Rend. Istit. Mat. Univ. Trieste
Volume 42 (2010), 147–163.

Hybrid Automata and Bisimulations

Alberto Casagrande

Abstract. *This paper surveys hybrid automata and bisimulation relations. We formally introduce both notions and briefly present the model checking problem over hybrid automata. We show how, in some cases, bisimulations can be used to quotient infinite state systems to finite ones and, hence, we reduce the model checking over hybrid automata to model checking over finite models. Finally, we review some classes of hybrid automata which admit finite bisimulation quotients.*

Keywords: Hybrid Systems, Bisimulation, Model Checking.
MS Classification 2010: 34A38 , 68Q05, 03B45

1. Introduction

Hybrid systems are systems exhibiting a mixed discrete-continuous behaviour which cannot be described in a proper way using either discrete or continuous models. Such systems consist of a discrete program within a continuously changing environment and they are very common in many fields such as automotive, where car engine's physics are ruled by four different phases, or control theory, in the case of digital devices designed to control continuous phenomena. Moreover, hybrid systems are even present in less "canonical" contexts in which the discrete program is not artificial. In particular, we can find hybrid systems in biology where substance concentrations of a living organism are ruled by continuous laws which change according to a phase cycle.

The notion of *hybrid automaton* was introduced to model hybrid systems (see e.g., [1]). Intuitively, a hybrid automaton is a "finite-state" automaton with continuous variables which evolve according to a set of continuous laws, called *dynamics*, characterising each discrete *location*. Hybrid automaton semantics can be given as a labelled transition system over an infinite set of states.

Once hybrid automata have been chosen as formalism to describe hybrid systems, one may want to use them to establish properties of the systems themselves. Model checking is a technique used to automatically investigate formal models and prove properties expressed by peculiar classes of logics such as temporal logics or μ-calculi. Although it was initially introduced to handle finite Kripke structures only, whenever we can encode the original model within such formalisms, we can apply the almost classical algorithms presented in the literature and deduce system properties such as safety or liveness.

In the following sections, we briefly introduce Kripke structures, temporal logics, and model checking, we describe bisimulation relations and formalize hybrid automata, and, finally, we suggest how bisimulation can be used to investigate hybrid automata and review some of the classes of such formalism which admit a finite bisimulation quotient.

2. Kripke Structures and Temporal Logics

In order to verify the behaviour of a system, we need both a formal model of it and a way to indicate its properties. *Kripke structures* and *temporal logics* are used to formally specify a system and to denote the system's properties, respectively. Roughly, a Kripke structure is a labelled transition system whose states are labelled through a set of propositional symbols.

Definition 2.1 (Labelled Transition System). *A* labelled transition system *is a tuple* $\langle \mathcal{Q}, \Delta, \rightarrow \rangle$ *such that:*

- $\mathcal{Q}$ *is a non empty* set of states*;*
- Δ *is a set of* edge's labels*;*
- $\rightarrow$ *is a* transition relation *on* $\mathcal{Q}$ *and* Δ*. Specifically,* $\rightarrow \subseteq \mathcal{Q} \times \Delta \times \mathcal{Q}$*. Any element in* $\rightarrow$ *is called* edge *or* arc *of the structure. We will write* $\ell \xrightarrow{\alpha} \ell'$ *meaning* $\langle \ell, \alpha, \ell' \rangle \in \rightarrow$*.*

We will use $\mathcal{S}$ to indicate a set of *propositional symbols* and, for any set X, 2^X to indicate the powerset of X.

Definition 2.2 (Kripke Structure [6]). *A* Kripke structure*,* $\mathcal{K}$*, is a tuple* $\mathcal{K} = \langle \mathcal{Q}, \Delta, \rightarrow, \mathcal{LS} \rangle$ *where:*

- $\langle \mathcal{Q}, \Delta, \rightarrow \rangle$ *is a labelled transition system;*
- $\mathcal{LS} \colon \mathcal{Q} \mapsto 2^{\mathcal{S}}$ *is a* labeling function *which tags each state* ℓ *with a set of propositional symbols.* $\mathcal{LS}(\ell)$ *is the set of all the symbols true in* ℓ*.*

Without loss of generality, we may assume that transition relations are total i.e., for every $\ell \in \mathcal{Q}$ there exists a $\ell' \in \mathcal{Q}$ such that $\ell \rightarrow \ell'$.

Temporal logics (see, for instance, [20, 10]) may be used to denote Kripke structure's properties. Such formalisms extend the propositional logic and express properties of transition sequences. In particular, besides using atomic propositions and traditional Boolean connectives, temporal logics may specify time properties such as: "property p will *eventually* hold", "*from now on* property q will hold" or "property r will *never* hold". These time properties are described using specific *temporal operators* and *path quantifiers* which depend on the particular temporal logic adopted. Examples of temporal logics are

*Computation Tree Logic-** (CTL*) [10], *Computation Tree Logic* (CTL) [10], and *Linear Temporal Logic* (LTL) [20].

EXAMPLE 2.3 (Computation Tree Logic-* [10]). *Let* $\mathcal{K} = \langle \mathcal{Q}, \Delta, \rightarrow, \mathcal{LS} \rangle$ *be a Kripke structure. There are two kinds of CTL*-formulæ over* $\mathcal{K}$*:* state formulæ*, which hold in a particular state of* $\mathcal{K}$*, and* path formulæ*, which hold along a* $\mathcal{K}$*'s path. A state formula has one of the following forms:*

- p*, where* $p \in \mathcal{S}$*;*
- $\neg\varphi$*,* $\varphi \vee \psi$*, or* $\varphi \wedge \psi$*, where both* φ *and* ψ *are state formulæ;*
- $\mathtt{E}\varphi$ *or* $\mathtt{A}\varphi$*, where* φ *is a state formula.*

The syntax of path formulæ is detailed in the following rules:

- *if* φ *is a state formula, then* φ *is also a path formula;*
- *if* φ *and* ψ *are path formulæ, then* $\neg\varphi$*,* $\varphi \vee \psi$*,* $\varphi \wedge \psi$*,* $\circ\varphi$*,* $\Diamond\varphi$*,* $\Box\varphi$*,* $\psi \mathcal{U} \varphi$*, and* $\psi \mathcal{R} \varphi$ *are path formulæ.*

We now give the semantics of CTL with respect to* $\mathcal{K}$*. We use the symbols* π *and* π^i *to denote a path of* $\mathcal{K}$ *and the suffix of* π *beginning with the state* $q_i \in \mathcal{Q}$*, respectivelly. If* φ_1 *and* φ_2 *are state formulæ and* ψ_1 *and* ψ_2 *are path formulæ, then the relation* $\models$ *is inductively defined as follows:*

$\mathcal{K}, \ell \models p$	$\iff$	$p \in \mathcal{LS}(\ell)$
$\mathcal{K}, \ell \models \neg\varphi_1$	$\iff$	$\mathcal{K}, \ell \models \varphi_1$ *does not hold*
$\mathcal{K}, \ell \models \varphi_1 \vee \varphi_2$	$\iff$	*either* $\mathcal{K}, \ell \models \varphi_1$ *or* $\mathcal{K}, \ell \models \varphi_2$
$\mathcal{K}, \ell \models \mathtt{E}\psi_1$	$\iff$	*there exists a path* π *issuing from* ℓ *such that* $\mathcal{K}, \pi \models \psi_1$
$\mathcal{K}, \pi \models \varphi_1$	$\iff$	*the first state* ℓ *of* π *is such that* $\mathcal{K}, \ell \models \varphi_1$
$\mathcal{K}, \pi \models \neg\psi_1$	$\iff$	$\mathcal{K}, \pi \models \psi_1$ *does not hold*
$\mathcal{K}, \pi \models \psi_1 \vee \psi_2$	$\iff$	*either* $\mathcal{K}, \pi \models \psi_1$ *or* $\mathcal{K}, \pi \models \psi_2$
$\mathcal{K}, \pi \models \circ\psi_1$	$\iff$	$\mathcal{K}, \pi^1 \models \psi_1$
$\mathcal{K}, \pi \models \psi_1 \mathcal{U} \psi_2$	$\iff$	*there exists a* k *such that* $\mathcal{K}, \pi^k \models \psi_2$ *and for all* $j < k$ $\mathcal{K}, \pi^j \models \psi_1$

The formulæ $\phi_1 \wedge \phi_2$*,* $\psi_1 \mathcal{R} \psi_2$*,* $\Diamond\psi$*,* $\Box\psi$*, and* $\mathtt{A}\varphi$ *are shortcuts for* $\neg(\neg\phi_1 \vee \neg\phi_2)$*,* $\neg(\neg\psi_1 \mathcal{U} \neg\psi_2)$*,* $\top \mathcal{U} \psi$*,* $\neg\Diamond\neg\psi$*, and* $\neg\mathtt{E}\neg\varphi$*, respectivelly.*

Given a Kripke structure $\mathcal{K}$, a state ℓ of $\mathcal{K}$, and a temporal formula φ, the problem of deciding whether $\mathcal{K}, \ell \models \varphi$ holds or not, i.e., whether φ hold or not in ℓ, is known in the literature as the *model checking problem* [8]. Despite this problem not being always decidable, many techniques and algorithms have been developed to answer specific instances of it; for the sake of example, there exist effective methods to solve CTL* model checking over finite Kripke structures [7, 8].

3. Bisimulation Relations

The notion of *bisimulation* has been introduced in many fields with different purposes (see, e.g., [21, 19, 11, 18, 9]). For instance, van Benthem proposed it as an equivalence principle between structures [21].

Roughly, a Kripke structure $\mathcal{K}$ bisimulates a Kripke structure $\mathcal{K}'$, if every behaviour of $\mathcal{K}'$ can be matched by $\mathcal{K}$ and vice versa.

DEFINITION 3.1 (Bisimulation Relation). *Let $\mathcal{K} = \langle \mathcal{Q}, \Delta, \rightarrow, \mathcal{LS} \rangle$ and $\mathcal{K}' = \langle \mathcal{Q}', \Delta, \rightarrow', \mathcal{LS}' \rangle$ be two structures. A relation $B \subseteq \mathcal{Q} \times \mathcal{Q}'$ is a* bisimulation relation *between $\mathcal{K}$ and $\mathcal{K}'$ if and only if, for all $\langle \ell, \ell' \rangle \in B$:*

- $\mathcal{LS}(\ell) = \mathcal{LS}'(\ell')$;
- *for all $\tilde{\ell} \in \mathcal{Q}$ and $\alpha \in \Delta$ such that $\ell \xrightarrow{\alpha} \tilde{\ell}$, there exists a $\tilde{\ell}'$ such that $\ell' \xrightarrow{\alpha}{}' \tilde{\ell}'$ and $\langle \tilde{\ell}, \tilde{\ell}' \rangle \in B$;*
- *for all $\tilde{\ell}' \in \mathcal{Q}'$ and $\alpha \in \Delta$ such that $\ell' \xrightarrow{\alpha}{}' \tilde{\ell}'$, there exists a $\tilde{\ell}$ such that $\ell \xrightarrow{\alpha} \tilde{\ell}$ and $\langle \tilde{\ell}, \tilde{\ell}' \rangle \in B$.*

If there exists a bisimulation relation B between $\mathcal{K}$ and $\mathcal{K}'$ and $\langle \ell, \ell' \rangle \in B$ then we say that ℓ and ℓ' are *bisimilar* and we write $\ell \approx \ell'$. By extension, if there exists a bisimulation relation B between $\mathcal{K}$ and $\mathcal{K}'$, then we say that $\mathcal{K}$ and $\mathcal{K}'$ are *bisimilar* and we write $\mathcal{K} \approx \mathcal{K}'$.

It is easy to prove that the reflexive, symmetric, and transitive closure of any bisimulation is a bisimulation. The following lemma characterizes the relation "*to be bisimilar to*", denoted by $\approx$.

LEMMA 3.2 (From [17]). *The relation $\approx$, restricted to the states of a Kripke structure $\mathcal{K}$, is an equivalence and it is the maximal bisimulation between the states of $\mathcal{K}$, i.e., if B is a bisimulation between the states of $\mathcal{K}$, then $B \subseteq \approx$.*

Bisimulation equivalence preserves CTL*-properties: if two Kripke structures are bisimulation equivalent, then they satisfy the same CTL* formulæ.

THEOREM 3.3 (From [19]). *Let $\mathcal{K} = \langle \mathcal{Q}, \Delta, \rightarrow, \mathcal{LS} \rangle$ and $\mathcal{K}' = \langle \mathcal{Q}', \Delta, \rightarrow', \mathcal{LS}' \rangle$ be two structures. For all CTL* formulæ φ with atomic propositions in $\mathcal{S}$ and for all $\ell \in \mathcal{Q}$ and $\ell' \in \mathcal{Q}'$ such that $\ell \approx \ell'$, $\mathcal{K}', \ell' \models \varphi$ if and only if $\mathcal{K}, \ell \models \varphi$.*

Let $\mathcal{K} = \langle \mathcal{Q}, \Delta, \rightarrow, \mathcal{LS} \rangle$ be a Kripke structure. Since $\approx$ restricted to $\mathcal{Q}$ is an equivalence relation, we can build the structure $\mathcal{K}_{\approx} \stackrel{\text{def}}{=} \langle \mathcal{Q}_{\approx}, \Delta, \rightarrow_{\approx}, \mathcal{LS}_{\approx} \rangle$, where $\mathcal{Q}_{\approx} \stackrel{\text{def}}{=} \{[\ell]_{\approx} | \ell \in \mathcal{Q}\}$, $[q]_{\approx}$ is the equivalence class of $\approx$ containing q, $\rightarrow_{\approx} \stackrel{\text{def}}{=} \{\langle [\ell]_{\approx}, \sigma, [\ell']_{\approx} \rangle | \langle \ell, \sigma, \ell' \rangle \in \rightarrow\}$, and $\mathcal{LS}_{\approx}([\ell]_{\approx}) \stackrel{\text{def}}{=} \mathcal{LS}(\ell)$. Whereas $\ell' \in [\ell]_{\approx}$ if and only if $\ell \approx \ell'$, $\mathcal{LS}(\ell) = \mathcal{LS}(\ell')$ for all $\ell' \in [\ell]_{\approx}$ by definition of bisimulation and $\mathcal{K}_{\approx}$ is well defined. The Kripke structure $\mathcal{K}_{\approx}$ is the *bisimulation quotient*, or *quotient by maximum bisimulation*, of $\mathcal{K}$.

It is easy to see that the relation $\{\langle \ell, [\ell]_{\approx}\rangle | \ell \in \mathcal{Q}\}$ is a bisimulation between a Kripke structure $\mathcal{K} = \langle \mathcal{Q}, \Delta, \rightarrow, \mathcal{LS}\rangle$ and its quotient by maximal bisimulation $\mathcal{K}_{\approx}$ and, by Theorem 3.3, that $\mathcal{K}$ and $\mathcal{K}_{\approx}$ are CTL*-equivalent, i.e., they satisfy the same CTL* formulæ. In particular, this is true even if the original Kripke structure $\mathcal{K}$ has an unbounded number of states and the state space of its bisimulation quotient is finite.

4. Hybrid Automata

Hybrid automata were introduced in [16, 1] as models for hybrid systems. In order to define such a formalism, we first need to introduce some conventions. Capital letters Z, Z', Z_m, and ${Z_m}'$, where $m \in \mathbb{N}$, denote variables ranging over $\mathbb{R}$. Analogously, $\mathbf{Z}$ denotes the vector of variables $\langle Z_1, \ldots, Z_d\rangle$ and $\mathbf{Z}'$ denotes the vector $\langle {Z_1}', \ldots, {Z_d}'\rangle$. The temporal variables T, T', T_0,$\ldots$, T_n model time and range over $\mathbb{R}_{\geq 0}$. We use the small letters p, q, r, s, $\ldots$ to denote vectors of real numbers. Occasionally, we write $\varphi[Z_1, \ldots, Z_m]$ to stress the fact that the set of free variables of the first-order formula φ is included in the set of variables $\{Z_1, \ldots, Z_m\}$. Given a formula $\varphi[\mathbf{Z_1}, \ldots, \mathbf{Z_i}, \ldots, \mathbf{Z_n}]$ and a vector p having the same dimension as $\mathbf{Z_i}$, the formula obtained by component-wise substitution of $\mathbf{Z_i}$ with p is denoted by $\varphi[\mathbf{Z_1}, \ldots, \mathbf{Z_{i-1}}, p, \mathbf{Z_{i+1}}, \ldots, \mathbf{Z_n}]$. When the only free variables in φ are the components of $\mathbf{Z_i}$, after the substitution we can determine the truth value of $\varphi[p]$.

We are now ready to define hybrid automata. For each node of a graph, we have an invariant condition and a dynamic law. The dynamic law may depend on the initial conditions, i.e., on the values of the continuous variables at the beginning of the evolution in the node. Jumps from one discrete location to another are regulated by activation conditions and reset maps.

Definition 4.1 (Hybrid Automata - Syntax). *A* hybrid automaton H *having dimension* $d(H) \in \mathbb{N}$ *is a tuple* $(\mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inv, f., Act, Res)$ *where:*

- $\mathbf{Z} = \langle Z_1, \ldots, Z_{d(H)}\rangle$ *and* $\mathbf{Z}' = \langle {Z_1}', \ldots, {Z_{d(H)}}'\rangle$ *are two vectors of variables ranging over the reals* $\mathbb{R}$*;*
- $\langle \mathcal{V}, \mathcal{E}\rangle$ *is a graph. Each element of* $\mathcal{V}$ *will be dubbed* location *or* mode*;*
- *Each vertex* $v \in \mathcal{V}$ *is labelled by both a formula* $Inv(v)[\mathbf{Z}]$*, called* invariant*, and a function* $f_v : \mathbb{R}^{d(H)} \longrightarrow (\mathbb{R}_{\geq 0} \longrightarrow \mathbb{R}^{d(H)})$*, called* dynamics*. The dynamics may be specified either by differential equations, i.e.,* f_v *is the solution of a given Cauchy problem, or by a logic formula. We use the formula* $Dyn(v)[\mathbf{Z}, \mathbf{Z}', T]$ *to denote the dynamics on* v *i.e.,* $Dyn(v)[\mathbf{Z}, \mathbf{Z}', T] \stackrel{def}{=} \mathbf{Z}' = f_v(\mathbf{Z})(T)$*;*
- *Each edge* $e \in \mathcal{E}$ *is labelled by the formulæ* $Act(e)[\mathbf{Z}]$ *and* $Res(e)[\mathbf{Z}, \mathbf{Z}']$ *which are called* activation *and* reset*, respectively.*

Let us notice that, whenever the dynamics are given as differential equations, they benefit from the properties of the differential equations themselves. For instance, the solutions of a vector field are derivable and transitive.

If all the defining formulæ belong to the same logical theory $\mathcal{T}$, then we say that the hybrid automaton is *definable in* $\mathcal{T}$ or that it is a $\mathcal{T}$ *hybrid automaton*.

Whenever a reset does not depend on the $\mathbf{Z}$ interpretation (i.e., for any p, q, and p' in $\mathbb{R}^{d(H)}$, if both $Res(e)[p, q]$ holds, then $Res(e)[p', q]$ holds too), we say that the reset is *constant*.

We present hybrid automaton semantics as transition systems: given an initial state, we can deduce the evolution of a hybrid automaton by iteratively applying the transition relation which is associated to the automaton itself. Since hybrid automata have a double nature, the transition systems defining their semantics contains two different transition relations: the *continuous reachability transition relation* and the *discrete reachability transition relation*.

Definition 4.2 (Hybrid Automaton - Semantics). *A* state ℓ *of* H *is a pair* $\langle v, r\rangle$*, where* $v \in \mathcal{V}$ *is a location and* $r \in \mathbb{R}^{d(H)}$ *is an assignment of values for the variables of* $\mathbf{Z}$*. A state* $\langle v, r\rangle$ *is said to be* admissible *if* $Inv(v)\,[r]$ *holds.*

The continuous reachability transition relation $\xrightarrow{t}_C$ *between states, with* $t \geq 0$ *denoting the transition elapsed time, is defined as follows:*

$$\langle v, r\rangle \xrightarrow{t}_C \langle v, s\rangle \iff \begin{array}{l} r = f_v(r)(0),\ s = f_v(r)(t),\ f_v(r) \text{ is continuous} \\ \text{in } [0, t], \text{ and } Inv(v)\,[f_v(r)(t')] \text{ hold for each } t' \in \\ [0, t]. \text{ In such a case, } f_v(r) \text{ is called flow function.} \end{array}$$

The discrete reachability transition relation $\xrightarrow{e}_D$ *among admissible states is defined as follows:*

$$\langle v, r\rangle \xrightarrow{e}_D \langle u, s\rangle \iff \begin{array}{l} e \in \mathcal{E}, \text{ with } v \text{ and } u \text{ source and destination of } e, \\ \text{respectively, and } Act(e)[r] \text{ and } Res(e)[r, s] \text{ hold.} \end{array}$$

We write $\ell \to_C \ell'$ and $\ell \to_D \ell'$ to mean that there exists a $t \in \mathbb{R}_{\geq 0}$ such that $\ell \xrightarrow{t}_C \ell'$ and that there exists an $e \in \mathcal{E}$ such that $\ell \xrightarrow{e}_D \ell'$, respectively.

Building upon a combination of both continuous and discrete transitions, we can formulate the notion of *reachability*.

Definition 4.3 (Hybrid Automata - Reachability). *The hybrid automaton* H reaches *a state* ℓ_n *from a state* ℓ_0 *if there exists a sequence of admissible states* $\ell_0, \ldots, \ell_n$ *such that* $\ell_{i-1} \to \ell_i$ *holds for all* $i \in [1, n]$ *and either* $\ell_{i-2} \to_C \ell_{i-1} \to_D \ell_i$*,* $\ell_{i-2} \to_D \ell_{i-1} \to_D \ell_i$*, or* $\ell_{i-2} \to_D \ell_{i-1} \to_C \ell_i$ *for all* $i \in [2, n]$[1]*. In such a case, we also say that* ℓ_n *is* reachable *from* ℓ_0 *in* H*.*

The problem of deciding whether a hybrid automaton H reaches a set of states T from a second set of states S is known as the *reachability problem* of T from S over H. There exist hybrid automata over which the reachability problem is not decidable [1].

[1]This last condition supports not transitive dynamics. See [5] for a complete discussion.

5. Hybrid Automata and Model Checking

In order to automatically verify properties of a hybrid automaton, one may be tempted to consider classical model checking techniques (see e.g., [8]). Unfortunately, this aim is suddenly frustrated because such techniques can be applied only to finite Kripke structures. We can obtain a Kripke structure from a hybrid automaton by considering the semantics of the automaton itself together with a function, definable in the same logical theory used to specify activations and invariants, which labels the system' states with sets of opportune propositional symbols (e.g., *"admissible state"* or *"unwanted state"*). The Kripke structure obtained in such a way has as many states as the original hybrid automaton and, since hybrid automata are infinite state models by definition, the standard model checking techniques cannot be directly applied in this context.

Many authors suggested the use of equivalence reductions based on relations such as simulation or bisimulation. Given a hybrid automaton, we can deduce the Kripke structure corresponding to its semantics and we may try to reduce the state space of it by computing the quotient by maximal bisimulation of the structure itself. Since such quotient is bisimilar to the original structure, they satisfy the same CTL*-formulæ. By Theorem 3.3, whenever the bisimulation quotient of a hybrid automaton is finite, we can verify CTL* properties of the system by applying finite model checking techniques to its quotient.

Since time domain is dense, the labels in the continuous transition system defining the semantics of any hybrid automaton are infinite in number and any quotient preserving them has an infinite set of states. Thus, we have the chance to shrink state set exclusively by abstracting the time and bisimulations may reduce to finite structures only time-abstract semantics i.e., $\langle \mathcal{V} \times \mathbb{R}^n, \rightarrow, \mathcal{E} \cup \{\bot\} \rangle$ where $\stackrel{e}{\rightarrow}$ is $\stackrel{e}{\rightarrow}_D$, if $e \in \mathcal{E}$, and it is $\rightarrow_C$ otherwise. We call such relations *time-abstract bisimulations*. In the context of hybrid systems, the terms bisimulation and bisimulation quotient are usually synonymous with time-abstract bisimulation and time-abstract bisimulation quotient.

6. Hybrid Automata and Finite Bisimulation Quotient

Any hybrid automaton reduced by time-abstract bisimulation preserves the original reachability properties. However, though there exists an effective way to establish the reachability on finite transition systems, the reachability problem over hybrid automata is not always decidable [1]. It follows that not all hybrid automata admit a finite time-abstract bisimulation quotient.

In this section, we report about some interesting classes of hybrid automata whose time-abstract bisimulation quotient is finite. If we are able to compute their quotients, and this is not always the case, then we can reduce them and apply classical model checking techniques for finite Kripke structures.

6.1. Timed Automata

Timed automata are hybrid automata whose variables represent time clocks. When a timed automaton crosses an edge, each variable can either be reset to zero or maintain its value. The following definition formalizes timed automata.

DEFINITION 6.1 (Timed Automaton [2]). *A* timed automaton H *is a hybrid automaton such that for each* $v \in \mathcal{V}$, $e \in \mathcal{E}$, *and variables* Z_i *and* Z_j*:*

- *the dynamics of* Z_i *on* v *are* $\dot{Z}_i = 1$*;*
- *Res*(e) *either does not change the value of* Z_i *or resets* Z_i *to* 0*;*
- *both Inv*$(v)[\mathbf{Z}]$ *and Act*$(e)[\mathbf{Z}]$ *are Boolean combinations of terms of the form either* $Z_i \asymp c$ *or* $Z_i - Z_j \asymp c$ *where* $c \in \mathbb{Q}$ *and* $\asymp \in \{<, \leq, =, \geq, >\}$.

All timed automata have a finite time-abstract bisimulation quotient [2]. In order to prove such a statement, we build a finite-index equivalence relation $\sim_H$ over the interpretations of the variables of a generic timed automaton H. For each variable Z_i, let c_i be the largest constant in the terms of H containing Z_i as well. For any $x, y \in \mathbb{R}^{d(H)}$, $x \sim_H y$ if and only if for all $i, j \in [1, d(H)]$:

- either $\lfloor x_i * d \rfloor = \lfloor y_i * d \rfloor$ or both $x_i > c_i$ and $y_i > c_i$,
- if $x_i \leq c_i$ and $x_j \leq c_j$, then fract $(x_i * d) \leq$ fract $(x_j * d)$ iff fract $(y_i * d) \leq$ fract $(y_j * d)$,
- if $x_i \leq c_i$, then fract $(x_i * d) = 0$ iff fract $(y_i * d) = 0$,

where x_i is the i-th component of x, fract (x) is the fractional part of x, and d is the least common denominator of all the c_i's. The relation $\sim_H$ is an equivalence and the classes induced by $\sim_H$ are dubbed *clock regions* for H.

The number of clock regions is upper bounded by $6^{d(H)} \Pi_{i=1}^{d(H)} (c_i + 1)$ and, hence, $\sim_H$ has finite index for any timed automaton H.

The relation $\simeq_H = \{\langle \langle v, r \rangle, \langle v, r' \rangle \rangle \mid r \sim_H r'\}$ is a bisimulation for H. As a matter of fact, invariants and activations of H cannot discriminate two variable interpretations in the same clock region. Moreover, the dynamics are such that, whenever two values $x, y \in \mathbb{R}^{d(H)}$ lay in the same clock region and there exists a $x' \in \mathbb{R}^{d(H)}$ such that $\langle v, x \rangle \rightarrow_C \langle v, x' \rangle$, there should exists a $y' \in \mathbb{R}^{d(H)}$ such that $\langle v, y \rangle \rightarrow_C \langle v, y' \rangle$ and $y \sim_H y'$. Thus, $\simeq_H$ is a bisimulation with respect to $\rightarrow_C$ and, since each variable can be reset to either 0 or identity, $\simeq_H$ is a bisimulation with respect to $\xrightarrow{e}_D$ too. It follows that $\simeq_H$ is a bisimulation and a finite-index equivalence relation between pairs of states of H.

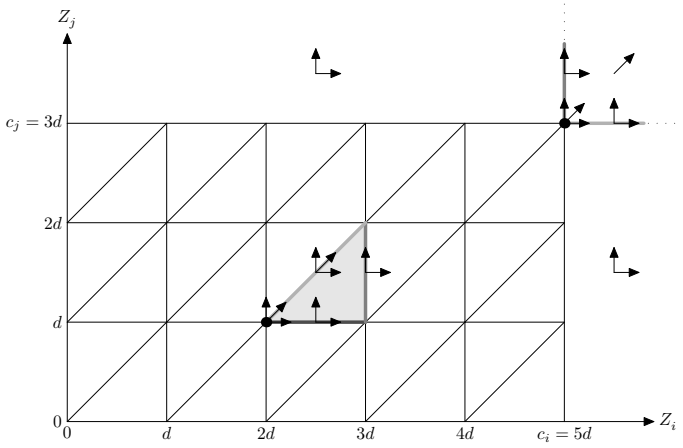

Figure 1: A two dimensional projection of timed automaton clock regions. Arrows over clock regions describe possible continuous evolutions.

6.2. Multirate Automata

Timed automaton variables correspond to clocks and their derivatives are set to 1. In *multirate automata*, variable derivatives may have any rational values.

Definition 6.2 (Multirate Automata [1]). *A* multirate automaton H *is a hybrid automaton such that for each* $v \in \mathcal{V}$, $e \in \mathcal{E}$, *and variables* Z_i *and* Z_j*:*

- *the dynamics of* Z_i *on* v *are* $\dot{Z}_i = c_i$ *where* $c_i \in \mathbb{Q}$ *does not depend on the location;*
- $Res(e)$ *either does not change the value of* Z_i *or resets* Z_i *to* 0*;*
- *both* $Inv(v)[\mathbf{Z}]$ *and* $Act(e)[\mathbf{Z}]$ *are Boolean combinations of terms of the form either* $Z_i \asymp c$ *or* $Z_i - Z_j \asymp c$ *where* $c \in \mathbb{Q}$ *and* $\asymp \in \{<, \leq, =, \geq, >\}$.

Example 6.3 (From [1]). *The water level of a sump should remain between* 1 *and* 12 *centimeters. For such a reason, a monitor continuously controls and regulates the water height by switching on and off a pump. When the pump is off, the water level decreases by* 2 *centimeters per second, while, when it is on, the water grows by* 1 *centimeter per second. The switching time is not instantaneous and the pump takes* 2 *seconds to turn on or off. For such a reason, the monitor should turn on the pump before the water height reaches* 1 *centimeters and turn off it before the water level raises over* 12 *centimeters.*

Figure 2 depicts the described monitor. The two variables Z_1 *and* Z_2 *represent the elapsed time from the last switch and the water level, respectively.*

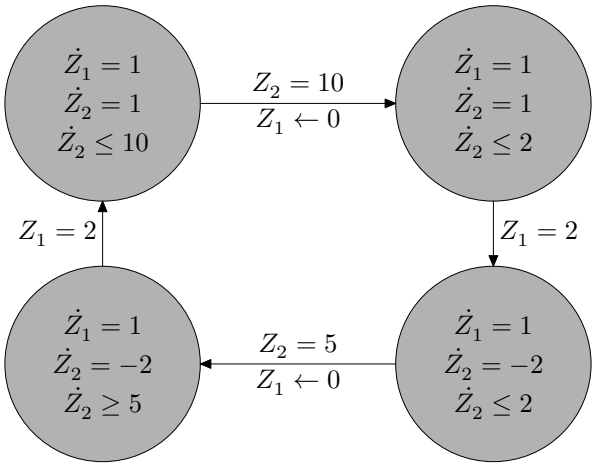

Figure 2: A water-level monitor.

It is known that the reachability problem for multirate automata is undecidable [1] and, hence, they do not admit finite time-abstract bisimulation quotient. However, such a result is based on one's ability to compare variables having different dynamics. Finite bisimulation quotient can be achieved nevertheless by avoiding comparisons between variables. *Simple multirate automata* have been introduced in [1] for such a purpose.

Definition 6.4 (Simple Multirate Automaton [1]). *A multirate automaton is* simple *if, for all locations v and edges e, all the atoms in the formulæ $Inv(v)$, $Act(e)$, and $Res(e)$ are of the form $Z_i \leq c_i$ or $c_i \leq Z_i$, where $c_i \in \mathbb{Q}$.*

Simple multirate automata can be encoded into timed automata by both adjusting variable derivatives to 1 and syntactically replacing all the occurrences of any variable Z_i in $Inv(v)$, $Act(e)$, and $Res(e)$ with $c_i * Z_i$. The obtained hybrid automaton is a timed automaton whose transitions mimic the transitions of the corresponding simple multirate automaton.

Theorem 6.5 (From [1]). *All the simple multirate automata admit finite time-abstract bisimulation quotients.*

Proof. Let $H = (\mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inv, f., Act, Res)$ be a simple multirate automaton such that the dynamics of Z_i are $\dot{Z}_i = c_i$. Let us consider the automaton $H' = (\mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inv', f'., Act', Res')$ such that, for any location $v \in \mathcal{V}$, edge $e \in \mathcal{E}$, and variable Z_i:

- the dynamics of Z_i in v are $\dot{Z}_i = 1$;

- the formulæ $Inv'(v)$, $Act'(e)$, and $Res'(e)$ are obtained from $Inv(v)$, $Act(e)$, and $Res(e)$, respectively, by syntactically replacing any occurrence of Z_i with $c_i * Z_i$.

The automaton H' is timed automaton. If $r = \langle r_1, \ldots, r_n \rangle$, $p = \langle p_1, \ldots, p_n \rangle$, $c = \langle c_1, \ldots, c_n \rangle$, and $c * p = \langle c_1 * p_1, \ldots, c_n * p_n \rangle$, then $\langle v, r \rangle \xrightarrow{t}_C \langle v, p \rangle$ in H if and only if $\langle v, c * r \rangle \xrightarrow{t}_C \langle v, c * p \rangle$ in H'. Analogously, $\langle v, r \rangle \xrightarrow{e}_D \langle v, p \rangle$ in H if and only if $\langle v, k * r \rangle \xrightarrow{e}_D \langle v, k * p \rangle$ in H'. It follows that B is a bisimulation between H's states if and only if $B' = \{\langle \langle v, c * p \rangle, \langle v', c * r \rangle \rangle \mid \langle \langle v, p \rangle, \langle v', r \rangle \rangle \in B\}$ is a bisimulation between H''s states. Since timed automata admit time-abstract finite quotient bisimulation, so do multirate automata. □

6.3. O-minimal Hybrid Automata

Both timed automata and simple multirate automata exhibit very simple dynamics and, hence, they can hardly be used to model complex phenomena. In order to bypass such a limitation, we can restrict resets to constant maps and focus on the theory used to express the dynamics themselves.

An interesting class of theories is the class of *O-minimal theories*.

Definition 6.6 (O-Minimal Theory [22]). *Let $\mathcal{L}$ be a first-order language whose set of relational symbols includes a binary symbol $<$ and let $\mathcal{M}$ be a model of $\mathcal{L}$ in which $<$ is interpreted as a linear order and whose support is M. The theory $\mathcal{T}(\mathcal{M})$ is* order minimal*, or simply* O-minimal*, if every subset of M definable in $\mathcal{T}(\mathcal{M})$ is a union of finitely many points and intervals (with respect to $<$).*

O-minimal theories include theories such as $\langle \mathbb{R}, 0, 1, +, *, < \rangle$, also known as Tarski theory, $\langle \mathbb{R}, 0, 1, +, *, e^x, < \rangle$, which augments the Tarski theory with the exponential function e^x, and $\langle \mathbb{R}, 0, 1, +, *, (f)_{f \in an}, < \rangle$, which is $\langle \mathbb{R}, 0, 1, +, *, < \rangle$ extended with the set of all the real-analytic functions from $[-1, 1]^n$ to $\mathbb{R}$.

O-minimal hybrid automata are hybrid automata whose invariants, dynamics, resets, and activations are definable in the same O-minimal theory and whose resets are constant.

Definition 6.7 (O-minimal Hybrid Automaton [15]). *An* O-minimal hybrid automaton *is a $\mathcal{T}$ hybrid automaton such that $\mathcal{T}$ is O-minimal, $Res(e)$ is constant for all $e \in \mathcal{E}$, and whose dynamics are given as vector fields.*

Example 6.8. *An O-minimal hybrid automaton can be used to model a simple thermostat. Two discrete locations represents the two states of the thermostat (i.e., "heater on" and "heater off") and dynamics depict the temperature evolution in each of such two states. Whenever the temperature reaches 15 degrees the heater is activated, while, if the temperature rises up to 20 degrees, the*

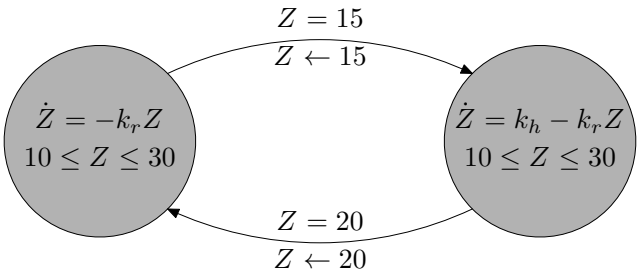

Figure 3: A simple thermostat.

heater is turned off. In the automaton graphical representation depicted in Figure 6.8, the two constants k_r and k_h are the dispersion and heating coefficients, respectively, while the variable Z represents the room temperature.

Every O-minimal hybrid automaton admits a finite bisimulation quotient. Let us notice that the existence of a finite bisimulation quotient does not imply that such a quotient is computable: there may be O-minimal hybrid automata whose maximal bisimulation is not computable at all.

Since the resets of any O-minimal hybrid automaton are constants, it admits a finite bisimulation quotient if and only if all the time-abstract transition systems induced by the continuous evolutions in each locations do the same. In order to prove such property, we first need to introduce the notion of *cell* of a theory $\mathcal{T}$ or, simply, $\mathcal{T}$-cell.

DEFINITION 6.9 ($\mathcal{T}$-Cell [14]). *Let $\mathcal{T}$ be a theory which interprets $<$ as a linear order and whose support is M. By induction on n, a* $\mathcal{T}$-cell *in M^n is:*

$n = 1$**:** *either a singleton $\{r\}$, where $r \in M$ is definable in $\mathcal{T}$, or a $\mathcal{T}$-definable open interval $(a, b) \subseteq M$;*

$n > 1$**:** *one of the following sets $C_f = \{\langle x_1, \ldots, x_{n-1}, r\rangle \,|\, f(x) = r\}$, $C \uparrow^f = \{\langle x_1, \ldots, x_{n-1}, r\rangle \,|\, f(x) < r\}$, $C \downarrow_f = \{\langle x_1, \ldots, x_{n-1}, r\rangle \,|\, r < f(x)\}$, or $C \updownarrow_g^f = C \uparrow^f \cap C \downarrow_g$ where $C \subseteq M^{n-1}$ is a $\mathcal{T}$-cell and $f, g : C \to M$ are continuous functions definable in $\mathcal{T}$ such that $f < g$.*

The following result has been given in [14].

THEOREM 6.10 (Cell Decomposition). *Let $\mathcal{T}$ be an O-minimal theory whose support is M. For any finite collection $\{A_1, \ldots, A_l\}$ of subsets of M^n definable in $\mathcal{T}$, there exists a partition $\{C_1, \ldots, C_m\}$ of M^n such that all the C_i's are $\mathcal{T}$-cells and the finite collection complies with the partition in the sense that all the A_i's are union of some of the classes of the partition.*

Since dynamics are expressed as vector fields and they are autonomous, we can deduce from Theorem 6.10 that any O-minimal hybrid automaton admits a finite time-abstract bisimulation quotient. As stated above, we may be unable to compute such a quotient and, actually, the computability of it is guaranteed only for automata whose defining theory is decidable [15].

6.4. STORMED Hybrid Automata

In previous subsections, we have reviewed some hybrid automaton classes which admit finite time-abstract bisimulation quotients. We imposed many limitations to either resets or dynamics to achieve such a result. For instance, simple multirate automata admit identity resets, but they are equipped with very simple dynamics. On the contrary, O-minimal hybrid automata have memoryless resets, while their continuous evolutions may be rather complex. All the limitations imposed in previous sections are, in some sense, both local and syntactic, as they can easily be checked by analyzing the syntactic structure of all the formulæ in hybrid automaton definition and do not constrain directly the overall automaton evolutions. Vladimerou *et al.* identify a new class of hybrid automata with finite time-abstract bisimulation quotient by imposing some global constraints on the semantics of both dynamics and resets [23]. Hybrid automata satisfying such constraints are called *STORMED hybrid automata*.

The definition of STORMED imposes a sort of spatial goal for the automaton evolutions and requires the distance from such a goal to decrease monotonically through both continuous flows and discrete jumps. The absence of Zeno behaviors (i.e., evolutions with an infinite number of discrete jumps in a finite amount of time) are guaranteed by the monotonicity and by guard separability. This is enough to ensure finite time-abstract bisimulation quotient.

In order to formalize STORMED automata, we first need some definitions.

Definition 6.11 (Time-Independent Spatially-Consistent). *Given a hybrid automaton H, its dynamics are said to be* time-independent spatially-consistent*, or TISC, if $f_v(r)$ is continuous, $f_v(r)(0) = r$, and $f_v(r)(t + t') = f_v(r')(t')$ for all $v \in \mathcal{V}$, all $r, r' \in \mathbb{R}^{d(H)}$, and all $t, t' \in \mathbb{R}_{\geq 0}$ such that $f_v(r)(t) = r'$.*

Let us notice that TISC property are ensured for hybrid automata whose dynamics are specified as vector fields.

Definition 6.12 (Separable Guards). *Let $H = (\mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inv, f., Act, Res)$ be a hybrid automaton. If there exists a $d \in \mathbb{R}_{>0}$ such that $\|r_1 - r_2\| \geq d$ for all $e_1, e_2 \in \mathcal{E}$ and all $r_1, r_2 \in \mathbb{R}^{d(H)}$ with $Act(e_1)[r_1]$ and $Act(e_2)[r_2]$, then H is said to have* separable guards*. In such a case, H is said to be d-*separable*.*

Separable guards avoid non-deterministic selections between two alternative discrete jumps from the same state and help to avoid Zeno behaviors, i.e., infinite jumps in a finite amount of time.

Definition 6.13 (Monotonic Flows). *Let $H = (\mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inv, f., Act, Res)$ be a hybrid automaton and let $r \in \mathbb{R}^{d(H)}$ be a vector. The flows of H are* monotonic *with respect to r if there exists an $\epsilon \in \mathbb{R}_{>0}$ such that for all $v \in \mathcal{V}$, all $s \in \mathbb{R}^{d(H)}$, and all $t, \tau \in \mathbb{R}_{\geq 0}$:*

$$r \cdot (f_v(s)(t+\tau) - f_v(s)(t)) \geq \epsilon \left\| f_v(s)(t+\tau) - f_v(s)(t) \right\|.$$

In such a case, such flows are (ϵ, r)-monotonic.

Definition 6.14 (Monotonic Resets). *Let $H = (\mathbf{Z}, \mathbf{Z}', \mathcal{V}, \mathcal{E}, Inv, f., Act, Res)$ be a hybrid automaton and let $r \in \mathbb{R}^{d(H)}$ be a vector. The resets of H are* monotonic *with respect to r if there exists an $\epsilon, \gamma \in \mathbb{R}_{>0}$ such that for all $e \in \mathcal{E}$ and all $s_1, s_2 \in \mathbb{R}^{d(H)}$ which satisfy $Res(e)[s_1, s_2]$, it holds that:*

- *if the source and the destination of e are the same location, then either $s_1 = s_2$ or $r \cdot (s_2 - s_1) \geq \gamma$;*
- *if e's source and destination differ, then $r \cdot (s_2 - s_1) \geq \epsilon \|s_2 - s_1\|$.*

In such a case, resets are (ϵ, γ, r)-monotonic.

If flows are (ϵ, r)-monotonic, then their projections on the vector r monotonically diverge from the initial evolution point, p_i. Analogously, (ϵ, γ, r)-monotonic resets do not decrease the distance from p_i. Hence, if we bound by definition the activation regions along the axis associated with r, then both the maximum number of admissible discrete jumps and the maximum elapsed time in valid evolutions of the automaton should be bounded.

We are now able to formally present STORMED hybrid automata.

Definition 6.15 (STORMED Hybrid Automata [23]). *A* STORMED hybrid automaton *H is a hybrid automaton such that there exist $b_-, b_+, d_{min} \in \mathbb{R}$, $\epsilon, \gamma \in \mathbb{R}_{>0}$, and $\psi \in \mathbb{R}^{d(H)}$ and the following conditions hold:*

S *the activation regions are d_{min}-**S**eparable;*

T *the flows are **T**ISC;*

O *H is definable in a **O**-minimal theory;*

RM ***R**esets and flows are (ϵ, γ, ψ)-**M**onotonic and (ϵ, ψ)-monotonic, respectively;*

ED ***E**nds are **D**elimited: the projections of the activation regions on ψ are upper bounded by b_+ and lower bounder by b_-.*

STORMED hybrid automata admit finite time-abstract bisimulation quotient and, if the theory used to define the automaton is decidable, then there exists an effective algorithm for computing such a quotient [23].

7. Conclusions

This article relates hybrid automata, bisimulation, and model checking, show that bisimulation can be used to reduce infinite systems to finite ones, and reviews some of the classes of hybrid automata presented in the literature which admit finite time-abstract bisimulation quotient. In order to achieve these goals, we formalized both labelled transition systems and Kripke structures and we briefly described temporal logics. We detailed bisimulation and bisimulation quotient and we related them with the model checking problem. Moreover, we introduced hybrid automata from both syntactic and semantical points of view and we suggested that time-abstract bisimulation quotient may be used to reduce model checking over hybrid automata to finite structure model checking. Finally, we surveyed some of the classes of hybrid automata for which the existence of a finite time-abstract bisimulation quotient is guaranteed.

Although not many classes of hybrid automata admit a finite time-abstract bisimulation quotient and though the expressiveness of such classes suffers many limitations in either dynamics, resets, or global evolutions, bisimulation still remains a powerful instrument for the investigation of such framework and, more in general, of infinite state models. In particular, the surroundings avoiding finite index bisimulations seem to be more linked to the extreme suppleness of the hybrid automaton formalism than to real world models. As, in some sense, proved by STORMED hybrid automata, the possibility of chopping the space further and further, due to the density of variable domain, together with the unboundedness of the real variable values are the main reasons of infinite-index time-abstract bisimulation. However, such conditions are far from been natural: which thermostat cares about one millionth or one million degrees? which rule discriminates one amstrong or one parsec? Because of such reasons, even if it is hard to identify general classes of hybrid automata having finite time-abstract bisimulation quotient, it is more likely to run into a particular model which possesses such feature. Moreover, in recent years, we have seen many proposals to identify either a coarser definition of bisimulation [13, 12] or a rougher interpretation for the automata evolutions [3, 4]. Such efforts will certainly open a new frontier in the model checking of infinite-state models and call again for the use of bisimulation and, more in general, of partial order reductions as appealing tools to investigate hybrid automata.

References

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis and S. Yovine, *The algorithmic analysis of hybrid systems*, Theoret. Comput. Sci. **138** (1995), 3–34.

[2] R. Alur and D.L. Dill, *A theory of timed automata*, Theoret. Comput. Sci. **126** (1994), 183–235.

[3] A. Casagrande, C. Piazza and A. Policriti, *Discreteness, hybrid automata and biology*, in the proceedings of the 9th *international workshop on discrete event systems* (Göteburg, Sweden). IEEE Computer Society Press (2008), 281–286.
[4] A. Casagrande, C. Piazza and A. Policriti, *Discrete semantics for hybrid automata*, Discrete Event Dyn. Syst. **19** (2009), 471–493.
[5] A. Casagrande, C. Piazza, A. Policriti and B. Mishra, *Inclusion dynamics hybrid automata*, Inform. and Comput. **206** (2008), 1394–1424.
[6] E.M. Clarke and E.A. Emerson, *Design and synthesis of synchronization skeletons using brancing time temporal logic*, in the proceedings of the workshop *Logics of programs* (Yorktown Heights, New York), LNCS volume 131, Springer, Berlin (1981), 52–71.
[7] E.M. Clarke, E.A. Emerson and A.P. Sistla, *Automatic verification of finite-state concurrent systems using temporal logic specifications*, ACM Trans. Progr. Lang. Syst. **8** (1986), 244–263.
[8] E.M. Clarke, O. Grumberg and D.A. Peled, *Model checking*, MIT Press, Boston (1999).
[9] E.E. Doberkat, *The converse of a stochastic relation*, J. Log. Algebr. Program. **62** (2005), 133–154.
[10] E.A. Emerson and E.M. Clarke, *Using branching time temporal logic to synthesize synchronization skeletons*, Sci. Comput. Program. **2** (1982), 241–266.
[11] M. Forti and F. Honsell, *Set theory with free construction principles*, Annali Scuola Normale Superiore di Pisa Cl. Sc. **IV** (1983), 493–522.
[12] A. Girard, A. Agung Julius and G. J. Pappas, *Approximate simulation relations for hybrid systems*, Discrete Event Dyn. Syst. **18** (2008), 163–179.
[13] A. Girard and G.J. Pappas, *Approximation metrics for discrete and continuous systems*, IEEE Trans. Automat. Control **52** (2007), 782–798.
[14] J.F. Knight, A. Pillay and C. Steinhorn, *Definable sets in ordered structures II*, Trans. Amer. Math. Soc. **2** (1986), 593–605.
[15] G. Lafferriere, G.J. Pappas and S. Sastry, *O-minimal hybrid systems*, Math. Control Sign. Syst. **13** (2000), 1–21.
[16] O. Maler, Z. Manna and A. Pnueli, *From timed to hybrid systems*, in J.W. de Bakker, C. Huizing, W.P. de Roever and G. Rozenberg, *Real-time: theory in practice*, Springer, Berlin (1991), 447–484.
[17] R. Milner, *A calculus of communicating systems*, Springer, Berlin (1982).
[18] R. Milner, *Communication and concurrency*, Prentice-Hall Inc., Upper Saddle River (1989).
[19] D.M.R. Park, *Concurrency and automata on infinite sequences*, in the proceedings of 5th *GI-Conference on Theoretical Computer Science* (London, U.K.), LNCS volume 104, Springer, Berlin (1981), 167–183.
[20] A. Pnueli, *The temporal logic of programs*, in the proceedings of the 18th *Annual Symposium on Foundations of Computer Science*, (Rhode Island, U.S.A.), IEEE Computer Society Press (1977), 46–57.
[21] J. van Benthem, *Modal correspondence theory*, Ph.D. thesis, Department of Mathematics, University of Amsterdam, Amsterdam, The Netherlands (1978).
[22] L. van den Dries and C. Miller, *Geometric categories and O-minimal struc-*

tures, Duke Math. J.l **84** (1996), 497–540.

[23] V. Vladimerou, P. Prabhakar, M. Viswanathan and G.E. Dullerud, *STORMED hybrid systems*, in the proceedings of the 35$^{\text{th}}$ *International Colloquium on Automata, Languages and Programming*, LNCS volume 5126, Springer, Berlin (2008), 136–147.

Author's address:

Alberto Casagrande
Dipartimento di Matematica e Informatica
Università degli Studi di Trieste
Via Valerio 12/1, 34127 Trieste, Italy
E-mail: acasagrande@units.it

Received September 15, 2010
Revised October 17, 2010