Delta-Complete Reachability Analysis (Part I)

Sicun Gao **Soonho Kong** **Edmund M. Clarke**

December 1, 2013
CMU-CS-13-131

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We give a new framework for safety verification of nonlinear hybrid systems, based on delta-decidability of first-order logic formulas over the real numbers. We use expressive logic formulas (which can contain nonlinear ODEs with no analytic solutions) to encode bounded model checking and invariant-based reasoning. Based on the encoding, we solve bounded reachability and invariant validation problems using delta-complete decision procedures. Such techniques allow us to take into account of robustness properties of a system under delta-bounded numerical perturbations. This report describes Part I of the work, focusing on basic definitions and bounded reachability problems.

This research was sponsored by the National Science Foundation grants no. CNS1330014, no. CNS0926181 and no. CNS0931985, the GSRC under contract no. 1041377, the Semiconductor Research Corporation under contract no. 2005TJ1366, and the Office of Naval Research under award no. N000141010188.

Keywords: Hybrid Systems, Reachability, Bounded Model Checking

1 Introduction

Formal verification is difficult for hybrid systems with nonlinear dynamics and complex discrete control [1, 9]. Few modern techniques from hardware and software verification have seen much success on hybrid systems, because these techniques are all highly dependent on scalable logic solvers. To apply them on hybrid systems, we have to solve logic formulas over the real numbers with (often a large number of) nonlinear functions, which is highly challenging both theoretically and practically.

In recent work [7, 6], we have shown that logic formulas over the real numbers become much easier to solve when we shift our focus from the standard decision problem to the *δ-decision problem*: Given an arbitrary positive rational number δ, we ask if a logic formula is false or *δ-true*. The latter answer can be given if the formula *would become true* under δ-bounded numerical perturbations on its constant terms. The δ-decision problem is decidable, with reasonable complexity, for bounded first-order sentences over the reals with arbitrary Type 2 computable functions, such as polynomials, trigonometric functions, and Lipschitz-continuous ODEs [17].

This series of reports describes how we use δ-decidability over the reals to develop a new framework for hybrid system verification.

First, δ-decidability results enable the use of an expressive first-order logic signature, which we denote as $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$, to represent general nonlinear hybrid systems. Here, $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$ allows the use of arbitrary Type 2 computable real functions, which, for instance, include nonlinear ODEs that only need to be numerically solvable. Almost all existing classes of hybrid systems that have been studied in the literature can be defined through restrictions on $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$.

Next, bounded model checking and invariant-based reasoning techniques for $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-representable hybrid systems are naturally expressed as decision problems for $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas. The key observation is that, when we shift to solving the δ-decision problem for these formulas, the verification results are not weakened. This motivates the definition of *δ-strengthened* versions of the verification techniques. For instance, with δ-strengthened bounded model checking, we always obtain one of the following answers:

- Safe (bounded): The system does not violate the safety property within a bounded time, and a bounded unrolling depth (for discrete mode changes).
- δ-Unsafe: Under some δ-perturbation on its $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-representation, the system would violate the safety property.

Thus, when the procedure returns "safe", it is a precise answer and no error is involved. On the other hand, when we choose a small enough δ, a system that is "δ-unsafe" exhibits robustness problems. Realistic hybrid systems interact with the physical world and it is impossible to avoid slight perturbations. Thus, under δ-perturbations and should indeed be regarded as unsafe. Note that such robustness problems can not be discovered by solving the precise decision problem. In short, the framework turns numerical errors into stronger verification results.

It follows from δ-decidability that δ-strengthened bounded reachability and invariant validation are computable for general nonlinear hybrid systems, which stands in sharp contrast to the standard undecidability of reachability of simple systems. Moreover, after bypassing the difficulties with

exact real computations, we gain a better understanding of intrinsic properties of hybrid systems. For instance:

- There exists a three-layer complexity hierarchy for bounded reachability, which depends on the use of mode invariants and nondeterministic flows.
- The search for sound and complete rules for exact checking of invariants is a major challenge, while switching to the δ-strengthened version allows a direct logical encoding.

This report focuses on basic definitions and theoretical results regarding bounded reachability.

2 $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-Representations of Hybrid Automata

2.1 $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-Formulas

We will use a logical language over the real numbers, written as $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$, that allows arbitrary *computable real functions*. Computability of real functions is a notion well-developed in Computable Analysis [17]. Intuitively, a real function is computable if it can be numerically simulated up to an arbitrary precision. For the purpose of this paper, it suffices to know that almost all the functions that are needed in describing hybrid systems are computable: polynomials, exponentiation, logarithm, trigonometric functions, and also the solution functions of Lipschitz-continuous ordinary differential equations. Compositions of computable functions are computable. This, as we will show, makes $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$ very powerful and can express almost any realistic hybrid system.

Formally, $\mathcal{L}_{\mathbb{R}_\mathcal{F}} = \langle \mathcal{F}, > \rangle$ represents the first-order signature over the reals with the set $\mathcal{F}$ of computable real functions, which contains all the functions mentioned above. Note that constants are included as 0-ary functions. $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas are evaluated in the standard way over the corresponding structure $\mathbb{R}_\mathcal{F} = \langle \mathbb{R}, \mathcal{F}^\mathbb{R}, >^\mathbb{R} \rangle$. It is not hard to see that we can put any $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formula in a normal form, such that its atomic formulas are of the form $t(x_1, ..., x_n) > 0$ or $t(x_1, ..., x_n) \geq 0$, with $t(x_1, ..., x_n)$ composed of functions in $\mathcal{F}$. This follows from the fact that $t(\vec{x}) = 0$ can be written as $-|t(\vec{x})| \geq 0$, $t(\vec{x}) < 0$ as $-t(\vec{x}) > 0$, and $t(\vec{x}) \leq 0$ as $-t(\vec{x}) \geq 0$. Also, negations in front of atomic formulas can be eliminated by replacing $\neg t(\vec{x}) > 0$ with $-t(\vec{x}) \geq 0$, and $\neg t(\vec{x}) \geq 0$ with $-t(\vec{x}) > 0$. To avoid extra preprocessing of formulas, we can explicitly define $\mathcal{L}_\mathcal{F}$-formulas as follows.

Definition 2.1 ($\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-Formulas)**.** *Let $\mathcal{F}$ be a collection of computable real functions. We define:*

$$t := x \mid f(t(\vec{x})), \text{ where } f \in \mathcal{F} \text{ (constants are 0-ary functions)};$$
$$\varphi := t(\vec{x}) > 0 \mid t(\vec{x}) \geq 0 \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \exists x_i \varphi \mid \forall x_i \varphi.$$

In this setting $\neg\varphi$ is regarded as an inductively defined operation which replaces atomic formulas $t > 0$ with $-t \geq 0$, atomic formulas $t \geq 0$ with $-t > 0$, switches $\wedge$ and $\vee$, and switches $\forall$ and $\exists$. Implication $\varphi_1 \to \varphi_2$ is defined as $\neg\varphi_1 \vee \varphi_2$.

For any $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formula φ with n free variables, we write $[\![\varphi]\!] = \{\vec{a} \in \mathbb{R}^n : \varphi(\vec{a}) \text{ is true over } \langle \mathbb{R}, \mathcal{F}^\mathbb{R}, >^\mathbb{R} \rangle\}$. If φ is a sentence (no free variables), we use the standard notation $\mathbb{R} \models \varphi$ to denote that φ is true over $\mathbb{R}$.

Definition 2.2 (Bounded Quantifiers). *The bounded quantifiers* $\exists^{[u,v]}$ *and* $\forall^{[u,v]}$ *are defined as*

$$\exists^{[u,v]}x.\varphi =_{df} \exists x.(u \leq x \wedge x \leq v \wedge \varphi),$$
$$\forall^{[u,v]}x.\varphi =_{df} \forall x.((u \leq x \wedge x \leq v) \rightarrow \varphi),$$

where u *and* v *denote* $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$ *terms, whose variables only contain free variables in* φ *excluding* x.

Definition 2.3 (Bounded $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-Sentences). *A* bounded $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-sentence *is*

$$Q_1^{[u_1,v_1]}x_1 \cdots Q_n^{[u_n,v_n]}x_n \ \psi(x_1, ..., x_n),$$

where $Q_i^{[u_i,v_i]}$ *are bounded quantifiers, and* $\psi(x_1, ..., x_n)$ *is a quantifier-free* $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$*-formula.*

2.2 δ-Perturbations and δ-Decidability

Definition 2.4 (δ-Variants). *Let* $\delta \in \mathbb{Q}^+ \cup \{0\}$*, and* φ *an* $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$*-formula of the form*

$$\varphi : \ Q_1^{I_1}x_1 \cdots Q_n^{I_n}x_n \ \psi[t_i(\vec{x}, \vec{y}) > 0; t_j(\vec{x}, \vec{y}) \geq 0],$$

where $i \in \{1, ...k\}$ *and* $j \in \{k+1, ..., m\}$. *The* δ-weakening φ^δ *of* φ *is defined as the result of replacing each atom* $t_i > 0$ *by* $t_i > -\delta$ *and* $t_j \geq 0$ *by* $t_j \geq -\delta$. *That is,*

$$\varphi^\delta : \ Q_1^{I_1}x_1 \cdots Q_n^{I_n}x_n \ \psi[t_i(\vec{x}, \vec{y}) > -\delta; t_j(\vec{x}, \vec{y}) \geq -\delta].$$

It is clear that $\varphi \rightarrow \varphi^\delta$ *(see [7]).*

In [7, 6], we have proved that the following δ-decision problem is decidable. This result serves as the basis of our framework.

Theorem 2.5 (δ-Decidability). *Let* $\delta \in \mathbb{Q}^+$ *be arbitrary. There is an algorithm which, given any bounded* φ*, correctly returns one of the following two answers:*

- *"*δ-True*":* φ^δ *is true.*
- *"*False*":* φ *is false.*

Note when the two cases overlap, either answer is correct.

We now turn to the complexity issues. Informally, a real function is (uniformly) P-computable (PSPACE-computable) over a compact domain, simply if it can be numerically computed within polynomial-time (polynomial-space). Details can be found in [14, 7]. It suffices to know that many common real functions are P-computable, which includes the polynomials, exp, log, sin, etc. The intuition is that they can be effectively approximated, for instance with Taylor expansions. It is also shown that the solution functions of P-computable Lipschitz-continuous differential equations are PSPACE-computable [14] (in fact, PSPACE-complete [13]).

To state the complexity of the δ-decision problems, we recall the definition of the relativized complexity classes and polynomial hierarchy. The polynomial hierarchy, relativized to a set A, is defined as $(\Sigma_0^\mathsf{P})^\mathsf{A} = (\Pi_0^\mathsf{P})^\mathsf{A} = \mathsf{P}^\mathsf{A}$, $(\Sigma_{k+1}^\mathsf{P})^\mathsf{A} = \mathsf{NP}^{(\Sigma_k^\mathsf{P})^\mathsf{A}}$, and $(\Pi_{k+1}^\mathsf{P})^\mathsf{A} = \mathsf{coNP}^{(\Sigma_k^\mathsf{P})^\mathsf{A}}$.

Theorem 2.6 (Complexity [7]). *Let* S *be a class of* $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$*-sentences, such that for any* φ *in* S*, the functions in* φ *are in complexity class* C*. Then, for any* $\delta \in \mathbb{Q}^+$*, the* δ*-decision problem for bounded* Σ_n*-sentences in* S *is in* $(\Sigma_n^\mathsf{P})^\mathsf{C}$.

2.3 Hybrid Automata with $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-Representations

Hybrid automata extend finite automata with continuous dynamics. We first show that $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas can be used as a concise and natural representation of general hybrid systems.

Definition 2.7 ($\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-Representation)**.** *A hybrid automaton in $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-representation is a tuple*

$$H = \langle X, Q, \{\mathsf{flow}_q(\vec{x}, \vec{x}_0, t) : q \in Q\}, \{\mathsf{inv}_q(\vec{x}) : q \in Q\}, \\ \{\mathsf{jump}_{q\to q'}(\vec{x}, \vec{x}') : q, q' \in Q\}, \{\mathsf{init}_q(\vec{x}) : q \in Q\}\rangle,$$

where $X \subseteq \mathbb{R}^n$ for some $n \in \mathbb{N}$, and $Q = \{q_1, ..., q_m\}$ is a finite set of modes, and the other components are sets of quantifier-free $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas.

Almost all hybrid systems studied in the existing literature can be defined by restricting the signature $\mathcal{F}$. For instance,

Example 2.8 (Linear and Polynomial Hybrid Automata)**.** *Let $\mathcal{F}^{\text{lin}} = \{+\} \cup \mathbb{Q}$ and $\mathcal{F}^{\text{poly}} = \{\times\} \cup \mathcal{F}^{\text{lin}}$ (Rational numbers are considered as 0-ary functions.) In existing literature, we say H is a* linear hybrid automaton *if it has an $\mathcal{L}_{\mathbb{R}_{\mathcal{F}^{\text{lin}}}}$-representation, and a* polynomial hybrid automaton *if it has an $\mathcal{L}_{\mathbb{R}_{\mathcal{F}^{\text{poly}}}}$-representation.*

Example 2.9 (Nonlinear Bouncing Ball)**.** *The bouncing ball is a standard hybrid system model. The point of the example is to emphasize that nonlinear components can be written directly in the $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-representation.*

$$H_{BB} = \langle X, Q, \mathsf{flow}, \mathsf{jump}, \mathsf{inv}, \mathsf{init}\rangle$$

where

- $X = \mathbb{R}^2$ *and* $Q = \{q_u, q_d\}$.
- $\mathsf{flow}_{q_u}(x_0, v_0, x_t, v_t, t)$*:*

  $$(x_t = x_0 + \int_0^t v(s)ds) \wedge (v_t = v_0 + \int_0^t g(1 - \beta v(s)^2)ds)$$

  $\mathsf{flow}_{q_d}(x_0, v_0, x_t, v_t, t)$*:*

  $$(x_t = x_0 + \int_0^t v(s)ds) \wedge (v_t = v_0 + \int_0^t g(1 + \beta v(s)^2)ds)$$

  where β is a constant. Note that the integration terms define Type 2 computable functions, and can be directly used in $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas.

- $\mathsf{jump}_{q_d\to q_u}(x, v, x', v')$*:*

  $$x = 0 \wedge v' = v \cdot \exp(-\frac{c\pi}{2m\omega_d}) \wedge x' = x$$

  $\mathsf{jump}_{q_u\to q_d}(x, v, x', v')$*:*

  $$v = 0 \wedge x' = x \wedge v' = v$$

- $\mathsf{init}_{q_d} : x = 10 \wedge v = 0$.
- $\mathsf{inv}_{q_d} : x >= 0 \wedge v >= 0$ *and* $\mathsf{inv}_{q_u} : x >= 0 \wedge v <= 0$.

2.4 Hybrid Trajectories

Trajectories of hybrid systems combine continuous flows and discrete jumps. This motivates the use of a hybrid time domain, with which we can keep track of both the discrete changes and the duration of each continuous flow.

Definition 2.10 (Hybrid time domain)**.** *A* hybrid time domain *is a subset of* $\mathbb{N} \times \mathbb{R}$ *of the form*

$$T_m = \{(i,t) : i < m \text{ and } t \in [t_i, t_i'] \text{ or } [t_i, +\infty)\},$$

where $m \in \mathbb{N} \cup \{+\infty\}$*,* $\{t_i\}_{i=0}^m$ *is an increasing sequence in* $\mathbb{R}^+$*,* $t_0 = 0$*, and* $t_i' = t_{i+1}$*.*

Definition 2.11 (Hybrid Trajectories)**.** *Let* $X \subseteq \mathbb{R}^n$ *be an Euclidean space and* T_m *a hybrid time domain. A* hybrid trajectory *is any continuous function* $\xi : T_m \to X$*.*

To define trajectories of hybrid systems, we use a labeling function $\sigma_{\xi,H}(i)$ to map a step i to the corresponding discrete mode in H. In each mode, the system flows continuously following the dynamics defined by $\mathsf{flow}(q, \vec{x}_0, t)$. Note that $(t - t_k)$ is the actual duration in the k-th mode. When a switch between two modes is performed, it is required that $\xi(k+1, t_{k+1})$ is updated from the exit value $\xi(k, t_k')$ in the previous mode, following the jump conditions.

Definition 2.12 (Trajectories of a Hybrid Automaton)**.** *Let* H *be a hybrid automaton,* T_m *a hybrid domain, and* $\xi : T_m \to X$ *a hybrid trajectory. We say that* ξ *is* a trajectory of H of discrete depth m*, written as* $\xi \in [\![H]\!]$*, if there exists a* labeling function $\sigma_{\xi,H} : \mathbb{N} \to Q$ *such that:*

- *For some* $q \in Q$*,* $\sigma_{\xi,H}(0) = q$ *and* $\mathbb{R}_{\mathcal{F}} \models \mathsf{init}_q(\xi(0,0))$*.*
- *For any* $(i,t) \in T_m$*,* $\mathbb{R}_{\mathcal{F}} \models \mathsf{inv}_{\sigma_{\xi,H}(i)}(\xi(i,t))$*.*
- *For any* $(i,t) \in T_m$*,*
  - *When* $i = 0$*,* $\mathbb{R}_{\mathcal{F}} \models \mathsf{flow}_{q_0}(\xi(0,0), \xi(0,t), t)$*.*
  - *When* $i = k+1$*, where* $0 < k+1 < m$*, we have*

$$\mathbb{R}_{\mathcal{F}} \models \mathsf{flow}_{\sigma_\xi^H(k+1)}(\xi(k+1, t_{k+1}), \xi(k+1, t), (t - t_{k+1})),$$
$$\mathbb{R}_{\mathcal{F}} \models \mathsf{jump}_{\sigma_{\xi,H}(k) \to \sigma_{\xi,H}(k+1)}(\xi(k, t_k'), \xi(k+1, t_{k+1})).$$

We can write the time domain T_m *of* ξ *as* $T(\xi)$*.*

Remark 2.13 (jump vs inv)**.** *The jump conditions specify when* H may *switch to another mode. The invariants (when violated) specify when* H must *switch to another mode. They will lead to different logical encodings in reachability analysis.*

2.5 δ-Perturbations

The key benefit of using $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-representations for describing hybrid automata is that operations on the logic formulas can be directly transferred.

Definition 2.14 (δ-Perturbations)**.** *Let $\delta \in \mathbb{Q}^+ \cup \{0\}$. Suppose $H = \langle X, Q, \mathsf{flow}, \mathsf{jump}, \mathsf{inv}, \mathsf{init}\rangle$ is an $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-representation of hybrid system H. We define the* δ-weakening *of H as*

$$H^\delta = \langle X, Q, \mathsf{flow}^\delta, \mathsf{jump}^\delta, \mathsf{inv}^\delta, \mathsf{init}^\delta\rangle.$$

Example 2.15. *The δ-weakening of the bouncing ball automaton has its component formulas by their δ-weakening. For instance, $\mathsf{flow}^\delta_{q_u}(x_0, v_0, x_t, v_t, t)$ is*

$$|x_t - (x_0 + \int_0^t v(s)ds)| \le \delta \wedge |v_t - (v_0 + \int_0^t g(1 - \beta v(s)^2)ds))| \le \delta,$$

and $\mathsf{jump}^\delta_{q_d \to q_u}(x, v, x', v')$ is

$$|x| \le \delta \wedge |v' - v \cdot \exp(-\frac{c\pi}{2m\omega_d})| \le \delta \wedge |x' - x| \le \delta.$$

It is important to note that the notion of δ-perturbations is a purely syntactic one (defined on the description of hybrid systems), instead of a semantic one (defined on the trajectories). Note that the syntactic perturbations naturally lead to a semantic over-approximation of H in the trajectory space:

Proposition 2.16. *For any H and $\delta \in \mathbb{Q}^+ \cup \{0\}$, $[\![H]\!] \subseteq [\![H^\delta]\!]$.*

Proof. Let $\xi \in [\![H]\!]$ be any trajectory of H. Following Definition 2.4, for any $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$ sentence φ, we have $\varphi \to \varphi^\delta$. Since ξ satisfies the conditions in Definition 2.12, after replacing each formula by their δ-weakening, we have $\xi \in [\![H^\delta]\!]$. □

Proposition 2.17. *The δ-weakening of any hybrid automaton is nondeterministic.*

2.6 Reachability

The safety/reachability problem for hybrid systems can now be formally stated as follows.

Definition 2.18 (Reachability)**.** *Let H be an n-dimensional hybrid automaton, and U a subset of its state space $Q \times X$. We say U* is reachable by *H, if there exists $\xi \in [\![H]\!]$ with its time domain T and labeling function σ^H_ξ, such that there exists $(i, t) \in T$ satisfying $(\sigma^H_\xi(i), \xi(i, t)) \in U$.*

The bounded reachability problem for hybrid systems is defined by restricting the continuous components and time duration to a bounded domain, and the number of discrete transitions to a finite number.

Definition 2.19 (Bounded Reachability)**.** *Let H be an n-dimensional hybrid automaton, whose continuous state space X is a bounded subset of $\mathbb{R}^n$. Let U be a subset of its state space. Let $k \in \mathbb{N}$ and $M \in \mathbb{R}$. The (k, M)-bounded reachability problem asks whether there exists $\xi \in [\![H]\!]$ with its time domain $T(\xi)$ and labeling function σ_ξ, such that there exists $(i, t) \in T(\xi)$ with $i \leq k$, $t = \sum_{i=0}^{k} t_i$ where $t_i \leq M$, and $(\sigma_\xi(i), \xi(i, t)) \in U$.*

Remark 2.20. *By "step", we mean the number of discrete jumps. We say H can reach U in k steps, if there exists $\xi \in [\![H]\!]$ that contains k discrete jumps, entering and exiting the continuous flows in $k + 1$ modes.*

In the seminal work of [3, 2], it is shown that the bounded reachability problem for simple classes of hybrid automata is undecidable. Note that a common restriction in the existing study is that all constants are rational numbers, which does not need to be the case in our definitions.

3 Bounded Reachability

In this section we study the bounded δ-reachability problem and how to solve it practice. At the core of our framework is the correspondence between δ-reachability problems of hybrid systems and δ-decision problems of $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas.

3.1 Encoding Bounded Reachability in $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$

We first show how to encode bounded reachability using $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas. The encoding is mostly standard bounded model checking. However, in hybrid systems the invariant conditions and nondeterminism in the continuous flows play a special role.

We say a hybrid system H is *invariant-free* if $\mathsf{inv} = \emptyset$. We say H has *nondeterministic flow* if for some $q \in Q$, there exists $\vec{a}_0, \vec{a}_t, \vec{a}'_t \in \mathbb{R}^n$ and $t \in \mathbb{R}$ such that $\vec{a}_t \neq \vec{a}'_t$ and $\mathbb{R} \models \mathsf{flow}_q(\vec{a}_0, \vec{a}_t, t)$ and $\mathbb{R} \models \mathsf{flow}_q(\vec{a}_0, \vec{a}'_t, t)$.

Definition 3.1 (Unsafe Region)**.** *We use $\mathsf{unsafe} = \{\mathsf{unsafe}_q : q \in Q\}$ to denote the $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-representation of a subset of H. For each $q \in Q$, we have $([\![\mathsf{unsafe}_q]\!], q) = U \cap (X \times \{q\})$. We also write $[\![\mathsf{unsafe}]\!] = \bigcup_{q \in Q} [\![\mathsf{unsafe}]\!] \times \{q\}$.*

Now we define the encoding for three cases: hybrid systems that have trivial invariants, nontrivial invariants with deterministic flow, and nontrivial invariants with nondeterministic flow.

Systems with no invariants. We start with the simplest case for hybrid systems with no invariants. We define the following formula that checks whether an unsafe region is reachable after exactly k steps of discrete transition in a hybrid system.

Definition 3.2 (k-Step Reachability, Invariant-Free Case)**.** *Suppose H is invariant-free, and U a subset of its state space represented by* unsafe*. The $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formula* $\mathsf{Reach}_{H,U}(k, M)$ *is defined as:*

$$\exists^X \vec{x}_{0,q_0} \exists^X \vec{x}^t_{0,q_0} \cdots \exists^X \vec{x}_{0,q_m} \exists^X \vec{x}^t_{0,q_m} \cdots \exists^X \vec{x}_{k,q_m} \exists^X \vec{x}^t_{k,q_m} \exists^{[0,M]} t_0 \cdots \exists^{[0,M]} t_k.$$
$$\bigvee_{q \in Q} \Big(\mathsf{init}_q(\vec{x}_{0,q}) \wedge \mathsf{flow}_q(\vec{x}_{0,q}, \vec{x}^t_{0,q}, t_0) \Big)$$
$$\wedge \bigwedge_{i=0}^{k-1} \Bigg(\bigvee_{q,q' \in Q} \Big(\mathsf{jump}_{q \to q'}(\vec{x}^t_{i,q}, \vec{x}_{i+1,q'}) \wedge \mathsf{flow}_{q'}(\vec{x}_{i+1,q'}, \vec{x}^t_{i+1,q'}, t_{i+1}) \Big) \Bigg)$$
$$\wedge \bigvee_{i=0}^{k} \bigvee_{q \in Q} \mathsf{unsafe}_q(\vec{x}^t_{k,q}).$$

Intuitively, the trajectories start with some initial state satisfying $\mathsf{init}_q(\vec{x}_{0,q})$ for some q. In each step, it follows $\mathsf{flow}_q(\vec{x}_{i,q}, \vec{x}^t_{i,q}, t)$ and makes a continuous flow from $\vec{x}_i$ to $\vec{x}^t_i$ after time t. When H makes a jump from mode q' to q, it resets variables following $\mathsf{jump}_{q' \to q}(\vec{x}^t_{k,q}, \vec{x}_{k+1,q'})$.

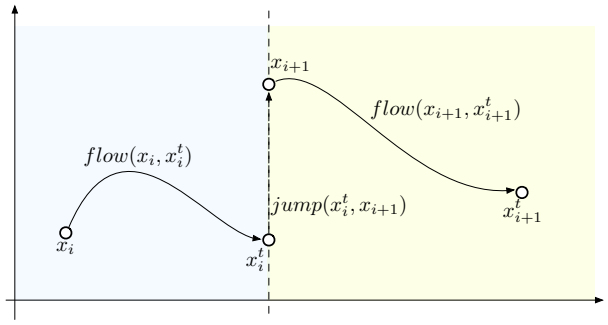

Figure 1

Systems with invariants and deterministic flows. When the invariants are not trivial, we need to ensure that during each continuous flow, the system always stays within the invariants. Such checking requires universal quantification over time.

Definition 3.3 (k-Step Reachability, Nontrivial Invariant and Deterministic Flow)**.** *Suppose H contains invariants and only deterministic flow , and U a subset of its state space represented by*

unsafe. *The $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formula* $\mathsf{Reach}_{H,U}(k, M)$ *is defined as:*

$$\exists^X \vec{x}_{0,q_0} \exists^X \vec{x}^t_{0,q_0} \cdots \exists^X \vec{x}_{0,q_m} \exists^X \vec{x}^t_{0,q_m} \cdots \exists^X \vec{x}_{k,q_m} \exists^X \vec{x}^t_{k,q_m} \exists^{[0,M]} t_0 \cdots \exists^{[0,M]} t_k.$$

$$\bigvee_{q \in Q} \Big(\mathsf{init}_q(\vec{x}_{0,q}) \wedge \mathsf{flow}_q(\vec{x}_{0,q}, \vec{x}^t_{0,q}, t_0) \wedge \forall^{[0,t_0]} t \forall^X \vec{x}\ (\mathsf{flow}_q(\vec{x}_{0,q}, \vec{x}, t) \rightarrow \mathsf{inv}_q(\vec{x})) \Big)$$

$$\wedge \bigwedge_{i=0}^{k-1} \Big(\bigvee_{q,q' \in Q} \Big(\mathsf{jump}_{q \rightarrow q'}(\vec{x}^t_{i,q}, \vec{x}_{i+1,q'}) \wedge \mathsf{flow}_{q'}(\vec{x}_{i+1,q'}, \vec{x}^t_{i+1,q'}, t_{i+1}) \wedge \forall^{[0,t_{i+1}]} t \forall^X \vec{x} (\mathsf{flow}_{q'}(\vec{x}_{i+1,q'}, \vec{x}, t) \rightarrow \mathsf{inv}_{q'}(\vec{x}))) \Big) \Big)$$

$$\wedge \bigvee_{i=0}^{k} \bigvee_{q \in Q} \mathsf{unsafe}_q(\vec{x}^t_{k,q}).$$

The extra universal quantifier for each continuous flow expresses the requirement that for all the time points between the initial and ending time point ($t \in [0, t_i + 1]$) in a flow, the continuous variables $\vec{x}$ must take values that satisfy the invariant conditions $\mathsf{inv}_q(\vec{x})$.

Systems with invariants and nondeterministic flows. In the most general case, a hybrid system can contain nondeterministic flow. When that is the case, for each time point, there is multiple possible values for the continuous variable. Yet it is not correct to universally quantify over all such possible values, because only one trajectory is needed. This problem is solved by introducing an additional level of existential quantification.

Definition 3.4 (k-Step reachability, Nontrivial Invariant, Nondeterministic Flow)**.** *Suppose H contains invariants and nondeterministic flow, and U a subset of its state space represented by* unsafe. *The $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formula* $\mathsf{Reach}_{H,U}(k, M)$ *is defined as:*

$$\exists^X \vec{x}_{0,q_0} \exists^X \vec{x}^t_{0,q_0} \cdots \exists^X \vec{x}_{0,q_m} \exists^X \vec{x}^t_{0,q_m} \cdots \exists^X \vec{x}_{k,q_m} \exists^X \vec{x}^t_{k,q_m} \exists^{[0,M]} t_0 \cdots \exists^{[0,M]} t_k.$$

$$\bigvee_{q \in Q} \Bigg(\mathsf{init}_q(\vec{x}_{0,q}) \wedge \mathsf{flow}_q(\vec{x}_{0,q}, \vec{x}^t_{0,q}, t_0) \wedge \forall^{[0,t_0]} t \forall^{[t,t_0]} t' \exists^X \vec{x} \exists^X \vec{x}' \Big(\mathsf{inv}_q(\vec{x}) \wedge \mathsf{inv}_q(\vec{x}') \mathsf{flow}_q(\vec{x}, \vec{x}', (t' - t)) \wedge \mathsf{flow}_q(\vec{x}_{0,q}, \vec{x}, t) \wedge \mathsf{flow}_q(\vec{x}', \vec{x}^t_{0,q}, t') \Big) \Bigg)$$

$$\wedge \bigwedge_{i=0}^{k-1} \Bigg(\bigvee_{q,q' \in Q} \Big(\mathsf{jump}_{q \rightarrow q'}(\vec{x}^t_{i,q}, \vec{x}_{i+1,q'}) \wedge \mathsf{flow}_{q'}(\vec{x}_{i+1,q'}, \vec{x}^t_{i+1,q'}, t_{i+1}) \wedge \forall^{[0,t_{i+1}]} t \forall^{[t,t_{i+1}]} t' \exists^X \vec{x} \exists^X \vec{x}' \Big(\mathsf{inv}_{q'}(\vec{x}) \wedge \mathsf{inv}_{q'}(\vec{x}') \wedge \mathsf{flow}_{q'}(\vec{x}, \vec{x}', (t' - t)) \wedge \mathsf{flow}_{q'}(\vec{x}_{i+1,q'}, \vec{x}, t) \wedge \mathsf{flow}_{q'}(\vec{x}', \vec{x}^t_{i+1,q'}, t') \Big) \Bigg)$$

$$\wedge \bigvee_{i=0}^{k} \bigvee_{q \in Q} \mathsf{unsafe}_q(\vec{x}^t_{k,q}).$$

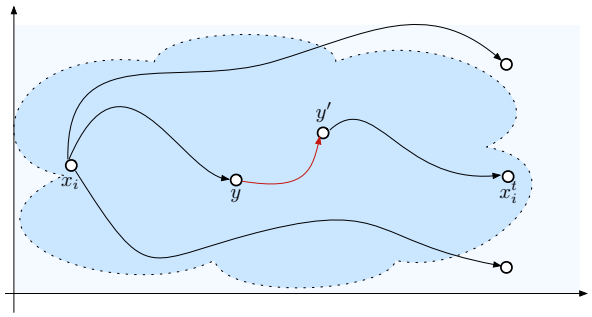

Figure 2

Intuitively, at each time point, the innermost existential quantifier asks for an assignment to the continuous variables $\vec{x}$ such that: first, there is a flow from the initial state in this step to the current assignment, as encoded by $\mathsf{flow}_{q'}(\vec{x}_{i+1,q'}, \vec{x}, t)$; second, from the current assignment there is a flow to $\vec{x}^t_{i+1,q'}$, the value that the continuous variables are supposed to take after the rest of the flow.

In the next section we will use these encodings to connect between δ-reachability and δ-decision problems of the corresponding $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas.

3.2 δ-Complete Bounded Reachability Analysis

Lemma 3.5. *Let $\delta \in \mathbb{Q}^+ \cup \{0\}$ be arbitrary. Suppose H is a hybrid system, U a subset of its state space represented by* unsafe*, and $\mathsf{Reach}_{H,U}(k, M)$ encodes (k, M)-bounded reachability. Let H, U, k, M all be arbitrary.*

We always have $\mathbb{R} \models (\mathsf{Reach}_{H,U}(k, M))^\delta$, iff, there exists a trajectory $\xi \in [\![H^\delta]\!]$ such that for some $(k, t) \in T^M(\xi)$, $(\xi(k, t), \sigma_\xi(k)) \in [\![\mathsf{unsafe}^\delta]\!]$.

Proof. We prove by induction on k, for the most general case of systems with nontrivial invariants and nondeterministic flows. The simpler cases then automatically hold.

(i) Case $k = 0$. Suppose $\mathsf{Reach}^\delta_{H,U}(0, M)$ is true. Then there exists $q \in Q$, $\vec{a}_0, \vec{a}^t_0 \in \mathbb{R}^n \cap X$ and $t_0 \in \mathbb{R}^+ \cap [0, M]$ such that for all $t \in [0, t_0]$, there exists $\vec{a}(t) \in X$ satisfying:

$$\mathsf{init}^\delta_q(\vec{a}_0) \wedge \mathsf{flow}^\delta_q(\vec{a}_0, \vec{a}^t_0, t_0) \wedge \mathsf{flow}^\delta_q(\vec{a}_0, \vec{a}(t), t) \wedge \mathsf{flow}^\delta_q(\vec{a}(t), \vec{a}_t, t_0 - t) \wedge \mathsf{inv}_q(\vec{a}(t)) \wedge \mathsf{unsafe}^\delta_q(\vec{a}_t).$$

Note that there is no discrete jump. Accordingly, set a trajectory ξ to be:

$$\xi(0, 0) = \vec{a}_0, \xi(0, t_0) = \vec{a}^t_0,$$

and for all time point $t \in [0, t_0]$, $\xi(0, t) \in \vec{a}(t)$. Following Definition 2.12 and Definition 2.7, $\xi \in [\![H^\delta]\!]$, and $\xi(0, \vec{a}^t_0) \in [\![\mathsf{unsafe}^\delta]\!]$.

On the other hand, suppose there is a $\xi \in [\![H^\delta]\!]$ such that $\xi(0, t_0)$ is in $[\![\mathsf{unsafe}_q]\!]$ for some $t_0 \in [0, M]$. We set $\vec{a}_0 = \xi(0, 0)$, $\vec{a}^t_0 = \xi(0, t_0)$. Then following the conditions that ξ satisfies in Definition 2.12, for every $t \in [0, t_0]$, there is $\vec{a}(t)$ such that $\mathsf{flow}^\delta_q(\vec{a}_0, \vec{a}(t), t)$ and $\mathsf{flow}^\delta_q(\vec{a}(t), \vec{a}_t, t)$. Consequently, $\mathsf{Reach}^\delta_{H,U}(0, M)$ is true, witnessed by these assignments.

(ii) Case $k \geq 1$. Suppose $\mathsf{Reach}^{\delta}_{H,U}(k, M)$ is true. Then there exists

$$q_0, ..., q_k \in Q, \vec{a}_0, \vec{a}^t_0, ..., \vec{a}_k, \vec{a}^t_k \in X, \text{ and } t_0, ..., t_k \in [0, M]$$

such that for all $t_{q_0} \in [0, t_0], ..., t_{q_k} \in [0, t_k]$ there exists $\vec{a}(t_{q_0}), ..., \vec{a}(t_{q_k}) \in X$ satisfying:

$$\begin{aligned}
&\mathsf{init}^{\delta}_{q}(\vec{a}_0) \wedge \mathsf{flow}^{\delta}_{q}(\vec{a}_0, \vec{a}^t_0, t_0) \wedge \mathsf{flow}^{\delta}_{q}(\vec{a}_0, \vec{a}(t), t) \wedge \mathsf{flow}^{\delta}_{q_0}(\vec{a}(t), \vec{a}^t_0, t_0 - t) \wedge \mathsf{inv}^{\delta}_{q_0}(\vec{a}(t)) \\
&\wedge \mathsf{jump}^{\delta}_{q_0 \to q_1}(\vec{a}^t_0, \vec{a}_1) \wedge \cdots \wedge \mathsf{flow}^{\delta}_{q_{k-1}}(\vec{a}_{k-1}, \vec{a}^t_{k-1}, t_0) \wedge \mathsf{flow}^{\delta}_{q_{k-1}}(\vec{a}_{k-1}, \vec{a}(t_{q_{k-1}}), t) \\
&\wedge \mathsf{flow}^{\delta}_{q_{k-1}}(\vec{a}(t_{q_{k-1}}), \vec{a}^t_k, (t_{k-1} - t)) \wedge \mathsf{inv}^{\delta}_{q_{k-1}}(\vec{a}(t_{q_{k-1}})) \\
&\wedge \mathsf{jump}^{\delta}_{q_{k-1} \to q_k}(\vec{a}^t_{k-1}, \vec{a}_k) \wedge \mathsf{flow}^{\delta}_{q_k}(\vec{a}_k, \vec{a}^t_k, t_k) \wedge \mathsf{flow}^{\delta}_{q_k}(\vec{a}_k, \vec{a}(t_{q_k}), t_k) \\
&\wedge \mathsf{flow}^{\delta}_{q_k}(\vec{a}(t_{q_{k-1}}), \vec{a}^t_k, (t_{q_{k-1}} - t)) \wedge \mathsf{inv}^{\delta}_{q_{k-1}}(\vec{a}(t_{q_k})) \wedge \mathsf{unsafe}^{\delta}_{q_k}(\vec{a}^t_k).
\end{aligned}$$

Now, to perform induction, we truncate the last step in the formula and define a new region U' represented by:

$$\begin{aligned}
\mathsf{unsafe}^{\mathrm{tail}}_{q_{k-1}}(\vec{x}) \;=\; & \mathsf{jump}_{q_{k-1} \to q_k}(\vec{x}, \vec{a}_k) \wedge \mathsf{flow}_{q_k}(\vec{a}_k, \vec{a}^t_k, t_k) \wedge \mathsf{flow}_{q_k}(\vec{a}_k, \vec{a}(t_{q_k}), t_k) \\
& \wedge \mathsf{flow}_{q_k}(\vec{a}(t_{q_{k-1}}), \vec{a}^t_k, (t_{q_{k-1}} - t)) \wedge \mathsf{inv}_{q_{k-1}}(\vec{a}(t_{q_k})) \wedge \mathsf{unsafe}_{q_k}(\vec{a}^t_k).
\end{aligned}$$

We then see that the formula $\mathsf{Reach}^{\delta}_{H,U'}(k-1, M)$ is true, as simply witnessed by the trace above, using the new formula $\mathsf{unsafe}^{\mathrm{tail}}_{q_{k-1}}$ to represent the last transition:

$$\begin{aligned}
&\mathsf{init}^{\delta}_{q}(\vec{a}_0) \wedge \mathsf{flow}^{\delta}_{q}(\vec{a}_0, \vec{a}^t_0, t_0) \wedge \mathsf{flow}^{\delta}_{q}(\vec{a}_0, \vec{a}(t), t) \wedge \mathsf{flow}^{\delta}_{q_0}(\vec{a}(t), \vec{a}^t_0, t_0 - t) \wedge \mathsf{inv}^{\delta}_{q_0}(\vec{a}(t)) \\
&\wedge \mathsf{jump}^{\delta}_{q_0 \to q_1}(\vec{a}^t_0, \vec{a}_1) \wedge \cdots \wedge \mathsf{flow}^{\delta}_{q_{k-1}}(\vec{a}_{k-1}, \vec{a}^t_{k-1}, t_0) \wedge \mathsf{flow}^{\delta}_{q_{k-1}}(\vec{a}_{k-1}, \vec{a}(t_{q_{k-1}}), t) \\
&\wedge \mathsf{flow}^{\delta}_{q_{k-1}}(\vec{a}(t_{q_{k-1}}), \vec{a}^t_k, (t_{k-1} - t)) \wedge \mathsf{inv}^{\delta}_{q_{k-1}}(\vec{a}(t_{q_{k-1}})) \wedge (\mathsf{unsafe}^{\mathrm{tail}}_{q_{k-1}}(\vec{a}^t_{k-1}))^{\delta}.
\end{aligned}$$

Consequently, by inductive hypothesis, there exists a trajectory $\xi_{k-1} \in [\![H^{\delta}]\!]$ that reaches the region U'. Now, we extend ξ_{k-1} with the assignments in the k-the step, i.e.:

$$\xi = \xi_{k-1} \cup \{(k, \vec{a}(t_{q_k})) : t \in [0, t_k]\}$$

where $\vec{a}(0) = \vec{a}_k, \vec{a}(t_k) = \vec{a}^t_k$. We now obtain $\xi \in [\![H^{\delta}]\!]$ such that ξ reaches the region represented by unsafe^{δ}.

On the other hand, suppose there is a trajectory $\xi \in [\![H^{\delta}]\!]$ such that ξ reaches the region represented by unsafe^{δ}. Again, following an argument similar to the above, and Definition 2.12 we can find the sequence of assignments that witnesses the formula $\mathsf{Reach}^{\delta}_{H,U}(k, M)$ to be true. □

Now we can easily show that the bounded δ-reachability problems is decidable for any $\mathcal{L}_{\mathbb{R}_{\mathcal{F}}}$-representable hybrid system.

Theorem 3.6 (Decidability)**.** *Let $\delta \in \mathbb{Q}^+$ be arbitrary. There exists an algorithm such that, for any hybrid system $\mathcal{L}_{\mathbb{R}_{\mathcal{F}}}$-represented by H and an unsafe region U $\mathcal{L}_{\mathbb{R}_{\mathcal{F}}}$-represented by* unsafe*, solves the (k, M)-bounded δ-reachability problem for H for any given bounds $k \in \mathbb{N}, M \in \mathbb{R}^+$.*

Proof. We need to show that there is an algorithm that correctly returns one of the following answers:

- safe: H does not reach the region represented by unsafe within the (k, M)-bound;
- δ-unsafe: H^δ reaches the region represented by unsafe^δ within the (k, M)-bound.

For this, we only need to solve the δ-decision problem for the formula $\mathsf{Reach}^k_{H,U}(i, M)$, from which we obtain an answer of either φ is false, or φ is δ-true (Theorem 2.5).

• Suppose φ is false. Then we know that for any $i \leq k$, $\mathsf{Reach}_{H,U}(i, M)$ is false. Using Lemma 3.5 for the special case $\delta = 0$, we know that there does not exist a trajectory $\xi \in [\![H]\!]$ that can reach U within i steps, and consequently the system is safe within the (k, M)-bound.

• Suppose φ is δ-true, we know that there exists $i \leq k$ such that $\mathsf{Reach}^\delta_{H,U}(i, M)$ is true. Using Lemma 3.5 for $\delta \in \mathbb{Q}^+$, we know that there exists a trajectory $\xi \in [\![H^\delta]\!]$ that can reach the region represented by unsafe^δ in i-steps, i.e., within the (k, M)-bound. □

From the structures of the $\mathcal{L}_{\mathbb{R}_\mathcal{F}}$-formulas encoding δ-reachability, we can obtain the following complexity results of the reachability problems.

Theorem 3.7 (Complexity)**.** *Suppose all the functions in the description of H is in complexity class* C*. Then deciding the (k, M)-bounded δ-reachability problem is in*

- NP^C *for an invariant-free H;*
- $(\Sigma^P_2)^\mathsf{C}$ *for H with nontrivial invariants and deterministic flows;*
- $(\Sigma^P_3)^\mathsf{C}$ *for H with nontrivial invariants and nondeterministic flows.*

Proof. It is clear that the logic structures of the $\mathsf{Reach}_{H,U}(k, M)$ formulas in the three cases are Σ_1, Σ_2, and Σ_3 respectively. Consequently, using complexity results for Theorem 2.6, the complexity of the δ-decision problems resides in NP^C, $(\Sigma^P_2)^\mathsf{C}$, and $(\Sigma^P_3)^\mathsf{C}$ respectively.

The missing step here is that the $\mathsf{Reach}_{H,U}(k, M)$ formulas are of exponential length, because of the enumeration of all possible paths through the discrete modes requires an exponential number (m^{k+1}, where m is the number of discrete modes in H) of copies of the continuous variables. Thus the $\mathsf{Reach}_{H,U}(k, M)$ encodings do not provide a polynomial-reduction to the δ-decision problems.

Observe that, however, we can nondeterministically select single paths through the modes. This is just what we did in the proof of Lemma 3.5. Here we show how to do this for the Σ_3 case of nontrivial invariants and nondeterministic flows and the other cases are subsumed. Nondeterministically, we can choose a sequence of modes $q_0, ..., q_k \in Q$ and solve the δ-decision problem for the formula:

$$
\begin{aligned}
&\exists^X \vec{x}_0 \exists^X \vec{x}^t_0 \cdots \exists^X \vec{x}_q \exists^X \vec{x}^t_q \exists^{[0,M]} t_0 \cdots \exists^{[0,M]} t_k \forall^{[0,t_0]} t_{q_0} \cdots \forall^{[0,M]} t_{q_k} \exists^X x_{q_0} \cdots \exists \vec{x}_{q_k} \\
&\Big(\mathsf{init}(\vec{x}_0) \wedge \mathsf{flow}_{q_0}(\vec{x}_0, \vec{x}^t_0, t_0) \wedge \mathsf{flow}_{q_0}(\vec{x}_0, \vec{x}_{q_0}, t_{q_0}) \wedge \mathsf{flow}_{q_0}(\vec{x}_{q_0}, \vec{x}^t_0, (t_0 - t_{q_0})) \\
&\wedge \mathsf{inv}_{q_0}(\vec{x}_{q_0}) \wedge \mathsf{jump}_{q_0 \to q_1}(\vec{x}'_0, \vec{x}_1) \wedge \cdots \wedge \mathsf{flow}_{q_k}(\vec{x}_k, \vec{x}_{q_k}, t_{q_k}) \wedge \mathsf{flow}_{q_k}(\vec{x}_{q_k}, \vec{x}^t_k, (t_k - t_{q_k})) \\
&\wedge \mathsf{inv}_{q_k}(\vec{x}_{q_k}) \wedge \mathsf{unsafe}_{q_k}(\vec{x}^t_k) \Big)
\end{aligned}
$$

Now, this formula is polynomial in H, unsafe, k, M. Thus, we can use the nondeterministic machine to randomly first select such a formula in polynomial time, and δ-decide its truth value, which is in $(\Sigma_3^{\mathsf{P}})^{\mathsf{C}}$. Thus, the complexity of the δ-reachability problem is still in $(\Sigma_3^{\mathsf{P}})^{\mathsf{C}}$ for this case. □

Corollary 3.8. *For linear and polynomial hybrid automata, the bounded δ-reachability problem ranges from being* NP*-complete to Σ_3^{P}-complete for the three cases. For hybrid automata that can be $\mathcal{L}_{\mathbb{R}_{\mathcal{F}}}$-represented with whose $\mathcal{F}$ contains the set of ODEs defined* P*-computable right-hand side functions, the problem is* PSPACE*-complete.*

Proof. The results come from the fact that the complexity of polynomials is in P, and the set of ODEs in questions are PSPACE-complete. □

References

[1] R. Alur. Formal verification of hybrid systems. In *EMSOFT*, pages 273–278, 2011.

[2] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, editors, *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*, pages 209–229. Springer, 1992.

[3] R. Alur and D. L. Dill. The theory of timed automata. In J. W. de Bakker, C. Huizing, W. P. de Roever, and G. Rozenberg, editors, *REX Workshop*, volume 600 of *Lecture Notes in Computer Science*, pages 45–73. Springer, 1991.

[4] M. Fränzle. Analysis of hybrid systems: An ounce of realism can save an infinity of states. In J. Flum and M. Rodríguez-Artalejo, editors, *CSL*, volume 1683 of *Lecture Notes in Computer Science*, pages 126–140. Springer, 1999.

[5] M. Fränzle, T. Teige, and A. Eggers. Engineering constraint solvers for automatic analysis of probabilistic hybrid automata. *J. Log. Algebr. Program.*, 79(7):436–466, 2010.

[6] S. Gao, J. Avigad, and E. M. Clarke. Delta-complete decision procedures for satisfiability over the reals. In B. Gramlich, D. Miller, and U. Sattler, editors, *IJCAR*, volume 7364 of *Lecture Notes in Computer Science*, pages 286–300. Springer, 2012.

[7] S. Gao, J. Avigad, and E. M. Clarke. Delta-decidability over the reals. In *LICS*, pages 305–314, 2012.

[8] S. Gulwani and A. Tiwari. Constraint-based approach for analysis of hybrid systems. In A. Gupta and S. Malik, editors, *CAV*, volume 5123 of *Lecture Notes in Computer Science*, pages 190–203. Springer, 2008.

[9] T. A. Henzinger. The theory of hybrid automata. In *LICS*, pages 278–292, 1996.

[10] T. A. Henzinger and J.-F. Raskin. Robust undecidability of timed and hybrid systems. In *HSCC*, pages 145–159, 2000.

[11] C. Herde, A. Eggers, M. Fränzle, and T. Teige. Analysis of hybrid systems using hysat. In *ICONS*, pages 196–201, 2008.

[12] Z. Huang and S. Mitra. Computing bounded reach sets from sampled simulation traces. In *HSCC*, pages 291–294, 2012.

[13] A. Kawamura. Lipschitz continuous ordinary differential equations are polynomial-space complete. In *IEEE Conference on Computational Complexity*, pages 149–160. IEEE Computer Society, 2009.

[14] K.-I. Ko. *Complexity Theory of Real Functions*. BirkHauser, 1991.

[15] P. Prabhakar, V. Vladimerou, M. Viswanathan, and G. E. Dullerud. Verifying tolerant systems using polynomial approximations. In *RTSS*, pages 181–190, 2009.

[16] S. Ratschan. Safety verification of non-linear hybrid systems is quasi-semidecidable. In *TAMC*, pages 397–408, 2010.

[17] K. Weihrauch. *Computable Analysis: An Introduction*. 2000.