Combinatorial Optimization Under Uncertainty: Probing and Stopping-Time Algorithms

Sahil Singla

CMU-CS-18-111
August, 2018

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Manuel Blum (Co-Chair)
Anupam Gupta (Co-Chair)
Robert D. Kleinberg (Cornell University)
R. Ravi (Carnegie Mellon University)
Jan Vondrák (Stanford University)

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

This research was sponsored by Google, PNC, Bosch, and the National Science Foundation under grant numbers CCF-1016799, CCF-1536002, and CCF-1617790. 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: Approximation Algorithms, Optimal Stopping Theory, Adaptivity Gaps, Combinatorial Optimization, Optimization Under Uncertainty, Probing Algorithms, Online Algorithms, Algorithmic Game Theory.

Dedicated to my family.

Abstract

Combinatorial optimization captures many natural problems such as matching, load balancing, social welfare, network design, clustering, and submodular optimization. Classically, these problems have been studied in the *full-information* setting, i.e., where the entire input—an objective function and some constraints—is given and the goal is to select a feasible set to maximize/minimize the objective function. In this thesis we focus on combinatorial problems in an uncertain environment where we only have partial knowledge about the input. In particular, we study models where the input is revealed to us element-by-element and we have to make irrevocable decisions. Depending on whether we can control the revelation order of these elements, we separate our models and algorithms into two groups.

(A) **Probing Algorithms**: In these models we have stochastic knowledge about the input, but the uncertainty of an element realizes only after we *probe* it. We can choose the order and the set of elements to probe; however, we do not wish to probe all of them as either probing incurs a price (the *price of information* model) or there are probing constraints (the *constrained stochastic probing* model). Some examples are the *Pandora's box* and the *Best-box* problems.

(B) **Stopping-Time Algorithms**: In these models the input is revealed element-by-element in an order that we cannot control. These models are inspired from work in the field of Optimal Stopping Theory. In particular, we consider combinatorial problems when either we have stochastic knowledge about the input but the revelation order is chosen by an adversary (the *Prophet Inequality* model) or when we have no prior knowledge about the input but the revelation order is chosen uniformly at random (the *Secretary* model).

Acknowledgments

First of all, I am extremely grateful to Anupam Gupta for being my PhD advisor. He taught me everything from how to find a research problem to how to write a research paper. I am thankful to him for introducing me to the field of stochastic combinatorial optimization. This area forms a perfect mix of my research interests: optimization and probability theory.

I am thankful to Manuel Blum, my second PhD advisor, for his constant support. Although we never worked on a problem together, he played an instrumental role during the early years of my PhD. I learnt from him that there are interesting problems everywhere, and if one works hard on a problem, one will find many ideas of great intellectual and practical value. Manuel's humble nature, despite his long list of accomplishments, played a crucial role in making me feel welcome and in finding a home in the theoretical CS community.

During my PhD, I was very fortunate to get the opportunities to visit and learn from Ravishankar Krishnaswamy and Deeparnab Chakrabarty at Microsoft Bangalore, from Mohit Singh at Microsoft Redmond, and from Matt Weinberg at Princeton. I will always remain grateful to them. I am also thankful to the Simons Institute for the Theory of Computing for hosting me at their beautiful Berkeley campus in Fall 2016 and Fall 2017.

I am grateful to my collaborators for bearing with me and having the patience to answer all my silly questions: Aviad Rubinstein, Viswanath Nagarajan, Soheil Ehsani, Thomas Kesselheim, R. Ravi, Ellis Hershkowitz, Domagoj Bradač, Haotian Jiang, Ziv Scully, Janardhan Kulkarni, Euiwoong Lee, Guru Prashanth, Mohammad Hajiaghayi, Goran Žužić, Ariel Procaccia, and Alex Psomas.

I would like to thank my thesis committee members, Robert Kleinberg, Jan Vondrak, and R. Ravi, for their invaluable feedback.

Many of the technical skills I developed were possible only because of the long discussions with the strong and diverse theory students at CMU. I am particularly thankful to Euiwoong Lee, Guru Prashanth, Goran Žužić, David Wajc, David Kurokawa, Ameya Velingkar, Collin White, Jakub Pachocki, Vijay Bhattiprolu, Nika Haghtalab, Nicholas Rech, Ziv Scully, Roie Levin, Ainesh Bakshi, Naama Ben-David, Jason Li, Ritesh Noothigattu, and Pedro Paredes.

I would also use this opportunity to thank my friends who helped me in various stages of life: Anuj Kalia, Vaibhav Raheja, Navdeep Singh, Shayak Sen, Manzil Zaheer, Ankit Garg, Manish Sinha, Ashish Baid, Saurabh Singla, Sahil Rastogi, Bijit Singha, Sukanya Kadam, Gagandeep Singh Bhatia, and Ravneet Singh.

Finally, I could not have achieved anything in life without the unconditional and constant support of my family: my mom, dad, late grandmother, sister Aarushi, and wife Akanksha.

Contents

List of Figures

Part I

Introduction

Chapter 1

Overview

1.1 Motivation

Suppose you want to purchase a house. You have some estimates on the value of every available house in the market, say based on its location, size, and photographs. However, to find the exact value of a site (house) you have to hire a house inspector and pay her a price. Your goal is to simultaneously maximize the value of the best site that you find and to minimize the total inspection price that you pay (House-Purchasing). One simple strategy is to inspect every potential site to find the best one, but this strategy incurs a large inspection price. Another strategy is to only inspect the most promising site, but this site might have a value much smaller than the maximum value site. What is the optimal inspection strategy?

In a different scenario, suppose you own a diamond (item) that you are trying to sell. A sequence of n buyers arrive, each with a different value for your item. On arriving a buyer makes a take-it-or-leave-it bid for your item. Your goal is to maximize the value that you get for your item. The benefit of declining the early bids is that you might get a better bid later, but then you also risk selling your item at a lower price later (Diamond-Selling).

The above two scenarios are examples of situations where we need to solve a combinatorial problem in an uncertain environment. The underlying combinatorial problem in the above scenarios is that of finding the best element (site/buyer). This is a trivial problem in the classical *full-information* setting where we know all element values: just select the maximum value element. However, in an uncertain environment where the element values are revealed one-by-one, it is not obvious how to solve such problems. The situation becomes even more complicated when the underlying combinatorial problem is more involved. For example, finding a max-weight matching in a graph where the weights of the edges are revealed one-by-one. This problem can be useful to model kidney exchanges where you find compatibility of donor-receiver pairs only by performing tests.

Why consider uncertainty? Some of the most common reasons are:

(i) lack of future knowledge,

(ii) imperfect prediction,

(iii) finding exact value is costly, and

(iv) noise in the input.

In this thesis, we focus on combinatorial problems where one starts with some partial knowledge (usually a probability distribution) about the input. The actual input is revealed to us *element-by-element* and we need to make decisions without knowing the future input. Notice that we may not be optimal in hindsight because our decisions are *irrevocable*. E.g., in the House-Purchasing scenario we cannot ask for a reimbursement if we do not like a site and in the Diamond-Selling scenario we cannot go back to a declined bid. The goal is to design optimal/approximation algorithms to maximize our expected value given the partial knowledge about the input.

Before describing our models, we stress on a crucial difference between the two scenarios discussed earlier. While for House-Purchasing one could choose the order in which the element values are revealed, for Diamond-Selling the arrival order of the buyers cannot be controlled. On one hand this simplifies the latter problem as you do not have to worry about choosing the order, on the other hand you now have to make decisions *immediately* as you cannot go back to a declined bid. In this thesis we will see how these distinctions lead to complementary challenges and give two groups of uncertainty models: (a) *probing problems* where you can control the order and (b) *stopping-time problems* where you cannot control the order and have to make immediate decisions.

In §1.2, we define how to model a typical combinatorial optimization problem. In §1.3, we define how to formally model the House-Purchasing and the Diamond-Selling scenarios in the probing and the stopping-time models, respectively.

1.2 How to Model Combinatorial Optimization

Given a *finite* set of ground elements V, some constraints $\mathcal{F} \subseteq 2^V$, and an objective function $f : 2^V \to \mathbb{R}$, a *combinatorial optimization* problem is to select a set $S \subseteq V$ that is *feasible*, i.e., $S \in \mathcal{F}$, while trying to maximize/minimize the objective $f(S)$. Since the number of subsets of V is finite, it is possible to run an algorithm that evaluates f at every feasible subset to find the optimal solution. However, such an approach is not computationally *efficient* as the number of subsets is exponential in $|V|$.

Getting poly($|V|$) running time algorithms for the above combinatorial problem seems impossible because both $\mathcal{F}$ and f might have exponential size descriptions. A standard way of getting around this difficulty is to assume access to the following *oracles*. For any set $S \subseteq V$, an *independence oracle* for $\mathcal{F}$ returns whether S is independent (i.e., $S \in \mathcal{F}$) and a *value oracle* for f tells returns the value $f(S)$. Now our goal is to design *efficient* algorithms that only make poly($|V|$) calls to these oracles. It turns out that for arbitrary $\mathcal{F}$ or f one can still not design efficient algorithms. We next briefly mention some natural families of constraints and objective functions where interesting results are possible, and discuss them in detail in Chapter 2.

The most common class of objective functions are additive.

Definition 1.2.1 (Additive/Linear function)**.** *Given a vector* $\mathbf{c} \in \mathbb{R}^{|V|}$*, we define* $f(S) := \mathbf{c}^\intercal \cdot \mathbf{1}_S$

for any $S \subseteq V$, where $\mathbf{1}_S$ is a vector of dimension $|V|$ that contains 1 for $i \in S$ and 0 otherwise.

In this thesis we also consider several general combinatorial functions that do not always have a polynomial size description, but given access to a value oracle can be optimized for many constraint families. These include submodular, XOS, and subadditive functions (defined in §2.2).

The most common families of constraints are packing and covering constraints.

Definition 1.2.2 (Packing/Downward-closed constraints)**.** *An independence family $\mathcal{F} \subseteq 2^V$ is called* packing *if $A \in \mathcal{F}$ and $B \subseteq A$ implies $B \in \mathcal{F}$.*

Some examples are acyclic subgraphs, matroids, matchings, knapsack, and orienteering.

Definition 1.2.3 (Upward-closed or covering constraints)**.** *An independence family $\mathcal{F} \subseteq 2^V$ is called* covering *if $A \in \mathcal{F}$ and $B \supseteq A$ implies $B \in \mathcal{F}$.*

Some examples are spanning graphs, vertex/set cover, Steiner tree, and facility location.

1.3 How to Model Uncertainty

The uncertainty models considered in this thesis assume some stochastic knowledge about the input. The uncertainty is then revealed to us element-by-element in an order that we can (probing models) or cannot (stopping-time models) control.

1.3.1 Group 1: Probing Models

Consider the House-Purchasing scenario from the introduction where you can can select the order in which you find the values of the sites. The goal in this problem is to both maximize the value of the site and to minimize the total inspection price. Two natural ways to model this problem are: (a) maximize the difference of the value of the site and the price spent on inspections, and (b) given an inspection budget $B > 0$, maximize the value of the site subject to total inspection price being smaller than B. We call these models the *price of information* (PoI) and *constrained stochastic probing* (CoSP), respectively. Below we formally describe how to model the House-Purchasing scenario in each of these models.

Definition 1.3.1 (House-Purchasing in the PoI model)**.** *Given probability distributions of n independent random variables X_i and given their probing prices π_i, the problem is to adaptively probe a subset* Probed $\subseteq [n]$ *to maximize the expected utility:*

$$\mathbb{E}\Big[\max_{i \in \text{Probed}} \{X_i\} - \sum_{i \in \text{Probed}} \pi_i\Big].$$

Note that by *adaptively* we mean that our decision to probe which element next can depend on the values of the already probed elements. This problem formulation turns out to be the same as Weitzman's "Pandora's box" problem [Wei79]. In Chapters 3, 4, and 7, we discuss how to extend this PoI to other combinatorial problems, e.g., max-weight matching and min-cost set cover, and to other related models.

Definition 1.3.2 (House-Purchasing in the CoSP model)**.** *Given probability distributions of n independent random variables X_i, probing prices π_i, and a budget B, the problem is to adaptively probe a subset* Probed $\subseteq [n]$ *s.t.* $\sum_{i \in \text{Probed}} \pi_i \leq B$ *while maximizing the expected value:*

$$\mathbb{E}\Big[\max_{i \in \text{Probed}} \{X_i\}\Big].$$

This problem formulation is the same as the *best-box* problem considered in Chapter 5. In Chapters 5-7 we discuss how to extend this CoSP model to the problem of maximizing an uncertain submodular/subadditive function over a packing constraint.

1.3.2 Group 2: Stopping-Time Models

Consider the Diamond-Selling scenario from the introduction where the potential buyers arrive in an order that you cannot control and bid take-it-or-leave-it values. There are two popular Stopping Theory models that can be used to study this problem: the *secretary* and the *prophet inequality* models. In the secretary model we assume no prior knowledge about the buyer values but the buyers arrive in a uniformly random order [Dyn63]. Meanwhile, in the prophet inequality model we assume stochastic knowledge about the buyer values but the arrival order of the buyers is chosen by an adversary [KS78, KS77]. Below we formally describe the Diamond-Selling secenario in these two models.

Definition 1.3.3 (Diamond-Selling in the Prophet Inequality model)**.** *Given probability distributions of n independent random variables X_i, suppose their outcome values are revealed in an adversarial order. Whenever a value is revealed we have to immediately and irrevocably decide if we want to select this element, while ensuring that we never select more than one element. The goal is to maximize the expected value of the element that we select.*

In Chapters 8 and 9 we extend this prophet inequality model to other combinatorial problems such as maximizing a submodular function over a matroid constraint and maximizing a subadditive function over a packing constraint.

Definition 1.3.4 (Diamond-Selling in the Secretary model)**.** *A sequence of n element values are revealed in a uniformly random order. Whenever a value is revealed we have to immediately and irrevocably decide if we want to select this element, while ensuring that we never select more than one element. The goal is to maximize the expected value of the element that we select.*

In Chapters 10-12 we extend this secretary model to other combinatorial problems such as max-cardinality matching and maximizing a subadditive function over a packing constraint. We also study a related *prophet secretary* model that assumes both stochastic knowledge on buyer values and the buyers arrive in a random order, and tries to get the best of both the prophet inequality and secretary models.

1.4 Thesis Statement

In this thesis we seek to address the question whether combinatorial optimization is possible when the input is not entirely known and we have to make irrevocable decisions to obtain

reduce input uncertainty. For example, in the House-Purchasing scenario these irrevocable decisions meant paying a price to find a site value and in the Diamond-Selling scenario an irrevocable decision meant permanently deciding whether to sell the item to the current buyer. The challenge is to balance between *exploration* vs. *exploitation*: at each moment we can either choose to exploit the currently most promising option or to further explore to reduce input uncertainty.

This thesis gives evidence to support the following statement:

> *Efficient combinatorial optimization is possible even when the underlying input is uncertain and we (adaptively) make irrevocable decisions to resolve this uncertainty.*

Besides their immediate applications, I believe the results and techniques developed in this thesis will motivate other researchers to develop algorithms with provable guarantees for other optimization under uncertainty problems where we currently rely on heuristics.

1.5 Thesis Contributions

In §1.5.1 and §1.5.2, we give informal descriptions of our results and techniques, often simplified to the House-Purchasing and the Diamond-Selling scenarios mentioned in the introduction. The hope is to give readers an idea of the challenges that different variants of these two "simple" scenarios already present. The results in the chapters are more generic, where we consider combinatorial objective functions and general packing/covering constraints. In §1.5.3 we explain how the contributions in this thesis can be viewed as designing algorithms for some Markov decision processes (MDPs). Since the MDPs obtained for our problems are exponentially sized, existing algorithms do not suffice: they are either not computationally efficient or have no provable performance guarantees. Finally, in §1.5.4 we present a roadmap of the chapters in this thesis.

1.5.1 Results and Techniques for Probing Algorithms

The House-Purchasing scenario is the running example for probing problems in Part II.

Price of Information (PoI) Consider House-Purchasing in the PoI model where the goal is to maximize the value of the best site *minus* the total probing (inspection) prices (Definition 1.3.1). Although this can be easily solved using Weitzman's algorithm for Pandora's box [Wei79], it leaves several important questions:

(a) What if we want to purchase multiple houses where our value function is combinatorial (e.g., sum of values of best k houses, or an independent set in a matroid, or a matching)?

(b) What if we need to perform multiple probes at a site before finding its value, where each inspection incurs a price and improves our estimate?

The above Question (a) is our focus in Chapter 3. Our main result is to give a general reduction that converts any "greedy" (Frugal) algorithm for a combinatorial problem into a probing

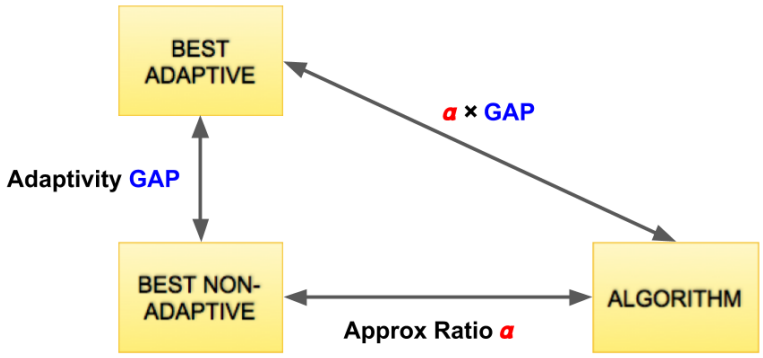

Figure 1.1: An α-approximation to the best non-adaptive solution implies an $(\alpha \cdot GAP)$-approximation to the best adaptive algorithm, where GAP is the adaptivity gap.

strategy with the same performance guarantees. Applications of this powerful reduction include optimal probing algorithms for sum of values of best k houses and a 2-approximation for max-weight matching. We also study several covering problem where our goal is to minimize *disutility*, which is the cost of our solution *plus* the total probing prices. Examples of such problems includes minimum spanning trees and minimum set-cover.

In Chapter 4 we focus on Question (b) where need to make several probes (inspections) before finding the value of a site. A natural way to model this multistage probing is to use a separate Markov chain for each site, where each probe at a site incurs a price and results in a random transition in its chain (the Markovian PoI). The probing algorithm selects which next Markov chain to probe and perform a random transition. Naïvely, it is not clear how this algorithm should compare different sites (chains): what if a site has currently no/low value but with 1/2 probability becomes a high value site after the next probe? The main idea in this chapter is to define a time varying "proxy" value (*grade*) for every site depending on its current state in its Markov chain. We use this grade to give a general reduction that converts any greedy algorithm for a combinatorial problem into a multistage probing strategy with the same performance guarantees.

Constrained Stochastic Probing (CoSP) Now consider a model for House-Purchasing where there is a *strict budget* constraint on the probing price and the goal is to maximize the value of the best site we find (Definition 1.3.2). Notice, the price does not appear in objective of this best-box problem. One might wonder if we can "Lagrangify" the budget constraint to move it into the objective, thereby reducing CoSP to PoI. Unfortunately, such a simple approach does not work; e.g., unlike Pandora's box, optimal probing strategies are not known for best-box.

Since in general the optimal adaptive strategy is exponentially sized and computationally inefficient to find, is there a single *non-adaptive* solution that gets almost the same value? Such a non-adaptive strategy decides all its probing sites independent of the outcomes. A small *adaptivity gap* result will therefore reduce the search space considerably at only a small loss in performance (see Figure 1.1). In Chapter 5 we show that such non-adaptive solutions exist. This raises the question if the adaptivity gap continues to remain small for the following

generalizations:

(c) What if the budget constraint is replaced by a more general constraint, e.g., an arbitrary packing constraint?

(d) What if the objective is a combinatorial function, e.g., submodular or subadditive?

In Chapters 5 and 6 we design new techniques to bound adaptivity gaps for combinatorial CoSP problems over arbitrary packing constraints. The crucial insight on which they rely is to argue that a randomized non-adaptive algorithm that probes every feasible subset of sites with exactly the same probability as the optimal adaptive strategy has a "high" value.

Finally, in Chapter 7 we show why our techniques for PoI and CoSP are amenable to several changes in the models. In particular, we discuss three such modeling changes:

(e) What happens when we combine PoI and CoSP models, i.e., the probing price appears both in the objective and as a constraint?

(f) What if whenever we probe a site, we need to immediately and irrevocably decide if this site is included in our final solution (*commitment* constraint)?

(g) What if the input probability distributions to our probing model is noisy (*robustness*)?

We obtain optimal/approximation algorithms for all the settings described above.

1.5.2 Results and Techniques for Stopping-Time Algorithms

The Diamond-Selling scenario is the running example for stopping-time problems in Part III.

Prophet Inequality Consider Diamond-Selling in a model where we are given probability distributions on *adversarially* arriving buyer values. On a buyer arrival, we immediately and irrevocably decide (*commitment* constraint) whether to sell the item (diamond). Although we know a tight 2-approximation algorithm for this problem [KS78], and its generalization to matroids [KW12], previous approaches are ad-hoc and raise the following questions:

(a) Is there a generic technique to handle commitment constraints for packing problems?

(b) How to maximize a combinatorial function, e.g., submodular or subadditive, over a matroid/packing constraint in the prophet inequality model?

In Chapter 8 we present *online contention resolution scheme* (OCRS), introduced in [FSZ16], a generic technique to handle commitment constraint over matroids, knapsacks, and their intersections. We prove that not only is this technique generic, it also gives *optimal* approximation factors for matroids. In Chapter 9 we again use this technique to obtain an $O(1)$-approximation prophet inequality to maximize a submodular function over a matroid. Since it is known that for general packing constraints one cannot obtain efficient algorithms, in Chapter 9 we also design inefficient but *information theoretically* possible $O(\text{poly}\log(n))$-subadditive prophet inequalities over arbitrary packing constraints.

Secretary and Prophet Secretary Unlike prophet inequalities, in many practical situations it is conceivable that the buyer arrival order for Diamond-Selling is not decided by an ad-

versary. This motivates us to consider the secretary model where the arrival order is chosen uniformly at random. While in the classical secretary problem buyer value distributions are unknown [Dyn63], in Chapter 10 we also consider the prophet secretary model where both the distributions are known and the arrival order is random.

(c) Can we improve the prophet inequality approximation factors under this assumption of uniformly random arrival order?

(d) How to design algorithms for combinatorial problems such as matching and combinatorial auctions in these models?

In Chapters 11 and 12 we answer both the above questions and give new algorithms with improved approximation factors for matroids, matchings, and combinatorial auctions. Our algorithms crucially exploit properties of uniformly random arrival; e.g., this allows us to convert our discrete problem into a continuous time problem where we can imagine each buyer arrives at a uniformly random time in $[0,1]$.

1.5.3 Connections to Markov Decision Processes

The contributions in this thesis can be also viewed as designing efficient algorithms with provable guarantees for some Markov decision processes (MDPs). This is because the problems that we consider can be captured using an MDP, albeit with a caveat that the size of the MDPs becomes *exponential*, thereby preventing use of any generic algorithms. We first recollect the definition of an MDP.

Definition 1.5.1 (Markov decision process)**.** *An MDP consist of a finite state space $\mathcal{S}$ and a finite set of actions $\mathcal{A}$. In each time step the algorithm, which starts in $r \in \mathcal{S}$ and is currently in state $s \in \mathcal{S}$, chooses an action $a \in \mathcal{A}$. The chosen action a determines the reward $r(a,s) : \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ and the probability of transitioning from state s to s', i.e., $P(a,s,s') : \mathcal{A} \times \mathcal{S} \times \mathcal{S} \to [0,1]$. The problem is to select an action for every state to maximize the expected total reward.*

It is known how to optimally solve an MDP in a time polynomial in $|\mathcal{S}|$ and $|\mathcal{A}|$ using dynamic programming [Ber95, SB98].

One can easily show that the stochastic models considered in this thesis can be captured using an MDP. E.g., we can model the House-Purchasing scenario as an MDP: the state space $\mathcal{S}$ captures the currently known/unknown values of the n sites (for Bernoulli variables, each site i has 3 possible values $\{0, v_i, \text{unknown}\}$, which implies $|\mathcal{S}| = 3^n$) and the actions space $\mathcal{A}$ consists of which site we probe next or whether we decide to purchase one of the probed houses ($|\mathcal{A}| = O(n)$). The stochastic value of the probed site determines the next stochastic state and we obtain a reward only by playing an action that purchases a house.

Although powerful, modeling the House-Purchasing scenario as an MDP is not very useful. This is because such a model consists of an *exponential* number of states and generic optimal MDP algorithm runs in time poly$(|\mathcal{S}|)$ (since the input size is already $\Omega(|\mathcal{S}|)$). On way to overcome this hurdle is to use some heuristics, e.g., Q-learning, that is the common approach in the reinforcement learning literature [Ber95, SB98, Pow07]. Unfortunately, such an approach does not immediately give us any provable performance guarantees on our algorithms. In this

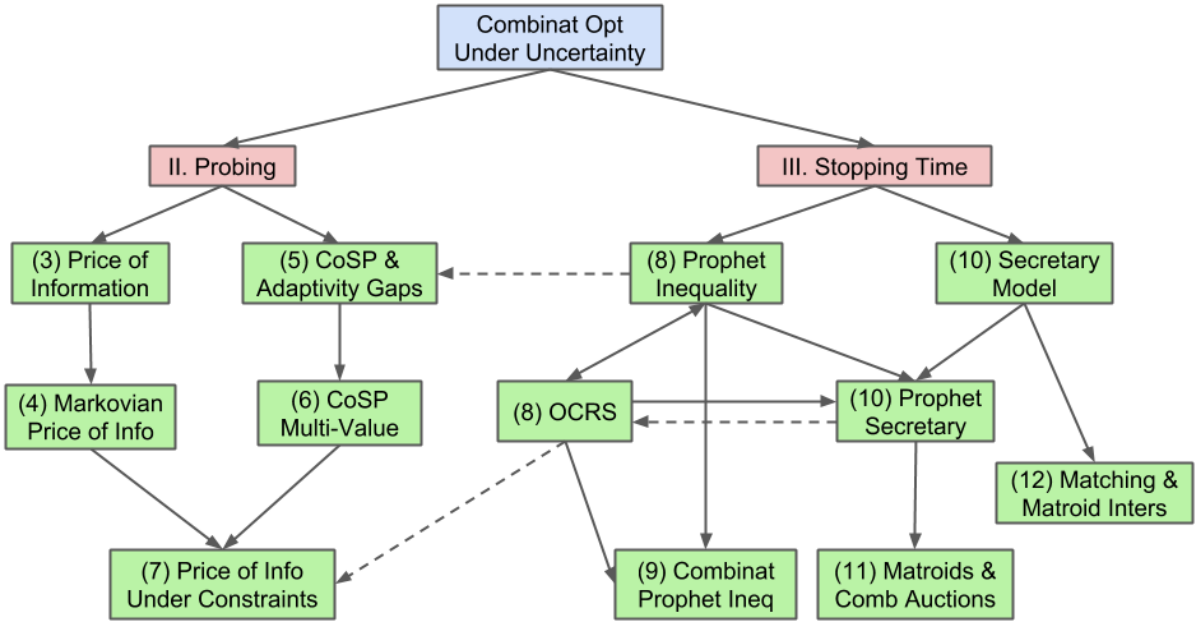

Figure 1.2: Roadmap of chapters

thesis we want to design *efficient* algorithms that have *provable* guarantees. We achieve this by exploiting the special "product" structure of the state space in our problems: each state is formed by the values of exactly n sites.

One naïve way to simplify our MDP is to decompose it by optimizing for each site separately. Such an approach, however, does not have a good performance because there is a common combinatorial constraint relating each of the sites: we can only purchase 1 house. There has been some previous work on studying MDPs that are formed by *weakly coupled* Markov chains [SC98, MHK+98, DGV04, GM07]. In these models, there are n independent Markov chains that are related by only a "few" common constraints (e.g., one budget constraint). The work in this thesis can be thought of as designing algorithms for settings where the Markov chains are "strongly coupled" because the underlying combinatorial constraint is more complicated (e.g., any packing/covering constraint).

1.5.4 Roadmap

The high level connections between various chapters is described in Figure 1.2.

Chapter 2: We start with some preliminaries and define the common notation used throughout this thesis. We also define the most common combinatorial functions and constraints, and discuss some of their properties.

Chapter 3: We motivate and introduce the price of information model (PoI) where you

maximize value minus the price. We formally define the notion of a "greedy" (FRUGAL) algorithm and show how to convert any such algorithm into a probing strategy. We also argue how (variants of) many existing algorithms are FRUGAL. This chapter is based on [Sin18].

Chapter 4: We define and study a multistage probing mode where a Markov chain represents the evolution of a site's value. This chapter is based on a joint work with Anupam Gupta, Haotian Jiang, and Ziv Scully [GJSS18].

Chapter 5: We consider the best-box problem (CoSP) and its generalization to maximizing a submodular function over arbitrary packing constraints. We prove the adaptivity gaps are small, $O(1)$, and discuss several applications of this model. This chapter is based on joint works with Anupam Gupta and Viswanath Nagarajan [GNS16, GNS17] and with Domagoj Bradač and Goran Žužić [BSZ18].

Chapter 6: We consider the notion of adaptivity gaps for more general combinatorial functions: submodular over multiple independent items, weighted rank function of a k-extendible system (which generalizes intersection of k matroids), and monotone subadditive functions. This chapter is based on joint works with Anupam Gupta and Viswanath Nagarajan [GNS17] and with Domagoj Bradač and Goran Žužić [BSZ18].

Chapter 7: We explain why our techniques are amenable to changes in the model. We discuss three types of constraints: (a) probing, (b) commitment, and (c) sampling. This chapter is based on [Sin18] and a joint work with Anupam Gupta, Haotian Jiang, and Ziv Scully [GJSS18].

Chapter 8: We introduce the prophet inequality model and explain a generic technique, online contention resolution scheme. In general, this techniques can be use to handle commitment constraints. This chapter is based on a joint work with Euiwoong Lee [LS18].

Chapter 9: We consider more general combinatorial functions, submodular and subadditive, in the prophet inequality model. We present the first prophet inequalities for these functions. This chapter is based on a joint work with Aviad Rubinstein [RS17].

Chapter 10: We introduce the random arrival order, which leads to the secretary and prophet secretary models. We present simplified proofs of some existing results in the prophet secretary model and the first $O(\operatorname{poly}\log(n))$ approximation algorithms for subadditive functions in the secretary model. This latter result is based on a joint work Aviad Rubinstein [RS17].

Chapter 11: We consider the prophet secretary model for combinatorial auctions and matroids. We show how the random arrival order assumption allows us to improve existing prophet inequality algorithms. This chapter is based on a joint work with Soheil Ehsani, Mohammad Hajiaghayi, and Thomas Kesselheim [EHKS18].

Chapter 12: We consider matchings and matroid intersection in the secretary model and provide the first algorithms performing better than the 2-approximation greedy algorithm. This chapter is based on a joint work with Guru Guruganesh [GS17].

Chapter 13: We conclude with potential further directions and several open problems in probing and stopping-time models.

Chapter 2

Preliminaries

2.1 General Notation

For integers $a, b \in \mathbb{Z}$, denote $[a, b]$ to mean the set $\{a, a+1, \ldots, b\}$. For $n \geq 1$, denote $[n]$ to mean $\{1, 2, \ldots, n\}$. We mostly denote the finite ground set (universe) of elements by $[n]$ or V with $|V| = n$. For $S \subseteq V$, we use $\mathbf{1}_S$ to denote an indicator vector with i'th coordinate 1 for $i \in S$ and 0 otherwise. Depending on the context, the running time of an algorithm means the number of operations on the RAM model or the number of calls to an oracle (e.g., value or independence oracle).

In this thesis, we assume that if a probability distribution $\mathcal{D}$ is part of the algorithm's input then it has some concise (polynomial in n) representation. E.g., this could mean the distribution comes from a natural family, like gaussian or exponential, where we know the parameters of the distribution. It could also mean the distribution is over a polynomial sized set and we know the probability of each outcome. By $X \sim \mathcal{D}$ we denote a random variable (r.v.) X drawn from probability distribution $\mathcal{D}$. We assume that "simple" statistics of this r.v. X can be computed efficiently, e.g., $\mathbb{E}[X]$ or $\Pr[X > \tau]$ for a given $\tau \in \mathbb{R}$. Unless explicitly stated, all input random variables are assumed to be independent. Some times the algorithm might only have oracles access to independent samples from a distribution.

To design randomized algorithms, we assume access to perfect random variables with any specified bias. We do not worry about the difficulty of finding perfect randomness.

2.2 Combinatorial Functions

We denote the ground set by V, with $n = |V|$. A function $f : 2^V \to \mathbb{R}$ is *monotone* if $f(S) \leq f(T)$ for all $S \subseteq T \subseteq X$. The most common class of objective functions are additive.

Definition 2.2.1 (Additive/Linear function)**.** *Given a vector $\mathbf{c} \in \mathbb{R}^{|V|}$, we define $f(S) := \mathbf{c}^\intercal \cdot \mathbf{1}_S$ for any $S \subseteq V$, where $\mathbf{1}_S$ is a vector of dimension $|V|$ that contains 1 for $i \in S$ and 0 otherwise.*

Next we consider more general functions that often do not have a polynomial size description, but given value oracle access can be optimized for several constraint families.

Definition 2.2.2 (Combinatorial functions). *A function $f : 2^V \to \mathbb{R}$ is*

- Submodular *if for every $S, T \subseteq V$ it satisfies*

$$f(S \cap T) + f(S \cup T) \leq f(S) + f(T).$$

E.g., max value function, number of distinct items, matroid rank function, coverage function, cut function, entropy, and log determinant.

- Monotone XOS (Fractionally subadditive) *if it can be written as the maximum of additive functions: i.e., there are vectors $\mathbf{c_i} \in \mathbb{R}^{\mathbf{n}}_{\geq \mathbf{0}}$ for $i \in \{1 \dots W\}$ such that*

$$f(S) := \max_{i=1}^{W} \left(\mathbf{c}_i^{\intercal} \cdot \mathbf{1}_S \right) = \max_{i=1}^{W} \left(\sum_{j \in S} \mathbf{c}_i(j) \right).$$

An alternative characterization due to Feige [Fei09], which motivates the name fractionally subadditive*: a function is XOS if $f(T) \leq \sum_i \alpha_i f(S_i)$ for all $\alpha_i \geq 0$ and $\chi_T = \sum_i \alpha_i \chi_{S_i}$. The* width *of an XOS function is the smallest number W such that f can be written as the maximum over W linear functions.*

E.g., maximum-weight matching, rank of intersection of k matroids, knapsack, longest increasing subsequence, and largest independent set in a graph.

- Subadditive *if for every $S, T \subseteq V$ it satisfies*

$$f(S \cup T) \leq f(S) + f(T).$$

E.g., Set cover, Steiner tree, facility location, multiway cut, TSP, and makespan.

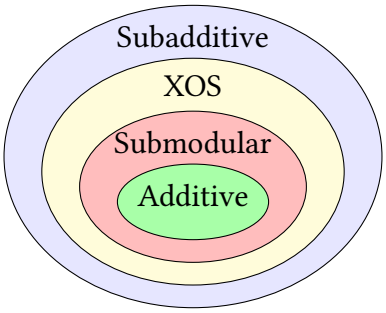

Figure 2.1: Hierarchy of monotone combinatorial functions.

Although the above combinatorial functions capture many natural problems, we stress that there are many interesting classes of functions beyond these classes. We list a few such functions below, but they will not be discussed in much detail in this thesis.

- k-median or k-center for a set of vertices in a metric.
- Maximum flow from s to t: a subadditive function over paths but not edges.
- Largest connected component.
- Some scheduling objectives like sum of job completion times.

2.3 Combinatorial Constraints

The most common families of constraints are packing and covering constraints.

Definition 2.3.1 (Downward-closed or packing constraints)**.** *An independence family $\mathcal{F} \subseteq 2^V$ is called* downward-closed *if $A \in \mathcal{F}$ and $B \subseteq A$ implies $B \in \mathcal{F}$. Some examples are:*

- Matroid $(V, \mathcal{F})$*: The elements of $\mathcal{F}$ satisfy the matroid exchange axiom: for all pairs of sets $I, J \in \mathcal{F}$ such that $|I| < |J|$, there exists an element $x \in J$ such that $I \cup \{x\} \in \mathcal{F}$. Elements of $\mathcal{F}$ are called independent sets. E.g., uniform matroid $\mathcal{F} = \{S : |S| \leq k\}$ and partition matroid $\mathcal{F} = \{S : |S \cap A_i| \leq a_i\}$ for a given partition $\{A_1, A_2, \ldots\}$ of V.*
- Intersection of matroids*: If there exist two matroids $\mathcal{M}_1 = (V, \mathcal{F}_1)$ and $\mathcal{M}_2 = (V, \mathcal{F}_2)$ such $\mathcal{F} = \mathcal{F}_1 \cap \mathcal{F}_2$. E.g., bipartite matching is intersection of two partition matroids.*
- Knapsack*: Given $w \in \mathbb{R}^n_{\geq 0}$ and a knapsack size $B > 0$, set $S \in \mathcal{F}$ iff $\sum_{i \in S} w_i \leq B$.*
- Orienteering*: Given a metric (V, E, d), a budget $B > 0$, and a root $r \in V$, a set $S \subseteq V$ is in $\mathcal{F}$ iff there exists a walk starting at r of length at most B that visits all nodes in S.*
- k-extendible system*: If for every $A \subseteq B \in \mathcal{F}$ and $e \in T$ where $A \cup \{e\} \in \mathcal{F}$, we have that there is a set $Z \subseteq B \setminus A$ such that $|Z| \leq k$ and $B \setminus Z \cup \{e\} \in \mathcal{F}$. These are more general than intersection of matroids; e.g., a 2-extendible system captures matching in general graphs.*
- k-system*: If for any $S \subseteq V$, we have*

$$\frac{\max_{J : J \text{ is a base of } S} |J|}{\min_{J : J \text{ is a base of } S} |J|} \leq k.$$

These are more general than a k-extendible system [Mes06, CCPV11].

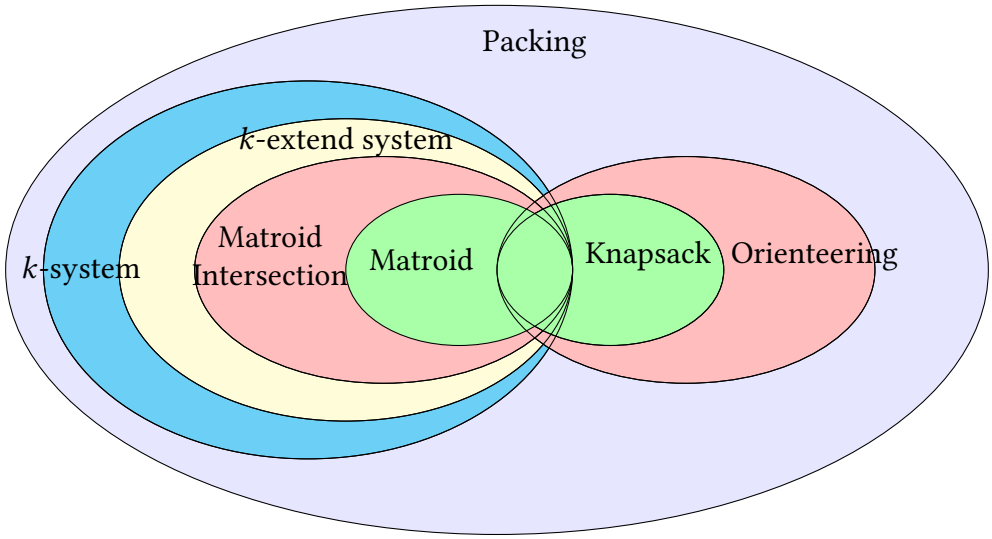

Figure 2.2: Hierarchy of packing constraints.

We now define and give some examples of covering constraints.

Definition 2.3.2 (Upward-closed or covering constraints)**.** *An independence family $\mathcal{F} \subseteq 2^V$ is called* downward-closed *if $A \in \mathcal{F}$ and $B \supseteq A$ implies $B \in \mathcal{F}$. Some examples are:*

- Matroid Basis*: Given a matroid, set S is feasible iff it contains a matroid basis.*
- Set Cover*: Suppose for every $i \in V$ we are guven a set $T_i \subseteq [n]$. Now a set $S \subseteq V$ is feasible iff $\bigcup_{i \in S} T_i \supseteq [n]$.*
- Uncapacitated Facility Location*: Given a graph $G = (V, E)$ with metric (V, d),* Clients $\subseteq V$*, and facility opening costs $\mathbf{X} : V \to \mathbb{R}_{\geq 0}$, open facilities at some locations $\mathbb{I} \subseteq V$ to minimize the sum of facility opening costs and the connection costs to* Clients*, i.e.,*

$$\sum_{i \in \mathbb{I}} X_i + \sum_{j \in \text{Clients}} \min_{i \in \mathbb{I}} d(j, i).$$

- Steiner Tree*: Given a graph $G = (V, E)$ with some edge costs $\mathbf{c} : E \to \mathbb{R}$ and a subset of nodes $S \subseteq V$. The goal is to find a tree $T \subseteq E$ that connects all the nodes in S (it may or may not connect nodes in $V \setminus S$) of minimum cost*

$$\sum_{e \in T} c_e.$$

- Prize-Collecting Steiner Tree*: Given a graph $G = (V, E)$ with some edge costs $\mathbf{c} : E \to \mathbb{R}$, a root node $r \in V$, and* penalties *$\mathbf{X} : V \to \mathbb{R}$. The goal is to find a tree that connects a subset of nodes to r, while trying to minimize the cost of the tree and the sum of the penalties of nodes $\mathbb{I}$ not connected to r, i.e.,*

$$\sum_{i \in \mathbb{I}} X_i + \textit{Min-Steiner-Tree}(V \setminus \mathbb{I}),$$

  where Min-Steiner-Tree$(V \setminus \mathbb{I})$ is the minimum cost tree connecting all nodes in $V \setminus \mathbb{I}$ to r.
- Feedback Vertex Set*: Given an undirected graph $G = (V, E)$, suppose each node $i \in V$ has a cost $X_i \in \mathbb{R}_{\geq 0}$. The problem is select a subset $\mathbb{I} \subseteq V$ s.t. the induced graph $G[V \setminus \mathbb{I}]$ contains no cycle, while minimizing the total cost*

$$\sum_{i \in \mathbb{I}} X_i.$$

Again, we stress that there are important families of constraints that are not packing/covering constraints, e.g., mixed packing-covering constraints. These will not be the focus of this thesis.

2.4 Some Properties of Combinatorial Functions

2.4.1 Submodular Functions

We recall some notation to extend submodular functions from the discrete hypercube $\{0, 1\}^n$ to relaxations whose domain is the continuous hypercube $[0, 1]^n$.

For any vector $\mathbf{x} \in [0,1]^n$, let $S \sim \mathbf{x}$ denote a random set S that contains each element $i \in [n]$ independently w.p. x_i. Moreover, let $\mathbf{1}_S$ denote a vector of length n containing 1 for $i \in S$ and 0 for $i \notin S$.

Definition 2.4.1. *We define important continuous extensions of any set function f.*

Multilinear extension F:

$$F(\mathbf{x}) \triangleq \mathbb{E}_{S \sim \mathbf{x}}\Big[f(S)\Big].$$

Concave closure f^+:

$$f^+(\mathbf{x}) \triangleq \max_{\alpha} \Big\{ \sum_{S \subseteq [n]} \alpha_S f(S) \mid \sum_S \alpha_S = 1 \text{ and } \sum_S \alpha_S \mathbf{1}_S = \mathbf{x} \Big\}.$$

Continuous relaxation f^*:

$$f^*(\mathbf{x}) \triangleq \min_{S \subseteq [n]} \Big\{ f(S) + \sum_{i \in [n] \setminus S} f_S(e) \cdot x_i \Big\}.$$

Some Useful Results

Lemma 2.4.2 (Correlation gap [CCPV07])**.** *For any monotone submodular function and $\mathbf{x} \in [0,1]^n$,*

$$F(\mathbf{x}) \leq f^+(\mathbf{x}) \leq f^*(\mathbf{x}) \leq \left(1 - \frac{1}{e}\right)^{-1} F(\mathbf{x}).$$

Lemma 2.4.3 (Lemma 2.2 of [BFNS14])**.** *Consider any submodular function f and any set $A \subseteq [n]$. Let S be a random subset of A that contains each element of S w.p. at most p (not necessarily independently), then*

$$\mathbb{E}_S[f(S)] \geq (1-p) \cdot f(\emptyset).$$

Lemma 2.4.4 (Lemma 2.3 of [FMV11])**.** *For any non-negative submodular function f and any sets $A, B \subseteq [n]$,*

$$\mathbb{E}_{\substack{S \sim \mathbf{1}_A/2 \\ T \sim \mathbf{1}_B/2}}[f(S \cup T)] \geq \frac{1}{4}\left(f(\emptyset) + f(A) + f(B) + f(A \cup B)\right).$$

Lemma 2.4.5 (Theorem 2.1 of [FMV11])**.** *For any non-negative submodular function f, any set $A \subseteq [n]$ and $p \in [0,1]$,*

$$F(1_A \cdot p) \geq p(1-p) \cdot \max_{T \subseteq A} f(T).$$

We can now prove the following useful variant of the previous two lemmas.

Lemma 2.4.6. *Consider any non-negative submodular function f and $0 \leq L \leq H \leq 1$. Let S^* be a set that maximizes f, and let $\mathbf{x} \in [0,1]^n$ be such that for all $i \in [n]$, $L \leq x_i \leq H$. Then,*

$$F(\mathbf{x}) \geq L(1-H) \cdot f(S^*).$$

Proof. Imagine a process in which in which we construct the random set $T \sim \mathbf{x}$, i.e. set T containing each element e independently w.p. x_e, in two steps. In the first step we construct set T' by selecting every element independently w.p. exactly L. In the second step we construct set $\tilde{T}$ containing each element e independently w.p. $(x_e - L)/(1-L)$. It's easy to verify that the union of the two sets $T' \cup \tilde{T}$ contains each element e independently with probability exactly x_e.

From Lemma 2.4.5, we know that at the end of first step, the generated set has expected value $\mathbb{E}[f(T')] \geq L(1-L)f(S^*)$. Now, we argue that the second step does not "hurt" the value by a lot. We note that in the second step each element is added w.p. at most $(H-L)/(1-L)$ because $x_e \leq H$. Let $g(S) := f(S \cup T')$ be a non-negative submodular function. We apply Lemma 5.4.3 on g to get $\mathbb{E}[g(\tilde{T})] \geq (1-H)/(1-L) \cdot g(\emptyset)$, which implies $\mathbb{E}[f(T' \cup \tilde{T})] \geq (1-H)/(1-L) \cdot \mathbb{E}[f(T')]$.

Together, we get $F(\mathbf{x}) = \mathbb{E}[T' \cup \tilde{T}] \geq L(1-L)(1-H)/(1-L)f(S^*) = L(1-H)f(S^*)$. □

Given any function $f : 2^X \to \mathbb{R}$, define $f^{\max}(S) := \max_{T \subseteq S} f(T)$ to be the maximum value subset contained within S. The function f is monotone if and only if $f^{\max} = f$. In general, $f^{\max}$ may be difficult to compute given access to f. However, Feige et al. [FMV11] show that for submodular functions

$$\frac{1}{4} f^{\max}(S) \leq \mathbb{E}_{R \sim S(\frac{1}{2})}[f(R)] \leq f^{\max}(S). \tag{2.1}$$

2.4.2 XOS and Subadditive Functions

A set function $f : \{0,1\}^n \to \mathbb{R}_+$ is *subadditive* if for all $S, T \subseteq [n]$ it satisfies $f(S \cup T) \leq f(S) + f(T)$. It's monotone if $f(S) \leq f(T)$ for any $S \subseteq T$. A set function $f : \{0,1\}^n \to \mathbb{R}_+$ is an *XOS* (alternately, *fractionally subadditive*) function if there exist linear functions $L_i : \{0,1\}^n \to \mathbb{R}_+$ such that $f(S) = \max_i \{L_i(S)\}$. (See Feige [Fei09] for illustrative examples.)

Lemma 2.4.7 ([Dob07, Lemma 3])**.** *Any subadditive function $v : 2^n \to \mathbb{R}_+$ can be $\left(\frac{\log |S|}{2e}\right)$-approximated by an XOS function $\widehat{v} : 2^n \to \mathbb{R}_+$; i.e. for every $S \subseteq [n]$,*

$$\widehat{v}(S) \leq v(S) \leq \left(\frac{\log |S|}{2e}\right) \widehat{v}(S).$$

Furthermore, the XOS function has the form $\widehat{v}(S) = \max_{T \subseteq [n]} p_T \cdot |T \cap S|$ for an appropriate choice of p_T's.

The following lemma is obvious if we represent a monotone XOS function as a max over linear functions.

Lemma 2.4.8. *For a monotone XOS function and any distribution $\mathcal{D}$ over subsets of S such that $\Pr_{T \sim \mathcal{D}}[i \in T] \geq p$, we have $\mathbb{E}_{T \sim \mathcal{D}} f(T) \geq p \cdot f(S)$.*

We remark that since Shannon's entropy is a monotone submodular function, which is a monotone XOS function, Lemma 2.4.8 gives Shearer's lemma as a corollary.

Part II

Probing Algorithms

Chapter 3

The Price of Information via Frugal Algorithms

3.1 Introduction

As discussed in the introduction (§1.3.1), the House-Purchasing scenario can be modeled as Weitzman's "Pandora's box" problem: Given probability distributions of n independent r.v.s $\{X_i\}$ (representing the value of house at site i) and their *probing* prices π_i, the goal is design a strategy to *adaptively* probe a set Probed $\subseteq \{1, 2, \ldots, n\}$ to maximize expected *utility*:

$$\mathbb{E}\Big[\max_{i \in \text{Probed}} \{X_i\} - \sum_{i \in \text{Probed}} \pi_i\Big].$$

An optimal policy for this problem was given by [Wei79]. Now suppose instead of probing values of sites, we probe weights of edges in a graph. Our utility is the maximum-weight matching that we find *minus* the total probing prices that we pay. What probing strategy should we adopt?

In a different scenario, consider a network design *minimization* problem. Suppose we wish to lay down a minimum-cost spanning tree in a graph; however, we only have stochastic information about the edge costs. To find the precise cost X_i of any edge, we have to conduct a study that incurs a price π_i. Our *disutility* is the sum of the tree cost and the total price that we spend on the studies. We want to design a strategy to minimize our expected disutility. An important difference between these two scenarios is that of maximizing utility vs minimizing disutility.

Situations like the above often arise where we wish to find a "good" solution to an optimization problem; however, we start with only some partial knowledge about the parameters of the problem. The missing information can be found only after paying a *probing* price, which we call the *price of information* (hereby, PoI). In this work we design optimal/approximation algorithms for several combinatorial optimization problems in an uncertain environment where we jointly optimize the value of the solution and the price of information [Sin18].

3.1.1 Model and Results

To begin, the above maximum-weight matching problem can be formally modeled as follows.

Max-Weight Matching Given a graph G with edges E, suppose each edge $i \in E$ takes some random weight X_i independently from a known probability distribution. We can find the exact outcome X_i only after paying a *probing price* π_i. The goal is to adaptively probe a set of edges $\text{Probed} \subseteq E$ and select a matching $\mathbb{I} \subseteq \text{Probed}$ to maximize the expected *utility*,

$$\mathbb{E}\Big[\sum_{i\in\mathbb{I}} X_i - \sum_{i\in\text{Probed}} \pi_i\Big],$$

where the expectation is over random variables $\mathbf{X} = (X_1, \ldots, X_n)$ and any internal randomness of the algorithm. We observe that we can only select an edge if it has been probed and we might select only a subset of the probed edges. This matching problem can be used to model kidney exchanges where testing compatibility of donor-receiver pairs has an associated price.

To capture value functions of more general combinatorial problems in a single framework, we define the notion of *semiadditive* functions.

Definition 3.1.1 (Semiadditive function)**.** *We say a function* $f(\mathbb{I}, \mathbf{X}) : 2^V \times \mathbb{R}_{\geq 0}^{|V|} \to \mathbb{R}_{\geq 0}$ *is* semi-additive *if there exists a function* $h : 2^V \to \mathbb{R}_{\geq 0}$ *such that*

$$f(\mathbb{I}, \mathbf{X}) = \sum_{i\in\mathbb{I}} X_i + h(\mathbb{I}).$$

For example, for max-weight matching our value function $f(\mathbb{I}, \mathbf{X}) = \sum_{i\in\mathbb{I}} X_i$ is *additive*, i.e. $h(\mathbb{I}) = 0$. We call these functions semiadditive because the second term $h(\mathbb{I})$ is allowed to effect the function in a "non-additive" way; however, not depending on $\mathbf{X}$. Consider other examples.

- *Uncapacitated Facility Location*: Given a graph $G = (V, E)$ with metric (V, d), $\textsc{Clients} \subseteq V$, and facility opening costs $\mathbf{X} : V \to \mathbb{R}_{\geq 0}$, we wish to open facilities at some locations $\mathbb{I} \subseteq V$. The function is the sum of facility opening costs and the connection costs to $\textsc{Clients}$. Hence, $f(\mathbb{I}, \mathbf{X}) = \sum_{i\in\mathbb{I}} X_i + \sum_{j\in\textsc{Clients}} \min_{i\in\mathbb{I}} d(j, i)$, where $h(\mathbb{I}) = \sum_{j\in\textsc{Clients}} \min_{i\in\mathbb{I}} d(j, i)$ only depends on $\mathbb{I}$, and not on facility opening costs $\mathbf{X}$.
- *Prize-Collecting Steiner Tree*: Given a graph $G = (V, E)$ with edge costs $\mathbf{c} : E \to \mathbb{R}_{\geq 0}$, a root $r \in V$, and *penalties* $\mathbf{X} : V \to \mathbb{R}_{\geq 0}$. The goal is to find a tree that connects a subset of nodes to r, while trying to minimize the cost of the tree and the sum of the penalties of nodes $\mathbb{I}$ not connected to r. Hence, $f(\mathbb{I}, \mathbf{X}) = \sum_{i\in\mathbb{I}} X_i + \text{Min-Steiner-Tree}(V \setminus \mathbb{I})$, where $\text{Min-Steiner-Tree}(V \setminus \mathbb{I})$ denotes the minimum cost tree connecting nodes in $V \setminus \mathbb{I}$ to r.

We can now describe an abstract utility-maximization model that captures problems such as Pandora's box, max-weight matching, and max-spanning tree, in a single unifying framework.

Utility-Maximization Suppose we are given a downward-closed (packing) constraint $\mathcal{F} \subseteq 2^V$ and a semiadditive function val. Each element $i \in V$ takes a value X_i independently from a known probability distribution. To find the outcome X_i we have to pay a known probing price π_i. The goal is to adaptively probe a set of elements $\text{Probed} \subseteq V$ and select $\mathbb{I} \subseteq \text{Probed}$ that is

feasible (i.e., $\mathbb{I} \in \mathcal{F}$) to maximize the expected *utility*,

$$\mathbb{E}\Big[\mathsf{val}(\mathbb{I}, \mathbf{X}) - \sum_{i \in \mathsf{Probed}} \pi_i\Big],$$

where the expectation is over random variables $\mathbf{X}$ and any internal randomness of the algorithm.

For example, in the max-weight matching problem val is an additive function and a subset of edges $\mathbb{I}$ is feasible if they form a matching. Similarly, when val is additive and $\mathcal{F}$ is a graphic matroid, this captures max-weight spanning tree.

The following is our main result for the utility-maximization problem:

Theorem 3.1.2. *For the utility-maximization problem for additive value functions and various packing constraints $\mathcal{F}$, we obtain the following efficient algorithms.*

- k-system*: For stochastic element values, we find a k-approximation algorithm.*
- Knapsack*: For stochastic item values, we find a 2-approximation algorithm.*

Some important corollaries of Theorem 3.1.2 are an optimal algorithm for the max-weight matroid rank problem[1] and a 2-approximation algorithm for the max-weight matching problem.

Next, we describe a disutility-minimization model to capture problems like min spanning tree.

Disutility-Minimization Suppose we are given an upward-closed (covering) constraints $\mathcal{F}' \subseteq 2^V$ and a semiadditive function cost. Each element $i \in V$ takes a value X_i independently from a known probability distribution. To find the outcome X_i we have to pay a known probing price π_i. The goal is to adaptively probe a set of elements Probed $\subseteq V$ and select $\mathbb{I} \subseteq$ Probed that is feasible (i.e., $\mathbb{I} \in \mathcal{F}'$) to minimize the expected *disutility*,

$$\mathbb{E}\Big[\mathsf{cost}(\mathbb{I}, \mathbf{X}) + \sum_{i \in \mathsf{Probed}} \pi_i\Big],$$

where the expectation is over random variables $\mathbf{X}$ and any internal randomness of the algorithm.

For example, in the min-cost spanning tree problem, cost is an additive function and a subset of edges $\mathbb{I}$ are in $\mathcal{F}'$ if they contain a spanning tree. Similarly, when val is the semiadditive facility location function as defined earlier and every non-empty subset of V is feasible in $\mathcal{F}'$, this captures the min-cost facility location problem. We now mention our results in this model.

Theorem 3.1.3. *For the disutility-minimization problem for various covering constraints $\mathcal{F}'$, we obtain the following efficient algorithms.*

- Matroid Basis*: For stochastic element costs, we find the optimal adaptive algorithm.*
- Set Cover*: For stochastic costs of the sets, we find a $\min\{O(\log |V|), f\}$-approximation algorithm, where V is the universe and f is the maximum number of sets in which an element can occur.*

[1] For weighted matroid rank functions, Kleinberg et al. [KWW16] independently obtained a similar result.

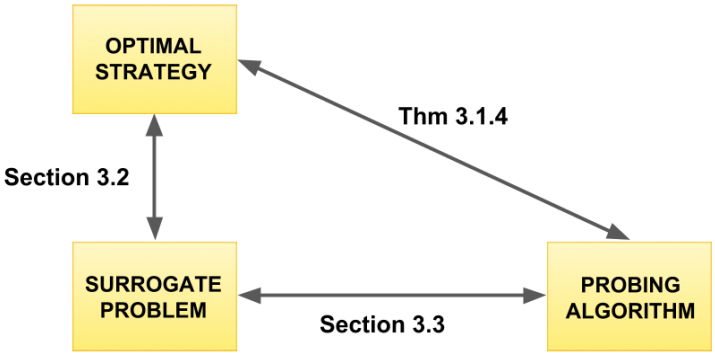

Figure 3.1: To prove Theorem 3.1.4, we first bound the optimal strategy using a surrogate problem in §3.2, and then obtain utility close to the surrogate by transforming the given Frugal algorithm to a probing algorithm in §3.3.

- Uncapacitated Facility Location*: For stochastic facility opening costs in a given metric, we find a* 1.861*-approximation algorithm.*
- Feedback Vertex Set*: For stochastic vertex costs in a given graph we find an* $O(\log n)$*-approximation algorithm.*

3.1.2 Techniques

How do we bound the utility/disutility of the optimal adaptive strategy? The usual techniques in approximation algorithms for stochastic problems (see related work in §4.1.4) either use a linear program (LP) to bound the optimal strategy, or directly argue about the adaptivity gap of the optimal decision tree. Neither of these techniques is helpful because the natural LPs fail to capture a mixed-sign objective—they wildly overestimate the optimal value. On the other hand, the adaptivity gap of our problems is large even for the special case of the Pandora's box problem.

We need two crucial ideas for both our utility-maximization and disutility-minimization results. Our first idea (§3.2) is to show that for semiadditive functions, one can bound the utility/disutility of the optimal strategy in the price-of-information world (hereafter, the *PoI world*) using a related instance in a world where there is no price to finding the parameters, i.e., $\pi_i = 0$ (hereafter, the *free-information world*). This proof crucially relies on the semiadditive nature of our value/cost function. Next, we show how to design strategies with utility/disutility close to this bound.

Our second idea (§3.3) is to show that any algorithm with "nice" properties in the free-information world can be used to get an algorithm with a similar expected utility/disutility in the PoI world. We call such a nice algorithm Frugal. For intuition, imagine a Frugal algorithm to be a greedy algorithm, or an algorithm that is not "wasteful"—it picks elements irrevocably. This includes simple primal-dual algorithms that do not have the *reverse-deletion* step. The following is our main technical theorem (see Figure 3.1 for proof idea).

Theorem 3.1.4. *If there exists a* Frugal *α-approximation Algorithm $\mathcal{A}$ to maximize (minimize) a semiadditive function over some packing constraints $\mathcal{F}$ (covering constraints $\mathcal{F}'$) in the free-information world then there exists an α-approximation algorithm for the corresponding utility-maximization (disutility-minimization) problem in the PoI world.*

Finally, to prove our results, in §3.4 we show modifications of the corresponding algorithms are Frugal.

3.1.3 Related Work

An influential work of Dean et al. [DGV04] considered the stochastic knapsack problem where we have stochastic knowledge about the sizes of the items. Chen et al. [CIK+09] studied stochastic matchings where we find about an edge's existence only after probing, and Asadpour et al. [ANS08] studied stochastic submodular maximization where the items may or may not be present. Several followup papers have appeared, e.g., for knapsack [BGK11, Ma14], packing integer programs [DGV05, CIK+09, BGL+12], budgeted multi-armed bandits [GM07, GKMR11, LY13, Ma14], orienteering [GM09, GKNR12, BN14], submodular objectives [GN13, ASW14], and matchings [Ada11, BGL+12, BCN+15, AGM15]. Most of these results show that the stochastic problem has a small adaptivity gap and focus on the non-adaptive problem. There is also a large body of work in related models where information has a price. We refer the readers to the following papers and the references therein [GK01, CFG+02, KK03, GMS07, GM12].

Most of the above works do not capture mixed-sign objective of maximizing the value *minus* the prices. Some of them instead model this as a knapsack constraint on the prices. Moreover, most of them are for maximization problems as for the minimization setting even the "simplest" problem of probing at most k elements to minimize the expected minimum value has no polynomial approximation [GGM10].

For recent work on Pandora's box, the readers are referred to [OW15, Dov18]. Another very relevant paper is that of Kleinberg et al. [KWW16], while their results are to design auctions. Their proof of the Pandora's box problem inspired us to study designing optimal/approximation algorithms for combinatorial problems in the PoI world.

3.2 Bounding the Optimal Strategy

In this section we bound the expected utility/disutility of the optimal adaptive strategy for a combinatorial optimization in the PoI world in terms of a surrogate problem in the Free-Info world. We first define the *grade* τ and *surrogate* Y of non-negative random variables.

Definition 3.2.1 (*Grade* τ)**.** *For any non-negative random variable X_i, let $\tau_i^{\max}$ be the solution to equation $\mathbb{E}[(X_i - \tau_i^{\max})^+] = \pi_i$ and let $\tau_i^{\min}$ be the solution to equation $\mathbb{E}[(\tau_i^{\min} - X_i)^+] = \pi_i$.*

Definition 3.2.2 (*Surrogate* Y)**.** *For any non-negative random variable X_i, we denote $Y_i^{\max} := \min\{X_i, \tau_i^{\max}\}$ and $Y_i^{\min} := \max\{X_i, \tau_i^{\min}\}$.*

Note that $\tau_i^{\max}$ could be negative in the above definition. The following lemmas bound the

optimal strategy in the PoI world in terms of the optimal strategy of a surrogate problem in the Free-Info world.

Lemma 3.2.3. *The expected utility of the optimal strategy to maximize a semiadditive function* val *over packing constraints* $\mathcal{F}$ *in the PoI world is at most*

$$\mathbb{E}_{\mathbf{X}}\left[\max_{\mathbb{I}\in\mathcal{F}}\{\mathsf{val}(\mathbb{I},\mathbf{Y}^{\max})\}\right].$$

Lemma 3.2.4. *The expected disutility of the optimal strategy to minimize a semiadditive function* cost *over covering constraints* $\mathcal{F}'$ *in the PoI world is at least*

$$\mathbb{E}_{\mathbf{X}}\left[\min_{\mathbb{I}\in\mathcal{F}'}\{\mathsf{cost}(\mathbb{I},\mathbf{Y}^{\min})\}\right].$$

We only prove Lemma 4.3.3 as the proof of Lemma 3.2.4 is similar. The ideas in this proof are similar to that of Kleinberg et al. [KWW16, Lemma 1] to bound the optimal adaptive strategy for Pandora's box.

Proof of Lemma 4.3.3. Consider a fixed optimal adaptive strategy. Let A_i denote the indicator variable that element i is selected into $\mathbb{I}$ and let $\mathbf{1}_i$ denote the indicator variable that element i is probed by the optimal strategy. Note that these indicators are correlated and the set of elements with non-zero A_i is feasible in $\mathcal{F}$. Now, the optimal strategy has expected utility

$$\begin{aligned}
&= \mathbb{E}\left[\mathsf{val}(\mathbb{I},\mathbf{X}) - \sum_{i\in\text{Probed}}\pi_i\right]\\
&= \mathbb{E}\left[\sum_i (A_iX_i - \mathbf{1}_i\pi_i)\right] + \mathbb{E}[h(\mathbb{I})]\\
&= \mathbb{E}\left[\sum_i (A_iX_i - \mathbf{1}_i\mathbb{E}_{X_i}[(X_i-\tau_i^{\max})^+])\right] + \mathbb{E}[h(\mathbb{I})],
\end{aligned}$$

using the definition of π_i. Since value of X_i is independent of whether it's probed or not, we simplify to

$$= \mathbb{E}\left[\sum_i (A_iX_i - \mathbf{1}_i(X_i-\tau_i^{\max})^+)\right] + \mathbb{E}[h(\mathbb{I})].$$

Moreover, since we can select an element into $\mathbb{I}$ only after probing, we have $\mathbf{1}_i \geq A_i$. This implies that the expected utility of the optimal strategy is

$$\begin{aligned}
&\leq \mathbb{E}\left[\sum_i (A_iX_i - A_i(X_i-\tau_i^{\max})^+)\right] + \mathbb{E}[h(\mathbb{I})]\\
&= \mathbb{E}\left[\sum_i A_iY_i^{\max}\right] + \mathbb{E}[h(\mathbb{I})]\\
&= \mathbb{E}[\mathsf{val}(\mathbb{I},\mathbf{Y}^{\max})].
\end{aligned}$$

Finally, since elements in $\mathbb{I}$ form a feasible set, this is at most $\mathbb{E}[\max_{\mathbb{I}\in\mathcal{F}}\{\mathsf{val}(\mathbb{I},\mathbf{Y}^{\max})\}]$. □

3.3 Designing an Adaptive Strategy

In this section we introduce the notion of a FRUGAL algorithm and prove Theorem 3.1.4. We need the following notation.

Definition 3.3.1 (Vector $\mathbf{Y}_M$). *For any vector* $\mathbf{Y}$ *with indices in* V *and any* $M \subseteq V$*, let* $\mathbf{Y}_M$ *denote a vector of length* $|V|$ *with entries* Y_j *for* $j \in M$ *and a symbol* $*$*, otherwise.*

3.3.1 A FRUGAL Algorithm

The notion of a FRUGAL algorithm is similar to that of a greedy algorithm, or any other algorithm that is not "wasteful"—it selects elements one-by-one and irrevocably. Its definition captures "non-greedy" algorithms such as the primal-dual algorithm for set cover that does not have the *reverse-deletion* step.

We define a FRUGAL algorithm in the packing setting. Consider a packing problem in the Free-Info world (i.e., $\forall i, \pi_i = 0$) where we want to find a feasible set $\mathbb{I} \in \mathcal{F}$ and $\mathcal{F} \subseteq 2^V$ are some downward-closed constraints, while trying to maximize a semiadditive function $\text{val}(\mathbb{I}, \mathbf{Y}) = \sum_{i \in \mathbb{I}} Y_i + h(\mathbb{I})$.

Definition 3.3.2 (FRUGAL Packing Algorithm)**.** *For a packing problem with constraints* $\mathcal{F}$ *and value function* val*, we say Algorithm* $\mathcal{A}$ *is* FRUGAL *if there exists a* marginal-value *function* $g(\mathbf{Y}, i, y) : \mathbb{R}^V \times V \times \mathbb{R} \to \mathbb{R}$ *that is increasing in* y*, and for which the pseudocode is given by Algorithm 6. We note that this algorithm always returns a feasible solution if we assume* $\emptyset \in \mathcal{F}$*.*

Algorithm 1 FRUGAL Packing Algorithm $\mathcal{A}$

1: Start with $M = \emptyset$ and $v_i = 0$ for each element $i \in V$.
2: For each element $i \notin M$, compute $v_i = g(\mathbf{Y}_M, i, Y_i)$. Let $j = \text{argmax}_{i \notin M \ \&\ M \cup i \in \mathcal{F}} \{v_i\}$.
3: If $v_j > 0$ then add j into M and go to Step 2. Otherwise, return M.

An example of a FRUGAL packing algorithm is the greedy algorithm to find the maximum weight spanning tree (or to maximize any weighted matroid rank function), where $g(\mathbf{Y}_M, i, Y_i) = Y_i$.

We similarly define a FRUGAL algorithm in the covering setting. Consider a covering problem in the Free-Info world where we want to find a feasible set $\mathbb{I} \in \mathcal{F}'$, where $\mathcal{F}' \subseteq 2^V$ is some upward-closed constraint, while trying to minimize a semiadditive function $\text{cost}(\mathbb{I}, \mathbf{Y}) = \sum_{i \in \mathbb{I}} Y_i + h(\mathbb{I})$.

Definition 3.3.3 (FRUGAL Covering Algorithm)**.** *For a covering problem with constraints* $\mathcal{F}'$ *and cost function* cost*, we say Algorithm* $\mathcal{A}$ *is* FRUGAL *if there exists a* marginal-value *function* $g(\mathbf{Y}, i, y) : \mathbb{R}^V \times V \times \mathbb{R} \to \mathbb{R}$ *that is increasing in* y*, and for which the pseudocode is given by Algorithm 2.*

We note that for a covering problem it is unclear whether Algorithm 2 returns a feasible solution as we do not appear to be looking at our covering constraints $\mathcal{F}'$*. To overcome this, we say the marginal-value function* g encodes $\mathcal{F}'$ *if whenever* M *is infeasible then there exists an element* $i \notin M$ *with* $v_i > 0$*. This means that the algorithm will return a feasible solution as long as* $V \in \mathcal{F}'$*.*

Algorithm 2 FRUGAL Coverage Algorithm $\mathcal{A}$

1: Start with $M = \emptyset$ and $v_i = 0$ for each element $i \in V$.
2: For each element $i \notin M$, compute $v_i = g(\mathbf{Y}_M, i, Y_i)$. Let $j = \text{argmax}_{i \notin M}\{v_i\}$.
3: If $v_j > 0$ then add j into M and go to Step 2. Otherwise, return M.

A simple example of a FRUGAL covering algorithm is the greedy min-cost set cover algorithm, where $g(\mathbf{Y}_M, i, Y_i) = \left(|\bigcup_{j \in M \cup i} S_j| - |\bigcup_{j \in M} S_j|\right)/Y_i$. Note that here g *encodes* our coverage constraints.

Remark: Observe that a crucial difference between FRUGAL packing and covering algorithms is that a FRUGAL packing algorithm has to handle $\mathbf{Y} \in \mathbb{R}^V$ (i.e. some entries in $\mathbf{Y}$ could be negative) but a FRUGAL covering algorithm has to only handle $\mathbf{Y} \in \mathbb{R}^V$. The intuition behind this difference is that unlike the disutility minimization problem, the utility maximization problem has a mixed-sign objective.

3.3.2 Using a FRUGAL Algorithm to Design an Adaptive Strategy

After defining the notion of a FRUGAL algorithm, we can now prove Theorem 3.1.4 (restated below).

Theorem 3.1.4. *If there exists a* FRUGAL *α-approximation Algorithm $\mathcal{A}$ to maximize (minimize) a semiadditive function over some packing constraints $\mathcal{F}$ (covering constraints $\mathcal{F}'$) in the free-information world then there exists an α-approximation algorithm for the corresponding utility-maximization (disutility-minimization) problem in the PoI world.*

We prove Theorem 3.1.4 only for the utility-maximization setting as the other proof is similar. Lemma 4.3.3 already gives us an upper bound on the expected utility of the optimal strategy for the utility-maximization problem in terms of the expected value of a problem in the Free-Info world. This Free-Info problem can be solved using Algorithm $\mathcal{A}$. The main idea in the proof of this theorem is to show that if Algorithm $\mathcal{A}$ is FRUGAL then we can also run a modified version of $\mathcal{A}$ in the PoI world and get the same expected utility.

Proof of Theorem 3.1.4. Let $Alg(\mathbf{Y}^{\max}, \mathcal{A})$ denote the set $\mathbb{I} \in \mathcal{F}$ returned by Algorithm $\mathcal{A}$ when it runs with element weights $\mathbf{Y}^{\max}$. Since $\mathcal{A}$ is an α-approximation algorithm (where $\alpha \geq 1$), we know

$$\text{val}(Alg(\mathbf{Y}^{\max}, \mathcal{A}), \mathbf{Y}^{\max}) \geq \frac{1}{\alpha} \cdot \max_{\mathbb{I} \in \mathcal{F}} \left\{\text{val}(\mathbb{I}, \mathbf{Y}^{\max})\right\}. \tag{3.1}$$

The following crucial lemma shows that one can design an adaptive strategy in the PoI world with the same expected utility.

Lemma 3.3.4. *If Algorithm $\mathcal{A}$ is* FRUGAL *then there exists an algorithm in the PoI world with expected utility*

$$\mathbb{E}_{\mathbf{X}}\left[\text{val}(Alg(\mathbf{Y}^{\max}, \mathcal{A}), \mathbf{Y}^{\max})\right].$$

Before proving Lemma 3.3.4, we finish the proof of Theorem 3.1.4. Recollect, Lemma 4.3.3 shows that $\mathbb{E}[\max_{\mathbb{I}\in\mathcal{F}}\{\mathsf{val}(\mathbb{I}, \mathbf{Y}^{\max})]$ is an upper bound on the expected optimal utility in the PoI world. Combining this with Lemma 3.3.4 and (3.1) gives an α-approximation algorithm in the PoI world. □

We first give some intuition for the missing Lemma 3.3.4. The lemma is surprising because it says that there exists an algorithm in the PoI world that has the same expected utility as Algorithm $\mathcal{A}$ in the Free-Info world, where there are no prices. The fact that in the Free-Info world Algorithm $\mathcal{A}$ can only get the smaller surrogate values $Y_i^{\max} = \min\{X_i, \tau_i^{\max}\}$, instead of the actual value X_i, comes to our rescue. We show that $Y^{\max}$ is defined in a manner to balance this difference in the values with the probing prices.

Proof of Lemma 3.3.4. Since $\mathcal{A}$ is Frugal, we would like to run Algorithm 6 in the PoI world. The difficulty is that we do not know $\mathbf{Y}^{\max}$ values of the unprobed elements. To overcome this hurdle, consider Algorithm 3 that uses the grade $\tau^{\max}$ as a proxy for $\mathbf{Y}^{\max}$ values of the unprobed elements.

Claim 3.3.5. *The set of elements returned by Algorithm 3 is the same as that by Algorithm 6 running with* $\mathbf{Y} = \mathbf{Y}^{\max}$.

Proof of Claim 3.3.5. We prove the claim by induction on the number of elements selected by Algorithm 3. Suppose the set of elements selected by both the algorithms into M are the same till now and Algorithm 3 decides to select element j in Step 3(a). This means that j is already probed before this step. The only concern is that Algorithm 3 selects j without probing some other element i based on its grade $\tau_i^{\max}$. We observe that this step is consistent with Algorithm 6 because $Y_i^{\max} \le \tau_i^{\max}$ and $g(\mathbf{Y}_M^{\max}, i, Y_i^{\max})$ is an increasing function in $Y_i^{\max}$, which implies

$$g(\mathbf{Y}_M^{\max}, i, Y_i^{\max}) \quad \le \quad g(\mathbf{Y}_M^{\max}, i, \tau_i^{\max}) \quad \le \quad g(\mathbf{Y}_M^{\max}, i, Y_j^{\max}).$$ □

An immediate corollary is that value of Algorithm 3 in the Free-Info world is

$$\mathbb{E}_{\mathbf{X}}\Big[\mathsf{val}(Alg(\mathbf{Y}^{\max}, \mathcal{A}), \mathbf{Y}^{\max})\Big]. \tag{3.2}$$

In Claim 3.3.6 we argue that this expression also gives expected utility of Algorithm 3 in the PoI world, which completes the proof of Lemma 3.3.4. □

Claim 3.3.6. *The expected utility of Algorithm 3 in PoI world is*

$$\mathbb{E}_{\mathbf{X}}\Big[\mathsf{val}(Alg(\mathbf{Y}^{\max}, \mathcal{A}), \mathbf{Y}^{\max})\Big].$$

Proof of Claim 3.3.6. We first expand the claimed expression,

$$\mathbb{E}_{\mathbf{X}}\Big[\mathsf{val}(Alg(\mathbf{Y}^{\max}, \mathcal{A}), \mathbf{Y}^{\max})\Big] = \mathbb{E}\Big[\sum_{i\in Alg(\mathbf{Y}^{\max}, \mathcal{A})} Y_i^{\max}\Big] + \mathbb{E}\Big[h(Alg(\mathbf{Y}^{\max}, \mathcal{A}))\Big]. \tag{3.3}$$

Algorithm 3 Utility-Maximization

1: Start with $M = \emptyset$ and $v_i = 0$ for all elements i.
2: For each element $i \notin M$:
   (a) if i is probed let $v_i = g(\mathbf{Y}^{\max}_M, i, Y^{\max}_i)$.
   (b) if i is unprobed let $v_i = g(\mathbf{Y}^{\max}_M, i, \tau^{\max}_i)$.
3: Consider the element $j = \text{argmax}_{i \notin M \ \&\ M \cup i \in \mathcal{F}}\{v_i\}$ and $v_j > 0$.
   (a) If j is already probed then select it into M and set $v_j = 0$.
   (b) If j is not probed then probe it. If $X_j \geq \tau^{\max}_j$ then select j into M and set $v_j = 0$.
4: If every element $i \notin M$ has $v_i = 0$ then return set M. Else, go to Step 2.

Observe that to prove the claim we can ignore the second term, $\mathbb{E}[h(Alg(\mathbf{Y}^{\max}, \mathcal{A}))]$, because it contributes the same in both the worlds (it is only a function of the returned feasible set). We now argue that in every step of Algorithm 3 the expected change in $\sum_{i \in M} Y^{\max}_i$ in the Free-Info world is the same as the expected increase in $\sum_{i \in M} X_i$ minus the probing prices in the PoI world.

We first consider the case when the next highest element j in Step 3 of Algorithm 3 is already probed and has $v_j > 0$. In this case, the algorithm selects element j. Since this element has been already probed before (but not selected then), it means $X_j < \tau^{\max}_j$ and $X_j = Y^{\max}_j$. Hence the increase in the value of the algorithm in both the worlds is X_j.

Next, consider the case that the next highest element j in Step 3 has not been probed before. Let μ_j denote the probability density function of random variable X_j. Now the expected increase in the value in the Free-Info world is

$$\tau^{\max}_j \cdot \Pr[X_j \geq \tau^{\max}_j] \quad = \quad \tau^{\max}_j \cdot \int_{t=\tau^{\max}_j}^{\infty} \mu_j(t)dt.$$

This is because the algorithm selects this element in this step only if its value is at least $\tau^{\max}_j$, in which case $Y^{\max}_j = \tau^{\max}_j$. On the other hand, the expected increase in the value in the PoI world is given by

$$-\pi_j + \int_{t=\tau^{\max}_j}^{\infty} t \cdot \mu_j(t)dt.$$

This is because we pay the probing cost π_j and get a positive value X_j only when $X_j \geq \tau^{\max}_j$. Now using the definition of $\tau^{\max}_j$, we can simplify the above equation to

$$-\int_{t=\tau^{\max}_j}^{\infty} (t - \tau^{\max}_j)\mu_j(t)dt + \int_{t=\tau^{\max}_j}^{\infty} t \cdot \mu_j(t)dt \quad = \quad \tau^{\max}_j \cdot \int_{t=\tau^{\max}_j}^{\infty} \mu_j(t)dt.$$

This shows that in every step of the algorithm the expected increase in the value in both the worlds is the same, thereby proving Claim 3.3.6. □

3.4 Applications to Utility/Disutility Optimization

In this section we show that for several combinatorial problems there exist FRUGAL algorithms. Hence we can use Theorem 3.1.4 to obtain optimal/approximation algorithms for the corre-

sponding utility-maximization or disutility-minimization problem in PoI world.

3.4.1 Utility-Maximization

To recollect, in the utility-maximization setting we are given a semiadditive value function $\text{val} \geq 0$ and a packing constraint $\mathcal{F}$. Our goal is to probe a set of elements Probed and select a feasible set $\mathbb{I} \subseteq \text{Probed}$ in $\mathcal{F}$ to maximize expected utility,

$$\mathbb{E}\Big[\text{val}(\mathbb{I}, \mathbf{X}) - \sum_{i \in \text{Probed}} \pi_i\Big].$$

We assume that $\emptyset \in \mathcal{F}$ and hence there always exist a solution of utility zero.

k-System

Let $\text{val}(\mathbb{I}, \mathbf{X}) = \sum_{i \in \mathbb{I}} X_i$ be an additive function and let $\mathcal{F}$ denote a k-system constraint in this setting. To prove Theorem 3.1.2, we observe that the greedy algorithm that starts with an empty set and at every step selects the next feasible element maximum marginal-value is an α-approximation algorithm is a Frugal algorithm as defined in Defn 3.3.2. We know that this greedy algorithm is a k-approximation for additive functions over a k-system in Free-Info world [Jen76, KH78]. Hence, Theorem 3.1.4 combined with the greedy algorithm gives the k-system part of Theorem 3.1.2 as a corollary.

Knapsack

Given a knapsack of size B, suppose each item i has a known size s_i ($\leq B$) but a stochastic value X_i. To find X_i, we have to pay probing price π_i. The goal is to probe a subset of items Probed and select a subset $\mathbb{I} \subseteq \text{Probed}$, where $\sum_{i \in \mathbb{I}} s_i \leq B$, to maximize the expected utility

$$\mathbb{E}\Big[\sum_{i \in \mathbb{I}} X_i - \sum_{i \in \text{Probed}} \pi_i\Big].$$

We can model this problem in our utility-maximization framework by taking $\text{val}(\mathbb{I}, \mathbf{X}) = \sum_{i \in \mathbb{I}} X_i$ and $\mathcal{F}$ to contain every subset S of items that fit into the knapsack.

In the Free-Info world, consider a greedy algorithm that sorts items in decreasing order based on the ratio of their value and size, and then selects items greedily in this order until the knapsack is full. This greedy algorithm does not always give a constant approximation to the knapsack problem. Similarly, an algorithm that selects only the most valuable item is not always a constant approximation algorithm (recollect that we can pick every item i because $s_i \leq B$). However, it's known that for any knapsack instance if we randomly run one of the previous two algorithms, each w.p. half, then this is a 2-approximation algorithm.

From Theorem 3.1.4, we can simulate the greedy algorithm in the PoI world. Also, using the solution to the Pandora's box problem, we can simulate selecting the most valuable item in the PoI world. Hence, consider an algorithm that for any given knapsack problem in the PoI world,

runs either the simulated greedy algorithm or the Pandora's box solution, each with probability half. Such an algorithm is a 2-approximation to the knapsack problem in the PoI world.

3.4.2 Disutility-Minimization

To recollect, in the disutility-minimization setting we are given a semiadditive cost function cost ≥ 0 and a covering constraint $\mathcal{F}'$. Our goal is to probe a set of elements Probed and select a feasible set $\mathbb{I} \subseteq$ Probed in $\mathcal{F}'$ to minimize expected disutility,

$$\mathbb{E}\Big[\text{cost}(\mathbb{I}, \mathbf{X}) + \sum_{i \in \text{Probed}} \pi_i\Big].$$

We will assume that $V \in \mathcal{F}$, and hence there always exists a feasible solution.

Matroid Basis

Given a matroid $\mathcal{M}$ of rank r on n elements, we consider the additive function $\text{cost}(\mathbb{I}, \mathbf{X}) = \sum_{i \in \mathbb{I}} X_i$ and let $\mathcal{F}'$ be subsets of elements that contain a basis of $\mathcal{M}$. To ensure that a feasible set of finite value exists, we make the following assumption.

Assumption 3.4.1. *We can always extend a set $\mathbb{I} \in \mathcal{M}$ to a basis by probing and selecting items with zero penalty but with large probing cost π_0. Thus it incurs an additional penalty of $(r - |rank(\mathbb{I})|) \cdot \pi_0$.*

To use Theorem 3.1.4, we notice that the greedy algorithm that always selects the minimum cost independent element is Frugal. This is true because we choose marginal-value function g to be the reciprocal of the weight of an element in Defn 4.4.3. Now since the greedy algorithm is optimal for min-cost matroid basis, this proves the first part of Theorem 3.1.3.

Set Cover

Consider a problem where we are given sets $S_1, \ldots, S_m \subseteq V$ that have some unknown stochastic costs X_i. The goal is to select a set cover with minimum disutility, which is the sum of the set cover solution costs and the probing prices. We can model this problem in our framework by considering a the additive function $\text{cost}(\mathbb{I}, \mathbf{X}) = \sum_{i \in \mathbb{I}} X_i$ and $\mathcal{F}'$ be set covers of V.

To ensure that the solution is always bounded, we make the following assumption.

Assumption 3.4.2. *There exists $S_0 = [n]$ that covers all elements and has $X_0 = 0$, but a finite large π_0.*

To prove the set cover part of Theorem 3.1.3, we first notice that the classical $O(\log |V|)$ greedy algorithm for the min-cost set cover problem is Frugal. This is because the marginal-value function $g(\mathbf{Y}_M, i, Y_i)$ in Defn 4.4.3 is equal to $\Big(|\bigcup_{j \in M \cup i} S_j| - |\bigcup_{j \in M} S_j|\Big)/Y_i$.

Next we give an f-approximation algorithm, where f is the maximum number of sets in which an element can appear. We observe that the primal-dual f-approximation algorithm (see pseudocode in Algorithm 4) for the min-cost set cover [BYE81, WS11] is also Frugal. This is

because we can encode the information about the order σ and the dual variables y_j for $j \in M$ in the marginal-value function $g(\mathbf{Y}_M, i, Y_i)$ in Defn 4.4.3.

Algorithm 4 Primal-dual algorithm for min-cost set cover

1: Fix an order σ on the ground elements. Start with $M = \emptyset$ and $y_j = 0$ for every ground element j.
2: Select the next element $j \notin \bigcup_{i \in M} S_i$ according to σ and raise its dual variable y_j until some set i becomes tight, i.e., $\sum_{j \in S_i} y_j = Y_i$.
3: Select every tight set into M.
4: If every ground element is covered in $\bigcup_{i \in M} S_i$ then return M, else go to Step 2.

Uncapacitated Facility Location

Consider an uncapacitated facility location problem where we are given a graph $G = (V, E)$ with metric (V, d) and Clients $\subseteq V$, however, facility opening costs X_i for $i \in V$ are stochastic and can be found by paying a probing price π_i. The goal is to probe a set of facility locations Probed $\subseteq V$ and open a non-empty subset $I \subseteq$ Probed to minimize expected disutility

$$\mathbb{E}\Big[\sum_{u \in \text{Clients}} d(u, \mathbb{I}) + \sum_{i \in \mathbb{I}} X_i + \sum_{i \in \text{Probed}} \pi_i \Big],$$

where $d(u, \mathbb{I}) = \min_{i \in \mathbb{I}} d(u, i)$.

We model the above problem in our framework by defining exponential number of elements that are indexed by (i, S), for $i \in V$ and $S \subseteq$ Clients, which denotes that facility i will serve clients S. Any subset of elements, say $\mathbb{I} = \{(i_1, S_1), (i_2, S_2), \ldots\}$, is feasible if the union of their clients covers Clients. The semiadditive cost$(\mathbb{I}, \mathbf{X})$ is given by $\sum_{(i,S) \in \mathbb{I}} \Big(X_i + \sum_{j \in S} d(i, j) \Big)$.

We notice that the 1.861-approximation greedy algorithm of Jain et al. [JMM+03] for the uncapacitated facility location problem is Frugal. In each step, their algorithm selects the next best element with minimum cost per client, where already opened facilities now have zero opening costs. The reciprocal of this value gives the marginal-value function g. Hence, we can use Theorem 3.1.4 to obtain a 1.861-approximation strategy.

Prize-Collecting Steiner Tree

Consider a Prize-Collecting Steiner tree problem (PCST) where we are given a graph $G = (V, E)$ with some edge costs $\mathbf{c} : E \to \mathbb{R}$, a root node $r \in V$, and probability distributions on the independent penalties X_i for $i \in V$. The stochastic penalties X_i can be found by paying a probing price π_i. The goal is to probe a set of nodes Probed $\subseteq V \setminus \{r\}$ and select a subset $I \subseteq$ Probed to minimize expected disutility,

$$\mathbb{E}\Big[\sum_{i \in \mathbb{I}} X_i + \text{Min-Steiner-Tree}(V \setminus \mathbb{I}) + \sum_{i \in \text{Probed}} \pi_i \Big],$$

where Min-Steiner-Tree$(V \setminus \mathbb{I})$ is the min-cost Steiner tree connecting all nodes in $V \setminus \mathbb{I}$ to r.

We can model the PCST in our disutility-minimization framework by noticing that the function $\text{cost}(\mathbf{X}, \mathbb{I}) = \sum_{i \in \mathbb{I}} X_i + \text{Min-Steiner-Tree}(V \setminus \mathbb{I})$ is semiadditive. We show that although the 2-approximation Goemans-Williamson [GW95] algorithm (hereafter, GW-algorithm) for PCST in the Free-Info world is not Frugal, it can be modified to obtain a 3-approximation Frugal algorithm for PCST. Combining this with Theorem 3.1.4 gives a 3-approximation algorithm for PCST in the PoI world.

We quickly recollect the 2-approximation primal-dual GW-algorithm. (We do not repeat their proof and refer to [WS11, Chapter 14] for details.) Their algorithm starts by making each node $i \in V \setminus \{r\}$ active with initial charge $p(\{i\}) = X_i$. At any time, the algorithm grows a *moat* around each active component C and discharges C at the same rate. If a component C runs out of charge, we make it inactive and mark every unlabeled node in the component with label C. If an edge e becomes *tight*, we pick e, merge the two components C, C' connected by e, make both C, C' inactive, and make $C \cup C'$ active with an initial charge of $p(C) + p(C')$. Any component that hits the component containing r is made inactive. In the *cleanup phase* we remove all edges that do not disconnect an unmarked node from r, while ensuring that if a component with label C is connected to r then every node with label $C' \supseteq C$ is also connected to r.

We first observe that the GW-algorithm is not Frugal. This is because whenever a node i is labeled with a component $C \ni i$, the algorithm looks at the penalty X_i; however, the decision of whether to select i into $\mathbb{I}$ (i.e., not connecting i to r) is not made until the cleanup phase. The reason is that some other active component C' might later come and merge with C, and eventually connect i to r. To fix this, we modify this algorithm to make it Frugal. The idea is to immediately include the labeled vertices into $\mathbb{I}$.

Consider an algorithm that creates the same tree as the GW-algorithm; however, any node that ever gets labeled during the run of the algorithm is imagined to be included into $\mathbb{I}$. This means that although our final tree might connect a labeled node i to r, our algorithm still pays its penalty X_i. We argue that these additional penalties are at most the optimal PCST solution in the Free-Info world, which gives us a 3-approximation Frugal algorithm.

Finally, to argue that the additional penalties are not large, consider the state of the GW-algorithm before the cleanup phase. Let $\mathcal{C}$ denote the set of maximal inactive components. Clearly, each node i that was every labeled belongs to some maximal tight component $C \in \mathcal{C}$. Hence, the sum of the additional penalties is upper bounded by $\sum_{C \in \mathcal{C}} \sum_{i \in C} X_i$. Since each component $C \in \mathcal{C}$ is tight, we know $\sum_{C \in \mathcal{C}} \sum_{i \in C} X_i = \sum_{C \in \mathcal{C}} \sum_{S \subseteq C} y_S$, where y_S are the dual variables corresponding to the moats. Since the dual solutions form a feasible dual-solution, they are a lower bound on the optimal solution for the problem. This proves that the additional penalty paid by our Frugal algorithm in comparison to GW-algorithm is at most the optimal solution.

Feedback Vertex Set

Given an undirected graph $G = (V, E)$, suppose each node $i \in V$ has a stochastic weight X_i, which we can find by probing and paying price π_i. The problem is to probe a set Probed $\subseteq V$ and select a subset $\mathbb{I} \subseteq$ Probed s.t. the induced graph $G[V \setminus \mathbb{I}]$ contains no cycle, while minimizing

the expected disutility

$$\mathbb{E}\Big[\sum_{i\in\mathbb{I}} X_i + \sum_{i\in\text{Probed}} \pi_i\Big].$$

The above problem can be modeled in our framework by considering the additive function $\text{cost}(\mathbb{I},\mathbf{X}) = \sum_{i\in\mathbb{I}} X_i$ and $\mathcal{F}'$ contains a set of nodes S if $G[V \setminus S]$ has no cycle. Becker and Geiger [BG96] showed that the greedy Algorithm 5 is an $O(\log n)$-approximation algorithm for the feedback vertex set problem in the Free-Info world. Since this algorithm is Frugal, using Theorem 3.1.4 we get an $O(\log n)$-approximation algorithm for minimizing disutility for the feedback vertex set problem in the PoI world.

Algorithm 5 Greedy Algorithm for Feedback Vertex Set

1: Start with $R = M = \emptyset$ and $v_i = 0$ for each element $i \in V$.
2: While $\exists i \in V \setminus (R \cup M)$ s.t. degree of i in $G[V \setminus (R \cup M)]$ is 0 or 1, add i to R.
3: For each element $i \notin R \cup M$, compute $v_i = degree(i, G[V \setminus (R \cup M)])/w(i)$, where $degree(i,G)$ is the degree of vertex i in G and $w(i)$ is the weight of vertex i.
4: Let $j = \text{argmax}_{i\notin R\cup M}\{v_i\}$. Add j to M.
5: If $R \cup M \neq V$, go to Step 2. Otherwise, return M.

Remark: The $O(\log n)$-approximation primal-dual algorithm in [BYGNR98] (or in Chapter 7.2 of [WS11]) can be also shown to be Frugal. This gives another $O(\log n)$-approximation algorithm for minimizing disutility for feedback vertex set problem.

3.5 Illustrative Examples

3.5.1 Why the naïve greedy algorithm fails for Pandora's box

Suppose *curr* denotes the maximum value in the currently opened set of boxes. The naïve greedy algorithm selects in any step the unopened box j corresponding to the maximum marginal value, i.e. $\text{argmax}\{\mathbb{E}[(X_j - curr)^+] - \pi_j\}$, and opens it if its marginal value is non-negative. The algorithm stops probing when every unopened box has a negative marginal value. We give an example where this algorithm can be made arbitrarily worse as compared to the optimum.

Consider $n-1$ iid boxes, each taking value $1/p^2$ w.p. p and 0 otherwise, where $p < 1$. The probing price of each of these boxes is 1. Also, there is a box which takes value $1/p^2$ w.p. 1 but has a probing price of $1/p^2 - 1/p + 1$. Note that in the beginning, the marginal value of every box is $1/p - 1$. Now the optimal strategy is to probe the boxes with price 1, until we see a box with value $1/p^2$. For large enough n, the expected utility of this strategy is $\approx 1/p^2 - 1/p$ because in expectation the algorithm stops after roughly $1/p$ probes. However, the naïve greedy algorithm will open the box with price $1/p^2 - 1/p - 1$ and then stop probing. Thus its expected utility will be $1/p - 1$. By choosing small enough p, the ratio $(1/p^2 - 1/p)$ to $(1/p - 1)$ can be made arbitrarily large.

3.5.2 The Pandora's box problem has no constant approximation non-adaptive solution

Consider an example where each element independently takes value 1 w.p. p ($\ll 1$) and value 0, otherwise. Suppose the price of probing any element is $1-\epsilon$, for some small $\epsilon > 0$. The optimal adaptive strategy is to continue probing till we see an element with value 1. Assuming n to be large, it is easy to see that this strategy has expected value $\approx \epsilon/p$. On the other hand, a non-adaptive strategy has to decide in the beginning which all elements S to probe, and then probe them irrespective of the consequence. Since all items are identical, the only decision it has to make is how many items to probe. One can verify that no such non-adaptive strategy can get value more than $O(\epsilon)$. By choosing p to be small enough, we can make the gap arbitrarily large.

3.5.3 Hardness for general submodular functions

To prove that one cannot obtain good approximation results for any monotone submodular functions f, we show that even when all variables are deterministic, the computational problem of selecting the best set $\mathbb{I} \subseteq V$ to maximize $f(\mathbb{I}) - \pi(\mathbb{I})$ is $\tilde{\Omega}(\sqrt{n})$ hard assuming $\mathbf{P} \neq \mathbf{NP}$, where $n = |V|$. The idea is to reduce from set packing. Let $\S = \{S_1, S_2, \ldots, S_m\}$ denote the sets of a set packing instance. For $S \subseteq \S$, let $f(S)$ denote the covering function. Let price of probing S_i be $\pi_i = |S_i| - 1$. Clearly, it doesn't make sense to probe sets that are not disjoint as the marginal utility will be non-positive. The optimal solution therefore equals the maximum number of disjoint sets. But no polynomial time algorithm can be $O(n^{1/2-\epsilon})$-approximation, for any $\epsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$ [Has96, HKT00].

Chapter 4

Multistage Probing via the Markovian Price of Information

4.1 Introduction

Consider a scenario where you run an oil company and want to set up a new oil drill. You have estimates on the amount of oil (the "value") available at potential sites, say, based on prior surveys. To find the exact value at a site you need to conduct a closer inspection that incurs some "price". What inspection strategy should you adopt to maximize the expected value of the *best site* you find *minus* the total inspection price you pay?

Similar to the House-Purchasing scenario discussed in §1.3, the above oil-drilling scenario can be also modeled as Weitzman's "Pandora's box" problem [Wei79]. Although powerful, the basic Pandora's box model is limited assuming we find the exact amount of oil at a site using a *single* inspection. What if we need to perform multiple inspections at a site before finding X_i, where each inspection incurs a price and improves our estimate? A natural way to model this evolution of X_i is to use a Markov chain for each site, where each probe at a site incurs a price and results in a random transition in the chain. It is only when a Markov chain reaches one of its "destination" states that we find X_i. This model was used in [DTW03] to study a related *minimization* problem where the goal is to minimize the total price paid to set up an oil-drill, while ensuring that one of the sites reaches its destination state. Moreover, a very general model for the maximization problem was proposed by [KWW16, Appendix G], who gave an optimal algorithm to pick a single site.

The basic Pandora's box model was extended to richer combinatorial constraints in Chapter 3. We recollect the setting: given a packing combinatorial constraint $\mathcal{F} \subseteq 2^J$ over some ground elements J, the *price of information* model asks for a strategy to probe a set Probed $\subseteq J$ of r.v.s and return a subset $\mathbb{I} \subseteq$ Probed that is feasible according to the constraint (i.e., $\mathbb{I} \in \mathcal{F}$) and maximizes utility:

$$\mathbb{E}\Big[\underbrace{\textstyle\sum_{i\in\mathbb{I}} X_i}_{\text{value}} - \underbrace{\textstyle\sum_{i\in\text{Probed}} \pi_i}_{\text{total price}}\Big]. \tag{4.1}$$

In this chapter we propose the *Markovian Price of Information* (Markovian PoI) model that combines these two lines of inquiry: the Markovian and the combinatorial generalizations of the Pandora's box. We design optimal/approximation algorithms in the setting where the objective is to select a feasible set of elements that have reached their destination states, while minimizing the total prices paid in advancing the Markov chains to get to their destination states.

In the following sections, we first describe the basic Markovian PoI model in more detail and present our results. We then talk about extensions of the model to a robust variant, and to a model with commitments.

4.1.1 The Markovian Price of Information Model

To capture the Markovian evolution of a r.v. in examples like oil-drilling, we use the notion of a Markov system inspired by [DTW03] (who did not consider values at the destinations).

Definition 4.1.1 (Markov System)**.** *A Markov system $\mathcal{S} = (V, P, s, T, \boldsymbol{\pi}, \mathbf{r})$ for an element consists of a discrete Markov chain with state space V, a transition matrix $P = \{p_{u,v}\}$ indexed by $V \times V$ (here $p_{u,v}$ is the probability of transitioning from u to v), a starting state s, a set of absorbing* destination *states $T \subseteq V$, a non-negative probing price $\pi^u \in \mathbb{R}_{\geq 0}$ for every state $u \in V \setminus T$, and a value $r^t \in \mathbb{R}$ for each destination state $t \in T$. We assume that every state $u \in V$ reaches some destination state.*

We have a collection J of *ground elements*, each associated with its own Markov system. An element is *ready* if its Markov system has reached one of its absorbing destination states. For a ready element, if ω is the (random) *trajectory* of its Markov chain then its associated destination state is denoted by $d(\omega)$. We now define the Markovian PoI game, which consists of an objective function on the ground elements J.

Definition 4.1.2 (Markovian PoI Game)**.** *Given a set of ground elements J, constraints $\mathcal{F} \subseteq 2^J$, an objective function $f : 2^J \times \mathbb{R}^{|J|} \to \mathbb{R}$, and a Markov system $\mathcal{S}_i = (V_i, P_i, s_i, T_i, \boldsymbol{\pi}_i, \mathbf{r}_i)$ for each element $i \in J$, the* Markovian PoI *game is the following. At each time step, we either advance a Markov system $\mathcal{S}_i$ from its current state $u \in V_i \setminus T_i$ by incurring price π_i^u, or we end the game by selecting a subset of* ready *elements $\mathbb{I} \subseteq J$ that are* feasible*—i.e., such that $\mathbb{I} \in \mathcal{F}$.*

An interesting special case of the objective function f is when it is *additive*, i.e., $f(\mathbb{I}, \mathbf{x}) = \sum_{i \in \mathbb{I}} x_i$ for $\mathbb{I} \subseteq J$ and $\mathbf{x} \in \mathbb{R}^J$. This already captures our oil-drilling example where our objective is the sum of the values of each oil-drill that we set up.

Let $\boldsymbol{\omega}$ denote the *trajectory profile* for the Markovian PoI game: it consists of the random trajectories ω_i taken by all the Markov chains i at the end of the game. To avoid confusion, we write the selected feasible solution $\mathbb{I}$ as $\mathbb{I}(\boldsymbol{\omega})$. A utility/disutility optimization problem is to give a strategy for a Markovian PoI game while optimizing some combination of the objective and the total price.

Utility Maximization Problem: A Markovian PoI game where the family of constraints $\mathcal{F}$ are downward-closed (i.e., *packing*) and the values $\mathbf{r}_i$ are non-negative for every $i \in J$ (i.e., $\forall t \in T_i$, $r_i^t \geq 0$, and can be understood as a reward obtained by selecting element i). The goal is

to find a strategy ALG that maximizes the *utility*:

$$U^{\max}(\text{ALG}) \triangleq \mathbb{E}_{\boldsymbol{\omega}}\Big[\underbrace{f\left(\mathbb{I}(\boldsymbol{\omega}), \{r_i^{d(\omega_i)}\}_{i\in\mathbb{I}(\boldsymbol{\omega})}\right)}_{\text{value}} - \underbrace{\textstyle\sum_i \sum_{u\in\omega_i} \pi_i^u}_{\text{total price}}\Big]. \tag{4.2}$$

Notice that when f is additive and each Markov chain consists of only start and destination states, the above expression captures the PoI model from (4.1). Moreover, since the empty set is always feasible for a packing constraint, the optimal strategy has a non-negative utility.

We also define a minimization variant of the problem that is useful to capture covering combinatorial problems such as minimum spanning trees and set cover.

Disutility Minimization Problem: A Markovian PoI game where the family of constraints $\mathcal{F}$ are *upward-closed* (i.e., *covering*) and the values $\mathbf{r}_i$ are non-negative for every $i \in J$ (i.e., $\forall t \in T_i$, $r_i^t \geq 0$, and can be understood as a cost we need to pay for selecting element i). The goal is to find a strategy ALG that minimizes the *disutility*:

$$U^{\min}(\text{ALG}) \triangleq \mathbb{E}_{\boldsymbol{\omega}}\Big[f\left(\mathbb{I}(\boldsymbol{\omega}), \{r_i^{d(\omega_i)}\}_{i\in\mathbb{I}(\boldsymbol{\omega})}\right) + \textstyle\sum_i \sum_{u\in\omega_i} \pi_i^u\Big].$$

We will assume that the function f is non-negative when all $\mathbf{r}_i$ are non-negative. Hence, the disutility of the optimal policy is non-negative.

In the special case where all the Markov chains for a Markovian PoI game are formed by a *directed acyclic graph* (Dag), we call the corresponding utility optimization problem Dag-Utility Maximization or Dag-Disutility Minimization.

4.1.2 Our Results

In [Sin18], Frugal algorithms were introduced to capture the intuitive notion of "greedy" algorithms. There are many known optimal/approximation Frugal algorithms for classical packing and covering problems. E.g., optimal algorithms for matroids and $O(1)$-approx algorithms for matchings, vertex cover, and facility location. These Frugal algorithms have been designed in the traditional *free information* (Free-Info) setting, where each ground element has a deterministic fixed value or cost. Can we use them in the Markovian PoI world?

Our main contribution is a technique that adapts *any* Frugal algorithm to the Markovian PoI world, achieving the *same approximation ratio* as the original algorithm achieves in the Free-Info world. The result applies to "semiadditive" objectives, which include all additive objectives as well as non-additive objectives of problems like facility location and prize-collecting Steiner tree.

Theorem 4.1.3. *For a semiadditive objective function* val*, if there exists an α-approximation* Frugal *algorithm for a* Utility Maximization *problem over some packing constraints $\mathcal{F}$ in the Free-Info world, then there exists an α-approximation strategy for the corresponding* Utility Maximization *problem in the* Markovian PoI *world.*

We prove an analogous result for Disutility Minimization in Appendix 4.4.2.

The following corollaries immediately follow using the known Frugal algorithms (§3.4).

Corollary 4.1.4. *In the* Markovian PoI *world, we have:*

- *An optimal algorithm for both* Utility Maximization *and* Disutility Minimization *for matroids.*
- *A* 2*-approx for* Utility Maximization *for matchings and a* k*-approx for a* k*-system.*
- *A* $\min\{f, \log n\}$*-approx for* Disutility Minimization *for set-cover, where* f *is the maximum number of sets in which a ground element is present.*
- *A* 1.861*-approx for facility location and a* 3*-approx for prize-collecting Steiner tree.*

For instance, the multi-stage oil-drilling example given at the start of the introduction is a Utility Maximization problem with laminar matroid constraints: so we can solve it optimally with an adapted Frugal algorithm.

4.1.3 Our Techniques

We first sketch how a Frugal algorithm solves a packing problem in the Free-Info world; the story for a covering problem is very similar. Recall that in a Free-Info packing problem, the goal is to select a set $\mathbb{I}$ of elements to maximize our total value. A Frugal algorithm does this by repeatedly selecting elements to add to $\mathbb{I}$ one at a time. Specifically, for each element i, a Frugal algorithm computes a *marginal value* of i based on the element's value y_i and the already-selected elements. It then selects the element j of maximal marginal value. A key property of Frugal algorithms is that such selections are *irrevocable*: once an element is selected, it will certainly be in the output set M.

How might we take a Frugal algorithm, originally designed for the Free-Info world, and apply it to the Markovian PoI world? Our general approach is the following.

- Instead of using a fixed value y_i for element i, we use a *time-varying "proxy" value* that depends on the state of i's Markov chain.
- Instead of immediately selecting the element j of greatest marginal value, we *advance* j *one step*, selecting j only if it is in a destination state (i.e., ready).

This outline leaves us with an important question: what proxy value should we use in place of y_i for element i? Simple heuristics, such as using i's expected value minus expected probing price, are suboptimal even for very simple Markov systems and packing constraints.

The key to our adaptation of Frugal algorithms is to use the right proxy value for each element i in place of the fixed value y_i used in the Free-Info algorithm. This proxy is called the *grade*, written τ_i^u, for state u of Markov system $\mathcal{S}_i$. The adapted algorithm then has a very simple form:

> Pretend each element i has value y_i equal to its current grade τ_i^u. If the Frugal algorithm selects element i next, then either advance i one step if it is not ready, or select i if it is ready.

To define the grade of a state, we consider that Markov system in isolation. Roughly, the grade denotes the maximum penalty we can put at the destinations, while ensuring that it is optimal

to advance this Markov system at least once. (For those familiar with the literature, this grade is closely related to the well-known Gittins index.) While our definition of the grade is an extension of similar past definitions [DTW03, Web92], it was an open problem to combine these ideas with combinatorial constraints—indeed, it had been unclear what the right algorithm should be, and how to argue about such an algorithm. We manage to give a simple efficient algorithm for such a generalization. This is the main conceptual contribution of this part of this work.

4.1.4 Related Work

The work on multi-armed bandits originated in the scheduling literature. The Gittins index theorem [GJ74] provides a simple optimal strategy for several scheduling problems where the objective is to maximize the long-term exponentially discounted reward. This theorem turned out to be fundamental and [Tsi94, Web92, Whi80] gave alternate proofs. It can be also used to solve Weitzman's Pandora's box. The reader is referred to the book [GGW11] for further discussions on this topic. Influenced by this literature, [DTW03] studied scheduling of Markovian jobs, which is a minimization variant of the Gittins index theorem without any discounting. Their paper is part of the inspiration for our Markovian PoI model.

The Lagrangian variant considered in [GM07] is similar to our Markovian PoI model. However, their approach using an LP relaxation to design a probing strategy is fundamentally different from our approach using a Frugal algorithm. E.g., unlike Corollary 4.1.4, their approach cannot give *optimal* probing strategies for matroid constraints due to an integrality gap. Also, their approach does not work for Disutility Minimization.

Finally, as discussed in the introduction, the works in [KWW16] and [Sin18] are directly relevant to this chapter. The former's primary focus is on *single item* settings and its applications to auction design, and the latter studies price of information in a *single-stage* probing model. The contributions in this chapter concern selecting *multiple items* in *multi-stage* probing model, in some sense unifying these two lines of work.

Organization of the Chapter In §4.2, we explain the concepts of grade and prevailing cost that form the key to our arguments. Then in §4.3, we formally define a Frugal algorithm and show how to use it to obtain a good adaptive strategy for the Utility Maximization problem. The corresponding proofs for Disutility Minimization are in §4.4.

4.2 Grade and Prevailing Cost

Inspired by [DTW03], we define a *grade* for every state in a Markov system in §4.2.1. This grade is a variant of the popular *Gittins index*. In §4.2.2, we use the grade to define a *prevailing cost* and an *epoch* for a trajectory.

4.2.1 Grade of a State

To define the *grade* τ^v of a state $v \in V$ in Markov system $\mathcal{S} = (V, P, s, T, \boldsymbol{\pi}, \mathbf{r})$, we consider the following Markov game called *τ-penalized* $\mathcal{S}$, denoted $\mathcal{S}(\tau)$. Roughly, $\mathcal{S}(\tau)$ is the same as $\mathcal{S}$ but with a *termination penalty*, which is a constant $\tau \in \mathbb{R}$.

Suppose $v \in V$ denotes the current state of $\mathcal{S}$ in the game $\mathcal{S}(\tau)$. In each move, the player has two choices: (a) *Halt* that immediately ends the game, and (b) *Play* that changes the state, price, and value as follows:

- If $v \in V \setminus T$, the player pays price π^v, the current state of $\mathcal{S}$ changes according to the transition matrix P, and the game continues.
- If $v \in T$, then the player receives *penalized value* $r^v - \tau$, where τ is the aforementioned termination penalty, and the game ends.

The player wishes to maximize his *utility*, which is the expected value he obtains minus the expected price he pays. We write $U^v(\tau)$ for the utility attained by optimal play starting from state $v \in V$.

The utility $U^v(\tau)$ is clearly non-increasing in the penalty τ, and one can also show that it is continuous [DTW03, Section 4]. In the case of extremely large penalty $\tau \to +\infty$, it is optimal to halt immediately, achieving $U^v(\tau) = 0$. In the opposite extreme $\tau \to -\infty$, it is optimal to play until completion, achieving $U^v(\tau) \to +\infty$. Thus, as we increase τ from $-\infty$ to $+\infty$, the utility $U^v(\tau)$ that starts positive, becomes 0 at some critical value $\tau = \tau^v$. This critical value τ^v that depends on the starting state v is the *grade*.

Definition 4.2.1 (Grade of a state)**.** *The* grade *of a state v in Markov system $\mathcal{S}$ is*

$$\tau^v = \sup\{\tau \in \mathbb{R} \mid U^v(\tau) > 0\}.$$

This quantity is well-defined from the discussion above. For a Utility Maximization *problem, we write the grade of a state v in Markov system $\mathcal{S}_i$ corresponding to element i as τ_i^v.*

We emphasize the grade of a state only depends on its own Markov system and is independent of other Markov systems. Put another way, the grade of a state is the penalty τ that makes the player *indifferent* between halting and playing. Hence, starting with state v in game $\mathcal{S}(\tau)$:

- If $\tau < \tau^v$, it is optimal to play on the first move.
- If $\tau > \tau^v$, it is optimal to halt on the first move.
- If $\tau = \tau^v$, *either halting or playing is optimal* on the first move.

For example, the grade of a destination state $t \in T$ is $\tau^t = r^t$, because then both halting and playing yield zero utility.

It is possible to efficiently compute the grade of a state [DTW03, Section 7].

4.2.2 Prevailing Cost and Epoch

We now define a *prevailing cost* [DTW03] and an *epoch*. Given a trajectory ω, intuitively the prevailing cost denotes the maximum termination penalty that we can charge for the game

$\mathcal{S}(\tau)$ such that for every state along ω the player does not want to halt.

Definition 4.2.2 (Prevailing Cost for Utility Maximization)**.** *The* prevailing cost *of a trajectory ω_i of Markov system $\mathcal{S}_i$ for* Utility Maximization *is*

$$Y^{\max}_{\omega_i} \triangleq \min_{v \in \omega_i} \{\tau_i^v\}.$$

For a trajectory profile $\boldsymbol{\omega}$, let $Y^{\max}_{\boldsymbol{\omega}}$ denote the list of prevailing costs for each Markov system.

Observe that in the above definition, prevailing cost of a trajectory can only decrease as it extends further. In particular, it decreases whenever the Markov system reaches a state with grade smaller than each of the previously visited states. We can therefore view the prevailing cost as a non-increasing piecewise constant function of time. This motivates us to define an epoch.

Definition 4.2.3 (Epoch for Utility Maximization)**.** *An* epoch *for a trajectory ω is any maximal continuous segment of ω where the prevailing cost does not change.*

Since the grade can be computed efficiently, we can also compute the prevailing cost and epochs of a trajectory efficiently.

4.3 Adaptive Algorithms for Utility Maximization

In this section, we prove our main result that adapts a Frugal algorithm in Free-Info world to a probing strategy in the Markovian PoI world. We restate our theorem.

Theorem 4.1.3. *For a semiadditive objective function* val*, if there exists an α-approximation* Frugal *algorithm for a* Utility Maximization *problem over some packing constraints $\mathcal{F}$ in the Free-Info world, then there exists an α-approximation strategy for the corresponding* Utility Maximization *problem in the* Markovian PoI *world.*

The theorem concerns *semiadditive functions*, which are useful to capture non-additive objectives of problems like facility location and prize-collecting Steiner tree. We recollect their definition from §3.1.1.

Definition 4.3.1 (Semiadditive function)**.** *A function $f(\mathbb{I}, \mathbf{X}) : 2^J \times \mathbb{R}^{|J|} \to \mathbb{R}$ is* semiadditive *if there exists a function $h : 2^J \to \mathbb{R}$ such that*

$$f(\mathbb{I}, \mathbf{x}) = \sum_{i \in \mathbb{I}} x_i + h(\mathbb{I}).$$

All additive functions are semiadditive with $h(\mathbb{I}) = 0$ for all $\mathbb{I}$. To capture the facility location problem on a graph $G = (J, E)$ with metric (J, d), clients $C \subseteq J$, and facility opening costs $\mathbf{x} : J \to \mathbb{R}_{\geq 0}$, we can define $h(\mathbb{I}) = \sum_{j \in C} \min_{i \in \mathbb{I}} d(j, i)$. Note that h only depends on the identity of facilities $\mathbb{I}$ and not their opening costs.

The proof of Theorem 4.1.3 takes two steps (see Figure 4.1). In §4.3.1, we give a randomized reduction to upper bound the utility of the optimal strategy in the Markovian PoI world with the optimum value of a *surrogate problem* in the Free-Info world. Then, in §4.3.2, we show how to transform a Frugal algorithm into a strategy that achieves utility close to this upper bound.

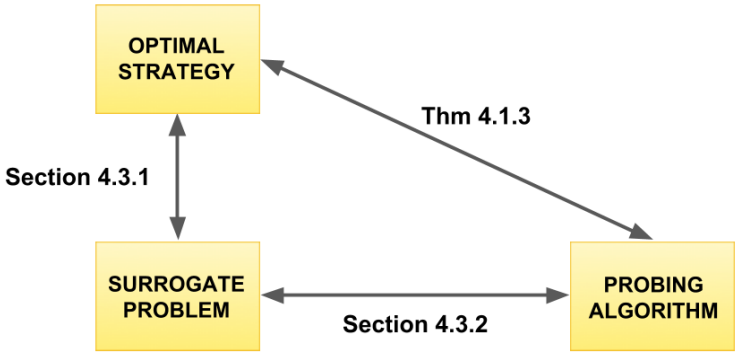

Figure 4.1: To prove Theorem 4.1.3, we first bound the optimal strategy using a surrogate problem in §4.3.1, and then obtain utility close to the surrogate by transforming the given FRUGAL algorithm to a probing algorithm in §4.3.2.

4.3.1 Upper Bounding the Optimal Strategy Using a Surrogate Problem

The *main idea* in this section is to show that for a UTILITY MAXIMIZATION problem, no strategy (in particular, optimal) can derive more utility from an element $i \in J$ than its prevailing cost. Here, the prevailing cost of i is for a random trajectory to a destination state in Markov system $\mathcal{S}_i$. Since the optimal strategy can only select a feasible set in $\mathcal{F}$, our main idea naturally leads to the following Free-Info *surrogate problem*: imagine each element's value is exactly its (random) prevailing cost, the goal is to select a set feasible in $\mathcal{F}$ to maximize the total value. In Lemma 4.3.3, we show that the expected optimum value of this surrogate problem is an upper bound on the optimum utility for UTILITY MAXIMIZATION. First, we formally define the surrogate problem.

Definition 4.3.2 (Surrogate Problem)**.** *Given a* UTILITY MAXIMIZATION *problem with semiadditive objective* val *and packing constraints* $\mathcal{F}$ *over universe* J*, the corresponding* surrogate *problem over* J *is the following. It consists of constraints* $\mathcal{F}$ *and (random) objective function* $\tilde{f} : 2^J \to \mathbb{R}$ *given by* $\tilde{f}(\mathbb{I}) = \text{val}(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega}))$*, where* $\mathbf{Y}^{\max}(\boldsymbol{\omega})$ *denotes the prevailing costs over a random trajectory profile* $\boldsymbol{\omega}$ *consisting of independent random trajectories for each element* $i \in J$ *to a destination state. The goal is to select* $\mathbb{I} \in \mathcal{F}$ *to maximize* $\tilde{f}(\mathbb{I})$*.*

Let $\text{SUR}(\boldsymbol{\omega}) \triangleq \max_{\mathbb{I}\in\mathcal{F}}\{\text{val}(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega}))\}$ denote the optimum value of this surrogate problem for trajectory profile $\boldsymbol{\omega}$. We now upper bound the optimum utility in the MARKOVIAN PoI world. Our proof borrows ideas from the "prevailing reward argument" in [DTW03].

Lemma 4.3.3. *For a* UTILITY MAXIMIZATION *problem with objective* val *and packing constraints* $\mathcal{F}$*, let OPT denote the utility of the optimal strategy. Then,*

$$OPT \le \mathbb{E}_{\boldsymbol{\omega}}[\text{SUR}(\boldsymbol{\omega})] = \mathbb{E}_{\boldsymbol{\omega}}\left[\max_{\mathbb{I}\in\mathcal{F}}\{\text{val}(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega}))\}\right],$$

where the expectation is over a random trajectory profile $\boldsymbol{\omega}$ *that has every Markov system reaching a destination state.*

Proof. We abuse the notation and use OPTto denote both the optimal policy and its utility. Suppose we fix a trajectory profile $\boldsymbol{\omega}$ where each Markov system $\mathcal{S}_i$ reaches a destination state. Let $\mathbb{I}(\boldsymbol{\omega})$ be the set of elements selected by OPTon $\boldsymbol{\omega}$, where notice that some of the unselected elements may not be ready: OPTmight have selected $\mathbb{I}(\boldsymbol{\omega})$ only after playing prefixes of trajectories in $\boldsymbol{\omega}$. The following observation follows from the definition of $\mathrm{SUR}(\boldsymbol{\omega})$.

Observation 4.3.4. *For any trajectory profile $\boldsymbol{\omega}$,*

$$\mathsf{val}(\mathbb{I}(\boldsymbol{\omega}), \mathbf{Y}^{\max}(\boldsymbol{\omega})) \leq \mathrm{SUR}(\boldsymbol{\omega}).$$

Now, the following Claim 4.3.5, along with Observation 4.3.4, implies Lemma 4.3.3.

Claim 4.3.5. *The utility of the optimal strategy*

$$OPT \leq \mathbb{E}_{\boldsymbol{\omega}}\left[\mathsf{val}(\mathbb{I}(\boldsymbol{\omega}), Y^{\max}(\boldsymbol{\omega}))\right]. \qquad \square$$

Proof of Claim 4.3.5. Since for every trajectory profile $\boldsymbol{\omega}$ both OPT in the Markovian PoI world and $\mathbb{E}_{\boldsymbol{\omega}}\left[\mathsf{val}(\mathbb{I}(\boldsymbol{\omega}), Y^{\max}(\boldsymbol{\omega}))\right]$ in the Free-Info world pick the same set of elements $\mathbb{I}(\boldsymbol{\omega})$, the expected value due to the set function h is the same. Hence, WLOG assume $h(\mathbb{I}) = 0$ for all $\mathbb{I} \in \mathcal{F}$.

Now consider the following *teasing game* G_T defined using the prevailing cost from Definition 4.2.2. Consider a game where each Markov system $\mathcal{S}_i$ starts at its initial state s_i and a player is invited to advance the Markov systems. Besides advancing, the player is allowed to select any arbitrary elements (need not be feasible in $\mathcal{F}$) or terminate the game at any time during the game. Whenever an element i is selected, the player pays a corresponding *cost*, which is set to be the prevailing cost defined by the trajectory that lead to the current state in $\mathcal{S}_i$. The player's goal is to maximize the expected *value*, which is the expected utility (as defined for Utility Maximization) from advancing the Markov systems *minus* the expected total cost he pays when some items are selected. Observe that in this game the costs are updated in a "teasing" manner according to the prevailing costs that motivates the player to continue playing. By an argument similar to [DTW03], we have the following claim.

Claim 4.3.6. *The teasing game G_T is* fair*, which means that no strategy achieves a positive expected value by playing it and that there exists a strategy with zero expected value. Moreover, the following strategy plays fairly: irrespective of the order in which the Markov systems are played, whenever the player starts to advance a Markov system, he continues to advance it through the entire epoch.*

Now consider running the optimal policy OPT in the teasing game. Let $\boldsymbol{\omega}$ be a trajectory profile in which each chain reaches its destination state. Let $\boldsymbol{\omega}_T$ denote a trajectory profile until the moment when OPT returns the solution $\mathbb{I}(\boldsymbol{\omega})$ on the trajectory profile $\boldsymbol{\omega}$. It should be noticed that each trajectory in $\boldsymbol{\omega}_T$ is a prefix of the corresponding trajectory in $\boldsymbol{\omega}$. In particular, for an element $i \in \mathbb{I}(\boldsymbol{\omega})$, ω_i coincides with $(\boldsymbol{\omega}_T)_i$ since the destination state of $\mathcal{S}_i$ is reached. For an element $i \notin \mathbb{I}(\boldsymbol{\omega})$, however, $(\boldsymbol{\omega}_T)_i$ may only be a prefix of ω_i. It follows that applying OPT in G_T along trajectory profile $\boldsymbol{\omega}$ incurs a cost of $\sum_{i \in \mathbb{I}(\boldsymbol{\omega})} Y^{\max}_{(\boldsymbol{\omega}_T)_i}$, where $Y^{\max}_{(\boldsymbol{\omega}_T)_i}$ is the prevailing cost

for $\mathcal{S}_i$ on trajectory $(\boldsymbol{\omega}_T)_i$ according to Definition 4.2.2. Since G_T is a fair game, the expected utility of OPT cannot be larger than the expected cost it pays, i.e.,

$$\mathrm{OPT} \le \mathbb{E}_{\boldsymbol{\omega}}\Big[\sum_{i\in\mathbb{I}(\boldsymbol{\omega})} Y^{\max}_{(\boldsymbol{\omega}_T)_i}\Big].$$

Since the elements $i \in \mathbb{I}(\boldsymbol{\omega})$ are ready, we have $\omega_i = (\boldsymbol{\omega}_T)_i$ and

$$\sum_{i\in\mathbb{I}(\boldsymbol{\omega})} Y^{\max}_{(\boldsymbol{\omega}_T)_i} = \sum_{i\in\mathbb{I}(\boldsymbol{\omega})} Y^{\max}_{\omega_i}.$$

This implies

$$\mathrm{OPT} \le \mathbb{E}_{\boldsymbol{\omega}}\Big[\sum_{i\in\mathbb{I}(\boldsymbol{\omega})} Y^{\max}_{\omega_i}\Big],$$

which finishes the proof of Claim 4.3.5. □

4.3.2 Designing an Adaptive Strategy Using a Frugal Algorithm

We use the same definition of a Frugal packing algorithm as in [Sin18]. A Frugal algorithm selects elements one-by-one and irrevocably. Besides greedy algorithms, its definition also captures "non-greedy" algorithms such as primal-dual algorithms that do not have the reverse-deletion step.

Definition 4.3.7 (Frugal Packing Algorithm)**.** *For a combinatorial optimization problem on universe J in the Free-Info world with packing constraints $\mathcal{F} \subseteq 2^J$ and objective $f : 2^J \to \mathbb{R}$, we say Algorithm $\mathcal{A}$ is* Frugal *if there exists a* marginal-value *function $g(\mathbf{Y}, i, y) : \mathbb{R}^J \times J \times \mathbb{R} \to \mathbb{R}$ that is increasing in y, and for which the pseudocode is given by Algorithm 6. Note that this algorithm always returns a feasible solution if $\emptyset \in \mathcal{F}$.*

Algorithm 6 Frugal Packing Algorithm $\mathcal{A}$

1: Start with $M = \emptyset$ and $v_i = 0$ for each element $i \in J$.
2: For each element $i \notin M$, compute $v_i = g(\mathbf{Y}_M, i, Y_i)$. Let $j = \mathrm{argmax}_{i\notin M \ \&\ M\cup i\in\mathcal{F}}\{v_i\}$.
3: If $v_j > 0$ then add j into M and go to Step 2. Otherwise, return M.

The following lemma shows that a Frugal algorithm can be converted to a strategy with the same utility in the Markovian PoI world.

Lemma 4.3.8. *Given a* Frugal *packing Algorithm $\mathcal{A}$, there exists an adaptive strategy $\mathrm{ALG}_{\mathcal{A}}$ for the corresponding* Utility Maximization *problem in the* Markovian PoI *world with utility at least*

$$\mathbb{E}_{\boldsymbol{\omega}}\Big[\mathsf{val}(\mathcal{A}(\mathbf{Y}^{\max}(\boldsymbol{\omega})), \mathbf{Y}^{\max}(\boldsymbol{\omega}))\Big],$$

where $\mathcal{A}(\mathbf{Y}^{\max}(\boldsymbol{\omega})$ is the solution returned by $\mathcal{A}$ for objective $f(\mathbb{I}) = \mathsf{val}(\mathbf{Y}^{\max}(\boldsymbol{\omega}), \mathbb{I})$.

Proof. We describe how to adapt the Frugal Algorithm $\mathcal{A}$ to an adaptive strategy $\text{ALG}_{\mathcal{A}}$ in the Markovian PoI world. $\text{ALG}_{\mathcal{A}}$ uses the grade τ as proxy for $\mathbf{Y}^{\max}$, since $\mathbf{Y}^{\max}$ is known only when the Markov systems reach their destination states. More specifically, at each moment when the Frugal Algorithm $\mathcal{A}$ is trying to evaluate the marginal-value function for each element, instead of using the $\mathbf{Y}^{\max}$ value for each element, which we may not yet know at the moment, the strategy uses the τ values to compute the marginal. For the element chosen by $\mathcal{A}$, the corresponding Markov system will be advanced one more step. A more specific description of our algorithm $\text{ALG}_{\mathcal{A}}$ is given Algorithm 7. Here $\mathbf{Y}^{\max}_{M}$ for a set $M \subseteq J$ is defined as the list of $\mathbf{Y}^{\max}$ values that are in the set M.

Algorithm 7 Algorithm $\text{ALG}_{\mathcal{A}}$ for Utility Maximization in Markovian PoI

1: Start with $M = \emptyset$ and $v_i = 0$ for all elements i.
2: For each element $i \notin M$, set $g(\mathbf{Y}^{\max}_{M}, i, \tau_i^{u_i})$ where u_i is the current state of i.
3: Consider the element $j = \text{argmax}_{i \notin M \ \&\ M \cup i \in \mathcal{F}}\{v_i\}$.
4: If $v_j > 0$, then if $\mathcal{S}_j$ is not in a destination state then proceed $\mathcal{S}_j$ by one step and go to Step 2. Else, when $v_j > 0$ but $\mathcal{S}_j$ is in a destination state t_j, select j into M and go to Step 2.
5: Else, if every element $i \notin M$ has $v_i \leq 0$ then return set M.

In the following Claim 4.3.9, we argue that for any trajectory profile $\boldsymbol{\omega}$, running $\text{ALG}_{\mathcal{A}}$ in Markovian PoI returns the same set of elements as running $\mathcal{A}$ for $\mathbf{Y}^{\max}(\boldsymbol{\omega})$.

Claim 4.3.9. *For any trajectory profile $\boldsymbol{\omega}$, the solution returned by running Algorithm 7 in the* Markovian PoI *world is the same as the solution by Algorithm $\mathcal{A}$ on $\mathbf{Y}^{\max}(\boldsymbol{\omega})$.*

Before proving Claim 4.3.9, we use it to prove Lemma 4.3.8 by showing that the utility of Algorithm 7 in the Markovian PoI world is at least

$$\mathbb{E}_{\boldsymbol{\omega}}\Big[\text{val}(\mathcal{A}(\mathbf{Y}^{\max}(\boldsymbol{\omega})), \mathbf{Y}^{\max}(\boldsymbol{\omega}))\Big].$$

By Claim 4.3.9, the value due to the set function h is the same for both algorithms. So without loss of generality, assume h is always 0. We consider the teasing game G_T as defined in Claim 4.3.6. By definition, g is an increasing function of the last parameter y. Since grade is used as that parameter and the grade of each state visited during an epoch is at least the grade of the initial state of that epoch, it follows that once Algorithm 7 starts to play a Markov system $\mathcal{S}_i$, it will not switch before finishing an epoch. Therefore, by Claim 4.3.6, Algorithm 7 plays a fair game. So the expected cost that Algorithm 7 pays is the same as its expected utility from playing the Markov systems. However, Claim 4.3.9 gives the expected cost payed by Algorithm 7 is the same as the utility of running Algorithm $\mathcal{A}$ in the Free-Info world, i.e., $\mathbb{E}_{\boldsymbol{\omega}}[\text{val}(\mathcal{A}(\mathbf{Y}^{\max}(\boldsymbol{\omega})), \mathbf{Y}^{\max}(\boldsymbol{\omega}))]$. Hence, the utility of running Algorithm 7 at least $\mathbb{E}_{\boldsymbol{\omega}}[\text{val}(\mathcal{A}(\mathbf{Y}^{\max}(\boldsymbol{\omega})), \mathbf{Y}^{\max}(\boldsymbol{\omega}))]$. □

Proof of Theorem 4.1.3. From Lemma 4.3.8, we know that the utility of $\text{ALG}_{\mathcal{A}}$ is at least

$$\mathbb{E}_{\boldsymbol{\omega}}\Big[\text{val}(\mathcal{A}(\mathbf{Y}^{\max}(\boldsymbol{\omega})), \mathbf{Y}^{\max}(\boldsymbol{\omega}))\Big].$$

Since Algorithm $\mathcal{A}$ is an α-approximation algorithm in the Free-Info world, it follows

$$\mathbb{E}_{\boldsymbol{\omega}}[\mathrm{val}(\mathcal{A}(\mathbf{Y}^{\max}(\boldsymbol{\omega})), \mathbf{Y}^{\max}(\boldsymbol{\omega}))] \geq \frac{1}{\alpha} \cdot \mathbb{E}_{\boldsymbol{\omega}} \left[\max_{\mathbb{I} \in \mathcal{F}} \{\mathrm{val}(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega}))\} \right].$$

Now using the upper bound on the optimal utility $\mathrm{OPT} \leq \mathbb{E}_{\boldsymbol{\omega}} [\max_{\mathbb{I} \in \mathcal{F}} \{\mathrm{val}(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega}))\}]$ from Lemma 4.3.3, we have utility of $\mathrm{ALG}_{\mathcal{A}}$ is at least $\frac{1}{\alpha} \cdot \mathrm{OPT}$. □

It remains to prove the missing Claim 4.3.9 in the proof of Lemma 4.3.8.

Proof of Claim 4.3.9. Suppose we fix a trajectory profile $\boldsymbol{\omega}$ where each Markov system reaches some destination state. We prove the claim by induction on the number of elements already selected into the set M. Suppose the set of elements selected into M is the same by running the two algorithms until now. We show that the next element selected by the algorithms into M is the same.

Assume for the purpose of contradiction that the next element picked by $\mathcal{A}$ is j but the next element picked by Algorithm 7 is $i \neq j$. By the definition of Algorithm $\mathcal{A}$,

$$j = \mathrm{argmax}_{i' \notin M} \left\{ g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), i', Y_{\omega_{i'}}^{\max}\right) \right\}. \tag{4.3}$$

where ω_i' denotes the trajectory of $\mathcal{S}_{i'}$ in $\boldsymbol{\omega}$. Now we look at the trajectory ω_i, it follows that the prevailing cost $Y_{\omega_i}^{\max}$ is non-increasing over this trajectory and is equal to $Y_{\omega_i}^{\max}$ when $\mathcal{S}_i$ reaches the destination state. We look at the last moment t_0 when the prevailing cost of $\mathcal{S}_i$ decreases. Consider the first moment t_1 after t_0 that our Algorithm 7 decides to play $\mathcal{S}_i$ (but has not actually played $\mathcal{S}_i$ yet). It follows that the prevailing cost of $\mathcal{S}_i$ at moment t_1 is exactly the same as $Y_{\omega_i}^{\max}$ and also the grade $\tau_i^{u_i}$ of the current state u_i. Denote $Y_{\omega_j'}^{\max}$ the prevailing cost of $\mathcal{S}_j$ and u_j the state of $\mathcal{S}_j$ at moment t_1. Then we have $Y_{\omega_j'}^{\max} \geq Y_{\omega_j}^{\max}$ because the prevailing cost of $\mathcal{S}_j$ is also non-increasing. By the definition of t_1, one has

$$g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), i, Y_{\omega_i}^{\max}\right) = g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), i, \tau_i^{u_i}\right) > g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), j, \tau_j^{u_j}\right) \geq g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), j, Y_{\omega_j'}^{\max}\right).$$

However, since g is increasing in the last parameter, it follows that

$$g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), j, Y_{\omega_j'}^{\max}\right) \geq g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), j, Y_{\omega_j}^{\max}\right),$$

which implies

$$g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), i, Y_{\omega_i}^{\max}\right) > g\left(\mathbf{Y}_M^{\max}(\boldsymbol{\omega}), j, Y_{\omega_j}^{\max}\right).$$

This contradicts with the definition of j in Eq (4.3). □

In Appendix 4.4.2, a similar approach is used for the Disutility Minimization problem with semi-additive function. We conclude that for either Utility Maximization or Disutility Minimization problem with semi-additive function, any Frugal approximation algorithm in the Free-Info world can be transformed into an approximation strategy with the same approximation ratio in the Markovian PoI world.

4.4 Illustrative Examples and Missing Proofs

4.4.1 Comparing Grade and Weitzman's Index for Pandora's Box

Recall Weitzman's Pandora's box formulation of the oil-drilling problem mentioned in §1. Given probability distributions of n independent random variables X_i (amount of oil at site i) and their *probing* (inspection) prices π_i, the goal is to design a strategy to *adaptively* probe a set Probed to maximize expected utility

$$\mathbb{E}\Big[\max_{i\in\text{Probed}}\{X_i\} - \sum_{i\in\text{Probed}}\pi_i\Big].$$

The Weitzman's index for site i, denoted by $\tau_i^{\max}$, is defined using the following equation $\mathbb{E}[(X_i - \tau_i^{\max})^+] = \pi_i$. It is known that the following strategy is optimal [Wei79].

Selection Rule: The next site to be probed is the one with with the highest Weitzman's index.

Stopping Rule: Terminate when the maximum realized value amongst the probed sites exceeds the Weitzman's index of every unprobed site.

It turns out that Weitman's index $\tau_i^{\max}$ is simply the grade, defined in §4.2.1, in disguise. To see this, we start by noticing that each variable X_i with probing price π_i can be thought of as the following Markov system. There is one initial state s_i with moving cost π_i. s_i has transitions, with probabilities according to the distribution of X_i, to a set T_i of destination states, each corresponding to a possible outcome of the variable X_i . The value of each destination state is naturally set to be the corresponding outcome of X_i. We show below that $\tau_i^{\max}$ is simply the grade $\tau_i^{s_i}$ of the initial state s_i.

According to our definition of grade in §4.2.1, in the $\tau_i^{s_i}$-penalized Markov game $\mathcal{S}(\tau_i^{s_i})$, there is a fair strategy that probes site i and achieves a zero utility. Such a strategy would pick site i (i.e., play in the corresponding destination state) if and only if $X_i - \tau_i^{s_i} \geq 0$. The utility of that policy is thus $-\pi_i + \mathbb{E}[(X_i - \tau_i^{s_i})^+] = 0$. Comparing with the definition of Weitzman's index, this shows $\tau_i^{\max} = \tau_i^{s_i}$. The optimality of Weitzman's strategy is therefore also implied by Theorem 4.1.3.

4.4.2 Adaptive Algorithms for Disutility Minimization

We give the corresponding definitions for the Disutility Minimization problem.

Definition 4.4.1 (Prevailing Reward for Disutility Minimization)**.** *The* prevailing reward *of $\mathcal{S}_i$ for the trajectory P_i in* Disutility Minimization *is defined as*

$$R_{P_i}^{\min} \triangleq \max_{u\in P_i}\{-\tau_i^u\}.$$

For a trajectory profile $\boldsymbol{\omega}$, denote $R_{\boldsymbol{\omega}}^{\min}$ the list of prevailing rewards for each Markov system.

For a trajectory P_i in the Disutility Minimization problem, consider the change of the prevailing reward as the Markov system starts from s_i and moves according to P_i. It follows

that the prevailing reward is non-decreasing in this process. Moreover, it increases whenever the Markov system reaches a state that has smaller grade than each previously visited state. Now we are ready to state the definition of an *epoch*.

Definition 4.4.2 (Epoch for Disutility Minimization)**.** *An* epoch *is defined to be the period from the time when the prevailng reward increases until the moment just before the next time it increases.*

It follows that within an epoch, all states visited has grade no smaller than the prevailing reward at the start of this epoch and thus the prevailing reward stays constant in an epoch. We can therefore view the prevailing reward as a non-decreasing piece-wise constant function of time.

Definition 4.4.3 (Frugal Covering Algorithm)**.** *For a* Disutility Minimization *problem in the* Deterministic *world with covering constraints $\mathcal{F}$ and cost function* cost*, we say Algorithm $\mathcal{A}$ is* Frugal *if there exists a* marginal-value *function $g(\mathbf{Y}, i, y) : \mathbb{R}^J \times J \times \mathbb{R} \to \mathbb{R}$ that is decreasing in y, and for which the pseudocode is given by Algorithm 8. Moreover, the function $g(\mathbf{Y}, i, y)$ should* encode *the constraints $\mathcal{F}$, such that whenever M is infeasible, then $\exists i \notin M$ with $v_i > 0$. This requirement will ensure that a feasible solution is returned.*

Algorithm 8 Frugal Covering Algorithm $\mathcal{A}$

1: Start with $M = \emptyset$ and $v_i = 0$ for each element $i \in J$.
2: For each element $i \notin M$, compute $v_i = g(\mathbf{Y}_M, i, Y_i)$. Let $j = \text{argmax}_{i \notin M}\{v_i\}$.
3: If $v_j > 0$ then add j into M and go to Step 2. Otherwise, return M.

With the definitions above, one can prove the following theorem for Disutility Minimization using similar techniques as in §4.3.

Theorem 4.4.4. *For a semiadditive objective function* cost*, if there exists an α-approximation* Frugal *algorithm for a* Disutility Minimization *problem over some covering constraints $\mathcal{F}$ in the Free-Info world, then there exists an α-approximation strategy for the corresponding* Disutility Minimization *problem in the* Markovian PoI *world.*

Chapter 5

Constrained Stochastic Probing via Adaptivity Gaps

5.1 Introduction

Consider the *best-box* problem: Suppose there are n independent Bernoulli random variables where X_i takes value v_i with probability p_i and is zero otherwise. Given a probing price π_i for each X_i and an overall probing budget B, design a strategy to *adaptively* probe a subset of variables Probed $\subseteq [n]$ within our budget, i.e., $\sum_{i \in \text{Probed}} \pi_i \leq B$, while maximizing the expected value of the best variable that we probe:

$$\mathbb{E}\left[\max_{i \in \text{Probed}} \{X_i\}\right].$$

As discussed in §1.3.1, the best-box problem can be used to model the House-Purchasing scenario. In this chapter we will always assume that our random variables are Bernoulli. In Chapter 6, we extend this model to capture more general random variables. Now suppose instead of finding the best box, what if we are interested in maximizing the sum of the top k boxes that we probe. In fact, what if our objective is given by an uncertain submodular function. Submodular maximization has been a very useful abstraction for many problems, both theoretical (e.g., the classical k-coverage problem [WS11]) and practical (e.g., the influence maximization problem [KKT15], or many problems in machine learning [Kra13]). We know how to perform constrained submodular maximization both when the function is monotone [FNW78, CCPV11] and when it's non-monotone [FMV11, LMNS09, FNS11b]. How to model and solve the problem of maximizing an uncertain submodular function over a packing constraint?

Consider the following setting. We have a submodular function over a ground set of elements V (e.g., the max value function). But the elements are not all *active*, and we can get value only for active elements. The bad news is that *a priori* we don't know the elements' status—whether it is active or not. The good news is that each element e is active independently with some known probability p_e. We find out an element e's status only by *probing* it. Once we know its status, we can use this information to decide which other elements to probe next, and in what order; i.e., be *adaptive*. We have some constraints on which subsets we are allowed to

probe (e.g., knapsack constraints). Eventually, we stop with some probed set S and a known subset active(S) of the active elements in S. At that time we can pick any $T \subseteq$ active(S) and get value $f(T)$.[1] What is a good strategy to probe elements to maximize the expected value?

As another example, consider the setting of influence maximization, the ground set is a set of email addresses (or Facebook accounts), and for a set S of email addresses $f(S)$ is the fraction of the network that can be influenced by seeding the set S. But not all email addresses are still active. For each email address e, we know the probability p_e that it is active. (Based, e.g., on when the last time we know it was used, or some other machine learning technique.) Now due to time constraints, or our anti-spam policies, or the fact that we are risk-averse and do not want to make introductory offers to too many people—we can only probe some K of these addresses, and make offers to the active ones in these K, to maximize our expected influence. Observe that it makes sense to be adaptive—if `t.theorist@cs.cmu.edu` happens to be active we may not want to probe `t.theorist@cmu.edu`, since we may believe they are the same person.

There are other examples: e.g., Bayesian mechanism design problems (see [GN13] for details), stochastic matching problems in kidney exchange and online dating [CIK+09, BGL+12], and stochastic set cover problems that arise in database applications [LPRY08, DHK14].

The question that is of primary interest to us is the following: *Even though our model allows for adaptive queries, what is the benefit of this adaptivity?* Note that there is price to adaptivity: the optimal adaptive strategy may be an exponentially-large decision tree that is difficult to store, and also may be computationally intractable to find. Moreover, in some cases the adaptive strategy would require us to be sequential (probe one email address, then probe the next, and so on), whereas a non-adaptive strategy may be just a set of K addresses that we can probe in parallel. So we want to bound the *adaptivity gap*: the ratio between the expected value of the best adaptive strategy and that of the best non-adaptive strategy. Secondly, if this adaptivity gap is small, we would like to find the best non-adaptive strategy efficiently (in polynomial time). This would give us our ideal result: a non-adaptive strategy that is within a small factor of the best adaptive strategy.

The goal of this work is to get such results for as broad a class of functions, and as broad a class of probing constraints as possible [GNS17, GNS16]. Recall that we are not allowed to probe all the elements, but only those that satisfy some problem-specific constraints (e.g., probe at most K email addresses, or probe a set of locations that can be reached using a path of length at most D.)

5.1.1 Model and Results

In this paper, we allow very general probing constraints: the sequence of elements we probe must satisfy a given *prefix-closed constraint*—e.g., these may be given by a matroid, or an orienteering constraint, or deadline, or precedence constraint, or an arbitrary downward-closed constraint—if we can probe some sequence of elements we can probe any prefix of it. Simple examples show that we cannot hope to get small adaptivity gaps for arbitrary functions even

[1] If the function is monotone, clearly we should choose T = active(S).

for a cardinality constraint, and hence we have to look at interesting sub-classes of functions.

Our first set of results are for the case where the function f is a non-negative submodular function. The first result is for monotone functions.

Theorem 5.1.1 (Monotone Submodular)**.** *For any monotone non-negative submodular function f and any prefix-closed probing constraints, the stochastic probing problem has adaptivity gap at most* 3*.*

The previous results in this vein severely restricted the probing constraints (e.g., Asadpour et al. [AN16] gave a gap of $\frac{e}{e-1}$ for matroid probing constraints). We discuss these and other prior works in §5.1.3. There is a *lower bound* of $\frac{e}{e-1}$ on the adaptivity gap for monotone submodular functions with prefix-closed probing constraints (in fact for the rank function of a partition matroid, with the constraint being a simple cardinality constraint). It remains an interesting open problem to close this gap.

We then turn to non-monotone submodular functions, and again give a constant adaptivity gap. While the constant can be improved slightly, we have not tried to optimize it, focusing instead on clarity of exposition.

Theorem 5.1.2 (Non-Monotone Submodular)**.** *For any non-negative submodular function f and prefix-closed probing constraints, the stochastic probing problem has adaptivity gap at most* 40*.*

Both Theorems 5.1.1 and 5.1.2 just consider the adaptivity gap. What about the computational question of finding the best non-adaptive strategy? This is where the complexity of the prefix-closed constraints come in. The problem of finding the best non-adaptive strategy with respect to some prefix-closed probing constraint can be reduced to the deterministic problem of maximizing a submodular function with respect to the same constraints. So we can use existing results on (deterministic) submodular maximization.

5.1.2 Techniques

Before talking about our techniques, a word about previous approaches to bounding the adaptivity gap. Several works, starting with the work of Dean et al. [DGV04] have used geometric "relaxations" (e.g., a linear program for linear functions [DGV04], or the multilinear extension for submodular settings [ANS08, ASW14]) to get an estimate of the value achieved by the optimal adaptive strategy. Then one tries to find a non-adaptive strategy whose expected value is not much less than this relaxation. This is particularly successful when the probing constraints are amenable to being captured by linear programs—e.g., matroid or knapsack constraints. Dealing with general constraints means we cannot use this approach.

The other approach is to argue about the optimal decision-tree directly. An induction on the tree was used, e.g., by Chen et al. [CIK+09] and Adamczyk [Ada11] to study stochastic matchings. A different approach is to use concentration bounds like Freedman's inequality to show that for most paths down the tree, the function on the path behaves like the path-mean [GKNR12]. However, this approach seems best suited to linear functions, and loses logarithmic factors due to the need for union bounds.

Given that we prove a general result for any submodular function, how do we show a good

non-adaptive strategy? Our approach is to take a random path down the tree (the randomness coming from the element activation probabilities) and to show the expected value of this path, when viewed as a non-adaptive strategy, to be good. To prove this, we use induction. An inductive approach is surprising because the objective is not additive over the nodes of the tree, and the natural induction down the tree does not seem to work. Instead, we perform a non-standard inductive argument, where we consider the all-no path (which we call the *stem*), show that a non-adaptive strategy would get value comparable to the decision tree on the stem, and then induct on the subtrees hanging off this stem. The proof for monotone functions, though basic, is subtle—requiring us to change representations and view things "right".

5.1.3 Related Work

The adaptivity gap of stochastic packing problems has seen much interest: e.g., for knapsack [DGV04, BGK11, Ma14], packing integer programs [DGV05, CIK+09, BGL+12], budgeted multi-armed bandits [GM07, GKMR11, LY13, Ma14] and orienteering [GM09, GKNR12, BN14]. All except the orienteering results rely on having relaxations that capture the constraints of the problem via linear constraints. A recent paper also designs a PTAS for the best-box problem [HFX18].

For stochastic monotone submodular functions where the probing constraints are given by matroids, Asadpour et al. [AN16] bounded the adaptivity gap by $\frac{e}{e-1}$; Hellerstein et al. [HKL15] bound it by $\frac{1}{\tau}$, where τ is the smallest probability of some set being materialized. Other relevant papers are [LPRY08, DHK14].

The work of Chen et al. [CIK+09] (see also [Ada11, BGL+12, BCN+15, AGM15]) sought to maximize the size of a matching subject to b-matching constraints; this was motivated by applications to online dating and kidney exchange. More generally, see, e.g. [RSÜ05, AR12], for pointers to other work on kidney exchange problems. The work of [GN13] abstracted out the general problem of maximizing a function (in their case, the rank function of the intersection of matroids or knapsacks) subject to probing constraints (again, intersection of matroids and knapsacks). This was improved and generalized by Adamczyk, et al. [ASW14] to submodular objectives. All these results use LP relaxations, or non-linear geometric relaxations for the submodular settings.

5.2 Adaptive Strategies and Notation

We denote the ground set by V, with $n = |V|$. Each element $e \in V$ has an associated probability p_e. Given subset $S \subseteq V$ and vector $\mathbf{p} = (p_1, p_2, \ldots, p_n)$, let $S(\mathbf{p})$ denote the distribution over subsets of S obtained by picking each element $e \in S$ independently with probability p_e. Specifying a single number $p \in [0,1]$ in $S(p)$ indicates each element is chosen with the same probability p.

All objective functions f that we deal with are non-negative with $f(\emptyset) = 0$. Given any function $f : 2^V \to \mathbb{R}$, define $f^{\max}(S) := \max_{T \subseteq S} f(T)$ to be the maximum value subset contained within S. The function f is monotone if and only if $f^{\max} = f$. In general, $f^{\max}$ may be difficult to

compute given access to f. However, Feige et al. [FMV11] show that for submodular functions

$$\frac{1}{4}f^{\max}(S) \quad \leq \quad \mathbb{E}_{R\sim S(\frac{1}{2})}[f(R)] \quad \leq \quad f^{\max}(S). \tag{5.1}$$

Also, for a subset S, define the "contracted" function $f_S(T) := f(S \cup T) - f(S)$. Note that if f is non-monotone, then f_S may take negative values even if f is non-negative.

An adaptive strategy tree $\mathcal{T}$ is a binary rooted tree where every internal node v represents some element $e \in V$ (denoted by $\mathsf{elt}(v) = e$), and has two outgoing arcs—the `yes` arc indicating the node to go to if the element $e = \mathsf{elt}(v)$ is active (which happens with probability p_e), and the `no` arc indicating the node to go to if e is not active (which happens with the remaining probability $q_e = 1 - p_e$). No element can be represented by two different nodes on any root-leaf path. Moreover, any root-leaf path in T should be feasible according to the constraints. Hence, each leaf ℓ in the tree $\mathcal{T}$ is associated with the root-leaf path P_ℓ: the elements probed on this path are denoted by $\mathsf{elt}(P_\ell)$. Let A_ℓ denote the active elements on this path P_ℓ—i.e., the elements represented by the nodes on P_ℓ for which we took the `yes` arc.

The tree $\mathcal{T}$ naturally gives us a probability distribution $\pi_{\mathcal{T}}$ over its leaves: start at the root, at each node v, follow the `yes` branch with probability $p_{\mathsf{elt}(v)}$ and the "no" branch otherwise, to end at a leaf.

Given a submodular function f and a tree $\mathcal{T}$, the associated adaptive strategy is to probe elements until we reach a leaf ℓ, and then to pick the max-value subset of the active elements on this path P_ℓ. Let $\mathsf{adap}(\mathcal{T}, f)$ denote the expected value obtained this way; it can be written compactly as

$$\mathsf{adap}(\mathcal{T}, f) := \mathbb{E}_{\ell\leftarrow\pi_{\mathcal{T}}}[f^{\max}(A_\ell)]. \tag{5.2}$$

Definition 5.2.1 (stem of $\mathcal{T}$)**.** *For any adaptive strategy tree $\mathcal{T}$ the* stem *represents the all-`no` path in $\mathcal{T}$ starting at the root, i.e., when all the probed elements turn out inactive.*

Definition 5.2.2 (deepness of $\mathcal{T}$)**.** *The* deepness *of a strategy tree $\mathcal{T}$ is the maximum number of* active *nodes that* adap *sees along a root-leaf path of $\mathcal{T}$.*

Note that the notion of deepness is different from the (more standard) "depth" used for trees. Deepness measures the number of `yes`-arcs on the path from the root to the leaf, rather than the number of arcs seen on the path (which is the depth). This definition is inspired by the induction we will do in the submodular sections.

We also define the *natural non-adaptive* algorithm given the tree $\mathcal{T}$: just pick a leaf $\ell \leftarrow \pi_{\mathcal{T}}$ from the distribution given by $\mathcal{T}$, probe all elements on that path, and choose the max-value subset of the active elements. We denote the expected value by $\mathsf{alg}(\mathcal{T}, f)$:

$$\mathsf{alg}(\mathcal{T}, f) := \mathbb{E}_{\ell\leftarrow\pi_T}\left[\mathbb{E}_{R\sim V(p)}[f^{\max}(R \cap \mathsf{elt}(P_\ell))]\right]. \tag{5.3}$$

5.3 Monotone Non-Negative Submodular Functions

We now prove Theorem 5.1.1, and bound the adaptivity gap for *monotone* submodular functions f over any prefix-closed set of constraints. The idea is a natural one in retrospect: we take an

adaptive tree $\mathcal{T}$, and show that the natural non-adaptive strategy (given by choosing a random root-leaf path down the tree, and probing the elements on that path) is within a factor of 3 of the adaptive tree. The proof is non-trivial, though. One strategy is to induct on the two children of the root (which, say, probes element e), but note that the objective f is *not* additive and the adaptive and non-adaptive algorithms recurse having seen different sets of active elements.[2] This issue caused the previous result [GNS16] to proceed along different lines, using massive union bounds over the paths in the decision tree, and hence losing logarithmic factors. They were also restricted to matroid rank functions, instead of all submodular functions.

A crucial insight in our proof is to focus on the *stem* of the tree (the all-no path from the root, see Definition 5.2.1), and induct on the subtrees hanging off this stem. Again we have issues of adaptive and non-adaptive recursing with different active elements, but we control this by amortizing the value obtained from the stem over the different non-adaptive strategies. The proof for non-monotone functions in §5.4 is even more tricky, and will build further on ideas from this monotone case. Formally, the main technical result is the following:

Theorem 5.3.1. *For any adaptive strategy tree $\mathcal{T}$, and any monotone non-negative submodular function $f : 2^V \to \mathbb{R}_{\geq 0}$ with $f(\emptyset) = 0$,*

$$\mathsf{alg}(\mathcal{T}, f) \geq \frac{1}{3}\,\mathsf{adap}(\mathcal{T}, f).$$

Theorem 5.1.1 follows by the observation that each root-leaf path in $\mathcal{T}$ satisfies the prefix-closed constraints, which gives us a feasible non-adaptive strategy. Some comments on the proof: because the function f is monotone, $f^{\max} = f$. Plugging this into (5.2) and (5.3), we want to show that

$$\mathbb{E}_{\ell \leftarrow \pi_T}\Big[\mathbb{E}_{R \sim V(p)}[f(R \cap \mathsf{elt}(P_\ell))]\Big] \;\geq\; \frac{1}{3}\,\mathbb{E}_{\ell \leftarrow \pi_{\mathcal{T}}}[f(A_\ell)]. \tag{5.4}$$

Since both expressions take expectations over the random path, the proof proceeds by induction on the deepness of the tree. (Recall the definition of deepness in Definition 5.2.2.) We argue that for the stem starting at the root, alg gets a value close to adap in expectation (Lemma 5.3.3). However, to induct on the subtree that the algorithms leave the stem on, the problem is that the two algorithms may have picked up different active elements on the stem, and hence the "contracted" functions may look very different. The idea now is to give adap the elements picked by alg for "free" and disallow alg (just for the analysis) to pick elements picked by adap after exiting the stem. Now both the algorithms work after contracting the same set of elements in f, and we are able to proceed with the induction.

5.3.1 Proof of Theorem 5.3.1

We proceed by induction on the deepness of the adaptive strategy tree $\mathcal{T}$. For the base case of deepness 0, $\mathcal{T}$ does not contain any internal node. So both alg and adap get zero value, and the theorem is vacuously true.

[2] The adaptive strategy sees e as active when it takes the `yes` branch (with probability p_e), and nothing as active when taking the `no` branch. Whereas, the non-adaptive strategy sees e as active with the same probability p_e in both branches.

To prove the induction step, recall that the stem is the path in $\mathcal{T}$ obtained by starting at the root node and following the no arcs until we reach a leaf. (See Figure 5.1.) Let $v_1, v_2, \ldots, v_\ell$ denote the nodes along the stem of $\mathcal{T}$ with v_1 being the root and v_ℓ being a leaf; let $e_i = \mathrm{elt}(v_i)$. For $i \geq 1$, let $\mathcal{T}_i$ denote the subtree hanging off the yes arc leaving v_i. The probability that a path following the probability distribution $\pi_{\mathcal{T}}$ enters $\mathcal{T}_i$ is $p_i \prod_{j<i} q_j$, where $p_i = 1 - q_i$ denotes the probability that the i^{th} element is active.

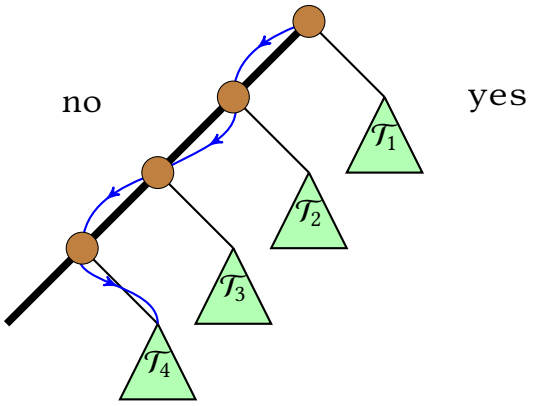

Figure 5.1: Adaptive strategy tree $\mathcal{T}$. The thick line shows the all-no path. The arrows show the path taken by adap. In this example $i = 4$ and $S_i = \{e_1, e_2, e_3, e_4\}$.

Let $S_i = \{e_1, e_2, \ldots, e_i\}$ be the first i elements probed on the stem, and $R_i \sim S_i(\mathbf{p})$ be a random subset of S_i that contains each element e of S_i independently w.p. p_e. We can now rewrite adap and alg in a form more convenient for induction. Here we recall the definition of a marginal with respect to subset Y: $f_Y(S) := f(Y \cup S) - f(Y)$. Note that the leaf v_ℓ has no associated element; to avoid special cases we define a dummy element e_ℓ with $f(\{e_\ell\}) = 0$ and $f_{\{e_\ell\}} = f$.

Claim 5.3.2. *Let I be the random variable denoting the index of the node at which a random walk according to $\pi_{\mathcal{T}}$ leaves the stem. (If $I = \ell$ then the walk does not leave the stem, and $\mathcal{T}_\ell$ is a deepness-zero tree.) Then,*

$$\mathrm{adap}(\mathcal{T}, f) = \mathbb{E}_I\Big[f(e_I) + \mathrm{adap}(\mathcal{T}_I, f_{\{e_I\}})\Big] \tag{5.5}$$

$$\leq \mathbb{E}_{I,R\sim S_I(\mathbf{p})}\Big[f(e_I) + f(R) + \mathrm{adap}(\mathcal{T}_I, f_{R\cup e_I})\Big] \tag{5.6}$$

$$\mathrm{alg}_H(\mathcal{T}) = \mathbb{E}_{I,R\sim S_I(\mathbf{p})}\Big[f(R) + \mathrm{alg}(\mathcal{T}_I, f_R)\Big] \tag{5.7}$$

$$\geq \mathbb{E}_{I,R\sim S_I(\mathbf{p})}\Big[f(R) + \mathrm{alg}(\mathcal{T}_I, f_{R\cup e_I})\Big]. \tag{5.8}$$

Proof. Equation (5.5) follows from the definition of adap; (5.6) follows from the monotonicity of f. (We are giving the adaptive strategy elements in R "for free".) Equation (5.7) follows from the definition of alg, and (5.8) uses the consequence of submodularity that marginals can only decrease for larger sets. □

Observe the expressions in (5.6) and (5.8) are ideally suited to induction. Indeed, since the function $f_{R\cup e_I}$ also satisfies the assumptions of Theorem 5.3.1, and the height of $\mathcal{T}_i$ is smaller

than that of $\mathcal{T}$, we use induction hypothesis on $\mathcal{T}_i$ with the monotone non-negative submodular function $f_{R\cup E_I}$ to get

$$\mathbb{E}_{I,R\sim S_I(\mathbf{p})}\Big[\mathrm{alg}(\mathcal{T}_I, f_{R\cup e_I})\Big] \geq \frac{1}{3}\,\mathbb{E}_{I,R\sim S_I(\mathbf{p})}\Big[\mathrm{adap}(\mathcal{T}_I, f_{R\cup e_I})\Big].$$

Finally, we use the following Lemma 5.3.3 to show that

$$\mathbb{E}_{I,R\sim S_I(\mathbf{p})}\Big[f(R)\Big] \geq \frac{1}{3}\,\mathbb{E}_{I,R\sim S_I(\mathbf{p})}\Big[f(R) + f(e_I)\Big].$$

Substituting these two into (5.6) and (5.8) finishes the induction step.

Lemma 5.3.3. *Let I be the random variable denoting the index of the node at which a random walk according to $\pi_{\mathcal{T}}$ leaves the stem. (If $I = \ell$ then the walk does not leave the stem.) Then,*

$$\mathbb{E}_{I,R\sim S_I(\mathbf{p})}\Big[f(R)\Big] \geq \frac{1}{2}\,\mathbb{E}_{I}\Big[f(e_I)\Big].$$

Proof. For brevity, we use $\mathbb{E}_{I,R}[\cdot]$ as shorthand for $\mathbb{E}_{I,R\sim S_I(\mathbf{p})}[\cdot]$ in the rest of the proof. We prove this lemma by showing that

$$\mathbb{E}_{I,R}\Big[f(R)\Big] \geq \mathbb{E}_{I,R}\Big[\max_{e\in R} f(e)\Big] \geq \frac{1}{2}\,\mathbb{E}_{I}\Big[f(e_I)\Big].$$

The first inequality uses monotonicity. The rest of the proof shows the latter inequality.

For any real $x \geq 0$, let W_x denote the indices of the elements e_j on the stem with $f(e_j) \geq x$, and let $\overline{W}_x$ denote the indices of stem elements not in W_x. Then,

$$\begin{aligned}\mathbb{E}_{I}\Big[f(e_I)\Big] &= \int_0^\infty \Pr_I[f(e_I) \geq x]\,dx \\ &= \int_0^\infty \Pr_I[I \in W_x]\,dx = \int_0^\infty \sum_{i\in W_x}\Big(p_i \prod_{j<i} q_j\Big)\,dx, \end{aligned} \tag{5.9}$$

where the last equality uses that the probability of exiting stem at i is $p_i \prod_{j<i} q_j$.

On the other hand, we have

$$\begin{aligned}\mathbb{E}_{I,R}\Big[\max_{e\in R} f(e)\Big] &= \int_0^\infty \Pr_{I,R}[\max_{e\in R} f(e) \geq x]\,dx \\ &= \int_0^\infty \Pr_{I,R}[R \cap W_x \neq \emptyset]\,dx. \end{aligned} \tag{5.10}$$

For any x, we have

$$\begin{aligned}&\Pr_{I,R}[R \cap W_x \neq \emptyset] \\ &= \sum_{k\in W_x} \Pr_{I,R}[e_k \in R \text{ and } e_j \notin R \text{ for all } j < k \text{ with } j \in W_x]\end{aligned}$$

$$= \sum_{k \in W_x} \Pr_I[I \geq k] \cdot \Pr[e_k \text{ active}] \cdot \Pr_I[\, e_j \text{ inactive for all } j < k \text{ with } j \in W_x \,] \tag{5.11}$$

$$= \sum_{k \in W_x} \Big(\prod_{j<k} q_j \Big) \cdot p_k \cdot \Big(\prod_{j<k \,\&\, j \in W_x} q_j \Big) \tag{5.12}$$

$$= \sum_{k \in W_x} \Big(\prod_{j<k \,\&\, j \in W_x} q_j^2 \Big) \cdot \Big(\prod_{j<k \,\&\, j \notin W_x} q_j \Big) \cdot p_k. \tag{5.13}$$

Recall that $R \sim S_I(\mathbf{p})$. Above (5.11) is because, for e_k to be the first element in $W_x \cap R$, (i) the index I must go past k, (ii) e_k must be active, and (iii) all elements before k on the stem with indices in W_x must be inactive (which are all independent events). Equation (5.12) is by definition of these probabilities. Combining (5.10) and (5.13), and renaming k to i,

$$\mathbb{E}_{I,R}\Big[\max_{e_j \in R} f(e_j) \Big] = \int_0^\infty \sum_{i \in W_x} \Big(p_i \Big(\prod_{j<i \,\&\, j \in W_x} q_j^2 \Big) \Big(\prod_{j<i \,\&\, j \notin W_x} q_j \Big) \Big)\, dx. \tag{5.14}$$

To complete the proof, we compare equations (5.9) and (5.14) and want to show that for every x,

$$\sum_{i \in W_x} \Big(p_i \Big(\prod_{j<i \,\&\, j \in W_x} q_j^2 \Big) \Big(\prod_{j<i \,\&\, j \notin W_x} q_j \Big) \Big) \geq \frac{1}{2} \sum_{i \in W_x} \Big(p_i \prod_{j<i} q_j \Big). \tag{5.15}$$

While the expressions look complicated, things simplify considerably when we condition on the outcomes of elements outside W_x. Indeed, observe that the left-hand-side of (5.15) equals

$$\mathbb{E}_{\overline{W}_x}\Big[\sum_{i \in W_x} \Big(p_i \Big(\prod_{j<i \,\&\, j \in W_x} q_j^2 \Big) \Big(\prod_{j<i \,\&\, j \notin W_x} \mathbf{1}_{q_j} \Big) \Big) \Big],$$

where $\mathbf{1}_{q_j}$ is an independent 0-1 random variable taking value 1 w.p. q_j, and we take the expectation over coin tosses for elements in $\overline{W}_x$. Similarly, the right-hand-side of (5.15) is

$$\frac{1}{2} \sum_{i \in W_x} \Big(p_i \prod_{j<i} q_j \Big) = \mathbb{E}_{\overline{W}_x}\Big[\frac{1}{2} \sum_{i \in W_x} \Big(p_i \Big(\prod_{j<i \,\&\, j \in W_x} q_j \Big) \Big(\prod_{j<i \,\&\, j \notin W_x} \mathbf{1}_{q_j} \Big) \Big) \Big].$$

Now we condition on the elements in $\overline{W}_x$: let $s \in \overline{W}_x$ denote the first element in $\overline{W}_x$ with $\mathbf{1}_{q_s} = 0$. Let W_x' denote all elements in W_x that appear before s. In order to prove (5.15) it now suffices to show:

$$\sum_{i \in W_x'} \Big(p_i \Big(\prod_{j<i \,\&\, j \in W_x'} q_j^2 \Big) \Big) \geq \frac{1}{2} \sum_{i \in W_x'} \Big(p_i \Big(\prod_{j<i \,\&\, j \in W_x'} q_j \Big) \Big).$$

This inequality can be proved using the following claim.

Claim 5.3.4. *For any ordered set A of probabilities $\{a_1, a_2, \ldots, a_{|A|}\}$, let b_j denote $1 - a_j$ for $j \in [1, |A|]$. Then,*

$$\sum_i a_i \Big(\prod_{j<i} b_j \Big)^2 \geq \frac{1}{2} \sum_i a_i \Big(\prod_{j<i} b_j \Big)$$

Proof. Observe that

$$\begin{aligned}
&\sum_i a_i \Big(\prod_{j<i} b_j\Big)^2 \\
&= \sum_i \frac{1-b_i^2}{1+b_i}\Big(\prod_{j<i} b_j\Big)^2 \geq \frac{1}{2}\sum_i (1-b_i^2)\Big(\prod_{j<i} b_j^2\Big) \\
&=^{(\star)} \frac{1}{2}\Big(1-\prod_i b_i^2\Big) = \frac{1}{2}\Big(1-\prod_i b_i\Big)\Big(1+\prod_i b_i\Big) \\
&\geq \frac{1}{2}\Big(1-\prod_i b_i\Big) =^{(\star)} \frac{1}{2}\sum_i a_i\Big(\prod_{j<i} b_j\Big),
\end{aligned}$$

where we have repeatedly used $a_j + b_j = 1$ for all j. The equalities marked $(\star)$ move between two ways of expressing the probability of at least one "heads" when the tails probability is b_j^2 and b_j respectively. □

Applying the claim to the elements in W_x, in order of their distance from the root, completes the proof of Lemma 5.3.3. □

5.3.2 Lower Bound of 2 for Submodular Functions

Our analysis cannot be substantially improved since Claim 5.3.4 is tight. Consider the setting with $|A|$ being infinite for now, and $a_i = \varepsilon$ for all i. Then the LHS of Claim 5.3.4 is $\varepsilon \sum_i (1-\varepsilon)^{2(i-1)} = \frac{\varepsilon}{1-(1-\varepsilon)^2} \approx \frac{1}{2} + O(\varepsilon)$, whereas the sum on the right is 1. Making $|A|$ finite but large compared to $\frac{1}{\varepsilon}$ would give similar results.

We now present a monotone non-negative submodular function and a prefix-closed set of constraints where the adaptivity gap for stochastic probing is arbitrarily close to 2.

Theorem 5.3.5. *The optimal adaptivity gap for stochastic probing where the constraints are prefix-closed and the function is a monotone non-negative submodular is at least 2.*

Proof. Our example is on a universe $V := \{e_{(k,l)} \mid k,l \in \mathbb{Z}_{\geq 0}\}$ where every element is independently active with probability ϵ for some $0 < \epsilon < 1$.

Example: We define our submodular objective f to be the weighted rank function of a partition matroid that selects at most one element from each part. The elements are partitioned according to their first label—for every $k \in \mathbb{Z}_{\geq 0}$ the set $\{e_{(k,l)} \mid l \in \mathbb{Z}_{\geq 0}\}$ is a part of the partition matroid with weight $(1-\epsilon)^k$. In other words, for any set $S \subseteq V$ let $K(S) := \{k \mid e_{(k,l)} \in S\}$ be the (unique) set of first labels, then

$$f(S) := \sum_{k \in K(S)} (1-\epsilon)^k.$$

Note that this series always converges so f is well defined.

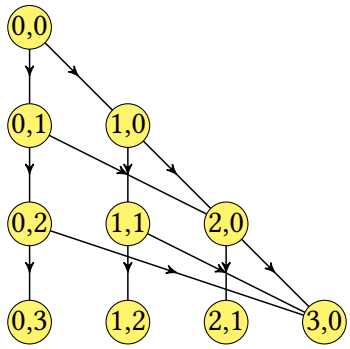

Figure 5.2: Adaptivity gap lower bound example for monotone submodular functions.

To define our prefix-closed constraints, we consider an infinite directed acyclic graph where every element is identified with a single node in the graph. Every node/element $e_{(k,l)}$ has exactly two outgoing edges: towards $e_{(k,l+1)}$ and towards $e_{(k+l+1,0)}$. We denote $\{e_{(k,0)}, e_{(k,1)}, \ldots\}$ as the elements on *column* k. The probing constraint is that a sequence of elements can be probed if and only if it corresponds to a directed path starting at $e_{(0,0)}$. See §5.2 for an illustration.

Analysis: We first give an adaptive strategy with value $2-\epsilon$ (in Eq. (5.16)) and later argue that every non-adaptive strategy has value at most 1 (in Eq. (5.17)); thereby, proving this theorem. Although, the probing constraint allows for infinite strategies, and in a different setting it would not be clear how to define their expected values, since f is monotone we include every active element in the solution. So the expected value of an infinite strategy can be defined as the limit of the strategies that only probe a finite number of elements.

Our adaptive strategy adap starts with probing element $e_{(0,0)}$. It is defined recursively: after probing $e_{(k,l)}$, the next element to probe is either $e_{(k+l+1,0)}$ if $e_{(k,l)}$ is found active, or $e_{(k,l+1)}$ otherwise. In other words, it probes elements on a column until it finds one active, and then probes another column.

Let $\mathrm{adap}(k)$ denote the expected additional value our above adaptive strategy if the next probed element is $e_{(k,0)}$ and let $\mathrm{adap} := \mathrm{adap}(0)$ denote the expected value of the entire strategy. Note that $\mathrm{adap}(k)$ does not depend on the set of elements found active before probing $e_{(k,0)}$ (i.e., the elements $e_{(k',l')}$ where $k' < k$). Furthermore, the subgraph reachable from $e_{(k,0)}$ is similar to the entire graph on V in the sense that one can relabel the elements in the subgraph to match the entire graph exactly, the only difference being that the value of any subset is multiplied by a factor of $(1-\epsilon)^k$. Therefore, we have

$$\mathrm{adap}(k) = (1-\epsilon)^k \cdot \mathrm{adap}(0).$$

Now, summing over the number of inactive elements on column 0, we get

$$\mathrm{adap}(0) \;=\; \sum_{k=0}^{\infty} (1-\epsilon)^k \cdot \epsilon \cdot \Big(1 + \mathrm{adap}(k+1)\Big) \;=\; \sum_{k=0}^{\infty} (1-\epsilon)^k \cdot \epsilon \Big(1 + (1-\epsilon)^{k+1} \cdot \mathrm{adap}(0)\Big),$$

which uses $\mathrm{adap}(k) = (1-\epsilon)^k \cdot \mathrm{adap}(0)$. Solving this equation yields the result:

$$\mathrm{adap} = \mathrm{adap}(0) = 2 - \epsilon. \tag{5.16}$$

Similarly, let $\mathrm{alg}(k)$ denote the expected additional value of the optimal non-adaptive strategy if the next probed element is $e_{(k,0)}$, and let $\mathrm{alg} = \mathrm{alg}(0)$ denote the expected value of the optimal non-adaptive strategy. By the same argument as $\mathrm{adap}(k)$, we have

$$\mathrm{alg}(k) = (1-\epsilon)^k \cdot \mathrm{alg}(0).$$

Let k denote the number of elements the optimal non-adaptive strategy probes on column 0. We get

$$\mathrm{alg}(0) \quad = \quad \max_{k\geq 1}\left\{1-(1-\epsilon)^k + \mathrm{alg}(k+1)\right\} \quad = \quad \max_{k\geq 1}\left\{1-(1-\epsilon)^k + (1-\epsilon)^k \cdot \mathrm{alg}(0)\right\},$$

which uses $\mathrm{alg}(k) = (1-\epsilon)^k \cdot \mathrm{alg}(0)$. This implies

$$\mathrm{alg} = \mathrm{alg}(0) = 1. \tag{5.17}$$

Combining Eq. (5.16) and Eq. (5.17), we get an adaptivity gap arbitrarily close to 2 for $\epsilon \to 0$. □

5.3.3 Finding Non-Adaptive Polices

A non-adaptive policy is given by a fixed sequence $\sigma = \langle e_1, e_2, \ldots, e_k\rangle$ of elements to probe (such that σ satisfies the given prefix-closed probing constraint. If A is the set of active elements, then the value we get is $\mathbb{E}_{A\sim V(\mathbf{p})}[f^{\max}(A\cap\{e_1,\ldots,e_k\})] = \mathbb{E}_A[f(A\cap\{e_1,\ldots,e_k\})]$, the equality holding for monotone functions. If we define $g(S) := \mathbb{E}_{A\sim V(\mathbf{p})}[f(V\cap A)]$, g is also a monotone submodular function. Hence finding good non-adaptive policies for f is just optimizing the monotone submodular function g over the allowed sequences. E.g., for the probing constraint being a matroid constraint, we can get a $\frac{e}{e-1}$-approximation [CCPV11]; for it being an orienteering constraint we can get an $O(\log n)$-approximation in quasi-polynomial time [CP05].

For non-monotone functions (discussed in the next section), we can approximate the $f^{\max}(S)$ function by $E_{R\sim V(\frac{1}{2})}[f(S\cap R)]$, and losing a factor of 4, reduce finding good non-adaptive strategies to (non-monotone) submodular optimization over the probing constraints.

5.4 Non-Monotone Non-Negative Submodular Functions

We now prove Theorem 5.1.2. The proof for the monotone case used monotonicity in several places, but perhaps the most important place was to claim that going down the tree, both adap and alg could add all active elements to the set. This "online" feature seemed crucial to the proof. In contrast, when the adaptive strategy adap reaches a leaf in the non-monotone setting, it chooses the best subset within the active elements; a similar choice is done by the non-adaptive algorithm. This is why we have $f^{\max}(A_\ell)$ in (5.2) versus $f(A_\ell)$ in (5.4).

Fortunately, a result from Feige el al. [FMV11] shows that for non-negative non-monotone submodular functions, the simple strategy of picking every active element independently w.p.

half gives us a near-optimal possible subset. Losing a factor of four, this result allows us to analyze the performance relative to an adaptive *online* algorithm adap_{on} which selects (with probability $\frac{1}{2}$) each probed element that happens to be active. By modifying the tree, we can view adap_{on} as selecting *every* active element: the modified adap_{on} tree probes each element w.p. $\frac{1}{2}$ and select every active element.

The rest of the proof is similar (at a high level) to the monotone case: to relate adap_{on} and alg we bound them using comparable terms ($\overline{\text{adap}}$ and $\overline{\text{alg}}$ in Definition 5.4.1) and apply induction. Altogether we will obtain:

$$\text{alg} \overset{\text{(Lemma 5.4.2}(ii))}{\geq} \overline{\text{alg}} \overset{\text{(Lemma 5.4.4)}}{\geq} \frac{1}{5} \cdot \overline{\text{adap}}$$
$$\overset{\text{(Lemma 5.4.2}(i))}{\geq} \frac{1}{10} \cdot \text{adap}_{on} \overset{(5.1)}{\geq} \frac{1}{40} \cdot \text{adap}.$$

In the inductive proof, we will work with "contracted" submodular functions g obtained from f, which may take negative values but have $g(\emptyset) = 0$. In order to deal with such issues, the induction here is more complex than in the monotone case.

We first define the surrogates $\overline{\text{adap}}$ and $\overline{\text{alg}}$ for adap and alg recursively as follows.

Definition 5.4.1. *For any adaptive online strategy tree $\mathcal{T}$ and submodular function g with $g(\emptyset) = 0$, let*

- *I be the node at which a random walk according to $\pi_{\mathcal{T}}$ exits the stem.*
- *$R \sim S_I(\mathbf{p})$ where S_I denotes the elements on the stem until node I.*
- *$J = \arg\max\{g(e) \mid e \in R, g(e) > 0\}$ w.p. $\frac{1}{2}$ and $J = \perp$ w.p. $\frac{1}{2}$.*

Then we define:

$$\overline{\text{adap}}(\mathcal{T}, g) := \mathbb{E}_{I,J}\left[g(I) + g(J) + \overline{\text{adap}}(\mathcal{T}_I, g_{I \cup J})\right] \quad \text{and}$$
$$\overline{\text{alg}}(\mathcal{T}, g) := \mathbb{E}_{I,J}\left[g(J) + \overline{\text{alg}}(\mathcal{T}_I, g_{I \cup J})\right].$$

Above we account for the non-monotonicity of the function, via this process of random sampling used in the definition of $\overline{\text{adap}}$ and $\overline{\text{alg}}$. One problem with following the proof from §5.3 is that when we induct on the "contracted" function f_S for some set S, this function may not be non-negative any more. Instead, our proof considers the entire path down the tree and argues about it at one shot; to make the analysis easier we imagine that the non-adaptive algorithm picks at most one item from the stem, i.e., the one with the highest marginal value.

Lemma 5.4.2. *For any adaptive online strategy tree $\mathcal{T}$, the following hold:*

(i) For any non-negative submodular function f, $\overline{\text{adap}}(\mathcal{T}, f) \geq \frac{1}{2}\text{adap}_{on}(\mathcal{T}, f)$.

(ii) For any submodular function g, $\text{alg}(\mathcal{T}, g) \geq \overline{\text{alg}}(\mathcal{T}, g)$.

We make use of the following property of submodular functions.

Lemma 5.4.3 ([BFNS14], Lemma 2.2)**.** *For any non-negative submodular function $h : 2^A \to \mathbb{R}_{\geq 0}$ (possibly with $h(\emptyset) \neq 0$) let $S \subseteq A$ be a random subset that contains each element of A with probability* at most *p (not necessarily independently). Then,*

$$\mathbb{E}_S[f(S)] \geq (1 - p) \cdot f(\emptyset).$$

Proof of Lemma 5.4.2. We condition on a random leaf ℓ drawn according to $\pi_{\mathcal{T}}$. Let $I_1, \ldots, I_d$ denote the sequence of nodes that correspond to active elements on the path P_ℓ, i.e., I_1 is the point where P_ℓ exits the stem of $\mathcal{T}$, I_2 is the point where P_ℓ exits the stem of $\mathcal{T}_{I_1}$ etc. Then, the adaptive online value is exactly $f(\{I_1, \ldots I_d\})$. For any $k = 1, \ldots, d$ let $P_\ell[I_{k-1}, I_k]$ denote the elements on path P_ℓ between I_{k-1} and I_k. Also let R denote the random subset where each element e on path P_ℓ is chosen independently w.p. p_e.

For $k = 1, \ldots, d$, define J_k as follows:

$$J_k = \arg\max\{f_{L_{k-1}}(e) \mid e \in R \cap P_\ell[I_{k-1}, I_k], f_{L_{k-1}}(e) > 0\}$$
$$\text{with probability } \frac{1}{2}, \quad \text{and}$$
$$J_k = \perp \text{ with probability } \frac{1}{2},$$

where $L_{k-1} := \{I_1, \ldots, I_{k-1}\} \cup \{J_1, \ldots, J_{k-1}\}$. In words, the sets L contain the exit points from the stems, and for each stem also the element with maximum marginal value (if any) with probability half.

For (i), by Definition 5.4.1, the value of $\overline{\text{adap}}(\mathcal{T}, f)$ conditioned on path P_ℓ and elements $J_1, \ldots, J_d$ is

$$\begin{aligned} \sum_{k=1}^{d} f_{L_{k-1}}(I_k) + f_{L_{k-1}}(J_k) \quad &\geq \quad \sum_{k=1}^{d} f_{L_{k-1}}(\{I_k, J_k\}) \\ &= \quad f(\{I_1, J_1, \ldots I_d, J_d\}). \end{aligned} \tag{5.18}$$

The inequality follows from the following two cases:

- If $I_k \neq J_k$, then by submodularity of $f_{L_{k-1}}$,

$$\begin{aligned} f_{L_{k-1}}(I_k) + f_{L_{k-1}}(J_k) &\geq f_{L_{k-1}}(\{I_k, J_k\}) + f_{L_{k-1}}(\emptyset) \\ &= f_{L_{k-1}}(\{I_k, J_k\}). \end{aligned}$$

- If $I_k = J_k$, then by choice of J_k we have $f_{L_{k-1}}(J_k) > 0$ and

$$f_{L_{k-1}}(I_k) + f_{L_{k-1}}(J_k) = 2 \cdot f_{L_{k-1}}(J_k) > f_{L_{k-1}}(J_k).$$

Using (5.18) and taking expectation over the Js, $\overline{\text{adap}}(\mathcal{T}, f)$ conditioned on path P_ℓ is at least

$$\mathbb{E}_{J_1, \ldots J_d}\left[f(\{I_1, J_1, \ldots I_d, J_d\})\right] \geq \frac{1}{2} \cdot f(\{I_1, \ldots I_d\}).$$

Above we used Lemma 5.4.3 on the non-negative submodular function $h(S) := f(S \cup \{I_1, \ldots, I_d\})$, using the fact that the set $\{J_1, \ldots, J_d\}$ contains each element with probability at most half. Finally, deconditioning over ℓ (i.e., over $I_1, \ldots I_d$) proves part (i).

For part (ii), by Definition 5.4.1, the value of $\overline{\text{alg}}(\mathcal{T}, g)$ conditioned on path P_ℓ and elements $J_1, \ldots, J_d$ is

$$\sum_{k=1}^{d} g_{L_{k-1}}(J_k) \leq \sum_{k=1}^{d} g_{J_1, \ldots J_{k-1}}(J_k) = g(\{J_1, \ldots, J_d\}),$$

where the inequality is by submodularity of g. Since alg chooses the maximum value subset in R and $\{J_1, \ldots, J_d\} \subseteq R$, taking expectations over ℓ and R, we prove part (ii). This completes the proof of Lemma 5.4.2. □

Lemma 5.4.4. *For any strategy tree $\mathcal{T}$ and submodular function g with $g(\emptyset) = 0$, $\overline{\text{alg}}(\mathcal{T}, g) \geq \frac{1}{5} \cdot \overline{\text{adap}}(\mathcal{T}, g)$.*

Proof. We proceed by induction. Recall the notation in Definition 5.4.1. For each node i on the stem of $\mathcal{T}$ define $a_i := \max\{g(i), 0\}$. Note that $g(J) = a_J$ by choice of J: if $J \neq \perp$ we have $g(J) > 0$ and if $J = \perp$, $g(J) = g(\emptyset) = 0 = a_J$. We will show that

$$\mathbb{E}_{I,J}[a_I] \leq 4 \cdot \mathbb{E}_{I,J}[a_J]. \tag{5.19}$$

Then the definition of $\overline{\text{adap}}(\mathcal{T}, g)$ and $\overline{\text{alg}}(\mathcal{T}, g)$, and induction on $\mathcal{T}_I$ and $g_{I \cup J}$, would prove the lemma.

Let $K = \arg\max\{a_e \mid e \in R\}$ be the r.v. denoting the maximum weight active (i.e., in R) element on the stem. Then, by definition of J, we have $\mathbb{E}_{I,J}[a_J] = \frac{1}{2}\mathbb{E}_{I,K}[a_K]$. Finally we can use Lemma 5.3.3 from Section 5.3 to obtain $\mathbb{E}_{I,K}[a_K] \geq \frac{1}{2}\mathbb{E}_{I,J}[a_I]$, which proves (5.19). This completes the proof of Lemma 5.4.4. □

5.5 Applications

5.5.1 Stochastic Probing on Metrics

In the stochastic probing problem on metrics, recall that the elements are in a metric space (V, d), and probing constraints enforce that the probed elements lie on a path of length at most B.

To begin, consider the simpler case where the objective is a simple submodular function given by rank function $r()$ of single matroid $\mathcal{M}$; here each element e is active and has value 1 with probability p_e, or is inactive (has value zero) with probability $1 - p_e$. Let $A \sim V(\mathbf{p})$ be a random set of active elements. The non-adaptive problem is now to find a path Q of total length B, to maximize the quantity $g(Q) := \mathbb{E}_{A \sim V(\mathbf{p})}[r(Q \cap A)]$. Since the rank function $r()$ is a submodular function, so is $g()$, and we have a problem of finding a path Q of length at most B to maximize the submodular function $g(Q)$. This is precisely the submodular orienteering problem for which Chekuri and Pál [CP05] gave an $O(\log n)$ approximation in *quasi-polynomial time*.

Theorem 5.5.1. *There is a quasi-polynomial time $O(\log n)$-approximation algorithm for non-adaptive stochastic probing on metrics when the objective function is a submodular.*

We next prove a "generalization" of Theorem 5.5.1 to objective functions that are given by intersection of k matroids. For the intersection of k matroids, define the "rank" $r(S)$ to be the size of the largest common independent set contained in S. The function $r()$ is no longer submodular, so we cannot use the same arguments as above. In Chapter 6 we prove a generalization of

our $O(1)$-adaptivity gaps for submodular functions to $O(\text{poly}(k))$-adaptivity gaps for functions given by intersection of k matroids (Corollary 6.4.2). Even if we assume this result, we need to still design a non-adaptive algorithm. We achieve this by reworking the proof of Chekuri and Pál, with an additional loss of a factor of k^2. First, we put the stochasticity aside, and try to maximize the cardinality of a common independent set in the intersection of k matroids.

Proposition 5.5.2. *There is a quasi-polynomial time $(k \log_2(2n))$-approximate algorithm to maximize the rank function of the intersection of k matroids, subject to an orienteering constraint.*

Proof. (Sketch) We assume the reader is familiar with the elegant Savitch-like idea from [CP05]. By suitable guessing, we want to find the best s-t path of length B with ℓ hops, where the "halfway" point is v, the length of the s-v portion is B_1 and the rest is B_2. Let $P_1^\star$ and $P_2^\star$ be the s-v and v-t portions of the optimal path $P^\star$, and let $I^\star$ be the optimal independent set. Since we are in the unweighted case, the optimum solution value is $|I^\star| = |I^\star \cap P_1^\star| + |I^\star \cap P_2^\star|$. We recursively find a $1/(k \log_2 \ell)$-approximate solution I_1 to the s-v problem (with $\ell/2$ hops and budget B_1):

$$|I_1| \geq \frac{1}{k \log_2 \ell} |I^\star \cap P_1^\star|.$$

Then we contract the elements of I_1 and find an approximate solution I_2 for the v-t problem (with $\ell/2$ hops and budget $B - B_1$). Since contracting each element of I_1 can only reduce the rank of the problem by k, we get

$$|I_2| \geq \frac{1}{k \log_2 \ell} \Big(|I^\star \cap P_2^\star| - k|I_1| \Big).$$

Combining, we have

$$|I| \quad = \quad |I_1| + |I_2| \quad \geq \quad \frac{1}{k \log_2 \ell} \Big(|I^\star \cap P_1^\star| + |I^\star \cap P_2^\star| - k|I_1| \Big),$$

which implies

$$|I| \quad \geq \quad \frac{|I^\star|}{k(1 + \log_2 \ell)} \quad = \quad \frac{|I^\star|}{k \log_2(2\ell)}.$$

For the base case $\ell = 1$ the only option is to go from s to t, and get a $1 \geq \frac{1}{k \log_2 2}$-approximation. □

The entire analysis above is linear, and immediately extends to any positive linear combination of such unweighted rank functions, say those functions of the form

$$g(Q) := \mathbb{E}_{A \sim V(\mathbf{p})}[r(Q \cap A)].$$

Finally, using the reduction from weighted to unweighted rank functions we lose another factor of $O(k)$ and get:

Theorem 5.5.3. *There is a quasi-polynomial time $O(k^2 \log n)$-approximation algorithm for non-adaptive stochastic probing on metrics when the objective is given by intersections of k matroids.*

Finally, combining with Corollary 6.4.2 completes the proof of the following theorem.

Theorem 5.5.4. *There is a quasi-polynomial time $O(k^3 \log k \log n)$-approximation algorithm for stochastic probing on metrics with objective function given by intersection of k-matroids.*

Stochastic Submodular Orienteering for Monotone Submodular Functions The above result also implies an approximation algorithm for the following *stochastic submodular orienteering* (StocSO) problem. Given a metric (V, d) with a root where each vertex $v \in V$ is active independently with probability p_v, a monotone submodular function $g : 2^V \to \mathbb{R}_+$ and length bound B, the goal is to find an adaptive path of length at most B that originates from the root and maximizes the expected function value on *active* elements. When function g is a *monotone submodular function*, notice that StocSO is precisely the above stochastic probing problem on metrics. Hence, by Theorem 5.5.1 there is a quasi-polynomial time $O(\log n)$-approximation algorithm for StocSO (we use the fact that [CP05] algorithm can be used to find a non-adaptive strategy). We summarize this discussion below:

Corollary 5.5.5. *There is a quasi-polynomial time $O(\log n)$-approximation algorithm for stochastic monotone submodular orienteering.*

This result will be useful for the next section.

5.5.2 Stochastic Minimum Latency Submodular Cover

The *minimum latency submodular cover* problem studied in [INvdZ12] is as follows: given a metric (V, d) with root r and m monotone submodular functions $f_i : 2^V \to \mathbb{R}_+$ (for $i \in [m]$), the goal is to find a path originating from r that minimizes the total "cover time" of the m functions. Here, function f_i is said to be covered at time t if this is the minimum value such that

$$f_i\Big(\{v \mid \text{ vertex } v \text{ visited before time } t\}\Big) \quad = \quad f_i(V).$$

In the *stochastic minimum latency submodular cover* (StocMLSC) problem, the input is the same as above and furthermore each vertex (element) $v \in V$ is *active* independently with probability p_v. The goal is to find an adaptive strategy to minimize the *expected* total cover time of the m functions. In the stochastic setting, function f_i is said to be covered at time t if this is the minimum value such that

$$f_i\Big(\{v \mid \ v \text{ visited before time } t \text{ and active}\}\Big) = f_i(V).$$

Due to the stochasticity, there may be situations where some functions f_i never reach the maximum value $f_i(V)$—in such cases the cover time is just set to be the entire path length. We assume, without loss of generality, that functions are normalized so that $f_i(V) = 1$ for $i \in [m]$.

Obtaining a poly-logarithmic approximation ratio for StocMLSC was left open in [INvdZ12]. We answer this question.

Theorem 5.5.6. *There is a quasi-polynomial time $O(\log n \cdot \log \frac{1}{\epsilon})$-approximation algorithm for stochastic minimum latency submodular cover, where ϵ is such that for any $i \in [m]$ and $S' \subseteq S$, if $f_i(S) > f_i(S')$ then $f_i(S) \geq f_i(S') + \epsilon$.*

Proof. (Sketch) The main ingredient in this algorithm is an approximation algorithm for stochastic submodular orienteering (StocSO). Given an α-approximation algorithm for this problem, the algorithm and analysis for the *deterministic* minimum latency submodular cover problem

in [INvdZ12] apply directly to yield an $O(\alpha \cdot \log \frac{1}{\epsilon})$-approximation algorithm for StocMLSC, where ϵ is the smallest positive marginal value of any function $\{f_i\}_{i=1}^m$. Moreover, the function g in the StocSO instances solved by this algorithm is always a non-negative linear combination of the f_is.

Since g is a linear combination of the f_is, Corollary 5.5.5 yields quasi-polynomial time $O(\log n)$-approximation algorithm for StocSO with objective g. Applying this within the algorithm from [INvdZ12], we obtain a $O(\log n \cdot \log \frac{1}{\epsilon})$-approximation algorithm for StocMLSC. □

As an example, consider the stochastic version of the *latency group Steiner* problem [GNR10, CS11]. The input consists of (i) metric (V, d) where each vertex $v \in V$ is *active* independently with probability p_v, (ii) root $r \in V$ and (iii) m groups $\{U_i \subseteq V\}_{i=1}^m$ of vertices. The goal is to find an adaptive path (starting from r) that minimizes the expected cover-time of the m groups. Group U_i is covered at the earliest time when some *active* vertex from U_i is observed. A direct application of Theorem 5.5.6 yields an $O(\log n)$-approximation algorithm (we use the fact that $\epsilon \geq 1$ in this example).

5.5.3 Stochastic Connected Dominating Set

The *budgeted connected dominating set* problem (BCDS) studied in [KPS14] is the following. Given an undirected graph $G = (U, E)$ and budget k, the goal is to choose a set S of k vertices such that $G[S]$ is a connected suggraph, and the number of vertices dominated by S are maximized. A vertex $v \in U$ is said to be dominated by set $S \subseteq U$ if $N(v) \cap S \neq \emptyset$, where $N(v)$ is the set of neighbors of v which includes v.

We consider the following stochastic version of BCDS. The elements are edges E of a graph, and each edge $e \in E$ is *active* for the purpose of dominating one end-point by the other only with some independent probability p_e. Furthermore, the status of an edge e is only known when we "probe" one of its end points. The (random) subset of active edges is denoted $A \subseteq E$. We want an adaptive strategy for probing a sequence of at most k vertices such that:

- the set P of probed vertices at any point in time is connected in the (deterministic) graph $G = (U, E)$, and
- the expected number of vertices dominated by P in the graph $G_p = (U, A)$ is maximized.

This models, for example, a sensor network application where vertices U denote potential coverage locations and edges E correspond to pairs of locations between which two sensors can communicate. A sensor placed at location u covers u, and additionally a (random) subset of u's neighbors between whom the sensing quality is good enough. *A priori* we only know the probabilities of having a good sensing quality; the exact status is known only when a sensor is placed at u or v. We want to set up a connected network using k sensors (possibly adaptively) so as to maximize the expected number of covered locations.

We formally cast the problem in our framework: The ground set contains the edge set E with probabilities $\{p_e\}_{e \in E}$. In our model we must probe elements/edges (and not vertices), but the generality of our framework helps capture this. The outer constraints require that there exists a subset $S \subseteq U$ of vertices such that (i) every probed edge has at least one end-point in S, (ii) S is connected in G, and (iii) $|S| \leq k$. It is easy to see that this can be captured by

a prefix-closed collection of sequences of elements in E. There is no inner constraint, but the objective is the coverage function

$$f(A) \quad := \quad \sum_{v \in U} \mathbf{1}_{A \text{ contains edge incident to } v},$$

where A is the set of probed edges that are *active*. By Theorem 5.1.1 the adaptivity gap of this stochastic probing problem is $O(1)$, so we can focus on the non-adaptive problem.

In the non-adaptive stochastic BCDS problem, we want a static set $S \subseteq U$ of at most k connected vertices in G that maximizes the function

$$\begin{aligned} g(S) \quad &:= \quad |S| + \sum_{v \in U \setminus S} \Pr[v \text{ dominated by } S] \\ &= \quad |S| + \sum_{v \in U \setminus S} \left(1 - \prod_{e \in E(S,v)} (1 - p_e)\right). \end{aligned}$$

Above $E(S, v)$ denotes the set of edges incident to v whose other endpoint lies in S. Not only is g is a submodular function on U, it is also a "special submodular" function as defined in [KPS14], so the algorithm from [KPS14] yields an $O(1)$-approximation for non-adaptive stochastic BCDS, and hence proves the following Theorem 5.5.7.

Theorem 5.5.7. *There is a constant factor approximation algorithm for stochastic connected dominating set.*

5.5.4 Stochastic Precedence Constrained Scheduling

Consider a set V of tasks with precedence constraints, where each task $e \in V$ has an independent random *value/reward* $X_e \in \mathbb{Z}_+$. The exact value of a task is known only when it is probed, and the order of probing tasks must satisfy the precedence. (I.e., a task can be probed only when its predecessors have all be probed.) There is a bound B on the total number of probes. Each task must be irrevocably accepted/rejected immediately after it is probed. Finally, we are allowed to accept at most k tasks. The objective is to maximize the expected total value of the selected tasks.

This can be modeled as a stochastic probing problem on the ground set V, with independent random values X_e for each $e \in V$. The outer constraints requires that we probe a set $S \subseteq V$ with $|S| \leq B$ that satisfies the precedence (i.e., it is a lower-ideal of the poset giving the precedence constraints). The inner constraint is a single uniform matroid of rank k. By Theorem 5.1.1 the adaptivity gap of this probing problem is $O(1)$, and hence it suffices to give an algorithm for the non-adaptive problem.

The non-adaptive problem involves choosing a precedence-constrained set $S \subseteq V$ with $|S| \leq B$ that maximizes:

$$\text{value}(S) \quad := \quad \mathbb{E}\left[\max\left\{\sum_{e \in I} X_e \;\middle|\; I \subseteq S, |I| \leq k\right\}\right].$$

This value is approximated (within factor 4) by value$'(S)$ which is:

$$\sum_{j\in\mathcal{Z}} 2^j \cdot \mathbb{E}\left[\max\{|I| \ : \ I \subseteq S \cap \{e \mid 2^{j-1} \le X_e < 2^j\}, |I| \le k\}\right]$$
$$= \sum_{j\in\mathcal{Z}} 2^j \cdot \mathbb{E}\left[\min\left\{S \cap \{e \mid 2^{j-1} \le X_e < 2^j\}, k\right\}\right].$$

This in turn is approximated within factor $\frac{e}{e-1}$ by

$$\text{value}''(S) = \sum_{j\in\mathcal{Z}} 2^j \cdot \min\left\{\sum_{e\in S} \Pr[2^{j-1} \le X_e < 2^j], k\right\}.$$

This follows from the fact that for independent $[0,1]$ valued random variables $X'_1, \cdots, X'_n$ with $Y = \sum X'_i$, it holds that $\mathbb{E}[\min\{Y,1\}] \ge (1-1/e)\cdot\min\{\mathbb{E}[Y],1\}$ (see, e.g., [AMM+11, Theorem 4]). For each j, we use this result with $X'_e = \frac{1}{k}\cdot \mathbf{1}_{2^{j-1}\le X_e<2^j}$ and $Y = \sum_{e\in S} X'_e$.

Define $p_{e,j} := \Pr[2^{j-1} \le X_e < 2^j]$ for each $e \in V$ and $j \in \mathcal{Z}$. Using standard scaling arguments, we may assume that the random variable takes on values in the range $[1, \text{poly}(n)]$, that so we have $L = O(\log n)$ weight classes $j = 1, \cdots, L$. Summarizing the above reductions, we know that with a constant factor loss, the non-adaptive problem reduces to finding a precedence-constrained set $S \subseteq V$ with $|S| \le B$ that maximizes $X'(S) = \sum_{j=1}^{L} 2^j \cdot \min\left\{\sum_{e\in S} p_{e,j}, r\right\}$.

When $L = 1$, this reduces to the *partially ordered knapsack* problem for which an FPTAS is known when the precedence constraint is an "2-dimensional order" [KS07].[3] The algorithm in [KS07] is a dynamic program which easily extends to give a quasi-polynomial $n^{O(L)}$ time approximation scheme for non-adaptive stochastic precedence constrained scheduling in this setting. Hence we get:

Theorem 5.5.8. *There is a quasi-polynomial time constant-factor approximation algorithm for stochastic precedence constrained scheduling under 2-dimensional orders.*

[3]This class is strictly more general than series-parallel precedence constraints. For general precedence constraints, no approximation algorithm is known: this is at least as hard as Dense-k-subgraph.

Chapter 6

Constrained Stochastic Multi-Value Probing

6.1 Introduction

In Chapter 5 we studied constrained stochastic probing (CoSP) and saw its several applications in §5.5. Although powerful, we make *two* crucial assumptions in that model: (i) each element i can only take a Bernoulli value, i.e., it is active w.p. p_i and is inactive, otherwise; (ii) the combinatorial objective function is submodular. For many applications, it is conceivable that "small" adaptivity gaps hold even when the variables have non-Bernoulli distributions or when the combinatorial function is more general. For example, is the adaptivity gap $O(1)$ for the Best-box problem when X_is are arbitrary non-negative r.v.s? In this chapter we study adaptivity gaps for CoSP both when the distribution of r.v.s goes beyond Bernoulli variables and also when the function is more general, e.g., intersection of matroids, XOS, or monotone subadditive.

For a general submodular function, it is not obvious how to formally define the notion "an element is not Bernoulli". We would like a model that allows an element to take one of k potential "types" for some given k; but then it is unclear how to define element marginals. For example, how should an element i's marginal change over set S when an element in S changes its type? In §6.2 we resolve this problem by modeling "combinatorial but independent functions" that are defined over a larger universe of elements, say $n \times k$ elements.

In §6.3 we prove that our above intuition of small adaptivity gaps for non-Bernoulli submodular functions is correct. In fact, we show that an improved analysis of the approach of taking a random root-leaf path in the decision tree from Chapter 5 can give the *tight adaptivity gap* of 2 for submodular functions (matching the lower bound from §5.3.2). In §6.4 we consider adaptivity gaps for a special class of subadditive functions that are given as weighted rank function of a k-extendible system. This class is more general than rank functions of intersection of k matorids, e.g., $k = 2$ captures the max-weight matching in general graphs. Finally, in §6.5 we bound the adaptivity gaps of XOS functions in terms of their *width* (recollect, Definition 2.2.2). We hope in future work to obtain "width-independent" small adaptivity gaps for XOS functions. Since an XOS function $O(\log n)$-approximates a subadditive function (Lemma 2.4.7),

such a width-independent result would imply a small adaptivity gap for general subadditive functions.

6.2 Combinatorial Functions over Independent Items

In this section, we define what we mean by "submodular (or subadditive) valuations over independent items". Formally, we define:

Definition 6.2.1 ($\mathcal{C}$ Valuations over Independent Items)**.** *Let $\mathcal{C}$ be a class of valuation functions (in particular, we are interested in $\mathcal{C} \in \{$submodular, XOS, subadditive$\}$). Consider:*

- *n sets $U_1, \ldots, U_n$ and distributions $\mathcal{D}_1, \ldots, \mathcal{D}_n$, where $\mathcal{D}_i$ returns a single item (type) from set U_i.*
- *A function $f : \{0,1\}^U \to \mathbb{R}_{\geq 0}$, where $U \triangleq \bigcup_{i=1}^{n} U_i$ and $f \in \mathcal{C}$.*
- *For $\mathbf{X} \in \prod_i U_i$ and subset $S \subseteq [n]$, let $v_{\mathbf{X}}(S) \triangleq f\Big(\{X_i : i \in S\}\Big)$.*

Let $\mathcal{D}$ be a distribution over valuation functions $v(\cdot) : 2^n \to \mathbb{R}_{\geq 0}$. We say that $\mathcal{D}$ is "$\mathcal{C}$ over independent items*" if it can be written as the distribution that first samples $\mathbf{X} \sim \bigtimes_i \mathcal{D}_i$, and then outputs valuation function $v_{\mathbf{X}}(\cdot)$.*

Sometimes, we also denote the sets of types U_i by set T_i and their union by $T \triangleq U = \bigcup_{i=1}^{n} T_i$.

Related notions in the literature

Combinatorial functions over independent items have been considered before. Agrawal et al. [ADSY12], for example, consider a different, incomparable framework defined via the generalization of submodular and monotone functions to non-binary, ordered domains (e.g. $f : \bigtimes_i U_i \to \mathbb{R}_{\geq 0}$ is submodular if $f(\max\{\mathbf{X},\mathbf{Y}\}) + f(\min\{\mathbf{X},\mathbf{Y}\}) \leq f(\mathbf{X}) + f(\mathbf{Y})$). In our definition, per contra, there is no natural way to define a full order over the set of items potentially available on each time period. For example, if we select an item on Day 1, an item X_2 on Day 2 may have a larger marginal contribution than item Y_2, but a lower contribution if we did not select any item on Day 1.

Another relevant definition has been considered in probability theory [Sch99] and more recently in mechanism design [RW15]. The latter paper considers auctioning n items to a buyer that has a random, "independent" monotone subadditive valuation over the items. Now the seller knows which items she is selling them, but different types of buyers may perceive each item differently. This is captured via an *attribute* of an item, which describes how each buyer values a bundle containing this item. Formally,

Definition 6.2.2 (Monotone Subadditive Valuations over Independent Items [Sch99, RW15])**.** *We say that a distribution $\mathcal{D}$ over valuation functions $v(\cdot) : 2^n \to \mathbb{R}$ is* subadditive over independent items *if:*

1. *All $v(\cdot)$ in the support of $\mathcal{D}$ exhibit* no externalities.

   Formally, let $\Omega_S = \bigtimes_{i \in S} \Omega_i$, where each Ω_i is a compact subset of a normed space. There exists

a distribution $\mathcal{D}^{\mathbf{X}}$ over $\Omega_{[n]}$ and functions $V_S : \Omega_S \to \mathbb{R}$ such that $\mathcal{D}$ is the distribution that first samples $\mathbf{X} \sim \mathcal{D}^{\mathbf{X}}$ and outputs the valuation function $v(\cdot)$ with $v(S) = V_S(\langle X_i \rangle_{i \in S})$ for all S.

2. *All $v(\cdot)$ in the support of $\mathcal{D}$ are monotone and subadditive.*
3. *The private information is* independent across items. *That is, the $\mathcal{D}^{\mathbf{X}}$ guaranteed in Property 1 is a product distribution.*

For monotone valuations, Definition 6.2.1 is stronger than Definition 6.2.2 as it assumes that the valuation function is defined over every subset of $U = \bigcup U_i$, rather than just the support of $\mathcal{D}$. However, it turns out that for monotone subadditive functions Definitions 6.2.1 and 6.2.2 are equivalent.

Observation 6.2.3. *A distribution $\mathcal{D}$ is subadditive over independent items according to Definition 6.2.1 if and only if it is subadditive over independent items according to Definition 6.2.2.*

Proof sketch. It's easy to see that Definition 6.2.1 implies Definition 6.2.2: We can simply identify between the set of attributes Ω_i on day i and the set of potential items U_i, and let $V_S(\langle X_i \rangle_{i \in S}) \triangleq f\left(\{X_i : i \in S\}\right)$. Observe that the desiderata of Definition 6.2.2 are satisfied.

In the other direction, we again identify between each U_i and Ω_i. For feasible set S which consists of only one item in U_i for each $i \in R$, we can let $\mathbf{X}_S$ be the corresponding vector in $\prod \Omega_i$, and define $f\left(S\right) \triangleq V_{\mathbf{X}_S}(R)$. Definition 6.2.1 requires that we define $f(\cdot)$ over any subset of $U \triangleq \bigcup U_i$. We do this by taking the maximum of $f(\cdot)$ over all feasible subsets. Namely, for set $T \subseteq U$, let $R_T \subseteq [n]$ denote again the set of i's such that $|T \cap U_i| \geq 1$. We set:

$$f(T) \triangleq \max_{\substack{S \subseteq T \\ \forall i \ |S \cap U_i| \leq 1}}^{\text{s.t.}} V_{\mathbf{X}_S}(R_T).$$

Now $f(\cdot)$ is monotone subadditive because it is maximum of monotone subadditive functions. □

6.3 Adaptivity Gaps Beyond Bernoulli Variables for Submodular Functions

We now prove a generalization of Theorem 5.3.1 for submodular functions over independent items, which also gives the optimal adaptivity gaps for monotone submodular functions.

Theorem 6.3.1. *The optimal adaptivity gap for stochastic probing where the constraints are prefix-closed and the function is a monotone non-negative submodular is exactly 2.*

From Theorem 5.3.5, we already know an example that shows a lower bound of 2. To prove the upper bound, our non-adaptive strategy will sample a random root-leaf path using the optimal adaptive strategy tree $\mathcal{T}$. In other words, it performs a "dry-run" of a random walk along the tree without probing anything. In the end it queries all the elements on this random root-leaf path. We argue that its expected value is at least half of the adaptive strategy.

Proof of upper bound in Theorem 6.3.1. We induct over the depth of the tree $\mathcal{T}$, i.e., for any monotone submodular function f and tree $\mathcal{T}$ of depth at most d, we have

$$\mathsf{alg}(\mathcal{T}, f) \geq \frac{1}{2}\mathsf{adap}(\mathcal{T}, f).$$

The base case for $d = 1$ is trivially true because the tree is a single node. For induction, let e be the root node of the optimal decision tree $\mathcal{T}$. Denote by $I := X_e \in U_e$ be (random) type of element e when probed by the adaptive strategy (and also the virtual type of the non-adaptive strategy), while $R := X'_e$ be the (random) true type when probed by the non-adaptive strategy. Also, let $\mathcal{T}_I$ denote the subtree the adaptive strategy goes to when the root element is in type I. This implies

$$\mathsf{adap}(\mathcal{T}, f) = \mathbb{E}_I[f(I) + \mathsf{adap}(\mathcal{T}_I, f_I)] \quad \text{and} \quad \mathsf{alg}(\mathcal{T}, f) = \mathbb{E}_{I,R}[f(R) + \mathsf{alg}(\mathcal{T}_I, f_R)]. \tag{6.1}$$

Now using submodularity and monotonicity of f, we bound the adaptive strategy

$$\begin{aligned}\mathsf{adap}(\mathcal{T}, f) &\leq \mathbb{E}_{I,R}[f(I \cup R) + \mathsf{adap}(\mathcal{T}_I, f_{I\cup R})] \\ &\leq \mathbb{E}_{I,R}[f(I) + f(R) + \mathsf{adap}(\mathcal{T}_I, f_{I\cup R})],\end{aligned}$$

where the last inequality uses that every monotone submodular function is subadditive. Notice that I and R are i.i.d. variables. This along with linearity of expectation implies

$$\mathsf{adap}(\mathcal{T}, f) \leq \mathbb{E}_{I,R}[2 \cdot f(R) + \mathsf{adap}(\mathcal{T}_I, f_{I\cup R})]. \tag{6.2}$$

Next, we lower bound the expected value of the non-adaptive strategy from Eq. (6.1). We use monotonicity of f to get

$$\mathsf{alg}(\mathcal{T}, f) \quad = \quad \mathbb{E}_{I,R}[f(R) + \mathsf{alg}(\mathcal{T}_I, f_R)] \quad \geq \quad \mathbb{E}_{I,R}[f(R) + \mathsf{alg}(\mathcal{T}_I, f_{I\cup R})]. \tag{6.3}$$

Since $f_{I\cup R}$ is also a monotone submodular function over independent elements and $\mathcal{T}_I$ is an adaptive strategy tree of depth at most $d - 1$, by induction hypothesis

$$\mathsf{alg}(\mathcal{T}_I, f_{I\cup R}) \geq \frac{1}{2}\mathsf{adap}(\mathcal{T}_I, f_{I\cup R}).$$

Combining this with Eq. (6.2) and Eq. (6.3), we get

$$\mathsf{alg}(\mathcal{T}, f) \geq \frac{1}{2}\mathsf{adap}(\mathcal{T}, f),$$

which finishes the proof of the upper bound by induction. □

6.4 Adaptivity Gaps for a Weighted Rank Function of a k-Extendible System

For a downward-closed family $\mathcal{F}$, we define its rank function $f_{\mathcal{F}} : 2^V \to \mathbb{R}_{\geq 0}$ to be the largest cardinality subset in $\mathcal{F}$, i.e., $f_{\mathcal{F}}(S) := \max_{T\subseteq S \;\&\; T\in\mathcal{F}} |T| = \max_{T\in\mathcal{F}} |S \cap T|$. In this section we prove our results on the optimal adaptivity gaps of a weighted rank function of a k-extendible system.

Theorem 6.4.1. *The optimal adaptivity gap for CoSP where the constraints are prefix-closed and the function is a weighted rank function of a k-extendible system is between k and $O(k \log k)$. Moreover, for unweighted rank functions, the optimal adaptivity gap is between k and $2k$.*

Since the weighted rank of function of intersection of k-matroids is a k-extendible system, Theorem 6.4.1 implies as a corollary that the adaptivity gaps for this class is at most $\tilde{\Theta}(k)$.

Corollary 6.4.2. *The optimal adaptivity gap for CoSP where the constraints are prefix-closed and the function is a weighted rank function of intersection of k-matroids system is $O(k \log k)$.*

In §6.4.2 we prove the upper bound for unweighted k-extendible systems, and in §6.4.3 we give a reduction from weighted to unweighted k-extendible systems that loses a factor $O(\log k)$ in the adaptivity gap. Our lower bound is presented in §6.4.4.

6.4.1 Preliminaries

We first formally define a k-extendible system.

Definition 6.4.3 (k-extendible system)**.** *A family $\mathcal{F} \ni \emptyset$ such that for every $A \subseteq B \in \mathcal{F}$ and $e \in T$ where $A \cup \{e\} \in \mathcal{F}$, we have that there is a set $Z \subseteq B \setminus A$ such that $|Z| \leq k$ and $B \setminus Z \cup \{e\} \in \mathcal{F}$.*

To simplify our proofs, we define an element $e \in T$ is a *loop* in $\mathcal{F} \subseteq 2^T$ if e is not in any $F \in \mathcal{F}$. Furthermore, given a non-loop element $e \in T$, we define the *contraction* $\mathcal{F}/e$ as $\{F \setminus \{e\} \mid F \in \mathcal{F}, e \in F\}$, i.e., the family of subsets that contain e but with e removed. We also need the following property of k-extendible systems, which intuitively means a set $E \in \mathcal{F}$ hurts at most $k \cdot |E|$ from another set $B \in \mathcal{F}$.

Fact 6.4.4. *Let $\mathcal{F} \subseteq 2^T$ be a k-extendible system. For every $A \subseteq B \in \mathcal{F}$ and $E \subseteq T$ where $A \cup E \in \mathcal{F}$, there exists a set $Z \subseteq B \setminus A$ such that $|Z| \leq k \cdot |E|$ and $B \setminus Z \cup E \in \mathcal{F}$.*

Proof. Enumerate the elements $E = \{e_1, \ldots, e_r\}$ where $r := |E|$ and denote by $E_i := \{e_1, \ldots, e_i\}$ for $0 \leq i \leq r$. Initialize $Z_0 := \emptyset$ and consider the following procedure to construct $Z_1, Z_2, \ldots, Z_r$ that satisfies the invariants $A \subseteq B \setminus Z_i$, $B \setminus Z_i \cup E_i \in \mathcal{F}$ and $|Z_i| \leq k \cdot i$.

In the i^{th} step we have that $A \cup E_{i-1} \cup \{e_i\} \in \mathcal{F}$ by downward-closeness and $A \cup E_{i-1} \subseteq B \setminus Z_{i-1} \cup E_{i-1}$ by the induction hypothesis. Hence by k-extendibility we can find $Z' \subseteq B \setminus (Z_{i-1} \cup A \cup E_{i-1})$ with $|Z'| \leq k$ and where $(B \setminus Z_{i-1} \cup E_{i-1}) \setminus Z' \cup \{e_i\} = B \setminus (Z_{i-1} \cup Z') \cup E_i \in \mathcal{F}$. Set $Z_i := Z_{i-1} \cup Z'$ and note that $|Z_i| \leq |Z_{i-1}| + |Z'| \leq (i-1) \cdot k + k = i \cdot k$. Furthermore, already deduced that $B \setminus Z_i \cup E_i \in \mathcal{F}$ and finally $A \subseteq B \setminus Z_i = B \setminus Z_{i-1} \setminus Z'$ since $Z' \cap A = \emptyset$. We satisfied all stipulations of the induction, hence we report Z_r as the solution. □

6.4.2 Upper Bound of $2k$ for an Unweighted k-Extendible System

Let $\mathcal{T}$ denote the optimal adaptive strategy for maximizing the rank function f of a given k-extendible system $\mathcal{F}$. We prove the following unweighted upper bound of Theorem 6.4.1.

Theorem 6.4.5. *The optimal adaptivity gap for CoSP where the constraints are prefix-closed and the function is an unweighted rank function of a k-extendible system is at most $2k$.*

We use the random walk strategy to convert the adaptive strategy $\mathcal{T}$ into a non-adaptive strategy. To analyze our algorithm, we define a natural *greedy procedure* to select a subset of $A \subseteq T$ that is also in $\mathcal{F} \subseteq 2^T$. First, consider elements of A in an arbitrary order (which can be even determined on the fly). If the currently considered element is a non-loop, it gets contracted in $\mathcal{F}$; otherwise it gets ignored. Any such computed set is in $\mathcal{F}$ and the final output, the number of contracted elements, is denoted by greedy(A). We first show that for k-extendible systems such a greedy procedure produce a k-approximation to the largest subset in $\mathcal{F}$. A similar statement has been proven by Mestre [Mes06].

Lemma 6.4.6. *Let f be a rank function of a k-extendible system $\mathcal{F} \subseteq 2^T$. Fix any subset $A \subseteq T$ and consider the output of the greedy procedure* greedy(A) *with an arbitrary ordering of A. We have that $f(A) \leq k \cdot$*greedy(A)*. Even more, for any $A \subseteq B \subseteq T$ we have that $f(A) \leq k \cdot greedy(B)$.*

Proof. Let $G \subseteq B$ be the set picked by greedy(B). Notice that G is a maximal set in $\mathcal{F}$ (need not be maximum). On the other hand, let OPT $\subseteq A$ be the set picked by $f(A)$, i.e., the maximum set in $\mathcal{F}$ on A. Our goal is to prove $|\text{OPT}| \leq k \cdot |G|$.

Let $C := \text{OPT} \cap G$, note that $G = C \cup (G \setminus C) \in \mathcal{F}$ and $C \subseteq$ OPT, hence by Fact 6.4.4 there is a $Z \subseteq \text{OPT} \setminus C$ with $|Z| \leq k \cdot |G \setminus C| = k \cdot |G| - k \cdot |C|$ such that $\text{OPT} \setminus Z \cup (G \setminus C) = (\text{OPT} \setminus C) \setminus Z \cup G \in \mathcal{F}$. However, since G is a maximal set and $(\text{OPT} \setminus C) \cap G = \emptyset$ we know that $\text{OPT} \setminus C \setminus Z = \emptyset$ and hence $|\text{OPT}| \leq |Z| + |C| \leq k \cdot |G| - k \cdot |C| + |C| = k \cdot |G| - (k-1)|C| \leq k \cdot |G|$. □

Given the above properties of a k-extendible system, we can now prove Theorem 6.4.5.

Proof of Theorem 6.4.5. Let $\mathbf{X}$ and $\mathbf{X}'$ denote the element types for the adaptive and the non-adaptive algorithms, respectively. The adaptive strategy on the optimal decision tree $\mathcal{T}$ gets value $f(X_S)$, where $S \subseteq V$ is the set of probed elements by strategy $\mathcal{T}$ for type vector $\mathbf{X}$. We compare this value to a greedy strategy greedy$(\mathbf{X}_S \cup \mathbf{X}'_S)$ in which

(a) we consider the elements of S in root-to-leaf order in which they appear on the tree and
(b) for any $e \in S$ we first consider $\mathbf{X}'_e$ (the true type) before $\mathbf{X}_e$ (the virtual type) in the greedy order.

Note by Lemma 6.4.6 we have

$$\text{adap}(\mathcal{T}, f) \leq k \cdot \mathbb{E}_{\mathbf{X},\mathbf{X}'}[\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)].$$

By induction on the subtrees, below we prove

$$\mathbb{E}_{\mathbf{X},\mathbf{X}'}[\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)] \leq 2 \cdot \text{alg}(\mathcal{T}, f). \tag{6.4}$$

This finishes the proof of Theorem 6.4.5 because the optimal non-adaptive algorithm has value at least

$$\text{alg}(\mathcal{T}, f) \geq \frac{1}{2} \cdot \mathbb{E}_{\mathbf{X},\mathbf{X}'}[\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)] \geq \frac{1}{2k} \cdot \text{adap}(\mathcal{T}, f).$$

To prove the missing Eq. (6.4), we induct on the height of the tree and $\mathcal{F}$ being any downward-closed family. For consistency, we define the notation of greedy$(\mathcal{T}, f)$ to denote the value of the above greedy strategy when run on $\mathcal{T}$ with a rank function f. Thus, greedy$(\mathcal{T}, f)$ =

$\mathbb{E}_{\mathbf{X},\mathbf{X}'}[\text{greedy}(\mathbf{X}_S \cup \mathbf{X}'_S)]$. Suppose $e \in V$ is the label the root of $\mathcal{T}$. Denote by $I := X_e$ the (random) type of element e when probed by the adaptive strategy (which is also the virtual type of the non-adaptive strategy), and denote $R := X'_e$ the (random) true type when probed by the non-adaptive strategy. Also, let $\mathcal{T}_I$ denote the subtree the adaptive strategy goes to when the root e is in state I. We have

$$\text{greedy}(\mathcal{T}, f) \le \mathbb{E}_{I,R}\Big[f(I \cup R) + \text{greedy}(\mathcal{T}_I, f/(R/I))\Big],$$

where by $(f/R)/I$ we mean the rank function of $\mathcal{F}$ after we first contract R if it a non-loop, and then contract I if it is still a non-loop. Now subadditivity of f gives

$$\begin{aligned}\text{greedy}(\mathcal{T}, f) &\le \mathbb{E}_{I,R}\Big[f(I) + f(R) + \text{greedy}(\mathcal{T}_I, f/(R/I))\Big] \\ &= \mathbb{E}_{I,R}\Big[2 \cdot f(R) + \text{greedy}(\mathcal{T}_I, f/(R/I))\Big], \end{aligned} \tag{6.5}$$

where the last equality uses linearity of expectation as I and R are identically distributed.

Next, we lower bound the value of our non-adaptive algorithm. Although it takes a random root-leaf path and decides the set of elements to retain in the end, we lower bound its value by an online algorithm that greedily selects R (unless it is a loop), however, always also contracts I if it is a non-loop. This gives,

$$\text{alg}(\mathcal{T}, f) \ge \mathbb{E}_{I,R}\Big[f(R) + \text{alg}(\mathcal{T}_I, f/(R/I))\Big]. \tag{6.6}$$

Since $f/(R/I)$ is also a rank function of a downward-closed system and $\mathcal{T}_I$ is an adaptive strategy, by induction hypothesis we have

$$\text{alg}(\mathcal{T}_I, f/(R/I)) \ge \frac{1}{2}\,\text{greedy}(\mathcal{T}_I, f/(R/I)).$$

Combining this with Eq. (6.5) and Eq. (6.6), we get

$$\text{greedy}(\mathcal{T}, f) \le 2 \cdot \text{alg}(\mathcal{T}, f),$$

which proves Eq. (6.4) by induction. □

6.4.3 Reducing Weighted to Unweighted k-Extendible System by Losing $O(\log k)$

We show how to extend the adaptivity gap result for an unweighted k-extendible system to a weighted k-extendible system by losing an $O(\log k)$ factor.

Theorem 6.4.7. *For CoSP over prefix-closed constraints, the adaptivity gap for a weighted rank function of a k-extendible system is at most $32k \log_2 k$.*

Proof. Given a weighted rank function f of a k-extendible system $\mathcal{F} \subseteq 2^T$ over a set of types T, we define f_j for $j \in \mathbb{Z}$ to be an unweighted rank function of the k-extendible system $\mathcal{F}$; however, the new weights are changed such that only the types with original weights in $(2^{j-1}, 2^j]$ participate with new weight of 1, while the other elements have a new weight of 0. Note that this partitions the set of types T into pairwise disjoint *classes*. Notice, we have

$$\text{adap}(\mathcal{T}, f) \leq \sum_j 2^j \cdot \text{adap}(\mathcal{T}, f_j), \tag{6.7}$$

where $adap(\mathcal{T}, f_j)$ denotes the expected value of an adaptive strategy given by the common decision tree $\mathcal{T}$ with respect to the rank function f_j.

Now, since $\text{adap}(\mathcal{T}, f_j)$ is an unweighted k-extendible system problem, we know that a random root-leaf path returns a solution with expected value

$$\text{alg}(\mathcal{T}, f_j) \geq \frac{1}{2k} \cdot \text{adap}(\mathcal{T}, f_j). \tag{6.8}$$

In the following lemma, we show that these non-adaptive solutions for f_j can be combined to obtain a feasible and "high-value" non-adaptive solution for f.

Lemma 6.4.8. *The random-walk non-adaptive algorithm* alg *has expected value*

$$\text{alg}(\mathcal{T}, f) \geq \frac{1}{16 \cdot \log k} \sum_j 2^j \cdot \text{alg}(\mathcal{T}, f_j).$$

Before proving Lemma 6.4.8, we finish the proof of Theorem 6.4.7 by combining it with Eq. (6.8) and Eq. (6.7):

$$\begin{aligned} \text{alg}(\mathcal{T}, f) \geq \frac{1}{16 \cdot \log k} \sum_j 2^j \cdot \text{alg}(\mathcal{T}, f_j) &\geq \frac{1}{32k \log k} \sum_j 2^j \cdot \text{adap}(\mathcal{T}, f_j) \\ &\geq \frac{1}{32k \log k} \cdot \text{adap}(\mathcal{T}, f). \end{aligned}$$ □

Informally, in the proof of Lemma 6.4.8 we combine the unweighted solutions of $\text{alg}(\mathcal{T}, f_i)$ by running a "greedy-optimal" algorithm from the higher weight to the smaller weight classes and fixing the types chosen in earlier classes. Unfortunately, in general such an approach loses an extra factor k in the approximation. To fix this, our second idea is to increase the weight gap between successive classes. We achieve this by combining $O(\log k)$ consecutive classes into a *bucket*, where in each bucket we focus on the class with the largest non-adaptive value. Because of boundary issues, we only take either odd or even buckets.

Proof of Lemma 6.4.8. Let $a \leq b \in \mathbb{Z}$ denote the indices of the smallest and the highest weight classes. We define buckets consisting of $2 \log k$ consecutive classes, where bucket B_i consists of classes $\{b - 2i \log k, b - 2i \log k - 1, \ldots, b - 2(i-1) \log k\}$. For each B_i, let

$$j(i) := \text{argmax}_{j \in B_i} \left\{ 2^j \cdot \text{alg}(\mathcal{T}, f_j) \right\}.$$

Since each bucket has size $2\log k$, this implies

$$\sum_i 2^{j(i)} \cdot \mathrm{alg}(\mathcal{T}, f_{j(i)}) \geq \frac{1}{2 \cdot \log k} \sum_j 2^j \cdot \mathrm{alg}(\mathcal{T}, f_j).$$

Without loss of generality we can assume the odd indices satisfy

$$\sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathrm{alg}(\mathcal{T}, f_{j(i)}) \geq \frac{1}{2} \sum_i 2^{j(i)} \cdot \mathrm{alg}(\mathcal{T}, f_{j(i)}).$$

Otherwise, use the same argument for even indices. Combining the last two equations, we get

$$\sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathrm{alg}(\mathcal{T}, f_{j(i)}) \geq \frac{1}{4 \cdot \log k} \sum_j 2^j \cdot \mathrm{alg}(\mathcal{T}, f_j). \tag{6.9}$$

We now claim that a *greedy-optimal* algorithm has a large value: It goes over classes $j(i)$ in decreasing order of (odd) buckets, but it always selects the maximum independent set (instead of selecting a maximal greedy set) in the current class $j(i)$ given its choices in the previous buckets. This algorithm is, therefore, a combination of greedy and optimal algorithms.

Claim 6.4.9. *Consider an algorithm that goes over the odd numbered buckets in decreasing order of weights and selects the* maximum *set from class $j(i)$ in bucket i such that the resulting set is still feasible in $\mathcal{F}$. (After a set in a class is selected, it gets fixed for all future choices.) The finally chosen set has value at least*

$$\frac{1}{4} \sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathrm{alg}(\mathcal{T}, f_{j(i)}).$$

Proof. The intuition is that for a k-extendible system by Fact 6.4.4 any selected member can "hurt" at most k members from lower buckets. Since we only consider odd numbered buckets, two types in different buckets differ in their weights by at least a factor of $2^{2\log k} = k^2$. Thus, losing k types of lower weight should not significantly impact the value.

Let ℓ be the random variable denoting the leaf reached by the random walk on the decision tree $\mathcal{T}$, and let R be the random set of elements seen by the random-walk non-adaptive strategy on this path. Furthermore, let A_i denote the set of elements picked by the non-adaptive strategy with respect to $f_{j(i)}$, let $A'_i \subseteq A_i$ be the set of elements picked by our greedy-optimal non-adaptive strategy from bucket i, and let $A'_{<i}$ denote $\bigcup_{i'<i \,:\, i' \text{ is odd}} A_{i'}$. In other words, $A'_{<i}$ is the greedy-optimal solution up to bucket number i and A'_i is the maximum subset of A_i such that $A'_i \cup A'_{<i} \in \mathcal{F}$. Note that A_i, A'_i and $A'_{<i}$ are random variables depending on ℓ and R.

Using Fact 6.4.4 on the k-extendible system $\mathcal{F}$ with the preconditions $\emptyset \cup A'_{<i} \in \mathcal{F}$ and $\emptyset \subseteq A_i$, there exists a set Z with $|Z| \leq k \cdot |A'_{<i}|$ such that $A_i \setminus Z \in \mathcal{F}$. Hence, we have

$$|A'_i| \geq |A_i \setminus Z| \geq |A_i| - k \cdot |A'_{<i}|.$$

Multiplying by $2^{j(i)}$ and summing over all odd i gives

$$\sum_{i \text{ is odd}} 2^{j(i)} \cdot |A'_i| \geq \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| - k \cdot \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A'_{<i}|$$

$$= \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| - k \cdot \sum_{i \text{ is odd}} |A'_i| \sum_{i' > i \,:\, i' \text{ is odd}} 2^{j(i')}. \tag{6.10}$$

Now, since every bucket i contains $2 \log k$ classes, where two successive class weights differ by a factor of 2, we know

$$2^{j(i+2)} \leq \frac{2^{j(i)}}{k^2}.$$

Combining this with Eq. (6.10) gives

$$\begin{aligned}\sum_{i \text{ is odd}} 2^{j(i)} \cdot |A'_i| &\geq \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| - k \cdot \sum_{i \text{ is odd}} |A'_i| \sum_{i' > i \,:\, i' \text{ is odd}} \frac{2^{j(i'+2)}}{k^2} \\ &\geq \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| - \sum_{i \text{ is odd}} |A'_i| \cdot 2^{j(i)},\end{aligned}$$

where the last inequality uses

$$\sum_{i' > i \,:\, i' \text{ is odd}} 2^{j(i'+2)} \quad = \quad \sum_{i' \geq i \,:\, i' \text{ is odd}} 2^{j(i')} \quad \leq \quad 2 \cdot 2^{j(i)} \quad \leq \quad k \cdot 2^{j(i)}.$$

After rearranging,

$$\sum_{i \text{ is odd}} 2^{j(i)} \cdot |A'_i| \geq \frac{1}{2} \cdot \sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i|.$$

Notice that by definition of a class, each type in class $j(i)$ has weight at least $2^{j(i)-1}$. Using this fact and taking expectation over ℓ and R, we get

$$\begin{aligned}\mathrm{alg}(\mathcal{T}, f) &\geq \mathbb{E}_{\ell,R}\Big[\sum_{i \text{ is odd}} 2^{j(i)-1} \cdot |A'_i| \Big] \\ &\geq \frac{1}{4} \mathbb{E}_{\ell,R}\Big[\sum_{i \text{ is odd}} 2^{j(i)} \cdot |A_i| \Big] = \frac{1}{4} \sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathrm{alg}(\mathcal{T}, f_{j(i)}),\end{aligned}$$

which finishes the proof of Claim 6.4.9. □

Using Claim 6.4.9, we have

$$\mathrm{alg}(\mathcal{T}, f) \geq \frac{1}{4} \sum_{i \text{ is odd}} 2^{j(i)} \cdot \mathrm{alg}(\mathcal{T}, f_{j(i)}),$$

which combined when with Eq. (6.9) proves Lemma 6.4.8. □

6.4.4 Lower Bounds

We present two very similar lower bound examples: one where the adaptivity gap is $k - o(1)$ for a rank function of an unweighted k-extendible system and another where the adaptivity gap is $\Omega(\sqrt{k})$ for a rank function of an intersection of k matroids.

Example: For generality we work in the Bernoulli setting where each element in V is either active or inactive. Consider a perfect w-ary tree of depth k whose edges correspond to the ground set V (see Figure 6.1). Each edge is active with probability $p > 0$. For any leaf ℓ, let P_ℓ denote the unique path from the root to ℓ. The objective value on any set is the maximum number of edges in the set on the *same* root-leaf path, i.e., for any $S \subseteq V$,

$$f(S) := \max_{\text{leaf } \ell} |P_\ell \cap S|.$$

The feasibility constraints are such that a set of edges can be probed if and only if there exists some root-leaf path P_ℓ such that every probed edge has at least one endpoint on P_ℓ. Note that this implies that a maximum of $w \cdot k$ edges can be probed.

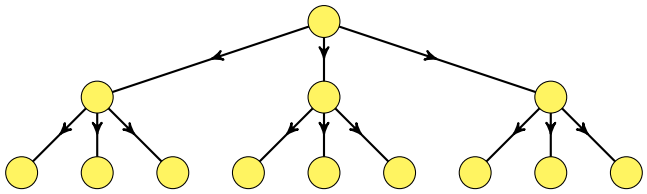

Figure 6.1: Adaptivity gap lower bound example: a $w = 3$-ary tree of depth $k = 2$.

Analysis: Let the adaptive strategy be the following: probe all w edges incident to the root. If any of them is active, start probing the edges directly below the active edge, otherwise below the first edge. Continue recursively until a leaf is reached. On every level, the adaptive strategy has $1 - (1-p)^w$ probability of finding an active edge. Therefore, the expected value of the adaptive strategy is $k \cdot (1 - (1-p)^w)$.

For any non-adaptive strategy, the feasibility constraints imply there exists a root-leaf path P_ℓ such that all probed edges have an endpoint on it. Suppose all $w \cdot k$ edges incident to P_ℓ are probed. The non-adaptive strategy can get value at most 1 from the edges not on P_ℓ and in expectation at most $k \cdot p$ from the edges on P_ℓ. So, the non-adaptive strategy has an expected value of at most $1 + k \cdot p$.

Lower Bound of k for an unweighted k-extendible system

Consider the example described above and set $w := k^4$ and $p := \frac{1}{k^3}$. The function f is trivially a rank function of a k-extendible system because the rank of the system is k, i.e., $f(V) = k$. The adaptive strategy has an expected value

$$k \cdot \left(1 - \left(1 - \frac{1}{k^3}\right)^{k^4}\right) \geq k \cdot \left(1 - \frac{1}{e^k}\right) = k - o(1),$$

whereas any non-adaptive strategy has an expected value at most $1 + \frac{1}{k^2}$. This gives an adaptivity gap of $k - o(1)$.

Lower Bound of $\Omega(\sqrt{k})$ for an unweighted intersection of k matroids

In this section we show how to model the above example as an intersection of $t = k^2$ matroids, yielding an adaptivity gap of $\Omega(\sqrt{t})$ for an intersection of t matroids. Consider the example described above and set $w := k$ and $p := \frac{1}{k}$. The adaptive strategy has an expected value of

$$k \cdot \left(1 - \left(1 - \frac{1}{k}\right)^k\right) \geq k \cdot \left(1 - \frac{1}{e}\right) = \Omega(k)$$

and the non-adaptive strategy gets at most 2 in expectation; so the adaptivity gap is $\Omega(k)$.

All that remains to show is that f can be represented as an intersection of k^2 simple partition matroids. We use the term simple partition matroid for a matroid that partitions the V into multiple parts and a set is independent if it contains at most one element in every part.

Suppose that k is prime and label each node v with a list L_v as follows: the root's label is an empty list (). Let $L(i)$ denote the i^{th} element of the list L and $L + x$ a list equal to L with x appended to it. All the other nodes are labeled recursively: let v be a node with children $\{v_0, v_1, \ldots v_{k-1}\}$. Define $L_{v_i} := L_v + i$. Hence, u is an ancestor of v if and only if L_u is a prefix of L_v, and otherwise $L_u(i) \neq L_v(i)$ for some i.

Let e_v denote the edge/element between v and its parent. We define k^2 partition matroids $M_{i,j}$ for $i \in \{1, 2, \ldots, k\}$ and $j \in \{0, 1, \ldots, k-1\}$. Each $M_{i,j}$ consists of k *big* partitions indexed from 0 to $k-1$, and all other partitions contain only a single element. Let

$$I_v(i,j) := L_v(i)j + d_v \pmod{k}.$$

For a node v on depth $d_v \geq i$, element e_v is in the $I_v(i,j)^{th}$ *big* partition of $M_{i,j}$. For a node v on depth $d_v < i$, e_v is the only element in its partition in $M_{i,j}$.

We claim that f is the rank function of $\mathcal{F} := \bigcap_{i=1}^{k} \bigcap_{j=0}^{k-1} M_{i,j}$, which is an intersection of k^2 matroids. Since $\mathcal{F}$ is an intersection of simple partition matroids, $S \in \mathcal{F}$ if and only if $\{a, b\} \in \mathcal{F}$ for every $a, b \in S$. Now consider two nodes u, v such that $\{e_u, e_v\} \notin \mathcal{F}$. This means $I_u(i,j) = I_v(i,j)$ for some $i \leq d_u, d_v$ and $j \in \{0, 1, \ldots, k-1\}$, which is equivalent to

$$L_u(i) \cdot j + d_u \equiv L_v(i) \cdot j + d_v \pmod{k}.$$

Since k is prime, this holds for some i, j if and only if $d_u = d_v$ (for $j = 0, i = 1$) or $L_u(i) \neq L_v(i)$ for any i. That is, $\{e_u, e_v\} \notin \mathcal{F}$ if and only if u and v are not ancestors of one another, which completes the proof.

6.5 Adaptivity Gaps for Subadditive Functions

In §6.3 we show adaptivity gap for monotone submodular functions is bounded by 2. Now we consider more general classes of functions. We *conjecture* that the adaptivity gap for all subadditive functions is poly-logarithmic in the size of the ground set. Since we know that any subadditive function can be approximated to within a logarithmic factor by an XOS function [Dob07], and every XOS function is subadditive, it suffices to focus on XOS functions. As a

step towards our conjecture, we show a nearly-tight logarithmic adaptivity gap for monotone XOS functions of small "*width*", which we explain below.

As defined in §1.2, a monotone XOS function $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ is one that can be written as the maximum of linear functions. We define the *width* of (the representation of) an XOS function as W, the number of linear functions in this representation. E.g., a width-1 XOS function is just a linear function. In general, even representing submodular functions in this XOS form requires an exponential width [BH11, BDF+12].

In this section we study adaptivity gaps for monotone non-negative XOS functions. To recall, a function is monotone XOS if there exist linear functions $C_1, C_2, \ldots, C_W : V \to \mathbb{R}^{\geq 0}$ such that $f(S) = \max_{i=1}^{W} \{\sum_{e \in S} C_i(e)\}$. To simplify notation we use $C_i(S) := \sum_{e \in S} C_i(e)$ for any i and subset $S \subseteq V$. The width of an XOS function is the smallest number W such that f can be written as the maximum over W linear functions. Let $\mathcal{T}^*$ denote the optimal adaptive strategy. By monotonicity $f^{\max} = f$ and (5.2) gives

$$\mathsf{adap}(\mathcal{T}^*, f) = \mathbb{E}_{\ell \leftarrow \pi_{\mathcal{T}^*}}[f(A_\ell)].$$

The following is our main result in this section.

Theorem 6.5.1. *The stochastic probing problem for monotone XOS functions of width W has adaptivity gap $O(\log W)$ for any prefix-closed constraints. Moreover, there are instances with $W = O(n)$ and adaptivity gap $\Omega(\frac{\log W}{\log \log W})$.*

In §6.5.2, we also present an efficient non-adaptive algorithm for XOS functions of width W that makes $O(W + \log n)$ calls to the following linear oracle.

Definition 6.5.2 (Oracle $\mathcal{O}$)**.** *Given a prefix-closed constraint family $\mathcal{F}$ and linear function $C : V \to \mathbb{R}^{\geq 0}$, oracle $\mathcal{O}(\mathcal{F}, C)$ returns a set $S \in \mathcal{F}$ that maximizes $\sum_{e \in S} C(e)$.*

6.5.1 Adaptivity Gap Upper Bound

We first state a useful property that is used critically later.

Claim 6.5.3 (Subtree property)**.** *For any node u in the optimal adaptive strategy tree $\mathcal{T}^*$, if we consider the subtree $\mathcal{T}'$ rooted at u then the expected value of $\mathcal{T}'$ is at most that of $\mathcal{T}^*$:*

$$\mathsf{adap}(\mathcal{T}', f) \leq \mathsf{adap}(\mathcal{T}^*, f), \text{ when } \mathcal{T}' \text{ is a subtree of } \mathcal{T}^*.$$

This is because otherwise a better strategy would be to go directly to u (probing all the element along the way, so that we satisfy the prefix-closed constraint, but ignoring these elements), and then to run strategy $\mathcal{T}'$.

Proof Idea. The proof consists of three steps. In the first step we argue that one can assume that every coefficient in every linear function C_i is smaller than $\frac{\mathsf{adap}(\mathcal{T}^*, f)}{\log W}$; otherwise, we show a simple non-adaptive strategy that is comparable to the adaptive value obtained from a single active item. The second step shows that by losing a constant factor, one can truncate the tree $\mathcal{T}^*$ to obtain tree $\mathcal{T}$, where the instantiated value at each leaf is at most $2 \cdot \mathsf{adap}(\mathcal{T}^*, f)$. The

combined benefit of these steps is to ensure that root-leaf paths have neither high variance nor too large a value. In the third step, we use Freedman's concentration inequality (which requires the above properties of $\mathcal{T}$) to argue that for any linear function C_i, the instantiated value on a random root-leaf path is close to its mean with high probability. Taking a union bound over the W linear functions, we can then show that (again with high probability), no linear function has an instantiation much more than its mean. Hence, for a random root-leaf path, adap gets value (the maximum instantiation over linear functions) that is not much more than the corresponding mean, which is a lower bound on the non-adaptive value.

Below we use $\text{OPT} = \text{adap}(\mathcal{T}^*, f)$ to denote the optimal adaptive value.

Small and large elements. Define $\lambda := 10^3 \log W$. An element $e \in V$ is called *large* if $\max_{i=1}^{W} C_i(e) \geq h := \frac{\text{OPT}}{\lambda}$; it is called *small* otherwise. Let L be the set of large elements, and let OPT_l (resp., OPT_s) denote the value obtained by tree $\mathcal{T}^*$ from large (resp., small) elements. By subadditivity, we have $\text{OPT}_l + \text{OPT}_s \geq \text{OPT}$.

Lemma 6.5.4 shows that when $\text{OPT}_l \geq \text{OPT}/2$, a simple non-adaptive strategy proves that the adaptivity gap is $O(\log W)$. Then Lemma 6.5.5 shows that when $\text{OPT}_s \geq \text{OPT}/2$, the adaptivity gap is $O(1)$. Choosing between the two by flipping an unbiased coin gives a non-adaptive strategy that proves the adaptivity gap is $O(\log W)$. This would prove the first part of Theorem 6.5.1.

Lemma 6.5.4. *Assuming that $OPT_l \geq OPT/2$, there is a non-adaptive solution of value at least $\Omega(1/\log W) \cdot OPT$. Moreover, there is a solution S satisfying the probing constraint with $h \cdot \min\{\sum_{e \in S \cap L} p_e, 1\} \geq \frac{OPT}{O(\log W)}$.*

Proof. We restrict the optimal tree $\mathcal{T}^*$ to the large elements. So each node in $\mathcal{T}^*$ either contains a large element, or corresponds to making a random choice (and adds no value). The expected value of this restricted tree is OPT_l. We now truncate $\mathcal{T}^*$ to obtain tree $\overline{\mathcal{T}}^*$ as follows. Consider the first active node u on any root-leaf path, remove the subtree below the `yes` (active) arc from u, and assign exactly a value of h to this instantiation. The subtree property (Assumption 6.5.3) implies that the expected value in this subtree below u is at most OPT. On the other hand, just before the truncation at u, the adaptive strategy gains value of $h = \frac{\text{OPT}}{\lambda}$ since it observed an active large element at node u. By taking expectations, we obtain that the value of $\overline{\mathcal{T}}^*$ is at least $\frac{1}{1+\lambda} \cdot \text{OPT}_l$.

Note that $\overline{\mathcal{T}}^*$ is a simpler adaptive strategy. In fact $\overline{\mathcal{T}}^*$ is a feasible solution to the stochastic probing instance with the probing constraint $\mathcal{F}$ and a different objective $g(R) = h \cdot \min\{|R \cap L|, 1\}$ which is the rank function (scaled by h) of the uniform-matroid of rank 1 over all large elements. As any matroid rank function is a monotone submodular function, Theorem 5.1.1 implies that there is a non-adaptive strategy which probes a feasible sequence of elements $S \in \mathcal{F}$, having value $\mathbb{E}_{R \sim S(\mathbf{p})}[g(R)] \geq \frac{1}{3} \cdot \text{adap}(\overline{\mathcal{T}}^*, g) \geq \frac{1}{3} \cdot \frac{\text{OPT}_l}{1+\lambda}$. Note that for any subset $R \subseteq L$ of large elements $f(R) \geq \max_{e \in R}\{\max_{i=1}^{W} C_i(e)\} \geq h \cdot \min\{|R|, 1\} = g(R)$; the first inequality is by monotonicity of f and the second is by definition of large elements. So we have:

$$\mathbb{E}_{R \sim S(\mathbf{p})}[f(R)] \geq \mathbb{E}_{R \sim S(\mathbf{p})}[g(R)] = \Omega(1/\log W) \cdot \text{OPT}_l.$$

It follows that S is the claimed non-adaptive solution for the original instance with objective f.

We now show the second part of the lemma using the above solution S. Note that

$$\frac{\mathrm{OPT}}{O(\log W)} \;=\; \mathbb{E}_{R\sim S(\mathbf{p})}[g(R)] \;=\; h\cdot \mathbb{E}_{R\sim S(\mathbf{p})}[\mathbf{1}(R\cap L\neq\emptyset)] \;\le\; h\cdot \min\Big\{\sum_{e\in S\cap L} p_e,\, 1\Big\},$$

as desired. This completes the proof of Lemma 6.5.4. □

In the rest of this section we prove the following, which implies an $O(\log W)$ adaptivity gap.

Lemma 6.5.5. *Assuming that $OPT_s \ge OPT/2$, there is a non-adaptive solution of value $\Omega(1)\cdot OPT$.*

Proof. We start with the restriction of the optimal tree $\mathcal{T}^*$ to the small elements; recall that OPT_s is the expected value of this restricted tree. The next step is to truncate tree $\mathcal{T}^*$ to yet another tree $\mathcal{T}$ with further useful properties. For any root-leaf path in $\mathcal{T}^*$ drop the subtree below the first node u (including u) where $f(A_u) > 2\cdot\mathrm{OPT}$; here A_u denotes the set of active elements on the path from the root to u. The subtree property (Assumption 6.5.3) implies that the expected value in the subtree below u is at most OPT. On the other hand, before the truncation at u, the adaptive value obtained is more that $2\cdot\mathrm{OPT}$. Hence, the expected value of $\mathcal{T}^*$ obtained at or above the truncated nodes is at least $\frac{2}{3}\cdot\mathrm{OPT}_s$. Finally, since all elements are small and thus the expected value from any truncated node itself is at most $h \le 0.01\cdot\mathrm{OPT}$, the tree $\mathcal{T}$ has at least $(\frac{2}{3}-0.01)\mathrm{OPT}_s \ge \frac{1}{2}\mathrm{OPT}_s$ value. This implies the next claim:

Claim 6.5.6. *Tree $\mathcal{T}$ has expected value at least $\frac{1}{2}\cdot OPT_s \ge \frac{1}{4}\cdot OPT$ and $\max_{\ell\in\mathcal{T}} \max_{i=1}^{W}\{C_i(P_\ell)\} \le 2\cdot OPT$.*

Next, we want to claim that each linear function behaves like its expectation (with high probability) on a random path down the tree. For any $i \in [W]$ and root-leaf path P_ℓ in $\mathcal{T}$, define

$$\mu_i(P_\ell) := \mathbb{E}_{R\sim V(\mathbf{p})}[C_i(R\cap P_\ell)] = \sum_{v\in P_\ell}\Big(p_{\mathrm{elt}(v)}\cdot C_i(\mathrm{elt}(v))\Big).$$

Claim 6.5.7. *For any $i \in [W]$,*

$$\Pr_{\ell\leftarrow\pi_{\mathcal{T}}}\Big[|C_i(A_\ell)-\mu_i(P_\ell)| > 0.1\ OPT\Big] \le \frac{1}{W^2}.$$

Proof. Our main tool in this proof is the following concentration inequality for martingales.

Theorem 6.5.8 (Freedman, Theorem 1.6 in [Fre75])**.** *Consider a real-valued martingale sequence $\{X_t\}_{t\ge0}$ such that $X_0 = 0$, and $\mathbb{E}[X_{t+1} \mid X_t, X_{t-1}, \ldots, X_0] = 0$ for all t. Assume that the sequence is uniformly bounded, i.e., $|X_t| \le M$ almost surely for all t. Now define the predictable quadratic variation process of the martingale to be $W_t = \sum_{j=0}^{t}\mathbb{E}\left[X_j^2 \mid X_{j-1}, X_{j-2}, \ldots, X_0\right]$ for all $t \ge 1$. Then for all $\ell \ge 0$ and $\sigma^2 > 0$, and any stopping time τ we have*

$$\Pr\left[\Big|\sum_{j=0}^{\tau} X_j\Big| \ge \ell \text{ and } W_\tau \le \sigma^2\right] \le 2\exp\left(-\frac{\ell^2/2}{\sigma^2 + M\ell/3}\right).$$

Consider a random root-leaf path $P_\ell = \langle r = v_0, v_1, \ldots, v_\tau = \ell \rangle$ in $\mathcal{T}$, and let $e_t = \text{elt}(v_t)$. Now define a sequence of random variables $X_0, X_1, \ldots,$ where

$$X_t = (\mathbf{1}_{e_t \in A} - p_{e_t}) \cdot C_i(e_t).$$

Let $\mathcal{H}_t$ be a filter denoting the sequence of variables before X_t. Observe that $\mathbb{E}[X_t \mid \mathcal{H}_t] = 0$, which implies $\{X_t\}$ forms a martingale. Clearly $|X_t| \le |C_i(e_t)| \le h$. Now,

$$\begin{aligned}
&\sum_{j=0}^{t} \mathbb{E}\left[|X_j| \mid \mathcal{H}_j\right] \\
&\le \sum_{j=0}^{t} (p_{e_t}(1-p_{e_t}) + (1-p_{e_t})p_{e_t}) \cdot C_i(e_t) \\
&\le \frac{1}{2}\sum_{j=0}^{t} C_i(e_t) \quad \le \quad \frac{1}{2} \cdot \max_{\ell} \max_{i=1}^{W} C_i(P_\ell) \quad \le \quad \text{OPT},
\end{aligned}$$

where the last inequality is by Claim 6.5.6. We use $|X_j| \le h = \frac{\text{OPT}}{\lambda}$ and the above equation to bound the path variance,

$$\sum_{j=0}^{t} \mathbb{E}\left[X_j^2 \mid \mathcal{H}_j\right] \quad \le \quad h \cdot \sum_{j=0}^{t} \mathbb{E}\left[|X_j| \mid \mathcal{H}_j\right] \quad \le \quad \frac{\text{OPT}^2}{\lambda}.$$

Applying Theorem 6.5.8, we get

$$\begin{aligned}
&\Pr\left[\Big|\sum_{j=0}^{\tau} X_j\Big| > 0.1\ \text{OPT}\right] \\
&= \Pr[|C_i(A_\ell) - \mu_i(P_\ell)| > 0.1\ \text{OPT}] \\
&\le 2\exp\left(-\frac{(0.1\ \text{OPT})^2/2}{\text{OPT}^2/(\lambda) \ + \ (\text{OPT}/(\lambda)) \cdot (0.1\ \text{OPT})/3}\right) \\
&\le \frac{1}{W^2}.
\end{aligned}$$

This completes the proof of Claim 6.5.7. □

Now we can finish the proof of Lemma 6.5.5. We label every leaf ℓ in $\mathcal{T}$ according to the linear function C_i that achieves the value $f(A_\ell)$, breaking ties arbitrarily. I.e., for leaf ℓ we define

$$c_\ell^{\max} := C_i, \qquad \text{where } C_i(A_\ell) = f(A_\ell).$$

Also define $\mu_\ell^{\max} := \mu_i$ for i as above. Using Claim 6.5.7 and taking a union bound over all $i \in [W]$,

$$\Pr_{\ell \leftarrow \pi_{\mathcal{T}}}\left[\left|c_\ell^{\max}(A_\ell) - \mu_\ell^{max}(P_\ell)\right| > 0.1\ \text{OPT}\right] \quad \le \quad \frac{1}{W}. \tag{6.11}$$

Consider the natural non-adaptive solution which selects $\ell \leftarrow \pi_{\mathcal{T}}$ and probes all elements in P_ℓ. This has expected value at least:

$$\begin{aligned}
&\mathbb{E}_{\ell \leftarrow \pi_{\mathcal{T}}}\left[\mu_\ell^{max}(P_\ell)\right] \\
&\overset{(6.11)}{\geq} \quad \mathbb{E}_{\ell \leftarrow \pi_{\mathcal{T}}}\left[c_\ell^{max}(A_\ell)\right] - 0.1\ \mathrm{OPT} - \frac{1}{W}(2\ \mathrm{OPT}) \\
&\overset{(\text{Claim } 6.5.6)}{\geq} \quad (0.15 - \frac{2}{W}) \cdot \mathrm{OPT}.
\end{aligned}$$

If $W \geq 20$ then we obtain the desired non-adaptive strategy. The remaining case of $W < 20$ is trivial: the adaptivity gap is 1 for a single linear function, and taking the best non-adaptive solution among the W possibilities has value at least $\frac{1}{W} \cdot \mathrm{OPT}$. This completes the proof of Lemma 6.5.5. □

Let us record an observation that will be useful for the non-adaptive algorithm.

Remark 6.5.9. *Observe that the above proof shows that when $OPT_s \geq OPT/2$, there exists a path Q in $\mathcal{T}^*$ (i.e. Q satisfies the probing constraints) and a linear function C_j with mean value $\mathbb{E}_{R\sim Q(\mathbf{p})}[C_j(R)] = \Omega(OPT)$.*

6.5.2 Polynomial Time Non-adaptive Algorithm

Consider any instance of the stochastic probing problem with a width-W monotone XOS objective and prefix-closed constraint $\mathcal{F}$. Our non-adaptive algorithm is the following (here $\lambda = 10^3 \log W$ as in §6.5.1).

Algorithm 9 Non-adaptive Algorithm for XOS functions

define $m := \max_{e\in V}\{p_e \cdot \max_{i\in[W]} C_i(e)\}$
for $j \in \{0,\ldots,1+\log n\}$ **do**
  define $\mathbf{b}_j$ as follows

$$\mathbf{b}_j(e) = \begin{cases} p_e & \text{if } \max_{i\in[W]}\{C_i(e)\} \geq \frac{2^j m}{\lambda} \\ 0 & \text{otherwise} \end{cases}.$$

  set $T_j \leftarrow O(\mathcal{F}, \mathbf{b}_j)$.
  set $v(T_j) \leftarrow \frac{2^j m}{\lambda} \cdot \min\{\mathbf{b}_j(T_j), 1\}$.
end for
for $i \in \{1,\ldots,W\}$ **do**
  define $\mathbf{c}_i$ with $\mathbf{c}_i(e) = p_e \cdot C_i(e)$
  $S_i \leftarrow O(\mathcal{F}, \mathbf{c}_i)$ and $v(S_i) \leftarrow \mathbf{c}_i(S_i)$.
end for
return set $S \in \{S_1,\ldots,S_W,T_0,T_1,\ldots,T_{1+\log n}\}$ that maximizes $v(S)$.

Case I: $\mathbf{OPT}_l \geq \mathbf{OPT}/2$. Lemma 6.5.4 shows that in this case it suffices to consider only the set of large elements and to maximize the probability of selecting a single large element. While we do not know OPT, and the large elements are defined in terms of OPT, we do know $m = \max_{e \in V}\{p_e \cdot \max_{i \in [W]} C_i(e)\} \leq \text{OPT} \leq n \cdot m$. In the above algorithm, consider the value of $j \in \{0, \ldots, 1 + \log n\}$ when $2^j \cdot m/\lambda$ is between h and $2h$. Let L denote the set of large elements; note that these correspond to the elements with positive $\mathbf{b}_j(e)$ values. By the second part of Lemma 6.5.4, the solution T_j returned by the oracle will satisfy $v(T_j) \geq \text{OPT}/O(\log W)$. Now interpreting this solution T_j as a non-adaptive solution, we get an expected value at least:

$$\begin{aligned} & h \cdot \mathbb{E}_{R \sim T_j(\mathbf{p})}[\mathbf{1}(R \cap L \neq \emptyset)] \\ = \; & h \cdot \left(1 - \Pi_{e \in T_j}(1 - \mathbf{b}_j(e))\right) \quad \geq \quad h \cdot \left(1 - e^{-\mathbf{b}_j(T_j)}\right) \\ \geq \; & (1 - 1/e)h \cdot \min\{\mathbf{b}_j(T_j), 1\} \\ = \; & (1 - 1/e) \cdot v(T_j) \quad \geq \quad \frac{\text{OPT}}{O(\log W)}. \end{aligned}$$

Case II: $\mathbf{OPT}_s \geq \mathbf{OPT}/2$. In this case Remark 6.5.9 following the proof of Lemma 6.5.5 shows that there exists a solution Q satisfying the probing constraints $\mathcal{F}$ and a linear function C_j with mean value $\mathbf{c}_j(Q) = \mathbb{E}_{R \sim Q(\mathbf{p})}[C_j(R)] = \Omega(\text{OPT})$. Since the above algorithm calls $\mathcal{O}(\mathcal{F}, \mathbf{c}_i)$ for each $i \in [W]$ and chooses the best one, it will return a set with value $\Omega(\text{OPT})$.

6.5.3 Adaptivity Gap Lower Bound

Consider a k-ary tree of depth k, whose edges are the ground set. Each edge/element has probability $p_e = \frac{1}{k}$. Here, imagine $k = \Theta(\frac{\log n}{\log \log n})$, so that the total number of edges is $\sum_{i=1}^{k} k^i = n$. For each of the k^k leaves l, consider the path P_l from the root to that leaf. The XOS function is $f(S) := \max_l |P_l \cap S|$. Note that the width $W = \Theta(n)$ in this case.

Suppose the probing constraint is the following prefix-closed constraint: there exists a root-leaf path P_l such that all probed edges have at least one endpoint on this path. This implies that we can probe at most k^2 edges.

- For an adaptive strategy, probe the k edges incident to the root. If any one of these happens to be active, start probing the k edges at the next level below that edge. (If none were active, start probing the edges below the left-most child, say.) Each level will have at least one active edge with probability $1 - (1 - \frac{1}{k})^k \geq 1 - 1/e$, so we will get an expected value of $\Omega(k)$.
- Now consider any non-adaptive strategy: it is specified by the path P_l whose vertices hit every edge that is probed. There are k^2 such edges, we can probe all of them. But the XOS function can get at most 1 from an edge not on P_l, and it will get at most $k \cdot 1/k = 1$ in expectation from the edges on P_l.

This shows a gap of $\Omega(k) = \Omega\left(\frac{\log n}{\log \log n}\right)$ for XOS functions with a prefix-closed (in fact subset-closed) probing constraint.

A Lower Bound for Cardinality Constraints

We can show a near-logarithmic lower bound for XOS functions even for the most simple cardinality constraints. The setup is the same as above, just the constraint is that a subset of at most k^2 edges can be probed.

- The adaptive strategy remains the same, with expected value $\Omega(k)$.
- We claim that any non-adaptive strategy gets expected value $O(\log k)$. Such a non-adaptive strategy can fix any set S of k^2 edges to probe. For each of these edges, choose an arbitrary root-leaf path passing through it, let T be the edges lying in these k^2 many root-leaf paths of length k. So $|T| \leq k^3$. Let us even allow the strategy to probe all the edges in T—clearly this is an upper bound on the non-adaptive value.

  The main claim is that the expected value to be maximized when T consists of k^2 many disjoint paths. (The k-ary tree does not have these many disjoint paths, but this is just a thought-experiment.) The claim follows from an inductive application of the following simple fact.

  Fact 6.5.10. *Given independent non negative random variables X, X', Y, Z, where X' and X have the same distribution, the following holds:*

$$\begin{aligned} & \mathbb{E}_{X,Y,Z}[\max\{X+Y, X+Z\}] \\ \leq\ & \mathbb{E}_{X,X',Y,Z}[\max\{X+Y, X'+Z\}]. \end{aligned}$$

  Proof. Follows from the fact that $\{\max\{X+Y, X+Z\} > c\} \subseteq \{\max\{X+Y, X'+Z\} > c\}$. □

  Finally, for any path with k edges, we expect to get value 1 in expectation. The probability that any one path gives value $c \log k$ is $\frac{1}{k^3}$, for suitable constant c. So a union bound implies that the maximum value over k^2 path is at most $c \log k$ with probability $1/k$. Finally, the XOS function can take on value at most k, so the expected value is at most $1 + c \log k$.

This shows an adaptivity gap of $\Omega\left(\frac{k}{\log k}\right) = \Omega\left(\frac{\log n}{(\log \log n)^2}\right)$ even for cardinality constraints.

Chapter 7

The Price of Information under Constraints

7.1 Introduction

In this chapter we discuss how our results on the Price of Information and the Markovian Price of Information in Chapters 3 and 4, respectively, can be extended to also handle some additional "constraints". In particular, we describe techniques to handle three types of constraints: (i) probing, (ii) commitment, and (iii) sampling.

7.1.1 Probing Constraints

Consider a generalization of the Pandora's box problem where besides paying probing prices, we can only probe at most k elements. This is different from the model discussed in §3.1.1 as earlier we could probe any set of elements but could get value for only one item. In general, we could be given a downward-closed constraints $\mathcal{J}$ that allow us to only probe a subset of elements Probed $\in \mathcal{J}$. We now formally define our problem.

Constrained Utility-Maximization Suppose we are given downward-closed probing constraints $\mathcal{J} \subseteq 2^V$ and probability distributions of independent non-negative variables X_i for $i \in V$. To find X_i we have to pay a probing price π_i. The goal is to probe a set of elements Probed $\in \mathcal{J}$ to maximize the expected *utility*:

$$\mathbb{E}\Big[\max_{i\in\text{Probed}}\{X_i\} - \sum_{i\in\text{Probed}}\pi_i\Big].$$

Depending on the family of constraints $\mathcal{J}$, one can design efficient approximation algorithms for some settings of the above problem. In [Sin18] we show the following result for this problem.

Theorem 7.1.1. *If the constraints $\mathcal{J}$ form an ℓ-system then the constrained utility-maximization problem has a $2(\ell+1)$-approximation algorithm.*

Since the cardinality (or any matroid) constraint forms a 1-system, Theorem 7.1.1 gives a 6-approximation algorithm for the Pandora's box problem under a cardinality probing constraint.

The above constrained utility-maximization problem can be used as a framework to study variants of Pandora's box. Consider a *set-probing utility-maximization* problem where the costs are on subsets of random variables, instead of individual variables: for a subset $S \subseteq V$, we pay price π_S to simultaneously probe all X_i for $i \in S$. Thus, to find X_i, we can now probe a "small" or a "large" set containing i, but at different prices. (Note that when we probe multiple sets containing i, we find the same value X_i and not a fresh sample from the distribution.)

Theorem 7.1.2. *The set-probing utility-maximization problem has a $2(\ell + 1)$-approximation efficient algorithm, where ℓ is the size of the largest set in $\mathcal{S}$. Moreover, no efficient algorithm can be $o(\ell/\log \ell)$-approximation, unless $P = NP$.*

7.1.2 Commitment Constraints

Consider the Markovian PoI model defined in §4.1.1 with an additional restriction that whenever we abandon advancing a Markov system, we need to *immediately* and *irrevocably* decide if we are selecting this element into the final solution $\mathbb{I}$. Since we only select ready elements, any element that is not ready when we abandon its Markov system is automatically discarded. We call this constraint *commitment*. The benchmark to which we compare our algorithm is the optimal policy *without* the commitment constraint.

We study the Utility Maximization problem in the Dag model with the commitment constraint. Our algorithms make use of the *online contention resolution schemes* (OCRSs) proposed in [FSZ16]. OCRSs address our problem in the Free-Info world[1] (i.e., we can see the realization of the r.v.s for free, but there is the commitment constraint). Constant factor "selectable" OCRSs are known for several constraint families such as matroids and matchings [FSZ16]. In §7.3, we show how to adapt any such OCRS to Markovian PoI with commitment.

Theorem 7.1.3. *For an additive objective, if there exists a $1/\alpha$-selectable OCRS ($\alpha \geq 1$) for a packing constraint $\mathcal{F}$, then there exists an α-approximation algorithm for the corresponding* Dag-Utility Maximization *problem with commitment.*

The proof of this result is based on a new LP relaxation (inspired from [GM07]) to bound the optimum utility of a Markovian PoI game *without* commitment. Although this relaxation is not exact even for Pandora's box (and cannot be used to design optimal strategies in Corollary 4.1.4), it turns out to suffice for our approximation guarantees. We use an OCRS to round this LP with only a small loss in the utility, while respecting the commitment constraint.

Remark 7.1.4. *We do not consider* Disutility Minimization *problem under the commitment constraint. This is because it captures prophet inequalities in a minimization setting where no polynomial approximation factors are possible even for i.i.d. r.v.s [EHLM17, Theorem 4].*

[1] In fact, OCRSs treat a variation of the problem where an adversary can choose the order in which the elements are tried. This, of course, handles the present problem in which we may choose the order.

7.1.3 Sampling Constraints

In practical applications, the parameters of Markov systems (i.e., transition probabilities, values, and prices) are not known exactly but are *estimated* by statistical sampling. In this setting, the *true parameters*, which govern how each Markov system evolves, differ from the estimated parameters that the algorithm uses to make decisions. This raises a natural question: how well does an adapted Frugal algorithm do when the true and the estimated parameters differ? We would hope to design a *robust* algorithm, meaning small additive estimation errors cause only small additive error in the utility objective.

In §7.4, we show that in the important special case where the Markov chain corresponding to each element is formed by a *directed acyclic graph* (Dag), a minor adaptation of our strategy in Theorem 4.1.3 is robust. This Dag assumption turns out to be necessary as similar results do not hold for general Markov chains (see an example in Appendix 7.4.5).

Theorem 7.1.5 (Informal statement of Theorem 7.4.2)**.** *If there exists an α-approximation* Frugal *algorithm ($\alpha \geq 1$) for a packing problem with an additive objective function, then it suffices to estimate the input parameters of a* Dag-Markovian PoI *game within an additive error of $\epsilon/$* poly*, where* poly *is some polynomial in the size of the input, to design a strategy with utility at least $\frac{1}{\alpha} \cdot OPT - \epsilon$, where OPTis the utility of the optimal policy that knows all the* true *input parameters.*

Specifically, we show the same strategy as in Theorem 4.1.3 works with one minor change: each time we advance an element's Markov system, we slightly increase that element's grade. Roughly, this is because we need to be "optimistic" in our estimate of each element's true grade.

The analogous result for Disutility Minimization is also true.

7.2 Probing Constraints via Adaptivity Gaps

In this section we consider a generalization of the Pandora's box problem where we have an additional constraint that allows us to only probe a subset of elements Probed $\subseteq V$ that belongs to a downward-closed constraint $\mathcal{J}$ (e.g., a cardinality constraint allowing us to probe at most k elements). We restate our main result for the constrained utility-maximization problem (see problem definition in §7.1.1).

Theorem 7.1.1. *If the constraints $\mathcal{J}$ form an ℓ-system then the constrained utility-maximization problem has a $2(\ell + 1)$-approximation algorithm.*

Similar to §3.2, our strategy to prove Theorem 7.1.1 is to bound the constrained utility-maximization problem in the PoI world (a mixed-sign objective function) with a surrogate constrained utility-maximization problem in the Free-Info world (i.e., where $\pi_i = 0$ for all $i \in V$). This latter problem turns out to be the same as the *stochastic probing* problem, which we define below in the form that is relevant to this paper.

Stochastic Probing Given downward-closed probing constraints $\mathcal{J} \subseteq 2^V$ and probability distributions of independent non-negative variables Y_i for $i \in V$, the stochastic probing problem is to *adaptively* probe a subset Probed $\in \mathcal{J}$ to maximize the expected value $\mathbb{E}[\max_{i \in \text{Probed}}\{Y_i\}]$.

Here, *adaptively* means that the decision to probe which element next can depend on the outcomes of the already probed elements.

7.2.1 Reducing to Non-adaptive Stochastic Probing

The following lemma bounds the expected utility of the constrained utility-maximization problem in the PoI world by the expected value of a stochastic probing problem in the Free-Info world.

Lemma 7.2.1. *The expected utility of the optimal strategy for the constrained utility-maximization problem is at most the expected value of the optimal adaptive strategy for a stochastic probing problem with the same constraints $\mathcal{J}$ and where the random variables Y_i for $i \in V$ have probability distributions $Y_i^{\max}$ (recollect, Defn 3.2.2).*

Proof of Lemma 7.2.1. We start by noticing that the optimal strategy for our problem is given by a decision tree T with leaves l. For any root leaf path P_l, the value of the optimal strategy is $\max_{i \in P_l}\{X_i\} - \pi(P_l)$. Thus the expected value of the optimal strategy is

$$\mathbb{E}_l\Big[\max_{i \in P_l}\{X_i\} - \pi(P_l)\Big]. \tag{7.1}$$

Now we design an adaptive strategy for the stochastic probing problem on random variables $Y_i^{\max}$ with expected value at least as given by (7.1). Consider the adaptive strategy that follows the same decision tree T (note, it pays no probing price). The expected value of such an adaptive strategy is given by

$$\mathbb{E}_l\Big[\max_{i \in P_l}\{Y_i^{\max}\}\Big]. \tag{7.2}$$

The following claim finishes the proof of this lemma.

Claim 7.2.2.

$$\mathbb{E}_l\Big[\max_{i \in P_l}\{X_i\} - \pi(P_l)\Big] \le \mathbb{E}_l\Big[\max_{i \in P_l}\{Y_i^{\max}\}\Big].$$

Proof. Let $A_l(i)$ and $\mathbf{1}_{i \in P_l}$ denote indicator variables that element i is selected and probed on a root-leaf path P_l of the optimal strategy, respectively. Note that these indicator variables are correlated. The expected utility of the optimal strategy equals

$$\begin{aligned}
\mathbb{E}_l\Big[\max_{i \in P_l}\{X_i\} - \pi(P_l)\Big] &= \mathbb{E}_l\Big[\sum_i \big(A_l(i)X_i - \mathbf{1}_{i \in P_l}\pi_i\big)\Big] \\
&= \mathbb{E}_l\Big[\sum_i \big(A_l(i)X_i - \mathbf{1}_{i \in P_l}\mathbb{E}_i[(X_i - \tau_i^{\max})^+]\big)\Big] \\
&= \mathbb{E}_l\Big[\sum_i \big(A_l(i)X_i - \mathbf{1}_{i \in P_l}(X_i - \tau_i^{\max})^+\big)\Big]
\end{aligned}$$

since X_i is independent of $\mathbf{1}_{i\in P_l}$. Now using $A_l(i) \le \mathbf{1}_{i\in P_l}$, we have

$$\begin{aligned}\mathbb{E}_l\Big[\max_{i\in P_l}\{X_i\} - \pi(P_l)\Big] &\le \mathbb{E}_l\Big[\sum_i \left(A_l(i)X_i - A_l(i)(X_i - \tau_i^{\max})^+\right)\Big] \\ &= \mathbb{E}_l\Big[\sum_i A_l(i)Y_i^{\max}\Big] \\ &\le \mathbb{E}_l\Big[\max_{i\in P_l}\{Y_i^{\max}\}\Big],\end{aligned}$$

where the last inequality uses $\sum_{i\in P_l} A_l(i) \le 1$.

□

The following Lemma 7.2.3 shows that we can further simplify the stochastic probing problem in the Free-Info world by focusing only on finding the best *non-adaptive* strategy for this problem, i.e. the problem of finding $\operatorname{argmax}_{\text{Probed}\in\mathcal{J}}\{\max_{i\in\text{Probed}}\{Y_i\}\}$. This is because the *adaptivity gap*—ratio of the expected values of the optimal adaptive and optimal non-adaptive strategies—for the stochastic probing problem is small.

Lemma 7.2.3. *The adaptivity gap for the stochastic probing problem is at most* 2.

Lemma 7.2.3 follows from Theorem 6.3.1. It tells us about the existence of a feasible set $S \in \mathcal{J}$ such that $\mathbb{E}[\max_{i\in S}\{Y_i^{\max}\}]$ is at least half of the optimal adaptive strategy for the stochastic probing problem. Suppose we have an oracle to (approximately) find this feasible set S for probing constraints $\mathcal{J}$.

Assumption 7.2.4. *Suppose there exists an oracle that finds $S \in \mathcal{J}$ that β approximately maximizes the non-adaptive stochastic probing solution.*

The above assumption is justifiable as it is a constrained submodular maximization problem that we know how to approximately solve for some many constraint families $\mathcal{J}$, e.g., an ℓ-system.

Lemma 7.2.5 (Greedy Algorithm [FNW78, CCPV11])**.** *The greedy algorithm has an $(\ell + 1)$-approximation for monotone submodular maximization over an ℓ-system.*

Finally, we need to show that given S there exists an efficient adaptive strategy in the PoI world with expected utility $\mathbb{E}[\max_{i\in S}\{Y_i^{\max}\}]$. But this is exactly the Pandora's box problem for which we know that Weitzman's index-based policy is optimal with expected utility $\mathbb{E}[\max_{i\in S}\{Y_i^{\max}\}]$. The above discussion can be summarized in the following theorem.

Theorem 7.2.6. *Given a β-approximation oracle for monotone submodular maximization over downward-closed constraints $\mathcal{J}$, there exists a 2β-approximation algorithm for constrained utility-maximization.*

Combining Lemma 7.2.5 and Theorem 7.2.6, we get Theorem 7.1.1 as a corollary.

7.2.2 An Application to the Set-Probing Utility-Maximization Problem

In this section we see an application of the constrained utility-maximization framework to the *set-probing utility-maximization* problem defined in §7.1.1. This problem is a generalization of

Pandora's box where we pay a price to simultaneously find values of a set of random variables. We restate Theorem 7.1.2 for convenience.

Theorem 7.1.2. *The set-probing utility-maximization problem has a $2(\ell+1)$-approximation efficient algorithm, where ℓ is the size of the largest set in $\mathcal{S}$. Moreover, no efficient algorithm can be $o(\ell/\log \ell)$-approximation, unless $P = NP$.*

The remaining section discusses the approximation algorithm. See the hardness proof in §7.2.3.

We first observe that WLOG one can assume that the given sets $\mathcal{S} = \{S_1, \ldots, S_m\}$ are downward-closed, i.e., if $S \in \mathcal{S}$ then any subset $T \subseteq S$ is also in $\mathcal{S}$. This is because a simple way to ensure downward-closedness is by adding every subset of S_j for the same price π_j into $\mathcal{S}$. Intuitively, this is equivalent to paying for the original set but choosing not to see the outcome of some of the random variables in it.

To construct our algorithm, we imagine solving a constrained utility-maximization problem. The random variables of this problem are indexed by sets $S \in \mathcal{S}$: variable X_S has value $\max_{i \in S}\{X_i\}$ and has price π_S. The problem is to adaptively probe some elements such that the sets corresponding to them are pairwise disjoint (a downward-closed constraint $\mathcal{J}$), while the goal is to maximize the utility that is given by max element value minus the total probing prices. Intuitively, the reason we need disjointness is to ensure independence between sets in our analysis as disjoint sets of random variables take values independently. We make the following simple observation.

Observation 7.2.7. *The optimal adaptive policy for this constrained utility-maximization problem with disjointness constraints is the same as the unconstrained set-probing utility-maximization problem.*

Given the above observation and noting that disjointness constraints are downward-closed, we want to use Theorem 7.2.6 to reduce our problem into a non-adaptive optimization problem. Although it appears that this is not possible because Theorem 7.2.6 is only for independent variables, and variables corresponding to non-disjoint sets are not independent. Fortunately, the proof of Theorem 7.2.6 only uses independence of random variables along any root-leaf path of the decision tree. Since our probing constraints ensure that the probed sets are disjoint, we get variables X_S along any root-leaf path to be independent, thereby allowing us to use Theorem 7.2.6. The final part in the proof of Theorem 7.1.2 is an approximation algorithm for this non-adaptive constrained utility-maximization problem.

Lemma 7.2.8. *There exists an efficient $(\ell+1)$-approximation algorithm for the non-adaptive problem of finding a family $\mathcal{S}' \subseteq \mathcal{S}$ of disjoint sets to maximize $E[\max_{S \in \mathcal{S}'}\{Y_S^{\max}\}]$.*

Proof. Observe that the function $g(\mathcal{S}') = E[\max_{S \in \mathcal{S}'}\{Y_S^{\max}\}]$ is submodular. Also, the disjointness constraints can be viewed as an ℓ-system constraints since each set S has size at most ℓ. Thus we can view the non-adaptive problem as maximizing a submodular function over an ℓ-system, where we know by Lemma 7.2.5 that the greedy algorithm has an $(\ell+1)$-approximation.

Moreover, to implement the greedy algorithm efficiently, we note that although $\mathcal{S}$ may contain an exponential number of elements, the initial set system $\mathcal{S}$ was polynomial sized (before we made $\mathcal{S}$ downward-closed). For sets A, B available at the same price, where $A \subseteq B$, it is ob-

vious that the greedy algorithm will always choose B before A. Hence, at every step our greedy algorithm only needs to consider the original sets, which are only polynomial in number, and select the set with the best marginal value. Since, this can be done in polynomial time, this completes the proof of Theorem 7.1.2. □

7.2.3 Hardness for the set-probing problem

To prove that no polynomial time algorithm for the set-probing problem can be $o(\ell/\log \ell)$-approximation, unless **P** = **NP**, we reduce ℓ-set packing problem into an instance of the set-probing problem. Given an ℓ-set packing instance with sets $S_1, S_2, \ldots, S_m$, each of size ℓ, we create the following set-probing problem. For every element $i \in \bigcup_j S_j$, w.p. $\frac{1}{n^3}$ variable X_i takes value 1, and is 0, otherwise. Also, let each set S_j have price $\frac{(\ell-0.5)}{n^3}$.

Since probability of two elements taking value 1 is really small $O(1/n^4)$, we see that it only makes sense to probe sets where none of the elements have been already probed: even if a single element is probed before, expected value from probing the set is at most $\frac{(\ell-1)}{n^3} - \frac{(\ell-0.5)}{n^3} = \frac{-0.5}{n^3} < 0$. Hence, $E[Opt]$ = (Max # disjoint sets) $\cdot \frac{0.5}{n^3}$. But this is exactly the ℓ-set packing problem and we know that unless $P = NP$, no poly time algorithm can be $o(\ell/\log \ell)$-approximation [HSS06].

7.3 Commitment Constraints via Linear Programs

In this section we handle the commitment constraint defined in §7.1.2 for the Dag-Utility Maximization problem. Specifically, we prove Theorem 7.1.3 that shows how to use an OCRS to design an approximation algorithm for our problem.

Theorem 7.1.3. *For an additive objective, if there exists a $1/\alpha$-selectable OCRS ($\alpha \geq 1$) for a packing constraint $\mathcal{F}$, then there exists an α-approximation algorithm for the corresponding* Dag-Utility Maximization *problem with commitment.*

Recollect that in the above theorem we compare to the optimal strategy without the commitment constraint. [FSZ16] designed selectable OCRS for several packing constraint families: $1/4$ for matroids, $1/(2e)$ for matchings, and $\Omega(1/k)$ for intersection of k matroids.

Our proof proceeds in two steps. In §7.3.1, we give an LP relaxation to upper bound the optimum utility without the commitment constraint. In §7.3.2, we apply an OCRS to round the LP solution to obtain an adaptive policy, while satisfying the commitment constraint.

7.3.1 Upper Bounding the Optimum Utility

Define the following variables, where i is an index for the Markov systems.

- y_i^u: probability we reach state u in Markov system $\mathcal{S}_i$ for $u \in V_i \setminus T_i$.
- z_i^u: probability we play $\mathcal{S}_i$ when it is in state u for $u \in V_i \setminus T_i$.
- $x_i = \sum_{u \in T_i} z_i^u$: probability $\mathcal{S}_i$ is selected into the final solution when in a destination state.

- $P_{\mathcal{F}}$ denotes a convex relaxation containing all feasible solutions for packing constraint $\mathcal{F}$.

Using these variables we can formulate the following LP, which is inspired from [GM07].

$$\begin{aligned}
\max_{z} \quad & \sum_i \Big(\sum_{u \in T_i} r_i^u z_i^u - \sum_{u \in V_i \setminus T_i} \pi_i^u z_i^u \Big) & \\
\text{subject to} \quad & y_i^{s_i} = 1 & \forall i \in J \\
& y_i^u = \textstyle\sum_{v \in V_i} (P_i)_{uv} z_i^v & \forall i \in J, \forall u \in V_i \setminus s_i \\
& x_i = \textstyle\sum_{u \in T_i} z_i^u & \forall i \in J \\
& z_i^u \le y_i^u & \forall i \in J, \forall u \in V_i \\
& \mathbf{x} \in P_{\mathcal{F}} & \\
& x_i, y_i^u, z_i^u \ge 0 & \forall i \in J, \forall u \in V_i
\end{aligned}$$

The first four constraints characterize the dynamics in advancing the Markov systems. The fifth constraint encodes the packing constraint $\mathcal{F}$. We denote the optimal solution of this LP as $(\mathbf{x}, y, z)$.

Claim 7.3.1. *The utility of the optimal strategy without commitment is at most the LP value.*

Proof. If we interpret the variables y_i^u, x_i, and z_i^u as the probabilities corresponding to the optimal strategy without commitment, it forms a feasible solution to the LP. □

We can efficiently solve the above LP for packing constraints such as matroids, matchings, and intersection of k matroids.

7.3.2 Rounding the LP Using an OCRS

Before describing our rounding algorithm, we define an OCRS (discussed in detail in Chapter 8). Intuitively, it is an online algorithm that given a random set ground elements, selects a feasible subset of them. Moreover, if it can guarantee that every i is selected w.p. at least $\frac{1}{\alpha} \cdot x_i$, it is called $\frac{1}{\alpha}$-selectable.

Definition 7.3.2 (OCRS [FSZ16])**.** *Given a point $x \in P_{\mathcal{F}}$, let $R(x)$ denote a random set containing each i independently w.p. x_i. The elements i reveal one-by-one whether $i \in R(x)$ and we decide whether to select an $i \in R(x)$ into the final solution is taken irrevocably before the next element is revealed. An OCRS is an online algorithm that selects a subset $I \subseteq R(x)$ such that $I \in \mathcal{F}$.*

Definition 7.3.3 ($\frac{1}{\alpha}$-Selectability [FSZ16])**.** *Let $\alpha \ge 1$. An OCRS for $\mathcal{F}$ is $\frac{1}{\alpha}$-selectable if for any $x \in P_{\mathcal{F}}$ and all i, we have $\Pr[i \in I \mid i \in R(x)] \ge \frac{1}{\alpha}$.*

Our algorithm ALG uses OCRS as an oracle. It starts by fixing an arbitrary order π of the Markov systems. (Our algorithm works even when an adversary decides the order of the Markov systems.) Then at each step, the algorithm considers the next element i in π and queries the OCRS whether to select element i if it is ready. If OCRS decides to select i, then ALG advances the Markov system such that it plays from each state u with independent probability

z_i^u/y_i^u. This guarantees that the desination state is reached with probability x_i. If OCRS is not going to select i, then ALG moves on to the next element in π. A formal description of the algorithm can be found in Algorithm 10.

Algorithm 10 Algorithm ALG for Handling the Commitment Constraint

1: Fix an arbitrary order π of the items. Set $M = \emptyset$ and pass $\mathbf{x}$ to OCRS.
2: Consider the next element i in the order of π. Query OCRS whether to add i to M if i is ready.
(a) If OCRS would add i to M, then keep advancing the Markov system as follows: play from each current state $u \in V_i \setminus T_i$ independently with probability z_i^u/y_i^u, and otherwise go to Step 2. If a destination state t is reached then add i to M w.p. z_i^t/y_i^t.
(b) Go to Step 2.

We show below that ALG achieves a utility of at least $1/\alpha$ times the LP optimum.

Lemma 7.3.4. *The utility of* ALG *is at least* $1/\alpha$ *times the LP optimum.*

Since by Claim 7.3.1 the LP optimum is an upper bound on the utility of any policy without commitment, this finishes the proof of Theorem 7.1.3. The only thing left is to prove Lemma 7.3.4.

Proof of Lemma 7.3.4. Recollect that we call a Markov system ready if it reaches an absorbing destination state. We first notice that once ALG starts to advance a Markov system i, then by Step 2 of Algorithm 10, element i is ready with probability exactly x_i. This agrees with what ALG tells the OCRS. Since the OCRS is $1/\alpha$-selectable, the probability that any Markov system $\mathcal{S}_i$ begins advancing is $1/\alpha$. Here the probability is both over the random choice of the OCRS and the randomness due to the Markov systems. Conditioning on the event that $\mathcal{S}_i$ begins advancing, the probability that it is selected into the final solution on reaching a destination state $t \in T_i$ is exactly z_i^t. Hence, the conditioned utility from Markov system $\mathcal{S}_i$ is exactly

$$\sum_{u \in T_i} r_i^u z_i^u - \sum_{u \in V_i \setminus T_i} \pi_i^u z_i^u.$$

By removing the conditioning and by linearity of expectation, the utility of ALG is at least

$$\frac{1}{\alpha} \cdot \sum_i \left(\sum_{u \in T_i} r_i^u z_i^u - \sum_{u \notin T_i} \pi_i^u z_i^u \right),$$

which finishes the proof of this lemma. □

7.4 Sampling Constraints via Robustness

In this section, we study the Robustness model from §7.1.3. Here we are only given approximations of the *input parameters* of the Markov systems, i.e., prices, values, and transition probabilities. Our main result is to show that to design *robust algorithms* for Dag-Utility Maximization

with utility close to that of an optimal policy that exactly knows all the input parameters, it suffices to know the input parameters within an additive error polynomially small in the *input size* (see §7.4.1). This DAG assumption turns out to be necessary as similar results do not hold for general Markov chains (see Appendix 7.4.5). A simple application of Chernoff bounds along with our result implies polynomial sample complexity for DAG-UTILITY MAXIMIZATION.

We formally state our main theorem and the parameters on which it depends in §7.4.1. §7.4.2 shows that close estimates of transition probabilities can be used to obtain close estimates of the grades. In §7.4.3, we use these estimated grades to transform a FRUGAL algorithm into a robust adaptive algorithm for DAG-UTILITY MAXIMIZATION. Similar arguments can be used to obtain the corresponding results for DAG-DISUTILITY MINIMIZATION (we omit this proof).

7.4.1 Main Results and Assumptions

We first explicitly define the *input size* of DAG-UTILITY MAXIMIZATION as follows.

(i) n is the number of Markov systems.

(ii) k is the maximum number of elements in a feasible solution, i.e., $k := \max_{\mathbb{I} \in \mathcal{F}} |\mathbb{I}|$.

(iii) D is the maximum depth of any DAG Markov system.

Denote B an upper bound on all input prices and values, i.e., $\forall i, \forall \pi \in \boldsymbol{\pi}_i, \forall r \in \mathbf{r}_i$, we have $|\pi| \leq B, |r| \leq B$. We make the following assumption.

Assumption 7.4.1. *The upper bound B is polynomial in n, k, and D.*

Such an assumption turns out to be necessary (see Appendix 7.4.5). We now state our main theorem of this section.

Theorem 7.4.2. *Consider a* DAG-UTILITY MAXIMIZATION *problem with a semiadditive objective and satisfying Assumption 7.4.1. Suppose there exists an α-approximation* FRUGAL *algorithm in the Free-Info world. If each input parameter is known to within an additive error of $\epsilon/$ poly, where* poly *is some polynomial in n, k, and D, then there exists an adaptive algorithm $\widehat{\text{ALG}}$ with utility at least*

$$\frac{1}{\alpha} \cdot OPT - \epsilon,$$

where OPTis the utility of the optimal policy that exactly knows the true input parameters.

To simplify the proof of Theorem 7.4.2, we also assume the following without loss of generality (see Appendix 7.4.5 for justifications).

(iv) All non-zero transition probabilities are lower bounded by $1/P$, where P is a polynomial in n, k, and D.

(v) We know the prices $\boldsymbol{\pi}$ and the rewards $\mathbf{r}$ exactly, i.e., the only unknown input parameters are the transition probabilities.

7.4.2 Well-Estimated Input Parameters Imply Well-Estimated Grades

We call the set of Markov systems constructed using our estimated transition probabilities the *estimated world*. The ith Markov system in this *estimated world* is denoted $\widehat{\mathcal{S}}_i = (V_i, \widehat{P}_i, s_i, T_i, \boldsymbol{\pi}_i, \mathbf{r}_i)$, where $\widehat{P}_i$ contains the estimated transition probabilities. Note, $\boldsymbol{\pi}_i$ and $\mathbf{r}_i$ are exact due to Assumption (v). We estimate the grade of a state by simply computing the grade of that state in the estimated world. The following Lemma 7.4.3 bounds the error in estimated grades in terms of the error in transition probabilities.

Lemma 7.4.3. *Consider the* Dag-Utility Maximization *problem satisfying the assumptions in §7.4.1. Suppose all transition probabilities are estimated to within an additive error of $\epsilon < 1/P$, then $\forall i, \forall u \in V_i$, the estimated grade $\widehat{\tau}_i^u$ is within an additive factor of $O(L \cdot \epsilon)$ from the real grade τ_i^u, where $L = D^2BP$.*

Proof. We show below that $\tau_i^u \geq \widehat{\tau}_i^u - L \cdot \epsilon$. A symmetrical argument shows $\widehat{\tau}_i^u \geq \tau_i^u - L \cdot \epsilon$, which finishes the proof of this lemma.

We consider the Markov game $\widehat{G}_u$ defined in §4.2.1 in the estimated world. By definition, there exists an optimal policy Pol that advances the chain at least one more step and achieves an expected utility of 0. Also consider the Markov game G_u in the real world and apply Pol in G_u. Notice Pol might be sub-optimal in G_u and might therefore obtain a negative expected value. Let τ_{fair} be the cost τ in G_u such that Pol obtains an expected value of 0. It follows that $\tau_i^u \geq \tau_{fair}$. It therefore suffices to show that $\tau_{fair} \geq \widehat{\tau}_i^u - L \cdot \epsilon$.

Denote the set of trajectories when applying Pol (in either world) by $\mathcal{S}$ and those in which the item is picked by $\mathcal{S}_{win}$. Denote p_ω the probability of a trajectory $\omega \in \mathcal{S}$ in the real world and $\widehat{p}_\omega$ the probability of it in the estimated world. Let r_ω be the utility of ω (as defined for Utility Maximization by ignoring the cost τ) in either world. It follows that

$$\tau_{fair} \quad = \quad \frac{1}{\sum_{\omega \in \mathcal{S}_{win}} p_\omega} \cdot \sum_{\omega \in \mathcal{S}} \left(p_\omega \cdot r_\omega \right) \quad = \quad \sum_{\omega \in \mathcal{S}} \left(\frac{p_\omega}{\sum_{\omega \in \mathcal{S}_{win}} p_\omega} \cdot r_\omega \right),$$

and that

$$\widehat{\tau}_i^u \quad = \quad \frac{1}{\sum_{\omega \in \mathcal{S}_{win}} \widehat{p}_\omega} \cdot \sum_{\omega \in \mathcal{S}} \left(\widehat{p}_\omega \cdot r_\omega \right) \quad = \quad \sum_{\omega \in \mathcal{S}} \left(\frac{\widehat{p}_\omega}{\sum_{\omega \in \mathcal{S}_{win}} \widehat{p}_\omega} \cdot r_\omega \right).$$

Since each transtion probability is lower bounded by $1/P$, it is estimated to within a multiplicative error of $(1 \pm O(P\epsilon))$. Since p_ω and $\widehat{p}_\omega$ can be written as the product of at most D probabilities, each term $\frac{p_\omega}{\sum_{\omega \in \mathcal{S}_{win}} p_\omega}$ is within a multiplicative error of $(1 \pm O(DP\epsilon))$ from $\frac{\widehat{p}_\omega}{\sum_{\omega \in \mathcal{S}_{win}} \widehat{p}_\omega}$. It follows that τ_{fair} is within a multiplicative factor of $(1 \pm O(DP\epsilon))$ from $\widehat{\tau}_i^u$. But notice that $\widehat{\tau}_i^u \leq DB$, which implies that $\tau_{fair} \geq \widehat{\tau}_i^u - O(D^2BP \cdot \epsilon) = \widehat{\tau}_i^u - O(L \cdot \epsilon)$. □

7.4.3 Designing an Adaptive Strategy for DAG-Utility Maximization

From the previous section we know how to obtain close estimates of the grades. Now we use well-estimated grades to design a robust adaptive strategy for Dag-Utility Maximization

and prove Theorem 7.4.2. Theorem 7.4.2 directly follows by combining Lemma 4.3.3 and the following Lemma 7.4.4.

Lemma 7.4.4. *Assuming the conditions of Theorem 7.4.2 and that the grade of each state is estimated to within an additive factor of $\epsilon/2kD_i$, where D_i is the depth of $\mathcal{S}_i$, then there exists an adaptive algorithm $\widehat{\mathrm{ALG}}$ with utility at least*

$$\frac{1}{\alpha} \cdot \mathbb{E}_{\boldsymbol{\omega}}\left[\max_{\mathbb{I}\in\mathcal{F}}\left\{\mathrm{val}\left(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega})\right)\right\}\right] - \epsilon.$$

To prove Lemma 7.4.4, we describe our algorithm $\widehat{\mathrm{ALG}}_{\mathcal{A}}$ (Algorithm 11). We define $\widehat{\mathbf{Y}}^{\max}$ as follows.

Definition 7.4.5. *Fix a trajectory profile $\boldsymbol{\omega}$ where each Markov system reaches the destination state. For each i and $u \in V_i$, let $d_u(\omega_i)$ be the number of transitions for $\mathcal{S}_i$ to reach u from s_i by taking the trajectory $\omega_i \in \boldsymbol{\omega}$. Let $\widehat{\gamma}_i^u(\omega_i) = \widehat{\tau}_i^u + d_u(\omega_i)\epsilon/2kD_i$. Define $\widehat{Y}_{\omega_i}^{\max} \triangleq \min_{u\in\omega_i}\{\widehat{\gamma}_i^u(\omega_i)\}$. Denote the list of $\widehat{Y}_{\omega_i}^{\max}$'s as $\widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega})$ and $\widehat{\mathbf{Y}}_M^{\max}(\boldsymbol{\omega})$ the list of $\widehat{Y}_{\omega_i}^{\max}$ values in the set M.*

The key idea in $\widehat{\mathrm{ALG}}_{\mathcal{A}}$ (the main difference from Algorithm 7) is the "upward shifting" technique in Step 2. As we advance a Markov system, we shift our estimates of its grades upward. This guarantees that our algorithm is optimal in the teasing game G_T defined for Claim 4.3.6.

Algorithm 11 Algorithm $\widehat{\mathrm{ALG}}_{\mathcal{A}}$ for Utility Maximization in Markovian PoI

1: Start with $M = \emptyset$. Set $v_i = 0$ and $\mathrm{ctr}_i = 0$ for all elements i.
2: For each element $i \notin M$, set $v_i = g\left(\widehat{\mathbf{Y}}_M^{\max}, i, \widehat{\tau}_i^u + \mathrm{ctr}_i \cdot \epsilon/2kD_i\right)$ where u is the current state of i.
3: Consider the element $j = \mathrm{argmax}_{i\notin M \ \&\ M\cup i\in\mathcal{F}}\{v_i\}$ and $v_j > 0$.
4: Proceed $\mathcal{S}_j$ for one step and set $\mathrm{ctr}_j = \mathrm{ctr}_j + 1$. If t_j is reached by $\mathcal{S}_j$, select j into M.
5: If every element $i \notin M$ has $v_i \le 0$ then return set M. Else, go to Step 2.

Proof of Lemma 7.4.4. This lemma immediately follows from the following two claims (whose proofs are in Appendix 7.4.4).

Claim 7.4.6. *The utility of running $\widehat{\mathrm{ALG}}_{\mathcal{A}}$ in the real world is exactly the same as*

$$\mathbb{E}_{\boldsymbol{\omega}}\left[\mathrm{val}\left(Alg(\widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega}), \mathcal{A}), \mathbf{Y}^{\max}(\boldsymbol{\omega})\right)\right].$$

Claim 7.4.7. *For any trajectory profile $\boldsymbol{\omega}$ and for any i, $|\widehat{Y}_{\omega_i}^{\max} - Y_{\omega_i}^{\max}| \le \epsilon/2k$. Thus*

$$\mathrm{val}\left(Alg(\widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega}), \mathcal{A}), \mathbf{Y}^{\max}(\boldsymbol{\omega})\right) \ge \frac{1}{\alpha} \cdot \max_{\mathbb{I}\in\mathcal{F}}\left\{\mathrm{val}\left(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega})\right)\right\} - \epsilon.$$

□

7.4.4 Missing Proofs in the Robustness Model

Proof of Claim 7.4.6. Because $\widehat{\text{ALG}}_{\mathcal{A}}$ shifts the estimated grade upward by $\epsilon/2kD_i$ each time we advance $\mathcal{S}_i$ and that each grade is estimated to within an additive error of $\epsilon/2kD_i$, whenever $\widehat{\text{ALG}}_{\mathcal{A}}$ starts to advance a Markov system, it continues to advance it through the whole epoch. It follows from Claim 4.3.6 that $\widehat{\text{ALG}}_{\mathcal{A}}$ is an optimal policy in the teasing game G_T. By a similar argument as the proof of Claim 4.3.9, one can show that for any list of trajectories $\boldsymbol{\omega}$, running $\widehat{\text{ALG}}_{\mathcal{A}}$ in the real world returns the same solution as running $\mathcal{A}$ on $\widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega})$. These imply the claim. □

Proof of Claim 7.4.7. Since Markov system i can be played at most D_i times, it follows that the estimated grade is shifted upward by at most $(D_i - 1)\epsilon/2kD_i$. It follows that each estimated grade after the upward shifting is still within an additive error of $\epsilon/2k$ from the real grade, which finishes the first part of the grade.

The second part follows from the following inequalities.

$$\begin{aligned}
&\text{val}\Big(Alg(\widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega}), \mathcal{A}), \mathbf{Y}^{\max}(\boldsymbol{\omega})\Big) \\
&\geq \text{val}\Big(Alg(\widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega}), \mathcal{A}), \widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega})\Big) - k \cdot \epsilon/2k \\
&\geq \frac{1}{\alpha} \cdot \max_{\mathbb{I} \in \mathcal{F}} \left\{\text{val}\Big(\mathbb{I}, \widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega})\Big)\right\} - \epsilon/2 \\
&\geq \frac{1}{\alpha} \cdot \text{val}\Big(\text{argmax}_{\mathbb{I} \in \mathcal{F}} \left\{\text{val}(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega}))\right\}, \widehat{\mathbf{Y}}^{\max}(\boldsymbol{\omega})\Big) - \epsilon/2 \\
&\geq \frac{1}{\alpha} \cdot \max_{\mathbb{I} \in \mathcal{F}} \left\{\text{val}\Big(\mathbb{I}, \mathbf{Y}^{\max}(\boldsymbol{\omega})\Big)\right\} - \epsilon,
\end{aligned}$$

where the last line follows because $\alpha \geq 1$. □

7.4.5 Assumptions in the Robustness Model

DAG assumption

We give an example to illustrate why the DAG assumption is necessary for our robustness results to hold. We show that if there are cycles in the Markov chains, one might need to estimate the input parameters to a super-exponentially accurate precision in order to achieve a small additive loss in the performance.

Consider the following UTILITY MAXIMIZATION problem of picking at most one item (i.e. the constraint $\mathcal{F}$ is the uniform Matroid with rank 1) where *all the input parameters are polynomially bounded*. We have n Markov systems $\{\mathcal{S}_i\}_{1 \leq i \leq n}$. The last $n - 2$ Markov systems each has only one state, which is a destination state, with value 0. These Markov systems can be safely ignored since one can pick nothing and obtains 0 utility. We can therefore focus only on the other two Markov systems.

The 2nd Markov system $\mathcal{S}_2$ has only one state, which is a destination state, with value 1. The first Markov system $\mathcal{S}_1$ has three states $\{s_1, v, t_1\}$, where s_1 is the initial state with playing

cost $n^2/2^{2^n}$, t_i is the destination state with value $n^2/2$, and v is some intermediate state with playing cost 0. The transitions in $\mathcal{S}_1$ are as follows. s_1 goes to v deterministically. v goes to s_1 with probability $1 - 1/p2^{2^n}$ and t_1 with probability $1/p2^{2^n}$, where $p \in (0,1]$. Notice that $\mathcal{S}_1$ contains a cycle and a negligible transition out of the cycle to the destination. It follows that the utility obtained by always playing $\mathcal{S}_1$ is $n^2/2 - pn^2$, which is $n^2/4$ if $p = 1/4$ and $-n^2/2$ if $p = 1$.

In this case, if we fail to estimate the transition probabilities of $\mathcal{S}_1$ to a super-exponentially accurate precision of $O(1/2^{2^n})$, it would render it impossible even to distinguish between the case where playing $\mathcal{S}_1$ has utility $\Theta(n^2)$ and the case where playing $\mathcal{S}_1$ has negative utility, which makes it impossible to obtain an approximation policy within a small additive error from the optimal policy.

Polynomial upper bound on input parameters

Here, we give an example to illustrate why Assumption 7.4.1 is necessary for our robustness results to hold. We show that if some parameters are exponential in the input parameter, then one might need to estimate some input parameters to within an additive error that is exponential in the input parameters.

Consider the following Utility Maximization problem of picking at most one item (i.e. the constraint $\mathcal{F}$ is the uniform Matroid with rank 1) where *all the input parameters are polynomially bounded*. We have n Markov systems $\{\mathcal{S}_i\}_{1\le i\le n}$. The last $n-1$ Markov systems deterministically give 0 utility. The first Markov system $\mathcal{S}_1$ has an initial state s_1 and two destination states t_1 and t_2. The initial state s_1 has price 3^n. It goes to t_1 with probability p and t_2 with probability $1-p$. t_1 has reward 2×3^n and t_2 has reward 0.

The player has to decide between playing $\mathcal{S}_1$ or doing nothing at all. If $p = 1/2 + \Theta(1/2^n)$, then the utility of playing $\mathcal{S}_1$ is $\Theta(1.5^n)$ and if $p = 1/2 - \Theta(1/2^n)$, then the utility of playing $\mathcal{S}_1$ is $-\Theta(1.5^n)$. It follows that one need to estimate the transition probabilities to within an additive error that is exponentially small.

Other assumptions without loss of generality

Recall that for the Dag-Utility Maximization problem in the robustness model, we made the following assumptions.

- All non-zero transition probabilities are lower bounded by $1/P$, where P is some polynomial in the parameters above.
- We can estimate the prices $\boldsymbol{\pi}$ and the rewards $\mathbf{r}$ exactly, i.e. the only unknown input parameters are the transition probabilities.

The assumption that all non-zero transition probabilities are polynomially lower bounded is without loss of generality. It can be removed by the following procedure. We start by setting a threshold $1/P$ and estimating all the data to within an additive error smaller than $1/P$. We then ignore the transitions that have estimated probabilities smaller than $2/P$. This is done by reallocating these probability masses to other transitions from the same state in both the orig-

inal Markov systems and the estimated Markov systems. After the removal of these negligible transition probabilities, the remaining Markov systems have a lower bound of $1/P$ on all the transition probabilities. Since the maximum price paid on any sample path in a Markov system is at most DB, it follows that this changes the optimal policy by at most a very small additive factor if the polynomial P we take is large enough. Therefore, we shall assume without loss of generality a lower bound on all non-zero transition probabilities.

The assumption that we can estimate the prices $\boldsymbol{\pi}$ and the rewards $\mathbf{r}$ exactly is again without loss of generality and can be removed by the following argument with a small additive term in the theoretical guarantee. Suppose all the prices $\boldsymbol{\pi}$ and the rewards $\mathbf{r}$ are estimated within an additive error of δ/nD. Since one needs at most D steps to reach the destination for each Markov system, the utility is affected by at most a small additive factor of $\delta/nD \times nD = \delta$ if we set δ to be small. Therefore, we will assume that estimations of the prices $\boldsymbol{\pi}$ and the rewards $\mathbf{r}$ are exact and only the estimations of transition probabilities have deviations from the real

Part III

Stopping-Time Algorithms

Chapter 8

The Prophet Inequality via Online Contention Resolution Schemes

8.1 Introduction

In the *prophet inequality* problem a decision maker must choose one of n buyers arriving one-by-one to purchase their *single* item. Each buyer has a value drawn independently from a *known distribution*, but the buyer arrival order is chosen by an adversary. As described in §1.3.2, we can use this to model the Diamond-Selling scenario.

Definition 8.1.1 (Single item prophet inequality)**.** *For $i \in [n]$, given probability distributions $\mathcal{D}_i$ of n independent random variables, suppose their outcome values $v_i \sim \mathcal{D}_i$ are revealed in an adversarial order. Whenever a value is revealed we have to* immediately *and* irrevocably *decide if we want to select this element, while ensuring that we never select more than one element. The goal is to maximize the expected value of the element that we select.*

The benchmark used in the study of prophet inequalities is the expected offline optimum, which is the expected value of an algorithm that knows all v_i from the beginning (and therefore gets $\max_i\{v_i\}$). We define the *competitive ratio* of an algorithm to be the ratio of the expected value of the algorithm to the expected offline optimum. The above single item prophet inequality problem was resolved by Krengel and Sucheston, who gave a tight 1/2-competitive algorithm [KS78, KS77]. How to design a prophet inequality where we sell multiple items?

Besides being a natural problem in the field of Stopping Theory, the motivation to design multiple item prophet inequalities comes from its applications in the field of Mechanism Design. Often while designing a mechanism, one has to balance between maximizing revenue/welfare and the simplicity of the mechanism. While there exist optimal mechanisms such as VCG or Myerson's mechanism, they are impractical in real markets [AM06, Rot07]. On the other hand, *Sequentially Posted Pricing mechanisms*, where we offer take-it-or-leave-it prices to buyers, are known to be both simple and to give good approximations to optimal mechanisms. This reduces the mechanism design problem to designing multiple-choice variants of the prophet inequality problem, where the decision maker selects a subset of buyers feasible in a given packing constraint $\mathcal{F} \subseteq 2^{[n]}$ with the goal of maximizing the sum of buyer values. Motivated by

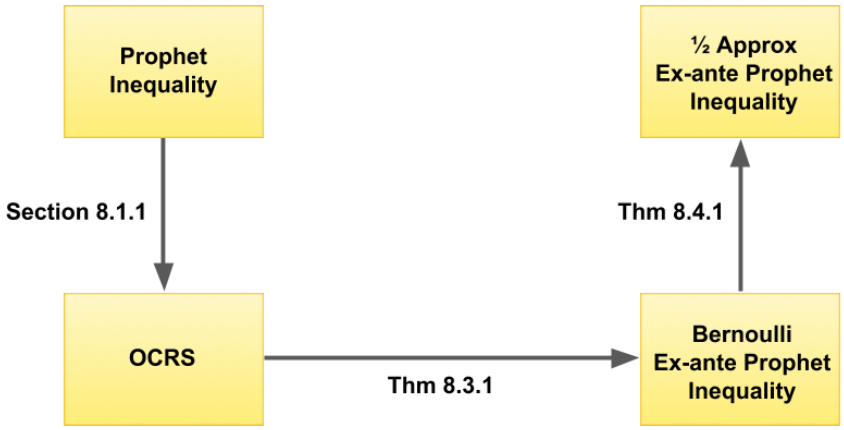

Figure 8.1: In §8.1.1 we show that it suffices to design an OCRS to prove a matroid prophet inequality (Theorem 8.1.2). In Theorem 8.3.1, we show that to design an OCRS it suffices to design an ex-ante prophet inequality for Bernoulli variables. Finally, in Theorem 8.4.1 we design a 1/2-approximation ex-ante prophet inequality, which implies Theorem 8.1.2.

these applications, tight prophet inequalities have been obtained subject to cardinality constraint [HKS07, Ala14], and its generalization to arbitrary matroid constraints [KW12].

8.1.1 Results

In this chapter we present a new proof of the matroid prophet inequality using a generic technique known as *online contention resolution scheme* (OCRS). See Figure 8.1.

Theorem 8.1.2 ([KW12]). *There exists a* 1/2*-matroid prophet inequality.*

To prove Theorem 8.1.2, we first bound the expected offline optimum using a convex relaxation. Given a prophet inequality problem instance with packing constraints $\mathcal{F}$ and random variables $v_i \sim \mathcal{D}_i$ for $i \in [n]$, we define the following *ex-ante relaxation*

$$\max_{\mathbf{x}} \sum_i x_i \cdot u_i(x_i) \quad \text{s.t.} \quad \mathbf{x} \in P_{\mathcal{F}}, \tag{8.1}$$

where $u_i(x_i)$ denotes $\mathbb{E}_{v_i \sim \mathcal{D}_i}[v_i \mid v_i \text{ takes value in its top } x_i \text{ quantile}]$. To prove (8.1) is an upper bound, we interpret x_i as the probability that i is in the offline optimum. The bound follows because any such i can contribute at most $x_i \cdot u_i(x_i)$ to the expected offline optimum. It is known that (8.1) is a convex program and can be solved efficiently; see [FSZ16] for more details.

Given the upper bound in (8.1), consider an algorithm that ignores constraints $\mathcal{F}$ and selects every element i when it takes a value in its top x_i quantile. Clearly, by linearity of expectation, the expected value of such an algorithm is $\sum_i x_i \cdot u_i(x_i)$. This raises the following question:

> Does there exist an online algorithm that selects each element i at least 1/2-fraction of the times when i takes value in its top x_i quantile, while ensuring that the selected

set of elements is feasible in $\mathcal{F}$?

Existence of a 1/2-OCRS, formally defined in §8.2, answers the above question affirmatively. The main result of this chapter, based on a joint work with Euiwoong Lee [LS18], is the following:

Theorem 8.1.3. *For matroids, there exists a* 1/2*-OCRS for adversarial arrival order.*

By the above discussion, Theorem 8.1.3 along with linearity of expectation implies a prophet inequality algorithm with expected value at least $\frac{1}{2}\sum_i x_i \cdot u_i(x_i)$. Since $\sum_i x_i \cdot u_i(x_i)$ is an upper bound on the expected offline optimum, this proves Theorem 8.1.2.

The main idea in the proof of Theorem 8.1.3 is to show a deep connection between OCRSs and prophet inequalities. In particular, we show that an *ex-ante prophet inequality* for Bernoulli variables can be used to design an OCRS.

Definition 8.1.4 (Ex-ante prophet inequality)**.** *For $\mathbf{x} \in P_{\mathcal{F}}$, consider a prophet inequality instance where r.v. v_i takes value $u_i \in \mathbb{R}_{\geq 0}$ w.p. x_i and is 0 otherwise. A c-approximation ex-ante prophet inequality for $0 \leq c \leq 1$ and packing constraints $\mathcal{F}$ is a prophet inequality algorithm with expected value at least $(c \cdot \sum_i x_i u_i)$.*

In §8.3 we show why an ex-ante prophet inequality implies an OCRS (Theorem 8.3.1), and in §8.4 we design an ex-ante prophet inequality for matroids (Theorem 8.4.1).

8.1.2 Related Work

The connection between multiple-choice prophet inequalities and mechanism design was recognized in the work of Hajiaghayi et al. [HKS07]. In particular, they proved a prophet inequality for uniform matroids; their bound was later improved by Alaei [Ala11]. Chawla et al. [CHMS10] further developed the connection between prophet inequalities and mechanism design, and proved that for general matroids one can be $O(1)$-competitive in a variant of the prophet inequality where the algorithm may choose the order in which the items are viewed. Yan [Yan11] improved this result to $e/(e-1)$-competitive. The *matroid prophet inequality* was first explicitly formulated by Kleinberg and Weinberg [KW12].

In a different direction, Alaei et al. [AHL12] considered a variant they call *prophet-inequality matching*, which is useful for online ad allocation. More generally, for intersection of a constant number of matroid, knapsack, and matching constraints, Feldman, Svensson, and Zenklusen [FSZ16] gave an $O(1)$-competitive algorithm; this is a corollary of their *online contention resolution schemes* that we also use heavily in our work. Feldman et al. [FGL15] consider combinatorial auctions in the prophet inequality settings. Azar, Kleinberg, and Weinberg [AKW14] study a limited information variant where the algorithm only has access to samples from each day's distributions. Esfandiari et al. [EHLM17] considered a mixed notion of "Prophet Secretary" where the items arrive in a uniformly random order and draw their values from known independent distributions. Finally, for general downward-closed constraint, Rubinstein [Rub16] gave an $O(\log n \log r)$-competitive prophet inequality.

8.2 Online Contention Resolution Schemes

Given a combinatorial optimization problem, a common algorithmic approach is to first solve a convex relaxation of the problem and to then round the obtained fractional solution $\mathbf{x}$ into a *feasible integral* solution while (approximately) preserving the objective. Contention resolution schemes (CRSs), introduced in [CVZ14], is a way to perform this rounding given a fractional solution $\mathbf{x} \in \mathbb{R}^n_{\geq 0}$. For $c > 0$, intuitively a c-CRS is a rounding algorithm that guarantees every element i is selected into the final feasible solution w.p. at least $c \cdot x_i$. To formally define a CRS, we need some notation: For a given $\mathbf{x} \in [0,1]^V$, let $R(\mathbf{x})$ denote a random set containing each element $i \in V$ independently w.p. x_i, and say an element i is *active* if it belongs to $R(\mathbf{x})$.

Definition 8.2.1 (Contention resolution scheme)**.** *Given a finite ground set V with $n = |V|$ and a packing (downward-closed) family of feasible subsets $\mathcal{F} \subseteq 2^V$, let $P_{\mathcal{F}} \subseteq [0,1]^V$ be the convex hull of all characteristic vectors of feasible sets. For a given $\mathbf{x} \in P_{\mathcal{F}}$, a* c-selectable *CRS (or simply, c-CRS) is a (randomized) mapping $\pi : 2^V \to V$ satisfying the following three properties:*

(i) $\pi(S) \subseteq S$ *for all* $S \subseteq V$.

(ii) $\pi(S) \subseteq \mathcal{F}$ *for all* $S \subseteq V$.

(iii) $\Pr_{R(\boldsymbol{x}),\pi}[i \in \pi(R(\boldsymbol{x}))] \geq c \cdot x_i$ *for all* $i \in V$.

Notice, if f is a monotone linear function then $\mathbb{E}[f(\pi(R(\mathbf{x})))] \geq c \cdot \mathbb{E}[f(R(\mathbf{x}))]$ by linearity of expectation. This inequality also holds for submodular functions when the CRS additionally satisfies a "greedy" property (see §9.3 for more details). By constructing CRSs for various families $\mathcal{F}$, Chekuri et al. [CVZ14] give improved approximation algorithms for linear and submodular maximization problems under knapsack, matroid, matchoid, and their intersections constraints.

In the above applications of CRSs to offline optimization problems, the algorithm first flips all the random coins to sample $R(\mathbf{x})$, and then obtains $\pi(R(\mathbf{x})) \subseteq R(\mathbf{x})$. For various online problems such as the prophet inequality, this randomness is an inherent part of the problem. Feldman et al. [FSZ16] therefore introduce an OCRS where whether $i \in R(\mathbf{x})$ (or not) is only revealed one-by-one to the CRS algorithm.

Definition 8.2.2 (Online contention resolution scheme)**.** *A* c-selectable *OCRS (or simply, c-OCRS) is a c-selectable contention resolution scheme π that is only revealed one-by-one whether $i \in R(\boldsymbol{x})$ (or not). After each revelation (online arrival), the OCRS algorithm π has to immediately and irrevocably decide whether to include $i \in R(\boldsymbol{x})$ into $\pi(R(\boldsymbol{x}))$ (if possible) without knowing the future revelations, and while always satisfying properties (i)-(iii) of a c-CRS.*

In this chapter, we assume that the adversarial arrival order is known to the algorithm in advance. This offline adversary is weaker than the almighty adversary considered in [FSZ16], but is common in the prophet inequality literature. To build some intuition for the above definitions we first discuss the special case of a rank 1 matroid, i.e., we can only select one element.

8.2.1 Example: Rank 1 matroid

For simplicity, in this section we assume that all random variables are Bernoulli, i.e., v_i takes value y_i independently w.p. p_i, and is 0 otherwise. We first show why a c-OCRS implies a

c-approximation prophet inequality for rank 1 matroids.

Consider the optimum solution $\mathbf{x}$ to the ex-ante relaxation (8.1) for the above Bernoulli instance. Its objective value is $\sum_i x_i y_i$ where $\mathbf{x}$ satisfies $\sum_i x_i \leq 1$. Moreover, $x_i \leq p_i$ for all i because selecting i beyond p_i does not increase (8.1). To see why (8.1) gives an upper bound on the expected offline maximum, observe that if we interpret x_i as the probability that v_i is the offline maximum, this gives a feasible solution to $\sum_i x_i \leq 1$ and with value at most $\sum_i x_i y_i$. Thus, to prove a c-approximation prophet inequality, it suffices to design an online algorithm with value at least $c \cdot \sum_i x_i y_i$. Consider an algorithm that runs a c-OCRS on $\mathbf{x}$, where i is considered active independently w.p. x_i/p_i whenever v_i takes value y_i. This ensures element i is active w.p. exactly x_i. Since a c-OCRS guarantees each element is selected w.p. $\geq c$ when it is active, by linearity of expectation such an algorithm has expected value at least $c \cdot \sum_i x_i y_i$.

We now discuss a simple 1/4-OCRS for a rank 1 matroid. Given $\mathbf{x}$ satisfying $\sum_i x_i \leq 1$, consider an algorithm that ignores each element i independently w.p. 1/2, and otherwise selects i only if it is active. Since this algorithm selects any element i w.p. at most $x_i/2$ (when i is not ignored and is active), by Markov's inequality the algorithm selects no element till the end w.p. at least $1 - \sum_i x_i/2 \geq 1/2$. Hence the algorithm reaches each element i w.p. at least 1/2 without selecting any of the previous elements. Moreover, it does not ignore i w.p. 1/2, which implies it considers each element w.p. at least 1/4. The OCRS due to Feldman et al. [FSZ16] can be thought of generalizing this approach to a general matroid.

An interesting result of Alaei [Ala14] shows that the above 1/4-OCRS can be improved to a 1/2-OCRS over a rank 1 matroid by "greedily" maximizing the probability of ignoring the next element i, but considering i w.p. 1/2 on average. This raises the question whether one can obtain a 1/2-OCRS for general matroids.

8.2.2 Our techniques

We first see the difficulty in extending Alaei's greedy approach from a rank 1 matroid to a general matroid. Consider the graphic matroid for the *Hat* example (see Figure 8.2). Suppose the base edge (u_1, u_2) appears in the end of an adversarial order. Notice that any algorithm which ignores the structure of the matroid is very likely to select some pair of edges (u_1, v_i) and (v_i, u_2) for some i. Since this pair spans the base edge (u_1, u_2), such an OCRS algorithm will not satisfy c-selectability for (u_1, u_2). To overcome this, Feldman et al. [FSZ16] decompose the matroid into "simpler" matroids using $\mathbf{x}$. However, it is not clear how to extend their approach beyond a 1/4-OCRS.

In this chapter we take an alternate LP based approach to design OCRSs, which was first used by Chekuri et al. [CVZ14] to design offline CRSs. The idea is to consider an exponential sized linear program where each variable denotes a deterministic OCRS algorithm. The objective is to maximize c s.t. each element is selected at least c fraction of the times (c-selectability). To argue that the optimal $c \geq 1/2$, in §8.3 we prove that the value of the dual LP is at least 1/2 because it can be interpreted as an ex-ante prophet inequality.

Next, to show there exists a 1/2 approximation ex-ante prophet inequality, our approach is inspired from the matroid prophet inequality of Kleinberg and Weinberg [KW12]. They give

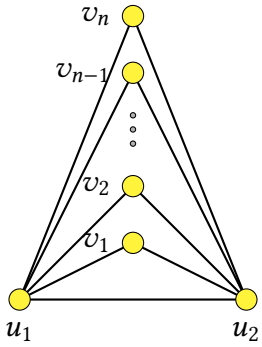

Figure 8.2: The Hat example on $n+2$ vertices. The following $\mathbf{x}$ belongs to the graphic matroid: $x_e = 1/2$ for $e = (u_i, v_j)$ where $i \in \{1,2\}$ and $j \in \{1,\ldots,n\}$, and $x_e = 1$ for $e = (u_1, u_2)$.

an online algorithm that gets at least half of the expected offline optimum for the product distribution (independent r.v.s). Unfortunately, their techniques do not directly extend because the ex-ante relaxation objective could be significantly higher than for the product distribution. (This is known as the *correlation gap* and can be $e/(e-1)$; see Chapter 9 for further discussion.) Our primary technique is to view the ex-ante relaxation solution as a "special kind" of a correlated value distribution. Although prophet inequalities are not possible for general correlated distributions [HK92], we show that in this special case the original proof of the matroid prophet inequality algorithm retains its $1/2$ approximation after some modifications.

8.3 Designing OCRS Assuming an Ex-Ante Prophet Inequality

In this section we prove the following approximation factor preserving reduction from OCRSs to ex-ante prophet inequalities.

Theorem 8.3.1. *If there exists a c-approximation ex-ante prophet inequality over a matroid $\mathcal{M}$ then there exists a c-OCRS over $\mathcal{M}$.*

Proof. Let Φ^* denote the set of all *deterministic* online rounding algorithms. For $\phi \in \Phi^*$ and element $i \in V$, let $q_{i,\phi}$ denote the probability that ϕ selects i in the adversarial order, where the randomness is over $R(\mathbf{x})$. Similar to Chekuri et al. [CVZ14, Section 4.2], we use the solution to the following linear program to design a randomized OCRS: randomly execute a ϕ w.p. λ_ϕ.

$$\begin{aligned}
&\max c \\
&\text{s.t.} \sum_{\phi \in \Phi^*} q_{i,\phi} \cdot \lambda_\phi \geq x_i \cdot c && i \in N \\
&\sum_{\phi \in \Phi^*} \lambda_\phi = 1 \\
&\lambda_\phi \geq 0 && \forall \phi \in \Phi^*
\end{aligned}$$

The dual is

$$\begin{aligned}
&\min \mu \\
&\text{s.t.} \sum_{i \in N} q_{i,\phi} \cdot y_i \le \mu && \phi \in \Phi^* \\
&\sum_{i \in N} x_i \cdot y_i = 1 \\
&y_i \ge 0 && \forall i \in N
\end{aligned}$$

The only difference from [CVZ14] is that Φ^* is the family of online set functions. Although this linear program has an exponential number of variables/constraints, we show there exists a polynomial time algorithm to obtain a solution with primal value at least $c - \epsilon$, where $\epsilon > 0$ is an arbitrarily small constant.

We first prove that the above pair of linear programs have value at least c. Given $\mathbf{y}$ such that $\sum_i x_i y_i = 1$, notice that a c-approximation ex-ante prophet inequality gives in polynomial time an online algorithm $\phi \in \Phi^*$ with expected value at least $c \cdot \sum_i x_i y_i$, i.e., $\sum_i q_{i,\phi} \cdot y_i \ge c$. This implies the optimal value of the LPs is at least c. In particular, for any $\epsilon > 0$, the polytope

$$Q_{c-\epsilon} := \left\{ y : y \ge 0, \sum_i x_i y_i = 1, \sum_i q_{i,\phi} y_i \le c - \epsilon \text{ for all } \phi \in \Phi^* \right\}$$

is empty. Since we have an efficient *separation oracle (for any y, we can find a violated constraint in polynomial time)* for $Q_{c-\epsilon}$, by running the ellipsoid algorithm, we can find a subset $\Phi' \subseteq \Phi^*$ with $|\Phi'| = \text{poly}(n)$ in polynomial time such that $Q'_{c-\epsilon} := \{y : y \ge 0, \sum_i x_i y_i = 1, \sum_i q_{i,\phi} y_i \le c - \epsilon \text{ for all } \phi \in \Phi'\}$ is empty. Then the following primal program with polynomial number of variables and constraints has the optimal value at least $c - \epsilon$.

$$\begin{aligned}
&\max c \\
&\text{s.t.} \sum_{\phi \in \Phi'} q_{i,\phi} \lambda_\phi \ge x_i c && i \in N \\
&\sum_{\phi \in \Phi'} \lambda_\phi = 1 \\
&\lambda_\phi \ge 0 && \forall \phi \in \Phi'.
\end{aligned}$$

The optimal solution to this linear program has value at least $c - \epsilon$ and gives a randomized $(c - \epsilon)$-OCRS for any $\epsilon > 0$. □

8.4 An Ex-Ante Prophet Inequality for Matroids

In this section we design an ex-ante prophet inequality for matroids.

Theorem 8.4.1. *For matroids, there exists a $1/2$-approximation ex-ante prophet inequality.*

The proof of this theorem is similar to [KW12]. Together with Theorem 8.3.1, this gives Theorem 8.1.3 as a corollary.

8.4.1 Notation

Given $\mathbf{x} \in P_{\mathcal{M}}$, let $\mathbf{v} \sim \mathcal{D}$ be a set of random element values $\{v_1, \ldots, v_n\}$ where each v_i independently takes value u_i w.p. x_i and is 0 otherwise. Since $\mathbf{x} \in P_{\mathcal{M}}$, we can write it as a convex combination of independent sets in the matroid. In particular, this gives a correlated distribution $\hat{\mathcal{D}}$ over independent sets of $\mathcal{M}$ such that for each $i \in V$, we have $\Pr_{I \sim \hat{\mathcal{D}}}[i \in I] = x_i$. Let $\hat{\mathbf{v}} = \{\hat{v}_1, \ldots, \hat{v}_n\}$ be a set of random values obtained by sampling $I \sim \hat{\mathcal{D}}$ and setting $\hat{v}_i = u_i$ for $i \in I$, and $\hat{v}_i = 0$ otherwise. Notice the optimal value of (8.1) is $\sum_i x_i u_i$ and for each $i \in V$, we have $\mathbb{E}[\hat{v}_i] = x_i u_i$.

We need the following notation to describe our algorithms.

Definition 8.4.2. *For a given vector $\hat{\mathbf{v}}$ of values of n elements and any $A \subseteq V$, we define:*

- *Let $Opt(\hat{\mathbf{v}} \mid A) \subseteq V \setminus A$ denote the maximum value independent set in the contracted matroid $\mathcal{M}/A$.*
- *Let $R(A, \hat{\mathbf{v}}) := \sum_{i \in Opt(\hat{\mathbf{v}}|A)} \hat{v}_i$ denote the* remaining value *after selecting set A.*

We next define a *base price* of for every element i.

Definition 8.4.3. *For $A \in \mathcal{I}$ denoting an independent set of elements accepted by our algorithm, we define*

- *Let $b_i(A, \hat{\mathbf{v}}) := R(A, \hat{\mathbf{v}}) - R(A \cup \{i\}, \hat{\mathbf{v}})$ denote a threshold for element i.*
- *Let $b_i(A) := \mathbb{E}_{\hat{\mathbf{v}} \sim \hat{\mathcal{D}}}[b_i(A, \hat{\mathbf{v}})]$ denote the* base price *for element i.*

8.4.2 Adversarial Order

Consider $\mathbf{v} \sim \mathcal{D}$ as the input to our online algorithm, where v_i takes value u_i w.p. x_i and is 0 otherwise. Given $\mathbf{v}$, our algorithm is deterministic and let $A := A(\mathbf{v})$ denote the set of elements that it selects. Relabel the elements such that the arrival order of the elements is $1, \ldots, n$. Let $A_i = A \cap \{1, \ldots, i\}$.

Our algorithm selects the next element i iff both $v_i > T_i := \alpha \cdot b_i(A_{i-1})$ and selecting i is feasible in $\mathcal{M}$, where $\alpha = \frac{1}{2}$. Thus, the total value of algorithm $\mathsf{Alg} := \sum_{i \in A} v_i = \mathsf{Revenue} + \mathsf{Utility}$, where

$$\mathsf{Revenue} := \sum_{i \in A} T_i \qquad \text{and} \qquad \mathsf{Utility} := \sum_{i \in A} (v_i - T_i)^+.$$

To prove Theorem 8.4.1 it suffices to show $\mathbb{E}[\mathsf{Alg}] = \mathbb{E}[\mathsf{Revenue}] + \mathbb{E}[\mathsf{Utility}] \geq \alpha \cdot \sum_{i \in V} x_i u_i$.

We keep track of the algorithm's progress using the following *residual* function:

$$r(i) := \mathbb{E}_{\mathbf{v} \sim \mathcal{D}, \hat{\mathbf{v}} \sim \hat{\mathcal{D}}}[R(A_{i-1}, \hat{\mathbf{v}})].$$

Clearly, $r(0) = \sum_{i \in V} x_i u_i$. In the following Lemma 8.4.4 and Lemma 8.4.5, we use the residual function to lower bound $\mathbb{E}[\mathsf{Revenue}]$ and $\mathbb{E}[\mathsf{Utility}]$.

Lemma 8.4.4.

$$\mathbb{E}_{\mathbf{v} \sim \mathcal{D}}[\mathsf{Revenue}] = \alpha \cdot \Big(r(0) - r(n) \Big).$$

Proof. From the definition of Revenue, we get

$$\begin{aligned}\text{Revenue} &= \alpha \cdot \sum_{i \in A} b_i(A_{i-1}) \\ &= \alpha \cdot \sum_{i \in A} \Big(\mathbb{E}_{\hat{\mathbf{v}}}[R(A_{i-1}, \hat{\mathbf{v}})] - \mathbb{E}_{\hat{\mathbf{v}}}[R(A_{i-1} \cup \{i\}, \hat{\mathbf{v}})] \Big) \\ &= \alpha \cdot \sum_{i \in A} \Big(\mathbb{E}_{\hat{\mathbf{v}}}[R(A_{i-1}, \hat{\mathbf{v}})] - \mathbb{E}_{\hat{\mathbf{v}}}[R(A_{i}, \hat{\mathbf{v}})] \Big) \\ &= \alpha \cdot \Big(\mathbb{E}_{\hat{\mathbf{v}}}[R(A_0, \hat{\mathbf{v}})] - \mathbb{E}_{\hat{\mathbf{v}}}[R(A, \hat{\mathbf{v}})] \Big).\end{aligned}$$

Taking expectation over $\mathbf{v} \sim \mathcal{D}$ and using the definitions of $r(0)$ and $r(n)$, the lemma follows. □

Lemma 8.4.5.

$$\mathbb{E}_{\mathbf{v} \sim \mathcal{D}}[\text{Utility}] \geq (1 - \alpha) \cdot r(n).$$

Proof. We prove the following two inequalities:

$$\mathbb{E}_{\mathbf{v} \sim \mathcal{D}}[\text{Utility}] \quad \geq \quad \mathbb{E}_{\mathbf{v} \sim \mathcal{D}, \hat{\mathbf{v}} \sim \hat{\mathcal{D}}} \Big[\sum_{i \in Opt(\hat{\mathbf{v}}|A)} (\hat{v}_i - T_i)^+ \Big] \tag{8.2}$$

and

$$\mathbb{E}_{\mathbf{v} \sim \mathcal{D}, \hat{\mathbf{v}} \sim \hat{\mathcal{D}}} \Big[\sum_{i \in Opt(\hat{\mathbf{v}}|A)} (\hat{v}_i - T_i)^+ \Big] \quad \geq \quad (1 - \alpha) \cdot \mathbb{E}_{\mathbf{v} \sim \mathcal{D}, \hat{\mathbf{v}} \sim \hat{\mathcal{D}}}[R(A, \hat{\mathbf{v}})]. \tag{8.3}$$

Lemma 8.4.5 now follows by summing (8.2) and (8.3), and using $r(n) = \mathbb{E}_{\mathbf{v} \sim \mathcal{D}, \hat{\mathbf{v}} \sim \hat{\mathcal{D}}}[R(A, \hat{\mathbf{v}})]$.

To prove (8.2), notice that for any element i not selected by the algorithm $v_i \leq T_i$. This implies

$$\mathbb{E}_{\mathbf{v} \sim \mathcal{D}}[\text{Utility}] = \mathbb{E}_{\mathbf{v}} \Big[\sum_{i \in A} (v_i - T_i)^+ \Big] = \mathbb{E}_{\mathbf{v}} \Big[\sum_{i \in V} (v_i - T_i)^+ \Big].$$

Now observe that for any fixed i and $v_1, \ldots, v_{i-1}$, the threshold T_i is determined. Since v_i and $\hat{v}_i$ are independent random variables with the same distribution, we get

$$\mathbb{E}_{\mathbf{v}}[(v_i - T_i)^+ | v_1, \ldots, v_{i-1}] = \mathbb{E}_{\mathbf{v}, \hat{\mathbf{v}}}[(\hat{v}_i - T_i)^+ | v_1, \ldots, v_{i-1}].$$

This implies

$$\begin{aligned}\mathbb{E}_{\mathbf{v} \sim \mathcal{D}}[\text{Utility}] &= \mathbb{E}_{\mathbf{v}} \Big[\sum_{i \in V} (v_i - T_i)^+ \Big] \\ &= \mathbb{E}_{\mathbf{v}, \hat{\mathbf{v}}} \Big[\sum_{i \in V} (\hat{v}_i - T_i)^+ \Big] \quad \geq \quad \mathbb{E}_{\mathbf{v}, \hat{\mathbf{v}}} \Big[\sum_{i \in Opt(\hat{\mathbf{v}}|A)} (\hat{v}_i - T_i)^+ \Big].\end{aligned}$$

Finally, to prove (8.3), we have

$$\mathbb{E}_{\mathbf{v}, \hat{\mathbf{v}}}[R(A, \hat{\mathbf{v}})] = \mathbb{E}_{\mathbf{v}, \hat{\mathbf{v}}} \Big[\sum_{i \in Opt(\hat{\mathbf{v}}|A)} \hat{v}_i \Big] \leq \mathbb{E}_{\mathbf{v}, \hat{\mathbf{v}}} \Big[\sum_{i \in Opt(\hat{\mathbf{v}}|A)} T_i \Big] + \mathbb{E}_{\mathbf{v}, \hat{\mathbf{v}}} \Big[\sum_{i \in Opt(\hat{\mathbf{v}}|A)} (\hat{v}_i - T_i)^+ \Big]$$

$$\leq \alpha \cdot \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}}}\Big[\sum_{i\in Opt(\hat{\mathbf{v}}|A)} \hat{v}_i\Big] + \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}}}\Big[\sum_{i\in Opt(\hat{\mathbf{v}}|A)} (\hat{v}_i - T_i)^+\Big],$$

where the first inequality uses $\hat{v}_i \leq T_i + (\hat{v}_i - T_i)^+$ and the second inequality uses Claim 8.4.6 for $S = Opt(\hat{\mathbf{v}} \mid A)$. After rearranging, this implies (8.3). □

We need the following Claim 8.4.6 in the proof of Lemma 8.4.5.

Claim 8.4.6. *For every pair of disjoint sets A, S such that $A \cup S \in \mathcal{M}$,*

$$\alpha \cdot \mathbb{E}_{\hat{\boldsymbol{v}}\sim\hat{\mathcal{D}}}\Big[\sum_{i\in S} R(A_{i-1}, \hat{\boldsymbol{v}}) - R(A_{i-1}\cup\{i\}, \hat{\boldsymbol{v}})\Big] = \sum_{i\in S} T_i \leq \alpha \cdot \mathbb{E}_{\hat{\boldsymbol{v}}\sim\hat{\mathcal{D}}}[R(A, \hat{\boldsymbol{v}})]. \tag{8.4}$$

Proof. This directly follows from [KW12], as they proved it for every fixed w'. The proof is similar to Claim 11.4.6 in Chapter 11. □

Proof of Theorem 8.4.1. Using Lemma 8.4.4 and Lemma 8.4.5, and substituting $\alpha = \frac{1}{2}$, we get

$$\mathbb{E}[\mathsf{Alg}] = \mathbb{E}[\mathsf{Utility}] + \mathbb{E}[\mathsf{Revenue}] \geq \frac{1}{2} r(0) = \frac{1}{2}\sum_{i\in V} x_i u_i.$$ □

Chapter 9

Combinatorial Prophet Inequalities

9.1 Introduction

Recall that in the classic prophet inequality, a gambler is asked to choose one of n independent non-negative random payoffs. In *multiple choice prophet inequalities* studied in Chapter 8, the gambler chooses multiple payoffs, subject to some known-in-advance feasibility constraint, and receives their sum, i.e., an additive objective. In this chapter, we are interested in the case where the gambler's value is not additive over the outcomes of the random draws: For example, instead of monetary payoffs, at each time period the gambler can choose to receive a random item, say a car, and the value from owning multiple cars diminishes quickly. As another example, the gambler receives monetary payoffs, but his marginal value for the one-millionth dollar is much smaller than for the first. Can we design prophet inequalities for more general combinatorial objective functions?

9.1.1 Model and Results

Our main conceptual contribution is a generalization of the Prophet Inequality setting to combinatorial valuations. Roughly, on each of n days, the decision maker knows an independent prior distribution over k potential items that could appear.[1] She also has access to a combinatorial (in particular, submodular or monotone subadditive) function f that describes the value of any subset of the $n \cdot k$ items. The goal is to maximize the value of f on the union of all selected items. Formally, we define the problem using the discussion of combinatorial but independent functions from Chapter 6.

Definition 9.1.1 (Combinatorial Prophet Inequality)**.** *The offline inputs to the problem are:*

- *n sets $U_1, \ldots, U_n$; we denote their union $U \triangleq \bigcup_{i=1}^{n} U_i$;*
- *a combinatorial function $f : \{0,1\}^U \to \mathbb{R}_{\geq 0}$;*
- *n distributions $\mathcal{D}_i$ over subset U_i; and*

[1] Note that some notion of independence assumption is necessary as even for the single choice problem, if values are arbitrarily correlated then every online algorithm is $\Omega(n)$ competitive [HK92].

- *a feasibility constraint $\mathcal{F}$ over $[n]$*

On the i-th time period, the algorithm observes an element $X_i \in U_i$ drawn according to $\mathcal{D}_i$, independently from outcomes and actions on past and future days. The algorithm must decide (immediately and irrevocably) whether to add i and X_i to sets W and X_W, respectively, subject to W remaining feasible in $\mathcal{F}$. The objective is to maximize $f(X_W)$.

We obtain the following combinatorial prophet inequalities in [RS17].

Theorem 9.1.2 (Submodular Prophet)**.** *There exists an efficient randomized $O(1)$-competitive algorithm for monotone/non-monotone submodular prophet over any matroid.*

Since for general packing constraints one cannot obtain efficient algorithms using membership queries even for additive valuations [Rub16], we design a computationally inefficient subadditive prophet inequality over packing constraints.

Theorem 9.1.3 (Monotone Subadditive Prophet)**.** *There exists an $O(\log n \cdot \log^2 r)$-competitive algorithm for monotone subadditive prophet inequality subject to any packing constraints.*

9.1.2 Techniques

Our main technique in the proof of Theorem 9.1.2 is a constant *correlation gap* for non-monotone submodular functions, which might be of independent interest. For a monotone submodular function f, [CCPV07] showed that the expected value of f over any distribution of subsets is at most a constant factor larger than the expectation over subsets drawn from the product distribution with the same marginals. This bound on the correlation gap has been very useful in the past decade with applications in optimization [CVZ14], mechanism design [ADSY12, Yan11, BCK12, BH16], social networks [RSS15, BPR+16], and recommendation systems [KSS13].

It turns out that when f is non-monotone, the correlation gap is unbounded, even for $n = 2$. Instead, we prove a correlation gap for a related function:

$$f_{\max}(S) \triangleq \max_{T \subseteq S} f(T).$$

(Note that $f_{\max}$ is monotone, but may not be submodular.)

Theorem 9.1.4 (Non-monotone correlation gap; informal)**.** *For any (non-monotone) submodular function f, the function $f_{\max}$ has a correlation gap of $O(1)$.*

Our techniques in the proof of Theorem 9.4.3 extend the ideas from [Rub16]. We show how to approximate a monotone subadditive function using an XOS function and interpreting it as another packing constraint can be used to obtain subadditive prophet inequalities. In Chapter 10 we will see a similar result for the subadditive secretary problem. However, unlike the subadditive secretary problem over a general packing constraint, we cannot obtain a black-box reduction to linear prophet inequalities over packing constraints (with a small loss in the approximation factor). Instead, we need to revisit every step in the proof from [Rub16].

9.1.3 Other notions of online submodular optimization

Online submodular optimization has been studied in contexts beyond secretary. In online submodular welfare maximization, there are m items, n people, and each person has a monotone submodular value function. Given the value functions, the items are revealed one-by-one and the problem is to immediately and irrevocably allocate it to a person, while trying to maximize the sum of all the value functions (welfare). The greedy strategy is already half competitive. Kapralov et al. [KPV13] showed that for adversarial arrival greedy is the best possible in general (competitive ratio of 1/2), but under a "large capacities" assumption, a primal-dual algorithm can obtain $1-1/e$-competitive ratio [DHK+13]. For random arrival Korula et al. [KMZ15] showed that greedy can beat half; obtaining $1-1/e$ in this settings remains open.

Buchbinder et al. [BFS15] considered the problem of (monotone) submodular maximization with *preemption*, when the items are revealed in an adversarial order. Since sublinear competitive ratio is not possible in general with adversarial order, they consider a relaxed model where we are allowed to drop items (preemption) and give constant-competitive algorithms. Submodular maximization has also been studied in the *streaming setting*, where we have space constraints but are again allowed to drop items [BMKK14, CK15, CGQ15].

The "learning community" has looked into *experts* and *bandits* settings for submodular optimization. In these settings, different submodular functions arrive one-by-one and the algorithm, which is trying to minimize/ maximize its value, has to select a set before seeing the function. The function is then revealed and the algorithm gets the value for the selected set. The goal is to perform as close as possible to the best fixed set in hindsight. Since submodular minimization can be reduced to convex function minimization using Lovász extension, sublinear regrets are possible [HK12]. For submodular maximization, the usual benchmark is a $1-1/e$ multiplicative loss and an additive regret [SG08, GK10, GKS14].

Interestingly, to the best of our knowledge none of those problems have been studied for subadditive functions.

Organization We begin by formalizing and proving a correlation gap for non-monotone submodular functions in Section 9.2; in Section 9.3 we prove the submodular prophet inequality; and in Section 9.4 we prove the subadditive prophet inequality

9.2 Correlation Gap for Non-Monotone Submodular Functions

For monotone submodular functions, [CCPV07] proved that

$$F(\mathbf{x}) \geq (1-1/e) \cdot f^+(\mathbf{x}). \tag{9.1}$$

This result was later rediscovered by [ADSY12], who called the ratio between $f^+(\mathbf{x})$ and $F(\mathbf{x})$ *correlation gap*. It's useful in many applications since it says that up to a constant factor, picking items independently is as good as the best correlated distribution with the same element marginals.

What is the correct generalization of (9.1) to non-monotone submodular functions? It is tempting to conjecture that $F(\mathbf{x}) \geq c \cdot f^+(\mathbf{x})$ for some constant $c > 0$. However, the following example shows that even for a function as simple as the directed cut function on a two-vertex graph, this gap may be unbounded.

Example 9.2.1. *Let f be the directed cut function on the two-vertex graph $u \to v$; i.e. $f(\emptyset) = 0$, $f(\{u\}) = 1$, $f(\{v\}) = 0$, and $f(\{u,v\}) = 0$. Let $\mathbf{x} = (\epsilon, 1-\epsilon)$. Then,*

$$F(\mathbf{x}) = \epsilon^2 \ll \epsilon = f^+(\mathbf{x}).$$

It turns out that the right way to generalize (9.1) to non-monotone submodular functions is to first make them monotone:

Definition 9.2.2 (Function $f_{\max}$)**.**

$$f_{\max}(S) \triangleq \max_{T \subseteq S} f(T).$$

For non-monotone submodular f, we have that $f_{\max}$ is monotone, but it may no longer be submodular, as shown by the following example:

Example 9.2.3 (Function $f_{\max}$ is not submodular)**.** *Let f be the directed cut function on the four-vertex graph $u \to v \to w \to x$. In particular, $f(\{v\}) = 1$, $f(\{u,v\}) = 1$, $f(\{v,w\}) = 1$, and $f(\{u,w\}) = 2$.*

$$\begin{aligned} f_{\max}(\{u,v\}) - f_{\max}(\{v\}) &= 1 - 1 \\ &< 2 - 1 \\ &= f_{\max}(\{u,v,w\}) - f_{\max}(\{v,w\}). \end{aligned}$$

Finally, we are ready to define correlation gap for non-monotone functions:

Definition 9.2.4 (Correlation gap)**.** *The* correlation gap *of any set function f is*

$$\max_{\mathbf{x} \in [0,1]^n} \max_{\alpha \geq 0} \left\{ \frac{f^+(\mathbf{x})}{F_{\max}(\mathbf{x})} \middle| \sum_S \alpha_S = 1 \text{ and } \sum_S \alpha_S 1_S = \mathbf{x} \right\},$$

where $F_{\max}$ is the multilinear extension of $f_{\max}$.

Notice that for monotone f, we have that $f_{\max} \equiv f$, so Definition 9.2.4 generalizes the correlation gap for monotone submodular functions. Furthermore, one could replace f^+ with $f^+_{\max}$ in Definition 9.2.4; observe that the resulting definition is equivalent.

Theorem 9.2.5 (Non-monotone correlation gap)**.** *For any non-monotone non-negative submodular function f, the correlation gap is at most* 200.

While the constant can be improved slightly, we have not tried to optimize it, focusing instead on clarity of exposition. Our proof goes through a third relaxation, $f^*_{1/2}$.

Definition 9.2.6 ($f^*_{1/2}$)**.** *For any set function f and any $\mathbf{x} \in [0,1]^n$,*

$$f^*_{1/2}(\mathbf{x}) \triangleq \min_{S \subseteq [n]} \left\{ \mathbb{E}_{T \sim 1_S/2} \left[f(T) + \sum_{i \in [n] \setminus S} f_T(e) \cdot x_i \right] \right\}.$$

Below, we will prove (Lemma 9.2.7) that $f^+(\mathbf{x}) \le 4 \cdot f^*_{1/2}(\mathbf{x})$. We then show (Lemma 9.2.9) that $f^*_{1/2}(\mathbf{x}) \le 50 \cdot F(\mathbf{x}/2)$, which implies

$$f^+(\mathbf{x}) \le 4 \cdot f^*_{1/2}(\mathbf{x}) \le 200 \cdot F(\mathbf{x}/2). \tag{9.2}$$

Finally, to finish the proof of Theorem 9.2.5, it suffices to show that $F(\mathbf{x}/2) \le F_{\max}(\mathbf{x})$. This is easy to see since drawing T according to $\mathbf{x}/2$ is equivalent to drawing S according to $\mathbf{x}$, and then throwing out each element from S independently with probability $1/2$. For $F_{\max}(\mathbf{x})$, on the other hand, we draw the same set S and then take the optimal subset.

9.2.1 Proof that $f^+(\mathbf{x}) \le 4 \cdot f^*_{1/2}(\mathbf{x})$

Lemma 9.2.7. *For any $\mathbf{x} \in [0,1]^n$ and non-negative submodular function $f : \{0,1\}^n \to \mathbb{R}_+$,*

$$f^+(\mathbf{x}) \le 4 \cdot f^*_{1/2}(\mathbf{x}).$$

We first prove the following auxiliary claim:

Claim 9.2.8. *For any sets $S, T \subseteq [n]$,*

$$\mathbb{E}_{T_{1/2} \sim 1_T/2}\Big[f((S \setminus T) \cup T_{1/2}\Big] \ge \frac{1}{4} f(S).$$

Proof. Define a new auxiliary function $h(U) \triangleq f((S \setminus T) \cup U)$. Observe that h continues to be non-negative and submodular. We now have,

$$\begin{aligned}
&\mathbb{E}_{T_{1/2} \sim 1_T/2}[f((S \setminus T) \cup T_{1/2}] \\
&= \mathbb{E}_{T_{1/2} \sim 1_T/2}[h(T_{1/2})] \\
&\ge \frac{1}{4} h(T) && \text{(Lemma 2.4.6 for } L = H = 1/2) \\
&= \frac{1}{4} f(S) && \text{(definition of } h).
\end{aligned}$$

□

Proof of Lemma 9.2.7. Fix $\mathbf{x}$, and let $S^* = S^*(\mathbf{x})$ denote the optimal set that satisfies $f^*_{1/2}(\mathbf{x}) = \mathbb{E}_{T \sim 1_{S^*}/2}\Big[f(T) + \sum_{i \in [n] \setminus S^*} f_T(e) x_i\Big]$. Let $\{\alpha_S\}$ be the optimal distribution that satisfies $f^+(\mathbf{x}) = \sum_S \alpha_S f(S)$. Then, $\frac{1}{4} f^+(\mathbf{x}) = \frac{1}{4} \sum_S \alpha_S f(S)$

$$\begin{aligned}
&\le \sum_S \alpha_S \cdot \mathbb{E}_{T \sim 1_{S^*}/2}[f((S \setminus S^*) \cup T)] && \text{(using Claim 9.2.8)} \\
&= \sum_S \alpha_S \cdot \mathbb{E}_{T \sim 1_{S^*}/2}\left[f(T) + f_T(S \setminus S^*)\right] \\
&\le \sum_S \alpha_S \cdot \mathbb{E}_{T \sim 1_{S^*}/2}\Big[f(T) + \sum_{i \in S \setminus S^*} f_T(e)\Big] && \text{(using submodularity)}
\end{aligned}$$

$$= \mathbb{E}_{T\sim 1_{S^*}/2}\left[f(T)\sum_S \alpha_S + \sum_S \alpha_S \sum_{i\in S\setminus S^*} f_T(e)\right]$$
$$= \mathbb{E}_{T\sim 1_{S^*}/2}\left[f(T) + \sum_{i\in[n]\setminus S^*} f_T(e)x_i\right] = f^*_{1/2}(\mathbf{x}) \qquad \text{(using } \sum_S \alpha_S 1_S = \mathbf{x}\text{)}.$$

□

9.2.2 Proof that $f^*_{1/2}(\mathbf{x}) \le 50 \cdot F(\mathbf{x}/2)$

The proof of the following lemma is similar to Lemma 5 in [CCPV07].

Lemma 9.2.9. .

$$f^*_{1/2}(\mathbf{x}) \le 50 \cdot F(\mathbf{x}/2)\,.$$

Proof. Consider an exponential clock running for each element $i \in [n]$ at rate x_i. Whenever the clock triggers, we update set S to $S \cup \{i\}$. For $t \in [0,1]$, let $S(t)$ denote the set of elements in S by time t. Thus, each element belongs to $S(1)$ w.p. $1 - \exp(-x_i)$, which is between $x_i(1-\frac{1}{e})$ and x_i. Let $V(t) \triangleq \mathbb{E}_{T\sim 1_{S(t)}/2}[f(T)]$, i.e. expected value of set that picks each element in $S(t)$ independently w.p. $\frac{1}{2}$. Our goal is to show that:

$$f^*_{1/2}(\mathbf{x}) \quad \le \quad \left(\frac{2}{1-e^{-1/2}}\right)\cdot \mathbb{E}[V(1)] \quad \le \quad \left(\frac{2}{1-e^{-1/2}}\right)\left(\frac{4(e-1)}{e-2}\right)\cdot F(\mathbf{x}/2)\,. \tag{9.3}$$

We begin with the second inequality of (9.3). Consider the auxiliary submodular function $g(S) \triangleq \mathbb{E}_{T\sim 1-\exp(-\mathbf{x})}[f(S\cap T)]$, and let G denote its multilinear extension. Let S^* be a maximizer of g, and observe that

$$V(1) = G(1_{[n]}/2) \le g(S^*).$$

Observe further that, with slight abuse of notation, $F(\mathbf{x}/2) = G\left(\frac{\mathbf{x}/2}{1-\exp(-\mathbf{x})}\right)$; this is well-defined since for any $x_i \in [0,1]$, we have

$$\frac{1}{2} \le \frac{x_i/2}{1-\exp(-x_i)} \le \frac{e}{2(e-1)} < 1.$$

Moreover, since $\frac{x_i/2}{1-\exp(-x_i)}$ is bounded, Lemma 2.4.6 gives

$$F(\mathbf{x}/2) \ge \frac{1}{2}\cdot\left(1-\frac{e}{2(e-1)}\right)\cdot g(S^*)$$
$$= \frac{e-2}{4(e-1)}g(S^*) \quad = \quad \Omega(1)\cdot g(S^*).$$

We now turn to the first inequality of (9.3). Consider an infinitesimal interval interval $(t, t+dt]$. For any $i \notin S(t)$ the exponential clock triggers with probability $x_i\ dt$, so it contributes to $V(t+dt)$ with probability $x_i/2\ dt$. The probability that two clocks trigger in the

same infinitesimal is negligible ($O(dt^2)$). Therefore,

$$\mathbb{E}[V(t+dt) - V(t)] = \mathbb{E}_{S(t)}\mathbb{E}_{T\sim 1_{S(t)}/2}\left[\sum_{j\in[n]\setminus S} \frac{x_i}{2} f_T(j)\, dt\right] - O(dt^2)$$
$$\geq \frac{1}{2}\left(f^*_{1/2}(\mathbf{x}) - \underbrace{\mathbb{E}_{S(t)}\mathbb{E}_{T\sim 1_{S(t)}/2}\mathbb{E}[f(T)]}_{\mathbb{E}[V(t)]}\right) dt - O(dt^2).$$

Dividing both sides by dt and taking the limit as $dt \to 0$, we get:

$$\frac{d}{dt}\mathbb{E}[V(t)] \geq \frac{1}{2}\left(f^*_{1/2}(\mathbf{x}) - \mathbb{E}[V(t)]\right).$$

To solve the differential inequality, let $\phi(t) = \mathbb{E}[V(t)]$ and $\psi(t) = \exp(\frac{t}{2})\,\phi(t)$. We get $\frac{d\phi}{dt} \geq \frac{1}{2}(f^*_{1/2}(\mathbf{x}) - \phi(t))$ and $\frac{d\psi}{dt} = \exp(\frac{t}{2})(\frac{d\phi}{dt} + \frac{\phi(t)}{2}) \geq \exp(\frac{t}{2})\,\frac{f^*_{1/2}(\mathbf{x})}{2}$. Since $\psi(0) = \phi(0) = 0$, integration over t gives

$$\mathbb{E}[V(t)] = \phi(t) = \exp(-t/2)\,\psi(x) \geq \frac{f^*_{1/2}(\mathbf{x})}{2}(1 - \exp(-t/2)).$$

In particular, plugging in $t = 1$ completes the proof of the first inequality in (9.3). □

9.3 Submodular Prophets over Matroids

Definition 9.3.1 (Submodular Matroid Prophet)**.** *The offline inputs to the problem are:*

- *n sets $U_1, \ldots, U_n$; we denote their union $U \triangleq \bigcup_{i=1}^n U_i$;*
- *a (not necessarily monotone) non-negative submodular function $f : \{0,1\}^U \to \mathbb{R}_+$;*
- *n distributions $\mathcal{D}_i$ over subset U_i; and*
- *a matroid $\mathcal{M}$ over $[n]$*

On the i-th time period, the algorithm observes an element $X_i \in U_i$ drawn according to $\mathcal{D}_i$, independently from outcomes and actions on past and future days. The algorithm must decide (immediately and irrevocably) whether to add i and X_i to sets W and X_W, respectively, subject to W remaining independent in $\mathcal{M}$. The objective is to maximize $f(X_W)$.

Theorem 9.3.2. *There is a randomized algorithm with a competitive ratio of $O(1)$ for any* Submodular Matroid Prophet

Before proving Theorem 9.3.2, we need to define greedy online contention resolution schemes.

Greedy online contention resolution scheme Given a point $\mathbf{x}$ in the matroid polytope P of matroid $\mathcal{M}$, many submodular maximization applications like to select each element i independently with probability x_i and claim that the selected set S has expected value $F(\mathbf{x})$ [CVZ14].

The difficulty is that S need not be feasible in $\mathcal{M}$, and we can only select $T \subseteq S$ that is feasible. Chekuri et al. [CVZ14] introduced the notion of *contention resolution schemes* (CRS) that describes how, given a random S, one can find a feasible $T \subseteq S$ such that the expected value $f(T)$ will be close to $F(\mathbf{x})$.

As discussed in Chapter 8, Feldman, Svensson, and Zenklusen introduced *online contention resolution schemes* (OCRSs), which informally says that the decision of whether to select element $i \in S$ into T can be made online, even before knowing the entire set S [FSZ16]. In particular, we need their definition of *greedy* OCRS, which is a property necessary to extend applications of OCRSs from linear to submodular functions. We define it below and state the results from [FSZ16] that we use in our $O(1)$-submodular prophet inequality result over matroids.

Definition 9.3.3 (Greedy OCRS)**.** *Let $\mathbf{x}$ belong to a matroid polytope P and $S \sim \mathbf{x}$. A greedy OCRS defines a downward-closed family $\mathcal{F}_{\mathbf{x}}$ of feasible sets in the matroid. All elements reveal one-by-one if they belong to S, and when element $i \in [n]$ reveals, the greedy OCRS selects it if, together with the already selected elements, the obtained set is in $\mathcal{F}_{\mathbf{x}}$.*

Lemma 9.3.4 (Theorems 1.8 and 1.10 of [FSZ16])**.** *Given a non-negative submodular function f, a matroid $\mathcal{M}$, and a vector $\mathbf{x}$ in the convex hull of independent sets in $\mathcal{M}$, there exists a deterministic greedy OCRS that outputs a set T satisfying $\mathbb{E}_T[F(1_T/2)] \geq (1/16) \cdot F(x)$.*

Proof overview The main ingredients in the proof of Theorem 9.3.2 are known *online contention resolution schemes* (OCRS) due to Feldman, Svensson, and Zenklusen [FSZ16], and our new bound on the *correlation gap* for non-monotone submodular functions (Theorem 9.2.5).

Let $\mathbf{x} \in [0,1]^U$ denote the vector of probabilities that each element realizes (i.e. $x_{(i,j)} = \mathcal{D}_i(j)$). A naive proof plan proceeds as follows: Select elements online using the OCRS (w.r.t $\mathbf{x}$); obtain a constant factor approximation to $F(\mathbf{x})$; use a "correlation gap" to show a constant factor approximation of $f^+(\mathbf{x})$; finally, observe that $f^+(\mathbf{x})$ is an upper bound on OPT.

There are two problems with that plan: First, the OCRS of Feldman et al. applies when elements realize independently. The realization of different elements for the same day is obviously correlated (exactly one element realizes), so we cannot directly apply their OCRS. The second problem is that for non-monotone submodular function, it is in general not true that $F(\mathbf{x})$ approximates $f^+(\mathbf{x})$ (see Example 9.2.1).

The solution to both obstacles is working with $\mathbf{x}/2$ instead of $\mathbf{x}$. In Section 9.2 we showed that $F(\mathbf{x}/2)$ is a constant factor approximation of $f^+(\mathbf{x})$ (Ineq. (9.2)). Then, in Subsection 9.3.1, we give an algorithm that approximates the selection of the greedy OCRS on $\mathbf{x}/2$. Our plan is then to show:

$$\begin{aligned} ALG &= \Omega(\mathbb{E}_{S \sim OCRS(\mathbf{x}/2)}[f(S)]) && \text{(Subsection 9.3.1)} \\ &= \Omega(F(\mathbf{x}/2)) && \text{(Lemma 9.3.4)} \\ &= \Omega(f^+(\mathbf{x})) && \text{(Inequality (9.2))} \\ &= \Omega(OPT). \end{aligned}$$

9.3.1 Applying the OCRS to our setting

In this subsection we show an algorithm that obtains, in expectation, $1/2$ of the expected value of the OCRS with probabilities $\mathbf{x}/2$.

Our algorithm uses the greedy OCRS as a black box. On each day, the algorithm (sequentially) feeds the OCRS a subset of the elements U_i that can potentially arrive on that day. The subset on each day is chosen at random; it is correlated with the element that actually arrives on that day, and independent from the subsets chosen on other days. The guarantee is that the distribution over sequences fed into the OCRS is identical to the distribution induced by $\mathbf{x}/2$.

Reduction

For each i, let U_i denote the set of elements that can arrive on day i, and fix some (arbitrary) order over U_i. For a subset $S_i \subseteq U_i$, let $P^i_{\mathbf{x}/2}(S_i)$ denote the probability that the set S_i is exactly the outcome of sampling from U_i according to $\mathbf{x}/2$. When element (i,j) arrives on day i, the algorithm feeds into the OCRS a random set T_i drawn from the following distribution. With probability $\frac{P^i_{\mathbf{x}/2}(\{(i,j)\})}{x_{i,j}}$, the algorithm feeds just element (i,j), i.e. $T_i = \{(i,j)\}$; notice that this guarantees $\Pr[T_i = \{(i,j)\}] = P^i_{\mathbf{x}/2}(\{(i,j)\})$. Otherwise, the algorithm lets T_i be a random subset of U_i, drawn according to $\mathbf{x}/2$, conditioned on $|T_i| \neq 1$. This guarantees that the probability mass on subsets of size $\neq 1$ is also allocated according to $\mathbf{x}/2$.

Now, if the algorithm fed the singleton $\{(i,j)\}$ and the OCRS selected it, then the algorithm also takes $\{(i,j)\}$; otherwise the algorithm does not take $\{(i,j)\}$. (In particular, if $|T_i| \neq 1$, the algorithm ignores the decisions of the OCRS.)

Analyzing the reduction

Observe that on each day the distribution over T_i's is identical to the distribution $P^i_{\mathbf{x}/2}(\cdot)$. Since the T_i's are also independent, it means that the distribution of inputs to the OCRS is indeed distributed according to $\mathbf{x}/2$.

Conditioning on (i,j) is being fed (i.e., with probability $x_{i,j}/2$), $P^i_{\mathbf{x}/2}(\cdot)$ assigns at least $1/2$ probability to the event where no other element is also being fed (this is precisely the reason we divide $\mathbf{x}$ by 2):

$$\Pr[T_i = \{(i,j)\} \mid T_i \ni (i,j)] \geq 1/2.$$

Since the OCRS is greedy, for any history on days $1,\ldots,i-1$, if it selects (i,j) when observing set $T_i \ni (i,j)$, it would also select (i,j) when observing only this element on day i. Furthermore, since the OCRS is only allowed to select one element on day i, conditioning on the OCRS selecting (i,j), the future days $(i+1,\ldots,n)$ proceed independently of whether the algorithm also selected (i,j). Therefore, conditioning on the greedy OCRS selecting any set S_{OCRS}, the algorithm selects a subset $T_{\text{ALG}} \subseteq S_{\text{OCRS}}$ where each element appears with probability at least $1/2$.

Finally, to argue that the algorithm obtains at least $1/2$ of the expected value of the set selected by the OCRS, fix the set S_{OCRS} selected by the OCRS, and consider the submodular

Figure 9.1: Reducing a subadditive objective to a $\{0,1\}$-XOS objective.

function $g(\bar{T}) \triangleq f(S_{\text{OCRS}} \setminus \bar{T})$. Setting $\bar{T} \triangleq T_{\text{ALG}} \setminus S_{\text{OCRS}}$, we have that $f(T_{\text{ALG}}) = g(\bar{T})$. Thus by Lemma 5.4.3,

$$\mathbb{E}[f(T_{\text{ALG}})] \quad \geq \quad \frac{1}{2}\mathbb{E}[g(\emptyset)] \quad = \quad \frac{1}{2}\mathbb{E}[f(S_{\text{OCRS}})].$$

9.4 Subadditive Prophets over Packing Constraints

Definition 9.4.1 (Monotone Subadditive Downward-Closed Prophet)**.** *The offline inputs to the problem are:*

- *n sets $U_1, \ldots, U_n$; we denote their union $U \triangleq \bigcup_{i=1}^{n} U_i$;*
- *a monotone non-negative subadditive function $f : \{0,1\}^U \to \mathbb{R}_+$;*
- *n distributions $\mathcal{D}_i$ over subset U_i; and*
- *a feasibility constraint $\mathcal{F}$ over $[n]$.*

On the i-th time period, the algorithm observes an element $X_i \in U_i$ drawn according to $\mathcal{D}_i$, independently from outcomes and actions on past and future days. The algorithm must decide (immediately and irrevocably) whether to add i and X_i to sets W and X_W, respectively, subject to the constraint that W remains feasible in $\mathcal{F}$. The objective is to maximize $f(X_W)$.

Let r denote the maximum cardinality of a feasible set $S \in \mathcal{F}$. As mentioned in the introduction, Rubinstein gave an $O(\text{poly} \log n)$-approximation prophet inequality for maximizing a linear function over a packing constraint.

Theorem 9.4.2 ([Rub16])**.** *When items take values in $\{0,1\}$, there are $O(\log n)$-competitive algorithms for (additive)* Downward-Closed Secretary *and* Downward-Closed Prophet*.*

We extend the above result to subadditive functions.

Theorem 9.4.3. *There is a deterministic algorithm for* Monotone Subadditive Downward-Closed Prophet *that achieves a competitive ratio of $O\left(\log n \cdot \log^2 r\right)$.*

The proof of Theorem 9.4.3 consists of three steps (see Figure 9.1): in Subsection 9.4.1 we reduce monotone subadditive valuations over independent items to monotone XOS subadditive valuations over independent items, with a loss of $O(\log r)$, using a lemma of Dobzinski [Dob07]. Then in Subsection 9.4.2 we use a standard reduction from general XOS valuations to XOS with $\{0,1\}$ marginal contributions, losing another factor of $O(\log r)$. Finally, in Subsection 9.4.3 we use techniques from [Rub16] to give an $O(\log n)$-competitive algorithm for monotone XOS with $\{0,1\}$ marginal contributions.

9.4.1 Subadditive to XOS

Definition 9.4.4 (Monotone XOS Downward-Closed Prophet)**.** *For any set M and items $[n]$, the offline inputs to the problem are:*

- *n sets $U_1, \ldots, U_n$ of* valuations vectors *in $\mathbb{R}_+^M$; we denote their union $U \triangleq \bigcup_{i=1}^n U_i$;*
- *a monotone XOS function $\widehat{f} : \{0,1\}^U \to \mathbb{R}_+$*

$$\widehat{f}(S) \triangleq \max_{m \in M} \sum_{\mathbf{u} \in S} u_m \text{ for } S \in \{0,1\}^U;$$

- *n distributions $\mathcal{D}_i$ over subset U_i; and*
- *a feasibility constraint $\mathcal{F}$ over $[n]$, which is a collection of subsets of $[n]$.*

On the i-th time period, the algorithm observes a valuations vector $X_i \in U_i$ drawn according to $\mathcal{D}_i$, independently from outcomes and actions on past and future days. The algorithm must decide (immediately and irrevocably) whether to add i and X_i to sets W and X_W, respectively, subject to the constraint that W remains feasible in $\mathcal{F}$. The objective is to maximize $\widehat{f}(X_W)$.

In Proposition 9.4.5 we give an $O(\log n \cdot \log r)$-competitive algorithm for Monotone XOS Downward-Closed Prophet. By Dobzinski's Lemma 2.4.7, this implies an $O\left(\log n \cdot \log^2 r\right)$-competitive algorithm for Monotone Subadditive Downward-Closed Prophet.

Proposition 9.4.5. *There is a deterministic algorithm for* Monotone XOS Downward-Closed Prophet *that achieves a competitive ratio of $O(\log n \cdot \log r)$.*

9.4.2 XOS to XOS with $\{0,1\}$ coefficients

Below (Proposition 9.4.6), we give an $O(\log n)$-competitive algorithm for Monotone XOS Downward-Closed Prophet in the special case where all the vectors $v \in U$ are in $\{0,1\}^M$. First, let us show why this would imply Proposition 9.4.5.

Proof of Proposition 9.4.5 from Proposition 9.4.6. We recover separately the contributions from "tail" events (a single item taking an exceptionally high value) and the "core" contribution that is spread over many items. Run the better of the following two algorithms:

Tail Let OPT denote the expected offline optimum value. Whenever we see a feasible item whose valuations vector X_i has value at least $2OPT$, we select it. For item i, let $p_i = \Pr[X_i \geq 2OPT]$. We have

$$OPT \quad \geq \quad 2OPT \cdot \Pr[\exists i \colon X_i \geq 2OPT] \quad = \quad 2OPT \cdot \left(1 - \prod (1 - p_i)\right).$$

Dividing by OPT and rearranging, we get

$$1/2 \quad \leq \quad \prod (1 - p_i) \quad \leq \quad e^{-\sum p_i},$$

and thus

$$\sum p_i \leq \ln 2.$$

Therefore the probability that we want to take an item but can't is at most $\ln 2$, so this algorithm achieves at least a $(1 - \ln 2)$-fraction of the expected contribution from values greater than $2OPT$.

Core Observe that we can safely ignore values less than $OPT/2r$, as those can contribute a total of at most $OPT/2$. Partition all remaining values into $2 + \log r$ intervals $[OPT/2r, OPT/r]$, $\ldots$, $[OPT, 2OPT]$. The expected contribution from the values in each interval is $\Omega(1/\log r)$-fraction of the expected offline optimum without values greater than $2OPT$. Pick the interval with the largest expected contribution, round down all the values in this interval, and run the algorithm guaranteed by Proposition 9.4.6. This achieves an $\Omega\left(\frac{1}{\log n \cdot \log r}\right)$-fraction of the expected contribution from values less than or equal to $2OPT$. □

9.4.3 XOS with $\{0,1\}$ coefficients

Proposition 9.4.6. *When the X_i's take values in $\{0,1\}^M$, there is a deterministic algorithm for* Monotone XOS Downward-Closed Prophet *with $O(\log n)$ competitive ratio.*

A dynamic potential function

At each iteration, the algorithm maintains a target value τ and a target probability π, where π is the probability (over future realizations) that the current restricted prophet beats τ. We say that an outcome (i.e. a pair of item and valuations vector) is *good* if selecting it does not decrease the probability of beating the target value by a factor greater than n^2, and *bad* otherwise. Notice that all the bad items together contribute at most a $(1/n)$-fraction of the probability of beating τ. A key ingredient is that τ is updated dynamically. If the probability of observing a good outcome is too low (less than $1/4$), we deduct 1 from τ. We show (Lemma 9.4.10) that this increases π by a factor of at least 2. Since π decreases by at most an n^2 factor when we select an item, and increases by a factor of 2 whenever we deduct 1 from τ: we balance $2\log n$ deductions for every item the algorithm selects, and this gives the $O(\log n)$ competitive ratio.

So far our algorithm is roughly as follows: set a target value τ; whenever the probability π of reaching the target τ drops below $1/4$, decrease τ; if $\pi > 1/4$, sit and wait for a good outcome - one will arrive with probability at least $1/4$ (we actually do this with $\Pr[A]$ instead of π, where A is a closely related event). There is one more subtlety: what should the algorithm do if no good outcomes arrive? In other words, what if the probability of observing a good outcome is neither very low nor very close to 1, say $1/2$ or even $1 - \frac{1}{\log n}$? On one hand, we can't decrease τ again, because we are no longer guaranteed a significant increase in π; on the other hand, after, say $\Theta\left(\log^2 n\right)$ iterations, we still have a high probability of having an iteration where none of the good outcomes arrive. (If no good outcomes are coming, we don't want the algorithm to wait forever...) Fortunately, there is a simple solution: the algorithm waits for the last item with a good outcome in its support; if, against the odds, no good outcomes have yet been observed, the algorithm "hallucinates" that this last item has a good valuations vector, and selects it. In expectation, at most a constant fraction of the items we select are "hallucinated",

so the competitive ratio is still $O(\log n)$.

Notation

We let OPT denote the expected (offline) optimum. W is the set of items selected so far (W for "Wins"), and $\ell_W \triangleq \max\{i \in W\}$ is the index of the last selected item.

Let $\mathcal{F}$ denote the family of all feasible subsets of $[n]$. For any $T \subseteq [n]$, let $\mathcal{F}_T$ denote the family of feasible sets whose intersection with $\{1,\ldots,\max(T)\}$ is exactly T.

Let $X_i = \left(X_i^m\right)_{m\in M} \in \{0,1\}^M$ denote the random vector drawn for the i-th item. We use z_i to refer to the observed realization of X_i. Our algorithm will maintain a subset $M' \subseteq M$. We let

$$V_{M'}\left(\mathcal{F}, X_{[n]}\right) \triangleq \max_{S\in\mathcal{F}} \max_{m\in M'} \sum_{i\in S} (X_i)_m$$

denote the value of optimum offline solution (note that this is also a random variable).

Let $\tau = \tau(W)$ be the current target value, and $\pi = \pi(\tau, W)$ denotes the current target probability:

$$\pi(\tau, W) \triangleq \Pr\left[V_{M'}\left(\mathcal{F}_W, X_{[n]}\right) > \tau \mid X_{[\ell_W]} = z_{[\ell_W]}\right].$$

For each $y_j \in \operatorname{supp}\left(X_j\right)$, we define $\pi^{j,y_j} = \pi^{j,y_j}(\tau, W)$ to be the probability of reaching τ, given that:

- $z_j = y_j$,
- j is the next item we select, and
- item j actually contributes 1 to the offline optimum.

Formally,

$$\begin{aligned}&\pi^{j,y_j}(\tau, W)\\ &\triangleq \Pr\left[V_{M'\cap y_j}\left(\mathcal{F}_{W+j}, X_{[n]}\right) > \tau \mid X_{[\ell_W]\cup[j]} = \left(z_{[\ell_W]}, y_j\right)\right],\end{aligned}$$

where we slightly abuse notation and also use y_j to denote the set of $m \in M$ such that $y_j^m = 1$.

We say that a future outcome $\left(j, y_j\right)$ is *good* if $\pi^{j,y_j} \geq n^{-2}\cdot\pi$ and j is feasible (and otherwise it is *bad*), and let $G = \left\{\text{good } \left(j, y_j\right)\right\}$ denote the set of good future outcomes. Finally,

$$A \triangleq A(\pi, \tau, W),$$

is the event that at least one good outcome occurs.

Updated proof plan and the algorithm

The idea is to always maintain a threshold τ such that probability of one of the good outcomes to occur is large, i.e. $\Pr[A]$ is at least a constant $\frac{1}{4}$. The way we do this is by showing in Claim 9.4.7 that at any time during the execution of the algorithm, conditioned on what all has happened till

now, the probability π that the offline algorithm achieves the threshold τ gives a lower bound on $\Pr[A]$. Hence, whenever $\Pr[A]$ goes below $\frac{1}{4}$, we decrease the threshold τ, which increases π due to Lemma 9.4.10 and, indirectly, increases $\Pr[A]$ by Claim 9.4.7.

Initialize $\tau \leftarrow OPT/2$, $M' \leftarrow M$, and $W \leftarrow \emptyset$. Lemma 9.4.9 uses a concentration bound due to Ledoux to show that in the beginning $\tau = OPT/2$ satisfies $\pi > \frac{1}{4}$.

After each update to W, decrease τ until $\Pr[A] \geq 1/4$, or until $|W| > \tau$. When $\Pr[A] \geq 1/4$, reveal the values of items until observing a good outcome. When we observe a good outcome z_j, add j to W and restrict M' to its intersection with z_j. Since we restrict M to M', this gives us that at any time

$$V_{M'}(\mathcal{F}, X_W) = |W|.$$

If we reach the last item with good outcomes in its support, and none of the good outcomes realize, add this last item to G and subtract 1 from τ (without modifying M'). See also pseudocode in Algorithm 12.

We first claim that π gives us a lower bound on $\Pr[A]$ because most of the mass in π comes from good outcomes.

Claim 9.4.7. *At any point during the run of the algorithm,*

$$\Pr[A(\pi, \tau, W)] \geq \left(1 - \frac{1}{n}\right) \pi(W, \tau).$$

Proof. For each $\left(j, y_j\right) \notin G$, we have, by definition of G,

$$\pi^{j,y_j}(W, \tau) < n^{-2} \cdot \pi(W, \tau).$$

Summing over all $\left(j, y_j\right) \notin G$,

$$\begin{aligned}
&\sum_j \sum_{y_j : \left(j, y_j\right) \notin G} \Pr\left[y_j\right] \cdot \pi^{j,y_j}(W, \tau) \\
&\leq \sum_j \sum_{y_j : \left(j, y_j\right) \notin G} \Pr\left[y_j\right] \cdot \left(n^{-2} \cdot \pi(W, \tau)\right) \\
&\leq \sum_j n^{-2} \cdot \pi(W, \tau) \\
&\leq n^{-1} \cdot \pi(W, \tau).
\end{aligned}$$

Thus, most of π comes from good $\left(j, y_j\right)$'s:

$$\begin{aligned}
\Pr[A] &= \sum_j \sum_{y_j : \left(j, y_j\right) \in G} \Pr\left[y_j\right] \cdot \pi^{j,y_j}(W, \tau) \\
&\geq (1 - 1/n)\, \pi(W, \tau).
\end{aligned}$$

□

Algorithm 12 Prophet

1. $\tau \leftarrow \frac{OPT}{2}$; $M' \leftarrow M$; $W \leftarrow \emptyset$
2. while $\tau > |W|$:
   (a) $\pi \leftarrow \Pr\left[V_{M'}\left(\mathcal{F}_W, X_{[n]}\right) > \tau \mid X_{[\ell_W]} = z_{[\ell_W]}\right]$
   # π is the probability that, given the history, the offline optimum can still beat τ.
   (b) $G \leftarrow \left\{\left(j, y_j\right) : j > \ell_W \text{ AND } \pi^{j,y_j} \geq n^{-2} \cdot \pi\right\} \cap \left(\bigcup_{S \in \mathcal{F}_W} S\right)$
   # G is the set of good and feasible outcomes.
   (c) if $\Pr[A] \geq 1/4$
   # A good outcome is likely occur.
      i. $j^* \leftarrow \min\left\{j \in G\colon\ \left(j, z_j\right) \in G\right\}$
      # Wait for a good and feasible outcome.
      ii. if $j^* = \infty$
      # No good outcomes.
         A. $j^* \leftarrow \max G$
         # Select the last potentially good item.
         B. $\tau \leftarrow \tau - 1$
         # Adjust the target value to account for select an item with value 0
      iii. else
      # j^* is actually a good item.
         A. $M' \leftarrow M' \cap z_j$
      iv. $W \leftarrow W \cup \{j^*\}$
   (d) else
      i. $\tau \leftarrow \tau - 1$
      # decrease target value τ until $\Pr[A] \geq 1/4$.

Concentration for the beginning

Theorem 9.4.8. *[Led97, Theorem 2.4] There exists some constant $K > 0$ such that the following holds. Let Y_i's be independent (but not necessarily identical) random variables in some space S; let C be a countable class of measurable functions $f : S \to [0,1]$; and let $Z = \sup_{f \in C} \sum_{i=1}^{n} f(Y_i)$. Then,*

$$\Pr[Z \geq \mathbb{E}[Z] + t] \leq \exp\left(-\frac{t}{K} \cdot \log\left(1 + \frac{t}{\mathbb{E}[Z]}\right)\right).$$

To make the connection to our setting, let Y_i be the vector in $[0,1]^{\mathcal{F} \times M}$ whose (S,m)-th coordinate is X_i^m if $i \in S$, and 0 otherwise. Let $f_{S,m}(Y_i) \triangleq [Y_i]_{S,m}$, so $\sum_{i=1}^{n} f_{S,m}(Y_i)$ is simply the value of the set S under the m-th summation in the XOS representation of the valuation function. Let $C \triangleq \{f_S\}_{S \in \mathcal{F}}$. The above concentration inequality can now be written as

$$\Pr\left[V\left(\mathcal{F}, X_{[n]}\right) \geq OPT + t\right] \leq \exp\left(-\frac{t}{K} \cdot \log\left(1 + \frac{t}{OPT}\right)\right). \tag{9.4}$$

Lemma 9.4.9. *Assume $OPT \geq \Omega(\log n)$. Then,*

$$\Pr\left[V\left(\mathcal{F}, X_{[n]}\right) \geq \frac{OPT}{2}\right] > 1/4.$$

Proof. We have,

$$OPT = \int_{-OPT}^{\infty} \Pr\left[V\left(\mathcal{F}, X_{[n]}\right) \geq OPT + t\right] dt, \tag{9.5}$$

which can be decomposed as to integrals over $[-OPT, -OPT/2]$, $[-OPT/2, OPT]$, and $[OPT, \infty]$.

The first two integrals can be easily bounded as

$$\int_{-OPT}^{-OPT/2} \Pr\left[V\left(\mathcal{F}, X_{[n]}\right) \geq OPT + t\right] dt \quad \leq \quad \int_{-OPT}^{-OPT/2} 1 \cdot dt \quad \leq \quad \frac{OPT}{2}$$

and

$$\begin{aligned}
&\int_{-OPT/2}^{OPT} \Pr\left[V\left(\mathcal{F}, X_{[n]}\right) \geq OPT + t\right] dt \\
&\leq \int_{-OPT/2}^{OPT} \Pr\left[V\left(\mathcal{F}, X_{[n]}\right) \geq OPT/2\right] dt \\
&\leq \frac{3OPT}{2} \cdot \Pr\left[V\left(\mathcal{F}, X_{[n]}\right) > \frac{OPT}{2}\right].
\end{aligned}$$

For the third integral we use the concentration bound (9.4):

$$\begin{aligned}
&\int_{OPT}^{\infty} \Pr\left[V\left(\mathcal{F}, X_{[n]}\right) \geq OPT + t\right] dt \\
&\leq \int_{OPT}^{\infty} \exp\left(-\frac{t}{K} \cdot \log\left(1 + \frac{t}{OPT}\right)\right) dt
\end{aligned}$$

$$\leq \int_{OPT}^{\infty} \exp\left(-\frac{t}{K}\right) dt \quad = \quad \left[Ke^{-t/K}\right]_{OPT}^{\infty} \quad = \quad K \cdot e^{-OPT/K},$$

which is negligible since $OPT = \omega(1)$.

Plugging into (9.5), we have:

$$OPT \leq \frac{OPT}{2} + \frac{3OPT}{2} \cdot \Pr\left[V\left(\mathcal{F}, X_{[n]}\right) > \frac{OPT}{2}\right] + o(1),$$

and after rearranging we get

$$\Pr\left[V\left(\mathcal{F}, X_{[n]}\right) > \frac{OPT}{2}\right] \geq 1/3 - o(1). \qquad \square$$

Main lemma

Lemma 9.4.10. *At any point during the run of the algorithm, if* $\Pr[A] \leq 1/4$*, then subtracting* 1 *from* τ *doubles* π*; i.e.*

$$\pi(W, \tau - 1) \geq 2\pi(W, \tau).$$

Proof of Lemma 9.4.10. Consider the event that the optimum solution (conditioned on the items W we already selected and the realizations $z_{[\ell_W]}$ we have already seen) reaches τ. We can write it as a union of disjoint events, depending on the next item $j > \ell_W$ that is part of the optimum solution, and its possible realizations y_j:

$$\pi(W, \tau) = \sum_j \sum_{y_j} \Pr\left[y_j\right] \cdot \underbrace{\Pr\left[V_{M' \cap y_j}\left(\mathcal{F}_{W \cup \{j\}}, X_{[n]}\right) > \tau \mid X_{[\ell_W] \cup [j]} = \left(z_{[\ell_W]}, y_j\right)\right]}_{\pi^{j, y_j}(W, \tau)}.$$

We break the RHS into the sum over $\left(j, y_j\right)$'s that are good and the sum over those that are bad. Now, Claim 9.4.7 gives

$$\sum_j \sum_{y_j : \left(j, y_j\right) \in G} \Pr\left[y_j\right] \cdot \pi^{j, y_j}(W, \tau) \geq (1 - 1/n)\, \pi(W, \tau). \tag{9.6}$$

Since $y_j \in \{0,1\}^M$, each item can contribute at most 1 to the offline optimum. Therefore:

$$\underbrace{\Pr\left[V_{M' \cap y_j}\left(\mathcal{F}_{W \cup \{j\}}, X_{[n]}\right) > \tau \mid X_{[\ell_W] \cup [j]} = \left(z_{[\ell_W]}, y_j\right)\right]}_{\pi^{j, y_j} = \pi^{j, y_j}(\tau, W)} \quad \leq \quad \pi^{j, 0}(W, \tau - 1).$$

Plugging into (9.6), we have

$$\begin{aligned}(1 - 1/n) \cdot \pi(W, \tau) \quad &\leq \quad \sum_j \sum_{y_j : \left(j, y_j\right) \in G} \Pr\left[y_j\right] \cdot \pi^{j, 0}(W, \tau - 1) \\ &\leq \quad \sum_j \Pr\left[\left(j, y_j\right) \in G\right] \cdot \pi^{j, 0}(W, \tau - 1)\end{aligned}$$

$$\leq \left(\sum_j \Pr\left[\left(j, y_j\right) \in G\right]\right) \cdot \pi\left(W, \tau - 1\right), \tag{9.7}$$

where the second inequality follows because $\pi^{j,0}\left(W, \tau - 1\right)$ doesn't depend on y_j, and the third because conditioning on the j-th item being 0 can only decrease the probability of reaching $\tau - 1$.

Recall that A is the union of all the events $\left(j, y_j\right) \in G$. Therefore,

$$\Pr\left[A\right] \geq \sum_j \Pr\left[\left(j, y_j\right) \in G\right]\left(1 - \Pr\left[A\right]\right)$$

Plugging in $\Pr\left[A\right] < 1/4$, we get that $\sum_j \Pr\left[\left(j, y_j\right) \in G\right] < 1/3$. Plugging into (9.7) and rearranging, we get

$$\pi\left(W, \tau - 1\right) \geq \frac{3n}{n-1}\pi\left(W, \tau\right). \qquad \square$$

Putting it all together

Lemma 9.4.11. *At any point during the run of the algorithm,*

$$\tau \geq \frac{OPT}{2} - \left(2\log n + 1\right) \cdot |W| - 2$$

Proof. We prove by induction that at any point during the run of the algorithm,

$$\log \pi \geq -2 - \left(2\log n + 1\right) \cdot |W| + \left(\frac{OPT}{2} - \tau\right). \tag{9.8}$$

After initialization, $\log \pi \geq -2$ by Lemma 9.4.9. By definition of G, whenever we add an item to W, we decrease $\log \pi$ by at most $2\log n$ - hence the $2\log n \cdot |W|$ term. Notice that when the algorithm "hallucinates" a 1, we also decrease τ by 1 to correct for the hallucination - at any point during the run of the algorithm, this has happened at most $|W|$ times. Recall that we may also decrease τ in the last line of Algorithm 12 (in order to increase π); whenever we do this, τ decreases by 1, but π doubles (by Lemma 9.4.10), so $\log \pi$ increases by 1, and Inequality (9.8) is preserved.

Finally, since π is a probability, we always maintain $\log \pi \leq 0$. $\square$

We are now ready to complete the proof of Theorem 9.4.3.

Proof of Proposition 9.4.6. The algorithm always terminates after at most $O\left(OPT\right)$ decreases to the value of τ. By Lemma 9.4.11, when the algorithm terminates, we have $|W| \geq \tau \geq \frac{OPT}{2} - \left(2\log n + 1\right) \cdot |W| - 2$, and therefore in particular $|W| \geq \frac{OPT-4}{4\log n+4}$.

Finally, recall that sometimes the algorithm "hallucinates" good realizations, i.e. for some items $i \in W$ that we select, $X_i = 0$. However, each time we add an item, the probability that we add a zero-value item is at most 3/4 (by the condition $\Pr\left[A\right] > 1/4$). Therefore in expectation the value of the algorithm is at least $|W|/4$. $\square$

Chapter 10

The Secretary and the Prophet Secretary Models

10.1 Introduction

In the DIAMOND-SELLING scenario discussed in the introduction (§1.1), there is a sequence of n buyers arriving with different *values* to your *single* item (diamond). On arrival a buyer offers a take-it-or-leave-it value for your item. The question is to decide which buyer to assign the item to in order to maximize the value. In Chapter 8 we model this problem as a prophet inequality model where the buyer arrival order is chosen by an *adversary*. In practice, however, it is often conceivable that there is no adversary acting against you. Can we design better strategies when the arrival order is chosen *uniformly at random*?

10.1.1 Model and Results

Suppose the arrival order of buyers in the DIAMOND-SELLING scenario is chosen uniformly at random. There are two natural ways to model this problem. Firstly, in the *secretary model* the decision maker has no prior information about the valuations of buyers to arrive (except their cardinality, n), but the buyers are guaranteed to arrive in a uniformly random order. This is a classical problem in stopping theory and Dynkin gave a tight e-competitive algorithm in in 1963 [Dyn63]. Secondly, in the *prophet secretary* model the decision maker knows the probability distributions of all the n buyers (similar to a prophet inequality) and also the buyers are guaranteed to to arrive in a uniformly random order. This model was introduced in [EHLM17] and they gave a $(1 - 1/e)$-competitive algorithm. In §10.2, we give an alternate proof of their result using OCRSs defined in Chapter 8.

Theorem 10.1.1. *There exists a* tight $(1 - 1/e)$*-OCRS for a* 1*-uniform matroid when the arrival order is chosen uniformly at random. This implies a* $(1 - 1/e)$*-prophet secretary for single item.*

Although the factor $1 - 1/e$ is tight for OCRSs, some recent works have shown that it is possible to go beyond for the prophet secretary problem [ACK18, AEE+17, CFH+17]. In §10.3 we present a new simple proof of this result for the special case of identically distributed buyers.

Theorem 10.1.2 ([HK+82, CFH+17]). *There exists a* tight *α-competitive algorithm for the prophet secretary problem when the number of buyers tends to infinity and their value distributions are identical, where α (≈ 0.74) satisfies the equation $\int_{y=0}^{y=1} \frac{dy}{y - y \ln y - 1 + \frac{1}{\alpha}} = 1$ with $y(0) = 1$.*

Motivated in part by applications to mechanism design, multiple-choice variants of the secretary problem have also been widely studied in the online algorithms community. The seminal papers of [HKP04, Kle05] introduced a secretary problem subject to a cardinality constraint (and [Kle05] also obtained a $1 - O(1/\sqrt{r})$-competitive algorithm). In 2007, Babaioff et al. introduced the famous *matroid secretary problem* [BIK07]. For general packing constraints, an $O(\log n \log r)$-competitive algorithm was recently obtained by Rubinstein [Rub16], where r is the size of the largest feasible set. All these results trivially carry over to the prophet secretary model since these secretary algorithms do not even need probability distributions.

In all the works mentioned in the previous paragraphs, the goal is to maximize the sum of selected items' values, i.e., an additive objective is optimized. For the secretary problem, there has also been significant work on optimizing more general, combinatorial objective functions. A line of great works [BHZ13, FNS11a, BUCM12, FZ15] on secretary problem with submodular valuations culminated with a general reduction by Feldman and Zenklusen [FZ15] from any submodular function to linear valuations with only $O(1)$ loss. Going beyond submodular is an important problem [FI17], but for subadditive objective functions there is a daunting $\Omega(\sqrt{n})$ lower bound on the competitive ratio [BHZ13][1]. We circumvent this impossibility by designing inefficient (but information theoretically possible) algorithms. In §10.4 we consider combinatorial variants of this problem where we want to maximize a subadditive function over a packing constraint and prove the following result (based on our work in [RS17]).

Theorem 10.1.3. *There exists an $O(\log n \cdot \log^2 r)$-competitive algorithm for monotone subadditive secretary problem subject to any packing constraints.*

Of course, all the above secretary results again trivially carry over to the prophet secretary model. In Chapter 11 we show how to improve some of them in the prophet secretary model.

10.1.2 Related Work

Starting with the work of Dynkin [Dyn63], there has been a long line of research on secretary problems. One of the first generalizations is the *multiple-choice secretary* problem in which we are allowed to pick k items and the goal is to maximize their sum [HKP04]. Kleinberg [Kle05] gives a $(1 - O(\sqrt{1/k}))$-approximation algorithm.

The connection between secretary problems and online auctions is first explored in Hajiaghayi et al. [HKP04]. Its generalization to matroids is considered in [BIK07, Lac14, FSZ15] and to matchings in [GM08, KP09, MY11, KMT11, KRTV13, GS17].

Secretary problems have also been studied beyond a matroid/matching. Submodular variants of the secretary problem have been considered in [BHZ13, GRST10, FZ15, KMZ15]. Rubinstein [Rub16] considers these problems for arbitrary downward-closed constraints. The sec-

[1]For general packing constraints, even with additive valuations one cannot obtain efficient algorithms using membership queries [Rub16].

retary model has been studied for many classical combinatorial problems (see e.g., [Mey01, GGLS08, GHK+14, DEH+17]).

In the prophet secretary model, Esfandiari et al. [EHLM17] give a $(1-1/e)$-approximation in the special case of a single item. Going beyond $1-1/e$ has been challenging. Only recently, Abolhasani et al. [AEE+17] and Correa et al. [CFH+17] improve this factor for the single item i.i.d. setting, and Azar et al. [ACK18] for single item non-i.i.d. setting. Extending this result to matroids is an interesting open question.

10.2 Prophet Secretary via Optimal $(1-1/e)$-OCRS

Given $\mathbf{x} \in [0,1]^n$ satisfying $\sum_i x_i \leq 1$, in this section we prove Theorem 10.1.1 that gives a $(1-1/e)$-selectable *tight* OCRS algorithm for uniformly random arrival order. Intuitively, this means there exists an algorithm that selects each element i when it is active (which happens w.p. x_i) at least $(1-1/e)$ fraction of the times. By the reduction from Chapter 8, this $(1-1/e)$-selectable OCRS directly implies a $(1-1/e)$-prophet secretary.

We first prove that no OCRS can be better than $(1-1/e)$-selectable for random arrival order. Consider the feasible solution $\mathbf{x}$ with $x_i = 1/n$ for every i. We argue that no online algorithm can guarantee each element is selected w.p. greater than $\frac{(1-1/e)}{n}$. This is because for the product distribution, w.p. $1/e$ none of the n elements is active (more precisely, w.p. $(1-1/n)^n$). Hence the OCRS algorithm, which only selects active elements, selects some element w.p. $1-1/e$. This implies on average it cannot select every element w.p. greater than $\frac{(1-1/e)}{n}$.

Next, we show how to design the $(1-1/e)$-selectable OCRS. Notice that the random arrival order can be emulated by assuming each element i selects a time t_i uniformly at random in the interval $[0,1]$ and then arrives at time t_i.

Theorem 10.2.1. *An algorithm that selects an active element i arriving at time $t \in [0,1]$ with probability $\exp(-t \cdot x_i)$ (and ignores i otherwise) is $(1-1/e)$-selectable for a rank 1 matroid; that is, on average this algorithm considers (not ignore) any element i at least $(1-1/e)$ fraction of the times.*

Proof. By reaching time t (element j), let us denote the event that no element is selected before time t (element j's arrival). We start by noticing that for any element $i \in [n]$,

$$\begin{aligned}\Pr[i \text{ is considered}] &= \int_{t=0}^{1} \Pr[i \text{ is considered at time } t \mid \text{reach time } t \ \&\ i \text{ arrives at } t] \\ &\qquad \cdot \Pr[\text{reach time } t \mid i \text{ arrives at } t] \cdot dt \\ &= \int_{t=0}^{1} \exp(-t \cdot x_i) \cdot \Pr[\text{reach time } t \mid i \text{ arrives at } t] \cdot dt. \end{aligned} \tag{10.1}$$

Now we can simplify $\Pr[\text{reach time } t \mid i \text{ arrives at } t]$ by

$$\prod_{j \neq i} \Big(1 - \Pr[j \text{ arrives before } t \ \&\ \text{is active} \ \&\ \text{is considered} \mid \text{reach } j]\Big)$$

$$= \prod_{j \neq i} \left(1 - x_j \cdot \Pr[j \text{ arrives before } t \text{ \& is considered} | \text{ reach } j]\right)$$
$$= \prod_{j \neq i} \left(1 - x_j \cdot \int_{a=0}^{t} \exp(-a \cdot x_j) \cdot da\right) \quad = \quad \prod_{j \neq i} \exp(-t \cdot x_j).$$

Now combining this equation with (10.1), we get

$$\Pr[i \text{ is considered}] = \int_{t=0}^{1} \exp(-t \cdot x_i) \cdot \prod_{j \neq i} \exp(-t \cdot x_j) \cdot dt$$
$$\geq \int_{t=0}^{1} \exp(-t) \cdot dt \quad = \quad 1 - \frac{1}{e},$$

where the inequality uses $\sum_i x_i \leq 1$.

□

10.3 A Simple Optimal I.I.D. Prophet Secretary

In this section we give a simple proof of Theorem 10.1.2, which gives a *tight* α-prophet secretary ($\alpha \approx 0.74$) in the special case of i.i.d. buyers for large n. We only show an algorithm achieving this factor α and refer the readers to the original papers that proved this theorem for a matching hardness example [HK+82, CFH+17].

The first crucial observation is that since the buyers are i.i.d. and n is large, we can imagine the arrival of the buyers as a Poisson arrival of i.i.d. buyers at rate n from time 0 to 1. This means that the inter-arrival time of the buyers is given by an exponentially distribution.

Let the value distribution of each i.i.d. buyer be $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$. Let $F(x) := \Pr[X \leq x] = \int_{a=0}^{x} f(a) \cdot da$ and $G(x) := 1 - F(x)$. Moreover, let Max denote $\max\{X_1, \ldots, X_n\}$ and Alg denote the value of the element chosen by our algorithm.

Our algorithm is threshold based, i.e., it selects a buyer only if its value is above the threshold.

Definition 10.3.1 (Functions h and τ)**.** *Let $h(t)$ denote the probability that no buyer is selected till time t by a threshold $\tau(t)$ algorithm, which selects buyer at time $t \in [0,1]$ only if its value is at least $\tau(t)$. This means $h(0) = 1$ and $h(t)$ is a non-increasing function of t.*

For Poisson arrival of buyers at rate n, the above definitions of h and τ imply that at any time time $t \in [0,1]$, we have $-h'(t)dt = h(t) \cdot n \cdot dt \cdot \Pr[X \geq \tau]$. This gives

$$\tau(t) = G^{-1}\left(-\frac{1}{n}\frac{h'(t)}{h(t)}\right).$$

Recollect that

$$\mathbb{E}[\text{Max}] = \int_{x=0}^{\infty} \Pr[\text{Max} \geq x] \cdot dx \qquad \text{and} \qquad \mathbb{E}[\text{Alg}] = \int_{x=0}^{\infty} \Pr[\text{Alg} \geq x] \cdot dx.$$

Thus, to prove Theorem 10.1.2, it suffices to prove for all $x > 0$,

$$\Pr[\text{Alg} \geq x] \quad \geq \quad \alpha \cdot \Pr[\text{Max} \geq x]. \tag{10.2}$$

Before proving this inequality, we first simplify the expressions for $\Pr[\text{Max} \geq x]$ and $\Pr[\text{Alg} \geq x]$ in the following Lemma 10.3.2 and Lemma 10.3.3, respectively. These lemmas are inspired from [AEE+17].

Lemma 10.3.2. *For any $x \geq 0$,*

$$\Pr[\mathit{Max} \geq x] = 1 - \exp(-n \cdot G(x)).$$

Proof. The probability that a buyer arrives with value more than x between time t and $t + dt$ is $n \cdot G(x)dt$. This is equivalent to Poisson arrivals with rate $n \cdot G(x)$, which implies that the probability of no such buyer arriving till time 1 is $\exp(-n \cdot G(x))$. □

Consider an algorithm that sets threshold $\tau(t)$ for a buyer arriving at time t.

Lemma 10.3.3. *For any $x \geq 0$,*

$$\Pr[\text{Alg} \geq x] = \left(1 - h(s(x)) + n \cdot G(x) \int_{r=s(x)}^{1} h(r) \cdot dr\right), \text{ where } \tau(s(x)) = x.$$

Proof. If the item is sold before time $s(x)$, clearly $\text{Alg} \geq x$. For time $r \geq s(x)$, the probability that item gets sold between r and $r + dr$ with value above x is

$$\begin{aligned}&\Pr[\text{Item unsold till } r] \cdot \Pr[\text{Arrival between } r \text{ and } r + dr] \cdot \Pr[\text{Arrived buyer value} \geq x] \\ &= h(r) \cdot n \cdot dr \cdot G(x).\end{aligned}$$

□

Finally, we prove the following Lemma 10.3.4, which implies (10.2) and gives Theorem 10.1.2 as a corollary.

Lemma 10.3.4. *For any $x \geq 0$, we get (10.2) is true.*

Proof. By Lemma 10.3.2 and Lemma 10.3.3, we know to prove this lemma it suffices to show

$$1 - h(s(x)) + n \cdot G(x) \int_{r=s(x)}^{1} h(r) \cdot dr \geq \alpha \cdot (1 - \exp(-n \cdot G(x))) . \tag{10.3}$$

Consider the function

$$\rho(s, g) := 1 - h(s) + g \cdot \int_{r=s}^{1} h(r) \cdot dr - \alpha \cdot (1 - \exp(-g)) .$$

To prove (10.3), it suffices to show $\min_{g \geq 0, 1 \geq s \geq 0} \rho(s, g) \geq 0$. Since the boundary $g = 0$ already satisfies $\rho(s, 0) \geq 0$, assume that at the minimizer

$$\frac{\partial \rho}{\partial g} = \int_{r=s}^{1} h(r) \cdot dr - \alpha \cdot \exp(-g) = 0.$$

Substituting this solution into ρ, and letting $y(s) := \frac{1}{\alpha}\int_{r=s}^{1} h(r)\cdot dr$, i.e., $-y'(s) = \frac{1}{\alpha}h(s)$, we get

$$\min_{g\geq 0, 1\geq s\geq 0} \rho(s,g) = \min_{1\geq s\geq 0}\{1+\alpha y' - \ln(y)\cdot \alpha y - \alpha + \alpha y\}.$$

Suppose we choose function $y(s)$ satisfying

$$y' = y\ln y - y + 1 - \frac{1}{\alpha}. \tag{10.4}$$

This gives $\min_{g\geq 0,1\geq s\geq 0}\rho(s,g) = 0$ and $h = \alpha\cdot(y - y\ln y - 1) + 1$. Moreover, since

$$\begin{aligned} 1 &= h(0) \\ &= -\alpha\cdot y'(0) \\ &= -\alpha\cdot\left(y(0)\cdot\ln y(0) - y(0) + 1 - \frac{1}{\alpha}\right), \end{aligned}$$

we get $y(0) = 1$. Finally, integrating (10.4) from $s = 0$ to $s = 1$ and using $y(1) = 0$ (by definition),

$$\begin{aligned} 1 &= \int_{s=0}^{s=1} \frac{dy}{y\ln y - y + 1 - \frac{1}{\alpha}} \\ &= \int_{y=1}^{y=0} \frac{dy}{y\ln y - y + 1 - \frac{1}{\alpha}} \\ &= \int_{y=0}^{y=1} \frac{dy}{y - y\ln y - 1 + \frac{1}{\alpha}}. \end{aligned}$$

Numerically, this equation givens $\alpha \approx 0.74$. □

10.4 Combinatorial Secretary Problems

We now consider the problem of maximizing a combinatorial function in the secretary model. A remarkable result of Feldman and Zenklusen shows that if our objective is submodular and the constraints form a matroid, then this problem is no harder than maximizing a linear function over a matroid [FZ15]. This reduces the submodular secretary problem to the matroid secretary problem, where a constant factor is still elusive but we know $O(\log\log n)$-competitive algorithms [Lac14, FSZ15].

In this section we focus on the more difficult subadditive secretary problem over arbitrary downward-closed constraints. Similar to §9.4, since one cannot hope to optimize general subadditive functions in polynomial time, we only aim for the best possible information theoretic approximation factors. We first formally define the problem.

Definition 10.4.1 (Monotone Subadditive Downward-Closed Secretary)**.** *Consider n items, a monotone subadditive valuation function from subsets of items to $\mathbb{R}_{\geq 0}$, and an arbitrary downward-closed set system $\mathcal{F}$ over the items; both f and $\mathcal{F}$ are adversarially chosen. The algorithm receives as input n (but not $\mathcal{F}$ or f). The items arrive in a* uniformly random *order. Initialize W as the*

Figure 10.1: Reducing a subadditive objective to an additive objective with additional packing constraints.

empty set. When item i arrives, the algorithm observes all feasible subsets of items that have already arrived, and their valuation in f. The algorithm then decides (immediately and irrevocably) whether to add i to set W, while always ensuring that W is feasible in $\mathcal{F}$. The goal is to maximize $f(W)$.

Our main result in this section is an $O(\text{poly}\log(n))$-competitive algorithm for the subadditive secretary problem by reducing maximizing a monotone subadditive function over packing constraints to maximizing a linear function over packing constraints (Theorem 10.1.3). Since we know an $O(\log n \log r)$-competitive algorithm for the latter problem [Rub16], and because the reduction only loses an $O(\log r)$ factor, we get an $O(\log n \log^2 r)$-competitive algorithm for subadditive secretary over packing constraints. The crucial ideas in the reduction is to observe that a monotone subadditive function can be approximated by an XOS function and that an unweighted XOS function can be interpreted as another packing constraint (see Figure 10.1).

Proof of Theorem 10.1.3. Let $T^\star$ be the set chosen by the offline algorithm ($OPT = f(T^\star)$). By Lemma 2.4.7 there exists a $p_{T^\star}$ such that for every $S \subseteq T^\star$:

$$f(S) \geq p_{T^\star} \left|S \cap T^\star\right|; \tag{10.5}$$

$$OPT = f(T^\star) = O\left(p_{T^\star} \left|T^\star\right| \log \left|T^\star\right|\right) = O\left(p_{T^\star} \left|T^\star\right|\right) \log r. \tag{10.6}$$

Assume that we know $p_{T^\star}$ (discussed later). We define a new feasibility constraint $\mathcal{F}'$ as follows: a set $T \subseteq [n]$ is feasible in $\mathcal{F}'$ iff it is feasible in $\mathcal{F}$ and for every subset $S \subseteq T$, we have $f(S) \geq p_{T^\star} |S|$. Notice that because we also force the condition on all subsets of T, $\mathcal{F}'$ is downward-closed and it does not depend on the order of arrival.

We run the algorithm for $\{0,1\}$-valued (additive) Downward-Closed Secretary (as guaranteed by Theorem 9.4.2) with feasibility constraint $\mathcal{F}'$ where all values are 1. By (10.5), $T^\star$ is feasible in $\mathcal{F}'$, and by (10.6) $p_{T^\star} |T^\star| = \Omega\left(\frac{OPT}{\log r}\right)$. Therefore, the additive $\{0,1\}$-values algorithm returns a set T^{ALG} of size $\left|T^{ALG}\right| = \Omega\left(\frac{OPT}{p_{T^\star} \log n \log r}\right)$. Furthermore, T^{ALG} is also feasible in $\mathcal{F}'$, i.e.

$$\begin{aligned} f(T^{ALG}) &\geq p_{T^\star} \left|T^{ALG}\right| \\ &= \Omega\left(\frac{OPT}{\log n \log r}\right). \end{aligned} \tag{10.7}$$

Guessing $p_{T^\star}$

Finally, we don't actually know $p_{T^\star}$, but we can guess it correctly, up to a constant factor, with probability $1/\log r$. We run the classic secretary algorithm over the first $n/2$ items, where we use the value of the singleton $f(\{i\})$ as "the value of item i": Observe the first $n/4$ items and select none; then take the next item whose value is larger than every item observed so far. With constant probability this algorithm selects the item with the largest value, which we denote by M.

Also, with constant probability the algorithm sees the item with the largest value too early and does not select it. Assume that this is the case. Since we obtained expected value of $\Omega(M)$ on the first $n/2$ items we can, without loss of generality, ignore values less than M/r. In particular, we know that $p_{T^\star} \in [M/r, M]$. Pick $\alpha \in \{M/r, M/(2r), \ldots, M/2, M\}$ uniformly at random, and use it instead of $p_{T^\star}$ to define $\mathcal{F}'$. With probability $1/\log r$, $p_{T^\star} \in [\alpha, 2\alpha]$, in which case the algorithm returns a set T^{ALG} satisfying (10.7). □

Chapter 11

Prophet Secretary for Matroids and Combinatorial Auctions via Residuals

11.1 Introduction

In this chapter, we consider generalizations of the single-item prophet secretary problem consider in Chapter 10 to combinatorial settings. Suppose a sequence of n buyers corresponding to elements of a matroid[1] arrive one-by-one in uniformly random order and offer take-it-or-leave-it value for being accepted. Which buyers should we accept to maximize our total value when we can only accept an independent set of buyers in this matroid.

In the prophet inequality model, in Chapter 8 we saw a $1/2$-approximation strategy to this problem, i.e., the value of their strategy, in expectation, is at least half of the value of the expected offline optimum that selects the best set of buyers in *hindsight*. Simple examples show that for adversarial arrival one cannot improve this factor. On the other hand, if we are also allowed to control the arrival order of the buyers, Yan [Yan11] gives a $1 - 1/e \approx 0.63$-approximation strategy. But what if the arrival order is neither adversarial and nor in your control. In particular, can we beat the $1/2$-approximation for a uniformly random arrival order?

Matroid Prophet Secretary Problem (MPS): *Given a matroid $\mathcal{M} = ([n], \mathcal{I})$ on n buyers (elements) and independent probability distributions on their values, suppose the outcome buyer values are revealed in a uniformly random order. Whenever a buyer value is revealed, the problem is to immediately and irrevocably decide whether to* select *the buyer. The goal is to maximize the sum of values of the selected buyers, while ensuring that they are always feasible in $\mathcal{I}$.*

Besides being a natural problem that relates two important Stopping Theory models, MPS is also interesting because of its applications in mechanism design. Often while designing mechanisms, we have to balance between maximizing revenue/welfare and the simplicity of the

[1] A matroid $\mathcal{M}$ consists of a ground set $[n] = \{1, 2, \ldots, n\}$ and a non-empty downward-closed set system $\mathcal{I} \subseteq 2^{[n]}$ satisfying the matroid exchange axiom: for all pairs of sets $I, J \in \mathcal{I}$ such that $|I| < |J|$, there exists an element $x \in J$ such that $I \cup \{x\} \in \mathcal{I}$. Elements of $\mathcal{I}$ are called independent sets.

mechanism. While there exist optimal mechanisms such as VCG or Myerson's mechanism, they are impractical in real markets [AM06, Rot07]. On the other hand, simple *Sequentially Posted Pricing mechanisms*, where we offer take-it-or-leave-it prices to buyers, are known to give good approximations to optimal mechanisms. This is because the problem gets reduced to designing a prophet inequality [CHMS10, Yan11, Ala14, KW12, FGL15].

Esfandiari et al. [EHLM17] study MPS in the special case of a rank 1 matroid and give a $(1-1/e)$-approximation algorithm. For general matroids, as in the original models of [CHMS10, Yan11, KW12], it was unclear prior to the work of this paper whether beating the factor of $1/2$ is possible. In §11.4 we prove the following result, which is based on our work in [EHKS18].

Theorem 11.1.1. *There exists a $(1-1/e)$-approximation algorithm to MPS.*

Note that the approximation in this theorem as well as the following ones compare to the expected *optimal offline solution* for the particular outcomes of the distributions. That is, in the case of matroids, we have $\mathbb{E}[\text{Alg}] \geq (1-1/e) \cdot \mathbb{E}[\max_{I \in \mathcal{I}} \sum_{i \in I} v_i]$, where v_i is the value of buyer i[2].

Next, let us consider a combinatorial auctions setting. Suppose there are n buyers that take combinatorial valuations (say, submodular) for m indivisible items from n independent probability distributions. The problem is to decide how to allocate the items to the buyers, while trying to maximize the *welfare*—the sum of valuations of all the buyers. Feldman et al. [FGL15] show that for XOS[3] (a generalization of submodular) valuations there exist *static prices* for items that gets a $1/2$-approximation for buyers arriving in an adversarial order. Since this factor cannot be improved for adversarial arrival, this leaves an important open question if we can design better algorithms when the arrival order can be controlled. Or ideally, we want to beat $1/2$ even when the arrival order cannot be controlled but is chosen uniformly at random.

Combinatorial Auctions Prophet Secretary Problem (CAPS): *Suppose n buyers take XOS valuations for m items from n independent probability distributions. The outcome buyer valuations are revealed in a uniformly random order. Whenever a buyer valuations is revealed, the problem is to immediately and irrevocably assign a subset of the remaining items to the buyer. The goal is maximize the sum of the valuations of all the buyers for their assigned subset of items.*

In §11.3.2 we improve the online approximation result of [FGL15] for random order.

Theorem 11.1.2. *There exists a $(1-1/e)$-approximation algorithm to CAPS.*

Given access to demand and XOS oracles for stochastic utilities of different buyers, the algorithm in Theorem 11.1.2 can be made efficient. This is interesting because it matches the best possible $(1-1/e)$-approximation for XOS-welfare maximization in the offline setting [DNS10, Fei09].

A desirable property in the design of an economically viable mechanism is *incentive-compatibility*. In particular, a buyer is more likely to make decisions about their allocations based on their own personal incentives rather than to accept a given allocation that might op-

[2]It's not known if $1-1/e$ is tight for MPS.

[3]A function $v\colon 2^M \to \mathcal{R}$ is an XOS function if there exists a collection of additive functions $A_1, \ldots, A_k$ such that for every $S \subseteq M$ we have $v(S) = \max_{1 \leq i \leq k} A_i(S)$.

timize the social welfare but not the individuals' profit. For the important case of unit-demand buyers (aka bipartite matching), in §11.3.1 we extend Theorem 11.1.2 to additionally obtain this property.

Theorem 11.1.3. *For bipartite matchings, when buyers arrive in a uniformly random order, there exists an incentive-compatible mechanism based on dynamic prices that gives a* $(1 - 1/e)$*-approximation to the optimal welfare.*

For this result, we require unit-demand buyers. This is because for general XOS functions shifting buyers to earlier arrivals can change the availability of items arbitrarily. For unit-demand functions, we show that this effect is bounded.

Finally, in §11.5 we conclude by showing that for the single-item case one can obtain a $(1 - 1/e)$-approximation even by using static prices, and that nothing better is possible.

11.1.1 Our Techniques

In this section we discuss our three main ideas for a combinatorial auction. In this setting, our algorithm is threshold based, which means that we set *dynamic prices* to the items and allow a buyer to purchase a set of items only if her value is more than the price of that set. This allows us to view total value as the sum of *utility* of the buyers and the total generated *revenue*. Although powerful, dynamic prices often lead to involved calculations and become difficult to analyze beyond a single item setting [EHLM17, AEE+17]. To overcome this issue, we convert our discrete problem into a continuous setting. This is possible because a random permutation of buyers can be viewed as each buyer arriving at a time chosen uniformly at random between 0 and 1. The benefit of such a transformation is that the arrival times are independent, which keeps correlations managable. Besides, it allow us to use tools from integral calculus such as integration by parts.

Our algorithm for combinatorial auctions sets a *base price* b_j for every item j based on its contribution to the expected offline optimum $\mathbb{E}[\text{OPT}]$. Our approach is to define two time varying continuous functions: *discount* and *residual*. The discount function $\alpha(t) \colon [0,1] \to [0,1]$ is chosen such that the price of an unsold item j at time t is exactly $\alpha(t) \cdot b_j$. We define a *residual* function $r(t) \colon [0,1] \to \mathbb{R}_{\geq 0}$ that intuitively denotes the expected value remaining in the instance at time t. Hence, $r(0) = \mathbb{E}[\text{OPT}]$ and $r(1) = 0$. Computing $r(t)$ is difficult for a combinatorial auction since it depends on several random variables. However, assuming that we know $r(t)$, we use application specific techniques to compute lower bounds on both the expected revenue and the expected utility in terms of the functions $r(t)$ and $\alpha(t)$.

Finally, although we do not know $r(t)$, we can choose the function $\alpha(t)$ in a way that allows us to simplify the sum of expected revenue and utility, without ever computing $r(t)$ explicitly. This step exploits properties of the exponential function for integration (see Lemma 11.2.3 and Figure 11.1).

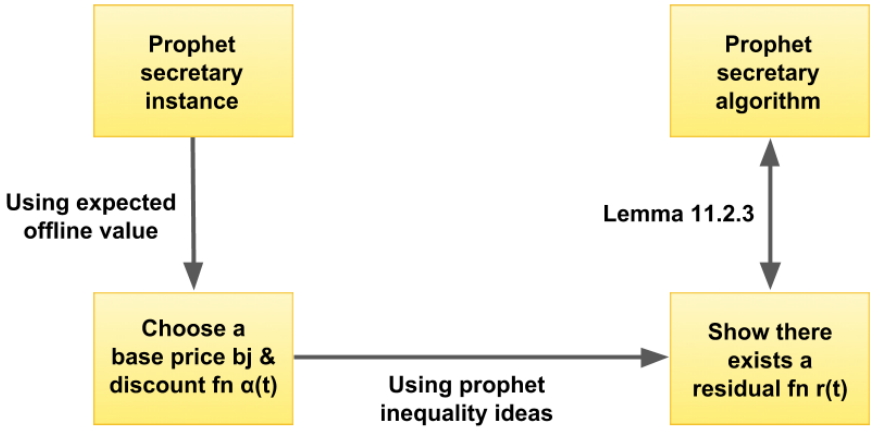

Figure 11.1: To obtain a prophet secretary algorithm for a given combinatorial auction instance, first choose a base price b_j for every item j based on its contribution to the expected offline optimum. Next, using ideas from prophet inequalities show there exists a residual function $r(t)$. Finally, Lemma 11.2.3 implies a good performance guarantee on the constructed algorithm.

11.1.2 Organization of the Chapter

In §11.2 we formally define a residual function. To give some intuition, we give an alternate proof of single item $1 - 1/e$ prophet secretary using this approach. In §11.3 we first extend this approach to bipartite matching and then to combinatorial auctions. In §11.4 we combine ideas from [KW12] with our residual approach to prove Theorem 11.1.1 for matroids. Finally in §11.5 we show the power and limitations of fixed threshold algorithms.

11.2 Our Approach using a Residual

In this section, we define a residual and discuss how it can be used to design an approximation algorithm for a prophet secretary problem. Suppose there are n requests that arrive at times $(T_i)_{i\in[n]}$ drawn i.i.d. from the uniform distribution in $[0,1]$. These requests correspond to buyers of a combinatorial auction or to elements of a matroid.

Whenever a request arrives, we have to decide if and how to serve it. Depending on how we serve request i, say x_i, we gain a certain value $v_i(x_i)$. Our task is to maximize the sum of values over all requests $\sum_{i=1}^{n} v_i(x_i)$. Our algorithm Alg includes a time-dependent payment component. The payment that request i has to make is the product of a time-dependent *discount* function $\alpha(t)$ and a *base price* $b(x_i)$. The base price depends on the allocation up to this point and how much the new choice limits other allocations in the future. However, it does not depend on t, the time that has passed up to this point. If request i has to pay $p_i(x_i, T_i) = \alpha(T_i, b(x_i)$ for our decision x_i, then its *utility* is given by $u_i = v_i(x_i) - p_i(x_i, T_i)$. We write Utility $= \sum_{i=1}^{n} u_i$ for the sum of utilities and Revenue $= \sum_{i=1}^{n} p_i(x_i, T_i)$ for the sum of payments. The value achieved by Alg equals Utility + Revenue.

Next we define a residual function that has the interpretation of "expected remaining value in the instance at time t". In Lemma 11.2.3 we show that the existence of a residual function for Alg suffices to give a $(1-1/e)$-approximation prophet secretary.

Definition 11.2.1 (Residual). *Consider a prophet secretary problem with expected offline value $\mathbb{E}[OPT]$. For any algorithm Alg based on a differentiable* discount *function $\alpha(t)\colon [0,1] \to [0,1]$, a differentiable function $r(t)\colon [0,1] \to \mathbb{R}_{\geq 0}$ is called a* residual *if it satisfies the following three conditions for every choice of α.*

$$r(0) = \mathbb{E}[OPT] \tag{11.1a}$$

$$\mathbb{E}[\text{Revenue}] \geq -\int_{t=0}^{1} \alpha(t) \cdot r'(t) \cdot dt \tag{11.1b}$$

$$\mathbb{E}[\text{Utility}] \geq \int_{t=0}^{1} (1-\alpha(t)) \cdot r(t) \cdot dt. \tag{11.1c}$$

We would like to remark here that this definition is similar in spirit to balanced thresholds [KW12] and balanced prices [DFKL17]. However, it is different because we have to take into account the random arrivals.

As an illustration of Definition 11.2.1, consider the case of a single item. That is, we are presented a sequence of n real numbers and may select only up to one of them (previously studied in [EHLM17]).

Example 11.2.2 (Single Item). *Suppose buyer $i \in [n]$ arrives with random value v_i at time T_i chosen uniformly at random between 0 and 1. Define $b = \mathbb{E}[\max_i v_i]$ as the* base price *of the single item. A buyer arriving at time t is offered the item at price $\alpha(t) \cdot b$, and she accepts the offer if and only if $v_i \geq \alpha(t) \cdot b$. We show that $r(t) = \Pr[\text{item not sold before } t] \cdot b$ is a* residual *function.*

By definition, (11.1a) holds trivially. To see that (11.1b) holds, observe that the increase in revenue from time t to time $t+\epsilon$ is approximately $\alpha(t) \cdot b$ if the item is allocated during this time, and is 0 otherwise. That is, the expected increase in revenue is approximately $\alpha(t)(r(t) - r(t+\epsilon))$. Taking the limit for $\epsilon \to 0$ then implies (11.1b), i.e., $\mathbb{E}[\text{Revenue}] = -\int_{t=0}^{1} \alpha(t) r'(t) dt$.

For (11.1c), consider the expected utility of a buyer i conditioning on her arriving at time t

$$\begin{aligned}\mathbb{E}[u_i \mid T_i = t] &= \mathbb{E}[\mathbf{1}_{\text{item not sold before } t} \cdot (v_i - \alpha(t) \cdot b)^+ \mid T_i = t] \\ &= \Pr[\text{item not sold before } t \mid T_i = t] \cdot \mathbb{E}[(v_i - \alpha(t) \cdot b)^+].\end{aligned}$$

Here we use that the event the item is sold before t does not depend on v_i because buyer i only arrives at time t. The expectation in turn only depends on v_i. It is also important to observe that $\Pr[\text{item not sold before } t \mid T_i = t] \geq \Pr[\text{item not sold before } t]$. Next, we take the sum over all buyers i and use that $\mathbb{E}[\sum_{i=1}^{n} (v_i - \alpha(t) \cdot b)^+] \geq \mathbb{E}[\max_i (v_i - \alpha(t) \cdot b)] = \mathbb{E}[\max_i v_i] - \alpha(t) \cdot b = (1-\alpha(t)) \cdot b$ to get

$$\sum_{i=1}^{n} \mathbb{E}[u_i \mid T_i = t] \quad \geq \quad \Pr[\text{item not sold before } t] \cdot (1-\alpha(t)) \cdot b \quad = \quad (1-\alpha(t)) \cdot r(t).$$

This implies

$$\mathbb{E}[\text{Utility}] = \sum_{i=1}^{n} \int_{t=0}^{1} \mathbb{E}[u_i \mid T_i = t]dt = \int_{t=0}^{1} \sum_{i=1}^{n} \mathbb{E}[u_i \mid T_i = t]dt \geq \int_{t=0}^{1} (1 - \alpha(t)) \cdot r(t) \cdot dt.$$

We now use the properties of a residual function to design a $(1 - 1/e)$-approximation algorithm. To this end, we choose $\alpha(t)$ in a manner that makes the sum of the expected revenue and buyers' utilities independent of $r(t)$. This allows us to compute expected welfare, even though we cannot compute $r(t)$ directly.

Lemma 11.2.3. *For a prophet secretary problem, if there exists a residual function $r(t)$ for algorithm Alg as defined in Definition 11.2.1, then setting $\alpha(t) = 1 - e^{t-1}$ gives a $(1 - 1/e)$-approximation.*

Proof. To further simplify (11.1b), we observe that applying integration by parts gives

$$\int r'(t) \cdot \alpha(t) \cdot dt = r(t) \cdot \alpha(t) - \int r(t)\alpha'(t) \cdot dt.$$

So in combination

$$\mathbb{E}[\text{Revenue}] \geq -\Big([r(t) \cdot \alpha(t)]_{t=0}^{1} - \int_{t=0}^{1} r(t) \cdot \alpha'(t) \cdot dt \Big). \tag{11.2}$$

Now adding (11.2) and (11.1c) gives,

$$\begin{aligned}
\mathbb{E}[\text{Alg}] &= \mathbb{E}[\text{Utility}] + \mathbb{E}[\text{Revenue}] \\
&\geq \int_{t=0}^{1} r(t) \cdot (1 - \alpha(t)) \cdot dt - [r(t)\alpha(t)]_{t=0}^{1} + \int_{t=0}^{1} r(t)\alpha'(t) \cdot dt \\
&= \int_{t=0}^{1} r(t) \cdot (1 - \alpha(t) + \alpha'(t)) \cdot dt - [r(t)\alpha(t)]_{t=0}^{1} \quad .
\end{aligned}$$

Although we do not know $r(t)$ and computing $\int_{t=0}^{1} r(t) \cdot (1 - \alpha(t) + \alpha'(t)) \cdot dt$ seems difficult, we have the liberty of selecting the function $\alpha(t)$. By choosing $\alpha(t)$ satisfying $1 - \alpha(t) + \alpha'(t) = 0$ for all t, this integral becomes independent of $r(t)$ and simplifies to 0. In particular, let $\alpha(t) = 1 - e^{t-1}$. This gives,

$$\begin{aligned}
\mathbb{E}[Alg] &\geq -[r(t) \cdot \alpha(t)]_{t=0}^{1} \\
&= \left(1 - \frac{1}{e}\right) r(0) \\
&= \left(1 - \frac{1}{e}\right) \mathbb{E}[\text{OPT}] \quad .
\end{aligned}$$

□

11.3 Prophet Secretary for Combinatorial Auctions

Let N denote a set of n buyers and M denote the set of m indivisible items. Suppose buyer i arrives at a time T_i chosen uniformly at random between 0 and 1. Let $v_i : 2^M \to \mathbb{R}_{\geq 0}$ (similarly $\hat{v}_i$) denote the random combinatorial valuation function of buyer i. In order to ensure polynomial running times, we assume that the distribution of v_i has a polynomial support $\{v_i^1, v_i^2, \dots, \}$, where $\sum_k \Pr[v_i = v_i^k] = 1$. Note that this assumption only simplifies notation. If we only have sample access to the distributions, then we can replace $\{v_i^1, v_i^2, \dots, \}$ by an appropriate number of samples. Within our proofs, we will use $\hat{\mathbf{v}}$ to denote an independent, fresh sample from the distribution.

By $\mathbf{T}$ and $\mathbf{v}$ (similarly $\hat{\mathbf{v}}$) we denote the vector of all the buyer arrival times and valuations, respectively. Also, let $\mathbf{v}_{-i}$ (similarly $\hat{\mathbf{v}}_{-i}$) denote valuations of all buyers except buyer i. For the special case of single items, we let v_{ij} denote $v_i(\{j\})$. Let $q_j(t)$ denote the probability that item j has not been sold before time t, where the probability is over valuations $\mathbf{v}$, arrival times $\mathbf{T}$, and any randomness of the algorithm.

11.3.1 Bipartite Matching

In the bipartite matching setting all buyers are unit-demand, i.e. $v_i(S) = \max_{j \in S} v_{ij}$. We can therefore assume that no buyer buys more than one item. See Figure 11.2 for an example. We restate our result.

Theorem 11.1.3. *For bipartite matchings, when buyers arrive in a uniformly random order, there exists an incentive-compatible mechanism based on dynamic prices that gives a $(1 - 1/e)$-approximation to the optimal welfare.*

To define prices of items, let *base price* b_j denote the expected value of the buyer that buys item j in the offline welfare maximizing allocation (maximum weight matching). Now consider an algorithm that prices item j at $\alpha(t) \cdot b_j$ at time t and allows the incoming buyer to pick any of the unsold items; here $\alpha(t)$ is a continuous differentiable discount function.

Consider the function $r(t) = \sum_j q_j(t) \cdot b_j$. Clearly, $r(0) = \mathbb{E}[\text{OPT}]$. Using the following Lemma 11.3.1 and Claim 11.3.2, we prove that r is a residual function for our algorithm. Since the algorithm is clearly incentive-compatible, Lemma 11.2.3 implies Theorem 11.1.3.

Lemma 11.3.1. *We can lower bound the total expected utility by*

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Utility}] \geq \sum_j \int_{t=0}^{1} q_j(t) \cdot (1 - \alpha(t)) \cdot b_j \cdot dt. \tag{11.3}$$

Proof. Since buyer i arriving at time t can pick any of the unsold items, we have

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] = \mathbb{E}_{\mathbf{v}}\left[\max_j \mathbf{1}_{j \text{ not sold before } t} \cdot \left(v_{i,j} - \alpha(t) \cdot b_j\right)^+ \,\middle|\, T_i = t\right].$$

One particular choice of buyer i is to choose item $OPT_i(v_i, \hat{\mathbf{v}}_{-i})$ if it is still available, and no item

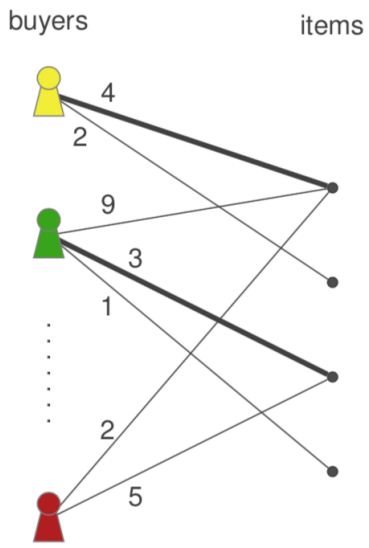

Figure 11.2: An example of a bipartite matching instance where edge numbers v_{ij} indicate value of buyer i for item j. A solid line means the buyer bought that item.

otherwise. This gives us a lower bound of

$$\begin{aligned}\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] &\geq \mathbb{E}_{\mathbf{v}}\left[\mathbf{1}_{OPT_i(v_i,\hat{\mathbf{v}}_{-i}) \text{ not sold before } t} \cdot \left(v_{i,OPT_i(v_i,\hat{\mathbf{v}}_{-i})} - \alpha(t)\cdot b_j\right)^+ \,\middle|\, T_i = t\right] \\ &= \sum_j \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}}}\left[\mathbf{1}_{j \text{ not sold before } t} \cdot \mathbf{1}_{j=OPT_i(v_i,\hat{\mathbf{v}}_{-i})} \cdot \left(v_{i,j} - \alpha(t)\cdot b_j\right)^+ \,\middle|\, T_i = t\right].\end{aligned}$$

Note that in the product, the fact whether j is sold before t only depends on $\mathbf{v}_{-i}$ and the arrival times of the other buyers. It does not depend on v_i or $\hat{\mathbf{v}}$. The remaining terms, in contrast, only depend on v_i and $\hat{\mathbf{v}}_{-i}$. Therefore, we can use independence to split up the expectation and get

$$\begin{aligned}&\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] \\ &\geq \sum_j \Pr[j \text{ not sold before } t \mid T_i = t] \cdot \mathbb{E}_{v_i,\hat{\mathbf{v}}_{-i}}\left[\mathbf{1}_{j=OPT_i(v_i,\hat{\mathbf{v}}_{-i})} \cdot (v_{i,j} - \alpha(t)\cdot b_j)^+ \,\middle|\, T_i = t\right].\end{aligned}$$

Next, we use that $\Pr[j \text{ not sold before } t \mid T_i = t] \geq q_j(t)$ by Lemma 11.3.3 and that v_i and $\hat{v}_i$ are identically distributed. Therefore, we can swap their roles inside the expectation. Overall, this gives us

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] \geq \sum_j q_j(t) \cdot \mathbb{E}_{\hat{\mathbf{v}}}\left[\mathbf{1}_{j=OPT_i(\hat{\mathbf{v}})} \cdot (\hat{v}_{i,j} - \alpha(t)\cdot b_j)\right]. \tag{11.4}$$

Next, observe that $\mathbb{E}_{\hat{\mathbf{v}}}[\sum_i \mathbf{1}_{j=OPT_i(\hat{\mathbf{v}})} \cdot \hat{v}_{i,j}] = b_j$ by the definition of b_j. Therefore, using linearity of expectation, summing up (11.4) over all buyers i gives us

$$\sum_i \mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] \geq q_j(t) \cdot (1 - \alpha(t)) \cdot b_j.$$

Now, taking the expectation over t, we get

$$\begin{aligned}
\mathbb{E}_{\mathbf{v},\mathbf{T}}\Big[\sum_i u_i\Big] &= \sum_i \int_{t=0}^{1} \mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] \cdot dt \\
&= \int_{t=0}^{1} \sum_i \mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] \cdot dt \\
&\geq \int_{t=0}^{1} \sum_j q_j(t) \cdot (1-\alpha(t)) \cdot b_j \cdot dt \\
&= \sum_j \int_{t=0}^{1} q_j(t) \cdot (1-\alpha(t)) \cdot b_j \cdot dt \quad . \qquad \square
\end{aligned}$$

We next give a bound on the revenue generated by our algorithm.

Claim 11.3.2. *We can bound the total expected revenue by*

$$\mathbb{E}_{\boldsymbol{v},\boldsymbol{T}}[\text{Revenue}] = -\sum_j \int_{t=0}^{1} q_j'(t)\alpha(t) \cdot b_j \cdot dt. \tag{11.5}$$

Proof. Since $-q_j'(t)dt$ is the probability that item j is bought between t and $t+dt$ (note $q_j(t)$ is decreasing in t), we have

$$\mathbb{E}[\text{Revenue}] = -\sum_j \int_{t=0}^{1} q_j'(t)\alpha(t) \cdot b_j \cdot dt \quad . \qquad \square$$

Finally, we prove the missing lemma that removes the conditioning on the arrival time.

Lemma 11.3.3. *We have*

$$\mathbb{E}_{\boldsymbol{v}_{-i}}\Big[\Pr_{\boldsymbol{T}}[j \text{ not sold before } t \mid T_i = t]\Big] \geq q_j(t).$$

The idea is that if buyers arrive earlier in the process, this only *reduces* the available items. It can never happen that such a change makes an item available at a later point. For a single item this is trivial, for multiple items and other combinatorial valuations it does not necessarily hold.

Proof. Consider the execution of our algorithm on two sequences that only differ in the arrival time of buyer i. To this end, let $\mathbf{v}$ be arbitrary values and $\mathbf{T}$ be arbitrary arrival times. Let $A_{t'}$ be the set of items that are sold before time t' on the sequence defined by $\mathbf{v}$ and $\mathbf{T}$. Furthermore, let $B_{t'}$ be the set of items sold before time t' if we replace T_i by t. Ties are broken in the same way in both sequences.

We claim that $B_{t'} \subseteq A_{t'}$ for all $t' \leq t$.

To this end, we observe that by definition $B_{t'} = A_{t'}$ for $t' \leq \min\{T_i, t\}$ because the two sequences are identical before $\min\{T_i, t\}$. This already shows the claim for $T_i \geq t$. Otherwise,

assume that there is some $t' \leq t$ for which $B_{t'} \not\subseteq A_{t'}$. Let t_{inf} be the infimum among these t'. It has to hold that some buyer i' arrives at time t_{inf} and buys item $j_A \notin A_{t_{\text{inf}}}$ in the original sequence and $j_B \notin B_{t_{\text{inf}}}$ in the modified sequence. Furthermore, we now have to have $B_{t_{\text{inf}}} \not\subseteq A_{t_{\text{inf}}}$ because t_{inf} was defined to be the infimum of all t' for which $B_{t'} \subseteq A_{t'}$ is not fulfilled. Therefore, $j_A \notin B_{t_{\text{inf}}}$. Additionally, $j_B \notin A_{t_{\text{inf}}}$. The reason is that for any $t' < t_{\text{inf}}$ before the next arrival $B_{t'} = B_{t_{\text{inf}}} \cup \{j_B\}$.

Overall this means that in both sequences at time t_{inf} buyer i' has the choice between j_B and j_A. As his values are identical and ties are broken the same way, it has to hold that $j_B = j_A$, which then contradicts that $B_{t_{\text{inf}}} \not\subseteq A_{t_{\text{inf}}}$.

Taking the expectation over both $\mathbf{v}$ and $\mathbf{T}$, we get

$$\Pr_{\mathbf{T},\mathbf{v}}[j \notin A_t] \leq \Pr_{\mathbf{T},\mathbf{v}}[j \notin B_t].$$

This implies the Lemma 11.3.3 because

$$\Pr_{\mathbf{T},\mathbf{v}}[j \text{ not sold before } t] = \Pr_{\mathbf{T},\mathbf{v}}[j \notin A_t]$$

$$\Pr_{\mathbf{T},\mathbf{v}}[j \text{ not sold before } t \mid T_i = t] = \Pr_{\mathbf{T},\mathbf{v}}[j \notin B_t]. \qquad \square$$

11.3.2 XOS Combinatorial Auctions

In this section we prove our main result (restated below) for combinatorial auctions.

Theorem 11.1.2. *There exists a $(1 - 1/e)$-approximation algorithm to CAPS.*

Recollect that the random valuation v_i of every buyer i has a polynomial support. We can therefore write the following expectation-version of the configuration LP, which gives us an upper bound on the expected offline social welfare.

$$\begin{aligned}
\max \sum_i \sum_k \sum_S v_i^k(S) \cdot x_{i,S}^k & & & \\
\text{s.t.} \sum_i \sum_k \sum_{S: j \in S} x_{i,S}^k &= 1 & & \text{for all } j \in M \\
\sum_S x_{i,S}^k &= \Pr[v_i = v_i^k] & & \text{for all } i, k
\end{aligned}$$

The above configuration LP can be solved with a polynomial number of calls to demand oracles of buyer valuations (see [DNS10]). Since all functions v_i^k are XOS, there exist additive supporting valuations; that is, there exist numbers $v_{i,j}^{k,S} \geq 0$ s.t. $v_{i,j}^{k,S} = 0$ for $j \notin S$, $\sum_{j \in S} v_{i,j}^{k,S} = v_i^k(S)$, and $\sum_{j \in S'} v_{i,j}^{k,S} \leq v_i^k(S')$ for all S'. Before describing our algorithm, we define a *base price* for every item.

Definition 11.3.4. *The* base price b_j *of every item $j \in M$ is* $\sum_{i,k} \sum_{S: j \in S} v_{i,j}^{k,S} x_{i,S}^k$.

Since $\sum_S x^k_{i,S} = \Pr[v_i = v^k_i]$, consider an algorithm that on arrival of buyer i with valuation v^k_i draws an independent random set S with probability $x^k_{i,S}/\Pr[v_i = v^k_i]$. Let S^*_i denote this drawn set. This distribution also satisfies that for every item j,

$$\sum_i \mathbb{E}_{v_i,S^*_i}\left[\mathbf{1}_{j\in S^*_i}\cdot v^{k,S^*_i}_{i,j}\right] \quad = \quad \sum_{i,k}\Pr[v_i = v^k_i]\cdot\sum_{S:j\in S}\frac{x^k_{i,S}}{\Pr[v_i = v^k_i]}\cdot v^{k,S^*_i}_{i,j} \quad = \quad b_j. \tag{11.6}$$

Now consider the supporting additive valuation for S^*_i in the XOS valuation function v^k_i of buyer i. This can be found using the XOS oracle for v^k_i [DNS10]. Our algorithm assigns her every item j that has not been allocated so far and for which $v^{k,S^*_i}_{i,j} \geq \alpha(t)\cdot b_j$, where $\alpha(t)$ is a continuous differentiable function of t. Note that since we do not allow buyer i to choose items outside set S^*_i, the mechanism defined by this algorithm need not be incentive compatible.

Consider the function $r(t) = \sum_j q_j(t)\cdot b_j$, where again $q_j(t)$ denotes the probability that item j has not been sold before time t. Clearly, $r(0) = OPT$. Using the following Lemma 11.3.5 and Claim 11.3.6, we prove that r is a residual function for our algorithm. Hence, Lemma 11.2.3 implies Theorem 11.1.2.

Lemma 11.3.5. *The expected utility of the above algorithm is lower bounded by*

$$\mathbb{E}_{\boldsymbol{v},\boldsymbol{T}}[\mathsf{Utility}] \geq \sum_j \int_{t=0}^{1} q_j(t)\cdot(1-\alpha(t))\cdot b_j\cdot dt. \tag{11.7}$$

Proof. Given that buyer i arrives at t and only buys item j if $v^{k,S^*_i}_{i,j} \geq \alpha(t)\cdot b_j$, her utility is

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] = \sum_j \mathbb{E}_{\mathbf{v},\mathbf{T},S^*_i}\left[\mathbf{1}_{j \text{ not sold by } t}\cdot\mathbf{1}_{j\in S^*_i}\cdot\left(v^{k,S^*_i}_{i,j} - \alpha(t)\cdot b_j\right)^+ \,\middle|\, T_i = t\right]$$

Using the fact that whether j is sold before t only depends on $\mathbf{v}_{-i}$ and $\mathbf{T}$, and not on v_i or S^*_i,

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] = \sum_j \Pr_{\mathbf{v}_{-i},\mathbf{T}}[j \text{ not sold by } t \mid T_i = t]\cdot\mathbb{E}_{v_i,S^*_i}\left[\mathbf{1}_{j\in S^*_i}\cdot\left(v^{k,S^*_i}_{i,j} - \alpha(t)\cdot b_j\right)^+\right].$$

Now, observe that in our algorithm every buyer i independently decides which set of items S^*_i it will attempt to buy. Crucially, the probability of an item j being sold by time t can only increase if more buyers arrive before t. Therefore,

$$\Pr_{\mathbf{v}_{-i},\mathbf{T}}[j \text{ not sold by } t \mid T_i = t] \quad \geq \quad \Pr_{\mathbf{v},\mathbf{T}}[j \text{ not sold by } t] \quad = \quad q_j(t).$$

Thus, we get

$$\begin{aligned}\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] &\geq \sum_j q_j(t)\cdot\mathbb{E}_{v_i,S^*_i}\left[\mathbf{1}_{j\in S^*_i}\cdot\left(v^{k,S^*_i}_{i,j} - \alpha(t)\cdot b_j\right)^+\right]\\ &\geq \sum_j q_j(t)\cdot\mathbb{E}_{v_i,S^*_i}\left[\mathbf{1}_{j\in S^*_i}\cdot\left(v^{k,S^*_i}_{i,j} - \alpha(t)\cdot b_j\right)\right].\end{aligned}$$

Finally, recollect from (11.6) that $\sum_i \mathbb{E}_{v_i,S_i^*}\left[\mathbf{1}_{j\in S_i^*}\cdot v_{i,j}^{k,S_i^*}\right] = b_j$. Moreover,

$$\sum_i \mathbb{E}_{v_i,S_i^*}\left[\mathbf{1}_{j\in S_i^*}\right] \quad = \quad \sum_{i,k}\Pr[v_i = v_i^k]\cdot\sum_{S:j\in S}\frac{x_{i,S}^k}{\Pr[v_i=v_i^k]} \quad = \quad 1.$$

Hence, by linearity of expectation

$$\sum_i \mathbb{E}[u_i \mid T_i = t] \geq \sum_j q_j(t)\cdot(1-\alpha(t))\cdot b_j.$$

□

We next give a bound on the revenue generated by our algorithm.

Claim 11.3.6. *We can bound the total expected revenue by*

$$\mathbb{E}_{\boldsymbol{v},\boldsymbol{T}}[\text{Revenue}] = -\sum_j \int_{t=0}^1 q_j'(t)\alpha(t)\cdot b_j\cdot dt. \tag{11.8}$$

Proof. Since $-q_j'(t)dt$ is the probability that item j is bought between t and $t+dt$ (note $q_j(t)$ is decreasing in t), we have

$$\mathbb{E}[\text{Revenue}] = -\sum_j \int_{t=0}^1 q_j'(t)\alpha(t)\cdot b_j\cdot dt \quad .$$

□

11.4 Prophet Secretary for Matroids

Let v_i denote the random value of the i'th buyer (element) and let $\hat{v}_i$ denote another independent draw from the value distribution of the i'th buyer. The problem is to select a subset I of the buyers that form a feasible set in matroid $\mathcal{M}$, while trying to maximize $\sum_{i\in I} v_i$. We restate our main result for the matroid setting.

Theorem 11.1.1. *There exists a $(1-1/e)$-approximation algorithm to MPS.*

We need the following notation to describe our algorithm.

Definition 11.4.1. *For a given vector $\hat{\boldsymbol{v}}$ of values of n items and $A\subseteq[n]$, we define the following:*

- *Let $Opt(\hat{\boldsymbol{v}}\mid A)\subseteq[n]\setminus A$ denote the optimal solution set in the contracted matroid $\mathcal{M}/A$.*
- *Let $R(A,\hat{\boldsymbol{v}}) := \sum_{i\in Opt(\hat{\boldsymbol{v}}|A)}\hat{\boldsymbol{v}}_i$ denote the remaining value after selecting set A.*

We next define a *base price* of for every buyer i.

Definition 11.4.2. *Let A denote the independent set of buyers that have been accepted till now.*

- *Let $b_i(A,\hat{\boldsymbol{v}}) := R(A,\hat{\boldsymbol{v}}) - R(A\cup\{i\},\hat{\boldsymbol{v}})$ denote a threshold for buyer i.*

- *Let* $b_i(A) := \mathbb{E}_{\hat{\mathbf{v}}}[b_i(A, \hat{\mathbf{v}})]$ *denote the* base price *for buyer* i.

Starting with $A_0 = \emptyset$, let A_t denote the set of accepted buyers *before* time t. This is a random variable that depends on the values $\mathbf{v}$ and arrival times $\mathbf{T}$. Suppose a buyer i arrives at time t, then our algorithm selects i iff both $v_i > \alpha(t) \cdot b_i(A_t)$ and selecting i is feasible in $\mathcal{M}$.

Consider the function $r(t) := \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}},\mathbf{T}}[R(A_t, \hat{\mathbf{v}})]$, where A_t is a function of $\mathbf{v}$ and $\mathbf{T}$. Clearly, $r(0) = \mathbb{E}[\text{OPT}]$. Using the following Lemma 11.4.4 and Claim 11.4.3, we prove that r is a residual function. Hence, Lemma 11.2.3 implies Theorem 11.1.1.

Claim 11.4.3.

$$\mathbb{E}_{\boldsymbol{v},\boldsymbol{T}}[\text{Revenue}] = -\int_{t=0}^{1} \alpha(t) \cdot r'(t) dt.$$

Proof. Consider the time from t to $t + \epsilon$ for some $t \in [0, 1]$, $\epsilon > 0$. Let us fix the arrival times $\mathbf{T}$ and values $\mathbf{v}$ of all elements. This also fixes the sets $(A_t)_{t\in[0,1]}$. Let $i_1, \ldots, i_k$ be the arrivals between t and $t + \epsilon$ that get accepted in this order. Note that it is also possible that $k = 0$. The revenue obtained between t and $t + \epsilon$ is now given as

$$\begin{aligned}
\text{Revenue}_{\leq t+\epsilon} - \text{Revenue}_{\leq t} &= \sum_{j=1}^{k} \alpha(t_{i_j}) b_{i_j}(A_{t_{i_j}}) \\
&= \sum_{j=1}^{k} \alpha(t_{i_j}) \mathbb{E}_{\hat{\mathbf{v}}} \left[R(A_t \cup \{i_1, \ldots i_{j-1}\}, \hat{\mathbf{v}}) - R(A_t \cup \{i_1, \ldots i_j\}, \hat{\mathbf{v}}) \right] \\
&\geq \alpha(t + \epsilon) \mathbb{E}_{\hat{\mathbf{v}}} \left[R(A_t, \hat{\mathbf{v}}) - R(A_{t+\epsilon}, \hat{\mathbf{v}}) \right].
\end{aligned}$$

Taking the expectation over $\mathbf{v}$ and $\mathbf{T}$, we get by linearity of expectation

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Revenue}_{\leq t+\epsilon}] - \mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Revenue}_{\leq t}] \geq \alpha(t + \epsilon)(r(t) - r(t + \epsilon)).$$

By the same argument, we also have

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Revenue}_{\leq t+\epsilon}] - \mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Revenue}_{\leq t}] \leq \alpha(t)(r(t) - r(t + \epsilon)).$$

In combination, we get that

$$\frac{d}{dt} \mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Revenue}_{\leq t}] = -\alpha(t) r'(t),$$

which implies the claim. □

Lemma 11.4.4.

$$\mathbb{E}_{\boldsymbol{v},\boldsymbol{T}}[\text{Utility}] \geq \int_{t=0}^{1} (1 - \alpha(t)) \cdot r(t) dt.$$

Proof. The utility of buyer i arriving at time t is given by

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] = \mathbb{E}_{\mathbf{v},\mathbf{T}_{-i}} \left[(v_i - \alpha(t) \cdot b_i(A_t))^+ \cdot \mathbf{1}_{i \notin Span(A_t)} \,\middle|\, T_i = t \right].$$

Observe that A_t does not depend on v_i if $T_i = t$ because it includes only the acceptances *before* t. It does not depend on $\hat{v}_i$ either, as $\hat{v}_i$ is only used for analysis purposes and not known to the algorithm. Since v_i and $\hat{v}_i$ are identically distributed, we can also write

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] = \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}},\mathbf{T}_{-i}}\left[(\hat{v}_i - \alpha(t)\cdot b_i(A_t))^+ \cdot \mathbf{1}_{i\notin Span(A_t)} \;\middle|\; T_i = t\right]. \tag{11.9}$$

Now observe that buyer i can belong to $Opt(\hat{\mathbf{v}} \mid A_t)$ only if it's not already in $Span(A_t)$, which implies $\mathbf{1}_{i\notin Span(A_t)} \geq \mathbf{1}_{i\in Opt(\hat{\mathbf{v}}|A_t)}$. Using this and removing non-negativity, we get

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] \geq \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}},\mathbf{T}_{-i}}\left[(\hat{v}_i - \alpha(t)\cdot b_i(A_t))\cdot \mathbf{1}_{i\in Opt(\hat{\mathbf{v}}|A_t)} \;\middle|\; T_i = t\right].$$

Now we use Lemma 11.4.5 to remove the conditioning on buyer i arriving at time t as this gives a valid lower bound on expected utility,

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t] \geq \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}},\mathbf{T}}\left[(\hat{v}_i - \alpha(t)\cdot b_i(A_t))\cdot \mathbf{1}_{i\in Opt(\hat{\mathbf{v}}|A_t)}\right]. \tag{11.10}$$

We can now lower bound sum of buyers' utilities using (11.10) to get

$$\begin{aligned}\mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Utility}] &= \sum_i \int_{t=0}^{1} \mathbb{E}_{\mathbf{v},\mathbf{T}}[u_i \mid T_i = t]\cdot dt \\ &\geq \sum_i \int_{t=0}^{1} \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}},\mathbf{T}}\left[(\hat{v}_i - \alpha(t)\cdot b_i(A_t))\cdot \mathbf{1}_{i\in Opt(\hat{\mathbf{v}}|A_t)}\right]\cdot dt.\end{aligned}$$

By moving the sum over buyers inside the integrals, we get

$$\begin{aligned}\mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Utility}] &\geq \int_{t=0}^{1} \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}},\mathbf{T}}\left[\sum_i (\hat{v}_i - \alpha(t)\cdot b_i(A_t))\cdot \mathbf{1}_{i\in Opt(\hat{\mathbf{v}}|A_t)}\right]\cdot dt \\ &= \int_{t=0}^{1} \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}},\mathbf{T}}\left[R(A_t,\hat{\mathbf{v}}) - \alpha(t)\cdot \sum_{i\in Opt(\hat{\mathbf{v}}|A_t)} b_i(A_t)\right]\cdot dt.\end{aligned}$$

Finally, using Lemma 11.4.6 for $V = Opt(\hat{\mathbf{v}} \mid A_t)$, we get

$$\mathbb{E}_{\mathbf{v},\mathbf{T}}[\text{Utility}] \geq \int_{t=0}^{1} \mathbb{E}_{\mathbf{v},\hat{\mathbf{v}},\mathbf{T}}\left[(1-\alpha(t))\cdot R(A_t,\hat{\mathbf{v}})\right]\cdot dt.$$

□

Finally, we prove the missing lemma that removes the conditioning on item i arriving at t.

Lemma 11.4.5. *For any i, any time t, and any fixed $\mathbf{v}, \hat{\mathbf{v}}$, we have*

$$\mathbb{E}_{\mathbf{T}_{-i}}\left[(\hat{v}_i - \alpha(t)\cdot b_i(A_t))\cdot \mathbf{1}_{i\in Opt(\hat{\mathbf{v}}|A_t)} \mid T_i = t\right] \;\geq\; \mathbb{E}_{\mathbf{T}}\left[(\hat{v}_i - \alpha(t)\cdot b_i(A_t))\cdot \mathbf{1}_{i\in Opt(\hat{\mathbf{v}}|A_t)}\right].$$

Proof. We prove the lemma for any fixed $\mathbf{T}_{-i}$. Suppose we draw a uniformly random $T_i \in [0,1]$. Observe that if $T_i \geq t$ then we have equality in the above equation because set A_t is the same both with and without i. This is also the case when $T_i < t$ but i is not selected into A_t. Finally, when $T_i < t$ and $i \in A_t$ we have $\mathbf{1}_{i\in Opt(\hat{\mathbf{v}}|A_t)} = 0$ in the presence of item i (i.e., RHS of lemma), making the inequality trivially true. □

Lemma 11.4.6. *For any fixed $\boldsymbol{v}, \boldsymbol{T}$, time t, and set of elements V that is independent in the matroid $\mathcal{M}/A_t$, we have*

$$\sum_{i \in V} b_i(A_t) \quad \leq \quad \mathbb{E}_{\hat{\boldsymbol{v}}}\left[R(A_t, \hat{\boldsymbol{v}})\right].$$

Proof. By definition

$$\sum_{i \in V} b_i(A_t) = \mathbb{E}_{\hat{\mathbf{v}}}\Big[\sum_{i \in V} \left(R(A_t, \hat{\mathbf{v}}) - R(A_t \cup \{i\}, \hat{\mathbf{v}})\right)\Big].$$

Fix the values $\hat{\mathbf{v}}$ arbitrarily, we also have

$$\sum_{i \in V} \left(R(A_t, \hat{\mathbf{v}}) - R(A_t \cup \{i\}, \hat{\mathbf{v}})\right) \leq R(A_t, \hat{\mathbf{v}}).$$

This follows from the fact that $R(A_t, \hat{\mathbf{v}}) - R(A_t \cup \{i\}, \hat{\mathbf{v}})$ are the respective critical values of the greedy algorithm on $\mathcal{M}/A_t$ with values $\hat{\mathbf{v}}$. Therefore, the bound follows from Lemma 3.2 in [LB10]. An alternative proof is given as Proposition 2 in [KW12] while in our case the first inequality can be skipped and the remaining steps can be followed replacing A by A_t.

Taking the expectation over $\hat{\mathbf{v}}$, the claim follows. □

11.5 Fixed Threshold Algorithms

In this section we discuss the powers and limitations of *Fixed-Threshold Algorithms* (*FTAs*) for single item prophet secretary. In an FTA, we set a fixed threshold for the item at the beginning of the process and then assign it to the first buyer whose valuation exceeds the threshold. The motivation to study FTAs comes from their simplicity, transparency, and fairness in the design of a posted price mechanism (see, e.g., [FGL15]).

In §11.5.1, we give a $(1 - 1/e)$-approximation FTA for single-item prophet secretary. This seemingly contradicts earlier impossibility results (e.g., [FGL15, EHLM17]). However, as we show, these impossibility results do not hold in case of continuous distributions or equivalently randomized tie-breaking. Next, in §11.5.2, we present an upper bound for FTAs. In particular, we show that there is no FTA, even for identical distributions, with an approximation factor better than $1 - 1/e$. This indicates the tightness of our algorithm for prophet secretary.

11.5.1 Single Item Prophet Secretary

Recall, by $\mathbf{T}$ and $\mathbf{v}$ we denote the random vector of all the buyer arrival times and valuations, respectively. Also, $q(t)$ denotes the probability that the item is unsold till time t, where the probability is over valuations $\mathbf{v}$, arrival times $\mathbf{T}$, and any randomness of the algorithm. We show that a fixed threshold algorithm that selects τ s.t.

$$\Pr_{\mathbf{v}}[\max\{v_i\} \leq \tau] = \prod_i \Pr[v_i \leq \tau] = \frac{1}{e}$$

gives a $(1 - 1/e)$-approximation.

Theorem 11.5.1. *There exists a* $(1-1/e)$*-approximation FTA to single-item prophet secretary.*

Proof. Without loss of generality, assume all distributions have a finite expectation and a continuous CDF[4]. As two extreme selections for the threshold, if we set τ to zero then the FTA selects the first item, and if we set it to infinity then no item will be selected. Therefore, the assumption for the continuity of the distribution function allows us to select a threshold τ such that the FTA reaches the end of the sequence with an exact probability of $1/e$. This means all of drawn values are below τ with probability $1/e$. In the remainder, we show that the FTA based on this choice of τ lead to a $(1-1/e)$-approximation algorithm.

Let OPT denote $\max_i\{v_i\}$ and Alg be a random variable that indicates the value selected by the algorithm, or is zero if no item is selected. The goal is to show

$$\mathbb{E}[\text{Alg}] \geq \left(1-\frac{1}{e}\right)\cdot\mathbb{E}[\text{OPT}].$$

We have $\mathbb{E}[\text{Alg}] = \mathbb{E}[\text{Revenue}] + \mathbb{E}[\text{Utility}]$. By definition of τ, the algorithm sells the item with probability exactly $1-1/e$; therefore, $\mathbb{E}[\text{Revenue}] = \left(1-\frac{1}{e}\right)\tau$. Below, we show

$$\mathbb{E}[\text{Utility}] \geq \left(1-\frac{1}{e}\right)\cdot\mathbb{E}[(\text{OPT}-\tau)^+]. \tag{11.11}$$

This suffices to prove Theorem 11.5.1 because

$$\mathbb{E}[\text{Alg}] = \mathbb{E}[\text{Revenue}] + \mathbb{E}[\text{Utility}] \geq \left(1-\frac{1}{e}\right)\tau + \left(1-\frac{1}{e}\right)\mathbb{E}[(\text{OPT}-\tau)^+] \geq \left(1-\frac{1}{e}\right)\mathbb{E}[\text{OPT}].$$

We now prove (11.11). For the utility, we know

$$\begin{aligned}
\mathbb{E}[\text{Utility}] &= \int_{t=0}^{1}\sum_{i=1}^{n}\mathbb{E}[u_i \mid T_i = t]\cdot dt \\
&= \int_{t=0}^{1}\sum_{i=1}^{n}\Pr[\text{item not sold before t} \mid T_i = t]\cdot\mathbb{E}[(v_i-\tau)^+]\cdot dt \\
&\geq \int_{t=0}^{1}\sum_{i=1}^{n} q(t)\cdot\mathbb{E}[(v_i-\tau)^+]\cdot dt \\
&= \sum_{i=1}^{n}\mathbb{E}[(v_i-\tau)^+]\cdot\int_{t=0}^{1} q(t)\cdot dt,
\end{aligned}$$

where the inequality uses the observation $\Pr[\text{item not sold before t} \mid T_i = t]$ is at least the probability that the item is not sold before t. In the following Lemma 11.5.2, we show $q(t) \geq \exp(-t)$. This implies $\int_{t=0}^{1} q(t)\cdot dt \geq 1-\frac{1}{e}$, which proves the missing (11.11) because

$$\sum_{i=1}^{n}\mathbb{E}[(v_i-\tau)^+] \quad\geq\quad \mathbb{E}[(\max_i\{v_i\}-\tau)^+] \quad=\quad \mathbb{E}[(\text{OPT}-\tau)^+]. \qquad \square$$

[4]This assumption is without loss because the actual CDF can be approximated with arbitrary precision by a continuous function. This approximation corresponds to a randomized tie-breaking in case of point masses.

Lemma 11.5.2. *For $t \in [0,1]$, we have*

$$q(t) \geq \exp(-t).$$

Proof. Observe that

$$q(t) = \prod_{i=1}^{n} \Pr[i \text{ does not buy the item till } t].$$

Since i gets the item only by arriving before t and having a value above τ, we get

$$q(t) = \prod_{i=1}^{n} (1 - t \cdot \Pr[v_i > \tau]) = \exp\left(\sum_i \ln(1 - t \cdot \Pr[v_i > \tau])\right).$$

Notice that for $t, x \in [0,1)$, we have $\ln(1 - tx) \geq t \cdot \ln(1 - x)$. This gives,

$$q(t) \geq \exp\left(t \cdot \sum_i \ln(1 - \Pr[v_i > \tau])\right) = \exp(t \cdot \ln(1/e)) = \exp(-t),$$

where we use $\prod_i (1 - \Pr[v_i > \tau]) = \frac{1}{e}$ by definition of τ. □

11.5.2 Impossibility for IID Prophet Inequalities

In the following we prove an impossibility result for FTAs for single item prophet secretary. We show this impossibility even for the special case of iid items. For every n, we give a common distribution D for every item such that no FTA can achieve an approximation factor better than $1 - 1/e$. This also implies the tightness of the algorithm discussed in §11.5.1.

Theorem 11.5.3. *Any FTA for iid prophet inequality is at most $(1 - \frac{1}{e} + O(\frac{1}{n}))$-approximation*[5].

Proof. We prove the theorem by giving a hard input instance for every n as follows: every v_i is $n/(e-1)$ with probability $1/n^2$ and is $(e-2)/(e-1)$ otherwise. The expected maximum value of these n items is

$$\begin{aligned}\mathbb{E}[\text{OPT}] &= \left(1 - \frac{1}{n^2}\right)^n \cdot \frac{e-2}{e-1} + \left(1 - \left(1 - \frac{1}{n^2}\right)^n\right) \frac{n}{e-1} \\ &= 1 - O\left(\frac{1}{n}\right) \quad .\end{aligned}$$

In this instance, if $\tau < (e-2)/(e-1)$ then the algorithm selects the first item, and if $(e-2)/(e-1) < \tau \leq n/(e-1)$ then the algorithm can only select $n/(e-1)$. In these cases the approximation factor can be at most $(e-2)/(e-1) \approx 0.58$.

Now, note that the CDF of this input distribution is not continuous. Reshaping a discrete distribution function into a continuous one, however, does not change the approximation factor because in the above example we only need a slight change at the point $(e-2)/(e-1)$ of the

[5] Jose Correa later pointed to us that the lower bound in Theorem 11.5.3 also follows from a result in [CFH+17].

CDF. This change gives us a randomness when $\tau = (e-2)/(e-1)$, which is equivalent to flipping a random coin and skipping every item with some probability $p \leq 1 - 1/n^2$ if the drawn value is $(e-2)/(e-1)$. With this assumption we have

$$\begin{aligned}
\mathbb{E}[\mathsf{Alg}] &= \sum_{i=1}^{n} p^i \;\; \mathbb{E}[v_i \cdot \mathbf{1}_{v_i \geq \tau}] \\
&= \frac{1-p^n}{1-p} \;\; \mathbb{E}[v_i \cdot \mathbf{1}_{v_i \geq \tau}] \\
&= \frac{1-p^n}{1-p} \;\; \left(\left(1 - \frac{1}{n^2} - p\right)\frac{e-2}{e-1} + \frac{1}{n^2}\frac{n}{e-1}\right) \\
&< \frac{1-p^n}{e-1}\left(e - 2 + \frac{1}{n(1-p)}\right) \quad . \qquad (11.12)
\end{aligned}$$

To complete the proof, it suffices to show that the right hand side of Inequality (11.12) is at most $1 - 1/e + O(1/n)$. To this end, we try to maximize this term based on parameter c where $p = 1 - c/n$. We can rewrite the right hand side of the inequality as

$$\frac{1-(1-\frac{c}{n})^n}{e-1}\left(e - 2 + \frac{1}{c}\right) \quad .$$

If $c = \Theta(n)$ then this term is at most $(e - 2 + \Theta(1/n))/(e-1) \approx 0.41 + O(1/n)$, which is below $1 - 1/e$ for sufficiently large n. Otherwise $c/n \ll 1$ and we can approximate $(1 - c/n)^n$ as $e^{-c} + O(1/n)$. This upper bounds Inequality (11.12) by $(1 - e^{-c})(e - 2 + 1/c)/(e-1) + O(1/n)$, where the first term is independent of n and is at most $1 - 1/e$ for different constants c; thereby completing the proof. □

We would like to note that the continuity of the CDF of the input distributions is a useful and natural property that can be used by an FTA. This is because making this assumption allows us to design a $(1 - 1/e)$-approximation algorithm, as shown by Theorem 11.5.1, but not assuming this puts a barrier of $1/2$ for any FTA, which is shown in [FGL15, EHLM17]. For example, in the above instance the approximation factor without assuming continuity would be at most $(e - 2)/(e-1) \approx 0.58$, which is below the $1 - 1/e \approx 0.63$ claim of Theorem 11.5.1. This contradiction is because without this assumption on the input distribution the algorithm could not set τ in a way that the probability of selecting an item becomes exactly $1 - 1/e$.

Chapter 12

Matching and Matroid Intersection in the Secretary Model

12.1 Introduction

The *online matroid intersection problem* in the secretary model (OMI) consists of two matroids $\mathcal{M}_1 = (E, \mathcal{I}_1)$ and $\mathcal{M}_2 = (E, \mathcal{I}_2)$, where the elements in E are presented one-by-one to an online algorithm whose goal is to construct a large common independent set. As an element arrives, the algorithm must immediately and irrevocably decide whether to *pick* it, while ensuring that the picked elements always form a common independent set. We assume that the algorithm knows the size of E and has access to independence oracles for the arrived elements. The greedy algorithm, which picks an element whenever possible, is 1/2-competitive. In [GS17], we show:

Theorem 12.1.1. *The online matroid intersection problem in the random arrival model has a $(\frac{1}{2} + \delta)$-competitive randomized algorithm, where $\delta > 0$ is a constant.*

A special case of OMI where both the matroids are partition matroids captures the *online bipartite matching problem* in the random edge arrival (OBME) model. Here, edges of a fixed (but adversarially chosen) bipartite graph G arrive in a uniformly random order and the algorithm must irrevocably decide whether to pick them into a matching. Despite tremendous progress made in the online vertex arrival model [KVV90, MSVV07, GM08, AGKM11, DJK13, WW15, KMZ15], nothing non-trivial was known in the edge arrival model where the edges arrive one-by-one (except in [LS17]). Theorem 12.1.1 gives the first algorithm that beats the greedy algorithm.

Corollary 12.1.2. *The online bipartite matching problem in the random edge arrival model has a $(\frac{1}{2} + \delta)$-competitive randomized algorithm, where $\delta > 0$ is a constant.*

Finally, the simplicity of our OMI algorithm allows us to extend our results to the more general problem of online matching in general graphs (see §12.5).

Theorem 12.1.3. *The online matching problem for general graphs in the random edge arrival model has a $(\frac{1}{2} + \delta')$-competitive randomized algorithm, where $\delta' > 0$ is a constant.*

12.1.1 Comparison to Previous Work

Our main OMI result is interesting in two different aspects: It gives the first linear time algorithm that beats greedy for the classical offline matroid intersection problem; also, it is the first non-trivial algorithm for the general problem of online matroid intersection, where previously nothing better than half was known even for online bipartite matching. Since offline matroid intersection problem is a fundamental problem in the field of combinatorial optimization [Sch03, Chapter 41] and online matching occupies a central position in the field of online algorithms [Meh12], there is a long list of work in both these areas. We state the most relevant works here and refer readers to further related work in §12.1.3.

Offline matroid intersection was brought to prominence in the groundbreaking work of Edmonds [Edm70]. To illustrate the difficulty in moving from bipartite matching to matroid intersection, we note that while the first linear time algorithms that beat half for bipartite matching were designed more than 20 years ago [HK73, ADFS95], the fastest known matroid intersection algorithms till today that beat half make $\Omega(rm)$ calls to the independence oracles, where r is the rank of the optimal solution [CQ16, HKK16]. The quadratic term appears because matroid intersection algorithms rely on constructing auxiliary graphs that needs $\Omega(rm)$ calls [KV08, Chapter 13]. Until our work, achieving a competitive ratio better than half with linear number of independence oracle calls was not known. The key ingredient that allows us to circumvent these difficulties is the *Sampling Lemma* for matroid intersection. We do not construct an auxiliary graph and instead show that any maximal common independent is either already a $(\frac{1}{2}+\delta)$ approximation, or we can improve it to a $(\frac{1}{2}+\delta)$ approximation in a single pass over all the elements.

Online bipartite matching has been studied extensively in the vertex arrival model (see a nice survey by Mehta [Meh12]). Since adversarial arrival order often becomes too pessimistic, the random arrival model (similar to the *secretary problem*) for online matching was first studied by Goel and Mehta [GM08]. Since then, this modeling assumption has become standard [KP09, MY11, KMT11, KMZ15]. The only progress when edges arrive one-by-one has been in showing lower bounds: no algorithm can achieve a competitive ratio better than 0.57 (see [ELSW13]), even when the algorithm is allowed to drop edges.

While nothing was previously known for online matching in the random edge arrival model, similar problems have been studied in the streaming model, most notably by Konrad et al. [KMM12]. They gave the first algorithm that beats the factor of half for bipartite matching in the random arrival streaming model. In this work we generalize their *Hastiness Lemma* to matroids. However, prior works on online matching (see §12.1.3) are not useful as they are tailored to graphs—for instance their reliance on notion of "vertices" cannot be easily extended to the framework of matroids.

The simplicity of our OMI algorithm and flexibility of our analysis allows us to tackle problems of much greater generality, such as general graphs and k-matroid intersection, when previously even special cases like bipartite matching had been considered difficult in the online regime [MV15]. While our results are a qualitative advance, the quantitative improvement is small ($\delta > 10^{-4}$). It remains an interesting challenge to improve the approximation factor δ. Perhaps a more interesting challenge is to relax the random order requirement.

12.1.2 Our Techniques

In this section, we present an overview of our techniques to prove Theorem 12.1.1. Our analysis relies on two observations about the greedy algorithm that are encompassed in the Sampling Lemma and the Hastiness Lemma; the latter being useful to extend our linear time offline matroid intersection result to the online setting. Informally, the Sampling Lemma states that the greedy algorithm cannot perform poorly on a randomly generated OMI instance, and the Hastiness Lemma states that if the greedy algorithm performs poorly, then it picks most of its elements quickly.

Let OPT denote a fixed maximum independent set in the intersection of matroids $\mathcal{M}_1$ and $\mathcal{M}_2$. WLOG, we assume that the greedy algorithm is *bad*—returns a common independent set T of size $\approx \frac{1}{2}|\text{OPT}|$. For offline matroid intersection, by running the greedy algorithm once, one can assume that T is known. For online matroid intersection, we use the Hastiness Lemma to construct T. It states that even if we run the greedy algorithm for a small fraction f (say $< 1\%$) of elements, it already picks a set T of elements of size $\approx \frac{1}{2}|\text{OPT}|$. This lemma was first observed by Konrad et al. [KMM12] for bipartite matching and is generalized to matroid intersection in this work. By running the greedy algorithm for this small fraction f, the lemma lets us assume that we start with an approximately maximal common independent set T with most of the elements ($1 - f > 99\%$) still to arrive.

The above discussion reduces the problem to improving a common independent set T of size $\approx \frac{1}{2}|\text{OPT}|$ to a common independent set of size $\geq (\frac{1}{2} + \delta)|\text{OPT}|$ in a single pass over all the elements. (This is true for both linear-time offline and OMI problems.) Since T is approximately maximal, we know that picking most elements in T eliminates the possibility of picking two OPT elements (one for each matroid). Hence, to beat half-competitiveness, we drop a uniformly random p fraction of these "bad" elements in T to obtain a set S, and try to pick $(1 + \gamma)$OPT elements (for constant $\gamma > 0$) per dropped element. Our main challenge is to construct an online algorithm that can get on average γ gain per dropped element of T in a single pass. The Sampling Lemma for matroid intersection, which is our main technical contribution, comes to rescue.

Sampling Lemma (informal): *Suppose T is a common independent set in matroids $\mathcal{M}_1$ and $\mathcal{M}_2$, and define $\tilde{E} = \text{span}_1(T)$. Let S denote a random set containing each element of T independently with probability $(1 - p)$. Then,*

$$\mathbb{E}_S[|Greedy(\mathcal{M}_1/S, \mathcal{M}_2/T, \tilde{E})|] \geq \left(\frac{1}{1+p}\right) \cdot \mathbb{E}_S[|OPT(\mathcal{M}_1/S, \mathcal{M}_2/T, \tilde{E})|].$$

Intuitively, it says that if we restrict our attention to elements in $\text{span}_1(T)$ then dropping random elements from T allows us to pick more than $1/(1+p) \geq 1/2$ fraction of the optimal intersection. The advantage over half yields the γ gain per dropped element. Applying the lemma requires care as we apply it twice, once for $(\mathcal{M}_1/S, \mathcal{M}_2/T)$ and once for $(\mathcal{M}_1/T, \mathcal{M}_2/S)$, while ensuring that the resulting solutions have few "conflicts" with each other. We overcome this by only considering elements that are in the span of T for exactly one of the matroids.

The proof of the Sampling Lemma involves giving an alternate view of the greedy algorithm for the random OMI instance. Using a carefully constructed invariant and the method of

deferred decisions, we show that the expected greedy solution is not too small.

12.1.3 Further Related Work

Online Matching in Vertex Arrival Model
Karp, Vazirani, and Vazirani [KVV90] presented the ranking algorithm for online bipartite matching in the vertex arrival model. The problem is to find a matching in a bipartite graph where one side of the bipartition is fixed, while the other side vertices arrive in an online fashion. Upon arrival of a vertex, its edges to the fixed vertices are revealed, and the algorithm must immediately and irrevocably decide where to match it. [KVV90] gives an optimal $\left(1 - \frac{1}{e}\right)$-competitive ranking algorithm for adversarial vertex arrival. Since their original proof was incorrect, new ways of analyzing the ranking algorithm have since been developed [BM08, DJK13]. Due to its many applications in the online ad-market, the vertex arrival model, its weighted generalizations, and vertex arrival on both sides have been studied thoroughly (see survey [Meh12, WW15]).

Goel and Mehta [GM08] introduced the random vertex-arrival model. In this model, the adversary may choose the worst instance of a graph, but the online vertices arrive in a random order. The greedy algorithm is already $\left(1 - \frac{1}{e}\right)$-competitive for this problem, as the analysis reduces to [KVV90]. Later works [MY11, KMT11] showed that the ranking algorithm has a competitive ratio of at least 0.69, beating the bounds for adversarial vertex arrival model. There is still a gap between known upper and lower bounds, and closing this gap remains an open problem.

Online Matching in Edge Arrival Model
In the edge arrival model, a fixed bipartite graph is chosen by an adversary and its edges are revealed one by one to an online algorithm that is trying to find a maximum matching. If the edge arrival is adversarial, this problem captures the adversarial vertex arrival model as a special case: constraint the edges incident to a vertex to appear together. The greedy algorithm has a competitive ratio of half and a natural open question is whether we can beat half. The current best hardness result for adversarial edge arrival is ~ 0.57, even when the algorithm is allowed to drop edges (see [ELMS11]).

Matching in the edge arrival model has also been studied in the streaming community. In the streaming model, the matching algorithm can revoke decisions made earlier, but has only a bounded memory; in particular, it has $\tilde{O}(1)$ memory in the streaming model and $\tilde{O}(n)$ memory in the semi-streaming model (see [FKM+05]). The algorithm may make multiple passes over the input; usually trading off the number of passes with the quality of the solution. For bipartite matching in adversarial edge arrival, Kapralov [Kap13] showed that no semi-streaming matching algorithm can do better than $1 - \frac{1}{e}$. Beating the factor of half remains a major open problem.

On the other hand, for uniformly random edge arrival Konrad, Magniez, and Mathieu [KMM12] gave the first single pass algorithm that obtains a 0.501-competitive ratio for bipartite matching in the semi-streaming setting. Their algorithm crucially used the ability to revoke earlier decisions. One of the contributions in this paper is to show that a variant of the

greedy algorithm, which appears simple in hindsight, achieves a competitive ratio better than half in the more restrictive online model.

A weighted generalization of OBME is online bipartite matching for random edge arrival in an edge weighted bipartite graph. This problem has exactly the same setting as OBME; however, the goal is to maximize the weight of the matching obtained. Since it is a generalization of the secretary problem, the greedy algorithm is no longer constant competitive. Korula and Pal [KP09] achieved a breakthrough and gave a constant competitive ratio algorithm for this problem[1]. Kesselheim et al. [KRTV13] later improved their results.

Randomized Greedy Matching Algorithms

Our result for matching in general graphs follows a line of work analyzing variants of the greedy algorithm for matching in general graphs. Dyer and Frieze [DF91] showed that greedy on a uniformly random permutation of the edges cannot achieve a competitive ratio better than half for general graphs; however, it performs well for some classes of sparse graphs. Aaronson et al. [ADFS95] proposed the Modified Randomized Greedy (MRG) algorithm and showed that it has a competitive ratio better than half for general graphs. Poloczek and Szegedy [PS12] provided an argument to improve the bounds on the competitive ratio of this algorithm; however, a gap has emerged in their contrast lemma. A ranking based randomized greedy algorithm has been also shown to have a competitive ratio better than half for general graphs (see [CCWZ14]). Neither MRG nor the ranking algorithm can be implemented in the original setting of [DF91] where the edges arrive in random order and the algorithm is only allowed a single pass. To prove Theorem 12.1.3, we give an algorithm that beats greedy for general graphs with a much simpler analysis and also works in the original setting of [DF91].

Online Matroid Problems

The OMI problem studied in this paper is much more general than online matching and has many other applications, such as the following online network design problem. Consider a central depot that stores different types of commodities and is connected to different cities by rail-links. At various points cities order one of the commodities from the depot and the central manager must immediately and irrevocably decide whether to fulfill the order. If the central manager chooses to fulfill the order, it needs to find a path of rail-links from the depot to that city. Moreover, any rail-link can be used to fulfill at most one order as it can only run a single train. The question is to maximize the number of accepted requests given that there is only a finite amount of each commodity at the depot. This is a matroid intersection problem between a gammoid and a partition matroid. Our result implies an algorithm that beats the factor of half for this problem if the orders arrive uniformly at random. The intersection of two graphic matroids, with applications to electrical networks [Rec05], is another special case of matroid intersection that has received attention in the past [GX96].

Offline Matroid Intersection

Until recently, the fastest unweighted offline matroids intersection algorithm was a variant of Hopcraft-Karp bipartite matching algorithm due to Cunningham [Cun86] taking $O(mr^{3/2}Q)$

[1]They also obtain similar results for hypergraphs and call it the "Hypergraph Edge-at-a-time Matching" problem.

time — m, r, and Q refer to the number of ground elements, the rank of matroid intersection, and to the independence oracle query time, respectively. In 2015, Lee, Sidford, and Wong [LSW15] improved this to $\tilde{O}(m^2Q + m^3)$, both for weighted and unweighted matroid intersection. Not much success has been achieved in proving lower bounds on the oracle complexity of matroid intersection algorithms [Har08]. When looking for a $(1-\epsilon)$ approximate weighted matroid intersection, recent works have improved the running time to $\tilde{O}(mrQ/\epsilon^2)$ [CQ16, HKK16]. Our main Theorem 12.1.1 gives the first algorithm that achieves an approximation factor greater than half with only a linear number of calls to the independence oracles, i.e., in $O(m\,Q)$ time.

12.2 Bipartite Matching

In this section, we consider a special case of online matroid intersection, namely online bipartite matching in the random edge arrival model. Although, this is a special case of the general Theorem 12.1.1, we present it because nothing non-trivial was known before (see §12.1.3) and several of our ideas greatly simplify in this case (in particular the Sampling Lemma), allowing us to lay the framework of our ideas.

12.2.1 Definitions and Notation

An instance of the online bipartite matching problem (G, E, π, m) consists of a bipartite graph $G = (U \cup V, E)$ with $m = |E|$, and where the edges in E arrive according to the order defined by π. We assume that the algorithm knows m but does not know E or π. For $1 \leq i \leq j \leq m$, let $E^{\pi}[i,j]$ denote the set of edges that arrive in between positions i through j according to π [2]. When permutation π is implicit, we abbreviate this to $E[i,j]$.

Greedy denotes the algorithm that picks an edge into the matching whenever possible. Let OPT denote a fixed maximum offline matching of graph G. For $f \in [0,1]$, let T_f^{π} denote the matching produced by Greedy after seeing the first f-fraction of the edges according to order π. For a uniformly random chosen order π,

$$\mathcal{G}(f) := \frac{\mathbb{E}_{\pi}[|T_f^{\pi}|]}{|\text{OPT}|}.$$

Hence, $\mathcal{G}(1)\,|\text{OPT}|$ is the expected output size of Greedy and $\mathcal{G}(\frac{1}{2})\,|\text{OPT}|$ is the expected output size of Greedy after seeing half of the edges. We observe that Greedy has a competitive ratio of at-least half and in §12.6, we show that this ratio is tight for worst case input graphs.[3]

12.2.2 Beating the Factor of Half

Lemma 12.2.1 shows that we can restrict our attention to the case when the expected Greedy size is small. Theorem 12.2.2 gives an algorithm that beats the factor of half for this restricted

[2] We emphasize that our definition also works when i and j are non-integral

[3] We also show that for regular graphs Greedy is at least $\left(1 - \frac{1}{e}\right)$ competitive, and that no online algorithm for OBME can be better than $\frac{69}{84} \approx 0.821$ competitive.

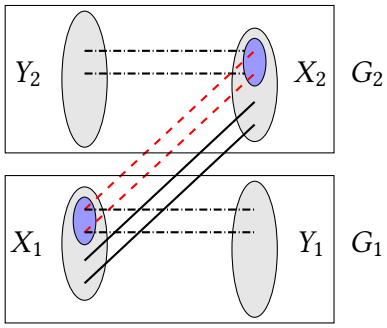

Figure 12.1: $U = X_1 \cup Y_2$ and $V = X_2 \cup Y_1$, where X_1 and X_2 denote the set of vertices matched by Greedy in Phase (a). Here thick-edges are picked and diagonal-dashed-edges are marked. Horizontal-dashed-edges show augmentations for the marked edges.

case.

Lemma 12.2.1. *Suppose there exists an Algorithm $\mathcal{A}$ that achieves a competitive ratio of $\frac{1}{2} + \gamma$ when $\mathcal{G}(1) \leq (\frac{1}{2} + \epsilon)$ for some $\epsilon, \gamma > 0$. Then there exists an algorithm with competitive ratio at least $\frac{1}{2} + \delta$, where $\delta = \frac{\epsilon\gamma}{\frac{1}{2}+\epsilon+\gamma}$.*

Proof. Consider the algorithm that tosses a coin at the beginning and runs Greedy with probability $1 - r$ and Algorithm $\mathcal{A}$ with probability r, where $r > 0$ is some constant. This lemma follows from simple case analysis.

- Case 1: $\mathcal{G}(1) < \frac{1}{2} + \epsilon$
  Since Greedy is always $\frac{1}{2}$ competitive, we can say that in expectation, the competitive ratio will be at least
  $$(1 - r)\frac{1}{2} + r\left(\frac{1}{2} + \gamma\right) = \frac{1}{2} + r\gamma.$$
- Case 2: $\mathcal{G}(1) \geq \frac{1}{2} + \epsilon$
  Since we have no guarantees on the performance of Algorithm $\mathcal{A}$ when Greedy performs well, we assume that it achieves a competitive ratio of 0. Our expected performance will be at least
  $$(1 - r)\left(\frac{1}{2} + \epsilon\right) + 0 = \frac{1}{2} + \epsilon - \frac{r}{2} - r\epsilon.$$

Choosing $r = \frac{\epsilon}{\frac{1}{2}+\epsilon+\gamma}$, we get $\delta \geq \frac{\epsilon\gamma}{\frac{1}{2}+\epsilon+\gamma}$. □

Theorem 12.2.2. *If $\mathcal{G}(1) \leq (\frac{1}{2}+\epsilon)$ for some constant $\epsilon > 0$ then the* Marking-Greedy *algorithm outputs a matching of size at least $(\frac{1}{2} + \gamma)\,|OPT|$ in expectation, where $\gamma > 0$ is a constant.*

Before describing Marking-Greedy, we need the following property about the performance of Greedy in the random arrival model — if Greedy is bad then it makes most of its decisions quickly and incorrectly. We will be interested in the regime where $0 < \epsilon \ll f \ll 1/2$.

Lemma 12.2.3 (Hastiness property: Lemma 2 in [KMM12]). *For any graph G if $\mathcal{G}(1) \leq (\frac{1}{2} + \epsilon)$ for some $0 < \epsilon < \frac{1}{2}$, then for any $0 < f < 1/2$*

$$\mathcal{G}(f) \geq \tfrac{1}{2} - \left(\tfrac{1}{f} - 2\right)\epsilon.$$

MARKING-GREEDY **for Bipartite Matching:**

MARKING-GREEDY consists of two phases (see the pseudocode). In Phase (a), it runs GREEDY for the first f-fraction of the edges, but *picks* each edge *selected* by GREEDY into the final matching only with probability $(1-p)$, where $p > 0$ is a constant. With the remaining probability p, it *marks* the edge e and its vertices, and behaves as if it had been picked. In Phase (b), which is for the remaining $1-f$ fraction of edges, the algorithm runs GREEDY to pick edges on two restricted disjoint subgraphs G_1 and G_2, where it only considers edges incident to exactly one marked vertex in Phase (a). (see Figure 12.1.)

Phase (a) is equivalent to running GREEDY to select elements, but then randomly dropping p fraction of the selected edges. The idea of marking some vertices (by marking an incident edge) is to "protect" them for augmentation in Phase (b). To distinguish if an edge is marked or picked, the algorithm uses auxiliary random bits Ψ that are unknown to the adversary. We assume that $\Psi(e) \sim \text{Bern}(1-p)$ i.i.d. for all $e \in E$.

Algorithm 13 MARKING-GREEDY(G, E, π, m, Ψ)

Phase (a)
Initialize S, T, N_1, N_2 to $\emptyset$
for each element $e \in E^{\pi}[1, fm]$ **do** ▷ GREEDY *while picking and marking*
  if $T \cup e$ is a matching in G **then**
    $T \leftarrow T \cup e$ ▷ *Elements selected by* GREEDY
    if $\Psi(e) = 1$ **then** ▷ *Auxiliary random bits* Ψ
      $S \leftarrow S \cup e$ ▷ *Elements picked into final solution*
    end if
  end if
end for
Phase (b)
Initialize set T_f to T. Let sets X_1, X_2 be vertices of U, V matched in T_f respectively.
Let G_1 be the subgraph of G induced on X_1 and $V \setminus X_2$.
Let G_2 be the subgraph of G induced on $U \setminus X_1$ and X_2.
for each edge $e \in (E^{\pi}[fm, m])$ **do** ▷ GREEDY *on two disjoint subgraphs*
  for $i \in \{1, 2\}$ **do**
    if $e \in G_i$ and $S \cup N_i \cup e$ is a matching **then** ▷ GREEDY step
      $N_i \leftarrow N_i \cup e$ ▷ *New edges picked*
    end if
  end for
end for
return $S \cup N_1 \cup N_2$

Comparison to Konrad et al. [KMM12] For the special case of bipartite matching, we can consider MARKING-GREEDY to be a variant of the streaming algorithm of [KMM12]. For graphs where GREEDY is bad, both algorithms use the first phase to pick an approximately maximal matching T using the Hastiness Lemma. [KMM12] divides the remaining stream into two portions and uses each portion to find greedy matchings, say F_1 and F_2. Since decisions in the streaming setting are revocable, at the end of the stream they use edges in $F_1 \cup F_2$ to find sufficient number of three-augmenting paths w.r.t. T. Their algorithm is not online because it keeps all the matchings till the end. One can view the current algorithm as turning their algorithm into an online one by flipping a coin for each edge in T. In the second phase, it runs GREEDY on two random disjoint subgraphs and use the Sampling Lemma to argue that in expectation the algorithm picks sufficient number of augmenting paths.

While our online matching algorithm is simple and succinct, the main difficulty lies in extending it to OMI as the notions of marking and protecting vertices do not exist. This is also the reason why obtaining a linear time algorithm for offline matroid intersection problem, where Hastiness Lemma is not needed, had been open. Defining and proving the correct form of Sampling Lemma forms the core of our OMI analysis in §12.3.

Proof that MARKING-GREEDY works for Bipartite Matching:

Let G_i denote graphs G_1 or G_2 for $i \in \{1,2\}$. For a fixed order π of the edges, graphs G_i in MARKING-GREEDY are independent of the randomness Ψ. Since the algorithm uses Ψ to pick a random subset of the GREEDY solution, this can be viewed as independently sampling each vertex matched by GREEDY in G_i. Lemma 12.2.4 shows that this suffices to pick in expectation more than the number of marked edges. In essence, we use the randomness Ψ to limit the power of an adversary deciding the order of the edges in Phase (b). While the proof follows from the more general Lemma 12.3.7, we include a simple self-contained proof.

Lemma 12.2.4 (Sampling Lemma[4]). *Consider a bipartite graph $H = (X \cup Y, \tilde{E})$ containing a matching $\tilde{I}$. Let $\Psi(x) \sim Bern(1-p)$ i.i.d. for all $x \in X$, and define $X' = \{x \mid x \in X \text{ and } \Psi(x) = 0\}$. I.e., the vertices of X' are obtained by independently sampling each vertex in X with probability p. Let H' denote the subgraph induced on X' and Y. Then for any arrival order of the edges in H',*

$$\mathbb{E}_\Psi[\text{GREEDY}(H', \tilde{E})] \geq \frac{1}{1+p}\left(p|\tilde{I}|\right).$$

Proof. We prove this statement by induction on $|\tilde{I}|$. Consider the base case $|\tilde{I}| = 1$. Whenever GREEDY does not select any edge, the vertex adjacent to $\tilde{I}$ in X is not sampled. This happens with probability $1-p$. Hence, the expected size of the matching is at least $p \geq \frac{p}{1+p}$, which implies the statement is true when $|\tilde{I}| = 1$.

From the induction hypothesis (I.H.) we can assume the statement is true when the matching size is at most $|\tilde{I}| - 1$. We prove the induction step by contradiction and consider the smallest graph in terms of $|X|$ that does not satisfy the statement. Note that $|X| \geq |\tilde{I}|$. Consider the first

[4]This special case of Lemma 12.3.7 for bipartite matching is also proved in [KMM12]. We are thankful to Deeparnab Chakrabarty for pointing this at IPCO 2017.

edge $e = (x, y)$ that arrives. The first case is when $x \notin X'$ and it happens with probability $1 - p$. Here any edge incident to x does not matter for the remaining algorithm. We use I.H. on the subgraph induced on $(X \backslash x, Y)$ as $|X \backslash x| = (|X| - 1)$. Since this subgraph has a matching of size at least $|\tilde{I}| - 1$, I.H. gives a matching of expected size at least $\frac{p}{1+p}(|\tilde{I}| - 1)$.

The second case is when $x \in X'$ and it happens with probability p. Now edge (x, y) is included in the GREEDY matching for the induced graph on (X', Y). Vertices x and y, along with the edges incident to them, do not participate in the remaining algorithm. We apply I.H. on the subgraph induced on the vertices $(X \backslash x, Y \backslash y)$. Noting that this graph has a matching of size at least $|\tilde{I}| - 2$, I.H. gives a matching of expected size at least $\frac{p}{1+p}(|\tilde{I}| - 2)$. Combining both cases, the expected matching size is at least

$$(1-p)\left(\frac{p}{1+p}(|\tilde{I}| - 1)\right) + p\left(1 + \frac{p}{1+p}(|\tilde{I}| - 2)\right) = \frac{p}{1+p}|\tilde{I}|.$$

This is a contradiction as we assumed that the graph did not satisfy the induction statement, which completes the proof of Lemma 12.2.4. □

We next prove the main lemma needed to prove Theorem 12.2.2. Setting $f = 0.07, p = 0.36$, and $\epsilon = 0.001$ in Lemma 12.2.5, the theorem follows by taking $\gamma > 0.05$.

Lemma 12.2.5. *For any $0 < f < 1/2$ and bipartite graph G,* MARKING-GREEDY *outputs a matching of expected size at least*

$$\left[(1-p)\left(\frac{1}{2} - \left(\frac{1}{f} - 2\right)\epsilon\right) + \frac{p}{1+p}\left(1 - \frac{2\epsilon}{f} - f\right)\right]|OPT|.$$

Proof. We remind the reader that for any $f \in [0,1]$ and any permutation π of the edges, T_f^π denotes the matching that GREEDY produces on $E^\pi[1, fm]$. For $i \in \{1,2\}$, let H_i denote the subgraph of G_i containing all its edges that appear in Phase (b). Let $\mathbb{I}_i$ denote the set of edges of OPT that appear in graph G_i. We use the following claim.

Claim 12.2.6.

$$\mathbb{E}_\pi\left[|\mathbb{I}_1| + |\mathbb{I}_2|\right] \geq \left(1 - \frac{2\epsilon}{f}\right)|OPT|.$$

Proof. We use the following two simple properties of T_1^π (proved in §12.6.6).

Fact 12.2.7.

$$|T_1^\pi| \geq \frac{1}{2}\left(|OPT| + \sum_{e \in OPT} \mathbf{1}[\textit{Both ends of } e \textit{ matched in } T_f^\pi]\right) \textit{ and} \tag{12.1}$$

$$|T_1^\pi| \geq |T_f^\pi| + \frac{1}{2}\sum_{e \in OPT} \mathbf{1}[\textit{Both ends of } e \textit{ unmatched in } T_f^\pi]. \tag{12.2}$$

Note that, $\mathbb{E}_\pi\left[|\mathbb{I}_1| + |\mathbb{I}_2|\right]$ is equal to

$$\mathbb{E}_\pi\left[|\text{OPT}| - \sum_{e \in \text{OPT}} \mathbf{1}[\text{Both ends of } e \text{ matched in } T_f^\pi] - \sum_{e \in \text{OPT}} \mathbf{1}[\text{Both ends of } e \text{ unmatched in } T_f^\pi]\right]$$

$$\geq |\mathrm{OPT}| - \mathbb{E}_\pi\left[2\,|T_1^\pi| - |\mathrm{OPT}|\right] - \mathbb{E}_\pi\left[2(|T_1^\pi| - |T_f^\pi|)\right] \qquad \text{(using (12.1) and (12.2))}$$

$$\geq |\mathrm{OPT}| - 2\epsilon\,|\mathrm{OPT}| - 2\left(\epsilon + \left(\frac{1}{f} - 2\right)\epsilon\right)|\mathrm{OPT}| \qquad \text{(using } \mathcal{G}(1) \leq \tfrac{1}{2} + \epsilon \text{ and Lemma 12.2.3)}$$

$$= \left(1 - \frac{2\epsilon}{f}\right)|\mathrm{OPT}|, \text{ which finishes the proof of the claim.} \qquad \square$$

For $i \in \{1,2\}$, let $\tilde{I}_i \subseteq \mathbb{I}_i$ denote the set of edges of OPT that appear in Phase (b) of Marking-Greedy, i.e., they appear in graph H_i. In expectation over uniform permutation π, at most $f|\mathrm{OPT}|$ elements of OPT can appear in Phase (a). Hence,

$$\mathbb{E}_\pi\left[|\tilde{I}_1| + |\tilde{I}_2|\right] \geq \mathbb{E}_\pi\left[|\mathbb{I}_1| + |\mathbb{I}_2|\right] - f|\mathrm{OPT}| \geq \left(1 - \frac{2\epsilon}{f} - f\right)|\mathrm{OPT}|.$$

Marking a random subset of T_f^π independently is equivalent to marking a random subset of vertices independently. Thus, we can apply Lemma 12.2.4 to both H_1 and H_2. The expected number of edges in $N_1 \cup N_2$ is at least $\frac{p}{1+p}(|\tilde{I}_1| + |\tilde{I}_2|)$, where the expectation is over the auxilary bits Ψ that distinguishes the random set of edges marked. Taking expectations over π and noting that Phase (a) picks $(1-p)\,\mathcal{G}(f)\,|\mathrm{OPT}|$ edges, we have

$$\begin{aligned}
\mathbb{E}_{\Psi,\pi}[|S \cup N_1 \cup N_2|] \quad &= \quad \mathbb{E}_{\Psi,\pi}[|S|] + \mathbb{E}_{\Psi,\pi}[|N_1| + |N_2|] \\
&\geq \mathcal{G}(f)(1-p)\,|\mathrm{OPT}| + \frac{p}{1+p}\mathbb{E}_\pi\left[|\tilde{I}_1| + |\tilde{I}_2|\right] \\
&\geq \left[(1-p)\left(\frac{1}{2} - \left(\frac{1}{f} - 2\right)\epsilon\right) + \frac{p}{1+p}\left(1 - \frac{2\epsilon}{f} - f\right)\right]|\mathrm{OPT}| \quad \text{(by Lemma 12.2.3)}.
\end{aligned}$$

$\square$

12.3 Matroid Intersection

12.3.1 Definitions and Notation

An instance of the online matroid intersection problem $(\mathcal{M}_1, \mathcal{M}_2, E, \pi, m)$ consists of matroids $\mathcal{M}_1$ and $\mathcal{M}_2$ defined on ground set E of size m, and where the elements in E arrive according to the order defined by π. For any $1 \leq i \leq j \leq m$, let $E^\pi[i,j]$ denote the ordered set of elements of E that arrive in positions i through j according to π. For any matroid $\mathcal{M}$ on ground set E, we use $T \in \mathcal{M}$ to denote $T \subseteq E$ is an independent set in matroid $\mathcal{M}$. We use the terminology of matroid restriction and matroid contraction as defined in Oxley [Oxl06]. To avoid clutter, for any $e \in E$ we abbreviate $A \cup \{e\}$ to $A \cup e$ and $A \setminus \{e\}$ to $A \setminus e$.

We note that Greedy is well defined even when matroids $\mathcal{M}_1$ and $\mathcal{M}_2$ are defined on larger ground sets as long as they contain E. This notation will be useful when we run Greedy on matroids after contracting different sets in the two matroids. Since Greedy always produces a maximal independent set, its competitive ratio is at least half (see Theorem 13.8 in [KV08]). This is because an "incorrect" element creates at most two circuits in OPT, one for each matroid.

Algorithm 14 GREEDY $(\mathcal{M}_1, \mathcal{M}_2, E, \pi)$

```
Initialize set T to ∅
for each element e ∈ E^π[1, |E|] do
    if T ∪ e ∈ M_1 ∩ M_2 then
        T ← T ∪ e
    end if
end for
return T
```

Let OPT denote a fixed maximum offline independent set in the intersection of both the matroids. For $f \in [0,1]$, let T_f^π denote the independent set that GREEDY produces after seeing the first f fraction of the edges according to order π. When clear from context, we will often abbreviate T_f^π with T_f. Let $\mathcal{G}(f) := \frac{\mathbb{E}_\pi[|T_f|]}{|\text{OPT}|}$, where π is a uniformly random chosen order.

For $i \in \{1,2\}$, let $\text{span}_i(T) := \{e \mid (e \in E) \wedge (\text{rank}_{\mathcal{M}_i}(T \cup e) = \text{rank}_{\mathcal{M}_i}(T))\}$ denote the span of set $T \subseteq E$ in matroid $\mathcal{M}_i$. Suppose we have $T \in \mathcal{M}_i$ and $e \in \text{span}_i(T)$, then we denote the unique circuit of $T \cup e$ in matroid $\mathcal{M}_i$ by $C_i(T \cup e)$. If $i = 1$, we use $\bar{\imath}$ to denote 2, and vice versa.

We provide a table of all notation used in §12.6.1.

12.3.2 Hastiness Property

Before describing our algorithm MARKING-GREEDY, we need an important hastiness property of GREEDY in the random arrival model. Intuitively, it states that if GREEDY's performance is bad then it makes most of its decisions quickly and incorrectly. This observation was first made by Konrad et al. [KMM12] in the special case of bipartite matching. We extend this property to matroids in Lemma 12.3.1 (proof in §12.6.7). We are interested in the regime where $0 < \epsilon \ll f \ll 1$.

Lemma 12.3.1 (Hastiness Lemma)**.** *For any two matroids $\mathcal{M}_1$ and $\mathcal{M}_2$ on the same ground set E, let T_f^π denote the set selected by* GREEDY *after running for the first f fraction of elements E appearing in order π. Also, for $i \in \{1,2\}$, let $\Phi_i(T_f^\pi) := \text{span}_i(T_f^\pi) \cap OPT$. Now for any $0 < f, \epsilon \leq \frac{1}{2}$, if $\mathbb{E}_\pi[|T_1^\pi|] \leq (\frac{1}{2} + \epsilon)\,|OPT|$ then*

$$\mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cap \Phi_2(T_f^\pi)|\right] \leq 2\epsilon\,|OPT| \qquad \text{and}$$
$$\mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cup \Phi_2(T_f^\pi)|\right] \geq \left(1 - \frac{2\epsilon}{f} + 2\epsilon\right)|OPT|.$$

This implies $\mathcal{G}(f) := \frac{\mathbb{E}_\pi[|T_f^\pi|]}{|OPT|} \geq \left(\frac{1}{2} - \left(\frac{1}{f} - 2\right)\epsilon\right)$.

12.3.3 Beating the Factor of Half for Online Matroid Intersection

Once again, we use Lemma 12.2.1 to restrict our attention to the case when the expected size of GREEDY is small. In Theorem 12.3.2, we give an algorithm that beats the factor of half for this

restricted case, which when combined with Lemma 12.2.1 finishes the proof of Theorem 12.1.1.

Theorem 12.3.2. *For any two matroids $\mathcal{M}_1$ and $\mathcal{M}_2$ on the same ground set E, there exist constants $\epsilon, \gamma > 0$ and a randomized online algorithm* Marking-Greedy *such that if $\mathcal{G}(1) \leq \left(\frac{1}{2} + \epsilon\right)$ then* Marking-Greedy *outputs an independent set in the intersection of both the matroids of expected size at least $\left(\frac{1}{2} + \gamma\right) |OPT|$.*

Marking-Greedy **for OMI:**

Algorithm 15 Marking-Greedy $(\mathcal{M}_1, \mathcal{M}_2, E, \pi, m, \Psi)$

Phase (a)
Initialize S, T to $\emptyset$
for each element $e \in E^{\pi}[1, fm]$ **do** ▷ Greedy *while picking and marking*
  if $T \cup e \in \mathcal{M}_1 \cap \mathcal{M}_2$ **then**
    $T \leftarrow T \cup e$ ▷ *Elements selected by* Greedy
    if $\psi(e) = 1$ **then** ▷ *Auxiliary random bits* Ψ
      $S \leftarrow S \cup e$ ▷ *Elements picked into the final solution*
    end if
  end if
end for
Phase (b)
Fix T_f to T and initialize sets N_1, N_2 to $\emptyset$
for each element $e \in E^{\pi}[fm, m]$ **do** ▷ Greedy *on two disjoint problems*
  for $i \in \{1, 2\}$ **do**
    if $e \in \text{span}_i(T_f)$ and $e \notin \text{span}_{\bar{i}}(T_f)$ **then** ▷ *To ensure disjointness*
      if $(S \cup N_i \cup e \in \mathcal{M}_i)$ and $(T_f \cup N_i \cup e \in \mathcal{M}_{\bar{i}})$ **then** ▷ Greedy step
        $N_i \leftarrow N_i \cup e$ ▷ *Newly picked elements*
      end if
    end if
  end for
end for
return $(S \cup N_1 \cup N_2)$

Marking-Greedy consists of two phases (see notation in §12.6.1). In Phase (a), it runs Greedy for the first f fraction of the elements, but *picks* each element *selected* by Greedy into the final solution only with probability $(1-p)$, where $p > 0$ is a constant. With the remaining probability p, it *marks* the element e, and behaves as if it had been selected. The idea of marking some elements in Phase (a) is that we hope to "augment" them in Phase (b). To distinguish if an element is marked or picked, the algorithm uses auxiliary random bits Ψ that are unknown to the adversary. We assume that $\Psi(e) \sim \text{Bern}(1-p)$ i.i.d. for all $e \in E$.

In Phase (b), one needs to ensure that the augmentations of the marked elements do not conflict with each other. The crucial idea is to use the span of the elements selected by Greedy in Phase (a) as a proxy to find two random disjoint OMI subproblems. The following Fact 12.3.3

(proof in §12.6.6) underlies this intuition. It states that given any independent set S, we can substitute it by any other independent set contained in the span of S. In Lemma 12.3.4 we use it to prove the correctness of Marking-Greedy.

Fact 12.3.3. *Consider any matroid $\mathcal{M}$ and independent sets $A, B, C \in \mathcal{M}$ such that $A \subseteq \text{span}_{\mathcal{M}}(B)$ and $B \cup C \in \mathcal{M}$. Then, $A \cup C \in \mathcal{M}$.*

Lemma 12.3.4. Marking-Greedy *outputs sets S, N_1, and N_2 such that*

$$(S \cup N_1 \cup N_2) \in \mathcal{M}_1 \cap \mathcal{M}_2.$$

Proof. Observe that the outputs sets S, N_1, and N_2 of Marking-Greedy satisfy the following for $i \in \{1,2\}$:

$$N_i \in \mathcal{M}_i/S \cap \mathcal{M}_{\bar{i}}/T_f \qquad \text{(due to Line 14)} \tag{12.3}$$

$$N_i \subseteq \text{span}_{\mathcal{M}_i/S}(T_f \setminus S) \qquad \text{(due to Line 13)} \tag{12.4}$$

From Property (12.3) above we know $N_{\bar{i}} \in \mathcal{M}_i/T_f$, which implies $N_{\bar{i}} \cup (T_f \setminus S) \in \mathcal{M}_i/S$ because $S \subseteq T_f \in \mathcal{M}_i$. Also, Property (12.4) implies $N_i \subseteq \text{span}_{\mathcal{M}_i/S}(T_f \setminus S)$. Using Fact 12.3.3, we have $\left(N_1 \cup N_2\right) \in \mathcal{M}_i/S$. □

Proof that Marking-Greedy works for OMI:

We know from Lemma 12.3.1 that $\mathcal{G}(f)$ is close to half for $\epsilon \ll f \ll 1$. In the following Lemma 12.3.5, we show that Marking-Greedy (which returns $S \cup N_1 \cup N_2$ by Lemma 12.3.4) gets an improvement over Greedy. This completes the proof of Theorem 12.3.2 to give $\gamma \geq 0.03$ for $\epsilon = 0.001$, $f = 0.05$, and $p = 0.33$. The rest of the section is devoted to proving the following lemma.

Lemma 12.3.5. Marking-Greedy *outputs sets S, N_1, and N_2 such that*

$$\mathbb{E}_{\pi,\Psi}[|S \cup N_1 \cup N_2|] \geq (1-p)\,\mathcal{G}(f)\,|OPT| + \frac{2p}{1+p}\left(1 - \frac{2\epsilon}{f} - 2\epsilon - f - \mathcal{G}(f)\right)|OPT|.$$

Lemma 12.3.5. We treat the sets $S \subseteq T_f, N_1$, and N_2 as random sets depending on π and Ψ. Since Marking-Greedy ensures the sets are disjoint,

$$\begin{aligned}\mathbb{E}_{\pi,\Psi}[|S \cup N_1 \cup N_2|] &= \mathbb{E}_{\pi,\Psi}[|S|] + \mathbb{E}_{\pi,\Psi}[|N_1| + |N_2|] \\ &\geq (1-p)\,\mathcal{G}(f)\,|\text{OPT}| + \mathbb{E}_{\pi,\Psi}[|N_1| + |N_2|]. \end{aligned} \tag{12.5}$$

Next, we lower bound $\mathbb{E}_{\pi,\Psi}[|N_1| + |N_2|]$ by observing that for $i \in \{1,2\}$, N_i is the result of running Greedy on the following restricted set of elements.

Definition 12.3.6 (Sets $\tilde{E}_i$)**.** *For $i \in \{1,2\}$, we define $\tilde{E}_i$ to be the set of elements e that arrive in Phase (b) and satisfy $e \in \text{span}_i(T_f)$ and $e \notin \text{span}_{\bar{i}}(T_f)$.*

It's easy to see that N_i is obtained by running Greedy on the matroids $\mathcal{M}_i/S$ and $\mathcal{M}_{\bar{i}}/T_f$ with respect to elements in $\tilde{E}_i$, i.e. $N_i = \text{Greedy}(\mathcal{M}_i/S, \mathcal{M}_{\bar{i}}/T_f, \tilde{E}_i)$. To lower bound $\mathbb{E}_{\pi,\Psi}[|N_1| + |N_2|]$,

we use the following Sampling Lemma (proved in §12.4) that forms the core of our technical analysis. Intuitively, it says that if S is a random subset of T_f then for the obtained random OMI instance, with optimal solution of expected size $p\,|\tilde{I}|$, Greedy performs better than half-competitiveness even for adversarial arrival order of ground elements.

Lemma 12.3.7 (Sampling Lemma). *Given matroids $\mathcal{M}_1, \mathcal{M}_2$ on ground set E, a set $T \in \mathcal{M}_1 \cap \mathcal{M}_2$, and $\Psi(e) \sim Bern(1-p)$ i.i.d. for all $e \in T$, we define set $S := \{e \mid e \in T \text{ and } \Psi(e) = 1\}$. I.e., S is a set achieved by dropping each element in T independently with probability p. For $i \in \{1,2\}$, consider a set $\tilde{E} \subseteq \mathsf{span}_i(T)$ and a set $\tilde{I} \subseteq \tilde{E}$ satisfying $\tilde{I} \in \mathcal{M}_i \cap (\mathcal{M}_{\bar{i}}/T)$. Then for any arrival order of the elements of $\tilde{E}$, we have*

$$\mathbb{E}_\Psi[\textsc{Greedy}(\mathcal{M}_i/S, \mathcal{M}_{\bar{i}}/T, \tilde{E})] \geq \frac{1}{1+p}\left(p\,|\tilde{I}|\right).$$

To use the Sampling Lemma, in Claim 12.3.8 we argue that in expectation there exist disjoint sets $\tilde{I}_i \subseteq \tilde{E}_i$ of "large" size that satisfy the preconditions of the Sampling Lemma (proof uses Hastiness Lemma and is deferred to §12.3.4).

Claim 12.3.8. *If $\mathcal{G}(1) \leq \left(\frac{1}{2} + \epsilon\right)$ then for $i \in \{1,2\}$ $\exists$ disjoint sets $\tilde{I}_i \subseteq \tilde{E}_i$ s.t.*

(i) $\mathbb{E}_\pi\left[|\tilde{I}_1| + |\tilde{I}_2|\right] \geq 2\left(1 - \frac{2\epsilon}{f} - f - \mathcal{G}(f)\right)|OPT|$.

(ii) $\tilde{I}_i \in \mathcal{M}_i \cap (\mathcal{M}_{\bar{i}}/T_f)$.

Finally, to finish the proof of Lemma 12.3.5, we use the sets $\tilde{I}_i$ from the above Claim 12.3.8 as $\tilde{I}$ and sets $\tilde{E}_i$ as $\tilde{E}$ in the Sampling Lemma 12.3.7. From (12.5) and Claim 12.3.8, we get

$$\begin{aligned}\mathbb{E}_{\pi,\Psi}[|S \cup N_1 \cup N_2|] &\geq (1-p)\,\mathcal{G}(f)\,|\mathrm{OPT}| + \frac{p}{1+p}\,\mathbb{E}_\pi\left[|\tilde{I}_1| + |\tilde{I}_2|\right] \\ &\geq (1-p)\,\mathcal{G}(f)\,|\mathrm{OPT}| + \frac{2p}{1+p}\left(1 - \frac{2\epsilon}{f} - f - \mathcal{G}(f)\right)|\mathrm{OPT}|. \qquad \square\end{aligned}$$

Claim 12.3.8. Recall $\Phi_i(T_f^\pi) := \mathsf{span}_i(T_f^\pi) \cap \mathrm{OPT}$. Let $\mathbb{I}_i$ denote $\Phi_i(T_f^\pi) \setminus \Phi_{\bar{i}}(T_f^\pi)$. We construct sets $\tilde{I}_i$ by removing some elements from $\mathbb{I}_i$, which implies $\tilde{I}_i \in \mathcal{M}_i$ because $\mathbb{I}_i \in \mathcal{M}_i$. We first show that $|\mathbb{I}_1| + |\mathbb{I}_2|$ is large. From the Hastiness Lemma 12.3.1, we have

$$\begin{aligned}\mathbb{E}_\pi\left[|\mathbb{I}_1| + |\mathbb{I}_2|\right] &= \mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cup \Phi_2(T_f^\pi)|\right] - \mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cap \Phi_2(T_f^\pi)|\right] \\ &\geq \left(1 - \frac{2\epsilon}{f}\right)|\mathrm{OPT}|. \qquad (12.6)\end{aligned}$$

Next, we ensure that $\tilde{I}_i \in \mathcal{M}_{\bar{i}}/T_f$. Note that $\mathbb{I}_{\bar{i}} \subseteq \mathsf{span}_{\bar{i}}(T_f)$. Let $X_{\bar{i}}$ denote a minimum subset of elements of T_f such that $\mathsf{span}_{\bar{i}}(X_{\bar{i}} \cup \mathbb{I}_{\bar{i}}) = \mathsf{span}_{\bar{i}}(T_f)$. Since $\mathbb{I}_{\bar{i}}$ and T_f are independent in $\mathcal{M}_{\bar{i}}$, we have $|X_{\bar{i}}| = |T_f| - |\mathbb{I}_{\bar{i}}|$. Now starting with $(\mathbb{I}_i \cup \mathbb{I}_{\bar{i}}) \in \mathcal{M}_{\bar{i}}$, we add elements of $X_{\bar{i}}$ into it. We will remove at most $|X_{\bar{i}}|$ elements from $\mathbb{I}_i$ to get a set $\mathbb{I}'_i$ such that $(\mathbb{I}'_i \cup X_{\bar{i}} \cup \mathbb{I}_{\bar{i}}) \in \mathcal{M}_{\bar{i}}$ as $(X_{\bar{i}} \cup \mathbb{I}_{\bar{i}}) \in \mathcal{M}_{\bar{i}}$. Using Fact 12.3.3 and $\mathsf{span}_{\bar{i}}(X_{\bar{i}} \cup \mathbb{I}_{\bar{i}}) = \mathsf{span}_{\bar{i}}(T_f)$, we also have $\mathbb{I}'_i \cup T_f \in \mathcal{M}_{\bar{i}}$. One can use a similar argument to obtain set $\mathbb{I}'_{\bar{i}}$ and X_i such that $\mathbb{I}'_{\bar{i}} \cup T_f \in \mathcal{M}_i$. Since $\mathbb{E}_\pi\left[|X_i|\right] = \mathbb{E}_\pi\left[|T_f| - |\mathbb{I}_i|\right]$,

$$\mathbb{E}_\pi[|\mathbb{I}'_1| + |\mathbb{I}'_2|] \geq \mathbb{E}_\pi[|\mathbb{I}_1| + |\mathbb{I}_2| - |X_1| - |X_2|] = 2\,\mathbb{E}_\pi[|\mathbb{I}_1| + |\mathbb{I}_2| - |T_f|] \qquad (12.7)$$

Finally, to ensure that $\tilde{I}_i \subseteq \tilde{E}_i$, observe that any element $e \in \mathbb{I}'_i$ already satisfies $e \in \mathsf{span}_i(T_f)$ and $e \notin \mathsf{span}_{\bar{i}}(T_f)$. To ensure that these elements also appear in Phase (b), note that all elements of $\mathbb{I}'_i$ belong to OPT. Hence, in expectation over π, at most $f\,|\mathrm{OPT}|$ of these elements can appear in Phase (a). The remaining elements appear in Phase (b). Thus, combining the following equation with (12.8) and (12.9) completes the proof of Lemma 12.3.5

$$\mathbb{E}_\pi\left[|\tilde{I}_1| + |\tilde{I}_2|\right] \geq \mathbb{E}_\pi\left[|\mathbb{I}'_1| + |\mathbb{I}'_2|\right] - f\,|\mathrm{OPT}|. \qquad \square$$

12.3.4 Existence of Large Disjoint Sets for Claim 12.3.8

Finally, we prove the missing Claim 12.3.8 that in expectation there exist disjoint sets $\tilde{I}_i \subseteq \tilde{E}_i$ of "large" size that satisfy the preconditions of the Sampling Lemma

Claim 12.3.8. Recall $\Phi_i(T_f^\pi) := \mathsf{span}_i(T_f^\pi) \cap \mathrm{OPT}$. Let $\mathbb{I}_i$ denote $\Phi_i(T_f^\pi) \setminus \Phi_{\bar{i}}(T_f^\pi)$. We construct sets $\tilde{I}_i$ by removing some elements from $\mathbb{I}_i$, which implies $\tilde{I}_i \in \mathcal{M}_i$ because $\mathbb{I}_i \in \mathcal{M}_i$. We first show that $|\mathbb{I}_1| + |\mathbb{I}_2|$ is large. From the Hastiness Lemma 12.3.1, we have

$$\begin{aligned}\mathbb{E}_\pi\left[|\mathbb{I}_1| + |\mathbb{I}_2|\right] &= \mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cup \Phi_2(T_f^\pi)|\right] - \mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cap \Phi_2(T_f^\pi)|\right] \\ &\geq \left(1 - \frac{2\epsilon}{f}\right)|\mathrm{OPT}|. \end{aligned} \tag{12.8}$$

Next, we ensure that $\tilde{I}_i \in \mathcal{M}_{\bar{i}}/T_f$. Note that $\mathbb{I}_{\bar{i}} \subseteq \mathsf{span}_{\bar{i}}(T_f)$. Let $X_{\bar{i}}$ denote a minimum subset of elements of T_f such that $\mathsf{span}_{\bar{i}}(X_{\bar{i}} \cup \mathbb{I}_{\bar{i}}) = \mathsf{span}_{\bar{i}}(T_f)$. Since $\mathbb{I}_{\bar{i}}$ and T_f are independent in $\mathcal{M}_{\bar{i}}$, we have $|X_{\bar{i}}| = |T_f| - |\mathbb{I}_{\bar{i}}|$. Now starting with $(\mathbb{I}_i \cup \mathbb{I}_{\bar{i}}) \in \mathcal{M}_{\bar{i}}$, we add elements of $X_{\bar{i}}$ into it. We will remove at most $|X_{\bar{i}}|$ elements from $\mathbb{I}_i$ to get a set $\mathbb{I}'_i$ such that $(\mathbb{I}'_i \cup X_{\bar{i}} \cup \mathbb{I}_{\bar{i}}) \in \mathcal{M}_{\bar{i}}$ as $(X_{\bar{i}} \cup \mathbb{I}_{\bar{i}}) \in \mathcal{M}_{\bar{i}}$. Using Fact 12.3.3 and $\mathsf{span}_{\bar{i}}(X_{\bar{i}} \cup \mathbb{I}_{\bar{i}}) = \mathsf{span}_{\bar{i}}(T_f)$, we also have $\mathbb{I}'_i \cup T_f \in \mathcal{M}_{\bar{i}}$. One can use a similar argument to obtain set $\mathbb{I}'_{\bar{i}}$ and X_i such that $\mathbb{I}'_{\bar{i}} \cup T_f \in \mathcal{M}_i$. Since $\mathbb{E}_\pi\left[|X_i|\right] = \mathbb{E}_\pi\left[|T_f| - |\mathbb{I}_i|\right]$,

$$\mathbb{E}_\pi[|\mathbb{I}'_1| + |\mathbb{I}'_2|] \geq \mathbb{E}_\pi[|\mathbb{I}_1| + |\mathbb{I}_2| - |X_1| - |X_2|] = 2\,\mathbb{E}_\pi[|\mathbb{I}_1| + |\mathbb{I}_2| - |T_f|] \tag{12.9}$$

Finally, to ensure that $\tilde{I}_i \subseteq \tilde{E}_i$, observe that any element $e \in \mathbb{I}'_i$ already satisfies $e \in \mathsf{span}_i(T_f)$ and $e \notin \mathsf{span}_{\bar{i}}(T_f)$. To ensure that these elements also appear in Phase (b), note that all elements of $\mathbb{I}'_i$ belong to OPT. Hence, in expectation over π, at most $f\,|\mathrm{OPT}|$ of these elements can appear in Phase (a). The remaining elements appear in Phase (b). Thus, combining the following equation with (12.8) and (12.9) completes the proof

$$\mathbb{E}_\pi\left[|\tilde{I}_1| + |\tilde{I}_2|\right] \geq \mathbb{E}_\pi\left[|\mathbb{I}'_1| + |\mathbb{I}'_2|\right] - f\,|\mathrm{OPT}|. \qquad \square$$

12.4 Sampling Lemma

We prove the lemma for $i = 1$ as the other case is analogous.

12.4.1 Alternate View of the Sampling Lemma

We prove the Sampling Lemma by first showing that Greedy$(\mathcal{M}_1/S, \mathcal{M}_2/T, \tilde{E})$ produces the same output as algorithm Samp-Alg (proof deferred to §12.4.3).

Lemma 12.4.1. *Given a fixed Ψ and assuming the elements of $\tilde{E}$ are presented in the same order, the output of* Samp-Alg *is the same as the output of* Greedy$(\mathcal{M}_1/S, \mathcal{M}_2/T, \tilde{E})$.

The idea behind Samp-Alg is to run Greedy, but postpone distinguishing between the elements that are selected by Greedy (set T) and picked by our algorithm (set S). This limits what an adversary can do while ordering the elements of $\tilde{E}$. Intuitively, the sets in Samp-Alg denote the following:

- N' denotes the new elements to be added to the independent set.
- T' are the elements of T for which we still haven't read the random bit Ψ.
- S' are the elements $e \in T$ for which we have read Ψ and they turn out to be picked, i.e., $\Psi(e) = 1$.

Algorithm 16 Samp-Alg

Input: $\mathcal{M}_1, \mathcal{M}_2, T$, and random bits $\Psi \in \{0,1\}^{|T|}$.
Initialize: N', S' to $\emptyset$, and $T' = T$
for each element $e \in \tilde{E}$ **do**
  if $T \cup N' \cup e \in \mathcal{M}_2$ **then**
    Let $C \leftarrow C_1(S' \cup N' \cup T', e) \cap T'$ ▷ *Unread elements of the formed circuit*
    for each element $f \in C$ **do**
      $T' \leftarrow T' \setminus f$
      if $\Psi(f) = 1$ **then** ▷ *Auxiliary random bits Ψ*
        $S' \leftarrow S' \cup f$ ▷ *Already picked elements*
      else
        $N' \leftarrow N' \cup e$ ▷ *Newly picked elements*
        Break
      end if
    end for
  end if
end for
return N'

12.4.2 Proof of the Sampling Lemma

By Lemma 12.4.1, it suffices to prove that given the preconditions of the Sampling Lemma, Samp-Alg produces an output of expected size at least $\frac{p}{1+p}|\tilde{I}|$. More precisely, we need to show that if Ψ in Samp-Alg is chosen as $\Psi(e) \sim \text{Bern}(1-p)$ i.i.d. for all $e \in T$, we have $\mathbb{E}_\Psi[|N'|] \geq \frac{p}{1+p}|\tilde{I}|$.

The main idea of the proof is to argue that before every iteration of the for-loop in Line 2, there are "sufficient" number of elements that are still to arrive and can be added to our solution. To achieve this, we define a set $\mathbb{I}'$, which intuitively denotes the set of OPT elements that are still to arrive and can be added to the current solution. The properties of $\mathbb{I}'$ are rigorously captured in Invariant 12.4.2, where $\tilde{E}_r$ denotes the remaining elements of $\tilde{E}$ that are still to be considered in the for-loop. Due to Lemma 12.4.1, this also denotes the elements of $\tilde{E}$ that are still to arrive for Greedy. Starting with $\mathbb{I}' = \tilde{I}$ at the beginning of Samp-Alg, we wish to maintain the following.

Invariant 12.4.2. *For given sets S', N', T, and $\tilde{E}_r \subseteq \tilde{E}$, set $\mathbb{I}'$ satisfies this invariant if*

$$S' \cup N' \cup I' \in \mathcal{M}_1 \tag{12.10}$$

$$T \cup N' \cup I' \in \mathcal{M}_2 \tag{12.11}$$

$$I' \subseteq \tilde{E}_r \tag{12.12}$$

As the algorithm Samp-Alg progresses, set $\mathbb{I}'$ has to drop some of its elements so that it continues to satisfy Invariant 12.4.2. These drops from $\mathbb{I}'$ are rigorously captured in Updates 12.4.3. Note that set $\mathbb{I}'$ and Updates 12.4.3 are just for analysis purposes, and never appear in the actual algorithm. Starting with $\mathbb{I}' = \tilde{I}$ at the beginning of Samp-Alg and satisfying Invariant 12.4.2, in Claim 12.4.4 we prove that Updates 12.4.3 to $\mathbb{I}'$ ensure that the invariant is always satisfied. This lets us use induction to prove in Claim 12.4.5 that Updates 12.4.3 never drop too many elements from $\mathbb{I}'$ and Samp-Alg returns an independent set of large size.

Updates 12.4.3. *We perform the following updates to $\mathbb{I}'$ whenever* Samp-Alg *reaches Line 8 or Line 10. Claim 12.4.4 shows that these updates are well-defined.*

- *Line 8: If circuit $C_1(S' \cup N' \cup \mathbb{I}' \cup f)$ is non-empty then remove an element from $\mathbb{I}'$ belonging to $C_1(S' \cup N' \cup \mathbb{I}' \cup f)$ to break the circuit.*
- *Line 10: If circuit $C_1(S' \cup N' \cup \mathbb{I}' \cup e)$ is non-empty then remove an element from $\mathbb{I}'$ belonging to $C_1(S' \cup N' \cup \mathbb{I}' \cup e)$ to break the circuit. If $C_2(T \cup N' \cup \mathbb{I}' \cup e)$ is non-empty then remove another element from $\mathbb{I}'$ belonging to $C_2(T \cup N' \cup \mathbb{I}' \cup e)$ to break the circuit. In the special case where $e \in \mathbb{I}'$, we remove e from $\mathbb{I}'$.*

The following claim (proof deferred to §12.4.4) shows that Updates 12.4.3 maintain Invariant 12.4.2.

Claim 12.4.4. *Given matroids $\mathcal{M}_1, \mathcal{M}_2$, a set $T \in \mathcal{M}_1 \cap \mathcal{M}_2$, a set $\tilde{E}_r \subseteq \mathsf{span}_1(T)$ (denoting the set of remaining elements), and $\Psi(e) \sim Bern(1-p)$ i.i.d. for all $e \in T$, suppose there exists a set I' satisfying Invariant 12.4.2 at the beginning of some iteration of the for-loop in Line 2 of* Samp-Alg*. Then*

(i) Updates 12.4.3 are well-defined.

(ii) Updates 12.4.3 ensure that Invariant 12.4.2 hold at the end of the iteration.

Finally, we use Invariant 12.4.2 to prove the main claim.

Claim 12.4.5. *Given matroids $\mathcal{M}_1, \mathcal{M}_2$, a set $T \in \mathcal{M}_1 \cap \mathcal{M}_2$, a set $\tilde{E}_r \subseteq \tilde{E} \subseteq \mathsf{span}_1(T)$ (denoting the set of remaining elements), and $\Psi(e) \sim Bern(1-p)$ i.i.d. for all $e \in T$, suppose there exists*

a set I' satisfying Invariant 12.4.2 at the beginning of some iteration of the for-loop of Line 2 in Samp-Alg. *Then the value of N' at the end of* Samp-Alg *satisfies*

$$\mathbb{E}_\Psi[|N'|] \geq \frac{p}{1+p}|I'|$$

Proof. To prove the claim we use induction on $|\mathbb{I}'|$ where $\mathbb{I}' \subseteq \tilde{E}$. WLOG we can assume that e is the first element such that C in Line 4 is non-empty. Let $C = \{t_1, \ldots, t_l\}$ where $l \geq 1$. For $j \in \{0, \ldots, l-1\}$, define event B_j as $\Psi(t_1) = \Psi(t_2) = \cdots = \Psi(t_j) = 1$ and $\Psi(t_{j+1}) = 0$. Also, define $\bar{B}$ as $\Psi(t_1) = \ldots \Psi(t_l) = 1$.

Base Case: Since C is a non-empty circuit, we can assume that any element $f \in C$ satisfies the condition $\Psi(f) = 0$ with probability p. Hence, $|N'| \geq 1$ with probability at least p, proving the required claim.

Induction Step: The events $B_0, \ldots, B_{l-1}$, and $\bar{B}$ partition the entire probability space.

Case 1 (Event B_j) : Since applying the Updates 12.4.3 preserves Invariant 12.4.2 by Claim 12.4.4, we can apply the induction hypothesis to the updated set I'. Moreover. Updates 12.4.3 remove at most $j+2$ elements from I' in the event B_j. Applying the Induction hypothesis, we can conclude that $\mathbb{E}_\Psi[|N'| \,\big|\, B_j] \geq 1 + \frac{p}{1+p}(|I'| - j - 2)$.

Case 2 (Event $\bar{B}$): Since applying the Updates 12.4.3 preserves Invariant 12.4.2 by Claim 12.4.4, we can apply the induction hypothesis to the updated set I'. Moreover, Updates 12.4.3 remove l elements from $\mathbb{I}'$ in the event $\bar{B}$. Conditioned on the event $\bar{B}$ and applying the induction hypothesis to the updated set I', we can conclude $\mathbb{E}_\Psi[|N'|] \geq \frac{p}{1+p}(|I'| - l)$.

Combining both the cases, we have $\mathbb{E}_\Psi[|N|]$ is at least

$$\begin{aligned}
&\sum_{j=0}^{l-1} \Pr[B_j] \cdot \mathbb{E}_\Psi[|N'| \,\big|\, B_j] + \Pr[\bar{B}] \cdot \mathbb{E}_{\bar{B}}[|N| \,\big|\, \bar{B}] \\
&\geq \sum_{j=0}^{l-1} (1-p)^j p \left(1 + \frac{p}{1+p}(|\mathbb{I}'| - 2 - j)\right) + (1-p)^l \left(\frac{p}{1+p}(|\mathbb{I}'| - l)\right) \\
&= \frac{p}{1+p}|\mathbb{I}'| \qquad \text{using } \sum_{j=0}^{l-1} j\,(1-p)^j = -\frac{l\,(1-p)^l}{p} - \frac{(1-p)}{p^2}((1-p)^l - 1).
\end{aligned}$$

□

To finish the proof of Lemma 12.3.7, we start with $\mathbb{I}' := \tilde{I}$, $T' := T$, $N' := \emptyset$, and $S' := \emptyset$ in Claim 12.4.5. The preconditions hold true because $T \cup \mathbb{I} \in \mathcal{M}_2$, $T \in \mathcal{M}_1$, and $\mathbb{I} \in \mathcal{M}_1$.

12.4.3 Proof of the Alternate View of Sampling Lemma

We restate the lemma for convenience.

Lemma 12.4.1. *Given a fixed Ψ and assuming the elements of $\tilde{E}$ are presented in the same order, the output of* Samp-Alg *is the same as the output of* Greedy$(\mathcal{M}_1/S, \mathcal{M}_2/T, \tilde{E})$.

Starting with $S' = \emptyset$, $N' = \emptyset$, and $T' = T$, we make some simple observations and prove a small claim before proving Lemma 12.4.1.

Observation 12.4.6. *The for-loop defined in Line 2 of* Samp-Alg *maintains the following invariant*

$$S \subseteq S' \cup T' \subseteq T$$

Proof. To show the first containment, observe that for each element if an $\Psi(e) = 1$ then it simply moves from T' to S'. Hence, all the elements of $S \subseteq S' \cup T'$. To observe, the second containment, note that an element of T' either moves into S' or gets removed. Since T' was initialized to T, the second containment follows. □

Observation 12.4.7. *The for-loop defined in Line 2 of* Samp-Alg *maintains the following invariant*

$$S' \cup N' \cup T' \in \mathcal{M}_1.$$

Proof. Since $T \in \mathcal{M}_1$ and $S' = T' = \emptyset$ at the beginning, we can conclude that this is correct at the beginning of Samp-Alg. Now consider an iteration of the for-loop defined in Line 2. When an element f is added to S' in Line 8, it must have belonged to T', implying that $S' \cup N' \cup T'$ is unchanged. If an element e is added to N' (in Line 10) then we must remove an element f from T' (due to Line 6), which belonged to the unique circuit $C_1(S' \cup T' \cup N', e)$. Hence, $S' \cup N' \cup e \cup (T' \setminus f)$ is still an independent set in $\mathcal{M}_1$. □

Claim 12.4.8. *For an element $e \in \tilde{E}$, if Line 4 of* Samp-Alg *is reached then $C_1(S' \cup N' \cup T', e)$ is non-empty.*

Proof. We know $\tilde{E} \subseteq \mathsf{span}_1(T)$. Moreover, $S' \cup T' \subseteq T \subseteq \mathsf{span}_1(T)$ (using Observation 12.4.6). Hence, $S' \cup T' \cup \tilde{E} \subseteq \mathsf{span}_1(T)$ implies

$$\mathsf{rank}_{\mathcal{M}_1}(S' \cup T' \cup \tilde{E}) \leq |T|. \tag{12.13}$$

We prove the lemma by contradiction and assume circuit $C_1(S' \cup N' \cup T', e)$ is empty. Using Observation 12.4.7, this implies $(S' \cup N' \cup T' \cup e) \in \mathcal{M}_1$. Now, $\mathsf{rank}_{\mathcal{M}_1}(S' \cup N' \cup T' \cup e) = |S' \cup N' \cup T'| + 1 \leq \mathsf{rank}_{\mathcal{M}_1}(S' \cup T' \cup \tilde{E}) \leq |T|$ using (12.13). In the next paragraph, we show that the algorithm always maintains $|S' \cup N' \cup T'| = |T|$, which gives the contradiction $|T| + 1 \leq |T|$.

To prove $|S' \cup N' \cup T'| = |T|$, we note that the only time T' decreases is in Line 6. In this case, we either add the dropped element to S' in Line 8 or a new element to N' in Line 10. Hence, the $|S' \cup N' \cup T'|$ is unchanged in the for-loop of Line 2. Since we initialize $S' = N' = \emptyset$ and $T' = T$, we can conclude that this $|S' \cup N' \cup T'| = |T|$ is maintained. □

We now have the tools to prove Lemma 12.4.1.

Lemma 12.4.1. Let us assume the elements of $\tilde{E}$ are presented in order $e_1, \ldots, e_t$ where $t = |\tilde{E}|$. We will use induction on the following hypothesis.

Induction Hypothesis (I.H.): After both algorithms have seen the first k elements $e_1, \ldots, e_k$, the set N' in Samp-Alg is the same as the set of elements selected by $\textsc{Greedy}(\mathcal{M}_1/S, \mathcal{M}_2/T, \tilde{E})$.

Base Case: Initially, both algorithms have not selected any element. Hence, $N' = \emptyset$ is the set of all elements selected by Greedy.

Induction Step: Suppose the I.H. is true for elements $e_1, \ldots, e_{k-1}$ and we are considering element e_k. If e_k does not satisfy $T \cup N' \cup e_k \in \mathcal{M}_2$, then it will also not satisfy the same condition for Greedy because N' is the set selected by Greedy (by I.H.) and $N' \cup e \notin \mathcal{M}_2/T$. In this case we are done with the induction step. From now assume $T \cup N' \cup e_k \in \mathcal{M}_2$.

Suppose e_k is added to N' in Samp-Alg, then we claim $\textsc{Greedy}(\mathcal{M}_1/S, \mathcal{M}_2/T, \tilde{E})$ will also select this element. The only location where e_k could be added is Line 10. This occurs when we remove some appropriate element $f \in T'$ to ensure $S' \cup (T' \setminus f) \cup N' \cup e \in \mathcal{M}_1$. Furthermore $\Psi(f) = 0$ implies $f \notin S$. By Observation 12.4.6, set $S \subseteq S' \cup T' \setminus f$. Hence, $S' \cup (T' \setminus f) \cup N' \cup e \in \mathcal{M}_1$ implies $S \cup N' \cup e \in \mathcal{M}_1$ and Greedy will also select this element.

Next, suppose e_k is not picked by the algorithm. By Claim 12.4.8, we know that $C_1(S' \cup N' \cup T', e)$ is non-empty. In this case, every element $f \in C$ encountered in the for-loop of Line 5 must have had $\Psi(f) = 1$. This implies that at the end of the for-loop of Line 5, circuit $C_1(S' \cup N' \cup T', e) \subseteq S' \cup N'$. Since $S' \subseteq S$ (by Observation 12.4.6), this gives $N' \cup e \notin \mathcal{M}_1/S$. Hence, Greedy cannot select element e_k. □

12.4.4 Proof that the Updates are valid

In this section we prove Claim 12.4.4 by showing that Updates 12.4.3 are well-defined and maintain Invariant 12.4.2.

Claim 12.4.4. Since Invariant 12.4.2 holds before entering into the for-loop in Line 2, we prove this claim by showing that after one iteration of the for-loop, i.e., after arrival of an element e, Properties ((i)) and ((ii)) hold.

We first show that the properties hold if the set C in Line 4 is empty. Since in this case we do not perform any updates to sets $S', N', \mathbb{I}'$, and T', Invariant 12.10, Invariant 12.11, and well-definedness trivially hold. To prove Invariant (12.12), we need to show $e \notin \mathbb{I}'$. This is true because by Claim 12.4.8 element e forms a circuit in $C_1(S' \cup N' \cup T', e)$, and by Invariant (12.10) we know $S' \cup N' \cup I' \in \mathcal{M}_1$. Hence, the circuit $C_1(S' \cup N' \cup T', e)$ contains some element of T', which gives the contradiction that C is non-empty.

Now WLOG, we can assume that element e forms a non-empty set C in Line 4. We prove Property ((i)), Invariant (12.10), and Invariant (12.11) by showing that they hold after any iteration of the for-loop of Line 5. Note that sets S', N', and $\mathbb{I}'$ can only change in Lines 8 or 10 of the for-loop. We prove the claim for both these cases.

Case 1 (Line 8): Since f belonged to T', from Observation 12.4.7 we know $(S' \cup N' \cup f) \in \mathcal{M}_1$. Now using Invariant (12.10) (which holds before the iteration), we can deduce that $C_1(S' \cup N' \cup I', f) \cap I'$ is non-empty and the update is well-defined. Invariant (12.10) holds because the update breaks any circuit in $S' \cup N' \cup \mathbb{I}'$ in $\mathcal{M}_1$. Since T and N' are unchanged and $\mathbb{I}'$ only gets smaller, Invariant (12.11) holds trivially.

Case 2 (Line 10): Since we are adding e to N', it must be the case that $S' \cup N' \cup e \in \mathcal{M}_1$ (by Lemma 12.4.1). If $C_1(S' \cup N' \cup I' \cup e)$ is non-empty then $C_1(S' \cup N' \cup \mathbb{I}' \cup e) \cap I'$ must be non-empty. Moreover, by Line 3, we know that $T \cup N' \cup e \in \mathcal{M}_2$. Hence, if $C_2(T \cup N' \cup I' \cup e)$ is non-empty then $C_2(T \cup N' \cup I' \cup e) \cap I'$ must be non-empty. Both of them together prove

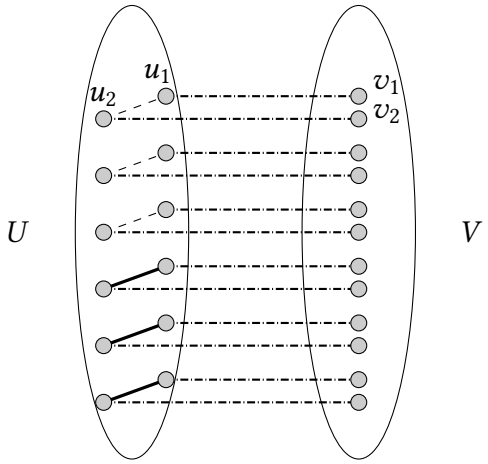

Figure 12.2: U denotes the set of vertices matched by Greedy in Phase (a) and V denotes the remaining vertices of G. Solid edges within U denote the picked edges and dashed edges within U denote the marked ones. Dashed edges from U to V denote the OPT edges.

the the update is well-defined in this case. Invariant (12.10) and Invariant (12.11) hold because Updates 12.4.3 break any circuit $C_1(S' \cup N' \cup \mathbb{I}' \cup e)$ and $C_2(T \cup N' \cup \mathbb{I}' \cup e)$.

Finally, to finish the proof of this claim, we show that Invariant (12.12) also holds at the end of every iteration of the for-loop of Line 2. If $e \notin \mathbb{I}'$ then Invariant (12.12) trivially holds as $\tilde{E}_r$ looses element e, and $\mathbb{I}' \subseteq \tilde{E}_r \setminus e$. Now suppose $e \in \mathbb{I}'$. Here we consider two cases.

Case 1 (e is added to N'): Here Samp-Alg reaches Line 10 and the special case of Update 12.4.3 ensures that e is removed from $\mathbb{I}'$. Hence $\mathbb{I}' \subseteq \tilde{E}_r$.

Case 2 (e is not added to N'): From the above proof, we know that Invariants (12.10) and (12.11) are preserved at the end of this iteration. We prove by contradiction and assume that $e \in I'$ at the end of this iteration. Since $e \notin N'$, all the elements of C in Line 4 are added to S' by the end of this iteration. Hence, the entire circuit in Line 4 (which is non-empty by Claim 12.4.8) is contained in $S' \cup N' \cup e$ at the end of the iteration. Since $e \in I'$, this implies that $S' \cup N' \cup I'$ is not independent. This is a contradiction as Invariant (12.10) is violated. □

12.5 General Graphs

Theorem 12.1.3. *In the random edge arrival model, the online matching problem for general graphs has a $(\frac{1}{2} + \delta')$-competitive randomized algorithm, where $\delta' > 0$ is a constant.*

Proof overview. Let G be the arrival graph with edge set E. Using the same idea as Lemma 12.2.1, we can again focus on graphs where Greedy has a competitive ratio of at most $\left(\frac{1}{2} + \epsilon\right)$ for any constant $\epsilon > 0$. We construct a two-phase algorithm that uses the algorithm from Theorem 12.1.2 as a subroutine. In Phase (a), we run Greedy; however, each edge selected by Greedy is picked only with probability $(1 - p)$. With probability p, we mark it along with its vertices and behave as if it has been matched for the rest of Phase (a). Since the hastiness property

(Lemma 12.2.3) is also true for general graphs, in expectation we pick $(1-p)\left(\frac{1}{2}-O(\frac{\epsilon}{f})\right)$ |OPT| edges and mark $p\left(\frac{1}{2}-O(\frac{\epsilon}{f})\right)$ |OPT| edges in Phase (a). Now we need to ensure that in expectation $(1+\gamma)$ edges, for some constant $\gamma > 0$, are picked per marked edge in Phase (b).

Let T_f denote the set of edges selected by Greedy in Phase (a), i.e., both picked and marked edges. Let U denote the set of vertices matched in T_f and V denote the remaining set of vertices of G. Using the following simple Fact 12.5.1 and Lemma 12.2.3, we can argue that $\left(1-O(\frac{\epsilon}{f})\right)$ OPT edges go from a vertex in U to a vertex in V in graph G.

Fact 12.5.1 (Lemma 1 in [KMM12]). *Consider a maximal matching T of graph G such that $|T| \leq \left(\frac{1}{2}+\epsilon\right)|OPT|$ for some $\epsilon \geq 0$. Then G contains at least $\left(\frac{1}{2}-3\epsilon\right)|OPT|$ vertex disjoint 3-augmenting paths with respect to T.*

Moreover, in expectation at most f fraction of these (U,V) OPT edges can appear in Phase (a). Thus, setting $\epsilon \ll f \ll 1$ gives that most of the OPT edges, i.e., $\left(1-O(\frac{\epsilon}{f})-f\right)$ fraction, appear in Phase (b). This implies that most of the marked edges contain two 3-augmentation edges as shown in Figure 12.2.

Now consider a marked edge (u_1,u_2) with (u_1,v_1) and (u_2,v_2) denoting its 3-augmentations. In comparison to bipartite graphs, the new concern in general graphs is that there might be an edge between u_1 and v_2 as triangles are possible in non-bipartite graphs. Hence, the Sampling Lemma 12.2.4 cannot be directly applied here. However, we are only interested in the bipartite graph between vertices U and V. Therefore, in Phase (b), we run the algorithm from Theorem 12.1.2 for bipartite graphs restricted to (U,V) edges. For sufficiently small values of constants ϵ and f, the constant δ gain in Theorem 12.1.2 is sufficient to obtain a constant δ' gain for this theorem. □

12.6 Miscellaneous Results and Missing Proofs

12.6.1 Table of Notation

We summarize the notation used in this chapter in Table 12.1.

12.6.2 Greedy Beats Factor of Half on Almost Regular Graphs

Theorem 12.6.1. *For online matching in random edge arrival model, Greedy has a competitive ratio of at least $(1-\frac{1}{e})$ on any d-regular graph.*

Proof. Consider a vertex v, and let $u_1,u_2,\ldots,u_d$ be its neighbours. The probability that (u_1,v) is the first to occur amongst all the edges of u_1 is exactly $\frac{1}{d}$. If this occurs, then we know that vertex v will be surely matched. Thus, the probability that v is not matched by the end of the algorithm is at most $(1-\frac{1}{d})^d \leq \frac{1}{e}$. This means that each vertex is matched with probability at least $1-\frac{1}{e}$, leading to the stated theorem. □

General Notation

$\mathcal{M}_i$	Matroid indexed by i
$A \in \mathcal{M}$	Subset A is an independent set in the matroid $\mathcal{M}$
$T \cup e$	Short form for notation $T \cup \{e\}$
$\mathsf{rank}_{\mathcal{M}}$	The rank function defined by matroid $\mathcal{M}$
$\overline{\imath}$	Denotes the index $3 - i$
$\mathcal{M}_1 \cap \mathcal{M}_2$	The set of subsets that are independent in both matroids $\mathcal{M}_1$ and $\mathcal{M}_2$
$\mathcal{M}/T$	The matroid resulting from contracting subset T in matroid $\mathcal{M}$
$\mathsf{span}_i(T)$	$\{e \mid (e \in E) \wedge (\mathsf{rank}_{\mathcal{M}_i}(T \cup \{e\}) = \mathsf{rank}_{\mathcal{M}_i}(T)\}$
$C_i(T \cup e)$	The unique circuit formed by $T \cup \{e\}$ in matroid $\mathcal{M}_i$. This is undefined when T is not an independent set and $e \notin \mathsf{span}_i(T)$.
E	The set of ground elements common to the matroids $\mathcal{M}_1$ and $\mathcal{M}_2$
π	A permutation on the set E
OPT	A fixed maximum independent set in the intersection of $\mathcal{M}_1 \cap \mathcal{M}_2$
$\mathcal{G}(f)$	$\mathbb{E}_\pi[\lvert T_f \rvert]/\lvert \text{OPT} \rvert$

Notation used by Marking-Greedy in §12.3.3

Ψ	The set of random bits used in the algorithm. For each $e \in E$, we have $\Psi(e) \sim Bern(1-p)$
selecting	The element is chosen by Greedy in Phase (a)
picking	The element is chosen by Marking-Greedy in the final solution
marking	The element is chosen by Greedy in Phase (a) but the algorithm does not pick it
T_f	The set of elements selected by Greedy in Phase (a)
S	The set of elements picked by Marking-Greedy in Phase (a)
N_i	The set of elements belonging to $\mathcal{M}_i/S \cap \mathcal{M}_{\overline{i}}/T$ picked by Marking-Greedy in Phase (b)

Table 12.1: Table of Notation

The same analysis also extends to graphs that are almost regular, i.e., graphs with vertex degrees between $d(1 \pm \epsilon)$, for any small constant ϵ.

12.6.3 Greedy Cannot Always Beat Half for Bipartite Graphs

Dyer and Frieze [DF91] showed a general graph[5] for which Greedy is half competitive. Inspired from their construction, we give the following bipartite graph for which Greedy is half competitive.

Definition 12.6.2 (Thick-$\mathcal{Z}$ graph). *Let graph Thick-$\mathcal{Z}$ $:= ((U_1 \cup U_2) \cup (V_1 \cup V_2)\}, E)$ be a bipartite graph with $|U_1| = |V_1|$ and $|U_2| = |V_2|$. The edge set E consists of the union of a perfect matching between U_i and V_i for $i \in \{1,2\}$ and a complete bipartite graph between U_2 and V_1. If additionally $|U_1| = |V_2|$, we call the graph a balanced Thick-$\mathcal{Z}$.*

Lemma 12.6.3. *When the edges of a balanced Thick-$\mathcal{Z}$ are revealed one-by-one in a random order to* Greedy *then in expectation it produces a matching of size $\left(\frac{1}{2} + o(1)\right)|OPT|$.*

Proof. We note that after an edge is picked by Greedy, both the end points of the edge do not participate later in the algorithm. Hence, at any instance during the execution of Greedy, the participating graph is still a Thick-$\mathcal{Z}$ graph $((U_1' \cup U_2') \cup (V_1' \cup V_2')\}, E')$, where $U_i' \subseteq U_i$ and $V_i' \subseteq V_i$ for $i \in \{1,2\}$.

We can view the choices made by Greedy as being done in time steps, where Greedy chooses one edge at each time step. At each time step, at least one of U_1 or U_2 decrease by 1, and Greedy halts when $|U_1'| = |U_2'| = 0$. Let t be the random variable indicating the first time step during the execution of Greedy when $\min\{|U_1'|, |U_2'|\} = n^{2/3}$. Let a, b be the random variables denoting $a := |U_1'| = |V_1'|$ and $b := |U_2'| = |V_2'|$ at time t. Let O_1 denote the number of edges of OPT chosen by Greedy before time t and let O_2 denote the number of edges of OPT chosen after time t.

We observe that the matching produced by Greedy is of size $\frac{n}{2} + |O_1| + |O_2|$. Observe $|\left(|U_1'| - |U_2'|\right)|$ changes only when Greedy chooses an edge from OPT, implying that we can bound $|a - b| \leq |O_1|$. Since O_2 is bounded by $|U_1'| + |U_2'|$ at time t, we can say

$$|O_2| \quad \leq \quad a + b \quad = \quad 2\,\min\{a,b\} + |a - b| \quad \leq \quad 2\,n^{2/3} + |O_1|.$$

Next, to bound $|O_1|$, we note that before time t the probability of an edge picked by Greedy being from OPT is at most $\frac{2n}{n^{2/3} \cdot n^{2/3}} = \frac{2}{n^{1/3}}$. Since Greedy picks at most n edges before time t, we have $\mathbb{E}[|O_1|] \leq \frac{2n}{n^{1/3}} = 2\,n^{2/3}$. This proves that expected size of the matching chosen by Greedy is $\frac{n}{2} + \mathbb{E}[|O_1| + |O_2|] \leq \frac{n}{2} + 2\,n^{2/3} + 2\,\mathbb{E}[|O_1|] \leq \frac{n}{2} + 6\,n^{2/3}$.

□

[5] This graph is popularly known as a *bomb graph*. It is obtained by adding a new vertex and edge adjacent to each vertex of a complete graph.

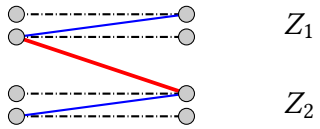

Figure 12.3: The above example is a conjunction of two Thick-$\mathcal{Z}$ graphs (Z_1 and Z_2) by a single edge (the thick red edge). Notice that for a Thick-$\mathcal{Z}$ graph even knowing the degree 2 vertex does not allow any algorithm to achieve more than $\frac{5}{3}$ edges in expectation.

12.6.4 Limitations on any OBME Algorithm

Lemma 12.6.4. *No randomized algorithm can achieve a competitive ratio greater than $\frac{5}{6} \sim 0.833$ for online bipartite matching in random edge arrival model when the graph is a balanced Thick-$\mathcal{Z}$ with $n = 1$. This is true, even the adversary knows the graph and can identify one vertex which has degree 2.*

Proof. The optimum offline matching size is two. However, no randomized online algorithm, (even one which knows the input graph), can obtain more than $\frac{5}{3}$ edges in expectation over the random edge order. To see this, let p denote the probability that the algorithm picks the first edge it sees.

Case 1: The first edge is from the optimal matching (i.e. the first edge is of the form (u_i, v_i) for $i \in \{1, 2\}$). In this case, the algorithm will achieve the optimal value 2 with probability p. If it skips one of these edges, it will retain at most 1 edge in the remaining graph.

Case 2: The first edge is not from the optimal matching (i.e. the first edge is (u_1, v_2)). In this case the algorithm will achieve a value of at most $1 \cdot p + 2 \cdot (1 - p)$.

Since Case 1 occurs with probability $\frac{2}{3}$ and Case 2 occurs with probability $\frac{1}{3}$, the expected value of the algorithm is $\frac{5}{3}p + \frac{4}{3}(1 - p) \leq \frac{5}{3}$. □

Lemma 12.6.5. *No randomized algorithm can achieve a competitive ratio greater than $\frac{69}{84} \sim 0.821$ for online bipartite matching in random edge arrival model.*

Proof. Our instance corresponds to the case where we take two copies of balanced Thick-$\mathcal{Z}$ graph joined by a single edge (see Figure 12.3). The input is some permutation of the graphs (where the vertices or edges may be permuted and U and V may be swapped). We show by case analysis that no algorithm can achieve a competitive ratio better than $\frac{69}{84} < \frac{5}{6}$. Intuitively, the addition of the single edge only hurts any algorithm without compromising the independence between the two instances.

Let p be the probability that the algorithm picks the first edge. Consider the following cases based on Figure 12.3:

Case 1: Suppose the first edge is the thick red edge. This occurs with probability $\frac{1}{7}$. If the algorithm picks this edge (which happens with probability p), then the optimal value in the remaining graph is 3. Otherwise, it can get at most $2 \cdot \frac{5}{3}$ as the two Thick-$\mathcal{Z}$ graphs are disjoint and we can use the previous lemma. Hence the expected outcome is $\frac{1}{7}\left(p \cdot 3 + (1 - p) \cdot \frac{10}{3}\right)$.

Case 2: Suppose the first edge is a blue edge, this occurs with probability $\frac{2}{7}$. If the algorithm chooses this edge, then we can get value of 1. Since this affords no information about the second Z, the best an algorithm can do is $\frac{5}{3}$. Hence the expected solution is $\frac{2}{7}\left(p(1+\frac{5}{3})+(1-p)(2+\frac{5}{3})\right)$.

Case 3: Suppose the first edge is a black edge. This occurs with probability $\frac{4}{7}$. If the algorithm chooses the first edge, then we can get value of 2 in this copy of the Thick-$\mathcal{Z}$. However, still the algorithms gets at most $\frac{5}{3}$ in the remaining copy of Thick-$\mathcal{Z}$. Hence the expected cost of the solution is $\frac{4}{7}\left(p(2+\frac{5}{3})+(1-p)(1+\frac{5}{3})\right)$

Adding these cases together, we get that expected solution has value at most $\frac{64+5p}{21}$. Since the optimal solution is 4, this gives an upper bound of $\frac{69}{84}$.

□

12.6.5 When Size of the Ground Set is Unknown

Theorem 12.6.6. *For any constant $\epsilon > 0$, any randomized algorithm $\mathcal{A}$ that does not know the number of edges to arrive has a competitive ratio $\alpha \leq \frac{2}{3}+\epsilon$ for online bipartite matching in random edge arrival model.*

Proof. To prove this theorem, we show that for any $\epsilon > 0$ there exists an instance where $\mathcal{A}$ is less than $\frac{2}{3}+\epsilon$-competitive.

Since $\mathcal{A}$ does not know the number of edges to arrive, it must maintain an α approximation in expectation after the arrival of every edge. This is because $\mathcal{A}$ does not know if the current edge will be the last edge.

Consider the instance given by the graph balanced Thick-$\mathcal{Z}$ (see Definition 12.6.2) where the size of the $|U_1| = |V_1| = N$ will be set later. Consider a random permutation π on the set of all edges and note that each edge $e$appears in the first T edges with probability $\frac{T}{N^2+2N}$, where $T = 4(N+2)\log N$. The previous probability is at least $\frac{4\log N}{N}$. Let G_T denote the set of edges from the perfect matching between U_i and V_i that appear in the first T edges. Let B_T denote the set of edges from U_2 to V_1 that appear in the first T edges. By linearity of expectation, we can say $\mathbb{E}[|G_T|] \leq 8\log N$ and $\mathbb{E}[|B_T|] \leq 4N\log N$.

Let OPT_T denote the expected size of the maximum matching on the graph induced by the first T edges.

Claim 12.6.7. $N(1-\epsilon) \leq \mathbb{E}_\pi[|OPT_T|]$

Proof. Consider the graph induced between U_2 and V_1 in the first T edges. Since any particular edge occurs with probability $\frac{4\log N}{N}$ and the edges are negatively correlated, we can conclude that

$$\Pr[\exists \text{a perfect matching between } U_2 \text{ and } V_1 \text{ in the first } T \text{ edges}] \geq \Pr[\exists \text{ a perfect matching in } \mathcal{G}_{N,N,\frac{4\log N}{N}}].$$

By a result of Erdos and Renyi (see [ER64]), we know that

$$\lim_{N\to\infty} \Pr[\exists \text{a perfect matching in } \mathcal{G}_{N,N,\frac{4\log N}{N}}] = 1$$

Hence, we can choose an N such that the above probability is at least $1-\epsilon$. Thus we can conlude that $\mathbb{E}[\text{OPT}_T] \geq N(1-\epsilon)$. □

Let M_{OPT} denote the expected number of edges picked by $\mathcal{A}$ that belong to the perfect matching between U_i and V_i (for $i = 1,2$) at time T. Similarly, let M_{Rest} denote the expected number of edges between U_2 and V_1 chosen by $\mathcal{A}$.

Since $\mathcal{A}$ must maintain an α approximation, we can say $M_{\text{OPT}} + M_{Rest} \geq \alpha(1-\epsilon)N$. Since $M_{\text{OPT}} \leq \mathbb{E}[|G_T|] = 8\log N \leq \alpha\epsilon N$, we can say

$$M_{Rest} \geq (\alpha - 2\epsilon)N \tag{12.14}$$

However, every edge chosen from M_{Rest} decreases the value of the optimal algorithm by one. Let F be the expected size of the matching chosen by the algorithm. We know that $\alpha \cdot 2N \leq F \leq 2N - M_{Rest}$. Substituting into (12.14) and dividing by $2N$, we get $\alpha \leq \frac{2}{3} + \epsilon$. □

12.6.6 Facts

Fact 12.2.7.

$$|T_1^\pi| \geq \frac{1}{2}\left(|\text{OPT}| + \sum_{e\in\text{OPT}} \mathbf{1}[\text{Both ends of } e \text{ matched in } T_f^\pi]\right) \text{ and}$$

$$|T_1^\pi| \geq |T_f^\pi| + \frac{1}{2}\sum_{e\in\text{OPT}} \mathbf{1}[\text{Both ends of } e \text{ unmatched in } T_f^\pi].$$

Proof. We start by counting the vertices matched in T_1^π,

$$2\,|T_1^\pi| \geq 2\sum_{e\in\text{OPT}} \mathbf{1}[\text{Both ends of } e \text{ matched in } T_1^\pi] + \sum_{e\in\text{OPT}} \mathbf{1}[\text{Exactly one end of } e \text{ matched in } T_1^\pi]$$

Since T_1^π is a maximal set,

$$|OPT| = \sum_{e\in\text{OPT}} \mathbf{1}[\text{Exactly one end of } e \text{ matched in } T_1^\pi] + \sum_{e\in\text{OPT}} \mathbf{1}[\text{Both ends of } e \text{ matched in } T_1^\pi]$$

Combining the previous two statements and the fact that $T_f^\pi \subseteq T_1^\pi$,

$$|T_1^\pi| \geq \frac{1}{2}\left(|\text{OPT}| + \sum_{e\in\text{OPT}} \mathbf{1}[\text{Both ends of } e \text{ matched in } T_f^\pi]\right).$$

To prove the second part, observe that $T_f^\pi \subseteq T_1^\pi$ and T_1^π is a maximal matching. For each edge of OPT that has both its end points unmatched in T_f^π, at least one end point is adjacent to an edge T_1^π. Since these edges must be part of $T_1^\pi \setminus T_f^\pi$,

$$|T_1^\pi| \geq |T_f^\pi| + \frac{1}{2}\sum_{e\in\text{OPT}} \mathbf{1}[\text{Both ends of } e \text{ unmatched in } T_f^\pi].$$

□

Fact 12.3.3. *Consider any matroid $\mathcal{M}$ and independent sets $A, B, C \in \mathcal{M}$ such that $A \subseteq \text{span}_{\mathcal{M}}(B)$ and $B \cup C \in \mathcal{M}$. Then we also have $A \cup C \in \mathcal{M}$.*

Proof. Suppose we start with $B \in M$ and add elements of $A = \{a_1, a_2, \ldots, a_k\}$ one by the one. We show that one can ensure that the set remains independent in $\mathcal{M}$ by removing some elements from B. First, note that $|B| = \text{rank}(B) = \text{rank}(B \cup A)$. Our algorithm removes an element from B only if addition of a_j creates a circuit. Hence the rank of the set is always $|B|$ and addition of every a_j creates a unique circuit. Moreover, this circuit contains an element $b_j \in B$ that can be removed as we know $A \in \mathcal{M}$.

Next we repeat the above procedure but by starting with $B \cup C \in \mathcal{M}$ and adding elements of A. We know from before that addition of each element a_j creates a unique circuit that does not contain an element of C. Hence we can remove element b_j while ensuring the set remains independent in $\mathcal{M}$. This will finally give $A \cup C \in \mathcal{M}$. □

12.6.7 Hastiness Lemma

The proof of the following lemma is similar to Lemma 2 in [KMM12].

Lemma 12.3.1 (Hastiness Lemma). *For any two matroids $\mathcal{M}_1$ and $\mathcal{M}_2$ on the same ground set E, let T_f^π denote the set selected by* Greedy *after running for the first f fraction of elements E appearing in order π. Also, for $i \in \{1,2\}$, let $\Phi_i(T_f^\pi) := \text{span}_i(T_f^\pi) \cap OPT$. Now for any $0 < f, \epsilon \leq \frac{1}{2}$, if $\mathbb{E}_\pi[|T_1^\pi|] \leq (\frac{1}{2} + \epsilon)\,|OPT|$ then*

$$\mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cap \Phi_2(T_f^\pi)|\right] \leq 2\epsilon\,|OPT| \qquad \textit{and} \tag{12.15}$$

$$\mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cup \Phi_2(T_f^\pi)|\right] \geq \left(1 - \frac{2\epsilon}{f} + 2\epsilon\right)|OPT|. \tag{12.16}$$

This implies $\mathcal{G}(f) := \frac{\mathbb{E}_\pi[|T_f^\pi|]}{|OPT|} \geq \left(\frac{1}{2} - \left(\frac{1}{f} - 2\right)\epsilon\right)$.

Proof. For ease of notation, we write T_f^π by T_f. To prove (12.15),

$$\begin{aligned}
\mathbb{E}_\pi\left[|\Phi_1(T_f) \cap \Phi_2(T_f)|\right] &\leq \mathbb{E}_\pi\left[|\Phi_1(T_1) \cap \Phi_2(T_1)|\right] && \text{(because } T_f \subseteq T_1\text{)} \\
&= \mathbb{E}_\pi\left[(|\Phi_1(T_1)| + |\Phi_2(T_1)|) - |\Phi_1(T_1) \cup \Phi_2(T_1)|\right] \\
&= \mathbb{E}_\pi\left[|\Phi_1(T_1)| + |\Phi_2(T_1)| - |\text{OPT}|\right] && \text{(because } T_1 \text{ is a maximal solution)} \\
&\leq 2\,\mathbb{E}_\pi\left[|T_1|\right] - |\text{OPT}| && \text{(because } |T_1| \geq |\Phi_i(T_1)|\text{)} \\
&\leq 2\epsilon\,|\text{OPT}|.
\end{aligned}$$

Now to prove Eq (12.16), we first bound $|\Phi_1(T_f)| + |\Phi_2(T_f)|$. It is at least

$$\begin{aligned}
&|\text{OPT}| + \sum_{e \in \text{OPT}} \mathbf{1}[e \in \text{span}_1(T_f) \cap \text{span}_2(T_f)] - \sum_{e \in \text{OPT}} \mathbf{1}[e \notin \left(\text{span}_1(T_f) \cup \text{span}_2(T_f)\right)] \\
&\geq |\text{OPT}| + \sum_{e \in \text{OPT}} \mathbf{1}[e \in T_f] - \sum_{e \in \text{OPT}} \mathbf{1}[e \notin \text{span}_1(T_f) \cup \text{span}_2(T_f)]. \quad \text{(because } T_f \subseteq \text{span}_i(T_f)\text{)}
\end{aligned}$$

Taking expectations and using Claim 12.6.8,

$$\mathbb{E}_\pi[|\Phi_1(T_f)| + |\Phi_2(T_f)|] \geq |\text{OPT}| - \left(\frac{1}{f} - 2\right)\mathbb{E}_\pi[|T_f \cap \text{OPT}|] \tag{12.17}$$

Since $f \leq \frac{1}{2}$, we can use an upper bound on $\mathbb{E}_\pi[|T_f \cap \text{OPT}|]$. Observe $T_1 \supseteq T_f$ is a maximal solution implying $|T_1| \geq |T_1 \cap \text{OPT}| + \frac{1}{2}(|\text{OPT}| - |T_1 \cap \text{OPT}|) \geq \frac{1}{2}(|\text{OPT}| + |T_f \cap \text{OPT}|)$. Taking expectations,

$$\mathbb{E}_\pi[|T_f \cap \text{OPT}|] \leq 2\,\mathbb{E}_\pi\left[|T_1| - \frac{1}{2}|\text{OPT}|\right] \leq 2\,\epsilon\,|\text{OPT}|. \qquad \left(\text{because } \mathbb{E}_\pi[|T_1|] \leq \left(\tfrac{1}{2} + \epsilon\right)|\text{OPT}|\right)$$

Combining this with (12.17) and (12.15) proves (12.16),

$$\begin{aligned}\mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cup \Phi_2(T_f^\pi)|\right] &= \mathbb{E}_\pi[|\Phi_1(T_f)| + |\Phi_2(T_f)|] - \mathbb{E}_\pi\left[|\Phi_1(T_f^\pi) \cap \Phi_2(T_f^\pi)|\right] \\ &\geq \left(1 - \frac{2\epsilon}{f} + 2\epsilon\right)|\text{OPT}|.\end{aligned}$$

Finally, using (12.17) and $|T_f| \geq |\Phi_i(T_f)|$, we also have $\mathbb{E}_\pi[|T_f^\pi|] \geq \frac{1}{2}\mathbb{E}_\pi[|\Phi_1(T_f)| + |\Phi_2(T_f)|] \geq \left(\frac{1}{2} - \left(\frac{1}{f} - 2\right)\epsilon\right)|\text{OPT}|$. □

For intuition, imagine the following claim for $f = \frac{1}{2}$, where it says that for a uniformly random order probability that e is in not in the span of T_f for either of the matroids is at most the probability e is selected by Greedy into T_f.

Claim 12.6.8. *Suppose $\mathcal{G}(1) \leq \left(\frac{1}{2} + \epsilon\right)|OPT|$ for some $\epsilon < \frac{1}{2}$ and T_f is the output of* Greedy *on $E([1, mf)]$, then*

$$\forall e \in OPT \qquad \Pr_\pi\left[e \notin \Phi_1(T_f) \wedge e \notin \Phi_2(T_f)\right] \leq \left(\frac{1}{f} - 1\right)\Pr_\pi[e \in T_f].$$

Proof. Let us define the event $\mathcal{X} = \left(e \notin \Phi_1(T_f) \wedge e \notin \Phi_2(T_f)\right) \vee (e \in T_f)$. Consider the mapping g from permutations to permutations. If e occurs in the first f fraction of elements then $g(\pi) = \pi$. If not, then remove e and insert it uniformly at randomly at a position in $[1, mf]$. This induces a mapping from the set of all permutations on the ground elements to the set of permutations that have e in the first f fraction of elements. The important observation is that the set of permutations satisfying the event $\mathcal{X}$ still satisfy the event under the mapping g. We can conclude that $\Pr[\mathcal{X}] \leq \Pr[\mathcal{X} \mid e \in [1, mf]]$. Conditioned on the event that $e \in [1, fm]$, event $\mathcal{X}$ means $e \in T_f$. This is because if $e \notin \Phi_1(T_f) \wedge e \notin \Phi_2(T_f)$ and $e \in E[1, fm]$ then $T_f \cup e \in \mathcal{M}_1 \cap \mathcal{M}_2$. Thus, we can conclude that $\Pr[\mathcal{X}] \leq \Pr[e \in T_f \mid e \in [1, mf]] = \frac{1}{f}\Pr[e \in T_f]$. Moreover, since $\left(e \notin \Phi_1(T_f) \wedge e \notin \Phi_2(T_f)\right)$ and $(e \in T_f)$ are disjoint events, $\Pr[\mathcal{X}] = \Pr\left[\left(e \notin \Phi_1(T_f) \wedge e \notin \Phi_2(T_f)\right)\right] + \Pr[e \in T_f]$, which proves this claim. □

Part IV

Conclusions

Chapter 13

Further Directions for Probing and Stopping-Time Algorithms

In this chapter we discuss several further directions for the probing and stopping-time models described in the previous chapters.

13.1 How to Find a Car Parking

Consider a scenario where you want to go to a restaurant for dinner. The problem is that the restaurant is located at a popular location/site where finding a car parking is difficult. Based on your prior experiences, suppose you know the probability of finding parking at different potential sites in that area. Since different parking sites incurs different costs (e.g., based on parking payment or the walking distance from the site to the restaurant), what strategy should you adopt to minimize your total *disutility*: sum of the distance that you drive and the parking cost. Clearly, one policy is to cancel your dinner plans and remain parked at your home. However, this incurs a a large disutility that you would like to reduce. Formally, we define this problem as follows.

The Car Parking Problem Given a graph $G = (V, E)$ with a metric (V, d) and a starting node $r \in V$, suppose each node $i \in V$ of the graph contains an empty parking spot independently with probability p_i (where $p_r = 1$). If i is empty and we park the car at i then we incur a cost c_i. The problem is to find a walk P starting at r, that minimizes the total expected disutility, $\mathbb{E}\left[\min_{i \in P}\{c_i\} + d(P)\right]$, where $d(P)$ is the length of the walk P.

The above car parking problem is similar to the disutility-minimization variant of the Pandora's box problem discussed in §3.1.1. The difference is that the probing prices are now given by an underlying metric. This inspires us to study more general variants of the car parking problem where rather than finding one empty parking lot, you need to find k empty parking sites. We call this the *k-car parking problem*. In an ongoing work, we show the following.

Theorem 13.1.1. *There exists an $O(k)$ approximation strategy for k-car parking.*

This theorem is interesting when k is small, but leaves an interesting open question.

Question 13.1.2. *Does there exist an $O(1)$-approximation algorithm for the k-car parking problem?*

The above question is also related the question of bounding adaptivity gaps beyond submodular functions from Chapter 6. A non-adaptive algorithm for the k-parking problem fixes its entire path independent of the actual instantiation of the parking sites. This raises the question of designing an $O(1)$-approximation non-adaptive algorithm.

Question 13.1.3. *Is the adaptivity gap for the k-car parking problem bounded by $O(1)$?*

13.2 Learning Probability Distributions for Probing

In both the PoI and the CoSP probing models considered in Part II, we assume the probability distributions of all input parameters is exactly known. This is a strong assumption because in practice we need to learn the probability distributions from samples. Since one can never learn the distributions exactly from a finite number of samples, this raises the question of designing algorithms that robust to noise. We model these issues in the following two ways.

(A) **Two-Phase Learning**: In this model the problem contains two-phases: a *learning phase* and an *optimization phase*. Given an $\epsilon > 0$, the problem is to use the minimum number of samples in the first phase to design a strategy which when used in the second-phase gives at most an ϵ-additive loss to the optimal strategy that knows the distributions. In Chapter 3 we saw how the results in the PoI model are robust and can achieve their guarantees given only a polynomial number of samples. However, this question is still open in the CoSP model.

Question 13.2.1. *Do* $\text{poly}(n)$ *samples suffice for two-phase learning in the CoSP model?*

(B) **Simultaneous Exploration and Exploitation**: This model has the classical exploration vs. exploitation tradeoff where the learning and the optimization phases operate together. For example, consider the Pandora's box problem where every day we play an instance of this problem drawn from the same underlying *unknown* value distribution. Suppose the probing prices are known, what strategy should we adopt to maximize the sum of our utility over T days? We define *regret* to be the difference of the total utility over T days obtained by the optimum strategy that knows all probability distributions starting from day one and the total utility of our algorithm.

Question 13.2.2. *Can we obtain sublinear regret, i.e., $o(T)$, for the Pandora's box problem in the simultaneous exploration and exploitation model?*

Similar questions are also open in the CoSP model, e.g., the Best-box problem.

13.3 Beyond Independent Probability Distributions

The probing models considered in Part II assume that the probability distributions of different elements behave independently. This is not without loss of generality and for many settings we would like our model to capture element correlations.

(A) **Hidden Markov Models**
In Chapter 4 we use the Markovian PoI model to capture correlations in the evolution of a given element. We now use a similar Markov model to capture correlations between elements.

Consider the disutility-minimization Pandora's box problem along with a hidden random state S from §3.1.1. Suppose S takes a value from a known probability distribution (e.g., S can be "high" or "low" w.p. half each). Conditioned on the value of S, the r.v.s $\{X_i\}_{i\in[n]}$ corresponding to the cost of each box take values independently from known distributions. This is also known as the Bayesian network model (see Figure 13.1). The algorithm is not allowed to probe the hidden state S; however, probing different boxes gives a better posterior on the value of S. What probing strategy should we adopt to minimize the expected disutility, which is the sum of total probing price and the cost of the box that we select.

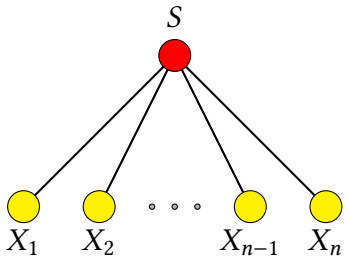

Figure 13.1: Random variables $X_1,\ldots,X_n$ are independent conditioned on hidden variable S.

Question 13.3.1. *Can we design optimal/$O(1)$-approximation strategy for the disutility-minimization Pandora's box problem in the Hidden Markov model?*

(B) **Common Base-Value and Linear Distribution Models**
One way of capturing mild positive correlations between random variables is using the *common base-value* model of Chawla et. al [CMS15]. In this model, n random variables $X_1,\ldots,X_n$ have a common base-value distribution if there exist $n+1$ *independent* non-negative random variables $Y_0, Y_1, \ldots, Y_n$ such that $X_i = Y_0 + Y_i$. In [BDHS15], Bateni et. al generalized the common base value model. A probability distribution on n random variables $X_1,\ldots,X_n$ is a *linear distribution* if there exist non-negative independent random variables $Y = \{Y_1, Y_2, \ldots., Y_k\}$ and a $k\times n$ matrix M with non-negative values such that $X_i = \langle M_i, Y\rangle$ is the dot product of the i'th column M_i of M and the random vector Y.

In some preliminary work, we design $O(1)$-approximation prophet inequalities for common base-value distributions. This raises the following question.

Question 13.3.2. *Can we design $O(1)$-approximation algorithms for single-item prophet inequalities or utility-maximization Pandora's box problem in the linear distribution model?*

13.4 Prophet Inequalities from Samples

The prophet inequalities designed in Chapter 8 assume that we exactly know the input distributions. Because of the concerns raised in §13.2, we want to design robust algorithms that give

good *multiplicative* approximation results even with a small number of samples. Intuitively, it might appear that such guarantees are not possible because in most learning algorithms we only obtain *additive* guarantees with $O(1)$-samples. A surprising result due to Azar, Kleinberg, and Weinberg [AKW14] shows that for single item prophet inequalities, one can obtain an e-approximation algorithm using only one sample from each distribution.

In [AKW14] the authors show a strong connection between the secretary and prophet inequality models. They proved that any α-approximation "order oblivious" secretary algorithm can be converted to obtain an α-approximation prophet inequality. As a corollary, this implies $O(1)$ prophet inequalities for several matroid constraints where we know $O(1)$-approximation secretary algorithms. Since such results are not known for general matroids, this raises the following interesting question.

Question 13.4.1. *Can we design $O(1)$-approximation matroid prophet inequality using only $O(1)$ independent samples from each distribution?*

In preliminary work along this direction, we observe that the "median based" 2-approximation prophet inequality algorithm due to Samuel-Cahn [SC84] for *single item* is robust: $O(1/\epsilon^2)$-samples from each distribution suffice to improve the e-approximation in [AKW14] to a $(2+\epsilon)$-approximation single item prophet inequality. Can we generalize this median based single item algorithm to general matroids?

13.5 Orienteering Secretary and Prophet Inequality Problems

Consider another example of a combinatorial optimization problem under uncertainty. Suppose you run a fedex service. Every day you receive a sequence of requests from a subset of the houses in the city to pick up their package, and how much they are willing to pay you for their pickup. On receiving a request, you have to immediately and irrevocably decide whether to accept the request, while always ensuring that the set of accepted offers can be visited between 9 am to 5 pm the following day. The goal is to maximize the sum of the payments of the accepted offers.

In the deterministic case where we know all the requests and their payment values, the above problem is exactly the *orienteering problem*. It is known that this problem is **NP**-hard, but $O(1)$-approximation efficient algorithms are possible [BCK+07]. In an uncertain environment, this problem can be modeled in both the secretary and the prophet inequality models. Since orienteering is a packing constraint, we know from the work of Rubinstein [Rub16] that $O(\log n \cdot \log r)$-approximation algorithms are information theoretically possible in both these stopping-time models. Unfortunately, these algorithms are not efficient. In an ongoing work we show that one cannot obtain $o(\log n/\log\log n)$-approximation for the orienteering secretary or the orienteering prophet inequality problems. However, it is not known if there exists an efficient algorithm that achieves this bound. Ideally, we would like to obtain the following black-box reduction.

Question 13.5.1. *Given access to an α-approximation oracle for linear function maximization*

over some packing constraint $\mathcal{F}$ (e.g., orienteering), can we design an efficient $\alpha \cdot \operatorname{poly}\log(n)$*-approximation algorithm for the same problem in the secretary/prophet-inequality model?*

13.6 Improving Approximation and Hardness Results

In this section we list a few problems where we do not know the tight approximation factors. First, consider probing algorithms.

(A) Consider the shortest-path problem in the PoI model. Given a graph $G = (V, E)$ where each edge e has a stochastic cost X_e that can be found by paying a probing price π_e. For given nodes $s, t \in V$, what strategy should we adopt to find a path from s to t while minimizing the length of the path plus the total probing price. Can we get an $O(1)$-approximation for this problem?

(B) Consider House-Purchasing in the CoSP model (Definition 1.3.2). Using techniques from §5 we can design an $O(1)$-approximation algorithm for this problem. (Also, see [HFX18].) Is it possible to find the optimal policy in polynomial time or prove that the problem is hard?

(C) In §13.1 we give an $O(1)$-approximation algorithm for the 1-car parking problem. Can we show that finding the optimal strategy is hard?

Next, we list some open problems for the stopping-time models.

(A) The *online submodular welfare* problem is a generalization of online bipartite matching where items arrive one-by-one and we have to immediately and irrevocably allocate them to users with different submodular valuations. The goal is to maximize the sum of the user valuations. A simple argument shows that the greedy algorithm that allocates the item to the user with largest marginal value is $1/2$-competitive. Can we show that this algorithm is $(1 - 1/e)$-competitive for random arrival order of the items? See [KMZ15].

(B) In Corollary 12.1.2 we gave a $(1/2+\epsilon)$-competitive algorithm for online bipartite matching in the random edge arrival model, where $\epsilon < 10^{-4}$. Can we obtain a significantly larger constant? Also, can we show that for adversarial edge arrival one cannot beat half?

(C) The important question of obtaining an $O(1)$-competitive algorithm for the *matroid secretary* problem is open. In §13.4 we discussed how secretary algorithms can be used to design a robust prophet inequality. Can we show the converse and use a robust matroid prophet inequality algorithm to obtain a matroid secretary algorithm?

Bibliography

[ACK18] Yossi Azar, Ashish Chiplunkar, and Haim Kaplan. Prophet secretary: Surpassing the 1-1/e barrier. In *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*, pages 303–318, 2018. 10.1.1, 10.1.2

[Ada11] Marek Adamczyk. Improved analysis of the greedy algorithm for stochastic matching. *Inf. Process. Lett.*, 111(15):731–737, 2011. 3.1.3, 5.1.2, 5.1.3

[ADFS95] Jonathan Aronson, Martin Dyer, Alan Frieze, and Stephen Suen. Randomized greedy matching. II. *Random Structures & Algorithms*, 6(1):55–73, 1995. 12.1.1, 12.1.3

[ADSY12] Shipra Agrawal, Yichuan Ding, Amin Saberi, and Yinyu Ye. Price of correlations in stochastic optimization. *Operations Research*, 60(1):150–162, 2012. 6.2, 9.1.2, 9.2

[AEE+17] Melika Abolhasani, Soheil Ehsani, Hossein Esfandiari, MohammadTaghi Hajiaghayi, Robert Kleinberg, and Brendan Lucier. Beating 1-1/e for ordered prophets. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 61–71. ACM, 2017. 10.1.1, 10.1.2, 10.3, 11.1.1

[AGKM11] Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1253–1264. SIAM, 2011. 12.1

[AGM15] Marek Adamczyk, Fabrizio Grandoni, and Joydeep Mukherjee. Improved approximation algorithms for stochastic matching. In *Algorithms-ESA 2015*, pages 1–12. Springer, 2015. 3.1.3, 5.1.3

[AHL12] Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. Online prophet-inequality matching with applications to ad allocation. In *ACM Conference on Electronic Commerce, EC '12, Valencia, Spain, June 4-8, 2012*, pages 18–35, 2012. 8.1.2

[AKW14] Pablo Daniel Azar, Robert Kleinberg, and S. Matthew Weinberg. Prophet inequalities with limited information. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1358–1377, 2014. 8.1.2, 13.4, 13.4

[Ala11] Saeed Alaei. Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 512–

521, 2011. 8.1.2

[Ala14] Saeed Alaei. Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. *SIAM Journal on Computing*, 43(2):930–972, 2014. 8.1, 8.2.1, 11.1

[AM06] Lawrence M Ausubel and Paul Milgrom. The lovely but lonely vickrey auction. *Combinatorial auctions*, 17:22–26, 2006. 8.1, 11.1

[AMM+11] Yossi Azar, Aleksander Madry, Thomas Moscibroda, Debmalya Panigrahi, and Aravind Srinivasan. Maximum bipartite flow in networks with adaptive channel width. *Theor. Comput. Sci.*, 412(24):2577–2587, 2011. 5.5.4

[AN16] Arash Asadpour and Hamid Nazerzadeh. Maximizing stochastic monotone submodular functions. *Management Science*, 62(8):2374–2391, 2016. 5.1.1, 5.1.3, 13.6

[ANS08] Arash Asadpour, Hamid Nazerzadeh, and Amin Saberi. Stochastic submodular maximization. In *International Workshop on Internet and Network Economics*, pages 477–489. Springer, 2008. Full version appears as [AN16]. 3.1.3, 5.1.2

[AR12] Itai Ashlagi and Alvin E. Roth. New challenges in multihospital kidney exchange. *American Economic Review*, 102(3):354–59, 2012. 5.1.3

[ASW14] Marek Adamczyk, Maxim Sviridenko, and Justin Ward. Submodular stochastic probing on matroids. In *STACS*, pages 29–40, 2014. 3.1.3, 5.1.2, 5.1.3

[BCK+07] Avrim Blum, Shuchi Chawla, David R Karger, Terran Lane, Adam Meyerson, and Maria Minkoff. Approximation algorithms for orienteering and discounted-reward tsp. *SIAM Journal on Computing*, 37(2):653–670, 2007. 13.5

[BCK12] Anand Bhalgat, Tanmoy Chakraborty, and Sanjeev Khanna. Mechanism design for a risk averse seller. In *Internet and Network Economics - 8th International Workshop, WINE 2012, Liverpool, UK, December 10-12, 2012. Proceedings*, pages 198–211, 2012. 9.1.2

[BCN+15] Alok Baveja, Amit Chavan, Andrei Nikiforov, Aravind Srinivasan, and Pan Xu. Improved bounds in stochastic matching and optimization. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015, August 24-26, 2015, Princeton, NJ, USA*, pages 124–134, 2015. 3.1.3, 5.1.3

[BDF+12] Ashwinkumar Badanidiyuru, Shahar Dobzinski, Hu Fu, Robert Kleinberg, Noam Nisan, and Tim Roughgarden. Sketching valuation functions. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1025–1035. SIAM, 2012. 6.5

[BDHS15] MohammadHossein Bateni, Sina Dehghani, MohammadTaghi Hajiaghayi, and Saeed Seddighin. Revenue maximization for selling multiple correlated items. In *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, pages 95–105, 2015. 13.3

[Ber95] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995. 1.5.3

[BFNS14] Niv Buchbinder, Moran Feldman, Joseph Seffi Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1433–1452. SIAM, 2014. 2.4.3, 5.4.3

[BFS15] Niv Buchbinder, Moran Feldman, and Roy Schwartz. Online submodular maximization with preemption. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1202–1216. SIAM, 2015. 9.1.3

[BG96] Ann Becker and Dan Geiger. Optimization of pearl's method of conditioning and greedy-like approximation algorithms for the vertex feedback set problem. *Artificial Intelligence*, 83(1):167–188, 1996. 3.4.2

[BGK11] Anand Bhalgat, Ashish Goel, and Sanjeev Khanna. Improved approximation results for stochastic knapsack problems. In *SODA*, pages 1647–1665, 2011. 3.1.3, 5.1.3

[BGL+12] Nikhil Bansal, Anupam Gupta, Jian Li, Julián Mestre, Viswanath Nagarajan, and Atri Rudra. When LP Is the Cure for Your Matching Woes: Improved Bounds for Stochastic Matchings. *Algorithmica*, 63(4):733–762, 2012. 3.1.3, 5.1, 5.1.3

[BH11] Maria-Florina Balcan and Nicholas JA Harvey. Learning submodular functions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 793–802. ACM, 2011. 6.5

[BH16] Eric Balkanski and Jason D. Hartline. Bayesian budget feasibility with posted pricing. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, pages 189–203, 2016. 9.1.2

[BHZ13] MohammadHossein Bateni, Mohammad Taghi Hajiaghayi, and Morteza Zadimoghaddam. Submodular secretary problem and extensions. *ACM Trans. Algorithms*, 9(4):32, 2013. 10.1.1, 10.1.2

[BIK07] Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, secretary problems, and online mechanisms. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 434–443, 2007. 10.1.1, 10.1.2

[BM08] Benjamin Birnbaum and Claire Mathieu. On-line bipartite matching made simple. *ACM SIGACT News*, 39(1):80–87, 2008. 12.1.3

[BMKK14] Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Streaming submodular maximization: Massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 671–680. ACM, 2014. 9.1.3

[BN14] Nikhil Bansal and Viswanath Nagarajan. On the adaptivity gap of stochastic orienteering. In *IPCO*, pages 114–125, 2014. 3.1.3, 5.1.3

[BPR+16] Ashwinkumar Badanidiyuru, Christos H. Papadimitriou, Aviad Rubinstein, Lior Seeman, and Yaron Singer. Locally adaptive optimization: Adaptive seeding for monotone submodular functions. In *Proceedings of the Twenty-Seventh Annual*

ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016, pages 414–429, 2016. 9.1.2

[BSZ18] Domagoj Bradac, Sahil Singla, and Goran Zuzic. (Near) optimal adaptivity gaps for stochastic multi-value probing, Manuscript. 2018. 1.5.4

[BUCM12] Siddharth Barman, Seeun Umboh, Shuchi Chawla, and David L. Malec. Secretary problems with convex costs. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, pages 75–87, 2012. 10.1.1

[BYE81] Reuven Bar-Yehuda and Shimon Even. A linear-time approximation algorithm for the weighted vertex cover problem. *Journal of Algorithms*, 2(2):198–203, 1981. 3.4.2

[BYGNR98] Reuven Bar-Yehuda, Dan Geiger, Joseph Naor, and Ron M. Roth. Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and bayesian inference. *SIAM journal on computing*, 27(4):942–959, 1998. 3.4.2

[CCPV07] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 182–196. Springer, 2007. 2.4.2, 9.1.2, 9.2, 9.2.2

[CCPV11] Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011. 2.3.1, 5.1, 5.3.3, 7.2.5

[CCWZ14] T-H. Hubert Chan, Fei Chen, Xiaowei Wu, and Zhichao Zhao. Ranking on arbitrary graphs: Rematch via continuous LP with monotone and boundary condition constraints. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1112–1122, 2014. 12.1.3

[CFG+02] Moses Charikar, Ronald Fagin, Venkatesan Guruswami, Jon M. Kleinberg, Prabhakar Raghavan, and Amit Sahai. Query strategies for priced information. *J. Comput. Syst. Sci.*, 64(4):785–819, 2002. 3.1.3

[CFH+17] José Correa, Patricio Foncea, Ruben Hoeksma, Tim Oosterwijk, and Tjark Vredeveld. Posted price mechanisms for a random stream of customers. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 169–186. ACM, 2017. 10.1.1, 10.1.2, 10.1.2, 10.3, 5

[CGQ15] Chandra Chekuri, Shalmoli Gupta, and Kent Quanrud. Streaming algorithms for submodular function maximization. In *International Colloquium on Automata, Languages, and Programming*, pages 318–330. Springer, 2015. 9.1.3

[CHMS10] Shuchi Chawla, Jason D. Hartline, David L. Malec, and Balasubramanian Sivan. Multi-parameter mechanism design and sequential posted pricing. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 311–320, 2010. 8.1.2, 11.1

[CIK+09] Ning Chen, Nicole Immorlica, Anna R. Karlin, Mohammad Mahdian, and Atri

Rudra. Approximating matches made in heaven. In *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*, pages 266–278, 2009. 3.1.3, 5.1, 5.1.2, 5.1.3

[CK15] Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: Matchings, matroids, and more. *Mathematical Programming*, 154(1-2):225–247, 2015. 9.1.3

[CMS15] Shuchi Chawla, David L. Malec, and Balasubramanian Sivan. The power of randomness in bayesian optimal mechanism design. *Games and Economic Behavior*, 91:297–317, 2015. 13.3

[CP05] Chandra Chekuri and Martin Pál. A recursive greedy algorithm for walks in directed graphs. In *FOCS*, pages 245–253, 2005. 5.3.3, 5.5.1, 5.5.1, 5.5.1

[CQ16] Chandra Chekuri and Kent Quanrud. Fast approximations for matroid intersection. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, 2016. 12.1.1, 12.1.3

[CS11] D. Chakrabarty and C. Swamy. Facility location with client latencies: Linear programming based techniques for minimum latency problems. In *15th International Conference on Integer Programming and Combinatoral Optimization (IPCO)*, pages 92–103, 2011. 5.5.2

[Cun86] William H Cunningham. Improved bounds for matroid partition and intersection algorithms. *SIAM Journal on Computing*, 15(4):948–957, 1986. 12.1.3

[CVZ14] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM J. Comput.*, 43(6):1831–1879, 2014. 8.2, 8.2, 8.2.2, 8.3, 9.1.2, 9.3

[DEH+17] Sina Dehghani, Soheil Ehsani, MohammadTaghi Hajiaghayi, Vahid Liaghat, and Saeed Seddighin. Stochastic k-Server: How Should Uber Work? In *ICALP 2017*, 2017. 10.1.2

[DF91] Martin Dyer and Alan Frieze. Randomized greedy matching. *Random Structures & Algorithms*, 2(1):29–45, 1991. 12.1.3, 12.6.3

[DFKL17] Paul Dütting, Michal Feldman, Thomas Kesselheim, and Brendan Lucier. Prophet inequalities made easy: Stochastic optimization by pricing non-stochastic inputs. In *IEEE 58th Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, 14-17 October, 2017*, pages 540–551, 2017. 11.2

[DGV04] Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 208–217. IEEE, 2004. 1.5.3, 3.1.3, 5.1.2, 5.1.3

[DGV05] Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Adaptivity and approximation for stochastic packing problems. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005, Vancouver, British Columbia, Canada, January 23-25, 2005*, pages 395–404, 2005. 3.1.3, 5.1.3

[DHK+13] Nikhil R. Devanur, Zhiyi Huang, Nitish Korula, Vahab S. Mirrokni, and Qiqi Yan. Whole-page optimization and submodular welfare maximization with online bidders. In *ACM Conference on Electronic Commerce, EC '13, Philadelphia, PA, USA, June 16-20, 2013*, pages 305–322, 2013. 9.1.3

[DHK14] Amol Deshpande, Lisa Hellerstein, and Devorah Kletenik. Approximation algorithms for stochastic boolean function evaluation and stochastic submodular set cover. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1453–1466, 2014. 5.1, 5.1.3

[DJK13] Nikhil R Devanur, Kamal Jain, and Robert D Kleinberg. Randomized primal-dual analysis of ranking for online bipartite matching. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 101–107, 2013. 12.1, 12.1.3

[DNS10] Shahar Dobzinski, Noam Nisan, and Michael Schapira. Approximation algorithms for combinatorial auctions with complement-free bidders. *Mathematics of Operations Research*, 35(1):1–13, 2010. 11.1, 11.3.2, 11.3.2

[Dob07] Shahar Dobzinski. Two randomized mechanisms for combinatorial auctions. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Princeton, NJ, USA, August 20-22, 2007, Proceedings*, pages 89–103, 2007. 2.4.7, 6.5, 9.4

[Dov18] Laura Doval. Whether or not to open pandora's box. *Journal of Economic Theory*, 175:127–158, 2018. 3.1.3

[DTW03] Ioana Dumitriu, Prasad Tetali, and Peter Winkler. On playing golf with two balls. *SIAM Journal on Discrete Mathematics*, 16(4):604–615, 2003. 4.1, 4.1.1, 4.1.3, 4.1.4, 4.2, 4.2.1, 4.2.1, 4.2.2, 4.3.1, 4.3.1

[Dyn63] Eugene B Dynkin. The optimum choice of the instant for stopping a markov process. In *Soviet Math. Dokl*, volume 4, 1963. 1.3.2, 1.5.2, 10.1.1, 10.1.2

[Edm70] Jack Edmonds. Submodular functions, matroids, and certain polyhedra. *Combinatorial structures and their applications*, pages 69–87, 1970. 12.1.1

[EHKS18] Soheil Ehsani, Mohammad Hajiaghayi, Thomas Kesselheim, and Sahil Singla. Prophet secretary for combinatorial auctions and matroids. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2018. 1.5.4, 11.1

[EHLM17] Hossein Esfandiari, MohammadTaghi Hajiaghayi, Vahid Liaghat, and Morteza Monemizadeh. Prophet secretary. *SIAM Journal on Discrete Mathematics*, 31(3):1685–1701, 2017. 7.1.4, 8.1.2, 10.1.1, 10.1.2, 11.1, 11.1.1, 11.2, 11.5, 11.5.2

[ELMS11] Leah Epstein, Asaf Levin, Julián Mestre, and Danny Segev. Improved approximation guarantees for weighted matching in the semi-streaming model. *SIAM Journal on Discrete Mathematics*, 25(3):1251–1265, 2011. 12.1.3

[ELSW13] Leah Epstein, Asaf Levin, Danny Segev, and Oren Weimann. Improved bounds for online preemptive matching. In *30th International Symposium on Theoretical Aspects of Computer Science, STACS 2013*, pages 389–399, 2013. 12.1.1

[ER64] Paul Erdos and Alfred Renyi. On random matrices. *Magyar Tud. Akad. Mat. Kutató Int. Közl*, 8(455-461):1964, 1964. 12.6.5

[Fei09] Uriel Feige. On maximizing welfare when utility functions are subadditive. *SIAM Journal on Computing*, 39(1):122–142, 2009. 2.2.2, 2.4.2, 11.1

[FGL15] Michal Feldman, Nick Gravin, and Brendan Lucier. Combinatorial auctions via posted prices. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 123–135. Society for Industrial and Applied Mathematics, 2015. 8.1.2, 11.1, 11.1, 11.5, 11.5.2

[FI17] Moran Feldman and Rani Izsak. Building a good team: Secretary problems and the supermodular degree. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1651–1670. SIAM, 2017. 10.1.1

[FKM+05] Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Elsevier Theoretical Computer Science*, 348(2):207–216, 2005. 12.1.3

[FMV11] Uriel Feige, Vahab S Mirrokni, and Jan Vondrák. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011. 2.4.4, 2.4.5, 2.4.1, 5.1, 5.2, 5.4

[FNS11a] Moran Feldman, Joseph Naor, and Roy Schwartz. Improved competitive ratios for submodular secretary problems (extended abstract). In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17-19, 2011. Proceedings*, pages 218–229, 2011. 10.1.1

[FNS11b] Moran Feldman, Joseph Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 570–579. IEEE, 2011. 5.1

[FNW78] Marshall L. Fisher, George L. Nemhauser, and Laurence A. Wolsey. An analysis of approximations for maximizing submodular set functionsfi!?ii. In *Polyhedral combinatorics*, pages 73–87. Springer, 1978. 5.1, 7.2.5

[Fre75] David A. Freedman. On tail probabilities for martingales. *Annals of Probability*, 3:100–118, 1975. 6.5.8

[FSZ15] Moran Feldman, Ola Svensson, and Rico Zenklusen. A simple $O(\log \log(\text{rank}))$-competitive algorithm for the matroid secretary problem. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1189–1201, 2015. 10.1.2, 10.4

[FSZ16] Moran Feldman, Ola Svensson, and Rico Zenklusen. Online contention resolution schemes. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages

1014–1033, 2016. 1.5.2, 7.1.2, 7.3, 7.3.2, 7.3.3, 8.1.1, 8.1.2, 8.2, 8.2, 8.2.1, 8.2.2, 9.3, 9.3.4, 9.3

[FZ15] Moran Feldman and Rico Zenklusen. The submodular secretary problem goes linear. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 486–505. IEEE, 2015. 10.1.1, 10.1.2, 10.4

[GGLS08] Naveen Garg, Anupam Gupta, Stefano Leonardi, and Piotr Sankowski. Stochastic analyses for online combinatorial optimization problems. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 942–951. Society for Industrial and Applied Mathematics, 2008. 10.1.2

[GGM10] Ashish Goel, Sudipto Guha, and Kamesh Munagala. How to probe for an extreme value. *ACM Transactions on Algorithms (TALG)*, 7(1):12, 2010. 3.1.3

[GGW11] John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-armed bandit allocation indices*. John Wiley & Sons, 2011. 4.1.4

[GHK+14] Oliver Göbel, Martin Hoefer, Thomas Kesselheim, Thomas Schleiden, and Berthold Vöcking. Online independent set beyond the worst-case: Secretaries, prophets, and periods. In *International Colloquium on Automata, Languages, and Programming*, pages 508–519. Springer, 2014. 10.1.2

[GJ74] John Gittins and David Jones. A dynamic allocation index for the sequential design of experiments. *Progress in statistics*, pages 241–266, 1974. 4.1.4

[GJSS18] Anupam Gupta, Haotian Jiang, Ziv Scully, and Sahil Singla. The markovian price of information, Manuscript. 2018. 1.5.4

[GK01] Anupam Gupta and Amit Kumar. Sorting and selection with structured costs. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 416–425. IEEE, 2001. 3.1.3

[GK10] Daniel Golovin and Andreas Krause. Adaptive submodularity: A new approach to active learning and stochastic optimization. In *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*, pages 333–345, 2010. 9.1.3

[GKMR11] Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, and R. Ravi. Approximation algorithms for correlated knapsacks and non-martingale bandits. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 827–836, 2011. 3.1.3, 5.1.3

[GKNR12] Anupam Gupta, Ravishankar Krishnaswamy, Viswanath Nagarajan, and R. Ravi. Approximation algorithms for stochastic orienteering. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1522–1538, 2012. 3.1.3, 5.1.2, 5.1.3

[GKS14] Daniel Golovin, Andreas Krause, and Matthew Streeter. Online submodular maximization under a matroid constraint with application to learning assignments. *arXiv preprint arXiv:1407.1082*, 2014. 9.1.3

[GM07] Sudipto Guha and Kamesh Munagala. Approximation algorithms for budgeted learning problems. In *STOC*, pages 104–113. 2007. Full version as: *Approximation*

Algorithms for Bayesian Multi-Armed Bandit Problems, http://arxiv.org/abs/1306.3525. 1.5.3, 3.1.3, 4.1.4, 5.1.3, 7.1.2, 7.3.1

[GM08] Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 982–991. Society for Industrial and Applied Mathematics, 2008. 10.1.2, 12.1, 12.1.1, 12.1.3

[GM09] Sudipto Guha and Kamesh Munagala. Multi-armed bandits with metric switching costs. In *Automata, Languages and Programming, 36th Internatilonal Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part II*, pages 496–507, 2009. 3.1.3, 5.1.3

[GM12] Sudipto Guha and Kamesh Munagala. Adaptive uncertainty resolution in bayesian combinatorial optimization problems. *ACM Transactions on Algorithms (TALG)*, 8(1):1, 2012. 3.1.3

[GMS07] Sudipto Guha, Kamesh Munagala, and Saswati Sarkar. Information acquisition and exploitation in multichannel wireless systems. In *IEEE Transactions on Information Theory*. Citeseer, 2007. 3.1.3

[GN13] Anupam Gupta and Viswanath Nagarajan. A stochastic probing problem with applications. In *Integer Programming and Combinatorial Optimization - 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18-20, 2013. Proceedings*, pages 205–216, 2013. 3.1.3, 5.1, 5.1.3

[GNR10] Anupam Gupta, Viswanath Nagarajan, and R. Ravi. Approximation algorithms for optimal decision trees and adaptive tsp problems. In *ICALP (1)*, pages 690–701, 2010. 5.5.2

[GNS16] Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Algorithms and adaptivity gaps for stochastic probing. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1731–1747. SIAM, 2016. 1.5.4, 5.1, 5.3

[GNS17] Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Adaptivity Gaps for Stochastic Probing: Submodular and XOS Functions. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1688–1702. SIAM, 2017. 1.5.4, 5.1

[GRST10] Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. Constrained non-monotone submodular maximization: Offline and secretary algorithms. In *International Workshop on Internet and Network Economics*, pages 246–257. Springer, 2010. 10.1.2

[GS17] Guru Prashanth Guruganesh and Sahil Singla. Online matroid intersection: Beating half for random arrival. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 241–253. Springer, 2017. 1.5.4, 10.1.2, 12.1

[GW95] Michel X Goemans and David P Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

3.4.2

[GX96] Harold N Gabow and Ying Xu. Efficient theoretic and practical algorithms for linear matroid intersection problems. *Journal of Computer and System Sciences*, 53(1):129–147, 1996. 12.1.3

[Har08] Nicholas J. A. Harvey. Matroid intersection, pointer chasing, and young's seminormal representation of S_n. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 542–549, 2008. 12.1.3

[Has96] Johan Hastad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 627–636. IEEE, 1996. 3.5.3

[HFX18] Jian Li Hao Fu and Pan Xu. A ptas for a class of stochastic dynamic programs. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2018*, 2018. 5.1.3, (B)

[HK73] John E Hopcroft and Richard M Karp. An $n\hat{5}/2$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on computing*, 2(4):225–231, 1973. 12.1.1

[HK+82] Theodore P Hill, Robert P Kertz, et al. Comparisons of stop rule and supremum expectations of iid random variables. *The Annals of Probability*, 10(2):336–345, 1982. 10.1.2, 10.3

[HK92] Theodore P Hill and Robert P Kertz. A survey of prophet inequalities in optimal stopping theory. *Contemp. Math*, 125:191–207, 1992. 8.2.2, 1

[HK12] Elad Hazan and Satyen Kale. Online submodular minimization. *Journal of Machine Learning Research*, 13(Oct):2903–2922, 2012. 9.1.3

[HKK16] Chien-Chung Huang, Naonori Kakimura, and Naoyuki Kamiyama. Exact and Approximation Algorithms for Weighted Matroid Intersection. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2016. 12.1.1, 12.1.3

[HKL15] Lisa Hellerstein, Devorah Kletenik, and Patrick Lin. Discrete stochastic submodular maximization: Adaptive vs. non-adaptive vs. offline. In *Algorithms and Complexity - 9th International Conference, CIAC 2015, Paris, France, May 20-22, 2015. Proceedings*, pages 235–248, 2015. 5.1.3

[HKP04] Mohammad Taghi Hajiaghayi, Robert D. Kleinberg, and David C. Parkes. Adaptive limited-supply online auctions. In *Proceedings 5th ACM Conference on Electronic Commerce (EC-2004), New York, NY, USA, May 17-20, 2004*, pages 71–80, 2004. 10.1.1, 10.1.2

[HKS07] Mohammad Taghi Hajiaghayi, Robert D. Kleinberg, and Tuomas Sandholm. Automated online mechanism design and prophet inequalities. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 58–65, 2007. 8.1, 8.1.2

[HKT00] Magnús M. Halldórsson, Jan Kratochvıl, and Jan Arne Telle. Independent sets with

domination constraints. *Discrete Applied Mathematics*, 99(1):39–54, 2000. 3.5.3

[HSS06] Elad Hazan, Shmuel Safra, and Oded Schwartz. On the complexity of approximating k-set packing. *Computational Complexity*, 15(1):20–39, 2006. 7.2.3

[INvdZ12] Sungjin Im, Viswanath Nagarajan, and Ruben van der Zwaan. Minimum latency submodular cover. In *ICALP*, pages 485–497, 2012. 5.5.2, 5.5.2

[Jen76] Thomas A. Jenkyns. The efficacy of the figreedyfi algorithm. In *Proc. of 7th South Eastern Conference on Combinatorics, Graph Theory and Computing*, pages 341–350, 1976. 3.4.1

[JMM+03] Kamal Jain, Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. *Journal of the ACM (JACM)*, 50(6):795–824, 2003. 3.4.2

[Kap13] Michael Kapralov. Better bounds for matchings in the streaming model. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1679–1697. SIAM, 2013. 12.1.3

[KH78] Bernhard Korte and Dirk Hausmann. An analysis of the greedy heuristic for independence systems. *Annals of Discrete Mathematics*, 2:65–74, 1978. 3.4.1

[KK03] Sampath Kannan and Sanjeev Khanna. Selection with monotone comparison costs. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 10–17. Society for Industrial and Applied Mathematics, 2003. 3.1.3

[KKT15] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11(4):105–147, 2015. 5.1

[Kle05] Robert D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005, Vancouver, British Columbia, Canada, January 23-25, 2005*, pages 630–631, 2005. 10.1.1, 10.1.2

[KMM12] Christian Konrad, Frédéric Magniez, and Claire Mathieu. Maximum matching in semi-streaming with few passes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 231–242. Springer, 2012. 12.1.1, 12.1.2, 12.1.3, 12.2.3, 12.2.2, 4, 12.3.2, 12.5.1, 12.6.7

[KMT11] Chinmay Karande, Aranyak Mehta, and Pushkar Tripathi. Online bipartite matching with unknown distributions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 587–596. ACM, 2011. 10.1.2, 12.1.1, 12.1.3

[KMZ15] Nitish Korula, Vahab Mirrokni, and Morteza Zadimoghaddam. Online submodular welfare maximization: Greedy beats 1/2 in random order. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 889–898. ACM, 2015. 9.1.3, 10.1.2, 12.1, 12.1.1, (A)

[KP09] Nitish Korula and Martin Pál. Algorithms for secretary problems on graphs and hypergraphs. In *International Colloquium on Automata, Languages and Programming*, pages 508–520. Springer, 2009. 10.1.2, 12.1.1, 12.1.3

[KPS14] Samir Khuller, Manish Purohit, and Kanthi K. Sarpatwar. Analyzing the opti-

mal neighborhood: Algorithms for budgeted and partial connected dominating set problems. In *SODA*, pages 1702–1713, 2014. 5.5.3

[KPV13] Michael Kapralov, Ian Post, and Jan Vondrák. Online submodular welfare maximization: Greedy is optimal. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1216–1225. Society for Industrial and Applied Mathematics, 2013. 9.1.3

[Kra13] Andreas Krause. Submodularity in machine learning and vision. In *British Machine Vision Conference, BMVC 2013, Bristol, UK, September 9-13, 2013*, 2013. 5.1

[KRTV13] Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *European Symposium on Algorithms*, pages 589–600. Springer, 2013. 10.1.2, 12.1.3

[KS77] Ulrich Krengel and Louis Sucheston. Semiamarts and finite values. *Bull. Am. Math. Soc*, 1977. 1.3.2, 8.1

[KS78] Ulrich Krengel and Louis Sucheston. On semiamarts, amarts, and processes with finite value. *Advances in Prob*, 4:197–266, 1978. 1.3.2, 1.5.2, 8.1

[KS07] Stavros G. Kolliopoulos and George Steiner. Partially ordered knapsack and applications to scheduling. *Discrete Applied Mathematics*, 155(8):889–897, 2007. 5.5.4

[KSS13] Pushmeet Kohli, Mahyar Salek, and Greg Stoddard. A fast bandit algorithm for recommendation to users with heterogenous tastes. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA.*, 2013. 9.1.2

[KV08] Bernhard Korte and Jens Vygen. *Combinatorial Optimization, Volume 21 of Algorithms and Combinatorics*. Springer-Verlag, Berlin,, 2008. 12.1.1, 12.3.1

[KVV90] Richard M Karp, Umesh V Vazirani, and Vijay V Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, pages 352–358, 1990. 12.1, 12.1.3

[KW12] Robert Kleinberg and S. Matthew Weinberg. Matroid prophet inequalities. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 123–136, 2012. 1.5.2, 8.1, 8.1.2, 8.1.2, 8.2.2, 8.4, 8.4.2, 11.1, 11.1.2, 11.2, 11.4

[KWW16] Robert D. Kleinberg, Bo Waggoner, and E. Glen Weyl. Descending price optimally coordinates search. In *Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16, Maastricht, The Netherlands, July 24-28, 2016*, pages 23–24, 2016. 1, 3.1.3, 3.2, 4.1, 4.1.4

[Lac14] Oded Lachish. O(log log rank) competitive ratio for the matroid secretary problem. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 326–335, 2014. 10.1.2, 10.4

[LB10] Brendan Lucier and Allan Borodin. Price of anarchy for greedy auctions. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*,

pages 537–553. Society for Industrial and Applied Mathematics, 2010. 11.4

[Led97] Michel Ledoux. On Talagrand's deviation inequalities for product measures. *ESAIM: Probability and Statistics*, 1:63–87, 1997. 9.4.8

[LMNS09] Jon Lee, Vahab S Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 323–332. ACM, 2009. 5.1

[LPRY08] Zhen Liu, Srinivasan Parthasarathy, Anand Ranganathan, and Hao Yang. Near-optimal algorithms for shared filter evaluation in data stream systems. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 133–146, 2008. 5.1, 5.1.3

[LS17] Euiwoong Lee and Sahil Singla. Maximum matching in the online batch-arrival model. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 355–367. Springer, 2017. 12.1

[LS18] Euiwoong Lee and Sahil Singla. Optimal online contention resolution schemes via ex-ante prophet inequalities. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-24, 2018, Helsinki, Finland*, 2018. 1.5.4, 8.1.1

[LSW15] Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A Faster Cutting Plane Method and its Implications for Combinatorial and Convex Optimization. In *Proceedings of the Fifty-Sixth Annual IEEE Symposium on Foundations of Computer Science*, 2015. 12.1.3

[LY13] Jian Li and Wen Yuan. Stochastic combinatorial optimization via poisson approximation. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 971–980, 2013. 3.1.3, 5.1.3

[Ma14] Will Ma. Improvements and generalizations of stochastic knapsack and multi-armed bandit approximation algorithms: Extended abstract. In *SODA*, pages 1154–1163, 2014. 3.1.3, 5.1.3

[Meh12] Aranyak Mehta. Online matching and ad allocation. *Theoretical Computer Science*, 8(4):265–368, 2012. 12.1.1, 12.1.3

[Mes06] Julián Mestre. Greedy in approximation algorithms. In *European Symposium on Algorithms*, pages 528–539. Springer, 2006. 2.3.1, 6.4.2

[Mey01] Adam Meyerson. Online facility location. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 426–431. IEEE, 2001. 10.1.2

[MHK+98] Nicolas Meuleau, Milos Hauskrecht, Kee-Eung Kim, Leonid Peshkin, Leslie Pack Kaelbling, Thomas L Dean, and Craig Boutilier. Solving very large weakly coupled markov decision processes. In *AAAI/IAAI*, pages 165–172, 1998. 1.5.3

[MSVV07] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized online matching. *Journal of the ACM (JACM)*, 54(5):22, 2007. 12.1

[MV15] Aranyak Mehta and Vijay Vazirani. Personal communication. 2015. 12.1.1

[MY11] Mohammad Mahdian and Qiqi Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing lps. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 597–606. ACM, 2011. 10.1.2, 12.1.1, 12.1.3

[OW15] Wojciech Olszewski and Richard Weber. A more general pandora rule? *Journal of Economic Theory*, 160:429–437, 2015. 3.1.3

[Oxl06] James G Oxley. *Matroid Theory*, volume 3. Oxford university press, 2006. 12.3.1

[Pow07] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007. 1.5.3

[PS12] Matthias Poloczek and Mario Szegedy. Randomized greedy algorithms for the maximum matching problem with new analysis. In *Proceedings of the Fifty-Third Annual IEEE Symposium on Foundations of Computer Science*, pages 708–717. IEEE, 2012. 12.1.3

[Rec05] András Recski. Maps of matroids with applications. *Discrete Mathematics*, 303(1):175–185, 2005. 12.1.3

[Rot07] Michael H Rothkopf. Thirteen reasons why the vickrey-clarke-groves process is not practical. *Operations Research*, 55(2):191–197, 2007. 8.1, 11.1

[RS17] Aviad Rubinstein and Sahil Singla. Combinatorial prophet inequalities. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1671–1687. SIAM, 2017. 1.5.4, 9.1.1, 10.1.1

[RSS15] Aviad Rubinstein, Lior Seeman, and Yaron Singer. Approximability of adaptive seeding under knapsack constraints. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC '15, Portland, OR, USA, June 15-19, 2015*, pages 797–814, 2015. 9.1.2

[RSÜ05] Alvin E. Roth, Tayfun Sönmez, and M.Ũtku Ünver. Pairwise kidney exchange. *J. Econom. Theory*, 125(2):151–188, 2005. 5.1.3

[Rub16] Aviad Rubinstein. Beyond matroids: secretary problem and prophet inequality with general constraints. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 324–332, 2016. 8.1.2, 9.1.1, 9.1.2, 9.4.2, 9.4, 10.1.1, 10.1.2, 1, 10.4, 13.5

[RW15] Aviad Rubinstein and S. Matthew Weinberg. Simple mechanisms for a subadditive buyer and applications to revenue monotonicity. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation, EC '15, Portland, OR, USA, June 15-19, 2015*, pages 377–394, 2015. 6.2, 6.2.2

[SB98] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998. 1.5.3

[SC84] Ester Samuel-Cahn. Comparison of threshold stop rules and maximum for independent nonnegative random variables. *the Annals of Probability*, pages 1213–1216, 1984. 13.4

[SC98] Satinder P Singh and David Cohn. How to dynamically merge markov decision

processes. In *Advances in neural information processing systems*, pages 1057–1063, 1998. 1.5.3

[Sch99] Gideon Schechtman. Concentration, results and applications, 1999. 6.2, 6.2.2

[Sch03] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003. 12.1.1

[SG08] Matthew Streeter and Daniel Golovin. An online algorithm for maximizing submodular functions. In *Advances in Neural Information Processing Systems*, pages 1577–1584, 2008. 9.1.3

[Sin18] Sahil Singla. The price of information in combinatorial optimization. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2018. 1.5.4, 3.1, 4.1.2, 4.1.4, 4.3.2, 7.1.1

[Tsi94] John N Tsitsiklis. A short proof of the gittins index theorem. *The Annals of Applied Probability*, pages 194–199, 1994. 4.1.4

[Web92] Richard Weber. On the gittins index for multiarmed bandits. *The Annals of Applied Probability*, 2(4):1024–1033, 1992. 4.1.3, 4.1.4

[Wei79] Martin L. Weitzman. Optimal search for the best alternative. *Econometrica: Journal of the Econometric Society*, pages 641–654, 1979. 1.3.1, 1.5.1, 3.1, 4.1, 4.4.1

[Whi80] Peter Whittle. Multi-armed bandits and the gittins index. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 143–149, 1980. 4.1.4

[WS11] David P. Williamson and David B. Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011. 3.4.2, 3.4.2, 3.4.2, 5.1

[WW15] Yajun Wang and Sam Chiu-wai Wong. Two-sided online bipartite matching and vertex cover: Beating the greedy algorithm. In *International Colloquium on Automata, Languages, and Programming*, pages 1070–1081. Springer, 2015. 12.1, 12.1.3

[Yan11] Qiqi Yan. Mechanism design via correlation gap. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 710–719. Society for Industrial and Applied Mathematics, 2011. 8.1.2, 9.1.2, 11.1