Available online at www.sciencedirect.com
SCIENCE DIRECT®

Electronic Notes in Theoretical Computer Science

ELSEVIER Electronic Notes in Theoretical Computer Science 96 (2004) 91–112
www.elsevier.com/locate/entcs

Expressive Power of Hybrid Systems with Variables, Parameters and Arrays

Ruggero Lanotte [1],[2]

Dipartimento di Informatica, Università di Pisa,
via Buonarroti 2, 56127, Pisa, Italy

Abstract

A hybrid system consists of a finite number of locations, variables and transitions. Different classes are considered in the literature. In this paper we study the different expressive power of these classes.

Keywords: Hybrid systems, Expressiveness, Arrays, Parameters, Real and Integer variables

1 Introduction

Often the systems one wants to model are control systems embedded in an environment from which stimuli may come with different laws. As an example, sensors controlling temperature or water level may have a non-linear evolution law. *Hybrid Systems* have been introduced to describe similar situations (see [1] and [2]). A hybrid system consists of a finite number of locations, variables and transitions. In each location variables change their value as a function of the time elapsed, and satisfy, at each instant, a formula called the invariant. The system can take a transition to evolve from a location to another location. The transition is labeled with a formula that gives the values of variables triggering the transition and their new values after the transition has been performed. Different variants of this model have been considered. Classes are

[1] Research partially supported by MIUR Progetto Cofinanziato MEFISTO.

[2] Email: lanotte@di.unipi.it

doi:10.1016/j.entcs.2004.04.023

usually distinguished by the mathematical logic which is used to model the system.

We consider the extension of the formalism proposed in [8] and [9] by introducing integers and arrays with infinite elements (which permits to model data structures with fixed, infinite and parametrized length).

Firstly, we recall the classes of *Linear Real Hybrid Systems* (see [10]), *Polynomial Real Hybrid Systems* (see [6]), *Linear Mixed Hybrid Systems* (see [9]), *Parametric Real Hybrid Systems* (see [8]), *D-Hybrid Systems* (see [9]) and *S-Hybrid Systems* (see [9]).

For the classical classes of Linear Real Hybrid Systems and Polynomial Real Hybrid Systems there exists an algorithm based on predicate transformation which permits to have a symbolic model checking, and to have semidecidability of reachability (which is implied by the decidability of satisfiability of the formulae used).

In [8] and [9] it is proved that the same result holds also for the non classical classes. Now the question we answer in this paper is whether the classes introduced in [8] and [9] are effectively an extension of the classical ones, namely we prove expressiveness results. We show that the class of Parametric Real Hybrid Systems is a subset of the class of Polynomial Real Hybrid Systems, but it extends the class of Linear Real Hybrid Systems. We prove that the class of Polynomial Real Hybrid Systems is the most expressive among the classical classes, but the class of D-Hybrid Systems extends it. Moreover, the classes of S-Hybrid Systems and Linear Mixed Hybrid Systems extend the classical class of Linear Real Hybrid Systems and include cases which cannot be described by D-Hybrid Systems and hence by Polynomial Real Hybrid Systems.

2 The formalism

In this section we recall the formalism of Hybrid Systems with identifiers.

2.1 Vectors of identifiers and valuations

Let A be a set; a *vector* $\overrightarrow{s}$ over A is a tuple $(s_1, \dots, s_n)$ with $s_1, \dots, s_n \in A$. We say that s is in $\overrightarrow{s}$, written $s \in \overrightarrow{s}$, iff $s = s_i$ for some $1 \leq i \leq n$.

Let $\overrightarrow{s_1} = (s^1_1, \dots, s^1_n)$ and $\overrightarrow{s_2} = (s^2_1, \dots, s^2_m)$ be two vectors; with $\overrightarrow{s_1} \uplus \overrightarrow{s_2}$ we denote the vector $(s^1_1, \dots, s^1_n, s^2_1, \dots, s^2_m)$.

In the following, names for parameters, real variables, integer variables and arrays are referred to as *identifiers*. A parameter is a real variable that does not change its value during the execution. An array is a function from integers to reals. This definition permits to avoid "index out of bounds" errors and to have arrays with parametric length. Given identifiers $id_1, \ldots, id_n$, $\overrightarrow{id} = (id_1, \ldots, id_n)$ is a *vector of identifiers*, provided that $id_1, \ldots, id_n$ are pairwise different, and a *valuation* over $\overrightarrow{id}$ is a vector $\overrightarrow{v} = (v_1, \ldots, v_n)$ where, for each $1 \leq i \leq n$:

- if id_i is either a real variable or a parameter, then $v_i \in \mathbb{R}$
- if id_i is an integer variable then $v_i \in \mathbb{Z}$
- if id_i is an array then $v_i : \mathbb{Z} \to \mathbb{R}$ is a function from integers to reals.

The set of valuations over $\overrightarrow{id}$ is denoted with $V(\overrightarrow{id})$.

We shall use $x, y, \ldots$ to denote real variables, $h, k, \ldots$ to denote integer variables, $a, b, \ldots$ to denote arrays and $m, n, \ldots$ to denote parameters.

With $\overrightarrow{id}' = (id'_1, \ldots, id'_n)$ we denote the vector of identifiers marked by $\cdot'$, where id'_i represents the new value that id_i can take, for instance, due to an assignment.

2.2 Quantified formulae

Let $\overrightarrow{id}$ be a vector of identifiers; we define the set $\mathcal{P}(\overrightarrow{id})$ of *polynomial terms* over $\overrightarrow{id}$ as follows:

$$\tau ::= c \mid c \cdot id \mid c \cdot a[\tau_1] \mid \tau_1 + \tau_2 \mid \tau_1 \cdot \tau_2$$

where $\tau, \tau_1, \tau_2 \in \mathcal{P}(\overrightarrow{id})$, $c \in \mathbb{Q}$, $id \in \overrightarrow{id}$ is not an array and $a \in \overrightarrow{id}$ is an array. A polynomial term is a *linear term* if it is constructed without operation $\tau_1 \cdot \tau_2$.

As an example, if x and y are real variables, a is an array and k is an integer variable, $(9 + k \cdot y \cdot x) \cdot (\frac{2}{3} \cdot y) + a[k]^2 \cdot y$ is a polynomial term in the set $\mathcal{P}((x, y, a, k))$. The polynomial term $9 + 2 \cdot k + a[k + 3]$ is also a linear term.

A polynomial term τ is an *integer term* iff each constant c in τ is an integer and each identifier id in $\overrightarrow{id}$ is an integer variable. As an example, the term $h^2 \cdot k + 10$ is an integer term over the vector of integer variables (h, k). In the following we suppose that for each $a[\tau]$ appearing in a term, τ is an integer term. We shall show that this choice does not cause loss of generality.

A valuation $\overrightarrow{v} = (v_1, \ldots, v_n)$ over $\overrightarrow{id}$ extends to $\mathcal{P}(\overrightarrow{id})$ as follows:

$$\begin{aligned}
\overrightarrow{v}(c) &= c\\
\overrightarrow{v}(c \cdot id) &= c \cdot v_i, \text{ if } id = id_i\\
\overrightarrow{v}(c \cdot a[\tau]) &= c \cdot v_i(\overrightarrow{v}(\tau)), \text{ if } a = id_i\\
\overrightarrow{v}(\tau_1 + \tau_2) &= \overrightarrow{v}(\tau_1) + \overrightarrow{v}(\tau_2)\\
\overrightarrow{v}(\tau_1 \cdot \tau_2) &= \overrightarrow{v}(\tau_1) \cdot \overrightarrow{v}(\tau_2).
\end{aligned}$$

Let $\overrightarrow{id}$ be a vector of identifiers; we define the set $\Phi(\overrightarrow{id})$ of *quantified formulae* over $\overrightarrow{id}$ as follows:

$$\phi ::= \tau \sim 0 \;\mid\; \exists id\,.\,\phi' \;\mid\; \neg\phi_1 \;\mid\; \phi_1 \vee \phi_2 \;\mid\; \phi_1 \wedge \phi_2$$

where ϕ, ϕ_1, ϕ_2 range over $\Phi(\overrightarrow{id})$, τ is in $\mathcal{P}(\overrightarrow{id})$, $\sim \in \{<, \leq, =, \geq, >\}$, $id \notin \overrightarrow{id}$, and $\phi' \in \Phi(\overrightarrow{id} \uplus (id))$. Notice that the identifier id that can appear in the scope of quantifier $\exists$ is not in $\overrightarrow{id}$. Notice also that $\Phi(\overrightarrow{id})$ is a subset of High Order Logic, since the arrays (which are functions) can be quantified.

A formula in $\Phi(\overrightarrow{id})$ is *linear* iff all terms $\tau \in \mathcal{P}(\overrightarrow{id})$ appearing in ϕ are linear. Sometimes we will write $\exists\overrightarrow{id}.\phi$ for $\exists id_1 \ldots \exists id_n\,.\,\phi$, where $\overrightarrow{id} = (id_1, \ldots, id_n)$, and $\forall id.\phi$ for $\neg\exists id.\neg\phi$.

To express a term $a[\tau]$ with τ a non integer term, we can introduce an integer variable k and force $k = \tau$. As an example, $a[x \cdot y] = 5$ can be expressed by $\exists k\,.\,a[k] = 5 \wedge k = x \cdot y$. The following example shows some properties over arrays that can be expressed in $\Phi(\overrightarrow{id})$.

Example 2.1 The following properties over arrays can be expressed in $\Phi(\overrightarrow{id})$:

- Equality of arrays: $\forall h \in [1, size].\, a[h] = b[h]$
- Membership of a value x: $\exists k.\, a[k] = x$
- k is the index of the minimum: $\forall h \in [1, size].\, a[h] \geq a[k]$
- Binary array: $\forall h.\, a[h] = 0 \vee a[h] = 1$
- Ordering: $\forall h \in [1, size - 1].\, a[h] \leq a[h + 1]$.

Let $\phi \in \Phi(\overrightarrow{id})$ and $\overrightarrow{v} \in V(\overrightarrow{id})$; we say that $\overrightarrow{v}$ *satisfies* ϕ, written $\overrightarrow{v} \models \phi$, iff

$$\begin{array}{lll}
\overrightarrow{v} \models \tau \sim 0 & \text{iff} & \overrightarrow{v}(\tau) \sim 0 \\
\overrightarrow{v} \models \exists id.\phi' & \text{iff} & \text{there is some } v' \in V(id) \text{ such that } \overrightarrow{v} \uplus (v') \models \phi' \\
\overrightarrow{v} \models \neg\phi_1 & \text{iff} & \overrightarrow{v} \not\models \phi_1 \\
\overrightarrow{v} \models \phi_1 \vee \phi_2 & \text{iff} & \text{either } \overrightarrow{v} \models \phi_1 \text{ or } \overrightarrow{v} \models \phi_2 \\
\overrightarrow{v} \models \phi_1 \wedge \phi_2 & \text{iff} & \text{both } \overrightarrow{v} \models \phi_1 \text{ and } \overrightarrow{v} \models \phi_2.
\end{array}$$

Let $\tau_1, \tau_2 \in \mathcal{P}(\overrightarrow{id})$ and $\phi \in \Phi(\overrightarrow{id})$; with $\phi[\tau_1 := \tau_2]$ we denote the formula obtained by replacing each occurrence in ϕ (also in subterms) of τ_1 with τ_2. Let a, b be arrays; $\phi[a := b]$ denotes the substitution $\phi[a[\tau_1] := b[\tau_1]] \ldots [a[\tau_n] := b[\tau_n]]$, where the array a occurs in ϕ with the terms $a[\tau_1], \ldots, a[\tau_n]$.

Let ϕ be a formula in $\Phi(\overrightarrow{id})$, and k be an integer variable; then k *is dependent* in ϕ iff there exist $\phi_1, \ldots, \phi_n \in \Phi(\overrightarrow{id})$ such that $\phi = \bigvee_{i=1}^{n} \phi_i$ and for any $i \in [1, n]$ there exists ϕ' such that $\phi_i \equiv (k = \tau) \wedge \phi'$, where τ is an integer term and h does not appear in τ. These dependent variables are important in programming. In fact, if we have the assignment $k := \tau$, with τ an integer term, then k is dependent. As an example, the assignment $k := k^2 + 1$ is translated into the formula $k' = k^2 + 1$. The term $k^2 + 1$ is an integer term, and hence k' 9is dependent in $k' = k^2 + 1$. Moreover, h is dependent in the formula $h \in [10, 20]$.

The satisfiability of a formula in $\Phi(\overrightarrow{id})$ is, in general, undecidable. In [11], [8], [11], [12], [8] and [8], respectively, the satisfiability has been proved to be decidable for the following classes of $\Phi(\overrightarrow{id})$:

- The set $\Phi_P(\overrightarrow{id})$ of *polynomial real formulae*, i.e. the set of the formulae using only real variables.
- The set $\Phi_{Par}(\overrightarrow{id})$ of *parametric formulae linear on real variable*, i.e. the set of the formulae using only parameters and real variables, and terms of the form $\tau_1 \cdot x_1 + \cdots + \tau_n \cdot x_n + \tau_{n+1}$, where τ_i is a polynomial term on parameters.
- The set $\Phi_L(\overrightarrow{id})$ of *linear real formulae*, i.e. the set of linear formulae using only real variables.
- The set $\Phi_{Mix}(\overrightarrow{id})$ of *mixed linear formulae*, i.e. the set of the linear formulae using only real and integer variables.
- The set $\Phi_D^k(\overrightarrow{id})$ of *D-formulae free on* k, with $k \in \overrightarrow{id}$ an integer variable, i.e. the set of the formulae ϕ such that k is not quantified in ϕ, and, for

each integer variable $h \neq k$, it holds that if $h \in \overrightarrow{id}$, then h is dependent in ϕ, and if $h \notin \overrightarrow{id}$, then for each subformula $\exists h.\phi'$ of ϕ it holds that h is dependent ϕ'.

- The set $\Phi_S(\overrightarrow{id})$ of *S-formulae*, i.e. the set of formulae obtained by repeated applications of disjunction and conjunction of formulae of the form

$$\exists \overrightarrow{id_1}. \left(\phi_1 \wedge \forall h. \exists \overrightarrow{id}_2.\phi_2\right)$$

  such that h is an integer variable, ϕ_1 and ϕ_2 are linear formulae without any quantifier and, for each $a[\tau]$ appearing in ϕ_2, either $\tau = h$ or no identifiers in $h \uplus \overrightarrow{id}_2$ appears in τ.

The set $\Phi_D^k(\overrightarrow{id})$ permits to express formulae in which integer variables (with the exception of at most the non-quantified integer variable k) are either dependent or bounded. The set $\Phi_S(\overrightarrow{id})$ permits to express formulae that define constraints on identifiers and some kind of invariants for arrays.

Example 2.2 $\exists z.\frac{6}{7}y^2x + 3y \cdot z > 0$ is in $\Phi_P((x, y))$, $\exists k.3h + 4k + \frac{5}{2}y = x$ is in $\Phi_{Mix}((y, h, x))$, $\exists l \,.\, l \in [1, 10] \wedge h = k^2 + 1 \wedge l > x$ is a quantified formula in $\Phi_D^k(h, k, x)$, and in Example 2.1 only the ordering formula is not an S-Formula.

2.3 Hybrid systems with identifiers

A *Hybrid System with Identifiers* is a tuple $H = \langle \Sigma, \overrightarrow{id}, Loc, Tr, Act, Init, F\rangle$, where:

- Σ is a finite set of *action symbols.*
- $\overrightarrow{id}$ is a vector of *identifiers.*
- *Loc* is a finite set of *locations.* A *state* is a pair $(l, \overrightarrow{v})$ where l is a location and $\overrightarrow{v}$ is a valuation in $V(\overrightarrow{id})$.
- *Tr* is a finite set of *transitions.* Each transition is a tuple $\langle l, a, \phi, l'\rangle$, where $l \in Loc$ is the *source location*, $l' \in Loc$ is the *target location*, $a \in \Sigma$, and $\phi \in \Phi(\overrightarrow{id} \uplus \overrightarrow{id}')$, where $\overrightarrow{id}$ represents the value of the identifiers when the transition fires and $\overrightarrow{id}'$ represents the new values that are assigned to the identifiers as effect of the transition firing.
- $Act : Loc \to \Phi(\overrightarrow{id} \uplus (t) \uplus \overrightarrow{id}')$ is the *activity function* that assigns to each location a formula, called *activity*, which expresses the relationship between the values of identifiers $\overrightarrow{id}$ when the location is entered and the new values $\overrightarrow{id}'$ taken by the identifiers after t units of time have elapsed.

- $Init = (l_0, \phi_0)$, where $l_0 \in Loc$ is the *initial location* and $\phi_0 \in \Phi(\overrightarrow{id})$ is the *initial condition* that must be satisfied by the system at the beginning.
- $F \subseteq Loc$ is the set of final locations.

The set of Hybrid Systems with Identifiers is denoted $\mathcal{H}_{id}$.

Example 2.3 In Fig. 1 we model a cache of a browser. A request r_c to the cache to obtain a file f of a web page has either a positive answer a_c, if the file is in the cache, or a negative answer n_c, otherwise. In this last case, when the file f is downloaded from the network by the browser, it is cached together with the date d of his download (action s_c). The file in the cache that is replaced by f is the eldest. The real variable d represents the date, the real variable x is used to quantify the answer time of the cache to a request. The real variable f represents the information on the file (name, location and contents). The array $file$ represents the files $file[1], \ldots, file[size]$ in the cache. The array *date* represents the date $date[1], \ldots, date[size]$ of the last requests to $file[1], \ldots, file[size]$. The initial location $Wait$ represents waiting for a request. The initial condition says that the date d is positive, x is equal to zero, and dates and file information are positive reals. After a request r_c, the real variable f' carries the information on the requested file, and the cache enters location $Check$. For readability, in each formula, we have omitted the subformula that expresses that each identifier that does not appear in the formula is not changed. As an example in the formula $f' \geq 0$ we have omitted condition $d' = d \wedge x' = x \wedge \forall h \in [1, size].file'[h] = file[h] \wedge date'[h] = date[h]$. Now, in location $Check$ the cache looks for the file f. If f is cached ($file[k] = f$), then the cache gives a positive answer a_c in a time enclosed in $[2, 5]$, updates the date ($date'[k] = d$), and returns to location $Wait$. If the file is not cached ($\forall h \in [1, size]\,.\,file[h] \neq f$), a negative answer n_c is given in a time enclosed in $[2, 5]$, and the cache enters location $Write$, where it is ready to cache the file. When the page is downloaded (action s_c), the file f is cached ($file'[k] = f$) in the position k of the eldest referred file.

Let us explain now the semantics of $\mathcal{H}_{id}$. Let H be a system in $\mathcal{H}_{id}$, and let $(l, \overrightarrow{v})$ be a state. The system can evolve from $(l, \overrightarrow{v})$ to another state by performing either an activity step or a transition step, as defined below:

- an *activity step* describes the evolution of H due to being in location l and passing of time. In t units of time, the activity $Act(l)$ takes H to a new valuation $\overrightarrow{v}'$:
$$\frac{t \geq 0 \quad \overrightarrow{v} \uplus (t) \uplus \overrightarrow{v}' \models Act(l)}{(l, \overrightarrow{v}) \rightarrow_t (l, \overrightarrow{v}')}$$
- a *transition step* describes the evolution of H due to the firing of a transition.

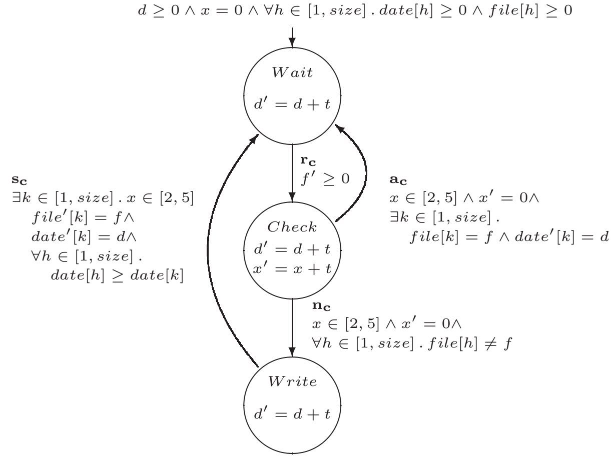
$d \geq 0 \wedge x = 0 \wedge \forall h \in [1, size] \,.\, date[h] \geq 0 \wedge file[h] \geq 0$
$Wait$
$d' = d + t$
$\mathbf{r_c}$
$f' \geq 0$
$\mathbf{s_c}$
$\exists k \in [1, size] \,.\, x \in [2, 5]$
$file'[k] = f \wedge$
$date'[k] = d \wedge$
$\forall h \in [1, size] \,.$
$date[h] \geq date[k]$
$Check$
$d' = d + t$
$x' = x + t$
$\mathbf{a_c}$
$x \in [2, 5] \wedge x' = 0 \wedge$
$\exists k \in [1, size] \,.$
$file[k] = f \wedge date'[k] = d$
$\mathbf{n_c}$
$x \in [2, 5] \wedge x' = 0 \wedge$
$\forall h \in [1, size] \,.\, file[h] \neq f$
$Write$
$d' = d + t$

Fig. 1. The cache of a browser

The formula ϕ of the transition must be satisfied by the valuation $\overrightarrow{v}$, and the transition takes H to the target location l' of the transition and to a new valuation $\overrightarrow{v}'$ according to ϕ. More precisely:

$$\frac{e = \langle l, a, \phi, l' \rangle \in Tr \quad \overrightarrow{v} \uplus \overrightarrow{v}' \models \phi}{(l, \overrightarrow{v}) \to_e (l', \overrightarrow{v}')}$$

A *run* r of H is a sequence of steps

$$(l_0, \overrightarrow{v}^1_0) \to_{t_0} (l_0, \overrightarrow{v}^2_0) \to_{e_0} (l_1, \overrightarrow{v}^1_1) \to_{t_1} (l_1, \overrightarrow{v}^2_1) \ldots (l_{n-1}, \overrightarrow{v}^2_{n-1}) \to_{e_{n-1}} (l_n, \overrightarrow{v}^1_n)$$

where l_0 is the initial location and l_n is a final location, $\overrightarrow{v}^1_0$ satisfies the initial condition ϕ_0, and, for all $0 \leq i \leq n$, t_i is a time and $e_i = \langle l_i, a_i, \phi_i, l_{i+1} \rangle$ is a transition. With $tw(r)$ we denote the sequence $(t_0, a_0)(t_1, a_1) \ldots (t_n, a_n)$. The *language* accepted by H (denoted by $\mathcal{L}(H)$) is the set $\{tw(r) \mid r \text{ is a run of } H\}$.

2.4 Subclasses

Let us consider now several subclasses of $\mathcal{H}_{id}$. Let $H = \langle \Sigma, \overrightarrow{id}, Loc, Tr, Act, Init \rangle$.

- H is in the class $\mathcal{H}_P$ (resp. $\mathcal{H}_{Par}$, $\mathcal{H}_L$ and $\mathcal{H}_{Mix}$) of *Polynomial Real Hybrid Systems* (resp. *Parametric Real Hybrid Systems*, *Linear Real Hybrid Systems* and *Linear Mixed Hybrid Systems*) if all formulae in H are polynomial real formulae (resp. parametric linear formulae on real variables, linear real formulae and mixed linear formulae).
- H is in the class $\mathcal{H}^k_D$ of *D-Hybrid Systems Free on k* if the formula in $Init$

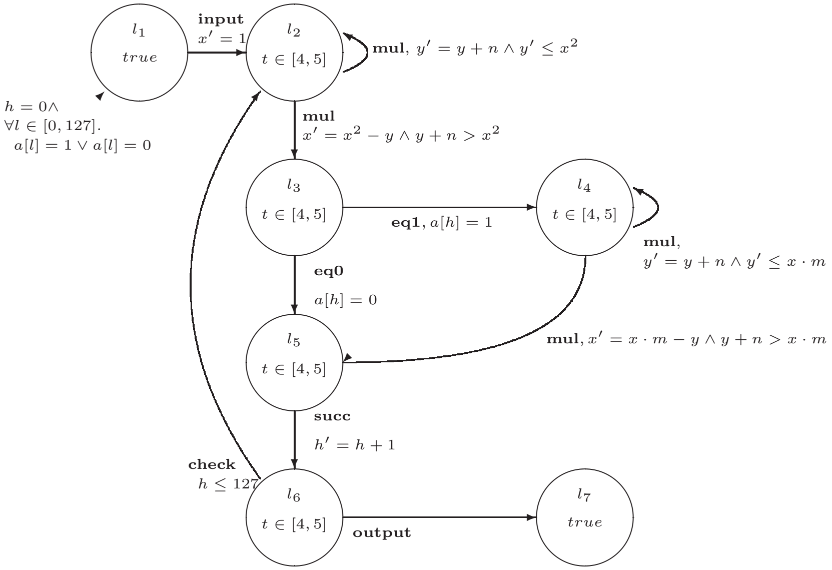

Fig. 2. RSA algorithm

is a D-formula free on k and all other formulae in H are D-formulae free on k'.

- H is in the class $\mathcal{H}_S$ of *S-Hybrid Systems* if all formulae in H are S-Formulae.

The class $\mathcal{H}_{Mix}$ permits to use both integer and real variables, provided that the formulae are linear. The classes $\mathcal{H}_D^k$ and $\mathcal{H}_S$ permit to use also arrays.

Example 2.4 In Fig. 2 we model with a system in $\mathcal{H}_D^n$ the cryptographic algorithm RSA for smart cards (see [3]). The array a represents the 128 bits of a key. The real variables x and y and the integer variable h are auxiliary variables. The encryption of a message m with both a public key n and a secret key k consists in computing $m^k mod\ \ n$. The algorithm is the following:

```
x=1
for h = 0 to 127
  {x = (x*x) mod  n
   If a[h] = 1 then x = (x*m) mod  n}
Return x
```

We assume Montgomery multiplication is used, for which a time in $[4,5]$ that does not depend on input is spent. The system computes $x = (x * x) \bmod n$ in location l_2 and $x = (x * m) \bmod n$ in location l_4. Moreover, the system checks whether $a[h] = 1$ or $a[h] = 0$ in location l_3 and whether $h \leq 127$ in location l_6.

Example 2.5 The system of Example 2.3 is in $\mathcal{H}_S$.

2.5 Properties

In the previous section we have recalled the classes described in [6], [1], [8] and [9]. In this section we recall the properties of these classes described in the same papers. More precisely, for each subclass of $\mathcal{H}_{id}$ considered it is possible to define a notion of symbolic representation and therefore it holds that reachability is semidecidable. Another interesting result of closure binds the classes of $\mathcal{H}_{Par}$ and $\mathcal{H}_L$.

The set of states and steps of a system $H \in \mathcal{H}_{id}$ is infinite. Therefore, we need a symbolic finite representation to enumerate all reachable states, so that the problem of reachability of states becomes semidecidable.

A *region* of H is a pair (l, ϕ), where l is a location and ϕ is in $\Phi(\overrightarrow{id})$. The region represents the set of all states $(l, \overrightarrow{v})$ such that $\overrightarrow{v} \models \phi$.

For a region (l, ϕ), we define the *successor formulae* over identifiers $[\phi]_l^{\uparrow}$ and $post_e\phi$, which hold after any activity step and after a transition step through transition $e = \langle l, a, \phi', l' \rangle$ is performed. Formally:

$$[\phi]_l^{\uparrow} \equiv \left(\exists \overrightarrow{id}\,.\,\exists\, t \geq 0\,.(\phi \wedge Act(l))\right)[\overrightarrow{id}' := \overrightarrow{id}]$$

$$post_e\phi \equiv \left(\exists \overrightarrow{id}\,.\,(\phi \wedge \phi')\right)[\overrightarrow{id}' := \overrightarrow{id}].$$

The following proposition is proved in [9] and states the correctness of the operators $[\_]_{\_}^{\uparrow}$ and $post_{e\_}$

Proposition 2.6 *Let (l, ϕ) be a region, $\overrightarrow{v}'$ be a valuation and e be the transition $\langle l, a, \phi', l' \rangle$. Then the following facts hold:*

- $\overrightarrow{v}' \models [\phi]_l^{\uparrow}$ *iff there are a valuation $\overrightarrow{v}$ with $v \models \phi$ and some $c \in \mathbb{R}^{\geq 0}$ such that $(l, \overrightarrow{v}) \rightarrow_c (l, \overrightarrow{v}')$*
- $\overrightarrow{v}' \models post_e\phi$ *iff there is a valuation $\overrightarrow{v}$ with $\overrightarrow{v} \models \phi$ such that $(l, \overrightarrow{v}) \rightarrow_e (l', \overrightarrow{v}')$.*

Let us recall now the form of the regions of the subclasses of $\mathcal{H}_{id}$.

- If H is in $\mathcal{H}_L$ (resp. $\mathcal{H}_{Par}$, $\mathcal{H}_P$ and $\mathcal{H}_{Mix}$) then a *linear real region* (resp. *linear real region*, *parametric real region*, *polynomial real region* and *linear mixed region*) is a pair (l, ϕ), where l is a location and ϕ is in $\Phi_L(\overrightarrow{id})$ (resp. $\Phi_{Par}(\overrightarrow{id})$, $\Phi_P(\overrightarrow{id})$ and $\Phi_{Mix}(\overrightarrow{id})$).
- If H is in $\mathcal{H}_D^k$ then a *D-region free on k* is a pair (l, ϕ), where l is a location and ϕ is in $\Phi_D^k(\overrightarrow{id})$.

- If H is in $\mathcal{H}_S$ then a *S-region* is a pair (l, ϕ), where l is a location and ϕ is in $\Phi_S(\overrightarrow{id})$.

A class of regions R is *closed* w.r.t. the successor formulae iff, given any region (l, ϕ) in R and any transition $e = \langle l, a, \phi', l' \rangle$, it holds that also $(l, [\phi]_l^{\uparrow})$ and $(l', post_e\phi)$ are in R.

The closure of regions w.r.t. successor formulae for the classes $\mathcal{H}_P$ and $\mathcal{H}_L$ has been proved in [6] and [1], respectively. The closure of regions w.r.t. successor formulae for the classes $\mathcal{H}_{Par}$, $\mathcal{H}_{Mix}$, $\mathcal{H}_D^k$ and $\mathcal{H}_S$ is proved in [8] and [9].

Since the satisfiability of the different classes of formulae used is decidable it is obvious that the reachability problem is semidecidable.

An interesting result proved in [8] is that if a region (l, ϕ) is reachable by an $H \in \mathcal{H}_{Par}$ with parameters in $\overrightarrow{m}$, and $\overrightarrow{u}$ is a rational instance in $V(\overrightarrow{m})$, then, if we instantiate the parameters in $\overrightarrow{m}$ with the values in $\overrightarrow{u}$ in both H and (l, ϕ), we have that $(l, \phi[\overrightarrow{m} := \overrightarrow{u}])$ is a reachable region for the Linear Real Hybrid System $H[\overrightarrow{m} := \overrightarrow{u}]$.

Therefore, in the class of $\mathcal{H}_{Par}$ it is allowed to use parameters as rates of real variables or coefficients of linear formulae. This is not allowed in the class of $\mathcal{H}_L$. By the result proved in [8], if one wants to calculate for which rates or coefficients a Linear Real Hybrid System satisfies a given property, it is sufficient to calculate the set of rational instances for which the Parametric Real Hybrid System which satisfies it.

3 Expressiveness

In the previous section we have recalled some interesting subclasses of $\mathcal{H}_{id}$ and we have shown how the new classes introduced in [8] and [9] are suitable to model real-life systems while preserving the same properties of the classical ones.

Now, we address the question whether these new classes really extend the classical ones. Therefore, in this section we will study the expressive power of Hybrid Systems with respect to the language accepted.

First of all, in the following proposition, we summarize containments which are trivial.

Proposition 3.1 *The following relations hold:*

- $\mathcal{L}(\mathcal{H}_{id}) \supseteq \mathcal{L}(\mathcal{H}')$, *where* $\mathcal{H}'$ *is one of the subclasses mentioned above;*
- $\mathcal{L}(\mathcal{H}_D^k) \supseteq \mathcal{L}(\mathcal{H}_P) \supseteq \mathcal{L}(\mathcal{H}_{Par}) \supseteq \mathcal{L}(\mathcal{H}_L)$*;*
- $\mathcal{L}(\mathcal{H}_S) \supseteq \mathcal{L}(\mathcal{H}_{Mix}) \supseteq \mathcal{L}(\mathcal{H}_L)$*.*

To prove expressiveness results we use the following languages:

- $\mathcal{L}_1 = \{(a, c_1)(a, c_2) \mid c_1, c_2 \in \mathbb{N}\}$
- $\mathcal{L}_2 = \{(a, t) \mid t \text{ is an odd natural}\}$
- $\mathcal{L}_3 = \{(a, t)(b, t') \mid t' = t^2 \text{ and } t \in \mathbb{R}^{\geq 0}\}$
- $\mathcal{L}_4 = \cup_{n\in\mathbb{N}}\{(a, t_1)(b, t'_1) \dots (a, t_n)(b, t'_n) \mid t_i \in \mathbb{R}^{\geq 0}$ and $t'_i = t_i^2,$
  for any $1 \leq i \leq n, \}$
- $\mathcal{L}_5 = \cup_{n\in\mathbb{N}}\{(a, t_1)(a, t_2) \dots (a, t_n)(b, t'_1) \dots (b, t'_n) \mid t_i \in \mathbb{R}^{\geq 0}$ and $t_i = t'_i,$
  for any $1 \leq i \leq n\}$
- $\mathcal{L}_6 = \cup_{n\in\mathbb{N}}\{(a, t_1)(a, t_2) \dots (a, t_n)(b, t'_1) \dots (b, t'_n) \mid t_i \in \mathbb{R}^{\geq 0}$ and $t'_i = (t_i)^2,$
  for any $1 \leq i \leq n\}$
- $\mathcal{L}_7 = \cup_{n\in\mathbb{N}}\{(a, t_1)(a, t_2) \dots (a, t_n)(b, t'_1) \dots (b, t'_n) \mid t'_i = t_i$ and $t_i \in \mathbb{N},$
  for any $1 \leq i \leq n\}$
- $\mathcal{L}_8 = \cup_{n\in\mathbb{N}}\{(a, t)(b, t^2)(c, t_1) \dots (c, t_n) \mid t \in \mathbb{R}^{\geq 0}$ and $t_1, \dots, t_n \in \mathbb{N}\}.$

Why we gain power by using integer variables? The idea is that with a step we cannot simulate an infinite set of natural numbers. In the following proposition we prove that integer variables give more expressiveness to $\mathcal{H}_{Mix}$ with respect to H_L. Moreover, since $\mathcal{H}_D^k$ may have only one free integer variable, which is not sufficient to model two or more integer variables, we prove that there exists a language recognized by an S-Hybrid System and not by a D-Hybrid system.

Proposition 3.2 *The language $\mathcal{L}_1$ is in $\mathcal{L}(\mathcal{H}_{Mix})$ and $\mathcal{L}(\mathcal{H}_S)$ but is neither in $\mathcal{L}(\mathcal{H}_D^k)$ nor in $\mathcal{L}(\mathcal{H}_L)$.*

Moreover, in the following proposition, we prove that the intersection of $\mathcal{L}(\mathcal{H}_{Mix})$ and $\mathcal{L}(\mathcal{H}_D^k)$ is a set greater than $\mathcal{L}(\mathcal{H}_L)$. It means that $\mathcal{H}_L$ is not able to simulate one integer variable.

Proposition 3.3 *The language $\mathcal{L}_2$ is in $\mathcal{L}(\mathcal{H}_{Mix}) \cap \mathcal{L}(\mathcal{H}_D^k)$ but is not in $\mathcal{L}(\mathcal{H}_L)$. Therefore $\mathcal{L}(\mathcal{H}_{Mix}) \cap \mathcal{L}(\mathcal{H}_D^k) \supset \mathcal{L}(\mathcal{H}_L)$.*

We consider now the expressive power of polynomial formulae. In the following proposition we prove that polynomial formulae give more expressiveness to Hybrid Systems. So, parameters of $\mathcal{H}_{Par}$ permit to express languages not accepted by Hybrid Systems in $\mathcal{H}_L$. But, since parameters do not change their value, there exists a language recognized by $\mathcal{H}_P$ and not by $\mathcal{H}_{Par}$.

Proposition 3.4 *The language $\mathcal{L}_3$ is in $\mathcal{L}(\mathcal{H}_{Par})$ and is neither in $\mathcal{L}(\mathcal{H}_{Mix})$ nor in $\mathcal{L}(\mathcal{H}_S)$. Moreover, the language $\mathcal{L}_4$ is in $\mathcal{L}(\mathcal{H}_P)$ but i not in $\mathcal{L}(\mathcal{H}_{Par})$.*

The previous proposition implies that polynomial formulae used in $\mathcal{H}_D^k$

cannot be simulated by linear formulae of $\mathcal{H}_S$.

Corollary 3.5 *$\mathcal{L}_3$ is in $\mathcal{L}(\mathcal{H}_D^k)$ but is not in $\mathcal{L}(\mathcal{H}_S)$.*

Finally, we consider the power gained with arrays. In the following proposition we prove that arrays give memory to Hybrid Systems, which therefore gain in expressive power. This gain of expressivity is obvious since we have considered arrays with infinite length, but, in the proposition, we prove that to gain expressive power it is sufficient to have arrays with finite parametric length.

Therefore the class of $\mathcal{H}_S$ is more expressive than the class of $\mathcal{H}_{Mix}$. Moreover, if we consider the intersection of $\mathcal{L}(\mathcal{H}_S)$ with $\mathcal{L}(\mathcal{H}_D^k)$ the power of arrays permits to prove that there exists a language in the intersection but not in $\mathcal{L}(\mathcal{H}_{Mix})$. As an example, the length of the cache of Example 1 is not fixed and is expressed with a generic value for the integer variable *size*. This definition means that the cache is finite but we do not make any assumption on the value of its length. This cannot be simulated without arrays with infinite length.

Proposition 3.6 *The language $\mathcal{L}_5$ is in $\mathcal{L}(\mathcal{H}_D^k) \cap \mathcal{L}(\mathcal{H}_S)$ but is not in $\mathcal{L}(\mathcal{H}_P) \cup \mathcal{L}(\mathcal{H}_{Mix})$.*

In the following proposition we prove that the polynomial formulae on arrays of the class $\mathcal{H}_D^k$ permit to express languages not recognized by both $\mathcal{H}_P$ and $\mathcal{H}_S$. Moreover, the formulae with integer variables and arrays of $\mathcal{H}_S$ permit to show that there exists a language accepted by $\mathcal{H}_S$ but not accepted by both $\mathcal{H}_D^k$ and $\mathcal{H}_{Mix}$.

Proposition 3.7 *The language $\mathcal{L}_6$ is in $\mathcal{L}(\mathcal{H}_D^k)$ but is not in $\mathcal{L}(\mathcal{H}_P) \cup \mathcal{L}(\mathcal{H}_S)$. Moreover, the language $\mathcal{L}_7$ is in $\mathcal{L}(\mathcal{H}_S)$ but is not in $\mathcal{L}(\mathcal{H}_{Mix}) \cup \mathcal{L}(\mathcal{H}_D^k)$.*

Finally we prove that the class of $\mathcal{H}_{id}$ is the most expressive of those considered. In fact we show that there exists a language recognized by $\mathcal{H}_{id}$ and not recognized by both $\mathcal{H}_D^k$ and $\mathcal{H}_S$.

Proposition 3.8 *the language $\mathcal{L}_8$ is in $\mathcal{L}(\mathcal{H}_{id})$ but is not in $\mathcal{L}(\mathcal{H}_D^k) \cup \mathcal{L}(\mathcal{H}_S)$.*

The previous results imply the following theorem.

Theorem 3.9 *The classification of expressiveness of languages and the Hybrid Systems memberships described in figure 3 hold.*

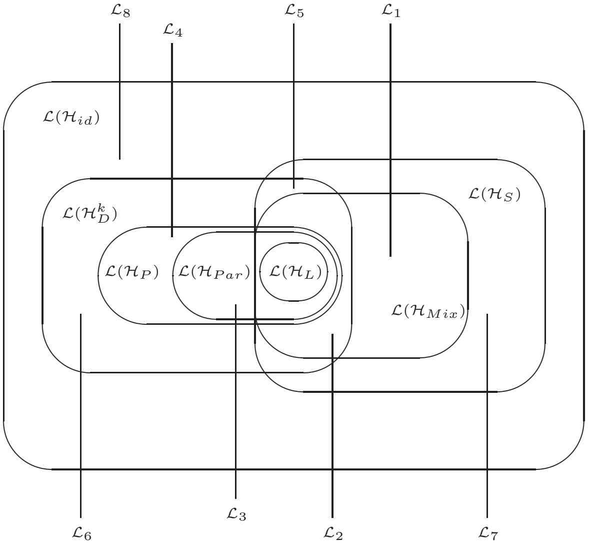
$\mathcal{L}_8$
$\mathcal{L}_4$
$\mathcal{L}_5$
$\mathcal{L}_1$
$\mathcal{L}(\mathcal{H}_{id})$
$\mathcal{L}(\mathcal{H}_S)$
$\mathcal{L}(\mathcal{H}_D^k)$
$\mathcal{L}(\mathcal{H}_P)$
$\mathcal{L}(\mathcal{H}_{Par})$
$\mathcal{L}(\mathcal{H}_L)$
$\mathcal{L}(\mathcal{H}_{Mix})$
$\mathcal{L}_3$
$\mathcal{L}_6$
$\mathcal{L}_2$
$\mathcal{L}_7$

Fig. 3. The classes.

4 Conclusions and future works

In [8] and [9] the symbolic model checking for linear and polynomial hybrid systems is extended for classes which consider arrays, integer variables and polynomial formulae.

Therefore, in this paper we have studied expressiveness results for hybrid systems so extended. We have shown that arrays, integer variables and polynomial formulae build an hierarchical expressivity in the framework of hybrid systems. We have proved that the class of $\mathcal{H}_P$ is the most expressive among the known classes. But the new class of $\mathcal{H}_D^k$ extends it. Moreover, the new classes of $\mathcal{H}_S$ and $\mathcal{H}_{Mix}$ extend the classical class of $\mathcal{H}_L$ and consider cases which cannot be recognized by $\mathcal{H}_D^k$ and so by $\mathcal{H}_P$.

As a future work we consider to tackle the problem of succinctness of the different classes.

References

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T.Ã. Henzinger, P.H̃. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[2] R. Alur, T.Ã. Henzinger, and P.H̃. Ho. Automatic symbolic verification of embedded systems. *IEEE Transactions on Software Engineering*, 22:181–201, 1996.

[3] CASCADE. Chip architecture for smart cards and portable intellident devices. http://www.dice.ucl.ac.be/crypto/cascade.

[4] L. N. Childs. *A Concrete Introduction to Higher Algebra*. Springer, 1979.

[5] K. Ferrante and C. Rackoff. A decision procedure for first-order theory of real addition with order. *SIAM Journal on Computing*, 4:69–76, 1975.

[6] M. Fränzle. Analysis of hybrid systems: An ounce of realism can save an infinity of states. In *Computer Science Logic*, number 1683 in Lecture Notes in Computer Science, pages 126–140, Berlin, 1999. Springer.

[7] B.E. Herbert. *Mathematical Introduction to Logic*. Academic Press, 1972.

[8] R. Lanotte. *An Automaton-Theoretic Approach to Safety and Security in Real-Time Systems*. PhD thesis, University of Pisa, Pisa, Italy, 2002.

[9] R. Lanotte, A. Maggiolo-Schettini, and S. Tini. Information flow in hybrid systems. Submitted, 2003.

[10] A. Pnueli and J. Sifakis. Special issue on hybrid systems. *Theoretical Computer Science*, 138, 1995.

[11] A. Tarski. A decision method for elementary algebra and geometry. Technical report, University of California, Berkeley, California, 1951.

[12] V. Weispfenning. Mixed real-integer linear quantifier elimination. In *ACM International Symposium on Symbolic and Algebraic Computation*, pages 129–136. ACM Press, 1999.

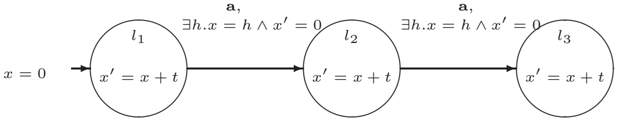

Fig. 4. The Hybrid System H_1.

Appendix

Quantifier elimination

Before proving the propositions we give the notion of quantifier elimination and the related results. Let $\exists id.\phi$ be a formula such that in ϕ there is no quantified identifiers. With $Del(id, \exists id.\phi)$ we denote the formula without quantified identifier equivalent to $\exists id.\phi$.

We enumerate some properties of Del proved in [11], [5] [12] and [8]. If ϕ is a formula without quantifiers, then:

(i) If ϕ is in $\Phi_L(\overrightarrow{id})$ (resp. $\Phi_{Par}(\overrightarrow{id})$, $\Phi_P(\overrightarrow{id})$ and $\Phi^k_D(\overrightarrow{id})$) and x is a real variable, then $Del(x, \exists x.\phi)$ is in $\Phi_L(\overrightarrow{id})$ (resp. $\Phi_{Par}(\overrightarrow{id})$, $\Phi_P(\overrightarrow{id})$ and $\Phi^k_D(\overrightarrow{id})$).

(ii) If ϕ is in $\Phi^k_D(\overrightarrow{id})$ (resp. $\Phi_S(\overrightarrow{id})$) and a is an array, then $Del(a, \exists a.\phi)$ is in $\Phi^k_D(\overrightarrow{id})$ (resp. $\Phi_S(\overrightarrow{id})$).

(iii) If ϕ is in $\Phi^k_D(\overrightarrow{id})$ and h is either a bounded or a dependent integer variable, then $Del(h, \exists h.\phi)$ is in $\Phi^k_D(\overrightarrow{id})$.

Proof of Proposition 3.1

The only non-immediate containment is $\mathcal{L}(\mathcal{H}_S) \supseteq \mathcal{L}(\mathcal{H}_{Mix})$. But, by results in [12], we can transform a formula in $\Phi_{Mix}(\overrightarrow{id})$ into a formula without quantifiers and which uses the relation $\equiv_n$ and the operator $[\tau]$. We can delete these by using the following equivalences

- $\neg(\tau \equiv_n 0)$ is equal to $\bigvee_{i=1}^{n-1} \tau + i \equiv_n 0$ (see [7]);
- $\tau \equiv_n 0$ is equivalent to $\exists k.\tau = n \cdot k$;
- $\tau_1 + c \cdot [\tau_2] \sim 0$ is equivalent to $\exists k.\tau_1 + c \cdot k \sim 0 \wedge k \in (\tau_2 - 1, \tau_2]$.

Therefore, each formula in $\Phi_{Mix}(\overrightarrow{id})$ is equivalent to a formula of the form $\exists \overrightarrow{k}.\phi'$, where $\overrightarrow{k}$ is a vector of integers and ϕ' is formula without quantifiers. But this formula is also an S-Formula. □

Proof of Proposition 3.2

Let H_1 be the Hybrid System in figure 4 with l_3 as final location. The system H_1 is both in $\mathcal{H}_{Mix}$ and $\mathcal{H}_S$. The language $\mathcal{L}(H_1)$ is equal to $\mathcal{L}_1$. We prove

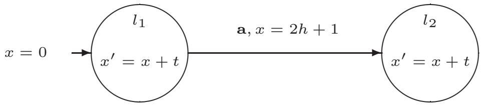

Fig. 5. The Hybrid System H_2.

that $\mathcal{L}_1$ is not accepted by any $\mathcal{H}_L$ and $\mathcal{H}_D^k$. Let us suppose, by contradiction, that there exists a H'_1 in $\mathcal{H}_L$ such that $\mathcal{L}(H'_1) = \mathcal{L}_1$. Let (l'_1, ϕ_0) be its initial condition. So, for each $(a, c_1)(a, c_2) \in \mathcal{L}_1$ there exists a run $(l'_1, \overrightarrow{v}_1) \rightarrow_{c_1} (l'_1, \overrightarrow{v}_2) \rightarrow_{e_1} (l'_2, \overrightarrow{v}_3) \rightarrow_{c_2} (l'_2, \overrightarrow{v}_4) \rightarrow_{e_2} (l'_3, \overrightarrow{v}_5)$.

The times c_1 and values $\overrightarrow{v}_2$ of real variables which are admissible after the first activity step is the valuation that satisfies the formula $\exists \overrightarrow{x}.\phi_0 \wedge Act(l'_1)$, where $\overrightarrow{x}$ is the vector of identifiers of H'_1 (see proposition 2.6). By using *Del* we have an equivalent linear real formula ϕ' without quantifiers such that, by renaming the real variables $\overrightarrow{x}'$ with $\overrightarrow{x}$, we have that $\phi'[\overrightarrow{x}' := \overrightarrow{x}] \in \Phi((t) \uplus \overrightarrow{x})$.

Now, the times c_1 and values $\overrightarrow{v}_3$ of real variables which are admissible after the transition step is the valuation which satisfies the formula $\exists \overrightarrow{x}.\phi'[\overrightarrow{x}' := \overrightarrow{x}] \wedge \phi$, where ϕ is the formula which labels the transition taken to perform the first transition step.

Therefore the set of the first admissible times is the set of valuations which satisfy $\exists \overrightarrow{x}'.\exists \overrightarrow{x}.\phi'[\overrightarrow{x}' := \overrightarrow{x}] \wedge \phi$ (see proposition 2.6). By using *Del* we have an equivalent linear real formula on t without quantifiers. So admissible times c_1 are in a finite union of intervals on reals. It is obvious that with a finite set of intervals on reals we cannot express the set of naturals. So we have proved that $\mathcal{L}(\mathcal{H}_{Mix}) \not\subseteq \mathcal{L}(\mathcal{H}_L)$.

Now we must prove that $\mathcal{L}_1$ is not recognized by any $\mathcal{H}_D^k$. This proof can be done by contradiction. By mimicking the proof done for $\mathcal{H}_L$ and by using *Del* on arrays and on integer and real variables, we have that the times t_1 and t_2 of the two activity steps which can be taken by a $\mathcal{H}_D^k$ are the valuations which satisfy a formula ϕ of the form $\exists k.\phi'$, where ϕ' is a D-formula free on k without quantifiers with real variables in (t_1, t_2, k).

Now $[\![\phi]\!]$ is equal to the set $\mathbb{N} \times \mathbb{N}$. The set $[\![\exists k.\exists t_2.\phi']\!]$ gives the times taken in the first activity step. By using *Del* we have an equivalent formula $\exists k.\phi''$.

The set $[\![\exists k.\phi'']\!]$ is equal to $\bigcup_{n \in \mathbb{N}} [\![\phi''[k := n]]\!]$. By using Sturm's algorithm (see [4]), we can transform $\phi''[k := n]$ into a finite union of intervals on reals.

Let p_1 be the maximum degree of t_1 in ϕ'' and p_2 be the size of ϕ''. Sturm's algorithm transforms $\phi''[k := n]$ into a number of intervals less or equal to $p_1^{p_2}$. In fact each $\tau \sim 0$ becomes an union of at most p_1 intervals. The union and the intersection of p_1 repeated p_2 times generate at most $p_1^{p_2}$ intervals.

Let $p^1 = p_1^{p_2}$; since the times must be naturals it means that the integers expressed by the formula $\phi''[k := n]$ are less or equal to p^1. But p^1 does not depend on n, and so we can conclude that for each n the integers expressed by the formula $\phi''[k := n]$ are less or equal to p^1. We call I_n^1 the set of intervals expressed by $\phi''[k := n]$. It is obvious that for each n the set I_n^1 has at most p^1 elements.

Let $\exists k.\exists t_1.\phi'$ be the formula which expresses the times of the second activity step. In the same manner we can find p^2 such that the times expressed by $\exists t_1.\phi'[k := n]$ are less or equal than p^2, for any n. Moreover we call I_n^2 the set of intervals expressed by $\exists t_1.\phi'[k := n]$. It is obvious that for each n the set I_n^2 has at most p^2 elements.

Now let $A_n = I_n^1 \times I_n^2$. Since I_n^1 and I_n^2 have naturals as elements then $\bigcup_{n\in\mathbb{N}} A_n$ is contained in $\mathbb{N} \times \mathbb{N}$. Moreover, it is obvious that the pair of times contained in $[\![\phi'[k := n]]\!]$ are contained in $I_n^1 \times I_n^2$. In fact I_n^1 is the set of times of the first activity step and I_n^2 is the set of times of the second activity step expressed by $\phi'[k := n]$. Therefore it holds that

$$[\![\exists k.\phi']\!] = \bigcup_{n\in\mathbb{N}} [\![\phi'[k := n]]\!] \subseteq \bigcup_{n\in\mathbb{N}} A_n \subseteq \mathbb{N} \times \mathbb{N}.$$

Since $[\![\exists k.\phi']\!]$ is equal to $\mathbb{N} \times \mathbb{N}$, it holds that

$$[\![\exists k.\phi']\!] = \bigcup_{n\in\mathbb{N}} A_n.$$

Moreover, for each n it holds that A_n has at most $p^1 \cdot p^2$ elements.

Let $p = p^1 \cdot p^2$; we can construct a function $f_1 : \mathbb{N} \times [0, p-1] \to \mathbb{N} \times \mathbb{N}$ such that for any $n \in \mathbb{N}$ it holds that

$$\{f_1(n, 0), \ldots, f_1(n, p-1)\} = A_n.$$

Since it holds that $\bigcup_{n\in\mathbb{N}} A_n = [\![\exists k.\phi']\!] = \mathbb{N} \times \mathbb{N}$, the function f_1 is surjective.

Let $f_2 : \mathbb{N} \to \mathbb{N} \times [0, p-1]$ be a function such that, for any natural n, it holds that $f_2(n) = (q, r)$ if and only if $n = q \cdot p + r$. By the theorem of division the function f_2 is well defined. Moreover the function f_2 is surjective. In fact, for each (q, r), it holds that $f_2(q \cdot p + r) = (q, r)$.

Let $f : \mathbb{N} \to \mathbb{N} \times \mathbb{N}$ be the composition of functions f_1 and f_2. Since f_1 and f_2 are surjective also f is surjective. So we have proved that there exists a function $f : \mathbb{N} \to \mathbb{N} \times \mathbb{N}$ which is surjective, but this is impossible.

□

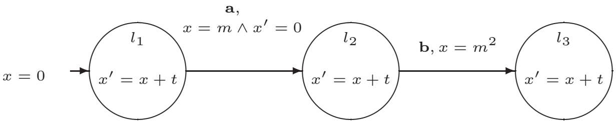

Fig. 6. The Hybrid System H_3.

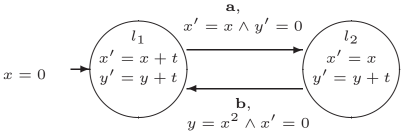

Fig. 7. The Hybrid System H_4.

Proof of Proposition 3.3

It is obvious that $\mathcal{L}(\mathcal{H}_{Mix}) \cap \mathcal{L}(\mathcal{H}^k_D) \supseteq \mathcal{L}(\mathcal{H}_L)$. This is verified directly by considering the syntax of the three classes. Now, we must prove that there exists a language recognized by a $\mathcal{H}_{Mix}$ and $\mathcal{H}^k_D$ and not by a $\mathcal{H}_L$.

Let H_2 be the Hybrid System of figure 5 with l_2 as final location. The system H_2 is both a $\mathcal{H}_{Mix}$ and a $\mathcal{H}^k_D$. The language $\mathcal{L}(H_2)$ is equal to $\mathcal{L}_2$. We prove by contradiction that there is no Hybrid System in $\mathcal{H}_L$ accepting $\mathcal{L}_2$. Similarly to the proof of proposition 3.2, the times expressible by an $\mathcal{H}_L$ is a finite set of intervals, and so we cannot express the set $\{t \,|\, \exists h \,.\, t = 2h+1\}$. □

Proof of Proposition 3.4

Let H_3 be the Hybrid System of figure 6 with l_3 as final location. The system H_3 is both $\mathcal{H}_{Par}$ and $\mathcal{H}^k_D$. The language $\mathcal{L}(H_3)$ is equal to $\mathcal{L}_3$.

We prove, by contradiction, that there exists no $\mathcal{H}_{Mix}$, and therefore no $\mathcal{H}_S$, which recognizes $\mathcal{L}_3$.

As done in the proof of proposition 3.2, the admissible times of a run composed by two activity steps and two transition steps are the valuations which satisfy a linear mixed formula without quantifiers in $\Phi_{Mix}((t_1, t_2))$, where t_1 represents the time of the first activity step and t_2 the time of the second activity step. It means that there exists a linear formula ϕ such that $[\![\phi]\!] = \{(c, c^2) \,|\, c \in \mathbb{R}\}$. But this is impossible. In fact, by the proof of proposition 3.1, we can suppose that ϕ is of the form $\exists \overrightarrow{k} . \phi'$, where ϕ' is a linear mixed formula without quantifiers. Therefore, $[\![\phi]\!]$ can be written as $\bigcup_{\overrightarrow{v} \in V(\overrightarrow{k})} \{(c, c') \,|\, (c, c') \uplus \overrightarrow{v} \models \phi'\}$. Now, $(c, c') \uplus \overrightarrow{v} \models \phi'$ if and only if $(c, c') \models \phi'[\overrightarrow{k} := \overrightarrow{v}]$. But $\phi'[\overrightarrow{k} := \overrightarrow{v}]$ is a linear mixed formula without quantifiers on (t_1, t_2) with rational coefficients and constants. So the formula describes a finite union of convex spaces. More precisely, if $\phi' = \phi_1 \vee \cdots \vee \phi_l$, with ϕ_i is a conjunction on inequalities, for $1 \leq i \leq l$, then $\phi_i[\overrightarrow{k} := \overrightarrow{v}]$ is a convex space. The pair $(\sqrt{2}, 2)$ is a valuation which satisfies ϕ. We note that

$\sqrt{2} \notin \mathbb{Q}$. So there exists i and $\overrightarrow{v}$ such that the convex space $S = \phi_i[\overrightarrow{k} := \overrightarrow{v}]$ contains the pair $(\sqrt{2}, 2)$.

If $S = \{(\sqrt{2}, 2)\}$ then it means that S can be written as $t_1 = \sqrt{2}$ and $t_2 = 2$, but this contradicts the fact that coefficients and constants are rational, in fact $\sqrt{2} \notin \mathbb{Q}$.

If $S \supset \{(\sqrt{2}, 2)\}$ then it means that S has infinite solutions. Let (c, c^2) in S such that $c \neq \sqrt{2}$; then since S is a convex the pair $(\frac{\sqrt{2}+c}{2}, \frac{2+c^2}{2})$, which is in the middle of (c, c^2) and $(\sqrt{2}, 2)$, is in S, and so it is also in $[\![\phi]\!]$.

But this means that the square of $\frac{\sqrt{2}+c}{2}$ is equal to $\frac{2+c^2}{2}$. This holds if and only if $(\sqrt{2} + c)^2 = 2(2 + c^2)$. But the only c which satisfies the equation is $c = \sqrt{2}$, which contradicts the hypothesis. So we have proved that $\mathcal{L}(\mathcal{H}_{Par}) \nsubseteq \mathcal{L}(\mathcal{H}_{Mix})$ and $\mathcal{L}(\mathcal{H}_D^k) \nsubseteq \mathcal{L}(\mathcal{H}_S)$.

Now, we must prove that $\mathcal{L}(\mathcal{H}_P) \nsubseteq \mathcal{L}(\mathcal{H}_{Par})$. Let H_4 be the Hybrid System of figure 7 with l_2 as final location. The language $\mathcal{L}(H_4)$ is equal to $\mathcal{L}_4$.

We prove, by contradiction, that there not exists a $\mathcal{H}_{Par}$ which recognizes $\mathcal{L}_4$. If $\mathcal{H}_{Par}$ exists with l parameters which recognizes $\mathcal{L}(H_4)$, then the admissible times of a run composed by $2 \cdot n$ activity steps and $2 \cdot n$ transition steps, with $n > l$, are given a formula without quantifiers ϕ in $\Phi_{Par}((t_1, t'_1, \ldots, t_n, t'_n, m_1, \ldots, m_l))$. Now, we want to express the fact that $t'_i = (t_i)^2$, but we have proved that is impossible with linear real formulae. Therefore we must use parameters. Since in $Act(l)$ only the time of the actual activity step appears, it means that in ϕ, to have that $t'_i = (t_i)^2$, we must use a parameter m_j and a formula $t'_i = (m_j)^2 \wedge t_i = m_j$. Since parameters cannot change their values, we need n parameters. But this contradicts the hypothesis $l < n$. □

Proof of Proposition 3.6

Let H_5 be the Hybrid System in figure 8 with l_3 as final location.

The language $\mathcal{L}(H_5)$ is equal to $\mathcal{L}_5$. The system H_5 is in $\mathcal{H}_D^k$ and $\mathcal{H}_S$. We prove, by contradiction, that there not exists a $\mathcal{H}_{Mix}$ and $\mathcal{H}_P$ which recognize $\mathcal{L}_5$. If a $\mathcal{H}_{Mix}$ or $\mathcal{H}_P$ exists, with l real variables as done in proposition 3.4, then the admissible times of a run with $2 \cdot n$ activity steps and $2 \cdot n$ transitions steps, with $n > l$, are given by a formula in which we must use a real variable id to express $t_i = id \wedge id = t'_i$. It means that we need n real variables, but this contradicts the hypothesis $n > l$. □

Proof of Proposition 3.7

Let H_6 be the Hybrid System in figure 9 with l_3 as final location; H_6 is in $\mathcal{H}_D^k$ and $\mathcal{L}(H_6)$ is equal to $\mathcal{L}_6$.

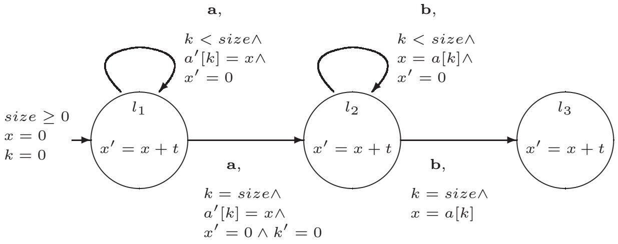

Fig. 8. The Hybrid System H_5.

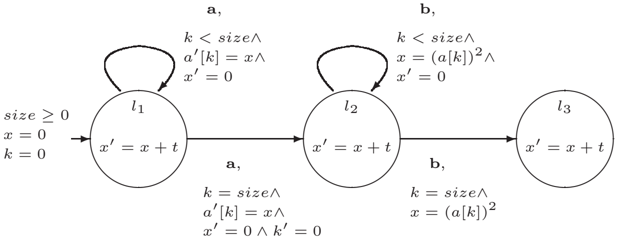

Fig. 9. The Hybrid System H_6.

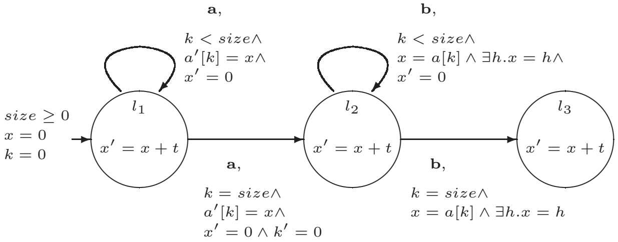

Fig. 10. The Hybrid System H_7.

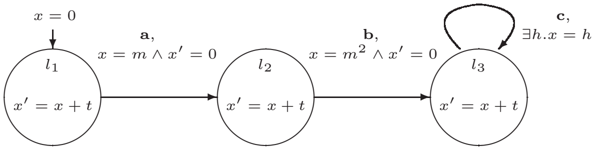

Fig. 11. The Hybrid System H_8.

We prove, by contradiction, that there are no $\mathcal{H}_P$ and $\mathcal{H}_S$ which recognize $\mathcal{L}_6$. If there exists a $\mathcal{H}_P$ which recognizes $\mathcal{L}_6$, then this contradicts the proof of proposition 3.6. Furthermore, if there exists a $\mathcal{H}_S$ which recognizes $\mathcal{L}_6$, then this contradicts the proof of proposition 3.4. So we have proved that $\mathcal{L}(\mathcal{H}^k_D) \not\subseteq \mathcal{L}(\mathcal{H}_P) \cup \mathcal{L}(\mathcal{H}_S)$.

Now we prove that $\mathcal{H}_S \not\subseteq \mathcal{H}_{Mix} \cup \mathcal{H}^k_D$. Let H_7 be the Hybrid System in figure 10 with l_3 as final location. The language $\mathcal{L}(H_7)$ is equal to $\mathcal{L}_7$.

The system H_7 is in $\mathcal{H}_S$, and we prove, by contradiction, that there are no

$\mathcal{H}_{Mix}$ and $\mathcal{H}_D^k$ which recognize $\mathcal{L}_7$. If there exists a $\mathcal{H}_{Mix}$ which recognizes $\mathcal{L}_7$, then this contradicts the proof of proposition 3.6. Furthermore, if there exists a $\mathcal{H}_D^k$ which recognizes $\mathcal{L}_7$, then this contradicts the proof of proposition 3.2. □

Proof of Proposition 3.8

Let H_8 be the Hybrid System in figure 11 with l_4 as final location. The language $\mathcal{L}(H_8)$ is equal to $\mathcal{L}_8$.

We prove, by contradiction, that there is no $\mathcal{H}_S$ and $\mathcal{H}_D^k$ which recognize $\mathcal{L}_8$. If there exists a $\mathcal{H}_S$ which recognizes $\mathcal{L}_8$, then this contradicts the proof of proposition 3.4. Moreover, if there exists a $\mathcal{H}_D^k$ which recognizes $\mathcal{L}_8$, then this contradicts the proof of proposition 3.2. □