} \geq 1$. We will now show that the results holds when $d = \bar{d}$ as well. Since $\bar{d} \geq 1$, $\det \mathbf{M}(0) = 0$ and therefore there exists a nonzero vector \mathbf{v} such that $\mathbf{v}^\top \mathbf{M}(0) = \mathbf{0}$. This necessarily means that ϵ divides all entries of $\mathbf{v}^\top \mathbf{M}(\epsilon)$, and therefore $\epsilon^{-1} \mathbf{v}^\top \mathbf{M}(\epsilon)$ is a vector with polynomial entries. Let i be any index such that $v_i \neq 0$, and consider the new matrix $\mathbf{M}'(\epsilon)$ obtained by substituting the i -th row in $\mathbf{M}(\epsilon)$ with $\epsilon^{-1} \mathbf{v}^\top \mathbf{M}(\epsilon)$. It is immediate to verify that we can write this operation compactly as

$$\mathbf{M}'(\epsilon) = \frac{1}{\epsilon} \mathbf{S}(\epsilon) \mathbf{M}(\epsilon).$$

where $\mathbf{S}(\epsilon)$ is a nonzero square matrix with polynomial entries. Hence,

$$\mathbf{M}(\epsilon) = \epsilon \mathbf{S}^{-1}(\epsilon) \mathbf{M}'(\epsilon). \quad (13.9)$$

$\mathbf{M}'(\epsilon)$ is a square matrix with polynomial entries not all of which are identically zero; however, the number of roots in 0 of $\det \mathbf{M}'(\epsilon)$ is smaller than \bar{d} since we multiplied one of the rows by ϵ^{-1} . Thus, we can apply our inductive hypothesis to $\mathbf{M}'(\epsilon)$ and write

$$\mathbf{M}'(\epsilon) = \epsilon^k \mathbf{T}'^{-1}(\epsilon) \tilde{\mathbf{M}}'(\epsilon)$$

for some integer $k \geq 0$. Substituting into (13.9), we obtain

$$\begin{aligned} \mathbf{M}(\epsilon) &= \epsilon^{k+1} \mathbf{S}^{-1}(\epsilon) \mathbf{T}'^{-1}(\epsilon) \tilde{\mathbf{M}}'(\epsilon) \\ &= \epsilon^{k+1} (\mathbf{T}'(\epsilon) \mathbf{S}(\epsilon))^{-1} \tilde{\mathbf{M}}'(\epsilon). \end{aligned}$$

Since $\mathbf{T}'(\epsilon) \mathbf{S}(\epsilon)$ is a square matrix with polynomial entries, concluding the proof. \square

Example 13.2. For the TLP in Figure 13.2,

$$\underbrace{\begin{pmatrix} 2 & 2 \\ 1 - \epsilon^2 & 1 + \epsilon - \epsilon^2 \end{pmatrix}}_{\mathbf{M}(\epsilon)} = \epsilon \cdot \underbrace{\begin{pmatrix} \epsilon & 0 \\ -1/2 & 1 \end{pmatrix}^{-1}}_{\mathbf{T}^{-1}(\epsilon)} \cdot \underbrace{\begin{pmatrix} 2 & 2 \\ -\epsilon & 1 - \epsilon \end{pmatrix}}_{\tilde{\mathbf{M}}(\epsilon)}.$$

We observe that $\mathbf{B}(\epsilon)$ respects the hypotheses of Lemma 13.3: its entries are not all identically zero since its determinant is not identically zero. Inverting Equation (13.8) in proximity of $\epsilon = 0$, we obtain

$$\mathbf{M}^{-1}(\epsilon) = \frac{1}{\epsilon^k} \tilde{\mathbf{M}}^{-1}(\epsilon) \mathbf{T}(\epsilon).$$

Now, given a matrix or vector with polynomial entries $\mathbf{H}(\epsilon)$, suppose that we seek to expand $\mathbf{M}^{-1}(\epsilon) \cdot \mathbf{H}(\epsilon)$ into its Laurent series. Due to Lemma 13.3, we can rewrite this product as

$$\mathbf{M}^{-1}(\epsilon) \cdot \mathbf{H}(\epsilon) = \frac{1}{\epsilon^k} (\tilde{\mathbf{M}}^{-1}(\epsilon) (\mathbf{T}(\epsilon) \mathbf{H}(\epsilon))),$$

where the equality holds in proximity of $\epsilon = 0$. Since $\tilde{\mathbf{M}}(\epsilon)$ is a square matrix with polynomial entries invertible at $\epsilon = 0$ and $\mathbf{T}(\epsilon) \mathbf{H}(\epsilon)$ is a vector or matrix with polynomial entries, we can leverage the machinery of Section 13.4.2 to expand $\tilde{\mathbf{M}}^{-1}(\epsilon) \cdot (\mathbf{T}(\epsilon) \mathbf{H}(\epsilon))$ into its Taylor series around $\epsilon = 0$. Multiplying this power series by ϵ^{-k} gives a Laurent series for $\mathbf{M}^{-1}(\epsilon) \mathbf{H}(\epsilon)$ at $\epsilon = 0$. The above shows how to deal with a singular basis matrix. The rest of the algorithm remains unchanged.

13.4.4 Limit of strategy

Together, [Sections 13.4.2](#) and [13.4.3](#) show that, for every TLP, there exists a polynomial-time basis-stability oracle. Finally, we deal with the last piece of the algorithm, which is the computation of the limit of optimal solutions $\lim_{\epsilon \rightarrow 0^+} \mathbf{x}(\epsilon) = \lim_{\epsilon \rightarrow 0^+} \mathbf{B}^{-1}(\epsilon)\mathbf{b}(\epsilon)$. This task is easy after having computed the Laurent series expansion of $\mathbf{x}(\epsilon)$ around $\epsilon = 0$ (see [Sections 13.4.2](#) and [13.4.3](#)). [Theorem 13.1](#) guarantees that all coefficients of the monomials of negative degree are zero vectors, so we conclude that computing $\lim_{\epsilon \rightarrow 0^+} \mathbf{x}(\epsilon)$ simply amounts to returning the coefficient of the term of degree 0 in the expansion.

13.5 Experimental evaluation

We evaluate the algorithm presented in this chapter by computing EFPEs, QPEs, and one-sided QPEs. The algorithm is single-threaded, was implemented in C++, and was run on a machine with 32GB of RAM and an Intel processor running at a nominal speed of 2.4GHz per core.

As mentioned in [Section 13.4](#), the algorithm computes, as an intermediate step at every iteration, an optimal basis of each trembling linear program where the perturbation magnitude ϵ has been set to a numerical value ϵ^* . We start from the value $\epsilon^* = 10^{-3}$ and use Gurobi to solve the linear program. After the first iteration, if the basis is not stable, we halve the magnitude of ϵ^* and re-solve the linear program using the previous basis as warm start. The process continues until $\epsilon^* < 10^{-6}$. If the basis is still not stable, we switch to a modification of Google’s open-source linear programming solver (GLOP), which we modified so as to use arbitrary-digit precision floating point numbers via GNU’s MPFR library (Fousse, Hanrot, Lefèvre, Pélissier, and Zimmermann, 2007). From there onward, if the basis is not stable we reduce the value of ϵ^* by a factor 1000 and solve again with our modified version of GLOP, until a stable basis is found or $\epsilon^* < 10^{-30}$. In the latter case, our implementation was set up to employ a rational-precision implementation (that is, one that represents all numbers as ratios of integers to achieve an exact “infinite-precision” solution) of the simplex algorithm, but that case was never hit in practice. The basis stability oracle is implemented using rational precision, using GNU’s GMP library (Granlund and the GMP development team, 2012). Therefore, our answer is exact (*i.e.*, infinite-precision) even though the intermediate steps are not.

13.5.1 Experiments on small and medium-sized benchmark games

We experimentally evaluate the performance of our practical algorithm against the following three algorithms in small and medium-sized benchmark games. A description of the games, as usual, is available in [Appendix A](#).

Exact Nash equilibrium solver (“Simplex’) via an implementation of the simplex method using

arbitrary-precision arithmetics implemented using the GMP library. We warm start the LP oracle with a Nash equilibrium found by an LP oracle that uses finite-precision arithmetics.

NPP-based naïve solver ('NPP', Algorithm 13.1) using an infinite-precision LP oracle; to improve the efficiency, we warm start the LP oracle with a NE found using an LP oracle with finite-precision arithmetics.

Symbolic-simplex QPE solver ('M&S') proposed by Miltersen and Sørensen (2010) to find a QPE. It is a modified simplex algorithm, where some entries are kept as polynomials. We implemented the algorithm as described in the original paper. However, we modified the pivoting rule from the suggested one (pick any nonbasic variable with positive reduced cost) to the greedy one (pick any nonbasic variable with maximum reduced cost). This greatly improved run time. This method does not apply to EFPE.

Experimental results for QPE and EFPE are given in Table 13.1.

Game	Nash	QPE				EFPE		
	Simplex Time	M&S Time	NPP Time	This chapter Time Final ϵ		NPP Time	This chapter Time Final ϵ	
K23	1ms	78ms	28ms	11ms	1/819200	14ms	26ms	1/819200
L2232	59ms	> 6h	1.56s	1.15s	10^{-6}	> 6h	535ms	1/819200
L2252	372ms	> 6h	27.81m	494ms	10^{-6}	> 6h	1.6s	10^{-6}
L2282	3.35s	> 6h	> 6h	1s	10^{-6}	> 6h	15s	10^{-6}
L2292	4.90s	> 6h	> 6h	1s	10^{-6}	> 6h	15s	10^{-6}
G23	33ms	21.64m	2.88s	203ms	1/819200	1.93m	489ms	1/819200
G24	1.01m	> 6h	> 6h	19s	1/819200	> 6h	54s	10^{-6}

Table 13.1: Comparison between different methods of computing Nash equilibrium refinements in two-player zero-sum small and medium-sized benchmark games benchmark games.

The largest poker games solvable by the NPP method within 6 hours is L2252 – Leduc poker with 5 ranks, for QPE. The NPP solver is significantly slower than the NE solver. This is because 1) it requires a larger number of pivoting steps, and 2) each pivoting step has a higher cost. Unlike the exact NE computation, additional pivoting steps are needed by the rational simplex to find a QPE or an EFPE, even after warm starting from a Nash equilibrium. These extra pivoting steps need to manipulate extremely small constants due to the values of ϵ , hence introducing a large overhead. For instance, already in L2252 the order of magnitude of the ϵ used for QPE is 10^{-5883} . In the QPE case, these expensive numerical values only appear in the objective function and in the right-hand-side constants. In the EFPE case, they appear in the constraint matrix, and there

are qualitatively more of them, making the computation even slower. Accordingly, the EFPE NPP solver scales very poorly.

The M&S solver only applies to QPEs. Empirically, it is significantly slower than the NPP solver. The reason is two-fold. First, the method is harder to warm start, as the initial basic solution has to be feasible for all sufficiently small $\epsilon > 0$. We initialize the method according to the suggestion of the authors, but this initial vertex is empirically farther away from the optimal one than a NE, which we use to warm start NPP solvers. Second, the pivoting step is more expensive, as the min-ratio test is substituted with a more sophisticated lexicographic test on polynomial coefficients.

Our solver represents a dramatic improvement over the prior state-of-the-art algorithms. It finds a QPE/EFPE in few minutes even on the largest game instances. This is a reduction in runtime by 3–4 orders of magnitude. This breakthrough is mainly due to the fact that, in practice, terminates with an ϵ that is drastically larger than that required by the NPP algorithms.

13.5.2 Experiments on real-world poker endgames

We use our scalable technique to compute sequentially-rational equilibria in real poker endgames that were encountered as part of the “*Brains vs AI*” competition played by the *Libratus AI* (Brown and Sandholm, 2017c). To increase the scalability of our solver, we employed the Kronecker-product-based sparsification technique described in Farina and Sandholm (2022) to bring down the number of nonzeros of the payoff matrix when solving the linear program at each iteration (see Section 13.2.4). The unsparisified and sparsified dimensions of each endgame are listed in Table 13.2. We remark that unlike other papers (for example, Brown and Sandholm, 2019), we do not use a simplified abstraction, and rather use the full-sized abstraction used by Libratus.

Game	Decision nodes		Sequences		Terminal nodes	Sparsified NNZ
	Pl. 1	Pl. 2	Pl. 1	Pl. 2		
REL26	26 550	26 460	64 606	77 617	71 270 327	271 364
REL27	20 801	21 297	61 062	62 518	75 928 256	279 902
REL28	33 930	33 501	85 261	98 786	111 580 420	395 700
REL22	49 470	49 674	146 471	147 075	185 831 684	677 886
REL25	120 744	121 446	360 169	362 263	494 214 830	1 660 170

Table 13.2: Unsparisified and sparsified size of River endgames encountered by Libratus during the “*Brains vs AI*” competition.

Runtimes for each of the solution concepts are given in Table 13.3. We observe that our algorithm is able to compute an exact refinement in all of the game instances. This would have not been possible for any of the other algorithms known for the problem. Our method pushes the boundary of what refinement technology can achieve today by several orders of magnitude,

hopefully adding a crucial step towards the investigation and adoption of sequentially-rational equilibrium refinements, which so far had only remained a theoretical remedy to a serious drawback of the Nash equilibrium, in practice.

Game	Refinement		Time					MPFR precision
			Gurobi	GLOP	Stability	Final ϵ	Total	
REL26	EFPE	Player 1	36s	35m 43s	6m 50s	10^{-15}	43m 11s	200 digits
	EFPE	Player 2	36s	43m 23s	9m 45s	10^{-15}		53m 45s
	QPE	Player 1	1m 8s	7m 50s	25s	10^{-15}	9m 24s	100 digits
	QPE	Player 2	21s	12m 44s	19s	10^{-12}		13m 26s
	OS-QPE	Player 1	20s	16m 15s	20s	10^{-9}	16m 56s	50 digits
	OS-QPE	Player 2	28s	2m 24s	21s	10^{-12}		3m 14s
REL27	EFPE	Player 1	46s	1h 6m	5m 12s	10^{-12}	1h 12m	200 digits
	EFPE	Player 2	3m 37s	1h 10m	9m 12s	10^{-12}		1h 22m
	QPE	Player 1	14s	7m 25s	20s	10^{-9}	8m 0s	100 digits
	QPE	Player 2	24s	4m 22s	19s	10^{-9}		5m 5s
	OS-QPE	Player 1	17s	11m 27s	20s	10^{-9}	12m 5s	50 digits
	OS-QPE	Player 2	33s	2m 47s	21s	10^{-12}		3m 42s
REL28	EFPE	Player 1	2m 52s	1h 3m	58m 7s	10^{-15}	2h 4m	200 digits
	EFPE	Player 2	4m 28s	1h 52m	29m 52s	10^{-15}		2h 26m
	QPE	Player 1	1m 7s	16m 32s	1m 0s	10^{-15}	18m 41s	100 digits
	QPE	Player 2	1m 27s	31m 3s	41s	10^{-15}		33m 12s
	OS-QPE	Player 1	4m 36s	14m 20s	1m 1s	10^{-15}	19m 59s	100 digits
	OS-QPE	Player 2	2m 26s	14m 31s	38s	10^{-12}		17m 37s
REL22	EFPE	Player 1	33m 24s	3h 58m	2h 48m	10^{-15}	7h 20m	200 digits
	EFPE	Player 2	1h 7m	4h 5m	1h 0m	10^{-18}		6h 13m
	QPE	Player 1	8m 7s	1h 43m	1m 45s	10^{-12}	1h 53m	100 digits
	QPE	Player 2	19m 22s	1h 19m	1m 44s	10^{-12}		1h 40m
	OS-QPE	Player 1	10m 11s	37m 21s	2m 19s	10^{-12}	49m 51s	100 digits
	OS-QPE	Player 2	17m 55s	48m 19s	1m 31s	10^{-12}		1h 7m
REL25	EFPE	Player 1	—	—	—	—	> 72h	—
	EFPE	Player 2	—	—	—	—		> 72h
	QPE	Player 1	39m 37s	13h 28m	4m 40s	10^{-12}	14h 13m	200 digits
	QPE	Player 2	1h 16m	8h 7m	3m 42s	10^{-12}		9h 27m
	OS-QPE	Player 1	23m 40s	2h 13m	4m 24s	10^{-9}	2h 42m	200 digits
	OS-QPE	Player 2	53m 7s	3h 5m	3m 54s	10^{-12}		4h 2m

Table 13.3: Computation of refined Nash equilibria in real poker endgames using our algorithm.

Chapter 14

Quantal response and regularization towards human play

All the solution concepts and algorithms we considered so far in this dissertation—Nash equilibrium, coarse correlated equilibrium, correlated equilibrium, team maxmin equilibrium with or without correlation, and trembling-hand refinements such as extensive-form perfect and quasi-perfect equilibria—implicitly assume a model of game-theoretic rationality in which players are perfectly utility-maximizing. For example, when faced with a choice between two actions, one of which produces an expected utility of 1.0, and the other an expected utility of 1.0001, rationality of the player would require that the player pick the latter action 100% of the times, and solution concepts learnt via self-play of learning dynamics would evolve to expect (or predict) such a behavior from each player, despite it is arguable that most human decision makers would instead be rather indifferent between the two actions.

The inability to take into account a model of human play in the computation of game-theoretic behavior is a serious problem for human-computer interactions that involve elements of cooperation rather than purely competition. In such settings, modeling the other human participants accurately is important for success. For example, it is important for a self-driving car at a four-way stop sign to conform to existing human conventions rather than an arbitrary convention derived from self-play (Lerer and Peysakhovich, 2019).

14.1 Contributions and related work

In this chapter, we propose a technique for anchoring no-external-regret dynamics to a given model of human play. At its core, our approach works by regularizing the utility that each player receives in the game by including a KL term that penalizes selecting strategies that are far away from the anchor. In the special case in which the anchor strategy is uniformly random,

our approach recovers logit quantal response equilibrium (McKelvey and Palfrey, 1995), which models players that are not perfectly utility-maximizing but rather pick actions with very similar utilities with roughly equal probability. After developing foundations for how this approach can be used in imperfect-information extensive-form games, and proposing efficient learning algorithms, we show that our approach yields strong human-compatible strategies in practice. In a 200-game no-press Diplomacy tournament involving 62 human participants spanning skill levels from beginner to expert, two AI bots trained with our algorithm both achieved a higher average score than all other participants who played more than two games, and ranked first and third according to an Elo ratings model. We also remark that the same methodology was recently used in building Cicero, a bot for playing the full version of Diplomacy using natural language communication, which was recently featured on *Science* (Bakhtin, Brown, Dinan, Farina, Flaherty, Fried, Goff, Gray, Hu, Jacob, Komeili, Konath, Kwon, Lerer, Lewis, Alexander H. Miller, Mitts, Renduchintala, Roller, Rowe, Shi, Spisak, A. Wei, D. Wu, H. Zhang, and Zijlstra, 2022).

Alternative approaches to human modeling *Behavioral cloning (BC)* is the standard approach for modeling human behaviors given data. Behavioral cloning learns a strategy that maximizes the likelihood of the human data by gradient descent on a cross-entropy loss. However, as observed and discussed in Jacob, David J. Wu, Farina, Lerer, Hu, Bakhtin, Andreas, and Brown, 2022, BC often falls short of accurately modeling or matching human-level performance, with BC models underperforming the human players they are trained to imitate in games such as Chess, Go, and Diplomacy. Intuitively, it might seem that initializing self-play with an imitation-learned strategy would result in an agent that is both strong and human-like. Indeed, Bakhtin, D. Wu, Lerer, and Brown, 2021 showed improved performance against human-like agents when initializing the training procedure from a human imitation strategy and value, rather than starting from scratch. However, such an approach still results in strategies that deviate from human-compatible equilibria, as shown in an appendix of Bakhtin, David J Wu, Lerer, Gray, Jacob, Farina, Alexander H Miller, and Brown (2023).

Prior work on Diplomacy As mentioned, we evaluate our imitation-anchored learning dynamics in the game of Diplomacy. After briefly describing that game specifically, we recall prior approaches that were investigated for the game, and how they relate to ours.

Diplomacy is a benchmark 7-player mixed cooperative/competitive game featuring simultaneous moves and a heavy emphasis on negotiation and coordination. In the *no-press* variant of the game, there is no cheap talk communication. Instead, players only implicitly communicate through moves. In the full version of the game, every pair of players has a private communication channel they can use to coordinate in natural language. In the game, seven players compete for majority control of 34 “supply centers” (SCs) on a map. On each turn, players simultaneously choose actions consisting of an order for each of their units to hold, move, support or convoy

another unit. If no player controls a majority of SCs and all remaining players agree to a draw or a turn limit is reached then the game ends in a draw. In this case, we use a common scoring system in which the score of Player i is $C_i^2 / \sum_{i'} C_{i'}^2$, where C_i is the number of SCs Player i owns.

Most recent successes in no-press Diplomacy use deep learning to imitate human behavior given a corpus of human games. The first Diplomacy agent to leverage deep imitation learning was Paquette, Y. Lu, Bocco, Smith, Satya, Kummerfeld, Pineau, Singh, and Courville, 2019. Subsequent work on no-press Diplomacy have mostly relied on a similar architecture with some modeling improvements (Gray, Lerer, Bakhtin, and Brown, 2020; Anthony, Eccles, Tacchetti, Kramár, Gemp, Hudson, Porcel, Lanctot, Perolat, Everett, Singh, Graepel, and Bachrach, 2020; Bakhtin, D. Wu, Lerer, and Brown, 2021).

Gray, Lerer, Bakhtin, and Brown, 2020 proposed an agent that plays an improved strategy via one-ply search. It uses strategy and value functions trained on human data to conduct search using regret minimization.

Several works explored applying self-play to compute improved strategies. Paquette, Y. Lu, Bocco, Smith, Satya, Kummerfeld, Pineau, Singh, and Courville (2019) applied an actor-critic approach and found that while the agent plays stronger in populations of other self-play agents, it plays worse against a population of human-imitation agents. Anthony, Eccles, Tacchetti, Kramár, Gemp, Hudson, Porcel, Lanctot, Perolat, Everett, Singh, Graepel, and Bachrach (2020) used a self-play approach based on a modification of fictitious play in order to reduce drift from human conventions. The resulting strategy is stronger than pure imitation learning in both 1vs6 and 6vs1 settings but weaker than agents that use search. Most recently, Bakhtin, D. Wu, Lerer, and Brown (2021) combined one-ply search based on equilibrium computation with value iteration to produce an agent called *Double Oracle Reinforcement learning for Action exploration (DORA)*. DORA achieved superhuman performance in a two-player zero-sum version of Diplomacy without human data, but in the full 7-player game plays poorly with agents other than itself.

Jacob, David J. Wu, Farina, Lerer, Hu, Bakhtin, Andreas, and Brown (2022) showed that regularizing inference-time search techniques can produce agents that are not only strong but can also model human behaviour well. In no-press Diplomacy, they show that regularizing hedge (an equilibrium-finding algorithm) with a KL-divergence penalty towards a human imitation learning strategy can match or exceed the human action prediction accuracy of imitation learning while being substantially stronger. KL-regularization toward human behavioral strategies has previously been proposed in various forms in single- and multiagent RL algorithms (Nair, McGrew, Andrychowicz, Zaremba, and Abbeel, 2018; Siegel, Springenberg, Berkenkamp, Abdolmaleki, Neunert, Lampe, Hafner, Heess, and Riedmiller, 2020; Nair, Dalal, Gupta, and Levine, 2021), and was notably employed in AlphaStar (Vinyals, Babuschkin, Czarnecki, Mathieu, Dudzik, Chung, Choi, Powell, Ewalds, Georgiev, et al., 2019), but this has typically been used to improve sample efficiency and aid exploration rather than to better model and coordinate with human play.

An alternative line of research has attempted to build human-compatible agents without

relying on human data (Hu, Lerer, Peysakhovich, and Foerster, 2020; Hu, Lerer, Cui, Pineda, D. Wu, Brown, and Foerster, 2021; Strouse, McKee, Botvinick, Hughes, and Everett, 2021). These techniques have shown some success in simplified settings but have not been shown to be competitive with humans in large-scale collaborative environments.

Related work on KL-regularized games Other work has studied learning dynamics in games regularized with a KL or entropic term, and their relationship to quantal response. Ling, Fang, and Kolter (2018) establishes the connection between quantal-response equilibria and dilated entropic regularization in imperfect-information extensive-form games. The work by Farina, Kroer, and Sandholm (2019a) shows that a decomposition framework similar to that of regret circuits enables to decompose the problem of minimizing regret with respect to the regularized utilities into each decision node, leading to a regularized version of the CFR algorithm. The work by Perolat, Remi Munos, Lespiau, Omidshafiei, Rowland, Ortega, Burch, Anthony, Balduzzi, De Vylder, et al. (2021) studies *continuous-time* learning dynamics with last-iterate convergence. The work by Sokota, D’Orazio, Kolter, Loizou, Lanctot, Mitliagkas, Brown, and Kroer (2023) studies an algorithm that is very similar to a special case of the DiL-piKL algorithm described in this chapter, proving additional properties related to the last-iterate convergence in the non-Bayesian setting.

14.2 Logit quantal responses and KL-anchored responses

In this section we review some standard idea related to logit quantal responses, and introduce the more general concept of KL-anchored responses. To better fix ideas, we begin by considering nonsequential games, that is, games in which each player’s tree-form decision problem only has a single decision node. Later on in the section, we discuss how the ideas extend to general imperfect-information extensive-form games. As we have seen in Chapter 4, a suitable no-regret algorithm leading to logit quantal response equilibria and KL-anchored equilibria in general imperfect-information extensive-form games can be constructed starting from any no-regret algorithm for the probability simplex.

14.2.1 Logit quantal responses

As mentioned in the preamble to the chapter, a standard assumption that underpins most of the material we have seen so far is that players best-respond, that is, when faced with actions they will tend to deterministically pick the one with the highest value, no matter how similar in value the other actions are. So, for example, when faced with the choice between an action with expected utility 1.0 and another action with expected utility 1.0001, it is assumed that rational players would favor the latter 100% of the times.

The concept of a quantal responses revises this assumption by instead postulating that the

probability with which actions tend to be played by agents rather follows a continuous function of the value of the actions. Specifically, when faced with a set of actions \mathcal{A} each of which has expected utility $\mathbf{u}[a]$, a player responding using a *logit* quantal response (the most common type of quantal response) picks actions according to the distribution $\mathbf{x} \in \Delta^{\mathcal{A}}$ defined by

$$\mathbf{x}[a] := \frac{\exp\{\mathbf{u}[a]/\lambda\}}{\sum_{a' \in \mathcal{A}} \exp\{\mathbf{u}[a']/\lambda\}}, \quad (14.1)$$

for some parameter $\lambda \geq 0$. When $\lambda = 0$, \mathbf{x} converges to being a best response, that is, putting mass 0 to any suboptimal action, no matter how small the suboptimality. For higher values of λ , \mathbf{x} puts mass also on suboptimal actions. For example, when $\lambda = 1$, the quantal response to two actions of expected utility 1.0 and 1.0001 puts probability $\approx 49.9975\%$ and $\approx 50.0025\%$ on the two actions. When $\lambda \rightarrow \infty$, the player would pick every action uniformly at random not matter its expected utility.

While best responses can be formulated as solutions to the linear optimization problem

$$\arg \max_{\mathbf{x} \in \Delta^{\mathcal{A}}} \langle \mathbf{u}, \mathbf{x} \rangle,$$

it is well-known (see also [Chapter 5](#)), that logit quantal responses (14.1) arise as solutions to the *entropy-regularized*, strongly convex optimization problem

$$\arg \max_{\mathbf{x} \in \Delta^{\mathcal{A}}} \{ \langle \mathbf{u}, \mathbf{x} \rangle - \lambda \varphi_{\text{ent}}(\mathbf{x}) \}, \quad \text{where} \quad \varphi_{\text{ent}}(\mathbf{x}) := \sum_{a \in \mathcal{A}} \mathbf{x}[a] \log \mathbf{x}[a] \quad (14.2)$$

is the negative entropy function (as is standard, we define $0 \log 0 = 0$).

Given an n -player nonsequential game, a logit quantal response equilibrium (QRE) relative to anchoring coefficients λ_i for each player $i \in \llbracket n \rrbracket$ is an assignment of strategies for each player in a game, such that each player plays the logit quantal response (relative to their own anchoring coefficient) to everyone else's strategies. Similarly to Nash equilibrium, in two-player zero-sum games, in a nonsequential game where the two players have action sets \mathcal{A}_1 and \mathcal{A}_2 and anchoring coefficients λ_1 and λ_2 , a logit QRE can be expressed as the solution to the strongly convex-concave saddle-point problem

$$\max_{\mathbf{x} \in \Delta^{\mathcal{A}_1}} \min_{\mathbf{y} \in \Delta^{\mathcal{A}_2}} \mathbf{x}^{\top} \mathbf{U}_1 \mathbf{y} - \lambda_1 \varphi_{\text{ent}}(\mathbf{x}) + \lambda_2 \varphi_{\text{ent}}(\mathbf{y}), \quad (14.3)$$

where \mathbf{U}_1 is the payoff matrix for Player 1.

14.2.2 Logit quantal response as an instances of KL-anchored response

Given the finite set of actions \mathcal{A} , the uniform distribution $\frac{1}{|\mathcal{A}|} \in \Delta^{\mathcal{A}}$ is such that

$$\text{KL}\left(\mathbf{x} \parallel \frac{\mathbf{1}}{|\mathcal{A}|}\right) = \varphi_{\text{ent}}(\mathbf{x}) \quad \forall \mathbf{x} \in \Delta^{\mathcal{A}}.$$

Hence, we can rewrite (14.2) as

$$\arg \max_{\mathbf{x} \in \Delta^{\mathcal{A}}} \left\{ \langle \mathbf{u}, \mathbf{x} \rangle - \lambda \text{KL}\left(\mathbf{x} \parallel \frac{\mathbf{1}}{|\mathcal{A}|}\right) \right\}. \quad (14.4)$$

Since the KL divergence is a notion of distance, the above optimization problem can be interpreted as interpolating between the objective of maximizing the utility of the response, while at the same time not picking a response that is too far from the uniform distribution. To highlight the effect that λ has on the rationality of the player's choice, we will refer to the parameter λ as the *anchoring coefficient*.

We call a composite problem of the form (14.4) a *KL-anchored response*, as formalized next.

Definition 14.1. Given the set of actions \mathcal{A} available to a player, an *anchor strategy* $\boldsymbol{\tau} \in \Delta^{\mathcal{A}}$, and a utility vector $\mathbf{u} \in \mathbb{R}^{\mathcal{A}}$, the *$\boldsymbol{\tau}$ -anchored response to utility vector \mathbf{u} with anchoring coefficient $\lambda > 0$* is the solution to the optimization problem

$$\arg \max_{\mathbf{x} \in \Delta^{\mathcal{A}}} \{ \langle \mathbf{u}, \mathbf{x} \rangle - \lambda \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}) \}.$$

Like QRE, given an n -player nonsequential game, an *anchored response equilibrium* relative to anchoring coefficients λ_i and anchor strategies $\boldsymbol{\tau}_i$ for each Player $i \in \llbracket n \rrbracket$ is an assignment of strategies for each player in a game, such that each player plays the logit quantal response (relative to their own anchoring coefficient) to everyone else's strategies. Similarly to (14.3), in the two-player zero-sum case we can define a KL-anchored equilibrium as the solution to the strongly convex-concave saddle-point problem

$$\max_{\mathbf{x} \in \Delta^{\mathcal{A}_1}} \min_{\mathbf{y} \in \Delta^{\mathcal{A}_2}} \mathbf{x}^\top \mathbf{U}_1 \mathbf{y} - \lambda_1 \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_1) + \lambda_2 \text{KL}(\mathbf{y} \parallel \boldsymbol{\tau}_2).$$

14.2.3 Learning dynamics for KL-anchored equilibria

In order to use learning dynamics to learn a KL-anchored equilibrium, it is necessary to modify the definition of regret to incorporate the penalty derived from selecting strategies that are far from the anchors. Specifically, consider a generic player i , and let $\boldsymbol{\tau}_i$ be his or her anchor policy. By introducing the regularized utility function

$$\tilde{u}_i(\mathbf{x}, \mathbf{u}) := \langle \mathbf{u}, \mathbf{x} \rangle - \lambda \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_i) \quad \forall \mathbf{x} \in \Delta^{\mathcal{A}_i}, \mathbf{u} \in \mathbb{R}^{\mathcal{A}_i}.$$

we define *KL-anchored external regret* cumulated by strategies $\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, \dots$ relative to utility vectors $\mathbf{u}_i^{(1)}, \mathbf{u}_i^{(2)}, \dots$ as

$$\text{Reg}_i^{(T)}(\hat{\mathbf{x}}) := \sum_{t=1}^T \tilde{u}(\hat{\mathbf{x}}, \mathbf{u}_i^{(t)}) - \tilde{u}(\mathbf{x}_i^{(t)}, \mathbf{u}_i^{(t)}), \quad \text{Reg}_{\text{KL}}^{(T)} := \max_{\hat{\mathbf{x}} \in \Delta^{\mathcal{A}_i}} \text{Reg}_{\text{KL}}^{(T)}(\hat{\mathbf{x}}).$$

A no-KL-anchored-external-regret algorithm will be given in [Section 14.3](#) as a special case of an even more general notion of regret.

14.2.4 Imitation-anchored responses in imperfect-information extensive-form games

the above construction can be extended naturally to imperfect-information extensive-form games. In particular, Ling, Fang, and Kolter (2018) shows that in imperfect-information extensive-form games, a logit quantal response is the solution to the optimization problem over sequence-form strategies defined as

$$\begin{aligned} \arg \max_{\mathbf{x} \in \mathcal{Q}} \left\{ \langle \mathbf{u}, \mathbf{x} \rangle - \lambda \sum_{j \in \mathcal{J}} \sum_{a \in \mathcal{A}_j} \mathbf{x}[ja] \log \frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} \right\} \\ = \arg \max_{\mathbf{x} \in \mathcal{Q}} \left\{ \langle \mathbf{u}, \mathbf{x} \rangle - \lambda \sum_{j \in \mathcal{J}} \mathbf{x}[p_j] \sum_{a \in \mathcal{A}_j} \varphi_{\text{ent}} \left(\left(\frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} \right)_{a \in \mathcal{A}_j} \right) \right\}. \end{aligned}$$

(We remark that the convex regularizer being subtracted is an instantiation of the dilated entropy DGF discussed in [Chapter 5](#).) Correspondingly, the KL-anchored response is naturally generalized as the solution to

$$\arg \max_{\mathbf{x} \in \mathcal{Q}} \left\{ \langle \mathbf{u}, \mathbf{x} \rangle - \lambda \sum_{j \in \mathcal{J}} \mathbf{x}[p_j] \text{KL} \left(\left(\frac{\mathbf{x}[ja]}{\mathbf{x}[p_j]} \right)_{a \in \mathcal{A}_j} \left\| \left(\frac{\boldsymbol{\tau}[ja]}{\boldsymbol{\tau}[p_j]} \right)_{a \in \mathcal{A}_j} \right) \right\}.$$

We remark that the regret circuit formalism ([Chapter 4](#)) can be extended in this case to show that a no-KL-anchored-external-regret algorithm for the sequence-form polytope of a generic imperfect-information extensive-form game can be constructed starting from any no-KL-anchored-external-regret algorithm for probability simplexes (Farina, Kroer, and Sandholm, 2019a).

14.3 Modeling uncertainty on the anchoring coefficients

The previous chapter lays the foundations for a solution concept that trades off utility maximization and playing close to an anchor policy. However, a major obstacle on the path to operationalizing KL-anchored equilibria is that in practice it is not easy to define exactly what anchoring coefficient λ_i the equilibrium notion should be configured with. In this section we propose a possible solution to this problem in nonsequential games, by modeling the uncertainty via a Bayesian game—a special type of imperfect-information extensive-form game with a strong combinatorial structure. In particular, for each player i , we let β_i be a distribution over the set Λ_i of anchoring coefficients (the “types”) that we think might describe the player. In the Bayesian game, at each time t each player i produces strategies $\mathbf{x}_{i,\lambda_i}^{(t)}$, one for each possible type $\lambda_i \in \Lambda_i$. The regularized utility relative to each type λ_i is then defined as

$$\tilde{u}_{i,\lambda_i}(\mathbf{x}, \mathbf{u}) := \langle \mathbf{u}, \mathbf{x} \rangle - \lambda_i \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_i), \quad (14.5)$$

and each player cares about minimizing (that is, keeping sublinear) the per-type regret

$$\max_{\hat{\mathbf{x}} \in \Delta^{A_i}} \left\{ \sum_{t=1}^T \tilde{u}_{i,\lambda_i}(\hat{\mathbf{x}}, \mathbf{u}_i^{(t)}) - \tilde{u}_{i,\lambda_i}(\mathbf{x}_{i,\lambda_i}^{(t)}, \mathbf{u}_i^{(t)}) \right\},$$

no matter the sequence of utility vectors $\mathbf{u}_i^{(t)}$ received as feedback. As usual (Chapter 3), the utility vectors $\mathbf{u}_i^{(t)}$ are the gradient of the player’s expected utility in the underlying game, where the expectation in this case takes into account the additional fact that the type of each player is sampled independently from each distribution β_i .

The algorithm we propose to minimize the notion of regret just defined is called DiL-piKL and can be seen in Algorithm 14.1.

The rest of the section will be focused on the analysis of DiL-piKL, specifically its KL-anchored external regret and its last-iterate convergence properties in two-player zero-sum games. Before delving into the details of the analysis, we summarize a few key takeaways.

DiL-piKL can be understood as a *sampled* form of follow-the-regularized-leader (FTRL). When a player i learns using DiL-piKL, the distributions $\mathbf{x}_{i,\lambda_i}^{(t)}$ for any type $\lambda_i \in \Lambda_i$ are no-regret with respect to the regularized utilities \tilde{u}_{i,λ_i} defined in (14.5). Formally:

Theorem 14.1 (Simplified version of Theorem 14.3). Let W be a bound on the maximum absolute value of any payoff in the game, and $Q_i := \frac{1}{n_i} \sum_{a \in A_i} \log \tau_i(a)$. Then, for any player i , type $\lambda_i \in \Lambda_i$, and number of iterations T , the regret cumulated can be upper bounded as

Algorithm 14.1: DiL-piKL algorithm, for a generic player i

Data: \mathcal{A}_i set of actions for Player i ; Λ_i a set of λ values to consider for Player i ;
 β_i a belief distribution over λ values for Player i .

```

1  $\theta_i^{(0)} \leftarrow \mathbf{0}$ 
2 function NextStrategy( $\_$ )                                [ $\triangleright$  Predictions are not used by DiL-piKL ]
3   Sample  $\lambda \sim \beta_i$ 
4   Let  $\mathbf{x}_{i,\lambda}^{(t)} \in \Delta^{\mathcal{A}_i}$  be the policy such that
      
$$\mathbf{x}_{i,\lambda}^{(t)}[a] \propto \exp\left\{\frac{\theta_i^{(t-1)}[a] + \lambda \log \tau_i[a]}{1/(\eta(t-1)) + \lambda}\right\} \quad \forall a \in \mathcal{A}$$

5   sample an action  $a_i^{(t)} \sim \mathbf{x}_{i,\lambda}^{(t)}$ 
6   return  $\mathbf{1}_{a_i^{(t)}} \in \Delta^{\mathcal{A}_i}$ 

```

```

7 function ObserveUtility( $\mathbf{u}_i^{(t)} \in \mathbb{R}^{\mathcal{A}_i}$ )
8    $\theta_i^{(t)} \leftarrow \frac{t-1}{t}\theta_i^{(t-1)} + \frac{1}{t}\mathbf{u}_i^{(t)}$ 

```

$$\max_{\mathbf{x} \in \Delta^{\mathcal{A}_i}} \left\{ \sum_{t=1}^T \tilde{u}_{i,\lambda_i}(\mathbf{x}, \mathbf{u}_i^{(t)}) - \tilde{u}_{i,\lambda_i}(\mathbf{x}_{i,\lambda_i}^{(t)}, \mathbf{u}_i^{(t)}) \right\} \leq \frac{W^2}{4} \min\left\{ \frac{2 \log T}{\lambda_i}, T\eta \right\} + \frac{\log n_i}{\eta} + \rho_{i,\lambda_i},$$

where the game constant ρ_{i,λ_i} is defined as $\rho_{i,\lambda_i} := \lambda_i(\log n_i + Q_i)$.

We remark that the result holds no matter the choice of learning rate $\eta > 0$, thus implying a $\mathcal{O}_T(\log T/(T\lambda_i))$ regret bound without assumptions on η other than $\eta = \Omega(1)$. Second, in the cases in which λ_i is tiny, by choosing $\eta = \Theta(1/\sqrt{T})$ we recover a sublinear guarantee (of order \sqrt{T}) on the regret.

In two-player zero-sum games, the logarithmic regret of [Theorem 14.1](#) immediately implies that the *average strategy* $\bar{\mathbf{x}}_{i,\lambda_i}^{(T)} := \frac{1}{T} \sum_{t=1}^T \mathbf{x}_{i,\lambda_i}^{(t)}$ of each player i is a $\frac{C \log T}{T}$ -approximate Bayes-Nash equilibrium strategy. In fact, a strong guarantee on the *last-iterate* convergence of the algorithm can be obtained too:

Theorem 14.2 (Simplified version of [Theorem 14.5](#); Last-iterate convergence of DiL-piKL in two-player zero-sum games). When both players in a two-player zero-sum game learn using DiL-piKL for $T \rightarrow \infty$ iterations, their strategies $\mathbf{x}_{i,\lambda_i}^{(T)}$ converge almost surely to the unique Bayes-Nash equilibrium $(\mathbf{x}_{i,\lambda_i}^*)$ of the regularized game defined by utilities \tilde{u}_{i,λ_i} , that is, the solution to the strongly convex-concave saddle-point problem

$$\max_{\mathbf{x}_{1,\lambda_1}} \min_{\mathbf{x}_{2,\lambda_2}} \mathbb{E}_{\lambda_1 \sim \beta_1} \mathbb{E}_{\lambda_2 \sim \beta_2} \left[\mathbf{x}_{1,\lambda_1}^\top \mathbf{U}_1 \mathbf{x}_{2,\lambda_2} - \lambda_1 \text{KL}(\mathbf{x}_{1,\lambda_1} \parallel \boldsymbol{\tau}_1) + \lambda_2 \text{KL}(\mathbf{x}_{2,\lambda_2} \parallel \boldsymbol{\tau}_2) \right],$$

where \mathbf{U}_1 is the utility matrix of Player 1 in the game.

The last-iterate guarantee stated in [Theorem 14.2](#) crucially relies on the strong convexity of the regularized utilities, and conceptually belongs with related efforts in showing last-iterate convergence of online learning methods. However, a key difficulty that sets apart [Theorem 14.2](#) is the fact that the learning agents observe *sampled* actions from the opponents, which makes the proof of the result (as well as the obtained convergence rate) different from prior approaches.

14.3.1 A technical lemma needed in the analysis

In this section we study the last-iterate convergence of DiL-piKL, establishing that in two-player zero-sum games DiL-piKL converges to the (unique) Bayes-Nash equilibrium of the regularized Bayesian game. As a corollary (in the case in which each player has exactly one type), we conclude that piKL converges to the Nash equilibrium of the regularized game in two-player zero-sum games. We start from a technical result.

Lemma 14.1. Fix any player i , $\lambda_i \in \Lambda_i$, and $t \geq 1$. For all $\mathbf{x}, \mathbf{x}' \in \Delta^{\mathcal{A}_i}$, the iterates $\mathbf{x}_{i,\lambda_i}^{(t)}$ and $\mathbf{x}_{i,\lambda_i}^{(t+1)}$ defined in [Line 4](#) of [Algorithm 14.1](#) satisfy

$$\left\langle \frac{\eta}{\eta\lambda_i t + 1} \left(-\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i) \right) + \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}' \right\rangle = 0.$$

Proof. If $t = 1$, then the results follows from direct inspection: $\mathbf{x}_{i,\lambda_i}^{(1)}$ is the uniform strategy (and so $\langle \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(1)}), \mathbf{x} - \mathbf{x}' \rangle = 0$ for any $\mathbf{x}, \mathbf{x}' \in \Delta^{\mathcal{A}_i}$, and so the statement reduces to the first-order optimality conditions for the problem $\mathbf{x}_{i,\lambda_i}^{(2)} = \arg \max_{\mathbf{x} \in \Delta^{\mathcal{A}_i}} \{ -\varphi_{\text{ent}}(\mathbf{x})/\eta + \langle \mathbf{u}_i^{(1)}, \boldsymbol{\pi} \rangle - \lambda_i \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_i) \}$. So, we now focus on the case $t \geq 2$. The iterates $\mathbf{x}_{i,\lambda_i}^{(t+1)}$ and $\mathbf{x}_{i,\lambda_i}^{(t)}$ produced by DiL-piKL are respectively the solutions to the optimization problem

$$\begin{aligned} \mathbf{x}_{i,\lambda_i}^{(t+1)} &= \arg \max_{\mathbf{x} \in \Delta^{\mathcal{A}_i}} \left\{ -\frac{\varphi_{\text{ent}}(\mathbf{x})}{\eta t} + \langle \bar{\mathbf{U}}_i^{(t)}, \mathbf{x} \rangle - \lambda_i \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_i) \right\}, \\ \mathbf{x}_{i,\lambda_i}^{(t)} &= \arg \max_{\mathbf{x} \in \Delta^{\mathcal{A}_i}} \left\{ -\frac{\varphi_{\text{ent}}(\mathbf{x})}{\eta(t-1)} + \langle \bar{\mathbf{U}}_i^{(t-1)}, \mathbf{x} \rangle - \lambda_i \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_i) \right\}, \end{aligned}$$

where we let the averages utility vectors be

$$\bar{\mathbf{U}}_i^{(t-1)} := \frac{1}{t-1} \sum_{t'=1}^{t-1} \mathbf{u}_i^{t'}, \quad \bar{\mathbf{U}}_i^{(t)} := \frac{1}{t} \sum_{t'=1}^t \mathbf{u}_i^{t'}.$$

Since the regularizing function negative entropy φ_{ent} is Legendre, the strategies $\mathbf{x}_{i,\lambda_i}^{(t+1)}$ and $\mathbf{x}_{i,\lambda_i}^{(t)}$ are in the relative interior of the probability simplex, and therefore the first-order optimality conditions for $\mathbf{x}_{i,\lambda_i}^{(t+1)}$ and $\mathbf{x}_{i,\lambda_i}^{(t)}$ are respectively

$$\left\langle -\bar{\mathbf{U}}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i) + \frac{1}{\eta t} \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}), \mathbf{x} - \mathbf{x}' \right\rangle = 0 \quad \forall \mathbf{x}, \mathbf{x}' \in \Delta^{A_i}, \quad (14.6)$$

$$\left\langle -\bar{\mathbf{U}}_i^{(t-1)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i) + \frac{1}{\eta(t-1)} \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}' \right\rangle = 0 \quad \forall \mathbf{x}, \mathbf{x}' \in \Delta^{A_i}.$$

Taking the difference between the equalities, we find

$$\left\langle -\bar{\mathbf{U}}_i^{(t)} + \bar{\mathbf{U}}_i^{(t-1)} + \left(\lambda_i + \frac{1}{\eta t} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \left(\lambda_i + \frac{1}{\eta(t-1)} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}' \right\rangle = 0$$

We now use the fact that

$$\bar{\mathbf{U}}_i^{(t)} - \bar{\mathbf{U}}_i^{(t-1)} = -\frac{1}{t-1} \bar{\mathbf{U}}_i^{(t)} + \frac{1}{t-1} \mathbf{u}_i^{(t)}.$$

to further write

$$\left\langle \frac{1}{t-1} \left(-\mathbf{u}_i^{(t)} + \bar{\mathbf{U}}_i^{(t)} \right) + \left(\lambda_i + \frac{1}{\eta t} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \left(\lambda_i + \frac{1}{\eta(t-1)} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}' \right\rangle = 0 \quad (14.7)$$

From Equation (14.6) we find

$$\langle \bar{\mathbf{U}}_i^{(t)}, \mathbf{x} - \mathbf{x}' \rangle = \left\langle \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i) + \frac{1}{\eta t} \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}), \mathbf{x} - \mathbf{x}' \right\rangle$$

and so, plugging back the previous relationship in Equation (14.7) we can write, for all $\mathbf{x}, \mathbf{x}' \in \Delta^{A_i}$,

$$\begin{aligned} 0 &= \left\langle \frac{1}{t-1} \left(-\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i) + \frac{1}{\eta t} \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) \right) \right. \\ &\quad \left. + \left(\lambda_i + \frac{1}{\eta t} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \left(\lambda_i + \frac{1}{\eta(t-1)} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}' \right\rangle \\ &= \left\langle \frac{1}{t-1} \left(-\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i) \right) + \left(\lambda_i + \frac{1}{\eta(t-1)} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) \right. \\ &\quad \left. - \left(\lambda_i + \frac{1}{\eta(t-1)} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}' \right\rangle \end{aligned}$$

$$\begin{aligned}
& - \left(\lambda_i + \frac{1}{\eta(t-1)} \right) \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}' \rangle \\
= & \left\langle \frac{1}{t-1} \left(-\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i) \right) + \frac{\eta \lambda_i t + 1}{\eta(t-1)} \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) \right. \\
& \left. - \frac{\eta \lambda_i t + 1}{\eta(t-1)} \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}' \right\rangle.
\end{aligned}$$

Dividing by $(\eta \lambda_i t + 1)/(\eta(t-1))$ yields the statement. \square

Corollary 14.1. Fix any player i , $\lambda_i \in \Lambda_i$, and $t \geq 1$. For all $\mathbf{x} \in \Delta^{A_i}$, the iterates $\mathbf{x}_{i,\lambda_i}^{(t)}$ and $\mathbf{x}_{i,\lambda_i}^{(t+1)}$ defined in Line 4 of Algorithm 14.1 satisfy

$$\begin{aligned}
& \left\langle -\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i), \mathbf{x} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle \\
& = \left(\lambda_i t + \frac{1}{\eta} \right) \left(\text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t+1)}) - \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) + \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) \right).
\end{aligned}$$

Proof. Since Lemma 14.1 holds for all $\mathbf{x}, \mathbf{x}' \in \Delta^{A_i}$, we can in particular set $\mathbf{x}' = \mathbf{x}_{i,\lambda_i}^{(t+1)}$, and obtain

$$\begin{aligned}
& \frac{\eta}{\eta \lambda_i t + 1} \left\langle -\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i), \mathbf{x} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle \\
& + \left\langle \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle = 0. \quad (14.8)
\end{aligned}$$

Using the three-point identity

$$\left\langle \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}) - \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle = \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) - \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t+1)}) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \mathbf{x}_{i,\lambda_i}^{(t)})$$

in Equation (14.8) yields

$$\begin{aligned}
& \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t+1)}) = \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) \\
& + \frac{\eta}{\eta \lambda_i t + 1} \left\langle -\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i), \mathbf{x} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle.
\end{aligned}$$

Multiplying by $\lambda_i t + 1/\eta$ yields the statement. \square

14.3.2 Regret analysis

Let $\tilde{u}_{i,\lambda_i}^{(t)}$ be the regularized utility of agent type $\lambda_i \in \Lambda_i$

$$\tilde{u}_{i,\lambda_i}^{(t)} : \Delta^{\mathcal{A}_i} \ni \mathbf{x} \mapsto \langle \mathbf{u}_i^{(t)}, \mathbf{x} \rangle - \lambda_i \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_i).$$

Observation 14.1. We note the following:

- For any $i \in \{1, 2\}$ and $\lambda_i \in \Lambda_i$, the function $\tilde{u}_{i,\lambda_i}^{(t)}$ satisfies

$$\tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}) = \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}') + \langle \nabla \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}'), \mathbf{x} - \mathbf{x}' \rangle - \lambda_i \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_i) \quad \forall \mathbf{x}, \mathbf{x}' \in \Delta^{\mathcal{A}_i}.$$

- Furthermore,

$$-\nabla \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}) = -\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i).$$

Using [Corollary 14.1](#) we have the following

Lemma 14.2. For any player i and type $\lambda_i \in \Lambda_i$,

$$\begin{aligned} \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}) - \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}) &\leq \frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \lambda_i \left(\text{KL}(\mathbf{x}_{i,\lambda_i}^{(t)} \parallel \boldsymbol{\tau}_i) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \boldsymbol{\tau}_i) \right) \\ &\quad - \left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t+1)}) + \left(\lambda_i(t-1) + \frac{1}{\eta} \right) \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}). \end{aligned}$$

Proof. From [Lemma 14.1](#),

$$\begin{aligned} 0 &= \left(\lambda_i t + \frac{1}{\eta} \right) \left(-\text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t+1)}) + \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) \right) \\ &\quad + \langle -\nabla \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \rangle \\ &= \left(\lambda_i t + \frac{1}{\eta} \right) \left(-\text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t+1)}) + \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) \right) \\ &\quad + \langle \nabla \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x}_{i,\lambda_i}^{(t)} + \mathbf{x}_{i,\lambda_i}^{(t+1)} \rangle + \langle -\nabla \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x} - \mathbf{x}_{i,\lambda_i}^{(t)} \rangle \\ &= \left(\lambda_i t + \frac{1}{\eta} \right) \left(-\text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t+1)}) + \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) \right) \\ &\quad + \langle -\nabla \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \rangle - \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}) + \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)}). \end{aligned}$$

Rearranging, we find

$$\tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}) - \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}) = -\left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t+1)}) + \left(\lambda_i(t-1) + \frac{1}{\eta} \right) \text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(t)})$$

$$- \left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \mathbf{x}_{i,\lambda_i}^{(t)}) + \underbrace{\langle -\nabla \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \rangle}_{(14.9)}. \quad (14.10)$$

We now upper bound the term in (14.9) using convexity of the function $\mathbf{x} \mapsto \text{KL}(\mathbf{x} \| \boldsymbol{\tau}_i)$, as follows:

$$\begin{aligned} & \langle -\nabla \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}), \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \rangle \\ &= \langle -\mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \rangle + \lambda_i \langle \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i), \mathbf{x}_{i,\lambda_i}^{(t+1)} - \mathbf{x}_{i,\lambda_i}^{(t)} \rangle \\ &\leq \langle -\mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \rangle + \lambda_i \left(\text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \boldsymbol{\tau}_i) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t)} \| \boldsymbol{\tau}_i) \right). \end{aligned}$$

Substituting the above bound into (14.10) yields

$$\begin{aligned} \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}) - \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}) &\leq \langle -\mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \rangle - \left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \mathbf{x}_{i,\lambda_i}^{(t)}) \\ &\quad + \lambda_i \left(\text{KL}(\mathbf{x}_{i,\lambda_i}^{(t)} \| \boldsymbol{\tau}_i) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \boldsymbol{\tau}_i) \right) \\ &\quad - \left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x} \| \mathbf{x}_{i,\lambda_i}^{(t+1)}) + \left(\lambda_i(t-1) + \frac{1}{\eta} \right) \text{KL}(\mathbf{x} \| \mathbf{x}_{i,\lambda_i}^{(t)}) \\ &\leq \frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \left(\lambda_i t + \frac{1}{\eta} \right) \|\mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)}\|_1^2 \\ &\quad - \left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \mathbf{x}_{i,\lambda_i}^{(t)}) \\ &\quad + \lambda_i \left(\text{KL}(\mathbf{x}_{i,\lambda_i}^{(t)} \| \boldsymbol{\tau}_i) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \boldsymbol{\tau}_i) \right) \\ &\quad - \left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x} \| \mathbf{x}_{i,\lambda_i}^{(t+1)}) + \left(\lambda_i(t-1) + \frac{1}{\eta} \right) \text{KL}(\mathbf{x} \| \mathbf{x}_{i,\lambda_i}^{(t)}), \end{aligned}$$

where the second inequality follows from Young's inequality. Finally, by using the strong convexity of the KL divergence between points $\mathbf{x}_{i,\lambda_i}^{(t)}$ and $\mathbf{x}_{i,\lambda_i}^{(t+1)}$, that is,

$$\text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \mathbf{x}_{i,\lambda_i}^{(t)}) \geq \|\mathbf{x}_{i,\lambda_i}^{(t+1)} - \mathbf{x}_{i,\lambda_i}^{(t)}\|_1^2,$$

yields the statement. \square

Noting that the right-hand side of Lemma 14.2 is telescopic, we immediately have the following.

Theorem 14.3. For any player i and type $\lambda_i \in \Lambda_i$, and strategy $\mathbf{x} \in \Delta^{A_i}$, the following regret bound holds at all times T :

$$\sum_{t=1}^T \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}) - \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}) \leq \frac{W^2}{4} \min\left\{\frac{2 \log T}{\lambda_i}, T\eta\right\} + \frac{\log n_i}{\eta} + \lambda_i(\log n_i + Q_i).$$

Proof. From Lemma 14.2 we have that

$$\begin{aligned} \sum_{t=1}^T \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}) - \tilde{u}_{i,\lambda_i}^{(t)}(\mathbf{x}_{i,\lambda_i}^{(t)}) &\leq \left(\frac{W^2}{4} \sum_{t=1}^T \frac{1}{\lambda_i t + 1/\eta}\right) + \lambda_i \text{KL}(\mathbf{x}_{i,\lambda_i}^{(1)} \parallel \boldsymbol{\tau}_i) + \frac{\text{KL}(\mathbf{x} \parallel \mathbf{x}_{i,\lambda_i}^{(1)})}{\eta} \\ &\leq \frac{W^2}{4} \left(\sum_{t=1}^T \min\left\{\frac{1}{\lambda_i t}, \eta\right\}\right) + \lambda_i(\log n_i + Q_i) + \frac{\log n_i}{\eta} \\ &\leq \frac{W^2}{4} \min\left\{\frac{2 \log T}{\lambda_i}, \eta T\right\} + \lambda_i(\log n_i + Q_i) + \frac{\log n_i}{\eta}, \end{aligned}$$

where the second inequality follows from the fact that $\lambda_i t + 1/\eta \geq \max\{\lambda_i t, 1/\eta\}$ and the fact that $\mathbf{x}_{i,\lambda_i}^{(1)}$ is the uniform strategy. \square

14.3.3 Last-Iterate Convergence in Two-Player Zero-Sum Games

In two-player game with payoff matrix \mathbf{U} for Player 1, a Bayes-Nash equilibrium to the regularized game is a collection of strategies $(\mathbf{x}_{i,\lambda_i}^*)$ such that for any supported type λ_i of Player $i \in \{1, 2\}$, the strategy $\mathbf{x}_{i,\lambda_i}^*$ is a best response to the average strategy of the opponent. In symbols,

$$\mathbf{x}_{1,\lambda_1}^* \in \arg \max_{\mathbf{x} \in \Delta^{A_1}} \left\{ \langle \mathbf{U} \mathbb{E}_{\lambda_2 \sim \beta_2} [\mathbf{x}_{2,\lambda_2}^*], \mathbf{x} \rangle + \lambda_1 \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_1) \right\} \quad \forall \lambda_1 \in \Lambda_1,$$

$$\mathbf{x}_{2,\lambda_2}^* \in \arg \max_{\mathbf{x} \in \Delta^{A_2}} \left\{ \langle -\mathbf{U}^\top \mathbb{E}_{\lambda_1 \sim \beta_1} [\mathbf{x}_{1,\lambda_1}^*], \mathbf{x} \rangle + \lambda_2 \text{KL}(\mathbf{x} \parallel \boldsymbol{\tau}_2) \right\} \quad \forall \lambda_2 \in \Lambda_2.$$

Denoting $\bar{\mathbf{x}}_1^* := \mathbb{E}_{\lambda_1 \sim \beta_1} [\mathbf{x}_{1,\lambda_1}^*]$, $\bar{\mathbf{x}}_2^* := \mathbb{E}_{\lambda_2 \sim \beta_2} [\mathbf{x}_{2,\lambda_2}^*]$, the first-order optimality conditions for the best response problems above are

$$\begin{aligned} \langle \mathbf{U} \bar{\mathbf{x}}_2^* + \lambda_1 \nabla \varphi_{\text{ent}}(\mathbf{x}_{1,\lambda_1}^*) - \lambda_1 \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_1), \mathbf{x}_{1,\lambda_1}^* - \mathbf{x}'_{1,\lambda_1} \rangle &\geq 0 \quad \forall \mathbf{x}'_{1,\lambda_1} \in \Delta^{A_1}, \\ \langle -\mathbf{U}^\top \bar{\mathbf{x}}_1^* + \lambda_2 \nabla \varphi_{\text{ent}}(\mathbf{x}_{2,\lambda_2}^*) - \lambda_2 \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_2), \mathbf{x}_{2,\lambda_2}^* - \mathbf{x}'_{2,\lambda_2} \rangle &\geq 0 \quad \forall \mathbf{x}'_{2,\lambda_2} \in \Delta^{A_2}. \end{aligned}$$

We also mention the following standard lemma.

Lemma 14.3. Let $(\mathbf{x}_{i,\lambda_i}^*)_{i \in \{1,2\}, \lambda_i \in \Lambda_i}$ be the unique Bayes-Nash equilibrium of the regularized game. Let strategies $\mathbf{x}'_{i,\lambda_i}$ be arbitrary, and let:

- $\bar{\mathbf{x}}'_1 := \mathbb{E}_{\lambda_1 \sim \beta_1} [\mathbf{x}'_{1,\lambda_1}]$, $\bar{\mathbf{x}}'_2 := \mathbb{E}_{\lambda_2 \sim \beta_2} [\mathbf{x}'_{2,\lambda_2}]$;
- $\alpha := \mathbb{E}_{\lambda_1 \sim \beta_1} [\langle -\mathbf{U}\bar{\mathbf{x}}'_2 + \lambda_1 \nabla \varphi_{\text{ent}}(\mathbf{x}'_{1,\lambda_1}) - \lambda_1 \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_1), \mathbf{x}_{1,\lambda_1}^* - \mathbf{x}'_{1,\lambda_1} \rangle]$;
- $\beta := \mathbb{E}_{\lambda_2 \sim \beta_2} [\langle \mathbf{U}^\top \bar{\mathbf{x}}'_1 + \lambda_2 \nabla \varphi_{\text{ent}}(\mathbf{x}'_{2,\lambda_2}) - \lambda_2 \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_2), \mathbf{x}_{2,\lambda_2}^* - \mathbf{x}'_{2,\lambda_2} \rangle]$.

Then,

$$\alpha + \beta \leq - \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} [\lambda_i \text{KL}(\mathbf{x}'_{i,\lambda_i} \| \mathbf{x}_{i,\lambda_i}^*) + \lambda_i \text{KL}(\mathbf{x}_{i,\lambda_i}^* \| \mathbf{x}'_{i,\lambda_i})].$$

The following potential function will be key in the analysis:

$$\Psi^{(t)} := \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left(\lambda_i(t-1) + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^* \| \mathbf{x}_{i,\lambda_i}^{(t)}) + \lambda_i \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t)} \| \boldsymbol{\tau}_i) \right], \quad t \in \{1, 2, \dots\}.$$

Proposition 14.1. At all times $t \in \{1, 2, \dots\}$, let

$$\bar{\mathbf{x}}_{-i}^t := \mathbb{E}_{\lambda_{-i} \sim \beta_{-i}} [\mathbf{x}_{-i,\lambda_{-i}}^t].$$

The potential $\Psi^{(t)}$ satisfies the inequality

$$\Psi^{(t+1)} \leq \Psi^{(t)} + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \rangle \right].$$

Proof. By multiplying both sides of [Corollary 14.1](#) for the choice $\mathbf{x} = \mathbf{x}_{i,\lambda_i}^*$, taking expectations over $\lambda_i \sim \beta_i$, and summing over the player $i \in \{1, 2\}$, we find

$$\begin{aligned} \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^* \| \mathbf{x}_{i,\lambda_i}^{(t+1)}) \right] &= \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^* \| \mathbf{x}_{i,\lambda_i}^{(t)}) \right] \\ &\quad - \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \mathbf{x}_{i,\lambda_i}^{(t)}) \right] \end{aligned}$$

$$+ \underbrace{\sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle -\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i), \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle \right]}_{(\clubsuit)}. \quad (14.11)$$

We now proceed to analyze the last summation on the right-hand side. First,

$$\begin{aligned} (\clubsuit) &= \underbrace{\sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle -\mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i), \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right]}_{(\spadesuit)} \\ &\quad + \underbrace{\sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle -\mathbf{u}_i^{(t)} + \lambda_i \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \lambda_i \nabla \varphi_{\text{ent}}(\boldsymbol{\tau}_i), \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle \right]}_{(\heartsuit)} \\ &\quad + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right]. \end{aligned} \quad (14.12)$$

Using [Lemma 14.3](#) we can immediately write

$$(\spadesuit) \leq \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[-\lambda_i \text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(t)}) \right].$$

By manipulating the inner product in (\heartsuit) , we have

$$\begin{aligned} (\heartsuit) &= \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle -\mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle - \lambda_i \left\langle \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t)}) - \nabla \varphi_{\text{ent}}(\mathbf{x}_{i,\lambda_i}^{(t+1)}), \mathbf{x}_{i,\lambda_i}^{(t+1)} - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right] \\ &\leq \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle -\mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\rangle + \lambda_i \left(\text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \boldsymbol{\tau}_i) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t)} \parallel \boldsymbol{\tau}_i) \right) \right] \\ &\leq \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \left(\lambda_i t + \frac{1}{\eta} \right) \|\mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)}\|_1^2 \right] \\ &\quad + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\lambda_i \left(\text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \boldsymbol{\tau}_i) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t)} \parallel \boldsymbol{\tau}_i) \right) \right], \end{aligned}$$

where the last inequality follow from the fact that $ab \leq a^2/(4\rho) + \rho b^2$ for all choices of $a, b \geq 0$ and $\rho > 0$. Substituting the individual bounds into [\(14.12\)](#) yields

$$(\clubsuit) \leq \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\lambda_i \left(\text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \parallel \boldsymbol{\tau}_i) - \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t)} \parallel \boldsymbol{\tau}_i) \right) \right]$$

$$\begin{aligned}
& + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \left(\lambda_i t + \frac{1}{\eta} \right) \left\| \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\|_1^2 \right] \\
& + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right].
\end{aligned}$$

Finally, plugging the above bound into (14.11) and rearranging terms yields

$$\begin{aligned}
\Psi^{(t+1)} & \leq \Psi^{(t)} + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[- \left(\lambda_i t + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^{(t+1)} \| \mathbf{x}_{i,\lambda_i}^{(t)}) \right] \\
& + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \left(\lambda_i t + \frac{1}{\eta} \right) \left\| \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\|_1^2 \right] \\
& + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right]. \\
& \leq \Psi^{(t)} + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[- \left(\lambda_i t + \frac{1}{\eta} \right) \left\| \mathbf{x}_{i,\lambda_i}^{(t+1)} - \mathbf{x}_{i,\lambda_i}^{(t)} \right\|_1^2 \right] \\
& + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \left(\lambda_i t + \frac{1}{\eta} \right) \left\| \mathbf{x}_{i,\lambda_i}^{(t)} - \mathbf{x}_{i,\lambda_i}^{(t+1)} \right\|_1^2 \right] \\
& + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right]. \\
& \leq \Psi^{(t)} + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right],
\end{aligned}$$

as we wanted to show. \square

Theorem 14.4. As in Proposition 14.1, let

$$\bar{\mathbf{x}}_{-i}^t := \mathbb{E}_{\lambda_{-i} \sim \beta_{-i}} \left[\mathbf{x}_{-i,\lambda_{-i}}^t \right].$$

Let \tilde{D}_{KL}^T be the notion of distance defined as

$$\tilde{D}_{\text{KL}}^T := \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[(\lambda_i + 1/(\eta(T-1))) \text{KL}(\mathbf{x}_{i,\lambda_i}^* \| \mathbf{x}_{i,\lambda_i}^{(T)}) \right].$$

At all times $T = 2, 3, \dots$,

$$\begin{aligned} \tilde{D}_{\text{KL}}^T \leq & \frac{1}{T} \left(\rho + \frac{\log n_i}{\eta} + \frac{W^2}{2} \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\min \left\{ \frac{2 \log T}{\lambda_i}, \eta T \right\} \right] \right) \\ & + \frac{2}{T} \sum_{t=1}^T \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right], \end{aligned}$$

where

$$\rho := 2 \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} [\lambda_i] (\log n_i + Q_i).$$

Proof. Using the bound on $\Psi^{(t+1)} - \Psi^{(t)}$ given by [Proposition 14.1](#) we obtain

$$\begin{aligned} \Psi^{(T)} - \Psi^{(1)} &= \sum_{t=1}^{T-1} (\Psi^{(t+1)} - \Psi^{(t)}) \\ &\leq \sum_{t=1}^{T-1} \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{4\lambda_i t + 4/\eta} + \left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right] \\ &= \frac{1}{4} \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\sum_{t=1}^T \frac{\|\mathbf{u}_i^{(t)}\|_\infty^2}{\lambda_i t + 1/\eta} \right] + \sum_{t=1}^T \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right] \\ &\leq \frac{1}{4} \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\sum_{t=1}^{T-1} \frac{W^2}{\lambda_i t + 1/\eta} \right] + \sum_{t=1}^T \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right]. \end{aligned}$$

We can now bound

$$\begin{aligned} \sum_{t=1}^T \frac{W^2}{\lambda_i t + 1/\eta} &\leq W^2 \sum_{t=1}^T \min \left\{ \frac{1}{\lambda_i t}, \eta \right\} \\ &\leq W^2 \min \left\{ \sum_{t=1}^T \frac{1}{\lambda_i t}, \sum_{t=1}^T \eta \right\} \\ &\leq W^2 \min \left\{ \frac{2 \log T}{\lambda_i}, T\eta \right\}. \end{aligned}$$

On the other hand, note that

$$\begin{aligned}
\Psi^{(T)} - \Psi^{(1)} &= -\Psi^{(1)} + \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left(\lambda_i(T-1) + \frac{1}{\eta} \right) \text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(T)}) + \lambda_i \text{KL}(\mathbf{x}_{i,\lambda_i}^{(T)} \parallel \boldsymbol{\tau}_i) \right] \\
&\geq -\Psi^{(1)} + \sum_{i \in \{1,2\}} (T-1) \mathbb{E}_{\lambda_i \sim \beta_i} \left[(\lambda_i + 1/(\eta(T-1))) \text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(T)}) \right] \\
&= (T-1) \tilde{D}_{\text{KL}}^T - \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(1)})}{\eta} - \lambda_i \text{KL}(\mathbf{x}_{i,\lambda_i}^{(1)} \parallel \boldsymbol{\tau}_i) \right] \\
&\geq (T-1) \tilde{D}_{\text{KL}}^T - \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\frac{\log n_i}{\eta} + \lambda_i (\log n_i + Q_i) \right] \\
&= (T-1) \tilde{D}_{\text{KL}}^T - \rho,
\end{aligned}$$

where the last inequality follows from expanding the definition of the KL divergence and using the fact that $\mathbf{x}_{i,\lambda_i}^{(1)}$ is the uniform strategy. Combining the inequalities and dividing by $T-1$ yields

$$\begin{aligned}
\tilde{D}_{\text{KL}}^T &\leq \frac{W^2}{4} \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\min \left\{ \frac{2 \log T}{(T-1)\lambda_i}, \frac{T}{T-1} \eta \right\} \right] + \frac{\rho}{T-1} \\
&\quad + \frac{1}{T-1} \sum_{t=1}^T \sum_{i \in \{1,2\}} \mathbb{E}_{\lambda_i \sim \beta_i} \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right].
\end{aligned}$$

Finally, using the fact that $2(T-1) \geq T$ yields the statement. \square

Theorem 14.5 (Last-iterate convergence of DiL-piKL in two-player zero-sum games). Let ρ be as in the statement of Theorem 14.4. When both players in a zero-sum game learn using DiL-piKL for T iterations, their strategies converge to the unique Bayes-Nash equilibrium $(\mathbf{x}_1^*, \mathbf{x}_2^*)$ of the regularized game defined by utilities (14.5), in the following senses:

(a) In expectation: for all $i \in \{1, 2\}$ and $\lambda_i \in \Lambda_i$, at a rate of roughly $\log T/(\lambda_i T)$

$$\mathbb{E} \left[\text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(T)}) \right] \leq \frac{1}{\lambda_i T} \left(\rho + \frac{\log n_i}{\eta} + \frac{W^2}{2} \sum_{j \in \{1,2\}} \mathbb{E}_{\lambda_j \sim \beta_j} \left[\min \left\{ \frac{2 \log T}{\lambda_j}, \eta T \right\} \right] \right).$$

(We remark that for $\eta = 1/\sqrt{T}$ the convergence is never slower than $1/\sqrt{T}$).

(b) With high probability, at a rate of roughly $1/\sqrt{T}$: for any $\delta \in (0, 1)$ and Player $i \in \{1, 2\}$,

$$\mathbb{P} \left[\forall \lambda_i \in \Lambda_i : \text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(T)}) \leq \mathbb{E} \left[\text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(T)}) \right] + \frac{8\sqrt{2}W}{\lambda_i\sqrt{T}} \sqrt{\log \frac{|\Lambda_i|}{\delta}} \right] \geq 1 - \delta.$$

A n upper bound on $\mathbb{E} \left[\text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(T)}) \right]$ was given in the previous point.

(c) Almost surely in the limit:

$$\mathbb{P} \left[\forall \lambda_i \in \Lambda_i : \text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(T)}) \xrightarrow{T \rightarrow +\infty} 0 \right] = 1 \quad \forall i \in \{1, 2\}.$$

Proof. We prove the three statements incrementally.

(a) Let \mathcal{F}_t be the σ -algebra generated by $\{\mathbf{u}_i^{(t')} : t' = 1, \dots, t-1, i \in \{1, 2\}\}$. We let $\mathbb{E}_t[\cdot] := \mathbb{E}[\cdot \mid \mathcal{F}_t]$. Since piKL is a deterministic algorithm, $\mathbf{x}_{i,\lambda_i}^{(t)}$ is \mathcal{F}_t -measurable. Hence, given that \mathbf{u}_i^t is an unbiased estimator of $\mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)}$ we have that at all times t

$$\mathbb{E}_t \left[\left\langle \mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)}, \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle \right] = \left\langle \mathbb{E}_t \left[\mathbf{U}_i \bar{\mathbf{x}}_{-i}^{(t)} - \mathbf{u}_i^{(t)} \right], \mathbf{x}_{i,\lambda_i}^* - \mathbf{x}_{i,\lambda_i}^{(t)} \right\rangle = 0. \quad (14.13)$$

Note that from the definition of \tilde{D}_{KL}^T given in [Theorem 14.4](#)

$$\text{KL}(\mathbf{x}_{i,\lambda_i}^* \parallel \mathbf{x}_{i,\lambda_i}^{(T)}) \leq \frac{1}{\lambda_i} \tilde{D}_{\text{KL}}^T. \quad (14.14)$$

Hence, taking expectations and using (14.13) yields the statement.

(b) To prove high-probability convergence, we use the Azuma-Hoeffding concentration inequality. In particular, (14.13) shows that the stochastic process

$$\left(\sum_{j \in \{1, 2\}} \mathbb{E}_{\lambda_j \sim \beta_j} \left[\left\langle \mathbf{U}_j \bar{\mathbf{x}}_{-j}^{(t)} - \mathbf{u}_j^{(t)}, \mathbf{x}_j^* - \mathbf{x}_j^{(t)} \right\rangle \right] \right)_{t=1, 2, \dots}$$

is a martingale difference sequence adapted to the filtration \mathcal{F}_t . Furthermore, note that

$$\left| \sum_{j \in \{1, 2\}} \mathbb{E}_{\lambda_j \sim \beta_j} \left[\left\langle \mathbf{U}_j \bar{\mathbf{x}}_{-j}^{(t)} - \mathbf{u}_j^{(t)}, \mathbf{x}_{j,\lambda_j}^* - \mathbf{x}_{j,\lambda_j}^{(t)} \right\rangle \right] \right| \leq 4W$$

for all t . Hence, using the Azuma-Hoeffding inequality for martingale difference sequences we obtain that for all $\delta \in (0, 1)$,

$$\mathbb{P} \left[\sum_{t=1}^T \sum_{j \in \{1,2\}} \mathbb{E}_{\lambda_j \sim \beta_j} \left[\langle \mathbf{U}_j \bar{\mathbf{x}}_{-j}^{(t)} - \mathbf{u}_j^{(t)}, \mathbf{x}_j^* - \mathbf{x}_j^{(t)} \rangle \right] \leq 4W \sqrt{2T \log \frac{1}{\delta}} \right] \geq 1 - \delta.$$

Plugging the above probability bound in the statement of [Theorem 14.4](#) and using the union bound over $\lambda_i \in \Lambda_i$ yields the statement.

(c) follows from (b) via a standard application of the Borel-Cantelli lemma. □

14.4 Experimental evaluation in no-press Diplomacy

We empirically investigate using DiL-piKL as the planning algorithm used at each decision-point within the DORA framework, which we now recall.

14.4.1 Background on Double Oracle Reinforcement learning for Action exploration (DORA)

DORA (Bakhtin, D. Wu, Lerer, and Brown, 2021) is an algorithm similar to past model-based reinforcement-learning methods such as AlphaZero (Silver, Hubert, Schrittwieser, Antonoglou, Lai, Guez, Lanctot, Sifre, Kumaran, Graepel, et al., 2018), except that in place of Monte Carlo tree search—which is unsound in simultaneous-action games such as Diplomacy or other imperfect-information extensive-form games—it instead uses an equilibrium-finding algorithm such as hedge or RM to iteratively approximate a Nash equilibrium for the current state (*i.e.*, one-step lookahead search). A deep neural net trained to predict the strategy is used to sample plausible actions for all players to reduce the large action space in Diplomacy down to a tractable subset for the equilibrium-finding procedure, and a deep neural net trained to predict state values is used to evaluate the results of joint actions sampled by this procedure. Beginning with a strategy and value network randomly initialized from scratch, a large number of self-play games are played and the resulting equilibrium strategies and the improved 1-step value estimates computed on every turn from equilibrium-finding are added to a replay buffer used for subsequently improving the strategy and value. Additionally, a double-oracle (McMahan, G. Gordon, and Blum, 2003) method was used to allow the strategy to explore and discover additional actions, and the same equilibrium-finding procedure was also used at test time.

14.4.2 Training of our bot Diplodocus

Our training algorithm closely follows that of DORA, described above. However, the key difference is that in place of RM to compute the equilibrium strategy σ on each turn of a game

during self-play, we use DiL-piKL with a λ distribution and human imitation anchor strategy τ that is fixed for all of training. We call this self-play algorithm RL-DiL-piKL. We use RL-DiL-piKL to train value and strategy proposal networks and use DiL-piKL during test-time search. The call final trained agent is called *Diplodocus*.

Following Bakhtin, D. Wu, Lerer, and Brown, 2021, at evaluation time we perform 1-ply lookahead where on each turn we sample up to 30 of the most likely actions for each player from the RL strategy proposal network. However, rather than using RM to compute the equilibrium σ , we apply DiL-piKL.

As also mentioned previously in Section 14.3, while our agent samples λ_i from the probability distribution β_i when computing the DiL-piKL equilibrium, the agent chooses its own action to actually play using a fixed low λ . For all experiments, including all ablations, the agent uses the same BC anchor strategy. For DiL-piKL experiments for each player i we set β_i to be uniform over $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and play according to $\lambda = 10^{-4}$, except for the first turn of the game. On the first turn we instead sample from $\{10^{-2}, 10^{-1.5}, 10^{-1}, 10^{-0.5}\}$ and play according to $\lambda = 10^{-2}$, so that the agent plays more diverse openings, which more closely resemble those that humans play.

14.4.3 Experimental setup

We compare the performance of two variants of Diplodocus both in a population of prior agents and other baseline agents, and in a tournament with humans.

In order to measure the ability of agents to play well against a diverse set of opponents, we play many games between AI agents where each of the seven players are sampled randomly from a population of baselines or the agent to be tested. We report scores for each of the following algorithms against the baseline population.

- **Diplodocus-Low** and **Diplodocus-High** are the proposed agents that use RL-DiL-piKL during training with player types $\{10^{-4}, 10^{-1}\}$ and $\{10^{-2}, 10^{-1}\}$, respectively.
- **DORA** is an agent that is trained via self-play and uses RM as the search algorithm during training and test-time. Both the strategy and the value function are randomly initialized at the start of training.
- **DNVI** is similar to DORA, but the strategy proposal and value networks are initialized from human behavioral-cloning pretraining.
- **DNVI-NPU** is similar to DNVI, but during training only the RL value network is updated. The strategy proposal network is trained but never fed back to self-play workers, to limit drift from human conventions. The trained strategy proposal is only used at test time (along with the RL value network).

- **BRBot** is an approximate best response to the behavioral-cloning strategy. It was trained the same as Diplodocus, except that during training the agent plays one distinguished player each game with $\lambda = 0$ while all other players use $\lambda \approx \infty$.
- **SearchBot** is a one-step lookahead equilibrium search agent from the paper by Gray, Lerer, Bakhtin, and Brown (2020), evaluated using their published model.
- **HedgeBot** is an agent similar to SearchBot (Gray, Lerer, Bakhtin, and Brown, 2020) but using our latest architecture and using hedge rather than RM as the equilibrium-finding algorithm.
- **FPPI-2** and **SL** are two agents from the work by Anthony, Eccles, Tacchetti, Kramár, Gemp, Hudson, Porcel, Lanctot, Perolat, Everett, Singh, Graepel, and Bachrach (2020), evaluated using their published model.^[14.a]

14.4.4 Performance compared to prior algorithms

We report results for Diplodocus against prior algorithms for the problem. The results, shown in Table 14.1, show Diplodocus-Low and Diplodocus-High perform the best by a wide margin.

Agent	Reference	Score against population
Diplodocus-Low		29% \pm 1%
Diplodocus-High		28% \pm 1%
DNVI-NPU (retrained)	Bakhtin, D. Wu, Lerer, and Brown, 2021	20% \pm 1%
BRBot		18% \pm 1%
DNVI (retrained)	Bakhtin, D. Wu, Lerer, and Brown, 2021	15% \pm 1%
HedgeBot (retrained)	Jacob, David J. Wu, Farina, Lerer, Hu, Bakhtin, Andreas, and Brown, 2022	14% \pm 1%
DORA (retrained)	Bakhtin, D. Wu, Lerer, and Brown, 2021	13% \pm 1%
FPPI-2	Anthony, Eccles, Tacchetti, Kramár, Gemp, Hudson, Porcel, Lanctot, Perolat, Everett, Singh, Graepel, and Bachrach, 2020	9% \pm 1%
SearchBot	Gray, Lerer, Bakhtin, and Brown, 2020	7% \pm 1%
SL	Anthony, Eccles, Tacchetti, Kramár, Gemp, Hudson, Porcel, Lanctot, Perolat, Everett, Singh, Graepel, and Bachrach, 2020	6% \pm 1%

Table 14.1: Performance of different algorithms. Agents above the line were retrained. Agents below the line were evaluated using the models and the parameters provided by the authors. The \pm shows one standard error.

^[14.a]<https://github.com/deepmind/diplomacy>

14.4.5 Experiments against human players

we organized a tournament where we evaluated four agents for 50 games each in a population of online human participants. We evaluated two baseline agents, BRBot and DORA, and our new agents, Diplodocus-Low and Diplodocus-High.

In order to limit the duration of games to only a few hours, these games used a time limit of 5 minutes per turn and a stochastic game-end rule where at the beginning of each game year between 1909 and 1912 the game ends immediately with 20% chance per year, increasing in 1913 to a 40% chance. Players were not told which turn the game would end on for a specific game, but were told the distribution it was sampled from. Our agents were also trained based on this distribution.^[14.b] Players were recruited from Diplomacy mailing lists and from webdiplomacy.net. In order to mitigate the risk of cheating by collusion, players were paid hourly rather than based on in-game performance. Each game had exactly one agent and six humans. The players were informed that there was an AI agent in each game, but did not know which player was the bot in each particular game. In total 62 human participants played 200 games with 44 human participants playing more than two games and 39 human participants playing at least 5 games.

We report results for the human tournament in [Table 14.2](#). For each listed player, we report their average score, Elo rating, and rank within the tournament based on Elo among players who played at least 5 games. Elo ratings were computed using a standard generalization of BayesElo (Coulom, 2005) to multiple players (Hunter, 2004). This gives similar rankings as average score, but also attempts to correct for both the average strength of the opponents, since some games may have stronger or weaker opposition, as well as for which of the seven European powers a player was assigned in each game, since some starting positions in Diplomacy are advantaged over others. To regularize the model, a weak Bayesian prior was applied such that each player’s rating was normally distributed around 0 with a standard deviation of around 350 Elo.

The results show that Diplodocus-High performed best among all the humans by both Elo and average score. Diplodocus-Low followed closely behind, ranking second according to average score and third by Elo. BRBot performed relatively well, but ended ranked below that of both DiL-piKL agents and several humans. DORA performed relatively poorly.

Two participants achieved a higher average score than the Diplodocus agents, a player averaging 35% but who only played two games, and a player with a score of 29% who played only one game.

We note that given the large statistical error margins, the results in [Table 14.2](#) do not conclusively demonstrate that Diplodocus outperforms the best human players, nor do they alone demonstrate an unambiguous separation between Diplodocus and BRBot. However, the results do indicate

^[14.b]Games were run by a third-party contractor. In contradiction of the criteria we specified, the contractor ended games artificially early for the first ~80 games played in the tournament, with end dates of 1909-1911 being more common than they should have been. We immediately corrected this problem once it was identified.

	Rank	Elo	Average score	# Games
Diplodocus-High	1	181	27% \pm 4%	50
Human	2	162	25% \pm 6%	13
Diplodocus-Low	3	152	26% \pm 4%	50
Human	4	138	22% \pm 9%	7
Human	5	136	22% \pm 3%	57
BRBot	6	119	23% \pm 4%	50
Human	7	102	18% \pm 8%	8
Human	8	96	17% \pm 3%	51
...
DORA	32	-20	13% \pm 3%	50
...
Human	43	-187	1% \pm 1%	7

Table 14.2: Performance of four different agents in a population of human players, ranked by Elo, among all 43 participants who played at least 5 games. The \pm shows one standard error.

that Diplodocus performs at least at the level of expert players in this population of players with diverse skill levels. Additionally, the superior performance of both Diplodocus agents compared to BRBot is consistent with the results from the agent population experiments in [Table 14.1](#).

In addition to the tournament, we asked three expert human players to evaluate the strength of the agents in the tournament games based on the quality of their actions. Games were presented to these experts with anonymized labels so that the experts were *not* aware of which agent was which in each game when judging that agent’s strategy. All the experts picked a Diplodocus agent as the strongest agent, though they disagreed about whether Diplodocus-High or Diplodocus-Low was best. Additionally, all experts indicated one of the Diplodocus agents as the one they would most like to cooperate with in a game.

Conclusions and future work

This dissertation sought to provide solid theoretical and algorithmic foundations for strategic, game-theoretic decision-making in imperfect-information extensive-form games.

In the first part of the dissertation, we focused on *learning dynamics*, procedures each player can use to iteratively refine their strategy. Learning dynamics are a fascinating technique, which can recover *global* notions of game-theoretic optimality from *local* notions of strategy improvements.

In this dissertation, we gave several new fundamental results about learning dynamics in imperfect-information extensive-form games. In [Chapter 4](#) we presented a methodology for constructing predictive dynamics with state-of-the-art practical performance in many game instances. In [Chapter 5](#) we investigated several fundamental questions about the metric structure of strategy spaces in imperfect-information extensive-form games, establishing the first notion of distance that enables linear-time projections while at the same time guaranteeing polynomial distance between any two strategies. In [Chapters 6 and 7](#) we settled for the positive the question of whether near-optimal no-regret learning can be achieved in general imperfect-information extensive-form games, all while improving bounds for normal-form games as well. Finally, in [Chapter 8](#) we provided the first uncoupled, polynomial-time learning dynamics leading to extensive-form correlated equilibrium (the natural generalization of the classic correlated equilibrium in normal-form games), closing a longstanding open problem in the literature. As we pointed out along the way, learning in imperfect-information games is significantly more challenging (and fascinating) than learning in normal-form (nonsequential) games, and often requires different techniques. However, surprisingly many positive results can be given. I hope these results will be useful foundations for future investigations of learning in imperfect-information games.

Several questions about learning in extensive-form games remain open. We mention just a few. First, what are lower bounds for no-external-regret dynamics, both in terms of dependence on the number of repetitions T of the game, and as a function of the dimensions of the game? More concretely, is the state-of-the-art dependence on the game size we developed in [Chapter 7](#) optimal? Can learning dynamics in games with provable $\mathcal{O}_T(1/T)$ convergence to coarse-correlated equilibrium be given? And is there an inherent tradeoff between rate of convergence

and complexity of iterations?^[14.c]

Another set of questions relate to alternative feedback models. Is it possible to extend most (or all) of the no-regret learning dynamics described so far to online learning models such as bandit and semi-bandit settings? To achieve that, we need to construct several tools of independent interest, such as efficient loss estimation techniques and the introduction of certain combinatorially-structured local norms to study convergence of the learning algorithm. While some work has already been done (see, *e.g.*, Farina, Schmucker, and Sandholm (2021), Farina and Sandholm (2021b), Kozuno, Ménard, Rémi Munos, and Valko (2021), Bai, C. Jin, Mei, Song, and Yu (2022), Song, Mei, and Bai (2022), McAleer, Farina, Lanctot, and Sandholm (2023), and Fiegel, Ménard, Kozuno, Rémi Munos, Perchet, and Valko (2023)), significant work remains to be done.

In the second part of the dissertation, we focused on the geometry of correlated strategies in imperfect-information games. This topic does not have a counterpart in the theory of normal-form games, as the difficulty in characterizing correlated strategies lies in the asymmetric information that each player observes while playing the imperfect-information game. By making progress on the fundamental problem of characterizing the polytope of correlated strategies in Chapter 9, we were able to provide positive complexity results and new algorithms for several domains, including welfare-maximizing solution concepts (Chapter 10) and TMECor equilibria in adversarial team games (Chapter 11). This rippling effect shows that much value is to be gained from attacking the fundamental question of the structure of correlation in imperfect-information games. In the future, it would be interesting to further expand the class of games for which the set of correlated strategies can be characterized (and optimized over) efficiently, and further push the scalability of existing methods.

The special combinatorial structure of the polytope of correlated strategies has also enabled us to develop the first learning algorithms for optimal correlated solution concepts and TMECor strategies. These techniques are the current practical state of the art. As of now, these learning algorithms require that correlating players update their strategies in a rather coupled manner. By shedding even more light on the structure of correlation, it would be interesting to understand whether this requirement could be somehow relaxed.

Finally, in the last part of the dissertation we focused on learning and equilibrium computation in the presence of imperfect players. As we argued in Chapter 14, it is important to train artificial intelligence by taking into account a model of human play, while at the same time avoiding the inherent strategic weaknesses and potential for manipulation of imitation learning. This area of the literature seems to still be in its infancy, both in terms of solution concepts and scalable computational techniques for computing those concepts. In Chapters 12 and 13 we focused on trembling-hand refinements in two-player games, standard refinements of the Nash equilibrium that provably avoid sequentially-irrational behavior and that are therefore more robust to mistakes

^[14.c]For example, can variance reduction methods, such as those of Carmon, Y. Jin, Sidford, and Tian (2019) for normal-form games, be meaningfully extended to imperfect-information extensive-form games?

of the players. These solution concepts had remained a theoretical technique, and before the work in this dissertation, it was an open question how to compute them. After settling the recognized open problem of the complexity of computing extensive-form perfect equilibria in [Chapter 12](#), in [Chapter 13](#) we gave the first practical algorithm for computing exact trembling-hand equilibrium refinements in real (non-synthetic) games with up to half a billion terminal nodes. These games are 4-5 orders of magnitude larger than what could be handled with prior techniques. Using our results, it would be interesting to assess empirically trembling-hand refinements deliver on their theoretical promises, for example, by evaluating the performance of artificial intelligence trained to converge to refinements against humans, and comparing with prior techniques for unrefined equilibria.

In addition to the above questions, furthering our understanding of the computational aspects of equilibrium computation and learning in other classes of structured games, beyond imperfect-information extensive-form games, is a natural and important direction of research. For example, some strategic interactions might be better conceptualized as taking place on a DAG or on a generic directed graph, rather than a tree. In other applications, the specific class of games has such a strong combinatorial structure that it might be advantageous to resort to different (for example, factorized) representations of the interaction. This might apply to sequential auctions, security games, and other games whose action space has a strong combinatorial flavor. More generally, one could also try to design algorithms that are able to automatically exploit symmetries, structures, and invariants of the games, constructing an appropriate representation on the fly. Infinite games might also be considered. These could arise from reasoning about infinite repetitions of a finite game, or from continuous action spaces. Orthogonally to these models, fundamental questions remain regarding what solution concepts are even meaningful in these settings.

With the rapid advancements in machine learning, game theory, and artificial intelligence, we have the potential to create systems that can make strategic, sound, intentional, optimal decisions in ways and at scales that were once thought impossible. I have no doubt that as we continue to rely more and more on technology and artificial intelligence in our daily lives, these systems will help us make better and more efficient decisions. In fact, systems developed using technology covered in this dissertation are already able, today, to make strategic decisions that are often superior to human ones in certain settings (including some involving natural language and cooperation, such as Diplomacy). It is my hope that the work presented in this dissertation will inspire others to continue pushing the boundaries of what is possible. As we move forward, I am eager to be a part of the ongoing effort to develop machines that can make strategic decisions, with both old and new colleagues, and to explore the many exciting open questions that lie ahead.

Bibliography

- Abe, Kenshi, Mitsuki Sakamoto, and Atsushi Iwasaki (2022). “Mutation-driven follow the regularized leader for last-iterate convergence in zero-sum games”. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.
- Abernethy, Jacob and Alexander Rakhlin (2009). *Beating the Adaptive Bandit with High Probability*. Tech. rep. UCB/EECS-2009-10. EECS Department, University of California, Berkeley. URL: <http://www2.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-10.html>.
- Anagnostides, Ioannis, Constantinos Daskalakis, Gabriele Farina, Maxwell Fishelson, Noah Golowich, and Tuomas Sandholm (2022). “Near-Optimal No-Regret Learning for Correlated Equilibria in Multi-Player General-Sum Games”. In: *Proceedings of the Annual Symposium on Theory of Computing (STOC)*.
- Anagnostides, Ioannis, Gabriele Farina, Christian Kroer, Andrea Celli, and Tuomas Sandholm (2022). “Faster No-Regret Learning Dynamics for Extensive-Form Correlated and Coarse Correlated Equilibrium”. In: *Proceedings of the ACM Conference on Economics and Computation (EC)*.
- Anagnostides, Ioannis, Gabriele Farina, Christian Kroer, Chung-Wei Lee, Haipeng Luo, and Tuomas Sandholm (2022). “Uncoupled Learning Dynamics with $O(\log T)$ Swap Regret in Multiplayer Games”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Anagnostides, Ioannis, Gabriele Farina, and Tuomas Sandholm (2023). “Near-Optimal Φ -Regret Learning in Extensive-Form Games”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Anagnostides, Ioannis, Ioannis Panageas, Gabriele Farina, and Tuomas Sandholm (2022). “On Last-Iterate Convergence Beyond Zero-Sum Games”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.

- Anthony, Thomas, Tom Eccles, Andrea Tacchetti, János Kramár, Ian Gemp, Thomas Hudson, Nicolas Porcel, Marc Lanctot, Julien Perolat, Richard Everett, Satinder Singh, Thore Graepel, and Yoram Bachrach (2020). “Learning to Play No-Press Diplomacy with Best Response Policy Iteration”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Audibert, Jean-Yves, Sébastien Bubeck, and Gábor Lugosi (2014). “Regret in online combinatorial optimization”. In: *Mathematics of Operations Research* 39.1, pp. 31–45.
- Azizian, Waiss, Franck Iutzeler, Jérôme Malick, and Panayotis Mertikopoulos (2021). “The Last-Iterate Convergence Rate of Optimistic Mirror Descent in Stochastic Variational Inequalities”. In: *Proceedings of the Conference on Learning Theory (COLT)*.
- Babichenko, Yakov and Aviad Rubinstein (2022). “Communication complexity of approximate Nash equilibria”. In: *Games Econ. Behav.* 134, pp. 376–398.
- Bai, Yu, Chi Jin, Song Mei, Ziang Song, and Tiancheng Yu (2022). In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Bakhtin, Anton, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, Athul Paul Jacob, Mojtaba Komeili, Karthik Konath, Minae Kwon, Adam Lerer, Mike Lewis, Alexander H. Miller, Sasha Mitts, Adithya Renduchintala, Stephen Roller, Dirk Rowe, Weiyang Shi, Joe Spisak, Alexander Wei, David Wu, Hugh Zhang, and Markus Zijlstra (2022). “Human-level play in the game of Diplomacy by combining language models with strategic reasoning”. In: *Science* 378.6624. URL: <https://www.science.org/doi/pdf/10.1126/science.ade9097>.
- Bakhtin, Anton, David Wu, Adam Lerer, and Noam Brown (2021). “No-Press Diplomacy from Scratch”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Bakhtin, Anton, David J Wu, Adam Lerer, Jonathan Gray, Athul Paul Jacob, Gabriele Farina, Alexander H Miller, and Noam Brown (2023). “Mastering the Game of No-Press Diplomacy via Human-Regularized Reinforcement Learning and Planning”. In: *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Basilico, Nicola, Andrea Celli, Giuseppe De Nittis, and Nicola Gatti (2017). “Team-maxmin equilibrium: efficiency bounds and algorithms”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Beck, Amir and Marc Teboulle (2003). “Mirror descent and nonlinear projected subgradient methods for convex optimization”. In: *Operations Research Letters* 31.3, pp. 167–175.

- Ben-Tal, Aharon and Arkadi Nemirovski (2005). “Non-Euclidean restricted memory level method for large-scale convex optimization”. In: *Mathematical Programming* 102.3, pp. 407–456.
- Ben-Tal, Ahron and Arkadi Nemirovski (2001). *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*. Vol. 2. Siam.
- Bošanský, B, Christopher Kiekintveld, V Lisý, and Michal Pěchouček (2014). “An Exact Double-Oracle Algorithm for Zero-Sum Extensive-Form Games with Imperfect Information”. In: *Journal of Artificial Intelligence Research*, pp. 829–866.
- Bošanský, Branislav and Jiří Čermák (2015). “Sequence-form algorithm for computing Stackelberg equilibria in extensive-form games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Bowling, Michael, Neil Burch, Michael Johanson, and Oskari Tammelin (2015). “Heads-up Limit Hold’em Poker is Solved”. In: *Science* 347.6218.
- Boyd, Stephen and Lieven Vandenberghe (2004). *Convex Optimization*. Cambridge University Press.
- Brown, Noam, Christian Kroer, and Tuomas Sandholm (2017). “Dynamic Thresholding and Pruning for Regret Minimization”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Brown, Noam and Tuomas Sandholm (2014). “Regret Transfer and Parameter Optimization”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Brown, Noam and Tuomas Sandholm (2015). “Regret-Based Pruning in Extensive-Form Games”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Brown, Noam and Tuomas Sandholm (2017a). “Reduced Space and Faster Convergence in Imperfect-Information Games via Pruning”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Brown, Noam and Tuomas Sandholm (2017b). “Safe and nested subgame solving for imperfect-information games”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Brown, Noam and Tuomas Sandholm (2017c). “Superhuman AI for heads-up no-limit poker: Libratus beats top professionals”. In: *Science* 359, pp. 418–424.
- Brown, Noam and Tuomas Sandholm (2019). “Solving imperfect-information games via discounted regret minimization”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

- Burch, Neil, Michael Johanson, and Michael Bowling (2014). "Solving Imperfect Information Games Using Decomposition". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Cai, Yang, Argyris Oikonomou, and Weiqiang Zheng (2022). "Tight Last-Iterate Convergence of the Extragradient Method for Constrained Monotone Variational Inequalities". In: *CoRR abs/2204.09228*.
- Carminati, Luca, Federico Cacciamani, Marco Ciccone, and Nicola Gatti (2022). "Public Information Representation for Adversarial Team Games". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Carmon, Yair, Yujia Jin, Aaron Sidford, and Kevin Tian (2019). "Variance reduction for matrix games". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Celli, Andrea and Nicola Gatti (2018). "Computational Results for Extensive-Form Adversarial Team Games". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Celli, Andrea, Alberto Marchesi, Gabriele Farina, and Nicola Gatti (2020). "No-Regret Learning Dynamics for Extensive-Form Correlated Equilibrium". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Cesa-Bianchi, Nicolo and Gábor Lugosi (2012). "Combinatorial bandits". In: *Journal of Computer and System Sciences* 78.5, pp. 1404–1422.
- Chen, Xi, Xiaotie Deng, and Shang-Hua Teng (2009). "Settling the Complexity of Computing Two-Player Nash Equilibria". In: *Journal of the ACM*.
- Chen, Xi and Binghui Peng (2020). "Hedging in games: Faster convergence of external and swap regrets". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Chiang, Chao-Kai, Tianbao Yang, Chia-Jung Lee, Mehrdad Mahdavi, Chi-Jen Lu, Rong Jin, and Shenghuo Zhu (2012). "Online optimization with gradual variations". In: *Proceedings of the Conference on Learning Theory (COLT)*.
- Chu, Francis and Joseph Halpern (2001). "On the NP-completeness of finding an optimal strategy in games with common payoffs". In: *International Journal of Game Theory*.
- Cococcioni, Marco, Massimo Pappalardo, and Yaroslav D. Sergeyev (2018). "Lexicographic multi-objective linear programming using grossone methodology: Theory and algorithm". In: *Applied Mathematics and Computation* 318, pp. 298–311.

- Condat, Laurent (2016). “Fast projection onto the simplex and the ℓ_1 ball”. In: *Mathematical Programming* 158.1, pp. 575–585.
- Conitzer, Vincent and Tuomas Sandholm (2008). “New Complexity Results about Nash Equilibria”. In: *Games and Economic Behavior* 63.2. Early version in IJCAI-03, pp. 621–641.
- Coulom, Rémi (2005). *BayesElo*. Software. URL: <https://www.remi-coulom.fr/Bayesian-Elo/#theory>.
- Dantzig, George B and Mukund N Thapa (2006). *Linear programming 2: theory and extensions*. Springer Science & Business Media.
- Daskalakis, Constantinos, Alan Deckelbaum, and Anthony Kim (2011). “Near-optimal no-regret algorithms for zero-sum games”. In: *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*.
- Daskalakis, Constantinos, Maxwell Fishelson, and Noah Golowich (2021). “Near-Optimal No-Regret Learning in General Games”. In: *CoRR* abs/2108.06924.
- Daskalakis, Constantinos, Paul W. Goldberg, and Christos H. Papadimitriou (2009). “The Complexity of Computing a Nash Equilibrium”. In: *Commun. ACM* 52.2, pp. 89–97.
- Daskalakis, Constantinos and Noah Golowich (2022). “Fast rates for nonparametric online learning: from realizability to learning in games”. In: *Proceedings of the Annual Symposium on Theory of Computing (STOC)*.
- Daskalakis, Constantinos and Ioannis Panageas (2019). “Last-Iterate Convergence: Zero-Sum Games and Constrained Min-Max Optimization”. In: *10th Innovations in Theoretical Computer Science Conference, ITCS 2019*.
- Dudik, Miroslav and Geoffrey J Gordon (2009). “A sampling-based approach to computing equilibria in succinct extensive-form games”. In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.
- Farina, Gabriele, Tommaso Bianchi, and Tuomas Sandholm (2020). “Coarse Correlation in Extensive-Form Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Farina, Gabriele, Andrea Celli, Nicola Gatti, and Tuomas Sandholm (2018). “Ex Ante Coordination and Collusion in Zero-Sum Multi-Player Extensive-Form Games”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele, Andrea Celli, Nicola Gatti, and Tuomas Sandholm (2021). “Connecting Optimal Ex-Ante Collusion in Teams to Extensive-Form Correlation: Faster Algorithms and Positive Complexity Results”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.

- Farina, Gabriele, Andrea Celli, Alberto Marchesi, and Nicola Gatti (2022). "Simple Uncoupled No-regret Learning Dynamics for Extensive-form Correlated Equilibrium". In: *Journal of the ACM* 69.6. URL: <https://dl.acm.org/doi/10.1145/3563772>.
- Farina, Gabriele and Nicola Gatti (2017). "Extensive-Form Perfect Equilibrium Computation in Two-Player Games". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Farina, Gabriele, Nicola Gatti, and Tuomas Sandholm (2018). "Practical Exact Algorithm for Trembling-Hand Equilibrium Refinements in Games". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2017). "Regret Minimization in Behaviorally-Constrained Zero-Sum Games". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2019a). "Online Convex Optimization for Sequential Decision Processes and Extensive-Form Games". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2019b). "Optimistic Regret Minimization for Extensive-Form Games via Dilated Distance-Generating Functions". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2019c). "Regret Circuits: Composability of Regret Minimizers". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2021a). "Better Regularization for Sequential Decision Spaces: Fast Convergence Rates for Nash, Correlated, and Team Equilibria". In: *ACM Conference on Economics and Computation*.
- Farina, Gabriele, Christian Kroer, and Tuomas Sandholm (2021b). "Faster Game Solving via Predictive Blackwell Approachability: Connecting Regret Matching and Mirror Descent". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Farina, Gabriele, Chung-Wei Lee, Haipeng Luo, and Christian Kroer (2022). "Kernelized Multiplicative Weights for 0/1-Polyhedral Games: Bridging the Gap Between Learning in Extensive-Form and Normal-Form Games". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Farina, Gabriele, Chun Kai Ling, Fei Fang, and Tuomas Sandholm (2019a). "Correlation in Extensive-Form Games: Saddle-Point Formulation and Benchmarks". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.

- Farina, Gabriele, Chun Kai Ling, Fei Fang, and Tuomas Sandholm (2019b). "Efficient Regret Minimization Algorithm for Extensive-Form Correlated Equilibrium". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele and Tuomas Sandholm (2020). "Polynomial-Time Computation of Optimal Correlated Equilibria in Two-Player Extensive-Form Games with Public Chance Moves and Beyond". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele and Tuomas Sandholm (2021a). "Equilibrium Refinement for the Age of Machines: The One-Sided Quasi-Perfect Equilibrium". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Farina, Gabriele and Tuomas Sandholm (2021b). "Model-Free Online Learning in Unknown Sequential Decision Making Problems and Games". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Farina, Gabriele and Tuomas Sandholm (2022). "Fast Payoff Matrix Sparsification Techniques for Structured Extensive-Form Games". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Farina, Gabriele, Robin Schmucker, and Tuomas Sandholm (2021). "Bandit Linear Optimization for Sequential Decision Making and Extensive-Form Games". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Fiegel, Côme, Pierre Ménard, Tadashi Kozuno, Rémi Munos, Vianney Perchet, and Michal Valko (2023). *Adapting to game trees in zero-sum imperfect information games*. arXiv: [2212.12567](https://arxiv.org/abs/2212.12567).
- Foster, Dean and Rakesh Vohra (1997). "Calibrated Learning and Correlated Equilibrium". In: *Games and Economic Behavior* 21, pp. 40–55.
- Foster, Dylan J., Zhiyuan Li, Thodoris Lykouris, Karthik Sridharan, and Éva Tardos (2016). "Learning in Games: Robustness of Fast Convergence". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Fousse, Laurent, Guillaume Hanrot, Vincent Lefèvre, Patrick Pélissier, and Paul Zimmermann (2007). "MPFR: A Multiple-Precision Binary Floating-Point Library with Correct Rounding". In: *ACM Trans. Math. Softw.* 33.2. doi: [10.1145/1236463.1236468](https://doi.org/10.1145/1236463.1236468).
- Ganzfried, Sam and Tuomas Sandholm (2015). "Endgame Solving in Large Imperfect-Information Games". In: *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

- Gatti, Nicola, Mario Gilli, and Alberto Marchesi (2020). “A characterization of quasi-perfect equilibria”. In: *Games and Economic Behavior* 122, pp. 240–255.
- Gilboa, Itzhak and Eitan Zemel (1989). “Nash and Correlated Equilibria: Some Complexity Considerations”. In: *Games and Economic Behavior* 1, pp. 80–93.
- Gilpin, Andrew (2009). “Algorithms for abstracting and solving imperfect information games”. PhD thesis. Computer Science Department, Carnegie Mellon University.
- Gilpin, Andrew, Javier Peña, and Tuomas Sandholm (2012). “First-Order Algorithm with $\mathcal{O}(\ln(1/\epsilon))$ Convergence for ϵ -Equilibrium in Two-Person Zero-Sum Games”. In: *Mathematical Programming* 133.1–2. Conference version appeared in AAAI-08, pp. 279–298.
- Golowich, Noah, Sarath Pattathil, and Constantinos Daskalakis (2020). “Tight last-iterate convergence rates for no-regret learning in multi-player games”. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*.
- Gorbunov, Eduard, Hugo Berard, Gauthier Gidel, and Nicolas Loizou (2022). “Stochastic Extragradient: General Analysis and Improved Rates”. In: *International Conference on Artificial Intelligence and Statistics*.
- Gorbunov, Eduard, Nicolas Loizou, and Gauthier Gidel (2022). “Extragradient Method: $O(1/K)$ Last-Iterate Convergence for Monotone Variational Inequalities and Connections With Cocoercivity”. In: *International Conference on Artificial Intelligence and Statistics*.
- Gordon, Geoffrey J, Amy Greenwald, and Casey Marks (2008). “No-regret learning in convex games”. In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 360–367.
- Granlund, Torbjörn and the GMP development team (2012). *GNU MP: The GNU Multiple Precision Arithmetic Library*. 5.0.5. <http://gmplib.org/>.
- Grant, Michael, Stephen Boyd, and Yinyu Ye (2006). “Disciplined convex programming”. In: *Global optimization*. Springer, pp. 155–210.
- Gray, Jonathan, Adam Lerer, Anton Bakhtin, and Noam Brown (2020). “Human-Level Performance in No-Press Diplomacy via Equilibrium Search”. In: *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Greenwald, Amy and Amir Jafari (2003). “A General Class of No-Regret Learning Algorithms and Game-Theoretic Equilibria”. In: *Proceedings of the Conference on Learning Theory (COLT)*.
- Hart, Sergiu and Yishay Mansour (2010). “How long to equilibrium? The communication complexity of uncoupled equilibrium procedures”. In: *Games Econ. Behav.* 69.1, pp. 107–126.

- Hart, Sergiu and Andreu Mas-Colell (2000). "A Simple Adaptive Procedure Leading to Correlated Equilibrium". In: *Econometrica* 68, pp. 1127–1150.
- Hart, Sergiu and Andreu Mas-Colell (2003). "Uncoupled dynamics do not lead to Nash equilibrium". In: *American Economic Review* 93, pp. 1830–1836.
- Held, Michael, Philip Wolfe, and Harlan P Crowder (1974). "Validation of subgradient optimization". In: *Mathematical programming* 6.1, pp. 62–88.
- Hirsch, Michael D., Christos H. Papadimitriou, and Stephen A. Vavasis (1989). "Exponential lower bounds for finding Brouwer fix points". In: *J. Complex.* 5.4, pp. 379–416.
- Hoda, Samid, Andrew Gilpin, Javier Peña, and Tuomas Sandholm (2010). "Smoothing Techniques for Computing Nash Equilibria of Sequential Games". In: *Mathematics of Operations Research* 35.2.
- Hsieh, Yu-Guan, Kimon Antonakopoulos, and Panayotis Mertikopoulos (2021). "Adaptive Learning in Continuous Games: Optimal Regret Bounds and Convergence to Nash Equilibrium". In: *Proceedings of the Conference on Learning Theory (COLT)*.
- Hu, Hengyuan, Adam Lerer, Brandon Cui, Luis Pineda, David Wu, Noam Brown, and Jakob Foerster (2021). "Off-Belief Learning". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Hu, Hengyuan, Adam Lerer, Alexander Peysakhovich, and Jakob Foerster (2020). "'Other-Play' for Zero-Shot Coordination". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Hunter, David R. (2004). "MM algorithms for generalized Bradley-Terry models." In: *The annals of statistics* 32, pp. 384–406.
- Isermann, H. (1982). "Linear lexicographic optimization". In: *Operations-Research-Spektrum* 4.4, pp. 223–228.
- Jacob, Athul Paul, David J. Wu, Gabriele Farina, Adam Lerer, Hengyuan Hu, Anton Bakhtin, Jacob Andreas, and Noam Brown (2022). "Modeling Strong and Human-Like Gameplay with KL-Regularized Search". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Jaggi, Martin (2013). "Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Karmarkar, Narendra (1984). "A new polynomial-time algorithm for linear programming". In: *Proceedings of the Annual Symposium on Theory of Computing (STOC)*, pp. 302–311.

- Koller, Daphne, Nimrod Megiddo, and Bernhard von Stengel (1996). "Efficient Computation of Equilibria for Extensive Two-Person Games". In: *Games and Economic Behavior* 14.2.
- Koo, Terry, Amir Globerson, Xavier Carreras Pérez, and Michael Collins (2007). "Structured prediction models via the matrix-tree theorem". In: *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 141–150.
- Kozuno, Tadashi, Pierre Ménard, Rémi Munos, and Michal Valko (2021). "Model-Free Learning for Two-Player Zero-Sum Partially Observable Markov Games with Perfect Recall". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Kreps, David M. and Robert Wilson (1982). "Sequential Equilibria". In: *Econometrica* 50.4, pp. 863–894.
- Kroer, Christian, Gabriele Farina, and Tuomas Sandholm (2017). "Smoothing Method for Approximate Extensive-Form Perfect Equilibrium". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.
- Kroer, Christian, Gabriele Farina, and Tuomas Sandholm (2018a). "Robust Stackelberg Equilibria in Extensive-Form Games and Extension to Limited Lookahead". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Kroer, Christian, Gabriele Farina, and Tuomas Sandholm (2018b). "Solving Large Sequential Games with the Excessive Gap Technique". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Kroer, Christian, Kevin Waugh, Fatma Kılınç-Karzan, and Tuomas Sandholm (2020). "Faster algorithms for extensive-form game solving via improved smoothing functions". In: *Mathematical Programming*.
- Kuhn, H. W. (1950). "A Simplified Two-Person Poker". In: *Contributions to the Theory of Games*. Vol. 1. Annals of Mathematics Studies, 24. Princeton University Press, pp. 97–103.
- Kuhn, H. W. (1953). "Extensive Games and the Problem of Information". In: *Contributions to the Theory of Games*. Vol. 2. Annals of Mathematics Studies, 28. Princeton University Press, pp. 193–216.
- Lanctot, Marc, Kevin Waugh, Martin Zinkevich, and Michael Bowling (2009). "Monte Carlo Sampling for Regret Minimization in Extensive Games". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Lee, Chung-Wei, Christian Kroer, and Haipeng Luo (2021). "Last-iterate convergence in extensive-form games". In: *Advances in Neural Information Processing Systems* 34.

- Lemke, Carlton E. (1970). "Recent results on complementarity problems". In: *Nonlinear programming*, pp. 349–384.
- Lerer, Adam and Alexander Peysakhovich (2019). "Learning Existing Social Conventions via Observationally Augmented Self-Play". In: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society (AIES)*.
- Lin, Tianyi, Zhengyuan Zhou, Panayotis Mertikopoulos, and Michael I. Jordan (2020). "Finite-Time Last-Iterate Convergence for Multi-Agent Learning in Games". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Ling, Chun Kai, Fei Fang, and J. Zico Kolter (2018). "What game are we playing? End-to-end learning in normal and extensive form games". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.
- Lisỳ, Viliam, Marc Lanctot, and Michael Bowling (2015). "Online Monte Carlo counterfactual regret minimization for search in imperfect information games". In: *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 27–36.
- Liu, Deyi, Volkan Cevher, and Quoc Tran-Dinh (2020). "A Newton Frank-Wolfe Method for Constrained Self-Concordant Minimization". In: *CoRR* abs/2002.07003.
- McAleer, Stephen, Gabriele Farina, Marc Lanctot, and Tuomas Sandholm (2023). "ESCHER: Eschewing Importance Sampling in Games by Computing a History Value Function to Estimate Regret". In: *Proceedings of the International Conference on Learning Representations (ICLR)*.
- McKelvey, Richard D. and Thomas R. Palfrey (1995). "Quantal Response Equilibria for Normal Form Games". In: *Games and Economic Behavior* 10.1.
- McMahan, Brendan, Geoffrey Gordon, and Avrim Blum (2003). "Planning in the Presence of Cost Functions Controlled by an Adversary". In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 536–543.
- Megiddo, Nimrod (1991). "On finding primal-and dual-optimal bases". In: *ORSA Journal on Computing* 3.1, pp. 63–65.
- Milionis, Jason, Christos H. Papadimitriou, Georgios Piliouras, and Kelly Spendlove (2022). "Nash, Conley, and Computation: Impossibility and Incompleteness in Game Dynamics". In: *CoRR* abs/2203.14129.
- Miltersen, Peter Bro and Troels Bjerre Sørensen (2006). "Computing Proper Equilibria of Zero-Sum Games". In: *Computers and Games*.

- Miltersen, Peter Bro and Troels Bjerre Sørensen (2010). “Computing a Quasi-Perfect Equilibrium of a Two-Player Game”. In: *Economic Theory* 42.1.
- Moravcik, Matej, Martin Schmid, Karel Ha, Milan Hladik, and Stephen Gaukrodger (2016). “Refining Subgames in Large Imperfect Information Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Moravčík, Matej, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling (2017). “DeepStack: Expert-level artificial intelligence in heads-up no-limit poker”. In: *Science*.
- Morrill, Dustin, Ryan D’Orazio, Marc Lanctot, James R Wright, Michael Bowling, and Amy R Greenwald (2021). “Efficient Deviation Types and Learning for Hindsight Rationality in Extensive-Form Games”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Morrill, Dustin, Ryan D’Orazio, Reza Sarfati, Marc Lanctot, James Wright, Amy Greenwald, and Michael Bowling (2020). “Hindsight and Sequential Rationality of Correlated Play”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Myerson, Roger B. (1978). “Refinements of the Nash equilibrium concept”. In: *International Journal of Game Theory* 15, pp. 133–154.
- Nair, Ashvin, Murtaza Dalal, Abhishek Gupta, and Sergey Levine (2021). “AWAC: Accelerating online reinforcement learning with offline datasets”. In: *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Nair, Ashvin, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel (2018). “Overcoming exploration in reinforcement learning with demonstrations”. In: *IEEE international conference on robotics and automation (ICRA)*.
- Nash, John (1950). “Equilibrium points in N-person games”. In: *Proceedings of the National Academy of Sciences* 36, pp. 48–49.
- Nayyar, Ashutosh, Aditya Mahajan, and Demosthenis Teneketzis (2013). “Decentralized stochastic control with partial history sharing: A common information approach”. In: *IEEE Transactions on Automatic Control* 58.7, pp. 1644–1658.
- Nemirovski, Arkadi (2004). “Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems”. In: *SIAM Journal on Optimization* 15.1.
- Nesterov, Yurii (2005a). “Excessive Gap Technique in Nonsmooth Convex Minimization”. In: *SIAM Journal of Optimization* 16.1.

- Nesterov, Yurii (2005b). "Smooth Minimization of Non-Smooth Functions". In: *Mathematical Programming*. A 103.
- Paige, C.C., George P.H. Styan, and Peter G. Wachter (1975). "Computation of the stationary distribution of a markov chain". In: *Journal of Statistical Computation and Simulation* 4.3, pp. 173–186.
- Paquette, Philip, Yuchen Lu, Seton Steven Bocco, Max Smith, O-G Satya, Jonathan K Kummerfeld, Joelle Pineau, Satinder Singh, and Aaron C Courville (2019). "No-press diplomacy: Modeling multi-agent gameplay". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Perolat, Julien, Remi Munos, Jean-Baptiste Lespiau, Shayegan Omidshafiei, Mark Rowland, Pedro Ortega, Neil Burch, Thomas Anthony, David Balduzzi, Bart De Vylder, et al. (2021). "From Poincaré Recurrence to Convergence in Imperfect Information Games: Finding Equilibrium via Regularization". In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Piliouras, Georgios, Ryann Sim, and Stratis Skoulakis (2021). "Optimal No-Regret Learning in General Games: Bounded Regret with Unbounded Step-Sizes via Clairvoyant MWU". In: *arXiv preprint arXiv:2111.14737*.
- Piliouras, Georgios, Ryann Sim, and Stratis Skoulakis (2022). "Beyond Time-Average Convergence: Near-Optimal Uncoupled Online Learning via Clairvoyant Multiplicative Weights Update". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Rakhlin, Alexander and Karthik Sridharan (2013). "Online Learning with Predictable Sequences". In: *Proceedings of the Conference on Learning Theory (COLT)*.
- Rakhlin, Sasha and Karthik Sridharan (2013). "Optimization, learning, and games with predictable sequences". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Romanovskii, I. (1962). "Reduction of a Game with Complete Memory to a Matrix Game". In: *Soviet Mathematics* 3.
- Ross, Sheldon M (1971). "Goofspiel—the game of pure strategy". In: *Journal of Applied Probability* 8.3, pp. 621–625.
- Roughgarden, Tim and Omri Weinstein (2016). "On the Communication Complexity of Approximate Fixed Points". In: *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 229–238.
- Selten, Reinhard (1975). "Reexamination of the perfectness concept for equilibrium points in extensive games". In: *International Journal of Game Theory*.

- Shalev-Shwartz, Shai (2012). "Online Learning and Online Convex Optimization". In: *Foundations and Trends in Machine Learning* 4.2.
- Shalev-Shwartz, Shai and Yoram Singer (2007). "A primal-dual perspective of online learning algorithms". In: *Machine Learning* 69.2-3, pp. 115–142.
- Siegel, Noah Y, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller (2020). "Keep doing what worked: Behavioral modelling priors for offline reinforcement learning". In: *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Silver, David, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, et al. (2018). "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". In: *Science* 362.6419, pp. 1140–1144.
- Sion, Maurice (1958). "On general minimax theorems." In: *Pacific J. Math.* 8.4, pp. 171–176.
- Sokota, Samuel, Ryan D'Orazio, J. Zico Kolter, Nicolas Loizou, Marc Lanctot, Ioannis Mitliagkas, Noam Brown, and Christian Kroer (2023). "A unified approach to reinforcement learning, quantal response equilibria, and two-player zero-sum games". In: *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Sokota, Samuel, Edward Lockhart, Finbarr Timbers, Elnaz Davoodi, Ryan D'Orazio, Neil Burch, Martin Schmid, Michael Bowling, and Marc Lanctot (2021). "Solving Common-Payoff Games with Approximate Policy Iteration". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Song, Ziang, Song Mei, and Yu Bai (2022). In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Southey, Finnegan, Michael Bowling, Bryce Larson, Carmelo Piccione, Neil Burch, Darse Billings, and Chris Rayner (2005). "Bayes' Bluff: Opponent Modelling in Poker". In: *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.
- Stoltz, Gilles and Gábor Lugosi (2007). "Learning correlated equilibria in games with compact sets of strategies". In: *Games and Economic Behavior* 59.1, pp. 187–208.
- Strouse, DJ, Kevin McKee, Matt Botvinick, Edward Hughes, and Richard Everett (2021). "Collaborating with humans without human data". In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*.
- Syrgekianis, Vasilis, Alekh Agarwal, Haipeng Luo, and Robert E Schapire (2015). "Fast convergence of regularized learning in games". In: *Advances in Neural Information Processing Systems*.

- Takimoto, Eiji and Manfred K Warmuth (2003). "Path kernels and multiplicative updates". In: *Journal of Machine Learning Research* 4, pp. 773–818.
- Tammelin, Oskari (2014). *Solving large imperfect information games using CFR+*. arXiv: [1407.5042](https://arxiv.org/abs/1407.5042).
- Tammelin, Oskari, Neil Burch, Michael Johanson, and Michael Bowling (2015). "Solving Heads-up Limit Texas Hold'em". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.
- Tran-Dinh, Quoc, Anastasios Kyrillidis, and Volkan Cevher (2015). "Composite self-concordant minimization". In: *Journal of Machine Learning Research* 16, pp. 371–416.
- van Damme, Eric (1984). "A relation between perfect equilibria in extensive form games and proper equilibria in normal form games". In: *International Journal of Game Theory*.
- Vinyals, Oriol, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. (2019). "Grandmaster level in StarCraft II using multi-agent reinforcement learning". In: *Nature* 575.7782.
- von Stengel, Bernhard (1996). "Efficient Computation of Behavior Strategies". In: *Games and Economic Behavior* 14.2, pp. 220–246.
- von Stengel, Bernhard (2002). "Computing Equilibria for Two-Person Games". In: *Handbook of game theory*. Ed. by Robert Aumann and Sergiu Hart. Vol. 3.
- Von Stengel, Bernhard and Françoise Forges (2008). "Extensive-form correlated equilibrium: Definition and computational complexity". In: *Mathematics of Operations Research* 33.4, pp. 1002–1022.
- Von Stengel, Bernhard and Daphne Koller (1997). "Team-maxmin equilibria". In: *Games and Economic Behavior* 21.1-2, pp. 309–321.
- Warmuth, Manfred K and Dima Kuzmin (2008). "Randomized online PCA algorithms with regret bounds that are logarithmic in the dimension". In: *Journal of Machine Learning Research* 9.10, pp. 2287–2320.
- Wei, Chen-Yu, Chung-Wei Lee, Mengxiao Zhang, and Haipeng Luo (2021). "Linear Last-iterate Convergence in Constrained Saddle-point Optimization". In: *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Zhang, Brian Hu, Gabriele Farina, Ioannis Anagnostides, Federico Cacciamani, Stephen McAleer, Andreas Haup, Andrea Celli, Nicola Gatti, Vincent Conitzer, and Tuomas Sandholm (2023). *Learning and Steering toward Optimal Equilibria and Mechanisms*.

- Zhang, Brian Hu, Gabriele Farina, Andrea Celli, and Tuomas Sandholm (2022). “Optimal Correlated Equilibria in General-Sum Extensive-Form Games: Fixed-Parameter Algorithms, Hardness, and Two-Sided Column-Generation”. In: *Proceedings of the ACM Conference on Economics and Computation (EC)*.
- Zhang, Brian Hu, Gabriele Farina, and Tuomas Sandholm (2023). “Team Belief DAG Form: A Concise Representation for Team-Correlated Game-Theoretic Decision Making”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Zhang, Brian Hu and Tuomas Sandholm (2020). “Sparsified Linear Programming for Zero-Sum Equilibrium Finding”. In: *Proceedings of the International Conference on Machine Learning (ICML)*.
- Zhang, Brian Hu and Tuomas Sandholm (2022a). “Team Correlated Equilibria in Zero-Sum Extensive-Form Games via Tree Decompositions”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Zhang, Brian Hu and Tuomas Sandholm (2022b). “Team Correlated Equilibria in Zero-Sum Extensive-Form Games via Tree Decompositions”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Zhang, Hugh (2022). *A Simple Adaptive Procedure Converging to Forgiving Correlated Equilibria*. arXiv: [2207.06548](https://arxiv.org/abs/2207.06548).
- Zhang, Youzhi, Bo An, and Jakub Černý (2021). “Computing Ex Ante Coordinated Team-Maxmin Equilibria in Zero-Sum Multiplayer Extensive-Form Games”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.
- Zinkevich, Martin, Michael Bowling, Michael Johanson, and Carmelo Piccione (2007). “Regret Minimization in Games with Incomplete Information”. In: *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.

Appendix A

Description of benchmark games used in experimental evaluations

In this appendix, we provide a detailed description of the game instances used throughout the different empirical evaluations in the dissertation. All games are identified with a mnemonic that identifies the game and parameters. A table summarizing several key dimensions of the games is available in [Appendix A.2](#).

A.1 Description of game instances

A.1.1 Battleship (B)

The game is a general-sum version of the classic game Battleship, where two players take turns placing ships of varying sizes and values on two separate grids of size $h \times w$, and then take turns firing at their opponent. Ships which have been hit at all their tiles are considered destroyed. The game ends when one player loses all their ships, or after each player has fired r shots. Each player's payoff is determined by the sum of the value of the opponent's destroyed ships minus $\gamma \geq 1$ times the number of their own lost ships. We denote by Bphwr an instance with p players on a grid of size $h \times w$, one unit-size ship for each player, and r rounds.

The Battleship game is known to possess a rich set of correlated solution concepts and were introduced by Farina, Ling, Fang, and Sandholm (2019a) as a benchmark for extensive-form correlated equilibrium.

A.1.2 Liar’s dice (D)

This is a standard benchmark game in the game-solving literature (see, *e.g.*, Lisý, Lanctot, and Bowling, 2015). At the start of the game, each of the n players rolls a fair k -sided die privately. Then, the players take turns making claims about the outcome of their roll. The first player starts by stating any number from 1 to k and the minimum number of dice they believe are showing that value among all players. On their turn, each player has the option to make a higher claim or challenge the previous claim by calling the previous player a “liar”. A claim is higher if the number rolled is higher or the number of dice showing that number is higher. If a player challenges the previous claim and the claim is found to be false, the challenger is rewarded +1 and the last bidder receives a penalty of -1. If the claim is true, the last bidder is rewarded +1, and the challenger receives -1. Game instances are encoded as Dnk .

A.1.3 Goofspiel (G)

This is a standard trick-taking game introduced by Ross (1971). It is an n -player card game, employing $n + 1$ identical decks of k cards each whose values range from 1 to k . At the beginning of the game, each player gets dealt a full deck as their hand, and the third deck (the “prize” deck) is shuffled and put face down on the board. In each turn, the topmost card from the prize deck is revealed. Then, each player privately picks a card from their hand. This card acts as a bid to win the card that was just revealed from the prize deck. The selected cards are simultaneously revealed, and the highest one wins the prize card. If the players’ played cards are equal, the prize card is split. The players’ score are computed as the sum of the values of the prize cards they have won. Game instances are encoded as Gnk .

A.1.4 Limited-information Goofspiel (GL)

This is a variant of Goofspiel with limited information. In this variation, in each turn the players do not reveal the cards that they have played. Instead, players show their cards to a neutral umpire, who then decides the winner of the round by determining which card is the highest. In the event of a tie, the umpire directs the players to divide the prize equally among the tied players, similar to the Goofspiel game. Game instances are encoded as $GLnk$.

A.1.5 Kuhn poker (K)

Three-player Kuhn Poker, an extension of the original two-player version proposed by Kuhn (1950), is played with n players and r cards. Each player begins by paying one chip to the pot and receiving a single private card. The first player can check or bet (*i.e.*, putting an additional chip in the pot). Then, the second player can check or bet after a first player’s check, or fold/call the first player’s bet. The third player can either check or bet if no previous bet was made, otherwise they

must fold or call. At the showdown, the player with the highest card who has not folded wins all the chips in the pot. We encode Kuhn poker instances with the mnemonic Knr , where r is encoded in hexadecimal (so, for example, $K2c$ is set up with $r = 12$).

A.1.6 Leduc poker (L)

This is another standard benchmark introduced by Southey, Bowling, Larson, Piccione, Burch, Billings, and Rayner (2005). In n -player Leduc poker the deck consists of s suits with r cards each. Our instances are parametric in the maximum number of bets b , which in limit hold'em is not necessarily tied to the number of players. The maximum number of raise per betting round can be either 1, 2 or 3. At the beginning of the game, players each contribute one chip to the pot. The game proceeds with two rounds of betting. In the first round, each player is dealt a private card, and in the second round, a shared board card is revealed. The minimum raise is set at 2 chips in the first round and 4 chips in the second round. We denote by $Lnbrs$ an instance with n players, b bets per round, r ranks, and s suits.

A.1.7 Pursuit-evasion (P)

This is a security-inspired pursuit-evasion game played on the graph shown in Figure A.1. It is a zero-sum variant of the one used by Kroer, Farina, and Sandholm (2018a), and a similar search game has been considered by Bošanský, Kiekintveld, Lisý, and Pěchouček (2014) and Branislav Bošanský and Čermák (2015).

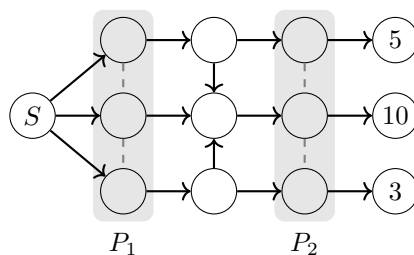


Figure A.1: The graph on which the search game is played.

In each turn, the attacker and the defender act simultaneously. The defender controls two patrols, one per each respective patrol areas labeled P_1 and P_2 . Each patrol can move by one step along the gray dashed lines, or stay in place. The attacker starts from the leftmost node (labeled S) and at each turn can move to any node adjacent to its current position by following the black directed edges. The attacker can also choose to wait in place for a time step in order to hide all their traces. If a patrol visits a node that was previously visited by the attacker, and the attacker did not wait to clean up their traces, they will see that the attacker was there. The goal of the attacker is to reach any of the rightmost nodes, whose corresponding payoffs are 5, 10, or 3,

respectively, as indicated in [Figure A.1](#). If at any time the attacker and any patrol meet at the same node, the attacker loses the game, which leads to a payoff of -1 for the attacker and of 1 for the defender. The game times out after m simultaneous moves, in which case both defenders receive payoffs 0 . We encode instances of this game via the mnemonic `P2m`.

A.1.8 River Endgame (REL)

The river endgame is structured and parameterized as follows. The game is parameterized by the conditional distribution over hands for each player, current pot size, board state (5 cards dealt to the board), and a betting abstraction. First, Chance deals out hands to the two players according to the conditional hand distribution. Then, Libratus has the choice of folding, checking, or betting by a number of multipliers of the pot size: $0.25x$, $0.5x$, $1x$, $2x$, $4x$, $8x$, and all-in. If Libratus checks and the other player bets then Libratus has the choice of folding, calling (i.e. matching the bet and ending the betting), or raising by pot multipliers $0.4x$, $0.7x$, $1.1x$, $2x$, and all-in. If Libratus bets and the other player raises Libratus can fold, call, or raise by $0.4x$, $0.7x$, $2x$, and all-in. Finally, when facing subsequent raises Libratus can fold, call, or raise by $0.7x$ and all-in. When faced with an initial check, the opponent can fold, check, or raise by $0.5x$, $0.75x$, $1x$, and all-in. When faced with an initial bet the opponent can fold, call, or raise by $0.7x$, $1.1x$, and all-in. When faced with subsequent raises the opponent can fold, call, or raise by $0.7x$ and all-in. The game ends whenever a player folds (the other player wins all money in the pot), calls (a showdown occurs), or both players check as their first action of the game (a showdown occurs). In a showdown the player with the better hands wins the pot. The pot is split in case of a tie. We denote with `REL2i` the instances of the river endgame, where i is a unique identifier of the endgame configuration.

A.1.9 Sheriff (S)

This game is a simplified version of the *Sheriff of Nottingham* board game, which models the interaction between a *Smuggler*—who is trying to smuggle illegal items in their cargo—and the *Sheriff*—whose goal is stopping the Smuggler. First, the Smuggler has to decide the number $n \in \{0, \dots, N\}$ of illegal items to load on the cargo. Then, the Sheriff decides whether to inspect the cargo. If they choose to inspect, and find illegal goods, the Smuggler has to pay $p \cdot n$ to the Sheriff. Otherwise, the Sheriff has to compensate the Smuggler with a reward of s . If the Sheriff decides not to inspect the cargo, the Sheriff’s utility is 0 , and the Smuggler’s utility is $v \cdot n$. After the Smuggler has loaded the cargo, and before the Sheriff decides whether to inspect, the Smuggler can try to bribe the Sheriff to avoid the inspection. In particular, they engage in r rounds of bargaining and, for each round i , the Smuggler proposes a bribe $b_i \in \{0, \dots, B\}$, and the Sheriff accepts or declines it. Only the proposal and response from the final round r are executed. If the Sheriff accepts a bribe b_r then they get b_r , while the Smuggler’s utility is $vn - b_r$. Further details on the game can be found in Farina, Ling, Fang, and Sandholm (2019a). An

instance $SpNB_r$ has p players, N illegal items (encoded in hexadecimal notation), a maximum bribe of B , and r rounds of bargaining. The other parameters are $v = 5$, $p = 1$, $s = 1$ and they are fixed across all instances.

A.1.10 Double-dummy bridge endgame (T, TP)

The double-dummy bridge endgame is a benchmark introduced by B. H. Zhang, Farina, Celli, and Sandholm (2022) which simulates a bridge endgame scenario. The game uses a fixed deck of playing cards that includes three ranks (2, 3, 4) of each of four suits (spades, hearts, diamonds, clubs). Spades are designated as the trump suit. There are four players involved: two defenders sitting across from each other, the dummy, and the declarer. The dummy's actions will be controlled by the declarer, so there are only three players actively participating. However, for clarity, we will refer to all four players throughout this section.

The entire deck of cards is randomly dealt to the four players. We study the version of the game that has perfect information, meaning that all players' cards are revealed to everyone, creating a game in which all information is public (*i.e.*, a *double-dummy game*). The game is played in rounds called *tricks*. The player to the left of the declarer starts the first trick by playing a card. The suit of this card is known as the *lead suit*. Going in clockwise order, the other three players play a card from their hand. Players must play a card of the lead suit if they have one, otherwise, they can play any card. If a spade is played, the player with the highest spade wins the trick. Otherwise, the highest card of the lead suit wins the trick. The winner of each trick then leads the next one. At the end of the game, each player earns as many points as the number of tricks they won. In this adversarial team game, the two defenders are teammates and play against the declarer, who controls the dummy.

The specific instance that we use (*i.e.*, TP3) has 3 ranks and perfect information. The dummy's hand is fixed as $2\spadesuit 2\heartsuit 3\heartsuit$.

In the *limited deals* variant T3[L], L deals are randomly selected at the beginning of the game, and it is common knowledge that the true deal is among them. This limits the size of the game tree, as well as the parameters on which the complexity of our algorithms depend. Note that $L = 9!/(3!)^3 = 1680$ is the full game.

A.1.11 Ridesharing game (RS)

This benchmark was first introduced by B. H. Zhang, Farina, Celli, and Sandholm (2022), and it models the interaction between two drivers competing to serve requests on a road network. The network is defined as an undirected graph $G^u = (V^u, E^u)$, where each vertex $v \in V^u$ corresponds to a ride request to be served. Each request has a reward in $\mathbb{R}_{\geq 0}$, and each edge in the network has some cost. The first driver who arrives at node $v \in V^u$ serves the corresponding ride, and receives the corresponding reward. Once a node has been served, it stays clean until the end of

the game. The game terminates when all nodes have been cleared, or when a timeout is met (*i.e.*, there's a fixed time horizon T). If the two drivers arrive simultaneously on the same vertex they both get reward 0. The final utility of each driver is computed as the sum of the rewards obtained from the beginning until the end of the game. The initial position of the two drivers is randomly selected at the beginning of the game. Finally, the two drivers can observe each other's position only when they are simultaneously on the same node, or they are in adjacent nodes.

Ridesharing games are particularly well-suited to study the computation of optimal equilibria because they are *not* triangle-free (Farina and Sandholm, 2020).

Setup We denote by $\text{RS}_{p iT}$ a ridesharing instance with p drivers, network configuration i , and horizon T . Parameter $i \in \{1, 2\}$ specifies the graph configuration. We consider the two network configurations of B. H. Zhang, Farina, Celli, and Sandholm (2022), their structure is reported in Figure A.2. All edges are given unitary cost. We consider a total of four instances: RS212 , RS222 , RS213 , RS223 .

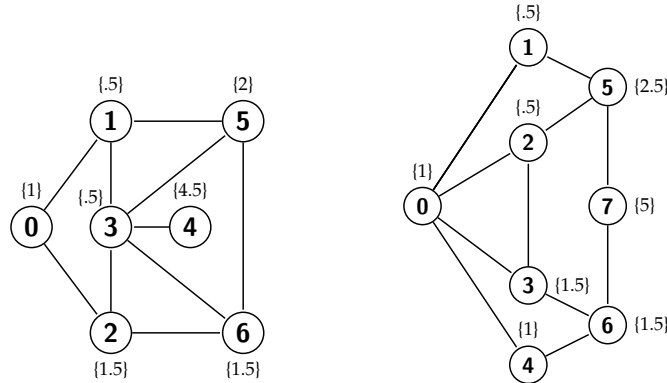


Figure A.2: *Left*: Graph configuration 1, used for RS212 RS213 . *Right*: Graph configuration 2, used for RS222 RS223 . In both cases the position of the two drivers is randomly chosen at the beginning of the game, edge costs are unitary, and the reward for each node is indicated between curly brackets.

A.1.12 Small matrix (SM)

This is a small 2×2 matrix game introduced by Farina, Kroer, and Sandholm (2019b) as a hard instance for the CFR^+ algorithm. Given a mixed strategy $\mathbf{x} = (x_1, x_2) \in \Delta^2$ for Player 1 and a mixed strategy $\mathbf{y} = (y_1, y_2) \in \Delta^2$ for Player 2, the payoff function for player 1 is defined as

$$u(\mathbf{x}, \mathbf{y}) := 5x_1y_1 - x_1y_2 + x_2y_2.$$

We denote the small matrix with the mnemonic SM2 .

A.2 Game dimensions

Game	Players	Decision nodes	Sequences	Terminal nodes	Constant sum?
B2222	2	162	506	1072	✗
B2322	2	938	4730	19 116	✗
B2323	2	15 338	62 330	191 916	✗
B2324	2	144 938	451 130	969 516	✗
D32	3	128	255	504	✓
D33	3	1536	3069	13 797	✓
D42	4	12 288	24 574	331 695	✓
G33	3	4383	4890	1296	✓
G24	2	34 952	42 658	13 824	✓
G25	2	4 369 010	5 332 052	1 728 000	✓
GL24	2	17 432	21 298	13 824	✓
GL25	2	1 175 330	1 428 452	1 728 000	✓
GL33	3	2511	2802	1296	✓
GL44	4	13 236	14 008	7776	✓
GL55	5	65 385	67 310	46 656	✓
K23	2	12	26	30	✓
K2c	2	5148	12 014	98 956	✓
K33	3	36	75	78	✓
K34	3	48	99	312	✓
K35	3	60	123	780	✓
K36	3	72	147	1560	✓
K3c	3	144	291	17 160	✓
K38	3	96	195	4368	✓
K45	4	160	324	3960	✓

Table A.1: Dimensions of game instances used in this dissertation.

Game	Players	Decision nodes	Sequences	Terminal nodes	Constant sum?
L2232	2	288	674	1116	✓
L2252	2	780	1822	5500	✓
L2282	2	1968	4594	22 936	✓
L2292	2	2484	5798	32 724	✓
L22d2	2	5148	12 014	98 956	✓
L3132	3	684	1371	4500	✓
L3133	3	684	1371	6477	✓
L3151	3	1500	3003	10 020	✓
L3223	3	1890	4329	8762	✓
L3523	3	148 752	369 459	775 148	✓
P24	2	382	2081	15 898	✓
P25	2	2078	11 899	61 084	✓
P26	2	11 888	69 029	118 514	✓
REL26	2	53 010	142 223	71 270 327	✓
REL27	2	42 098	123 580	75 928 256	✓
REL28	2	67 431	184 047	111 580 420	✓
REL22	2	99 144	293 546	185 831 684	✓
REL25	2	242 190	722 432	494 214 830	✓
S2122	2	99	286	396	✗
S2123	2	603	1690	2376	✗
S2133	2	1096	3809	5632	✗
S2254	2	50 896	260 153	435 456	✗
S2264	2	82 741	475 778	806 736	✗
SM2	2	2	6	4	✓
T3[50]	3	14 448	24 304	10 300	✓
TP3	3	531 728	909 059	379 008	✓
RS212	2	68	214	400	✗
RS213	2	704	2370	4356	✗
RS222	2	76	218	484	✗
RS223	2	672	1982	4096	✗

Table A.2: (Continued) Dimensions of game instances used in this dissertation.

Appendix B

Summary of notation

B.1 Tree-form decision processes

Symbol	Description
\mathcal{J}	Set of decision nodes
\mathcal{A}_j	Set of legal actions at decision node $j \in \mathcal{J}$
\mathcal{K}	Set of observation nodes
\mathcal{S}_k	Set of possible signals at observation node $k \in \mathcal{K}$
ρ	Transition function: <ul style="list-style-type: none"> • given $j \in \mathcal{J}$ and $a \in \mathcal{A}_j$, $\rho(j, a)$ returns the next point $v \in \mathcal{J} \cup \mathcal{K}$ in the decision tree that is reached after selecting legal action a in j, or \perp if the decision process ends; • given $k \in \mathcal{K}$ and $s \in \mathcal{S}_k$, $\rho(k, s)$ returns the next point $v \in \mathcal{J} \cup \mathcal{K}$ in the decision tree that is reached after observing signal s in k, or \perp if the decision process ends
Σ^*	Set of non-empty sequences, defined as $\Sigma^* := \{(j, a) : j \in \mathcal{J}, a \in \mathcal{A}_j\}$
Σ	Set of sequences, defined as $\Sigma := \Sigma^* \cup \{\emptyset\}$ where the special element \emptyset is called the <i>empty sequence</i>
p_j	Parent sequence of decision node $j \in \mathcal{J}$, defined as the last sequence (decision node-action pair) on the path from the root of the TFDP to decision node j ; if the player does not act before j , $p_j = \emptyset$
\mathcal{C}_σ	Decision nodes $j \in \mathcal{J}$ with parent sequence $\sigma \in \Sigma$: $\mathcal{C}_\sigma := \{j \in \mathcal{J} : p_j = \sigma\}$
$j' \preceq j$	$j' \in \mathcal{J}$ is on the path from the root to $j \in \mathcal{J}$
$j'a' \preceq ja$	Action a' at j' is on the path from the root to action a at $j \in \mathcal{J}$
$\sigma \succcurlyeq j$	Shorthand for $\sigma = j'a'$ with $j' \succcurlyeq j$
$\Sigma_{\succcurlyeq j}$	Sequences at or below j : $\Sigma_{\succcurlyeq j} := \{\sigma \in \Sigma^* : \sigma \succcurlyeq j\}$

B.2 Correlated strategies

Symbol	Description
$j_1 \rightleftharpoons j_2$	Connected decision nodes: if $j_1 \rightleftharpoons j_2$, it is possible that the trajectory in the decision process goes through both $j_1 \in \mathcal{J}_1$ and $j_2 \in \mathcal{J}_2$
$\sigma_1 \bowtie \sigma_2$	Relevant sequence pair: $\sigma_1 \bowtie \sigma_2$ if at least one between σ_1 and σ_2 is the empty sequence \emptyset , or if $\sigma_1 = (j_1, a_1), \sigma_2 = (j_2, a_2)$ with $j_1 \rightleftharpoons j_2$
$\Sigma_1 \bowtie \Sigma_2$	Set of all relevant sequence pairs $\{(\sigma_1, \sigma_2) \in \Sigma_1 \times \Sigma_2 : \sigma_1 \bowtie \sigma_2\}$
$j_1 \bowtie \sigma_2$	Shorthand for: $(j_1, a_1) \bowtie \sigma_2$ for all $a_1 \in A_{j_1}$
$\sigma_1 \bowtie j_2$	Shorthand for: $\sigma_1 \bowtie (j_2, a_2)$ for all $a_2 \in A_{j_2}$

B.3 Mathematical notation

Vectors and matrices

- Vectors and matrices are marked in bold.
- Given a finite set $S = \{s_1, \dots, s_n\}$, we denote as \mathbb{R}^S (resp., $\mathbb{R}_{\geq 0}^S$) the set of real (resp., nonnegative real) $|S|$ -dimensional vectors whose entries are denoted as $\mathbf{x}[s_1], \dots, \mathbf{x}[s_n]$.
- Similarly, given finite sets S, S' , we denote as $\mathbb{R}^{S \times S'}$ (resp., $\mathbb{R}_{\geq 0}^{S \times S'}$) the set of real (resp., nonnegative real) $S \times S'$ square matrices \mathbf{M} whose entries are denoted as $\mathbf{M}[s_r, s_c]$ ($s_r \in S, s_c \in S'$), where s_r corresponds to the row index and s_c to the column index.

Standard sets

- We denote the set of real numbers as \mathbb{R} , the set of nonnegative real numbers as $\mathbb{R}_{\geq 0}$, and the set $\{1, 2, \dots\}$ of positive integers as \mathbb{N} .
- The set $\{1, \dots, n\}$, where $n \in \mathbb{N}$, is compactly denoted as $\llbracket n \rrbracket$.
- The empty set is denoted as $\{\}$.
- Given a finite set S , we denote by Δ^S the simplex $\Delta^S := \{\mathbf{x} \in \mathbb{R}_{\geq 0}^S : \sum_{s \in S} \mathbf{x}[s] = 1\}$. The symbol Δ^n , with $n \in \mathbb{N}$, is used to mean $\Delta^{\llbracket n \rrbracket}$.
- Given a finite set S , we use the symbol $\mathbb{S}^S \subseteq \mathbb{R}_{\geq 0}^{S \times S}$ to denote the set of stochastic matrices, that is, nonnegative square matrices whose columns all sum up to 1. The symbol \mathbb{S}^n , where $n \in \mathbb{N}$, is used to mean $\mathbb{S}^{\llbracket n \rrbracket}$.

Operations on sets

- Given a set S , we denote its convex hull with the symbol $\text{co } S$. The convex hull of the union of finitely many sets S_1, \dots, S_m is denoted $\text{co}\{S_1, \dots, S_m\}$.
- Disjoint union of set is denoted with the symbol \sqcup .

Functions

- Given two functions $f : X \rightarrow Y$ and $g : Y \rightarrow Z$, we denote by $g \circ f : X \rightarrow Z$ their composition $x \mapsto g(f(x))$.
- Given a set S and a function f , the *image of S via f* is denoted as $f(S) := \{f(s) : s \in S\}$.
- Given a proposition p , we denote with $\mathbb{1}_p$ the indicator function of that proposition:

$$\mathbb{1}_p = \begin{cases} 1 & \text{if } p \text{ is true} \\ 0 & \text{otherwise.} \end{cases}$$

Partial orders

- Given a partially ordered set $(S, <)$ and two elements $s, s' \in S$, we use the standard derived symbols $s \preceq s'$ to mean that $(s = s') \vee (s < s')$, $s \succ s'$ to mean that $s' < s$, and $s \succcurlyeq s'$ to mean that $s' \preceq s$.

Asymptotic notation

- We use the symbols $\mathcal{O}, \Omega, \Theta$ to denote the usual asymptotic notation.
- The symbol \mathcal{O}_T is used to denote that only dependence on the parameter T is highlighted, treating all other parameters as constants.