Control in o-minimal hybrid systems

Patricia Bouyer*[1], Thomas Brihaye**[2], Fabrice Chevalier*[1]

* LSV - CNRS & ENS de Cachan
61, avenue du Président Wilson, 94230 Cachan, France
e-mails: {bouyer,chevalie}@lsv.ens-cachan.fr

** Université de Mons-Hainaut, Institut de Mathématique
6, avenue du Champ de Mars, 7000 Mons, Belgium
e-mail: thomas.brihaye@umh.ac.be

Abstract

In this paper, we consider the control of general hybrid systems. In this context we show that time-abstract bisimulation is not adequate for solving such a problem. That is why we consider an other equivalence, namely the suffix equivalence based on the encoding of trajectories through words. We show that this suffix equivalence is in general a correct abstraction for control problems. We apply this result to o-minimal hybrid systems, and get decidability and computability results in this framework.

1 Introduction

Control of hybrid systems. Hybrid systems are finite-state machines equipped with a continuous dynamics. In the last thirty years, formal verification of such systems has become a very active field of research in computer science, with numerous success stories. In this context, hybrid automata, an extension of timed automata [1], have been intensively studied [12, 13], and decidable subclasses of hybrid systems have been drawn like initialized rectangular hybrid automata [13]. More recently, the control of hybrid systems has appeared as a new interesting and active field of research, and many results have already been obtained, like the (un)decidability of control problems for hybrid automata [14], or (semi-)algorithms for solving such problems [10]. Given a system S (with controllable and uncontrollable actions) and a property φ, controlling the system means building another system C (which can only enforce controllable actions), called the controller, such that $S \parallel C$ (the system S guided by the controller C) satisfies the property φ. In our context, the property is a reachability property and our aim is to build a controller enforcing a given location of the system, whatever the environment does (which plays with the uncontrollable actions).

O-minimal hybrid systems. O-minimal hybrid systems have been first proposed in [18] as an interesting class of systems (see [21] for an overview of properties of o-minimal structures). They have very rich continuous dynamics, but limited discrete steps (at each discrete step, all variables have to be reset, independently from their initial values). This allows to decouple the continuous and discrete components of the hybrid system (see [18]). Thus, properties of a global o-minimal system can be deduced directly from properties of the continuous parts of the system. Since the introductory paper [18], several works have considered o-minimal hybrid systems [9, 8, 7, 17], mostly focusing on abstractions of such systems, on reachability properties, and on bisimulation properties.

Word encoding. In [8], an encoding of trajectories with words has been proposed in order to prove the existence of finite bisimulations for o-minimal hybrid systems (see also [7]). Let us mention that this technique has been used in [17] in order to provide an exponential bound on the size of the finite bisimulation in the case of pfaffian hybrid systems. Let us also notice that similar techniques already appeared in the literature, see for instance the notion of *signature* in [4]. Different word encoding techniques have been studied in a wider context in [6]. In this paper we use the so-called suffix encoding, which was shown to be in general too fine to provide the coarsest time-abstract bisimulation.

[1]Work partly supported by ACI Cortos, a program of the French ministry of research.

[2]This author is supported by the following research programs: FRFC 2.4.530.02., FRFC 2.4.564.02, Modnet MRTN-CT-2004-512234 and by a grant from the National Bank of Belgium.

However, based on this encoding, a semi-algorithm has been proposed in [6] for computing a time-abstract bisimulation, and it terminates in the case of o-minimal hybrid systems (under some word uniqueness hypothesis).

Contributions of this paper. In this paper, we focus on the control of hybrid systems, and use the above-mentioned suffix word encoding of trajectories for giving sufficient computability conditions for the winning states of a game. Time-abstract bisimulation is an equivalence relation which is correct with respect to reachability properties [2]. Game bisimulation is correct for discrete infinite-state games [10]. Here, we show that the time-abstract bisimulation is not correct for solving control problems: we exhibit a system in which two states are time-abstract bisimilar, but one of the states is winning and the other is not winning. Using the word encoding of trajectories of [6], we prove that two states having the same suffixes in this encoding are equivalently winning or losing (this is a stronger condition than the one for the time-abstract bisimulation). We finally focus on o-minimal hybrid games and prove that, under the assumption that the theory of the underlying o-minimal structure is decidable, the control problem can be solved and that winning states and winning strategies can be computed.

Related work. The most relevant related works are those on hybrid games [14, 10]. However the framework of these papers is pretty different from ours:

1) In their framework, time is considered as a discrete action, and once action "let time elapse" has been chosen, it is not possible to bound the time elapsing, which is quite restrictive. For instance, the timed game of Figure 1 is winning from $(\ell_0, x = 0)$ in our framework (the strategy is to wait some amount of time $t \in [2, 5]$ and to take the controllable action c), whereas it is not winning in their framework (once x is above 5, it is no more possible to take the transition and reach the winning location ℓ_1, and there is no way to impose a delay within $[2, 5]$). This yields significant differences in the properties: in their framework, game bisimulation is one of the tools for solving the games, and as stated by [14, Prop. 1], the classical bisimulation tool is then sufficient to solve games. On the contrary, in our framework, the notion of bisimulation relevant to our model (time-abstract bisimulation) is not correct for solving games, as will be explored in this paper.

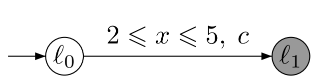


Figure 1. A simple game

2) Our games are control games, they are thus asymmetric, which is not the case of the games in the above-mentioned works; in our framework, the environment is more powerful than the controller in that it can outstrip the controller and do an action right before the controller decides to do a controllable action.

Let us also mention the paper [22] on control of Linear Hybrid Automata. In [22] the author proposes a semidecision procedure for synthesizing controllers for such automata. No general decidability result is given in this paper.

Plan of the paper. In Section 2, we define the games over dynamical systems we consider, and we show that time-abstract bisimulation is not correct for solving them. The word encoding technique is presented in Section 3 and used in Section 4 to present a general framework for solving games over dynamical systems. We apply and extend these results in Section 5 for computing winning states and winning strategies in o-minimal games.

2 Games over dynamical systems

2.1 Dynamical systems

Let $\mathcal{M}$ be a structure. In this paper when we say that some relation, subset or function is *definable*, we mean it is first-order definable in the sense of the structure $\mathcal{M}$. A general reference for first-order logic is [16]. We denote by $\mathsf{Th}(\mathcal{M})$ the theory of $\mathcal{M}$. In this paper we only consider structures $\mathcal{M}$ that are expansions of ordered groups, we also assume that the structure $\mathcal{M}$ contains two symbols of constants, *i.e.* $\mathcal{M} = \langle M, +, 0, 1, <, \dots \rangle$ and w.l.o.g. we assume that $0 < 1$.

Definition 2.1 *A* dynamical system *is a pair* $(\mathcal{M}, \gamma)$ *where:*

- $\mathcal{M} = \langle M, +, 0, 1, <, \dots \rangle$ *is an expansion of an ordered group,*
- $\gamma : V_1 \times M^+ \to V_2$ *is a function definable in* $\mathcal{M}$ *(where* $M^+ = \{m \in M \mid m \geqslant 0\}$*,* $V_1 \subseteq M^{k_1}$*, and* $V_2 \subseteq M^{k_2}$*).*[3]

The function γ *is called the dynamics of the system.*

Classically, when M is the field of the reals, we see M^+ as the time, $V_1 \times M^+$ as the space-time, V_2 as the (output) space and V_1 as the input space. We keep this terminology in the more general context of a structure $\mathcal{M}$.

The definition of *dynamical systems* encompasses a lot of different behaviors. Let us first give a simple example, several others will be presented later.

[3] We use these notations in the rest of the paper.

Example 2.2 We can recover the continuous dynamics of *timed automata* (see [1]). In this case, we have that $\mathcal{M} = \langle \mathbb{R}, <, +, 0, 1\rangle$ and the dynamics $\gamma : \mathbb{R}^n \times [0, +\infty[\to \mathbb{R}^n$ is defined by $\gamma(x_1, \ldots, x_n, t) = (x_1 + t, \ldots, x_n + t)$.

Definition 2.3 *If we fix a point $x \in V_1$, the set $\Gamma_x = \{\gamma(x,t) \mid t \in M^+\} \subseteq V_2$ is called the* trajectory *determined by x.*

We define a transition system associated with the dynamical system, this definition is an adaptation to our context of the classical *continuous transition system* in the case of hybrid systems (see [18] for example).

Definition 2.4 *Given $(\mathcal{M}, \gamma)$ a dynamical system, we define a* transition system $T_\gamma = (Q, \Sigma, \to_\gamma)$ associated with the dynamical system *by:*

- *the set Q of states is V_2;*
- *the set Σ of events is $M^+ = \{m \in M \mid m \geqslant 0\}$;*
- *the transition relation $y_1 \xrightarrow{t}_\gamma y_2$ is defined by:* $\exists x \in V_1,\ \exists t_1, t_2 \in M^+$ *such that* $t_1 \leqslant t_2$, $\gamma(x, t_1) = y_1,\ \gamma(x, t_2) = y_2$ *and* $t = t_2 - t_1$

2.2 $\mathcal{M}$-games

In this subsection, we define $\mathcal{M}$-automata, which are automata with guards, resets and continuous dynamics definable in the $\mathcal{M}$-structure. We then introduce our model of dynamical game which is an $\mathcal{M}$-automaton with two sets of actions, one for each player; we finally express in terms of winning strategy the main problem we will be interested in, the control problem in a class $\mathcal{C}$ of $\mathcal{M}$-automata.

Definition 2.5 ($\mathcal{M}$-automaton) *An* $\mathcal{M}$-automaton $\mathcal{A}$ *is a tuple $(\mathcal{M}, Q, \mathsf{Goal}, \Sigma, \delta, \gamma)$ where $\mathcal{M} = \langle M, +, 0, 1, <, \ldots\rangle$ is an expansion of an ordered group, Q is a finite set of locations, $\mathsf{Goal} \subseteq Q$ is a subset of winning locations, Σ is a finite set of actions, δ consists in a finite number of transitions $(q, g, a, R, q') \in Q \times 2^{V_2} \times \Sigma \times (V_2 \to 2^{V_2}) \times Q$ where g and R are definable in $\mathcal{M}$, and γ maps every location $q \in Q$ to a dynamic $\gamma_q : V_1 \times M^+ \to V_2$.*

We use a general definition for resets: a reset R is indeed a general function from V_2 to 2^{V_2}, which may correspond to a non-deterministic update. If the current state is (q, y) the system will jump to some (q', y') with $y' \in R(y)$.

An $\mathcal{M}$-automaton $\mathcal{A} = (\mathcal{M}, Q, \mathsf{Goal}, \Sigma, \delta, \gamma)$ defines a *mixed transition system* $T_\mathcal{A} = (S, \Gamma, \to)$ where:

- the set S of states is $Q \times V_2$;
- the set Γ of labels is $M^+ \cup \Sigma$;
- the transition relation $(q, y) \xrightarrow{e} (q', y')$ is defined when:
  - $e \in \Sigma$ and there exists $(q, g, e, R, q') \in \delta$ with $y \in g$ and $y' \in R(y)$, or
  - $e \in M^+$, $q = q'$, and $y \xrightarrow{e}_{\gamma_q} y'$ where γ_q is the dynamic in location q.

In the sequel, we will focus on behaviors of $\mathcal{M}$-automata which alternate between continuous transitions and discrete transitions.

We will also need more precise notions of transitions. When $(q, y) \xrightarrow{t'} (q, y')$ with $t' \in M^+$, this is due to some choice of $(x, t) \in V_1 \times M^+$ such that $\gamma_q(x, t) = y$. We say that $(q, y) \xrightarrow{t'}_{x,t} (q, y')$ if $\gamma_q(x, t) = y$ and $\gamma_q(x, t + t') = y'$. To ease the reading of the paper, we will sometimes write $(q, x, t, y) \xrightarrow{t'} (q, x, t + t', y')$ for $(q, y) \xrightarrow{t'}_{x,t} (q, y')$. We say that an action $(d, a) \in M^+ \times \Sigma$ is enabled in a state (q, x, t, y) if there exists (q', x', t', y') and (q'', x'', t'', y'') such that $(q, x, t, y) \xrightarrow{d} (q', x', t', y') \xrightarrow{a} (q'', x'', t'', y'')$. We then write $(q, x, t, y) \xrightarrow{d,a} (q'', x'', t'', y'')$.

A *run* of $\mathcal{A}$ is a finite or infinite sequence $(q_0, x_0, t'_0, y_0) \xrightarrow{t_1, a_1} (q_1, x_1, t'_1, y_1) \ldots$ where for every i, $(q_i, y_i) \xrightarrow{t_i}_{x_i, t'_i} (q_i, y'_i) \xrightarrow{a} (q_{i+1}, y_{i+1})$. Such a run is said *winning* if $q_i \in \mathsf{Goal}$ for some i.

We note $\mathsf{Runs}_f(\mathcal{A})$ the set of finite runs in $\mathcal{A}$. If ρ is a finite run $(q_0, x_0, t'_0, y_0) \xrightarrow{t_1, a_1} \ldots \xrightarrow{t_n, a_n} (q_n, x_n, t'_n, y_n)$ we define $last(\rho) = (q_n, x_n, t'_n, y_n)$.

Definition 2.6 ($\mathcal{M}$-game) *An* $\mathcal{M}$-game *is an $\mathcal{M}$-automaton $(\mathcal{M}, Q, \mathsf{Goal}, \Sigma,\ \delta, \gamma)$ where Σ is partitioned into two subsets Σ_c and Σ_u corresponding to controllable and uncontrollable actions.*

Without loss of generality, we assume that there is a loop labeled by a controllable action on every state of Goal.

Definition 2.7 (Strategy) *A* strategy[4] *is a partial function λ from $\mathsf{Runs}_f(\mathcal{A})$ to $M^+ \times \Sigma_c$ such that for all runs ρ in $\mathsf{Runs}_f(\mathcal{A})$, $\lambda(\rho)$ is enabled in $last(\rho)$.*

The strategy tells what needs to be done for controlling the system: at each instant it tells what delay we need to wait and which controllable action needs to be done after this delay. Note that the environment may have to choose between several edges, each labeled by the action given by the strategy (because the original game is not deterministic).

A strategy λ is said *memoryless* if for all finite runs ρ_1 and ρ_2, $last(\rho_1) = last(\rho_2)$ implies $\lambda(\rho_1) = \lambda(\rho_2)$. Let $\rho = (q_0, x_0, t'_0, y_0) \xrightarrow{t_1, a_1} \ldots$ be a run, and set for every i,

[4] In the context of control problems, a strategy is also called a *controller*.

ρ_i the prefix of length i of ρ. The run ρ is said *consistent with a strategy* λ when for all i, if $\lambda(\rho_i) = (t, a)$ then either $t_{i+1} = t$ and $a_{i+1} = a$, or $t_{i+1} \leqslant t$ and $a_{i+1} \in \Sigma_u$. A run ρ is said *maximal* if it is infinite or if it is finite ending in (q, x, t, y) and satisfies that for all $t' \geqslant 0$, for all $a \in \Sigma$, "$(q, x, t, y) \xrightarrow{t',a}$" implies $a \in \Sigma_u$. A strategy λ is *winning from a state (q,y)* if for all (x, t) such that $\gamma(x, t) = y$, all maximal runs starting in (q, x, t, y) compatible with λ are winning. The *set of winning states* is the set of states from which there is a winning strategy.

We can now define the control problem we will study.

Problem (Control problem in a class $\mathcal{C}$ of $\mathcal{M}$-automata). Given an $\mathcal{M}$-game $\mathcal{A} \in \mathcal{C}$, and a definable initial state (q, y), determine whether there exists a winning strategy in $\mathcal{A}$ from (q, y).

2.3 $\mathcal{M}$-game and bisimulation

Time-abstract bisimulation [9, 2, 12] is a sufficient behavioral relation to check reachability properties of timed systems, and in particular of $\mathcal{M}$-automata [6]. When considering control problems, we will see that this tool is not sufficient for solving control problems.

Definition 2.8 *Given a mixed transition system $T = (S, \Gamma, \rightarrow)$, a* time-abstract bisimulation for T *is an equivalence relation $\sim \subseteq S \times S$ such that $\forall q_1, q_1', q_2 \in S$, the two following conditions are satisfied:*

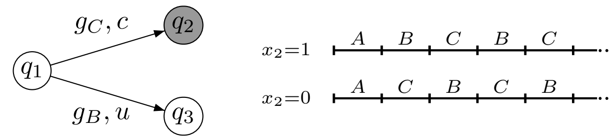
$$\forall a \in \Sigma,\ \left(q_1 \sim q_1' \textit{ and } q_1 \xrightarrow{a} q_2\right) \Rightarrow \left(\exists q_2' \in S \textit{ s.t. } q_2 \sim q_2' \textit{ and } q_1' \xrightarrow{a} q_2'\right)$$

$$\forall t \in M^+,\ \left(q_1 \sim q_1' \textit{ and } q_1 \xrightarrow{t} q_2\right) \Rightarrow \left(\exists t' \in M^+,\ \exists q_2' \in S \textit{ s.t. } q_2 \sim q_2' \textit{ and } q_1' \xrightarrow{t'} q_2'\right)$$

Example 2.9 Let us consider the $\mathcal{M}$-game $\mathcal{A} = (\mathcal{M}, Q, \mathsf{Goal}, \Sigma, \delta, \gamma)$ where $\mathcal{M} = \langle \mathbb{R}, <, +, 0, 1, \equiv_2 \rangle$ ($\equiv_2$ denotes the "modulo 2" relation), $Q = \{q_1, q_2, q_3\}$, $\mathsf{Goal} = \{q_2\}$, $\Sigma = \Sigma_c \cup \Sigma_u = \{c, u\}$ where $\Sigma_c = \{c\}$ (resp. $\Sigma_u = \{u\}$) is the set of controllable (resp. uncontrollable) actions. The dynamic in q_1, $\gamma_{q_1} : \mathbb{R}^+ \times \{0,1\} \times \mathbb{R}^+ \rightarrow \mathbb{R}^+ \times \{0,1\}$ is defined as $\gamma_{q_1}(x_1, x_2, t) = (x_1 + t, x_2)$.

We consider the partition depicted on Figure 2(b). The guard g_C is satisfied on C-states and the guard g_B is satisfied on B-states. Note that this partition is compatible with Goal and w.r.t. discrete transitions.

In this game, the controller can win when it enters a C-state by performing action c and it loses when entering a B-state because it cannot prevent the environment from performing a u and going in the losing state q_3.



(a) The $\mathcal{M}$-game $\mathcal{A}$ (b) Dynamics in q_1

Figure 2. Time-abstract bisimulation does not preserve winning states

It follows that the state $s_1 = (q_1, (0, 1))$ is losing, whereas the state $s_2 = (q_1, (0, 0))$ is winning. However the equivalence relation induced by the partition $\{A, B, C\}$ is a time-abstract bisimulation: the two states s_1 and s_2 are thus time-abstract bisimilar, but not equivalent for the game. It follows that time-abstract bisimulation is not correct for solving control problems, in the sense that a time-abstract bisimulation cannot always distinguish between winning and losing states.

Proposition 2.10 *Let $\mathcal{M}$ be a structure and $\mathcal{A}$ an $\mathcal{M}$-game. A partition respecting* Goal *and inducing a time-abstract bisimulation on $Q \times V_2$ does not necessarily respect the set of winning states of $\mathcal{A}$.*

3 Suffix and dynamical type

In this section we explain how to encode trajectories of dynamical systems with words. This technique was introduced in [8, 7] in order to study o-minimal hybrid systems. We focus on the *suffix partition* introduced in [6].

We first explain how to build words associated with trajectories. Given $(\mathcal{M}, \gamma)$ a dynamical system and $\mathcal{P}$ a finite partition of V_2, given $x \in V_1$ we associate a word with the trajectory $\Gamma_x = \{\gamma(x, t) \mid t \in M^+\}$ in the following way. We consider the sets $\{t \in M^+ \mid \gamma(x, t) \in P\}$ for $P \in \mathcal{P}$. This gives a partition of the time M^+. In order to define a word on $\mathcal{P}$ associated with the trajectory determined by x, we need to define the set of intervals $\mathcal{F}_x = \{I \mid I$ is a time interval or a point and is maximal for the property $\exists P \in \mathcal{P},\ \forall t \in I,\ \gamma(x, t) \in P\}$. For each x, the set $\mathcal{F}_x$ is totally ordered by the order induced from M. This allows us to define *the word on $\mathcal{P}$ associated with* Γ_x denoted ω_x.

Definition 3.1 *Given $x \in V_1$,* the word associated with Γ_x *is given by the function $\omega_x : \mathcal{F}_x \rightarrow \mathcal{P}$ defined by $\omega_x(I) = P$, where $I \in \mathcal{F}_x$ is such that $\forall t \in I$, $\gamma(x, t) \in P$.*

The set of words associated with $(\mathcal{M}, \gamma)$ over $\mathcal{P}$ gives in some sense a complete *static* description of the dynamical system $(\mathcal{M}, \gamma)$ through the partition $\mathcal{P}$. In order to recover the *dynamics*, we need further information.

Given a point x of the input space V_1, we have associated with x a trajectory Γ_x and a word ω_x. If we consider (x, t) a point of the space-time $V_1 \times M^+$, it corresponds to a point $\gamma(x, t)$ lying on Γ_x. To recover in some sense the position of $\gamma(x, t)$ on Γ_x from ω_x, we associate with (x, t) a suffix of the word ω_x denoted $\omega_{(x,t)}$. The construction of $\omega_{(x,t)}$ is similar to the construction of ω_x, we only need to consider the sets of intervals $\mathcal{F}_{(x,t)} = \{I \cap \{t' \in M^+ \mid t' \geqslant t\} \mid I \in \mathcal{F}_x\}$.

Let us notice that given (x, t) a point of the space-time $V_1 \times M^+$ there is a unique suffix $\omega_{(x,t)}$ of ω_x associated with (x, t). Given a point $y \in V_2$ it may have several (x, t) such that $\gamma(x, t) = y$ and so several suffixes are associated with y. In other words, given $y \in V_2$, the *future* of y is non-deterministic, and a single suffix $\omega_{(x,t)}$ is thus not sufficient to recover the dynamics of the transition system through the partition $\mathcal{P}$. To encode the dynamical behavior of a point y of the output space V_2 through the partition $\mathcal{P}$, we introduce the notion of *suffix dynamical type* of a point y w.r.t. $\mathcal{P}$.

Definition 3.2 *Given a dynamical system $(\mathcal{M}, \gamma)$, a finite partition $\mathcal{P}$ of V_2, a point $y \in V_2$, the* suffix dynamical type of y w.r.t. $\mathcal{P}$ *is denoted* $\mathsf{Suf}_{\mathcal{P}}(y)$ *and defined by* $\mathsf{Suf}_{\mathcal{P}}(y) = \{\omega_{(x,t)} \mid \gamma(x, t) = y\}$.

This allows us to define an equivalence relation on V_2. Given $y_1, y_2 \in V_2$, we say that they are *suffix-equivalent* if and only if $\mathsf{Suf}_{\mathcal{P}}(y_1) = \mathsf{Suf}_{\mathcal{P}}(y_2)$.

We denote by $\mathsf{Suf}(\mathcal{P})$ the partition induced by this equivalence. We say that a partition $\mathcal{P}$ is *suffix-stable* if $\mathsf{Suf}(\mathcal{P}) = \mathcal{P}$ (it implies that if y_1 and y_2 belong to the same piece of $\mathcal{P}$ then $\mathsf{Suf}_{\mathcal{P}}(y_1) = \mathsf{Suf}_{\mathcal{P}}(y_2)$).

To understand the word encoding technique, we provide several examples.

Example 3.3 We first consider a two dimensional timed automata dynamics (see Example 2.2). In this case we have that $\gamma(x_1, x_2, t) = (x_1 + t, x_2 + t)$. We associate with this dynamics the partition $\mathcal{P} = \{A, B\}$ where $B = [1, 2]^2$ and $A = \mathbb{R}^2 \setminus B$. In this example the suffix partition is made of three pieces, which are depicted in Figure 3.

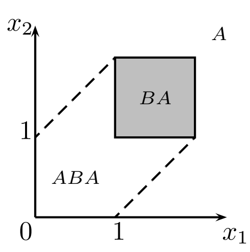


Figure 3. Suffixes for the timed automata dynamics

Example 3.4 We consider the dynamical system $(\mathcal{M}, \gamma)$ where $\mathcal{M} = \langle \mathbb{R}, +, \cdot, 0, 1, <, \sin_{|[0,2\pi]}, \cos_{|[0,2\pi]} \rangle$ [5] and $\gamma : \mathbb{R}^2 \times [0, 2\pi] \times \mathbb{R} \to \mathbb{R}^2$ is defined as follows.
$\gamma(x_1, x_2, \theta, t) =$
$$\begin{cases} (t.\cos(\theta), t.\sin(\theta)) & \text{if } (x_1, x_2) = (0,0) \\ (x_1 + t.x_1, x_2 + t.x_2) & \text{if } (x_1, x_2) \neq (0,0) \end{cases}$$

We associate with this dynamical system the partition $\mathcal{P} = \{A, B, C\}$ where $A = \{(0,0)\}$, $B = \{(\theta\cos(\theta), \theta\sin(\theta)) \mid 0 < \theta \leqslant 2\pi\}$ and $C = \mathbb{R}^2 \setminus (A \cup C)$. Let us call piece B *the spiral* (see Figure 4). There are four dynamical types for this system: $\{ACBC\}$ for the central point $(0,0)$, $\{CBC\}$ for the "interior" of the spiral, $\{BC\}$ for the spiral, and $\{C\}$ for the "exterior" of the spiral. Let us notice that though the dynamical system is infinitely branching in $(0,0)$, there is a unique suffix associated with each point y of the output space.

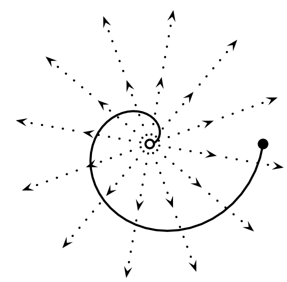
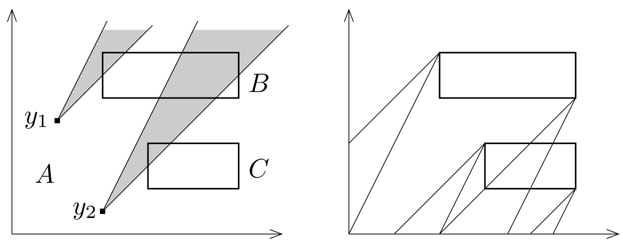


Figure 4. The dynamical system of the spiral

Dynamical systems and suffix dynamical type allow also to encode more sophisticated continuous dynamics. In the next example we recover in some sense the continuous dynamics of *rectangular automata* [15], which requires to use the suffix dynamical types (some of the points do not have a unique suffix).

Example 3.5 We consider the dynamical system $(\mathcal{M}, \gamma)$ where $\mathcal{M} = \langle \mathbb{R}, +, \cdot, 0, 1, < \rangle$ and $\gamma : \mathbb{R}^2 \times [0, 2] \times \mathbb{R}^+ \to \mathbb{R}^2$ is defined by $\gamma(x_1, x_2, p, t) = (x_1 + t, x_2 + p \cdot t)$. We associate with this dynamical system the partition $\mathcal{P} = \{A, B, C\}$ where $B = [2, 5] \times [3, 4]$, $C = [3, 5] \times [1, 2]$ and $A = \mathbb{R}^2 \setminus (B \cup C)$ (see Figure 5(a)). Let us focus on the suffix dynamical types of the two points $y_1 = (1, 2.5)$ and $y_2 = (2, 0.5)$. We have that $\mathrm{Suf}_{\mathcal{P}}(y_1) = \{A, ABA\}$ and $\mathrm{Suf}_{\mathcal{P}}(y_2) = \{ABA, ACABA\}$. Though several points have several possible suffixes, the partition induced by the suffix dynamical type is finite and illustrated in Figure 5(b).

[5] $\sin_{|[0,2\pi]}$ and $\cos_{|[0,2\pi]}$ correspond to the sinus and cosinus functions restricted to the segments $[0, 2\pi]$.



(a) The dynamics (b) The suffix partition

Figure 5. A rectangular dynamics

4 Solving an $\mathcal{M}$-game

In this section we present a general procedure to compute the set of winning states for an $\mathcal{M}$-game. We then show that if a partition is *suffix-stable*, the procedure can be performed symbolically on pieces of the partition. The procedure described is not always effective [6] and we will point out specific $\mathcal{M}$-structures for which each step of the procedure is computable.

4.1 Controllable predecessors

As for classical reachability games [11], one way of computing winning states is to compute the *attractor* of goal states by iterating a *controllable predecessor* operator.

Let $\mathcal{A} = (\mathcal{M}, Q, \mathsf{Goal}, \Sigma, \delta, \gamma)$ be an $\mathcal{M}$-game. For $A \subseteq Q \times V_2$ and $a \in \Sigma$ we define the controllable and uncontrollable discrete predecessors as follows:
$\mathsf{cPred}(A) =$

$$\left\{ (q,y) \in Q \times V_2 \left| \begin{array}{l} \exists c \in \Sigma_c,\ \text{c is enabled in } (q,y), \\ \text{and } \forall (q',y') \in Q \times V_2, \\ (q,y) \xrightarrow{c} (q',y') \Rightarrow (q',y') \in A \end{array} \right. \right\}$$

$\mathsf{uPred}(A) =$

$$\left\{ (q,y) \in Q \times V_2 \left| \begin{array}{l} \exists u \in \Sigma_u,\ \exists (q',y') \in Q \times V_2 \text{ s.t.} \\ (q,y) \xrightarrow{u} (q',y') \text{ and } (q',y') \in A \end{array} \right. \right\}$$

As for timed and hybrid games [3, 14], we also define a *safe* time predecessor of a set A w.r.t. a set B: a state (q,y) is in $\mathsf{Pred}_t(A,B)$ if, by letting time elapse, one reaches $(q',y') \in A$, avoiding B. Formally the operator Pred_t is defined as follows:
$\mathsf{Pred}_t(A,B) =$

$$\left\{ (q,y) \in Q \times V_2 \left| \begin{array}{l} \forall (x,t) \text{ s.t. } \gamma_q(x,t) = y,\ \exists t' \in M^+ \\ (q,y) \xrightarrow{t'}_{x,t} (q',y'),\ (q',y') \in A, \\ \text{and } \mathsf{Post}^{q,x}_{[t,t+t']} \subseteq \overline{B} \end{array} \right. \right\}$$

[6] The effectivity of the computation will be discussed later.

where $\mathsf{Post}^{q,x}_{[t,t+t']} = \{\gamma_q(x,t'') \mid t \leqslant t'' \leqslant t+t'\}$.

The *controllable predecessor* operator is then defined as:

$$\pi(A) = A \cup \mathsf{Pred}_t(\mathsf{cPred}(A), \mathsf{uPred}(\overline{A}))$$

Intuitively, a state (q,y) is in $\pi(A)$ whenever either it is already in A or there is a way of waiting some amount of time, and of performing a controllable action to enter A, and no uncontrollable action leads outside A.

We say that a partition $\mathcal{P}$ is *stable under* π if for every piece $A \in \mathcal{P}$, $\pi(A)$ is a union of pieces of $\mathcal{P}$.

Remark 4.1 Note that the operator π is definable in any expansion of an ordered group. Hence, if A is definable, so is $\pi(A)$.

We will compute the set of winning states by iterating the operator π. Denoting $\pi^*(\mathsf{Goal}) = \bigcup_{k \geqslant 0} \pi^k(\mathsf{Goal})$, we will show that if the iterative computation of $\pi^k(\mathsf{Goal})$ stabilizes, the set of winning states for the game is precisely $\pi^*(\mathsf{Goal})$. This will help getting further effective definability and computability results of winning states and winning strategies under some assumption on the underlying structure.

Proposition 4.2 *Let $\mathcal{A} = (\mathcal{M}, Q, \mathsf{Goal}, \Sigma, \delta, \gamma)$ be an $\mathcal{M}$-game, and $(q,y) \in Q \times V_2$. If there exists $n \in \mathbb{N}$ s.t. $\pi^n(\mathsf{Goal}) = \pi^{n+1}(\mathsf{Goal})$ then $\pi^*(\mathsf{Goal}) = \pi^n(\mathsf{Goal})$ is the set of winning states of $\mathcal{A}$.*

This property is quite classical in the framework of hybrid games, see for example [5]. Note that the hypothesis that π stabilizes is really needed.

We now deduce an algorithmic result from proposition 4.2. The set of winning states is $\pi^*(\mathsf{Goal})$ but this does not imply that we can compute this set as many $\mathcal{M}$-structures are already intrinsically undecidable. The following corollary states that if some conditions on the structure and on π are satisfied, then this procedure provides an algorithmic solution to the control problem.

Corollary 4.3 *Let $\mathcal{M}$ be a structure such that $\mathsf{Th}(\mathcal{M})$ is decidable.[7] Let $\mathcal{C}$ be a class of $\mathcal{M}$-games such that for every $\mathcal{A}$ in $\mathcal{C}$, there exists a finite partition $\mathcal{P}$ of $Q \times V_2$ definable in $\mathcal{M}$, respecting Goal[8], and stable under π. Then the control problem in the class $\mathcal{C}$ is decidable. Moreover if $\mathcal{A} \in \mathcal{C}$, the set of winning states of $\mathcal{A}$ is computable.*

[7] We recall that a theory $\mathsf{Th}(\mathcal{M})$ is decidable iff there is an algorithm which can determine whether or not any sentence (*i.e.* a formula with no free variable.) is a member of the theory (*i.e.* is true). We suggest to readers interested in general decidability issues on o-minimal hybrid systems to refer to Section 5 of [7].

[8] *i.e.* Goal is a union of pieces of $\mathcal{P}$.

4.2 Stability of $\mathsf{Suf}(\mathcal{P})$ under π

In section 2.3, we have presented a counter-example which showed that time-abstract bisimulation is not always correct to solve control problems. The main reason is that the partition induced by time-abstract bisimilarity is not stable under the operator π.

We now present a sufficient condition for a partition to be stable under the operator π: we require that the partition is stable under cPred and uPred to handle the discrete part of the automaton and we show that the stability by suffix is fine enough to be correct for solving control problems.

Proposition 4.4 *Let $\mathcal{A}$ be an $\mathcal{M}$-game, $\mathcal{P}$ be a partition of $Q \times V_2$ and π be the controllable predecessor operator. If $\mathcal{P}$ respects* Goal*, is stable under* cPred, uPred *and suffix-stable, then $\mathcal{P}$ is stable under the operator π.*

Proof. The idea of the proof of Proposition 4.4 is the following. Given X a piece of $\mathcal{P}$ and $y \in \pi(X)$. The fact that $y \in \pi(X)$ can be translated as follows in term of words. There exists a word $\omega \in \mathrm{Suf}_{\mathcal{P}}(y)$ with prefix ω_s such that the last letter of ω_s belongs to $\mathsf{cPred}(X)$ and ω_s contains no occurence of letters included in $\mathsf{uPred}(\overline{X})$. By *suffix-stability hypothesis* we know that any y' belonging to the same piece of $\mathcal{P}$ as y has the same suffix as y. This allows to conclude that $y' \in \pi(X)$. □

As a corollary of this proposition and of Corollary 4.3, we get the following general decidability result.

Corollary 4.5 *Let $\mathcal{M}$ be a structure such that* $\mathsf{Th}(\mathcal{M})$ *is decidable. Let $\mathcal{C}$ be a class of $\mathcal{M}$-games such that for every $\mathcal{A}$ in $\mathcal{C}$, there exists a finite partition $\mathcal{P}$ of $Q \times V_2$ definable in $\mathcal{M}$, respecting* Goal*, and suffix-stable. Then the control problem in the class $\mathcal{C}$ is decidable, and if $\mathcal{A} \in \mathcal{C}$, the set of winning states of $\mathcal{A}$ is computable.*

Note that being suffix-stable is a stronger condition than being a time-abstract bisimulation [6], and we see here that this is one of the right tools to solve control problems. For instance in Example 2.9 the partition $\mathcal{P}$ is a time-abstract bisimulation but is not suffix-stable. Indeed $s_1,\ s_2 \in A$ but $\mathrm{Suf}_{\mathcal{P}}(s_1) \neq \mathrm{Suf}_{\mathcal{P}}(s_2)$.

Remark 4.6 The results of this section permit to recover the results of [3] about control of timed automata. Indeed we consider the classical finite partition of timed automata that induces the region graph (see [1]). Let us call $\mathcal{P}_R$ this partition, and notice that $\mathcal{P}_R$ is definable in $\langle \mathbb{R}, <, +, 0, 1 \rangle$. $\mathcal{P}_R$ is stable under the action of cPred and uPred. By Example 2.2 the continuous dynamics of timed automata is definable in $\langle \mathbb{R}, <, +, 0, 1 \rangle$. Hence it makes sense to encode continuous trajectories of timed automata as words. Then one can easily be convinced that $\mathsf{Suf}(\mathcal{P}_R) = \mathcal{P}_R$. By Corollary 4.5 we get the decidability and computability of winning states in timed games [3] as a side result.

Corollary 4.7 *The control problem in the class of timed automata is decidable. Moreover the set of winning states $\pi^*(\mathsf{Goal})$ is computable.*

5 O-minimal games

In this section, we focus on the particular case of o-minimal games (*i.e.* $\mathcal{M}$-games where $\mathcal{M}$ is an o-minimal structure and in which extra assumptions are made on the resets) [18].

We first briefly recall definitions and results related to o-minimality [19]. The reader interested in o-minimality should refer to [21] for further results and an extensive bibliography on this subject. Then we focus on o-minimal structures with a decidable theory in order to obtain decidability and computability results.

Definition 5.1 *An extension of an ordered structure $\mathcal{M} = \langle M, <, \ldots \rangle$ is* o-minimal *if every definable subset of M is a finite union of points and open intervals (possibly unbounded).*

In other words the definable subsets of M are the simplest possible: the ones which are definable in $\langle M, < \rangle$. The following are examples of o-minimal structures.

Example 5.2 There are many examples of o-minimal structures: the ordered group of rationals $\langle \mathbb{Q}, <, +, 0, 1 \rangle$, the ordered field of reals $\langle \mathbb{R}, <, +, \cdot, 0, 1 \rangle$, the field of reals with exponential function, the field of reals expanded by restricted pfaffian functions and the exponential function, and many more interesting structures.

5.1 Generalities on o-minimal games

Definition 5.3 *Given $\mathcal{A}$ an $\mathcal{M}$-game, we say that $\mathcal{A}$ is an* o-minimal game *if the structure $\mathcal{M}$ is o-minimal and if all transitions (q, g, a, R, q') of $\mathcal{A}$ belong to*[9] $Q \times 2^{V_2} \times \Sigma \times 2^{V_2} \times Q$.

Let us notice that the previous definition implies that given $\mathcal{A}$ an o-minimal game, the guards, the resets and the dynamics are definable in the underlying o-minimal structure. We denote by $\mathcal{P}_{\mathcal{A}}$ the partition of the state space $S = Q \times V_2$ which respects Goal, and all guards and resets in $\mathcal{A}$. Note that $\mathcal{P}_{\mathcal{A}}$ is a finite definable partition of S.

Due to the strong reset condition we have that $\mathcal{P}_{\mathcal{A}}$ is stable under the action of cPred and uPred. This holds by the

[9] This is a particular case of reset for $\mathcal{M}$-game where we consider only constant functions for resets.

same argument that allows to decouple the continuous and discrete components of the hybrid system in [18]. Let us also notice that, in the framework of o-minimal games, any refinement of $\mathcal{P}_{\mathcal{A}}$ is stable under the action of cPred and uPred.

O-minimal games are *o-minimal hybrid systems* (as defined in [7]). With slight adaptations of Lemma 4.13 and Theorem 4.18 of [7], we can easily deduce the following result.

Theorem 5.4 ([7]) *Let $\mathcal{A}$ be an o-minimal game.*

- *Given $y \in V_2$ we have that $\mathsf{Suf}_{\mathcal{P}_{\mathcal{A}}}(y)$ consists of finitely many finite words on $\mathcal{P}_{\mathcal{A}}$,*
- *the partition $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$ is finite and definable,*
- *if there exists a unique suffix on $\mathcal{P}_{\mathcal{A}}$ associated with each $y \in V_2$ we have that $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$ is a time-abstract bisimulation.*

5.2 O-minimal games with unique suffixes

In this subsection, we apply the general results obtained in Section 4 to the particular case of o-minimal games, and we get partial results when we assume that the game satisfies a suffix uniqueness hypothesis.

Proposition 5.5 *Let $\mathcal{A}$ be an o-minimal game, $\mathcal{P}$ a partition inducing a time-abstract bisimulation and respecting* Goal*, guards and resets of $\mathcal{A}$. If there exists a unique suffix on $\mathcal{P}$ associated with each $(q, y) \in Q \times V_2$ then $\mathcal{P}$ is stable under the action of π.*

Proof. This proposition holds because:

- if $\mathcal{P}$ is a partition inducing a time-abstract bisimulation, and if there is a unique suffix on $\mathcal{P}$, then $\mathsf{Suf}(\mathcal{P}) = \mathcal{P}$
- we can then apply Proposition 4.4 with partition $\mathcal{P}$.

□

Note that thanks to Theorem 5.4 the *suffix partition* $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$ is a partition which satisfies the hypotheses of the above proposition under the suffix uniqueness hypothesis.

This result does however not contradict Proposition 2.10 which stated that time-abstract bisimulation is in general not a correct tool to solve control problems. Indeed, the example of Figure 2(b) satisfies the suffix uniqueness hypothesis but is not o-minimal. It is also possible to construct another example which is o-minimal but does not satisfy the suffix uniqueness hypothesis. The above proposition thus really requires both hypotheses "o-minimal" and "unique suffix".

In the next subsection, we will describe another partition, which satisfies a stronger property than time-abstract bisimulation, and which will also be a correct tool for analyzing all o-minimal games, even the ones which don't satisfy the suffix uniqueness hypothesis.

Remark 5.6 Let us notice that the "unique suffix" assumption of Proposition 5.5 already encompasses the continuous behavior allowed in [18] (where the dynamics γ is the flow of a vector field that does not depend on the time, and is thus time-deterministic). More general systems can also be handled, for example the spiral dynamics (Example 5.9) which is an infinitely branching system with unique suffix.

5.3 Relaxing the suffix uniqueness hypothesis

In the previous subsection, under a suffix uniqueness assumption, applying Proposition 4.4, we have shown that $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$ is stable under the action of π. We will now prove that we can remove this suffix uniqueness assumption and keep the stability of $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$ under the action of π. From now and for the rest of the paper, we ignore the suffix uniqueness hypothesis. Of course in this more general framework we can not apply Proposition 4.4 anymore as in general the partition $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$ is not suffix-stable (even in the restricted framework of o-minimal systems); this is for instance the case for the rectangular semantics described in Figure 5(a)).

The goal of this subsection is to provide a new tool, namely the suffix partition $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$, for analyzing o-minimal games (even when the suffix uniqueness assumption is removed). Theorem 5.4 then ensures finiteness of $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$. Even though $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$ is not always a time-abstract bisimulation (as on Figure 5(b)), we will show that it is stable under the action of π.

Proposition 5.7 *Let $\mathcal{A}$ be an o-minimal game. The suffix partition $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$ is finite and stable under the action of π.*

Proof. The proof of this proposition uses same kinds of ideas as the proof of Proposition 4.4. The difficult point is the translation of "the belonging to $\pi(X)$" in term of words. However we can not rely anymore on the suffix stability of the partition we are working with, namely $\mathsf{Suf}(\mathcal{P}_{\mathcal{A}})$, (see Figure 6). That is why we need to consider the strong reset conditions of o-minimal games in order to conclude. □

5.4 Synthesis of winning strategies

We now prove that given $\mathcal{A}$ an o-minimal game definable in $\mathcal{M}$, we can construct a *definable* strategy (in the same structure $\mathcal{M}$) for the winning states. The effectiveness of this construction will be discussed in subsection 5.5.

Theorem 5.8 *Given $\mathcal{A}$ an o-minimal game, there exists a definable memoryless winning strategy for each $(q, y) \in \pi^*(\mathsf{Goal})$.*

Proof. The key point in the proof of Theorem 5.8 is to be able to definably pick a delay $d \in M^+$ making the strategy $\lambda(q, x, t, y) = (d, a)$ winning, for some $a \in \Sigma_c$. This is possible by using the *curve selection* for o-minimal expansions of ordered groups (see [21, chap.6]). This allows for example to definably pick the middle point of a open interval of the form (m, m'). □

Let us now illustrate Theorem 5.8 on two examples.

Example 5.9 Let us consider again the automaton shape of Example 2.9. We now define from $\mathcal{A}$ an o-minimal game $\mathcal{A}_s$ related to the spiral example (Example 3.4). The underlying o-minimal structure[10] $\mathcal{M}$ is $\langle \mathbb{R}, +, \cdot, 0, 1, <, \sin_{|[0,2\pi]}, \cos_{|[0,2\pi]} \rangle$. The o-minimal game $\mathcal{A}_s$ has the same set of locations, same Goal, same set of actions and same underlying finite automaton as $\mathcal{A}$ (*i.e.* Figure 2(a) represents also $\mathcal{A}_s$). The two differences between $\mathcal{A}$ and $\mathcal{A}_s$ are the guards and the continuous dynamics. Let us first define the guards. We have that g_B can be taken on B-states (*i.e.* points on the spiral) and g_C on C-states (points not on the spiral and different from the origin). The continuous dynamics in q_1 are the one described by the dynamical system of Example 3.4 (the continuous dynamics in q_2 and q_3 do not play any role). Clearly g_B, g_C and γ_{q_1} are definable in $\mathcal{M}$.

The winning strategy in point $(0, 0)$ given by Theorem 5.8 is $\lambda(0, 0, \theta, t) = (\frac{\theta}{2}, c)$ where c consists in taking the transition leading to state q_2 (which is winning).

Example 5.10 Let us notice that in the case of timed automata dynamics (described in Example 2.2), our definable strategies correspond in some sense to the realizable strategies obtained in [5].

5.5 Decidability result

Theorem 5.8 is an existential result. It claims that given an o-minimal game, there exists a definable memoryless strategy for each $y \in \pi^*(\mathsf{Goal})$, and by Theorem 5.4 we know that $\mathsf{Suf}(\mathcal{P})$ is finite. The conclusion of the previous subsection is that given an o-minimal game there exists a definable memoryless winning strategy for each $y \in \pi^*(\mathsf{Goal})$.

In general, Theorem 5.8 does not allow to conclude that the control problem in an $\mathcal{M}$-structure is decidable. Indeed it depends on the decidability of $\mathsf{Th}(\mathcal{M})$. We can state the following theorem:

[10] This structure is o-minimal (see [20]).

Theorem 5.11 *Let $\mathcal{M}$ be an o-minimal structure such that $\mathsf{Th}(\mathcal{M})$ is decidable and $\mathcal{C}$ a class of $\mathcal{M}$-automata. Then the control problem in class $\mathcal{C}$ is decidable. Moreover if $\mathcal{A} \in \mathcal{C}$, the set of winning states $\pi^*(\mathsf{Goal})$ is computable and a memoryless strategy can be effectively computed for each $(q, y) \in \pi^*(\mathsf{Goal})$.*

Proof. By Theorem 5.4, for each $\mathcal{A} \in \mathcal{C}$, $\mathsf{Suf}(\mathcal{P}_\mathcal{A})$ is a definable finite partition respecting Goal; Proposition 5.7 ensures that this partition is stable under π. Hypothesis of Corollary 4.3 are thus satisfied and we get that the control problem in class $\mathcal{C}$ is decidable and that the winning states of a game $\mathcal{A} \in \mathcal{C}$ are computable. □

Remark 5.12 Let us notice that $\langle \mathbb{R}, <, +, 0, 1 \rangle$ and $\langle \mathbb{R}, <, +, \cdot, 0, 1 \rangle$ are examples of o-minimal structures with decidable theory.

Remark 5.13 In fact, Theorem 5.11 can be proved for a wider class than o-minimal systems, the condition that every variable is reset on every transition is not mandatory: it is sufficient to have a suffix-stable partition which is stable under the action of cPred and uPred; if this condition is satisfied (and the dynamic in every state is o-minimal) the resets can be arbitrary.

Timed automata can be treated in this framework. Theorem 5.11 thus provides in particular a way to compute winning strategies for timed games.

6 Conclusion

In this paper we have studied the control problem of dynamical systems with general dynamics. We have shown that time-abstract bisimulation is not fine enough to solve them, which is a major difference with the discrete case. Using an encoding of trajectories by words [6], we have proved that the so-called suffix partition is a good abstraction for control problems (with reachability winning conditions, but it applies also to basic safety winning conditions). We have finally provided decidability and computability results for o-minimal games. Our technique applies to timed automata, and we get the decidability of timed games [3], as well as the construction of winning strategies [5] as side results.

There are several interesting further research directions: we could try to assume only partial observability of the system, or we could try to apply similar techniques to systems where there is not such a strong reset condition when a discrete action is done.

References

[1] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[2] R. Alur, Th. A. Henzinger, G. Lafferriere, and G. J. Pappas. Discrete abstractions of hybrid systems. *Proc. of the IEEE*, 88:971–984, 2000.

[3] E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. Controller synthesis for timed automata. In *Proc. IFAC Symposium on System Structure and Control*, pages 469–474. Elsevier Science, 1998.

[4] E. Asarin, G. Schneider, and S. Yovine. On the decidability of the reachability problem for planar differential inclusions. In *Proc. 4th International Workshop on Hybrid Systems: Computation and Control (HSCC'01)*, volume 2034 of *Lecture Notes in Computer Science*, pages 89–104. Springer, 2001.

[5] P. Bouyer, F. Cassez, E. Fleury, and K. G. Larsen. Optimal strategies in priced timed game automata. In *Proc. 24th Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS'04)*, volume 3328 of *Lecture Notes in Computer Science*, pages 148–160. Springer, 2004.

[6] Th. Brihaye. Words and bisimulation of dynamical systems. *Journal of Automata, Languages and Combinatorics*, 2006. To appear.

[7] Th. Brihaye and Ch. Michaux. On the expressiveness and decidability of o-minimal hybrid systems. *Journal of Complexity*, 21(4):447–478, 2005.

[8] Th. Brihaye, Ch. Michaux, C. Rivière, and C. Troestler. On o-minimal hybrid systems. In *Proc. 7th International Workshop on Hybrid Systems: Computation and Control (HSCC'04)*, volume 2993 of *Lecture Notes in Computer Science*, pages 219–233. Springer, 2004.

[9] J. M. Davoren. Topologies, continuity and bisimulations. *Informatique Théorique et Applications*, 33(4-5):357–382, 1999.

[10] L. de Alfaro, Th. A. Henzinger, and R. Majumdar. Symbolic algorithms for infinite-state games. In *Proc. 12th International Conference on Concurrency Theory (CONCUR'01)*, volume 2154 of *Lecture Notes in Computer Science*, pages 536–550. Springer, 2001.

[11] E. Grädel, W. Thomas, and Th. Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.

[12] Th. A. Henzinger. Hybrid automata with finite bisimulations. In *Proc. 22nd International Colloquium on Automata, Languages and Programming (ICALP'95)*, volume 944 of *Lecture Notes in Computer Science*, pages 324–335. Springer, 1995.

[13] Th. A. Henzinger. The theory of hybrid automata. In *Proc. 11th Annual Symposim on Logic in Computer Science (LICS'96)*, pages 278–292. IEEE Computer Society Press, 1996.

[14] Th. A. Henzinger, B. Horowitz, and R. Majumdar. Rectangular hybrid games. In *Proc. 10th International Conference on Concurrency Theory (CONCUR'99)*, volume 1664 of *Lecture Notes in Computer Science*, pages 320–335. Springer, 1999.

[15] Th. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? *Journal of Computer and System Sciences*, 57(1):94–124, 1998.

[16] W. Hodges. *A Shorter Model Theory*. Cambridge University Press, 1997.

[17] M. V. Korovina and N. Vorobjov. Pfaffian hybrid systems. In *Proc. 18th International Workshop on Computer Science Logic (CSL'04)*, volume 3210 of *Lecture Notes in Computer Science*, pages 430–441. Springer, 2004.

[18] G. Lafferriere, G. J. Pappas, and S. Sastry. O-minimal hybrid systems. *Mathematics of Control, Signals, and Systems*, 13(1):1–21, 2000.

[19] A. Pillay and C. Steinhorn. Definable sets in ordered structures. *Transactions of the American Mathematical Society*, 295(2):565–592, 1986.

[20] L. Van den Dries. O-minimal structures. In *Logic: From Foundations to Applications*, Oxford Science Publications, pages 137–185. Oxford University Press, 1996.

[21] L. van den Dries. *Tame Topology and O-Minimal Structures*, volume 248 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1998.

[22] H. Wong-Toi. The synthesis of controllers for linear hybrid automata. *Proceedings of IEEE Conference on Decision and Control*, pages 4607–4612, 1997.