Theoretical Computer Science 138 (1995) 3–34

The Algorithmic Analysis of Hybrid Systems

R. Alur* C. Courcoubetis† N. Halbwachs‡ T.A. Henzinger§ P.-H. Ho§
X. Nicollin‡ A. Olivero‡ J. Sifakis‡ S. Yovine‡

Abstract

We present a general framework for the formal specification and algorithmic analysis of hybrid systems. A hybrid system consists of a discrete program with an analog environment. We model hybrid systems as finite automata equipped with variables that evolve continuously with time according to dynamical laws. For verification purposes, we restrict ourselves to linear hybrid systems, where all variables follow piecewise-linear trajectories. We provide decidability and undecidability results for classes of linear hybrid systems, and we show that standard program-analysis techniques can be adapted to linear hybrid systems. In particular, we consider symbolic model-checking and minimization procedures that are based on the reachability analysis of an infinite state space. The procedures iteratively compute state sets that are definable as unions of convex polyhedra in multidimensional real space. We also present approximation techniques for dealing with systems for which the iterative procedures do not converge.

1 Introduction

A hybrid system consists of a discrete program with an analog environment. We assume that a run of a hybrid system is a sequence of steps. Within each step the system state evolves continuously according to a dynamical law until a transition occurs. Transitions are instantaneous state changes that separate continuous state evolutions.

We model a hybrid system as a finite automaton that is equipped with a set of variables. The control locations of the automaton are labeled with evolution laws. At a location the values of the variables change continuously with time according to the associated law. The transitions of the automaton are labeled with guarded sets of assignments. A transition is enabled when the associated guard is true, and its execution modifies the values of the variables according to the assignments. Each location is also labeled with an invariant condition that must hold when the control resides at the location. This model for hybrid systems is inspired by the phase transition systems of [MMP92, NSY93], and can be viewed as a generalization of timed safety automata [AD94, HNSY94].

The purpose of this paper is to demonstrate that standard program-analysis techniques can be adapted to hybrid systems. For verification purposes we restrict ourselves to linear hybrid systems. In a linear hybrid system, for each variable the rate of change is constant—though this constant may vary from location to location—and the terms involved in the invariants, guards, and assignments are required to be linear. An interesting special case of a linear hybrid system is a timed automaton [AD94]. In a timed automaton each continuously changing variable is an accurate clock whose rate of change with time is always 1. Furthermore, in a timed automaton all terms involved in assignments are constants, and all invariants and guards only involve comparisons of clock values with constants. Even though the

*AT&T Bell Laboratories, Murray Hill, NJ, U.S.A.
†University of Crete and ICS, FORTH, Heraklion, Greece. Partially supported by Esprit-BRA 6021 REACT-P.
‡VERIMAG-SPECTRE, Grenoble, France. VERIMAG is a joint laboratory of CNRS, INPG, UJF, and VERILOG S.A., associated with Institut IMAG. SPECTRE is an INRIA project. Partially supported by Esprit-BRA 6021 REACT-P.
§Computer Science Department, Cornell University, Ithaca, NY, U.S.A. Supported in part by the National Science Foundation under grant CCR-9200794, by the United States Air Force Office of Scientific Research under contract F49620-93-1-0056, and by the Defense Advanced Research Projects Agency under grant NAG2-892.

reachability problem for linear hybrid systems is undecidable, it can be solved for timed automata. In this paper, we provide new decidability and undecidability results for classes of linear hybrid systems, and we show that some algorithms for the analysis of timed automata can be extended to linear hybrid systems to obtain semidecision procedures for various verification problems.

In particular, we consider the symbolic model-checking method for timed automata presented in [HNSY94], and the minimization procedure for timed automata presented in [ACD+92]. Both methods perform a reachability analysis over an infinite state space. The procedures compute state sets by iterative approximation such that each intermediate result is definable by a linear formula; that is, each computed state set is a finite union of convex polyhedra in multidimensional real space. The termination of the procedures, however, is not guaranteed for linear hybrid systems. To cope with this problem, approximate analysis techniques are used to enforce the convergence of iterations by computing upper approximations of state sets. Approximate techniques yield either necessary or sufficient verification conditions.

The paper is essentially a synthesis of the results presented in [ACHH93, NOSY93, HPR94]. Section 2 presents a general model for hybrid systems. Section 3 defines linear hybrid systems, and presents decidability and undecidability results for the reachability problem of subclasses of linear hybrid systems. The verification methods are presented in Section 4. Some paradigmatic examples are specified and verified to illustrate the application of our results. These examples are analyzed using the Kronos tool [NSY92, NOSY93] (available from Grenoble) and the HyTech tool [AHH93, HH94] (available from Cornell), two symbolic model checkers for timed and hybrid systems.

2 A Model for Hybrid Systems

We specify hybrid systems by graphs whose edges represent discrete transitions and whose vertices represent continuous activities.

A *hybrid system* $H = (Loc, Var, Lab, Edg, Act, Inv)$ consists of six components:

- A finite set *Loc* of vertices called *locations*.
- A finite set *Var* of real-valued *variables*. A *valuation* ν for the variables is a function that assigns a real-value $\nu(x) \in \mathsf{R}$ to each variable $x \in Var$. We write V for the set of valuations.

  A *state* is a pair (ℓ, ν) consisting of a location $\ell \in Loc$ and a valuation $\nu \in V$. We write Σ for the set of states.
- A finite set *Lab* of *synchronization labels* that contains the *stutter label* $\tau \in Lab$.
- A finite set *Edg* of edges called *transitions*. Each transition $e = (\ell, a, \mu, \ell')$ consists of a *source* location $\ell \in Loc$, a *target* location $\ell' \in Loc$, a synchronization label $a \in Lab$, and a *transition relation* $\mu \subseteq V^2$. We require that for each location $\ell \in Loc$, there is a *stutter transition* of the form (ℓ, τ, Id, ℓ) where $Id = \{(\nu, \nu) \mid \nu \in V\}$.

  The transition e is *enabled* in a state (ℓ, ν) if for some valuation $\nu' \in V$, $(\nu, \nu') \in \mu$. The state (ℓ', ν'), then, is a *transition successor* of the state (ℓ, ν).
- A labeling function *Act* that assigns to each location $\ell \in Loc$ a set of *activities*. Each activity is a function from the nonnegative reals $\mathsf{R}^{\geq 0}$ to V. We require that the activities of each location are *time-invariant*: for all locations $\ell \in Loc$, activities $f \in Act(\ell)$, and nonnegative reals $t \in \mathsf{R}^{\geq 0}$, also $(f + t) \in Act(\ell)$, where $(f + t)(t') = f(t + t')$ for all $t' \in \mathsf{R}^{\geq 0}$.

  For all locations $\ell \in Loc$, activities $f \in Act(\ell)$, and variables $x \in Var$, we write f^x the function from $\mathsf{R}^{\geq 0}$ to R such that $f^x(t) = f(t)(x)$.
- A labeling function *Inv* that assigns to each location $\ell \in Loc$ an *invariant* $Inv(\ell) \subseteq V$.

The hybrid system H is *time-deterministic* if for every location $\ell \in Loc$ and every valuation $\nu \in V$, there is at most one activity $f \in Act(\ell)$ with $f(0) = \nu$. The activity f, then, is denoted by $\varphi_\ell[\nu]$.

The runs of a hybrid system

At any time instant, the state of a hybrid system is given by a control location and values for all variables. The state can change in two ways:

- By a *discrete* and *instantaneous* transition that changes both the control location and the values of the variables according to the transition relation;
- By a *time delay* that changes only the values of the variables according to the activities of the current location.

The system may stay at a location only if the location invariant is true; that is, some discrete transition must be taken before the invariant becomes false.

A *run* of the hybrid system H, then, is a finite or infinite sequence

$$\rho: \quad \sigma_0 \mapsto_{f_0}^{t_0} \sigma_1 \mapsto_{f_1}^{t_1} \sigma_2 \mapsto_{f_2}^{t_2} \cdots$$

of states $\sigma_i = (\ell_i, \nu_i) \in \Sigma$, nonnegative reals $t_i \in \mathsf{R}^{\geq 0}$, and activities $f_i \in Act(\ell_i)$, such that for all $i \geq 0$,

1. $f_i(0) = \nu_i$,
2. for all $0 \leq t \leq t_i$, $f_i(t) \in Inv(\ell_i)$,
3. the state σ_{i+1} is a transition successor of the state $\sigma_i' = (\ell_i, f_i(t_i))$.

The state σ_i' is called a *time successor* of the state σ_i; the state σ_{i+1}, a *successor* of σ_i. We write $[H]$ for the set of runs of the hybrid system H.

Notice that if we require all activities to be smooth functions, then the run ρ can be described by a piecewise smooth function whose values at the points of higher-order discontinuity are sequences of discrete state changes. Also notice that for time-deterministic systems, we can omit the subscripts f_i from the *next relation* $\mapsto$.

The run ρ *diverges* if ρ is infinite and the infinite sum $\sum_{i\geq 0} t_i$ diverges. The hybrid system H is *nonzeno* if every finite run of H is a prefix of some divergent run of H. Nonzeno systems can be executed [AH94].

Hybrid systems as transition systems

With the hybrid system H, we associate the labeled transition system $\mathcal{T}_H = (\Sigma, Lab \cup \mathsf{R}^{\geq 0}, \rightarrow)$, where the *step relation* $\rightarrow$ is the union of the *transition-step relations* $\rightarrow^a$, for $a \in Lab$,

$$\frac{(\ell, a, \mu, \ell') \in Edg \qquad (\nu, \nu') \in \mu \qquad \nu \in Inv(\ell) \qquad \nu' \in Inv(\ell')}{(\ell, \nu) \rightarrow^a (\ell', \nu')}$$

and the *time-step relations* $\rightarrow^t$, for $t \in \mathsf{R}^{\geq 0}$,

$$\frac{f \in Act(\ell) \qquad f(0) = \nu \qquad \forall 0 \leq t' \leq t.\ f(t') \in Inv(\ell)}{(\ell, \nu) \rightarrow^t (\ell, f(t))}$$

Notice that the stutter transitions ensure that the transition system $\mathcal{T}_H$ is reflexive.

There is a natural correspondence between the runs of the hybrid system H and the paths through the transition system $\mathcal{T}_H$: for all states $\sigma, \sigma' \in \Sigma$, where $\sigma = (\ell, \nu)$, and for all $t \in \mathsf{R}^{\geq 0}$,

$$\exists f \in Act(\ell), \sigma \mapsto_f^t \sigma' \quad \text{iff} \quad \exists \sigma'' \in \Sigma, a \in Lab.\ \sigma \rightarrow^t \sigma'' \rightarrow^a \sigma'.$$

It follows that for every hybrid system, the set of runs is closed under prefixes, suffixes, stuttering, and fusion [HNSY94].

For time-deterministic hybrid systems, the rule for the time-step relation can be simplified. *Time can progress* by the amount $t \in \mathsf{R}^{\geq 0}$ from the state (ℓ, ν) if this is permitted by the invariant of location ℓ; that is,

$$\mathsf{tcp}_\ell[\nu](t) \quad \text{iff} \quad \forall 0 \leq t' \leq t.\ \varphi_\ell[\nu](t') \in Inv(\ell).$$

Now we can rewrite the time-step rule for time-deterministic systems as

$$\frac{\mathsf{tcp}_\ell[\nu](t)}{(\ell,\nu) \rightarrow^t (\ell, \varphi_\ell[\nu](t))}$$

Example: thermostat

The temperature of a room is controlled through a thermostat, which continuously senses the temperature and turns a heater on and off. The temperature is governed by differential equations. When the heater is off, the temperature, denoted by the variable x, decreases according to the exponential function $x(t) = \theta e^{-Kt}$, where t is the time, θ is the initial temperature, and K is a constant determined by the room; when the heater is on, the temperature follows the function $x(t) = \theta e^{-Kt} + h(1 - e^{-Kt})$, where h is a constant that depends on the power of the heater. We wish to keep the temperature between m and M degrees and turn the heater on and off accordingly.

The resulting time-deterministic hybrid system is shown in Figure 1. The system has two locations: in location ℓ_0, the heater is turned off; in location ℓ_1, the heater is on. The transition relations are specified by guarded commands; the activities, by differential equations; and the location invariants, by logical formulas.

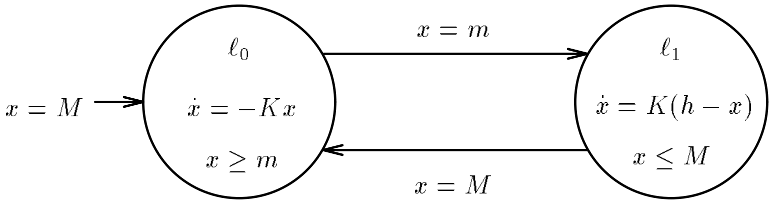


Figure 1: Thermostat

The parallel composition of hybrid systems

Let $H_1 = (Loc_1, Var, Lab_1, Edg_1, Act_1, Inv_1)$ and $H_2 = (Loc_2, Var, Lab_2, Edg_2, Act_2, Inv_2)$ be two hybrid systems over a common set Var of variables. The two hybrid systems synchronize on the common set $Lab_1 \cap Lab_2$ of synchronization labels; that is, whenever H_1 performs a discrete transition with the synchronization label $a \in Lab_1 \cap Lab_2$, then so does H_2.

The *product* $H_1 \times H_2$ is the hybrid system $(Loc_1 \times Loc_2, Var, Lab_1 \cup Lab_2, Edg, Act, Inv)$ such that

- $((\ell_1, \ell_2), a, \mu, (\ell'_1, \ell'_2)) \in Edg$ iff

  (1) $(\ell_1, a_1, \mu_1, \ell'_1) \in Edg_1$ and $(\ell_2, a_2, \mu_2, \ell'_2) \in Edg_2$,
  (2) either $a_1 = a_2 = a$, or either $a_1 = a \notin (Lab_1 \cap Lab_2)$ and $a_2 = \tau$, or $a_1 = \tau$ and $a_2 = a \notin (Lab_1 \cap Lab_2)$,
  (3) $\mu = \mu_1 \cap \mu_2$;

- $Act(\ell_1, \ell_2) = Act_1(\ell_1) \cap Act_2(\ell_2)$;

- $Inv(\ell_1, \ell_2) = Inv_1(\ell_1) \cap Inv_2(\ell_2)$.

It follows that all runs of the product system are runs of both component systems:

$$[H_1 \times H_2]_{Loc_1} \subseteq [H_1] \quad \text{and} \quad [H_1 \times H_2]_{Loc_2} \subseteq [H_2]$$

where $[H_1 \times H_2]_{Loc_i}$ is the projection of $[H_1 \times H_2]$ on Loc_i.

Notice also that the product of two time-deterministic hybrid systems is again time-deterministic.

3 Linear Hybrid Systems

A *linear term* over the set *Var* of variables is a linear combination of the variables in *Var* with integer coefficients. A *linear formula* over *Var* is a boolean combination of inequalities between linear terms over *Var*.

The time-deterministic hybrid system $H = (Loc, Var, Lab, Edg, Act, Inv)$ is *linear* if its activities, invariants, and transition relations can be defined by linear expressions over the set *Var* of variables:

1. For all locations $\ell \in Loc$, the activities $Act(\ell)$ are defined by a set of differential equations of the form $\dot{x} = k_x$, one for each variable $x \in Var$, where $k_x \in \mathsf{Z}$ is an integer constant: for all valuations $\nu \in V$, variables $x \in Var$, and nonnegative reals $t \in \mathsf{R}^{\geq 0}$,
$$\varphi_\ell^x[\nu](t) \; = \; \nu(x) + k_x \cdot t.$$
We write $Act(\ell, x) = k_x$ to refer to the *rate* of the variable x at location ℓ.

2. For all locations $\ell \in Loc$, the invariant $Inv(\ell)$ is defined by a linear formula ψ over *Var*:
$$\nu \in Inv(\ell) \quad \text{iff} \quad \nu(\psi).$$

3. For all transitions $e \in Edg$, the transition relation μ is defined by a guarded set of nondeterministic assignments
$$\psi \; \Rightarrow \; \{x := [\alpha_x, \beta_x] \mid x \in Var\},$$
where the guard ψ is a linear formula and for each variable $x \in Var$, both interval boundaries α_x and β_x are linear terms:
$$(\nu, \nu') \in \mu \quad \text{iff} \quad \nu(\psi) \wedge \forall x \in Var.\, \nu(\alpha_x) \leq \nu'(x) \leq \nu(\beta_x).$$
If $\alpha_x = \beta_x$, we write $\mu(e, x) = \alpha_x$ to refer to the updated value of the variable x after the transition e.

Notice that every run of a linear hybrid system can be described by a piecewise linear function whose values at the points of first-order discontinuity are finite sequences of discrete state changes.

Special cases of linear hybrid systems

Various special cases of linear hybrid systems are of particular interest:

- If $Act(\ell, x) = 0$ for each location $\ell \in Loc$, then x is a *discrete variable*. Thus, a discrete variable changes only when the control location changes. A *discrete system* is a linear hybrid system all of whose variables are discrete.

- A discrete variable x is a *proposition* if $\mu(e, x) \in \{0, 1\}$ for each transition $e \in Edg$. A *finite-state system* is a linear hybrid system all of whose variables are propositions.

- If $Act(\ell, x) = 1$ for each location ℓ and $\mu(e, x) \in \{0, x\}$ for each transition e, then x is a *clock*. Thus, (1) the value of a clock increases uniformly with time, and (2) a discrete transition either resets a clock to 0, or leaves it unchanged. A *timed automaton* [AD94] is a linear hybrid system all of whose variables are propositions or clocks, and the linear expressions are boolean combinations of inequalities of the form $x \# c$ or $x - y \# c$ where c is a nonnegative integer and $\# \in \{<, \leq, =, >, \geq\}$.

- If there is a nonzero integer constant $k \in \mathsf{Z}$ such that $Act(\ell, x) = k$ for each location ℓ and $\mu(e, x) \in \{0, x\}$ for each transition e, then x is a *skewed clock*. Thus, a skewed clock is similar to a clock except that it changes with time at some fixed rate different from 1. A *multirate timed system* is a linear hybrid system all of whose variables are propositions and skewed clocks. An *n-rate timed system* is a multirate timed system whose skewed clocks proceed at n different rates.

- If $Act(\ell, x) \in \{0, 1\}$ for each location ℓ and $\mu(e, x) \in \{0, x\}$ for each transition e, then x is an *integrator*. Thus, an integrator is a clock that can be stopped and restarted; it is typically used to measure accumulated durations. An *integrator system* is a linear hybrid system all of whose variables are propositions and integrators.
- A discrete variable x is a *parameter* if $\mu(e, x) = x$ for each transition $e \in Edg$. Thus, a parameter is a symbolic constant. For each of the subclasses of linear hybrid systems listed above, we obtain *parameterized* versions by admitting parameters.

Notice that linear hybrid systems, and all of the subclasses of linear hybrid systems listed above, are closed under parallel composition.

3.1 Examples of Linear Hybrid Systems

A water-level monitor

The water level in a tank is controlled through a monitor, which continuously senses the water level and turns a pump on and off. The water level changes as a piecewise-linear function over time. When the pump is off, the water level, denoted by the variable y, falls by 2 inches per second; when the pump is on, the water level rises by 1 inch per second. Suppose that initially the water level is 1 inch and the pump is turned on. We wish to keep the water level between 1 and 12 inches. But from the time that the monitor signals to change the status of the pump to the time that the change becomes effective, there is a delay of 2 seconds. Thus the monitor must signal to turn the pump on before the water level falls to 1 inch, and it must signal to turn the pump off before the water level reaches 12 inches.

The linear hybrid system of Figure 2 describes a water-level monitor that signals whenever the water level passes 5 and 10 inches, respectively. The system has four locations: in locations 0 and 1, the pump is turned on; in locations 2 and 3, the pump is off. The clock x is used to specify the delays: whenever the control is in location 1 or 3, the signal to switch the pump off or on, respectively, was sent x seconds ago. In the next section, we will prove that the monitor indeed keeps the water level between 1 and 12 inches.

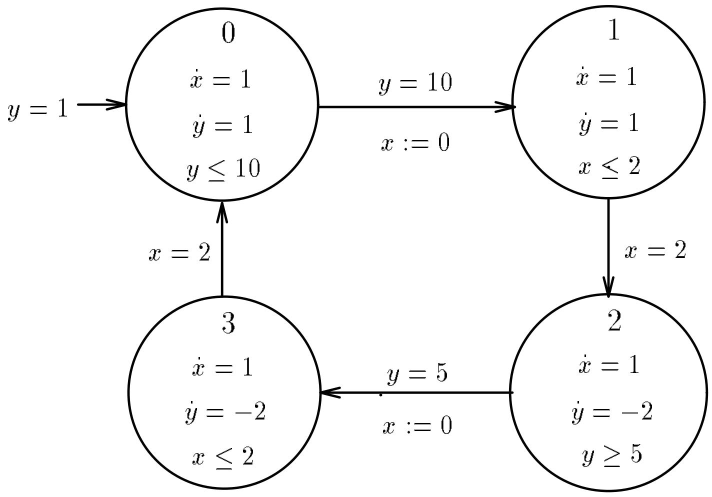


Figure 2: Water-level monitor

A mutual-exclusion protocol

This example describes a parameterized multirate timed system. We present a timing-based algorithm that implements mutual exclusion for a distributed system with skewed clocks. Consider an asynchronous shared-memory system that consists of two processes P_1 and P_2 with atomic read and write operations. Each process has a critical section and at each time instant, at most one of the two processes is allowed to be in its critical section. Mutual exclusion is ensured by a version of Fischer's protocol [Lam87], which we describe first in pseudocode. For each process P_i, where $i = 1, 2$:

```
repeat
    repeat
        await k = 0
        k := i
        delay b
    until k = i
    Critical section
    k := 0
forever
```

The two processes P_1 and P_2 share a variable k and process P_i is allowed to be in its critical section iff $k = i$. Each process has a private clock. The instruction **delay** b delays a process for at least b time units as measured by the process's local clock. Furthermore, each process takes at most a time units, as measured by the process's clock, for a single write access to the shared memory (i.e., for the assignment $k := i$). The values of a and b are the only information we have about the timing behavior of instructions. Clearly, the protocol ensures mutual exclusion only for certain values of a and b. If both private processor clocks proceed at precisely the same rate, then mutual exclusion is guaranteed iff $a < b$.

To make the example more interesting, we assume that the two private clocks of the processes P_1 and P_2 proceed at different rates, namely, the local clock of P_2 is 1.1 times faster than the clock of P_1. The resulting system can be modeled by the product of the two hybrid systems presented in Figure 3.

Each of the two graphs models one process, with the two critical sections being represented by the locations 4 and D. The private clocks of the processes P_1 and P_2 determine the rate of change of the two skewed-clock variables x and y, respectively.

A leaking gas burner

Now we consider an integrator system. In [CHR91], the duration calculus is used to prove that a gas burner does not leak excessively. It is assumed that (1) any leakage can be detected and stopped within 1 second and (2) the gas burner will not leak for 30 seconds after a leakage has been stopped. We wish to prove that the accumulated time of leakage is at most one twentieth of the time in any interval of at least 60 seconds. The system is modeled by the hybrid system of Figure 4. The system has two locations: in location 1, the gas burner leaks; location 2 is the nonleaking location. The integrator z records the cumulative leakage time; that is, the accumulated amount of time that the system has spent in location 1. The clock x records the time the system has spent in the current location; it is used to specify the properties (1) and (2). The clock y records the total elapsed time. In the next section, we will prove that $y \geq 60 \;\Rightarrow\; 20z \leq y$ is an invariant of the system.

A temperature control system

This example appears in [JLHM91]. The system controls the coolant temperature in a reactor tank by moving two independent control rods. The goal is to maintain the coolant between the temperatures θ_m and θ_M. When the temperature reaches its maximum value θ_M, the tank must be refrigerated with one of the rods. The temperature rises at a rate v_r and decreases at rates v_1 and v_2 depending on which rod is being used. A rod can be moved again only if T time units have elapsed since the end of its previous movement. If the temperature of the coolant cannot decrease because there is no available rod, a complete shutdown is required. Figure 5 shows the hybrid system of this example: variable θ measures

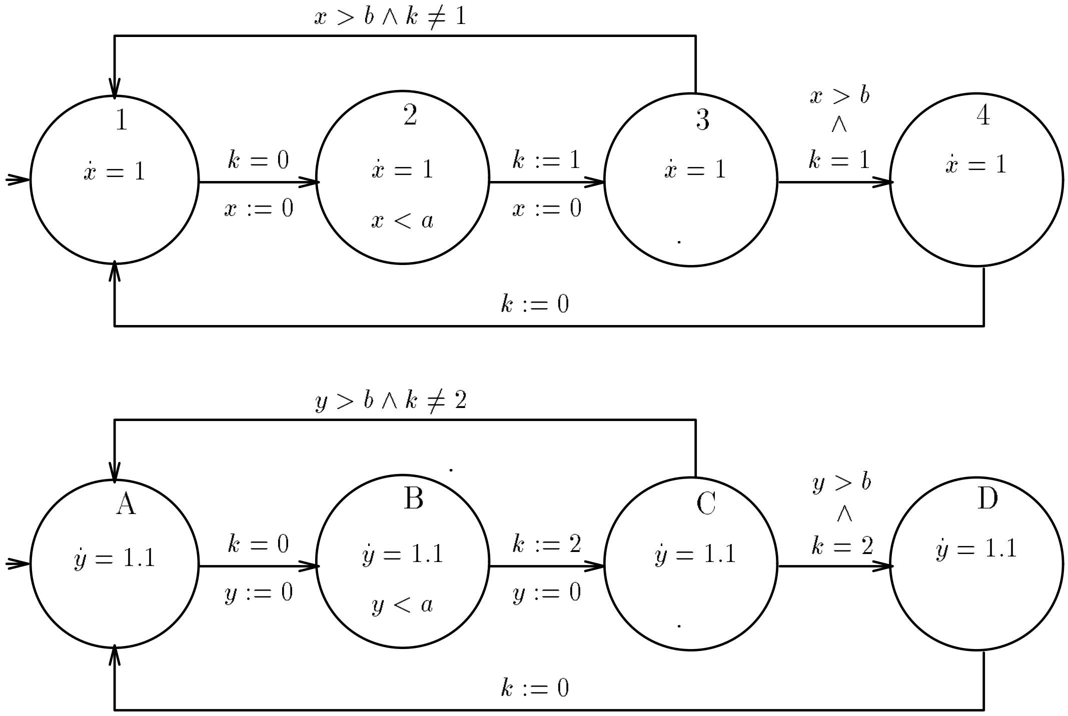


Figure 3: Mutual-exclusion protocol

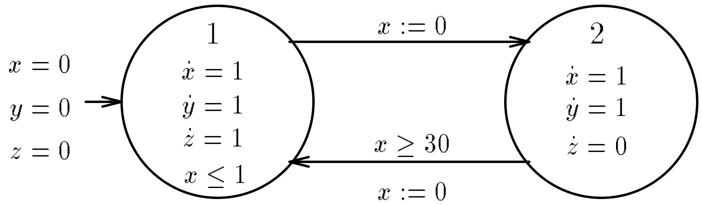


Figure 4: Leaking gas burner

the temperature, and the values of clocks x_1 and x_2 represent the times elapsed since the last use of rod 1 and rod 2, respectively.

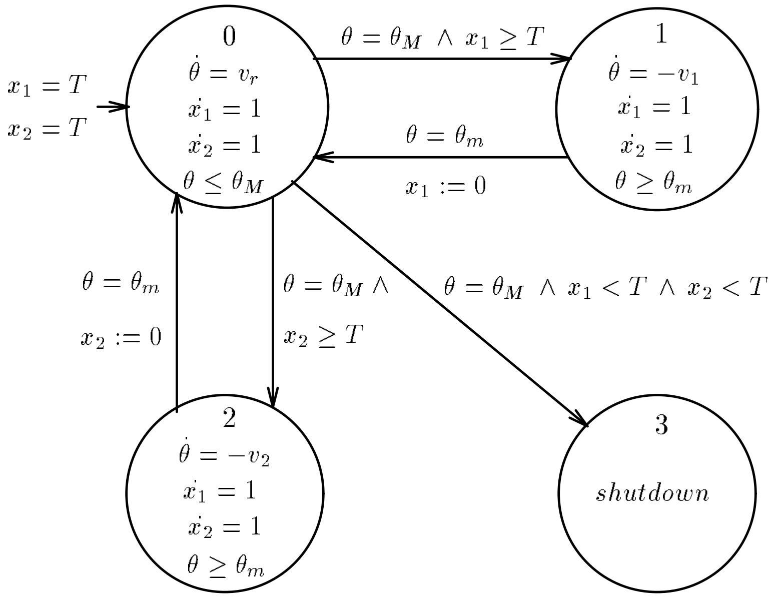


Figure 5: Temperature control system

A game of billiards

Consider a billiard table of dimensions l and h, with a grey ball and a white ball (Figure 6).

Initially, the balls are placed at positions $b_g = (x_g, y_g)$ and $b_w = (x_w, y_w)$. The grey ball is knocked and starts moving with constant velocity v. If the ball reaches a vertical side then it rebounds, i.e., the sign of the horizontal velocity component v_x changes. The same occurs with the vertical velocity component v_y when the ball reaches a horizontal side. The combination of signs of velocity components gives four different directions of movement.

The hybrid system shown in Figure 7 describes the movement of the grey ball for the billiards game. Each possible combination of directions is represented by a location. The rebounds correspond to the execution of transitions between locations.

3.2 The Reachability Problem for Linear Hybrid Systems

Let σ and σ' be two states of a hybrid system H. The state σ' is *reachable* from the state σ, written $\sigma \mapsto^* \sigma'$, if there is a run of H that starts in σ and ends in σ'. The *reachability question* asks, then, if $\sigma \mapsto^* \sigma'$ for two given states σ and σ' of a hybrid system H.

The reachability problem is central to the verification of hybrid systems. In particular, the verification of invariance properties is equivalent to the reachability question: a set $R \subseteq \Sigma$ of states is an invariant of the hybrid system H iff no state in $\Sigma - R$ is reachable from an initial state of H.

A decidability result

A linear hybrid system is *simple* if all linear atoms in location invariants and transition guards are of the form $x \leq k$ or $k \leq x$, for a variable $x \in Var$ and an integer constant $k \in \mathbb{Z}$. In particular, for multirate timed systems the simplicity condition prohibits the comparison of skewed clocks with different rates.

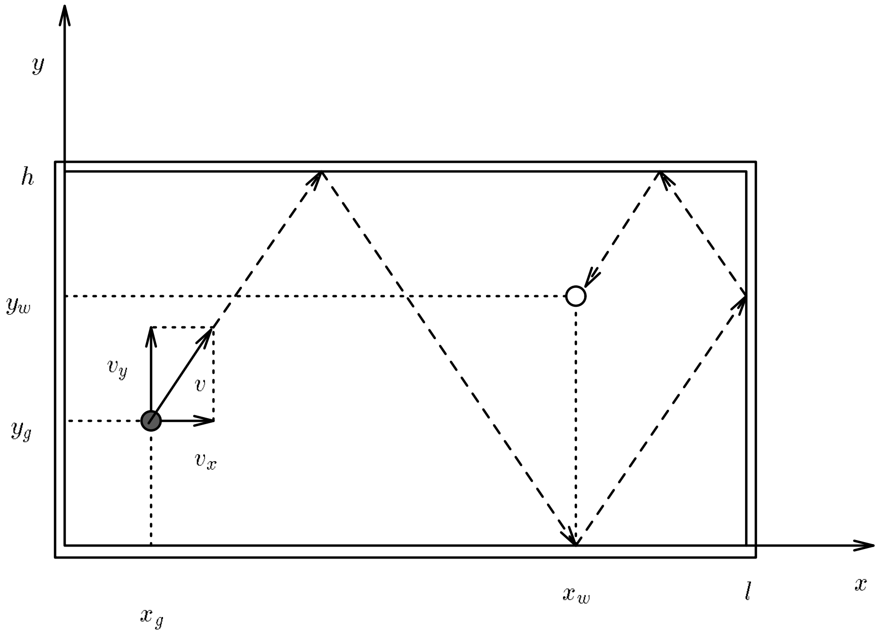


Figure 6: Billiards game

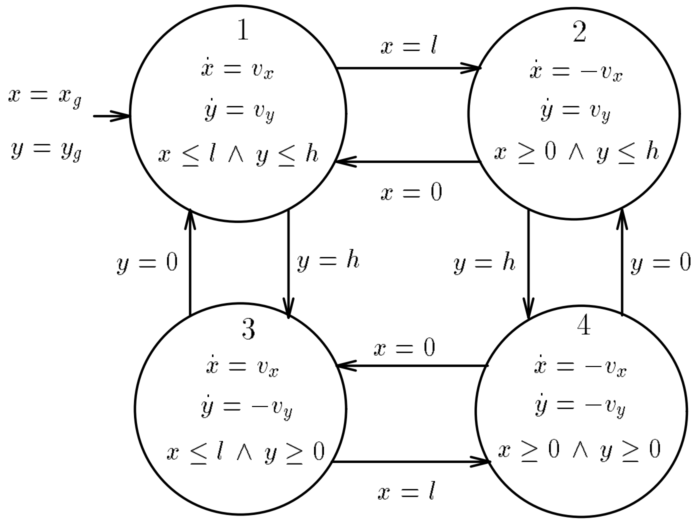


Figure 7: Movement of the grey ball

Theorem 3.1 *The reachability problem is decidable for simple multirate timed systems.*

Proof. Let H be a simple multirate timed system. We translate H into a timed automaton $sc(H)$: (1) adjust the rates of all skewed clocks to 1, and (2) replace all occurrences of each skewed clock x in location invariants and transition guards with $k_x \cdot x$. Given a valuation ν of H, let the valuation $sc(\nu)$ be such that $sc(\nu)(x) = k_x \cdot \nu(x)$ for all skewed clocks x and $sc(\nu)(p) = \nu(p)$ for all propositions p; moreover, $sc(\ell, \nu) = (\ell, sc(\nu))$. It is not difficult to check that there is a run of H from σ to σ' iff there is a run of $sc(H)$ from $sc(\sigma)$ to $sc(\sigma')$. The reachability problem for timed automata is solved in [ACD93]. ■

Two undecidability results

Theorem 3.2 *The reachability problem is undecidable for 2-rate timed systems.*

Proof. The theorem follows from the undecidability of the halting problem for nondeterministic 2-counter machines. Given any two distinct clock rates, a 2-rate timed system can encode the computations of the given 2-counter machine M. For the 2-rate timed system H, we use "accurate" clocks of rate 1 and skewed clocks of rate 2. We use an accurate clock y to mark intervals of length 1: the clock y is zero initially, and is reset whenever it reaches 1. The i-th configuration of the machine M is encoded by the state of H at time i. The location of H encodes the program counter of M, and the values of two accurate clocks x_1 and x_2 encode the counter values: the counter value n is encoded by the clock value $1/2^n$.

Encoding the program counter, setting up the initial configuration, and testing a counter for being 0, is straightforward. Hence it remains to be shown how to update the counter values. Suppose at time i the value of an accurate clock x is $1/2^n$, that is, suppose that the clock x is reset to 0 at time $i - 1/2^n$. Suppose the value of the counter encoded by x stays unchanged. Then simply reset x to 0 when its value reaches 1 (that is, at time $(i + 1 - 1/2^n)$); the value of x at time $i + 1$ will then be $1/2^n$. To increment the counter represented by x, reset an accurate clock z when the value of x reaches 1, then nondeterministically reset both x and a skewed clock z' in the interval $(i + 1 - 1/2^n, i + 1)$ and test $z = z'$ at time $i + 1$. The equality test ensures that the value of the skewed clock z' is $1/2^n$ at time $i + 1$, and hence, the value of x is $1/2^{n+1}$ at time $i + 1$. To decrement the counter represented by x, nondeterministically reset an accurate clock z in the interval $(i - 1, i - 1/2^n)$, reset a skewed clock z' simultaneously with x at time $i - 1/2^n$, and test the condition $z = z'$ at time i. This ensures that the value of z at time i is $1/2^{n-1}$. Then resetting the clock x when the value of z reaches 1 ensures that the value of x is $1/2^{n-1}$ at time $i + 1$.

Thus, the runs of H encode the runs of M, and the halting problem for M is reduced to a reachability problem for H. ■

Theorem 3.3 *The reachability problem is undecidable for simple integrator systems.*

Proof. This is proved in [Čer92]. ■

4 The Verification of Linear Hybrid Systems

We present a methodology for analyzing linear hybrid systems that is based on predicate transformers for computing the step predecessors and the step successors of a given set of states. Throughout this section, let $H = (Loc, Var, Lab, Edg, Act, Inv)$ be a linear hybrid system.

4.1 Forward Analysis

Given a location $\ell \in Loc$ and a set of valuations $P \subseteq V$, the *forward time closure* $\langle P \rangle_\ell^{\nearrow}$ of P at ℓ is the set of valuations that are reachable from some valuation $\nu \in P$ by letting time progress:

$$\nu' \in \langle P \rangle_\ell^{\nearrow} \quad \text{iff} \quad \exists \nu \in V, t \in \mathsf{R}^{\geq 0}.\ \nu \in P \wedge \mathsf{tcp}_\ell[\nu](t) \wedge \nu' = \varphi_\ell[\nu](t).$$

Thus, for all valuations $\nu' \in \langle P \rangle_\ell^{\nearrow}$, there exist a valuation $\nu \in P$ and a nonnegative real $t \in \mathsf{R}^{\geq 0}$ such that $(\ell, \nu) \rightarrow^t (\ell, \nu')$.

Given a transition $e = (\ell, a, \mu, \ell')$ and a set of valuations $P \subseteq V$, the *postcondition* $\mathsf{post}_e[P]$ of P with respect to e is the set of valuations that are reachable from some valuation $\nu \in P$ by executing the transition e:

$$\nu' \in \mathsf{post}_e[P] \quad \text{iff} \quad \exists \nu \in V.\ \nu \in P \wedge (\nu, \nu') \in \mu.$$

Thus, for all valuations $\nu' \in \mathsf{post}_e[P]$, there exists a valuation $\nu \in P$ such that $(\ell, \nu) \rightarrow^a (\ell', \nu')$.

A set of states is called a *region*. Given a set $P \subseteq V$ of valuations, by (ℓ, P) we denote the region $\{(\ell, \nu) \mid \nu \in P\}$; we write $(\ell, \nu) \in (\ell, P)$ iff $\nu \in P$. The forward time closure and the postcondition can be naturally extended to regions: for $R = \bigcup_{\ell \in Loc} (\ell, R_\ell)$,

$$\begin{aligned} \langle R \rangle^{\nearrow} &= \bigcup_{\ell \in Loc} (\ell, \langle R_\ell \rangle_\ell^{\nearrow}) \\ \mathsf{post}[R] &= \bigcup_{e = (\ell, \ell') \in Edg} (\ell', \mathsf{post}_e[R_\ell]) \end{aligned}$$

A *symbolic run* of the linear hybrid system H is a finite or infinite sequence

$$\varrho: \quad (\ell_0, P_0)\,(\ell_1, P_1) \ldots (\ell_i, P_i) \ldots$$

of regions such that for all $i \geq 0$, there exists a transition e_i from ℓ_i to ℓ_{i+1} and

$$P_{i+1} = \mathsf{post}_{e_i}[\langle P_i \rangle_{\ell_i}^{\nearrow}];$$

that is, the region (ℓ_{i+1}, P_{i+1}) is the set of states that are reachable from a state $(\ell_0, \nu_0) \in (\ell_0, P_0)$ after executing the sequence $e_0, \ldots, e_i$ of transitions. There is a natural correspondence between the runs and the symbolic runs of the linear hybrid system H. The symbolic run ϱ represents the set of all runs of the form

$$(\ell_0, \nu_0) \mapsto^{t_0} (\ell_1, \nu_1) \mapsto^{t_1} \cdots$$

such that $(\ell_i, \nu_i) \in (\ell_i, P_i)$ for all $i \geq 0$. Besides, every run of H is represented by some symbolic run of H.

Given a region $I \subseteq \Sigma$, the *reachable region* $(I \mapsto^*) \subseteq \Sigma$ of I is the set of all states that are reachable from states in I:

$$\sigma \in (I \mapsto^*) \quad \text{iff} \quad \exists \sigma' \in I.\ \sigma' \mapsto^* \sigma.$$

Notice that $I \subseteq (I \mapsto^*)$.

The following proposition suggests a method for computing the reachable region $(I \mapsto^*)$ of I.

Proposition 4.1 *Let $I = \bigcup_{\ell \in Loc} (\ell, I_\ell)$ be a region of the linear hybrid system H. The reachable region $(I \mapsto^*) = \bigcup_{\ell \in Loc} (\ell, R_\ell)$ is the least fixpoint of the equation*

$$X = \langle I \cup \mathsf{post}[X] \rangle^{\nearrow}$$

or, equivalently, for all locations $\ell \in Loc$, the set R_ℓ of valuations is the least fixpoint of the set of equations

$$X_\ell = \langle I_\ell \cup \bigcup_{e = (\ell', \ell) \in Edg} \mathsf{post}_e[X_{\ell'}] \rangle_\ell^{\nearrow}.$$

Let ψ be a linear formula over *Var*. By $[\![\psi]\!]$ we denote the set of valuations that satisfy ψ. A set $P \subseteq V$ of valuations is *linear* if P is definable by a linear formula; that is, $P = [\![\psi]\!]$ for some linear formula ψ. If *Var* contains n variables, then a linear set of valuations can be thought of as a union of polyhedra in n-dimensional space.

Lemma 4.1 *For all linear hybrid systems H, if $P \subseteq V$ is a linear set of valuations, then for all locations $\ell \in Loc$ and transitions $e \in Edg$, both $\langle P \rangle_\ell^{\nearrow}$ and $\mathsf{post}_e[P]$ are linear sets of valuations.*

Given a linear formula ψ, we write $\langle\psi\rangle_\ell^{\nearrow}$ and $\mathsf{post}_e[\psi]$ for the linear formulas that define the sets of valuations $\langle[\![\psi]\!]\rangle_\ell^{\nearrow}$ and $\mathsf{post}_e[[\![\psi]\!]]$, respectively.

Let $pc \notin Var$ be a *control variable* that ranges over the set Loc of locations and let $R = \bigcup_{\ell\in Loc}(\ell, R_\ell)$ be a region. The region R is *linear* if for every location $\ell \in Loc$, the set R_ℓ of valuations is linear. If the sets R_ℓ are defined by the linear formulas ψ_ℓ, then the region R is defined by the linear formula

$$\psi \;=\; \bigvee_{\ell\in Loc} (pc = \ell \wedge \psi_\ell);$$

that is, $[\![\psi]\!] = R$. Hence, by Lemma 4.1, for all linear hybrid systems, if R is a linear region, then so are both $\langle R\rangle^{\nearrow}$ and $\mathsf{post}[R]$.

Using Proposition 4.1, we compute the reachable region $(I \mapsto^*)$ of a region I by successive approximation. Lemma 4.1 ensures that all regions computed in the process are linear. Since the reachability problem for linear hybrid systems is undecidable, the successive-approximation procedure does not terminate in general. The procedure does terminate for simple multirate timed systems (Theorem 3.1) and for the following example.

Example: the leaking gas burner

Let I be the set of initial states defined by the linear formula

$$\psi_I \;=\; (pc = 1 \wedge x = y = z = 0).$$

The set $(I \mapsto^*)$ of reachable states is characterized by the least fixpoint of the two equations

$$\begin{aligned}\psi_1 &= \langle x = y = z = 0 \vee \mathsf{post}_{(2,1)}[\psi_2]\rangle_1^{\nearrow}\\ \psi_2 &= \langle \mathit{false} \vee \mathsf{post}_{(1,2)}[\psi_1]\rangle_2^{\nearrow}\end{aligned}$$

which can be iteratively computed as

$$\begin{aligned}\psi_{1,i} &= \psi_{1,i-1} \vee \langle \mathsf{post}_{(2,1)}[\psi_{2,i-1}]\rangle_1^{\nearrow}\\ \psi_{2,i} &= \psi_{2,i-1} \vee \langle \mathsf{post}_{(1,2)}[\psi_{1,i-1}]\rangle_2^{\nearrow}\end{aligned}$$

where $\psi_{1,0} = \langle x = y = z = 0\rangle_1^{\nearrow} = (x \leq 1 \wedge y = x = z)$ and $\psi_{2,0} = \mathit{false}$. For $i = 1$, we have

$$\begin{aligned}\psi_{1,1} &= \psi_{1,0} \vee \langle \mathsf{post}_{(2,1)}[\psi_{2,0}]\rangle_1^{\nearrow}\\ &= \psi_{1,0}\\ \psi_{2,1} &= \psi_{2,0} \vee \langle \mathsf{post}_{(1,2)}[\psi_{1,1}]\rangle_2^{\nearrow}\\ &= \langle \mathsf{post}_{(1,2)}[x \leq 1 \wedge y = x = z = 0]\rangle_2^{\nearrow}\\ &= \langle (x = 0 \wedge y \leq 1 \wedge z = y)\rangle_2^{\nearrow}\\ &= (z \leq 1 \wedge y = z + x)\end{aligned}$$

Now, it is easy to show by induction that for all $i \geq 2$,

$$\psi_{1,i} = \psi_{1,i-1} \vee x \leq 1 \wedge 0 \leq z - x \leq i \wedge 30i + z \leq y$$

and

$$\psi_{2,i} = \psi_{2,i-1} \vee y \leq i + 1 \wedge 30i + x + z \leq y$$

Hence, the least solution of the equations above is the linear formula

$$\psi_R \;=\; (pc = 1 \wedge \psi_1) \vee (pc = 2 \wedge \psi_2)$$

where

$$
\begin{aligned}
\psi_1 &= x \leq 1 \wedge x = y = z \vee \exists i \geq 1.\, (x \leq 1 \wedge 0 \leq z - x \leq i \wedge 30i + z \leq y) \\
&= (x \leq 1 \wedge x = y = z) \vee (x \leq 1 \wedge x \leq z \wedge y + 30x \geq 31z) \\
\psi_2 &= z \leq 1 \wedge y = x + z \wedge x \geq 0 \vee \exists i \geq 1.\, (z \leq i + 1 \wedge 30i + x + z \leq y) \\
&= (z \leq 1 \wedge y = x + z \wedge x \geq 0) \vee y \geq x + 31z - 30
\end{aligned}
$$

This characterization of the reachable states can be used to verify invariance properties of the gas burner system (ψ_R is the strongest invariant of the system). For instance, the formula ψ_R implies the design requirement $y \geq 60 \Rightarrow 20z \leq y$.

4.2 Backward Analysis

The forward time closure and the postcondition define the successor of a region R. Dually, we can compute the predecessor of R.

Given a location $\ell \in Loc$ and a set of valuations $P \subseteq V$, the *backward time closure* of P at ℓ is the set of valuations from which it is possible to reach some valuation $\nu \in P$ by letting time progress:

$$\nu' \in \langle P \rangle_\ell^{\swarrow} \quad \text{iff} \quad \exists \nu \in V, t \in \mathsf{R}^{\geq 0}.\, \nu = \varphi_\ell[\nu'](t) \wedge \nu \in P \wedge \mathsf{tcp}_\ell[\nu'](t).$$

Thus, for all valuations $\nu' \in \langle P \rangle_\ell^{\swarrow}$, there exist a valuation $\nu \in P$ and a nonnegative real $t \in \mathsf{R}^{\geq 0}$ such that $(\ell, \nu') \rightarrow^t (\ell, \nu)$.

Given a transition $e = (\ell, a, \mu, \ell')$ and a set of valuations $P \subseteq V$, the *precondition* $\mathsf{pre}_e[P]$ of P with respect to e is the set of valuations from which it is possible to reach a valuation $\nu \in P$ by executing the transition e:

$$\nu' \in \mathsf{pre}_e[P] \quad \text{iff} \quad \exists \nu \in V.\, \nu \in P \wedge (\nu', \nu) \in \mu.$$

Thus, for all valuations $\nu' \in \mathsf{pre}_e[P]$, there exists a valuation $\nu \in P$ such that $(\ell, \nu') \rightarrow^a (\ell', \nu)$.

The backward time closure and the precondition can be naturally extended to regions: for $R = \bigcup_{\ell \in Loc} (\ell, R_\ell)$,

$$
\begin{aligned}
\langle R \rangle^{\swarrow} &= \bigcup_{\ell \in Loc} (\ell, \langle R_\ell \rangle_\ell^{\swarrow}) \\
\mathsf{pre}[R] &= \bigcup_{e = (\ell', \ell) \in Edg} (\ell', \mathsf{pre}_e[R_\ell])
\end{aligned}
$$

Given a region $R \subseteq \Sigma$, the *initial region* $(\mapsto^* R) \subseteq \Sigma$ of R is the set of all states from which a state in R is reachable:

$$\sigma \in (\mapsto^* R) \quad \text{iff} \quad \exists \sigma' \in R.\, \sigma \mapsto^* \sigma'.$$

Notice that $R \subseteq (\mapsto^* R)$.

The following proposition suggests a method for computing the initial region $(\mapsto^* R)$ of R.

Proposition 4.2 *Let $R = \bigcup_{\ell \in Loc} (\ell, R_\ell)$ be a region of the linear hybrid system H. The initial region $I = \bigcup_{\ell \in Loc} (\ell, I_\ell)$ is the least fixpoint of the equation*

$$X = \langle R \cup \mathsf{pre}[X] \rangle^{\swarrow}$$

or, equivalently, for all locations $\ell \in Loc$, the set I_ℓ of valuations is the least fixpoint of the set

$$X_\ell = \langle R_\ell \cup \bigcup_{e = (\ell, \ell') \in Edg} \mathsf{pre}_e[X_{\ell'}] \rangle_\ell^{\swarrow}$$

of equations.

Lemma 4.2 *For all linear hybrid systems H, if $P \subseteq V$ is a linear set of valuations, then for all locations $\ell \in Loc$ and transitions $e \in Edg$, both $\langle P \rangle_\ell^{\swarrow}$ and $\mathsf{pre}_e[P]$ are linear sets of valuations.*

It follows that for all linear hybrid systems, if R is a linear region, then so are both $\langle R \rangle^{\swarrow}$ and $\mathsf{pre}[R]$. Given a linear formula ψ, we write $\langle \psi \rangle_\ell^{\swarrow}$ and $\mathsf{pre}_e[\psi]$ for the linear formulas that define the sets of valuations $\langle [\![\psi]\!] \rangle_\ell^{\swarrow}$ and $\mathsf{pre}_e[[\![\psi]\!]]$, respectively.

Example: the leaking gas burner

We apply backward analysis to prove that the design requirement $y \geq 60 \;\Rightarrow\; 20z \leq y$ is an invariant of the gas burner system; that is, the region R defined by the linear formula

$$\psi_R \;=\; (y \geq 60 \,\wedge\, 20z > y)$$

is not reachable from the set I of initial states defined by the linear formula

$$\psi_I \;=\; (pc = 1 \,\wedge\, x = y = z = 0).$$

The set $(\mapsto^* R)$ of states from which it is possible to reach a state in R is characterized by the least fixpoint of the two equations

$$\begin{aligned}
\psi_1 &= \langle (y \geq 60 \,\wedge\, 20z > y) \vee \mathsf{pre}_{(1,2)}[\psi_2] \rangle_1^{\swarrow} \\
\psi_2 &= \langle (y \geq 60 \,\wedge\, 20z > y) \vee \mathsf{pre}_{(2,1)}[\psi_1] \rangle_2^{\swarrow}
\end{aligned}$$

which can be iteratively computed as

$$\begin{aligned}
\psi_{1,i} &= \langle \mathsf{pre}_{(1,2)}[\psi_{2,i-1}] \rangle_1^{\swarrow} \\
\psi_{2,i} &= \langle \mathsf{pre}_{(2,1)}[\psi_{1,i-1}] \rangle_2^{\swarrow}
\end{aligned}$$

where $\psi_{1,0} = \langle (y \geq 60 \,\wedge\, 20z > y) \rangle_1^{\swarrow}$ and $\psi_{2,0} = \langle (y \geq 60 \,\wedge\, 20z > y) \rangle_2^{\swarrow}$. Then,

$$\begin{aligned}
\psi_{1,0} &= (-19 < 20\,z - 19\,x - y \,\wedge\, 59 < -x + y \,\wedge\, x \leq 1), \\
\psi_{2,0} &= (0 < 20\,z + x - y \,\wedge\, 0 < 20\,z - y \,\wedge\, 3 < z), \\
\psi_{1,1} &= (-19 < 20\,z - y - 19\,x \,\wedge\, 2 < z - x \,\wedge\, x \leq 1), \\
\psi_{2,1} &= (-19 < 20\,z - y \,\wedge\, 2 < z \,\wedge\, 11 < 20\,z + x - y), \\
\psi_{1,2} &= (-8 < 20\,z - 19\,x - y \,\wedge\, 1 < z - x \,\wedge\, x \leq 1), \\
\psi_{2,2} &= (-19 < 20\,z - y \,\wedge\, 2 < z \,\wedge\, 11 < 20\,z + x - y), \\
\psi_{1,3} &= (-8 < 20\,z - 19\,x - y \,\wedge\, 1 < z - x \,\wedge\, x \leq 1), \\
\psi_{2,3} &= (-8 < 20\,z - y \,\wedge\, 1 < z \,\wedge\, 22 < 20\,z + x - y), \\
\psi_{1,4} &= (3 < 20\,z - 19\,x - y \,\wedge\, 0 < z - x \,\wedge\, x \leq 1), \\
\psi_{2,4} &= (-8 < 20\,z - y \,\wedge\, 1 < z \,\wedge\, 22 < 20\,z + x - y), \\
\psi_{1,5} &= (3 < 20\,z - 19\,x - y \,\wedge\, 0 < z - x \,\wedge\, x \leq 1), \\
\psi_{2,5} &= (3 < 20\,z - y \,\wedge\, 0 < z \,\wedge\, 33 < 20\,z + x - y), \\
\psi_{1,6} &= (14 < 20\,z - 19\,x - y \,\wedge\, -1 < z - x \,\wedge\, x \leq 1), \text{ and} \\
\psi_{2,6} &= (3 < 20\,z - y \,\wedge\, 0 < z \,\wedge\, 33 < 20\,z + x - y).
\end{aligned}$$

Since $\psi_{1,7} \;\Rightarrow\; \psi_{1,6}$ and $\psi_{2,7} \;\Rightarrow\; \psi_{2,6}$, the solution ψ is

$$\bigvee_{0 \leq i \leq 6} (pc = 1 \,\wedge\, \psi_{1,i}) \vee (pc = 2 \,\wedge\, \psi_{2,i}),$$

which contains no initial states; that is, $\psi_I \,\wedge\, \psi = \mathit{false}$. It follows that the design requirement is an invariant.

4.3 Approximate Analysis

In this section, we briefly present an approximate technique for dealing with systems where the (forward or backward) iterative procedure does not converge. For more details, see [HH94, HPR94]. We will compute *upper approximations* of the sets

- $(I \mapsto^*)$ of states which are reachable from the initial states I (forward analysis)
- $(\mapsto^* R)$ of states from which the region R is reachable (backward analysis)

We focus on forward analysis, backward analysis is similar. Let us come back to the system of fixpoint equations whose least solution gives, for each location ℓ, the set X_ℓ of reachable states at location ℓ:

$$X_\ell \quad = \quad \langle I_\ell \cup \bigcup_{e=(\ell',\ell)\in Edg} \mathsf{post}_e[X_{\ell'}] \rangle_\ell^{\nearrow}$$

Two problems arise in the practical resolution of such a system:

- Handling disjunctions of systems of linear inequalities; for instance there is no easy way for deciding if a union of polyhedra is included into another.
- The fixpoint computation may involve infinite iteration.

An approximate solution to these problems is provided by abstract interpretation techniques [CC77, CH78].

First, union of polyhedra is approximated by their *convex hull*, i.e., the least convex polyhedron containing the operands of the union. Let $\sqcup$ denote the convex hull operator:

$$P \sqcup P' = \{\lambda x + (1-\lambda)x' \mid x \in P, x' \in P', \lambda \in [0,1]\}$$

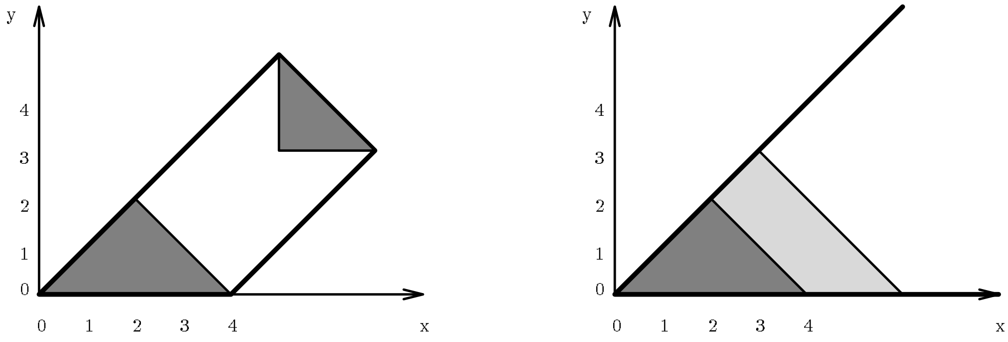
Fig. 8.a shows an example of convex hull. See [CH78, LeV92] for efficient algorithms to compute the convex hull. The system of equations becomes:

$$X_\ell \quad = \quad \langle I_\ell \sqcup \bigsqcup_{e=(\ell',\ell)\in Edg} \mathsf{post}_e[X_{\ell'}] \rangle_\ell^{\nearrow}$$

To enforce the convergence of iterations, we apply Cousot's "widening technique" [CC77, CH78]. The idea is to extrapolate the limit of a sequence of polyhedra, in such a way that an upper approximation of the limit be always reached in a finite number of iterations. We define a *widening operator*, noted ∇, on polyhedra, such that

- For each pair (P, P') of polyhedra, $P \sqcup P' \ \subseteq \ P \nabla P'$
- For each infinite increasing sequence $(P_0, P_1, \ldots, P_n, \ldots)$ of polyhedra, the sequence defined by $Q_0 = P_0$, $Q_{n+1} = Q_n \nabla P_{n+1}$ is not strictly increasing (i.e., remains constant after a finite number of terms).

A widening operator on polyhedra has been defined in [CH78, Hal93]. Intuitively, the system of linear constraints of $P\nabla P'$ is made of exactly those constraints of P which are also satisfied by P'. So it is built by removing constraints from P and since we cannot remove infinitely many constraints, the finiteness property follows. Fig. 8.b illustrates the widening operation. Now, this operator is used as follows: Choose, in each loop of the graph of the hybrid system, at least one location, and call them "*widening locations*" (So, removing these locations would cut each loop in the graph). Let $X_\ell^{(n)} = F(X^{(n-1)})$ be the n-th step computation at location ℓ; that is, $F(X^{(n-1)}) = \langle I_\ell \sqcup \bigsqcup_{e=(\ell',\ell)\in Edg} \mathsf{post}_e[X_{\ell'}^{(n-1)}]\rangle_\ell^{\nearrow}$. Instead, for each widening location ℓ and each step $n \geq 1$, compute $X_\ell^{(n)} = X_\ell^{(n-1)} \nabla F(X^{(n-1)})$. Then, the new iterative computation converges after a finite number of steps toward an upper approximation of the least solution of the original system.



$$\{0 \le y \le x \le 4 - y\} \sqcup \{x \ge 5 \wedge y \ge 3 \wedge x + y \le 10\} = \{0 \le y \le x \le y + 4 \wedge x + y \le 10\}$$

(a). Convex hull

$$\{0 \le y \le x \le 4 - y\} \nabla \{0 \le y \le x \le 6 - y\} = \{0 \le y \le x\}$$

(b). Widening

Figure 8: Approximation operators

Example: the leaking gas burner

With I defined by $\psi_I = (pc = 1 \wedge x = y = z = 0)$, we have $(I \mapsto^*) = X_1 \cup X_2$, with $X_i = \lim X_i^{(n)}, (i = 1, 2)$ and (choosing location 1 as the only widening location)

$$\begin{array}{rcl} X_1^{(n)} & = & X_1^{(n-1)} \nabla \langle (x = y = z = 0) \sqcup \mathsf{post}_{(2,1)}[X_2^{(n-1)}] \rangle_1^{\nearrow} \\ X_2^{(n)} & = & \langle \mathsf{post}_{(1,2)}[X_1^{(n)}] \rangle_2^{\nearrow} \end{array}$$

The successive iterations are as follows:

Step 1:

$$\begin{array}{rcl} X_1^{(1)} & = & x = y = z \ \wedge \ 0 \le x \le 1 \\ X_2^{(1)} & = & y = x + z \ \wedge \ 0 \le x \ \wedge \ 0 \le z \le 1 \end{array}$$

Step 2:

$$\begin{array}{rcl} X_1^{(2)} & = & 31z \le 30x + y \ \wedge \ x \le z \ \wedge \ 0 \le x \le 1 \\ X_2^{(2)} & = & x + z \le y \ \wedge \ 0 \le x \ \wedge \ 0 \le z \ \wedge \ x + 31z \le y + 30 \end{array}$$

and Step 3 shows the convergence:

$$X_1^{(3)} = X_1^{(2)} \quad , \quad X_2^{(3)} = X_2^{(2)}$$

So the final results are:

$$\begin{array}{rcl} X_1 & = & 0 \le x \le 1 \ \wedge \ x \le z \ \wedge \ 31z \le y + 30x \\ X_2 & = & 0 \le x \ \wedge \ 0 \le z \ \wedge \ x + z \le y \ \wedge \ x + 31z \le y + 30 \end{array}$$

These results are obtained in 0.2 sec. on SUN 4 Sparc Station. Notice that, in this case, the results are almost exact, and have been obtained automatically, without the induction step used in §4.1.

Other examples

Water-level monitor. Choosing location 0 as the only widening location, we get (in 0.4 sec.) the following results:

$$X_0 \quad = \quad 1 \le y \le 10$$

$$\begin{array}{rcl} X_1 & = & y = x + 10 \wedge 0 \leq x \leq 2 \\ X_2 & = & 2x + y = 16 \wedge 4 \leq 2x \leq 11 \\ X_3 & = & 2x + y = 5 \wedge 0 \leq x \leq 2 \end{array}$$

We can easily check that X_i implies $1 \leq y \leq 12$ for $0 \leq i \leq 3$. So, the water level is kept between 1 and 2 inches as required.

Fischer's mutual-exclusion protocol. In this example, we can consider delays a and b as symbolic constants, letting the analysis *discover* sufficient conditions for the algorithm to work. With two processes, the results (obtained in 0.3 sec.) show that the locations where the mutual exclusion is violated can only be reached when $a \geq b$ (resp., $11a \geq 10b$ when P_2's local clock runs 1.1 faster than P_1's).

4.4 Minimization

We extend the next relation $\mapsto$ to regions: for all regions R and R', we write $R \mapsto R'$ if some state $\sigma' \in R'$ is a successor of some state $\sigma \in R$, that is

$$R \mapsto R' \quad \text{iff} \quad \exists \sigma \in R, \sigma' \in R'.\, \sigma \mapsto \sigma'.$$

We write $\mapsto^*$ for the reflexive-transitive closure of $\mapsto$.

Let π be a partition of the state space Σ. A region $R \in \pi$ is *stable* if for all $R' \in \pi$,

$$R \mapsto R' \quad \text{implies} \quad \forall \sigma \in R.\, \{\sigma\} \mapsto R'$$

or, equivalently,

$$R \cap \mathsf{pre}[\langle R' \rangle \swarrow] \neq \emptyset \quad \text{implies} \quad R \subseteq \mathsf{pre}[\langle R' \rangle \swarrow].$$

The partition π is a *bisimulation* if every region $R \in \pi$ is stable. The partition π *respects* the region R_F if for every region $R \in \pi$, either $R \subseteq R_F$ or $R \cap R_F = \emptyset$.

If a partition π that respects the region R_F is a bisimulation, then it can be used to compute the initial region $(\mapsto^* R_F)$: for all regions $R \in \pi$, if $R \mapsto^* R_F$ then $R \subseteq (\mapsto^* R_F)$, otherwise $R \cap (\mapsto^* R_F) = \emptyset$. Thus, our objective is to construct the coarsest bisimulation that respects a given region R_F, provided there is a finite bisimulation that respects R_F.

If we are given, in addition to R_F, an initial region I that restricts our interest to the reachable region $(I \mapsto^*)$, then it is best to use an algorithm that performs a simultaneous reachability and minimization analysis of transition systems [BFH90, LY92].

The minimization procedure of [BFH90] is given below. Starting from the initial partition $\{R_F, \Sigma - R_F\}$ that respects R_F, the procedure selects a region R and checks if R is stable with respect to the current partition; if not, then R is split into smaller sets. Additional book-keeping is needed to record which regions are reachable from the initial region I. In the following procedure, π is the current partition, $\alpha \subseteq \pi$ contains the regions R that have been found reachable from I, and $\beta \subseteq \pi$ contains the regions R that have been found stable with respect to π. The function $\mathsf{split}[\pi](R)$ splits the region $R \in \pi$ into subsets that are "more" stable with respect to π:

$$\mathsf{split}[\pi](R) := \begin{cases} \{R', R - R'\} & \text{if} \quad \exists R'' \in \pi.\, R' = \mathsf{pre}[\langle R'' \rangle \swarrow] \cap R \,\wedge\, R' \subset R, \\ \{R\} & \text{otherwise.} \end{cases}$$

The minimization procedure returns YES iff $I \mapsto^* R_F$.

State-space minimization:

$\pi := \{R_F, \Sigma - R_F\}$; $\alpha := \{R \mid R \cap I \neq \emptyset\}$; $\beta := \emptyset$
while $\alpha \neq \beta$ **do**
  choose $R \in (\alpha - \beta)$
  let $\alpha' := \mathsf{split}[\pi](R)$
  if $\alpha' = \{R\}$ **then**
    $\beta := \beta \cup \{R\}$
    $\alpha := \alpha \cup \{R' \in \pi \mid R \mapsto R'\}$
  else
    $\alpha := \alpha - \{R\}$
    if $\exists R' \in \alpha'$ such that $R' \cap I \neq \emptyset$ **then** $\alpha := \alpha \cup \{R'\}$ **fi**
    $\beta := \beta - \{R' \in \pi \mid R' \mapsto R\}$
    $\pi := (\pi - \{R\}) \cup \alpha'$
  fi
od
return there is $R \in \alpha$ such that $R \subseteq R_F$.

If the regions R_F and I are linear, from Lemma 4.2 it follows that all regions that are constructed by the minimization procedure are linear. The minimization procedure terminates if the coarsest bisimulation has only a finite number of equivalence classes. An alternative minimization procedure is presented in [LY92], which can also be implemented using the primitives $\langle\rangle^{\swarrow}$ and pre.

Example: the water-level monitor

Let H be the hybrid automaton defined in Figure 2. We use the minimization procedure to prove that the formula $1 \leq y \leq 12$ is an invariant of H. It follows that the water-level monitor keeps the water level between 1 and 12 inches.

Let the set I of initial states be so defined by the linear formula

$$\psi_I \quad = \quad (pc = 0 \wedge x = 0 \wedge y = 1)$$

and let the set R_F of "bad" states be defined by the linear formula

$$\psi_f \quad = \quad (y < 1 \vee y > 12).$$

The initial partition is $\pi_1 = \{$

$$\begin{array}{llll}
\psi_{00} = & (pc = 0 \wedge 1 \leq y \leq 12), & \psi_{01} = & (pc = 0 \wedge (y < 1 \vee y > 12)), \\
\psi_{10} = & (pc = 1 \wedge 1 \leq y \leq 12), & \psi_{11} = & (pc = 1 \wedge (y < 1 \vee y > 12)), \\
\psi_{20} = & (pc = 2 \wedge 1 \leq y \leq 12), & \psi_{21} = & (pc = 2 \wedge (y < 1 \vee y > 12)), \\
\psi_{30} = & (pc = 3 \wedge 1 \leq y \leq 12), & \psi_{31} = & (pc = 3 \wedge (y < 1 \vee y > 12))\}.
\end{array}$$

The bad states are represented by ψ_{i1}, for $i \in \{0,1,2,3\}$. Since the set I of initial states is contained in ψ_{00}, that is $\psi_I \Rightarrow \psi_{00}$, let $\alpha = \{\psi_{00}\}$. Considering $\psi = \psi_{00} \in \alpha$, we find that $\mathsf{split}[\pi_1](\psi_{00}) = \{$

$$\begin{array}{lll}
\psi_{000} & = & (pc = 0 \wedge 1 \leq y \leq 10), \\
\psi_{001} & = & (pc = 0 \wedge 10 < y \leq 12)\}.
\end{array}$$

Therefore, $\pi_2 = \{\psi_{000}, \psi_{001}, \psi_{01}, \psi_{10}, \psi_{11}, \psi_{20}, \psi_{21}, \psi_{30}, \psi_{31}\}$. Now $\psi_I \Rightarrow \psi_{000}$, so take $\alpha = \{\psi_{000}\}$ and $\beta = \emptyset$. Considering $\psi = \psi_{000}$, we find that it is stable with respect to π_2. Thus $\alpha = \alpha \cup \{R' \in \pi \mid R \mapsto R'\} = \{\psi_{000}, \psi_{001}, \psi_{10}\}$ and $\beta = \{\psi_{000}\}$. Since $\psi = \psi_{001}$ is also stable in π_2 and is not reaching any new states not in α, α remains the same and $\beta = \{\psi_{000}, \psi_{001}\}$. However, considering $\psi = \psi_{10}$, we obtain $\mathsf{split}[\pi_2](\psi_{10}) = \{$

$$\begin{aligned}\psi_{100} &= (pc = 1 \wedge 0 \le x \le 2 \wedge 1 \le y \le 12),\\ \psi_{101} &= (pc = 1 \wedge x > 2 \wedge 1 \le y \le 12)\}.\end{aligned}$$

Now, ψ_{100} and ψ_{101} together with π_2, except for ψ_{10}, constitute π_3. The new β is obtained by removing $\{R' \in \pi \mid R' \mapsto R\} = \psi_{000}$ from the old β. The new α becomes $\{\psi_{000}, \psi_{001}\}$. Now $\psi = \psi_{000}$ is stable in π_3. Hence $\alpha = \{\psi_{000}, \psi_{001}, \psi_{100}\}$ and $\beta = \{\psi_{000}, \psi_{001}\}$. Since $\psi = \psi_{100}$ is stable in π_3, we have $\alpha = \{\psi_{000}, \psi_{001}, \psi_{100}, \psi_{101}, \psi_{20}\}$ and $\beta = \{\psi_{000}, \psi_{001}, \psi_{100}\}$. $\psi = \psi_{101}$ is also stable in π_3, so $\beta = \{\psi_{000}, \psi_{001}, \psi_{100}, \psi_{101}\}$ and α remains unchanged. Considering $\psi = \psi_{20}$, we obtain $\mathsf{split}[\pi_3](\psi_{20}) = \{$

$$\begin{aligned}\psi_{200} &= (pc = 2 \wedge 5 \le y \le 12),\\ \psi_{201} &= (pc = 2 \wedge 1 \le y < 5)\}.\end{aligned}$$

Now π_4 contains ψ_{200} and ψ_{201}, and thus ψ_{100} must be reconsidered. It is split into $\mathsf{split}[\pi_4](\psi_{100}) = \{$

$$\begin{aligned}\psi_{1000} &= (pc = 1 \wedge 0 \le x \le 2 \wedge 3 \le y \le 12 \wedge 3 \le y - x \le 12),\\ \psi_{1001} &= (pc = 1 \wedge 0 \le x \le 2 \wedge 1 \le y < 3 \wedge 1 \le y - x < 3)\}.\end{aligned}$$

Thus π_5 contains ψ_{1000} and ψ_{1001}. After finding that ψ_{000}, ψ_{1000} and ψ_{200} all are stable, we finally have $\alpha = \{\psi_{000}, \psi_{001}, \psi_{1000}, \psi_{200}, \psi_{201}, \psi_{30}\}$ and $\beta = \{\psi_{000}, \psi_{001}, \psi_{1000}, \psi_{200}\}$. So let $\psi = \psi_{201}$. It is stable, so $\beta = \beta \cup \{\psi_{200}\}$ and α does not change. Then $\psi = \psi_{30}$ is partitioned into $\{$

$$\begin{aligned}\psi_{300} &= (pc = 3 \wedge 0 \le x \le 2 \wedge 1 \le y \le 12),\\ \psi_{301} &= (pc = 3 \wedge x > 2 \wedge 1 \le y \le 12)\}.\end{aligned}$$

ψ_{200} has to be considered again. It is stable with respect to the current partition. Then $\psi = \psi_{300}$ is considered and $\mathsf{split}[\pi_6](\psi_{300}) = \{$

$$\begin{aligned}\psi_{3000} &= (pc = 3 \wedge 0 \le x \le 2 \wedge 5 \le y \le 12 \wedge 5 \le y + 2x \le 14),\\ \psi_{3001} &= (pc = 3 \wedge 0 \le x \le 2 \wedge 1 \le y < 5 \wedge 1 \le y + 2x < 5)\}.\end{aligned}$$

We must consider ψ_{200} again. It turns out that it is still stable. After considering $\psi = \psi_{3000}$, we have $\beta = \{\psi_{000}, \psi_{001}, \psi_{1000}, \psi_{200}, \psi_{201}, \psi_{3000}\}$ and $\alpha = \alpha \cup \{\psi_{000}\}$. Now the partition is

$$\pi_7 = \{\psi_{000}, \psi_{001}, \psi_{01}, \psi_{1000}, \psi_{1001}, \psi_{101}, \psi_{11}, \psi_{200}, \psi_{201}, \psi_{21}, \psi_{3000}, \psi_{3001}, \psi_{301}, \psi_{31}\}.$$

Since ψ_{000} is stable in π_7, we have $\alpha = \beta = \{\psi_{000}, \psi_{001}, \psi_{1000}, \psi_{200}, \psi_{201}, \psi_{3000}\}$. Notice that α contains no bad states from R_F, that is $\psi \wedge \psi_f = \mathit{false}$ for all $\psi \in \alpha$. Therefore, the invariant property has been verified.

4.5 Model Checking

Previously, we presented three semidecision procedures for the reachability problem of linear hybrid systems. Now we address the more general problem of whether the given linear hybrid system H satisfies a requirement that is expressed in the real-time temporal logic TCTL [ACD93].

Timed computation tree logic

Let C be a set of clocks not in Var; that is, $C \cap \mathit{Var} = \emptyset$. A *state predicate* is a linear formula over the set $\mathit{Var} \cup C$ of variables.

The formulas of TCTL are built from the state predicates by boolean connectives, the two temporal operators $\exists\mathcal{U}$ and $\forall\mathcal{U}$, and the reset quantifier for the clocks in C. The formulas of TCTL, then, are defined by the grammar

$$\phi ::= \psi \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid z.\, \phi \mid \phi_1 \exists\mathcal{U} \phi_2 \mid \phi_1 \forall\mathcal{U} \phi_2$$

where ψ is a state predicate and $z \in C$. The formula ϕ is *closed* if all occurrences of a clock $z \in C$ are within the scope of a reset quantifier z.

The closed formulas of TCTL are interpreted over the state space Σ of the linear hybrid system H. Intuitively, a state σ satisfies the TCTL-formula $\phi_1 \exists\mathcal{U} \phi_2$ if there exists a run of H from σ to a state σ' satisfying ϕ_2 such that $\phi_1 \vee \phi_2$ continuously holds along the run. Dually, the state σ satisfies the TCTL-formula $\phi_1 \forall\mathcal{U} \phi_2$ if every divergent run from σ leads to a state σ' satisfying ϕ_2 such that $\phi_1 \vee \phi_2$ continuously holds along from σ to σ'. Clocks can be used to express timing constraints. For instance, the TCTL-formula $z.\,(true \exists\mathcal{U}(\phi \wedge z \leq 5))$ asserts that there is a run on which ϕ is satisfied within 5 time units.

We use the standard abbreviations such as $\forall\Diamond\phi$ for $true\forall\mathcal{U}\phi$, $\exists\Diamond\phi$ for $true\exists\mathcal{U}\phi$, $\exists\Box\phi$ for $\neg\forall\Diamond\neg\phi$, and $\forall\Box\phi$ for $\neg\exists\Diamond\neg\phi$. We also put timing constraints as subscripts on the temporal operators. For example, the formula $z.\,\exists\Diamond(\phi \wedge z < 5)$ is abbreviated to $\exists\Diamond_{<5}\phi$.

Let $\rho = \sigma_0 \mapsto^{t_0} \sigma_1 \mapsto^{t_1} \ldots$ be a run of the linear hybrid system H, with $\sigma_i = (\ell_i, \nu_i)$ for all $i \geq 0$. A *position* π of ρ is a pair (i,t) consisting of a nonnegative integer i and a nonnegative real $t \leq t_i$. The positions of ρ are ordered lexicographically; that is, $(i,t) \leq (j,t')$ iff $i < j$, or $i = j$ and $t \leq t'$. For all positions $\pi = (i,t)$ of ρ,

- the state $\rho(\pi)$ at the position π of ρ is $(\ell_i, \varphi_{\ell_i}[\nu_i](t))$, and
- the time $\delta_\rho(\pi)$ at the position π of ρ is $t + \sum_{j<i} t_j$.

A *clock valuation* ξ is a function from C to $\mathsf{R}^{\geq 0}$. For any nonnegative real $t \in \mathsf{R}^{\geq 0}$, by $\xi + t$ we denote the clock valuation ξ' such that $\xi'(z) = \xi(z) + t$ for all clocks $z \in C$. For any clock $z \in C$, by $\xi[z := 0]$ we denote the valuation ξ' such that $\xi'(z) = 0$ and $\xi'(z') = \xi(z')$ for all clocks $z' \neq z$.

An *extended state* (σ, ξ) consists of a state $\sigma \in \Sigma$ and a clock valuation ξ. The extended state (σ, ξ) *satisfies* the TCTL-formula ϕ, denoted $(\sigma, \xi) \models \phi$, if

$(\sigma,\xi) \models \psi$ iff $(\sigma,\xi)(\psi)$;

$(\sigma,\xi) \models \neg\phi$ iff $(\sigma,\xi) \not\models \phi$;

$(\sigma,\xi) \models \phi_1 \vee \phi_2$ iff $(\sigma,\xi) \models \phi_1$ or $(\sigma,\xi) \models \phi_2$;

$(\sigma,\xi) \models z.\,\phi_1$ iff $(\sigma,\xi[z := 0]) \models \phi_1$;

$(\sigma,\xi) \models \phi_1 \exists\mathcal{U} \phi_2$ iff there is a run ρ of H with $\rho(0,0) = \sigma$, and a position π of ρ such that (1) $(\rho(\pi), \xi + \delta_\rho(\pi)) \models \phi_2$, and (2) for all positions $\pi' \leq \pi$ of ρ, $(\rho(\pi'), \xi + \delta_\rho(\pi')) \models \phi_1 \vee \phi_2$;

$(\sigma,\xi) \models \phi_1 \forall\mathcal{U} \phi_2$ iff for all divergent runs ρ of H with $\rho(0,0) = \sigma$ there is a position π of ρ such that (1) $(\rho(\pi), \xi + \delta_\rho(\pi)) \models \phi_2$, and (2) for all positions $\pi' \leq \pi$ of ρ, $(\rho(\pi'), \xi + \delta_\rho(\pi')) \models \phi_1 \vee \phi_2$.

Let ϕ be a closed formula of TCTL. A state $\sigma \in \Sigma$ satisfies ϕ, denoted $\sigma \models \phi$, if $(\sigma, \xi) \models \phi$ for all clock valuations ξ. The linear hybrid system H satisfies ϕ, denoted $H \models \phi$, if all states of H satisfy ϕ. The *characteristic set* $[\![\phi]\!] \subseteq \Sigma$ of ϕ is the set of states that satisfy ϕ.

The model-checking algorithm

Given a closed TCTL-formula ϕ, a model-checking algorithm computes the characteristic set $[\![\phi]\!]$. We present the symbolic model-checking algorithm for timed automata [HNSY94], which is a semidecision procedure for model checking TCTL-formulas over linear hybrid systems.

The procedure is based on fixpoint characterizations of the TCTL-modalities in terms of a binary next operator $\triangleright$. Given two regions $R, R' \subseteq \Sigma$, the region $R \triangleright R'$ is the set of states σ that have a successor $\sigma' \in R'$ such that all states between σ and σ' are contained in $R \cup R'$: $(\ell, \nu) \in (R \triangleright R')$ iff

$$\exists(\ell', \nu') \in R', t \in \mathsf{R}^{\geq 0}.\ ((\ell, \nu) \mapsto^t (\ell', \nu') \ \wedge\ \forall 0 \leq t' \leq t.\ (\ell, \nu + t') \in (R \cup R'));$$

that is, the $\triangleright$ operator is a “single-step until” operator.

To define the $\triangleright$ operator syntactically, we introduce some notation. For a linear formula ψ, we extend the tcp operator such that

$$\mathsf{tcp}_\ell[\psi][\nu](t) \quad \text{iff} \quad \forall 0 \leq t' \leq t.\ \varphi_\ell[\nu](t') \in (Inv(\ell) \cap [\![\psi]\!]);$$

that is, all valuations along the evolution by time t from the state (ℓ, ν) satisfy not only the invariant of location ℓ but also ψ. For a state $\sigma = (\ell, \nu) \in \Sigma$ we write $\varphi[\sigma]$ for the function $\varphi_\ell[\nu]$, and for a region $R = \bigcup_{\ell \in Loc}(\ell, R_\ell)$ we write

$$\mathsf{tcp}[R][\sigma](t) \quad \text{iff} \quad \mathsf{tcp}_\ell[R_\ell][\nu](t).$$

Now, for two regions $R, R' \subseteq \Sigma$, we define the region $R \triangleright R'$ as

$$\sigma \in (R \triangleright R') \quad \text{iff} \quad \exists t \in \mathsf{R}^{\geq 0}.\ (\varphi[\sigma](t) \in \mathsf{pre}[R'] \ \wedge\ \mathsf{tcp}[R \cup R'][\sigma](t)).$$

Lemma 4.3 *For all linear hybrid systems H, if R and R' are two linear regions of H, then so is $R \triangleright R'$.*

In [HNSY94] it is shown that for nonzeno timed automata, the meaning of both TCTL-modalities $\exists\mathcal{U}$ and $\forall\mathcal{U}$ can be computed iteratively as fixpoints, using the $\triangleright$ operator. While for multirate timed systems, the iterative fixpoint computation always terminates, this is no longer the case for linear hybrid systems in general. Lemma 4.3, however, ensures that all regions that are computed by the process are linear and each step of the procedure is, therefore, effective.

Here, we present the method for some important classes of TCTL-formulas:

- Let R and R' be the characteristic sets of the two TCTL-formulas ϕ and ϕ', respectively. The characteristic set of the formula $\phi \exists\mathcal{U} \phi'$ can be iteratively computed as $\bigcup_i R_i$ with
  - $R_0 = R'$, and
  - for all $i \geq 0$, $R_{i+1} = R_i \cup (R \triangleright R_i)$.
- To check if the TCTL-formula ϕ is an invariant of H, we check if the set of initial states is contained in the characteristic set of the formula $\forall\Box\phi$. This characteristic set can be iteratively computed as $\bigcap_i R_i$ with
  - $R_0 = [\![\phi]\!]$, and
  - for all $i \geq 0$, $R_{i+1} = R_i \cap \neg(true \triangleright \neg R_i)$.
- The real-time response property asserting that a given event occurs within a certain time bound is expressed in TCTL by a formula of the form $\forall\Diamond_{\leq c}\,\phi$, whose characteristic set can be iteratively computed as $\neg \bigcup_i R_i[z := 0]$ with
  - $R_0 = [\![z > c]\!]$, and
  - for all $i \geq 0$, $R_{i+1} = R_i \cup ((\neg R) \triangleright R_i)$,

  where $R = [\![\phi]\!]$ and $z \in C$.

Example: the temperature control system

The goal is to maintain the temperature of the coolant between lower and upper bounds θ_m and θ_M. If the temperature rises to its maximum θ_M and it cannot decrease because no rod is available, a complete shutdown is required.

Now, let $\Delta\theta = \theta_M - \theta_m$. Clearly, the time the coolant needs to increase its temperature from θ_m to θ_M is $\tau_r = \frac{\Delta\theta}{v_r}$, and the refrigeration times for rod 1 and rod 2 are $\tau_1 = \frac{\Delta\theta}{v_1}$ and $\tau_2 = \frac{\Delta\theta}{v_2}$, respectively.

The question is whether the system will ever reach the shutdown state. Clearly, if temperature rises at a rate slower than the time of recovery for the rods, i.e., $\tau_r \geq T$, shutdown is unreachable. Moreover, it can be seen that $2\tau_r + \tau_1 \geq T \ \wedge\ 2\tau_r + \tau_2 \geq T$ is a necessary and sufficient condition for never reaching the shutdown state (see Fig. 9).

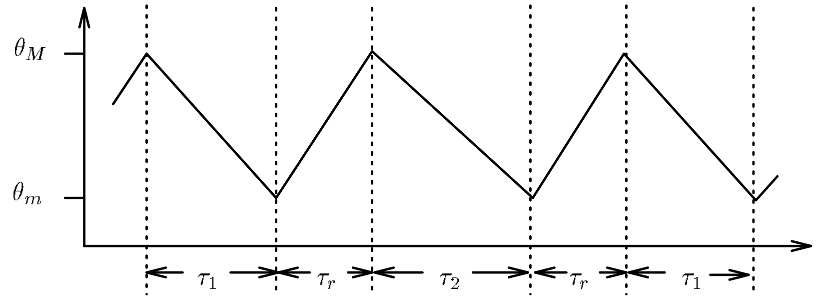


Figure 9: Refrigeration times

The property stating that state 3 (shutdown) is always unreachable corresponds to the following TCTL formula:

$$(pc = 0 \wedge \theta \leq \theta_M \wedge x_1 \geq T \wedge x_2 \geq T) \Rightarrow \forall\Box\neg(pc = 3)$$

or equivalently,

$$(pc = 0 \wedge \theta \leq \theta_M \wedge x_1 \geq T \wedge x_2 \geq T) \Rightarrow \neg\exists\Diamond(pc = 3)$$

- Let $v_r = 6$, $v_1 = 4$, $v_2 = 3$, $\theta_m = 3$, $\theta_M = 15$ and $T = 6$. In this case the condition $2\tau_r + \tau_1 \geq T \wedge 2\tau_r + \tau_2 \geq T$ holds. Using Kronos, we compute the characteristic set of $\exists\Diamond pc = 3$. The results obtained at each iteration are shown below, where each ψ_i has been computed according to the method described above:

$$\begin{array}{rcl}
\psi_0 & = & pc = 3 \\
\psi_1 & = & (pc = 0 \wedge \theta \leq 15 \wedge 6x_1 < \theta + 21 \wedge 6x_2 < \theta + 21) \vee pc = 3 \\
\psi_2 & = & (pc = 0 \wedge \theta \leq 15 \wedge 6x_1 < \theta + 21 \wedge 6x_2 < \theta + 21) \vee \\
 & & (pc = 1 \wedge 3 \leq \theta \leq 15 \wedge 4x_2 + \theta < 19) \vee \\
 & & (pc = 2 \wedge 3 \leq \theta \leq 15 \wedge 3x_1 + \theta < 15) \vee pc = 3 \\
\psi_3 & = & (pc = 0 \wedge \theta \leq 15 \wedge (6x_1 < \theta + 21 \wedge 6x_2 < \theta + 21 \vee 6x_2 + 3 < \theta)) \vee \\
 & & (pc = 1 \wedge 3 \leq \theta \leq 15 \wedge 4x_2 + \theta < 19) \vee \\
 & & (pc = 2 \wedge 3 \leq \theta \leq 15 \wedge 3x_1 + \theta < 15) \vee pc = 3 \\
\psi_4 & = & \psi_3
\end{array}$$

The state predicate $\neg \bigvee_{i=0}^{3} \psi_i[z := 0]$ representing the meaning of $\neg\exists\Diamond(pc = 3)$ is

$$\begin{array}{l}
pc = 0 \wedge \theta \leq 15 \wedge (\theta + 21 \leq 6x_1 \wedge \theta \leq 6x_2 + 3 \vee \theta + 21 \leq 6x_2) \vee \\
pc = 1 \wedge 3 \leq \theta \leq 15 \wedge 19 \leq 4x_2 + \theta \vee \\
pc = 2 \wedge 3 \leq \theta \leq 15 \wedge 15 \leq 3x_1 + \theta
\end{array}$$

Since the state predicate $pc = 0 \wedge \theta \leq 15 \wedge x_1 \geq 6 \wedge x_2 \geq 6$ characterizing the set of initial states implies the predicate above, the system satisfies the invariant as required.

- Suppose that we change the time of recovery to $T = 8$. Now, the condition $2\tau_r + \tau_1 \geq T \wedge 2\tau_r + \tau_2 \geq T$ is no longer satisfied. Again, we compute using Kronos the characteristic set of $\exists\Diamond pc = 3$. The results obtained at each iteration are the following:

$$\psi_0 \ = \ pc = 3$$

parameters						number of	running
θ_m	θ_M	v_r	v_1	v_2	T	iterations	times
3	15	6	4	3	6	4	0.033
3	15	6	4	3	8	4	0.033
10	190	45	30	18	20	6	0.083
250	1100	34	25	10	80	4	0.033

Table 1: Performances for the temperature control system

$$
\begin{aligned}
\psi_1 &= (pc = 0 \wedge \theta \leq 15 \wedge 6x_1 < \theta + 33 \wedge 6x_2 < \theta + 33) \vee pc = 3 \\
\psi_2 &= (pc = 0 \wedge \theta \leq 15 \wedge 6x_1 < \theta + 33 \wedge 6x_2 < \theta + 33) \vee \\
&\quad (pc = 1 \wedge 3 \leq \theta \leq 15 \wedge 4x_2 + \theta < 27) \vee \\
&\quad (pc = 2 \wedge 3 \leq \theta \leq 15 \wedge 3x_1 + \theta < 21) \vee pc = 3 \\
\psi_3 &= (pc = 0 \wedge \theta \leq 15 \wedge (6x_1 + 3 < \theta \vee 6x_2 < \theta + 3 \vee \\
&\quad (6x_1 < \theta + 33 \wedge 6x_2 < \theta + 33))) \vee \\
&\quad (pc = 1 \wedge 3 \leq \theta \leq 15 \wedge 4x_2 + \theta < 27) \vee \\
&\quad (pc = 2 \wedge 3 \leq \theta \leq 15 \wedge 3x_1 + \theta < 21) \vee pc = 3 \\
\psi_4 &= (pc = 0 \wedge \theta \leq 15 \wedge (6x_1 + 3 < \theta \vee 6x_2 < \theta + 3 \vee \\
&\quad (6x_1 < \theta + 33 \wedge 6x_2 < \theta + 33))) \vee \\
&\quad (pc = 1 \wedge 3 \leq \theta \leq 15 \wedge 4x_2 + \theta < 27) \vee \\
&\quad (pc = 2 \wedge 3 \leq \theta \leq 15) \vee pc = 3 \\
\psi_5 &= (pc = 0 \wedge \theta \leq 15 \wedge (\theta + 33 \leq 6x_2 \vee 6x_1 < \theta + 33 \vee 6x_2 < \theta + 3)) \vee \\
&\quad (pc = 1 \wedge 3 \leq \theta \leq 15 \wedge 4x_2 + \theta < 27) \vee (pc = 2 \wedge 3 \leq \theta \leq 15) \vee \\
&\quad pc = 3 \\
\psi_6 &= (pc = 0 \wedge \theta \leq 15 \wedge (\theta + 33 \leq 6x_2 \vee 6x_1 < \theta + 33 \vee 6x_2 < \theta + 3)) \vee \\
&\quad (pc = 1 \wedge 3 \leq \theta \leq 15) \vee (pc = 2 \wedge 3 \leq \theta \leq 15) \vee pc = 3 \\
\psi_7 &= (pc = 0 \wedge \theta \leq 15) \vee (pc = 1 \wedge 3 \leq \theta \leq 15) \vee (pc = 2 \wedge 3 \leq \theta \leq 15) \vee \\
&\quad pc = 3 \\
\psi_8 &= \psi_7
\end{aligned}
$$

The state predicate $\neg \bigvee_{i=0}^{7} \psi_i[z := 0]$ representing the meaning of $\neg\exists\Diamond(pc = 3)$ is

$$
\begin{aligned}
&pc = 0 \wedge \theta > 15 \vee \\
&pc = 1 \wedge (\theta < 3 \vee \theta > 15) \vee \\
&pc = 2 \wedge (\theta < 3 \vee \theta > 15)
\end{aligned}
$$

and since the state predicate $pc = 0 \wedge \theta \leq 15 \wedge x_1 \geq 6 \wedge x_2 \geq 6$ characterizing the set of initial states does not imply the predicate above we have that shutdown is reachable.

Table 1 shows the number of iterations and the running times (measured in seconds) obtained with Kronos on a SUN 4 Sparc Station for verifying the formula on the system for different values of the parameters. (Performance figures for HyTech can be found in [AHH93, HH94].)

Example: the billiards game

Consider the movement of the grey ball on the billard table. It is possible that the grey ball returns to the initial position with the initial direction. In this case the movement is periodic. A sufficient condition

parameters								formula	number of iterations	running times
l	h	v_x	v_y	x_g	y_g	x_w	y_w			
13	10	2	1	0	0	10	8	[*periodT*]	55	7.77
								[*touch*]	55	6.69
								[*touchT*]	55	8.17
4	2	5	1	0	0	1	1	[*periodT*]	24	1.97
								[*touch*]	24	1.58
								[*touchT*]	24	1.90
3	8	1	2	0	0	1	6	[*periodT*]	10	0.56
								[*touch*]	10	0.40
								[*touchT*]	10	0.48

Table 2: Performances for the billards game

for the periodicity is that l, h, v_x and v_y are integers. The period T is calculated as follows:

$$T = \mathsf{lcm}\left(\frac{2l}{v_x}\ ,\ \frac{2h}{v_y}\right)$$

Now, since the movement of the grey ball has period T, the first collision with the white ball, if it takes place, will occur before time T. We can express this property in TCTL as follows:[1]

$$[periodT] \qquad \neg(\neg(x = x_w \ \wedge\ y = y_w)\exists\mathcal{U}_{>T}(x = x_w \ \wedge\ y = y_w))$$

We would like to characterize also all the positions where the grey ball may be placed in order to be able to touch the white ball. This set of points is characterized by the formula:

$$[touch] \qquad \exists\Diamond(x = x_w \ \wedge\ y = y_w)$$

Since the movement of the grey ball has period T, this property can also be specified by the formula

$$[touchT] \qquad \exists\Diamond_{\leq T}\,(x = x_w \ \wedge\ y = y_w)$$

Table 2 shows the number of iterations and the running times (measured in seconds) obtained with Kronos on a SUN 4 Sparc Station for verifying the formulas [*periodT*], [*touch*] and [*touchT*] on the billiards game for different values of the parameters.

5 Conclusion

We showed that the verification problem for hybrid systems is intrinsically difficult even under severe restrictions. Then we identified linear hybrid systems as a class of hybrid systems for which algorithmic analysis techniques exist and perform reasonably well. For general hybrid systems our analysis methods can be applied modulo limitations that concern the effective computation of boolean operations, time closures, preconditions, and postconditions of state sets.

Future work is necessary to improve both the cost and the scope of our approach. The cost can be improved by designing efficient algorithms for representing, comparing, manipulating, and approximating state sets. The scope can be improved by identifying other classes of hybrid systems to which semidecision procedures based on reachability analysis apply. For example, our results have recently been extended to a more general model, where the rates of variables are not constant in each location, but vary arbitrarily between given constant lower and upper bounds [AHH93, OSY94]. In that case the state sets that are

[1] If T is not an integer, but is a rational $\frac{p}{q}$, we have to multiply l, h, x_g, y_g, x_w and y_w by q to make it an integer.

computed by the verification procedures are also definable by linear formulas. The more general case is interesting for the approximation of nonlinear hybrid systems.

We did not discuss any analysis techniques that cannot be formulated within the framework of reachability analysis. Most of these techniques are based on digitization methods that reduce verification problems for hybrid systems to verification problems for discrete systems, which are decidable [KPSY93, PV94].

References

[ACD93] R. Alur, C. Courcoubetis, and D.L. Dill. Model checking in dense real time. *Information and Computation*, 104(1):2–34, 1993.

[ACD+92] A. Alur, C. Courcoubetis, D. Dill, N. Halbwachs, and H. Wong-Toi. Minimization of timed transition systems. In W.R. Cleaveland, editor, *CONCUR 92: Theories of Concurrency*, Lecture Notes in Computer Science 630, pages 340–354. Springer-Verlag, 1992.

[ACHH93] R. Alur, C. Courcoubetis, T.A. Henzinger, and P.-H. Ho. Hybrid automata: an algorithmic approach to the specification and analysis of hybrid systems. In R.L. Grossman, A. Nerode, A.P. Ravn, and H. Rischel, editors, *Workshop on Theory of Hybrid Systems*, Lecture Notes in Computer Science 736, pages 209–229. Springer-Verlag, 1993.

[AD94] R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[AH94] R. Alur and T.A. Henzinger. Real-time system = discrete system + clock variables. In T. Rus, editor, *Proceedings of the First AMAST Workshop on Real-time Systems*, to appear. Available as Technical Report CSD-TR-94-1403, Cornell University, January 1994.

[AHH93] R. Alur, T.A. Henzinger, and P.-H. Ho. Automatic symbolic verification of embedded systems. In *Proceedings of the 14th Annual Real-time Systems Symposium*, pages 2–11. IEEE Computer Society Press, 1993.

[BFH90] A. Bouajjani, J.-C. Fernandez, and N. Halbwachs. Minimal model generation. In E.M. Clarke and R.P. Kurshan, editors, *Proceedings of the Second Annual Workshop on Computer-Aided Verification*, Lecture Notes in Computer Science 531, pages 197–203. Springer-Verlag, 1990.

[CC77] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th Annual Symposium on Principles of Programming Languages*. ACM Press, 1977.

[Čer92] K. Čerāns. Decidability of bisimulation equivalences for parallel timer processes. In G.v. Bochman and D.K. Probst, editors, *Proceedings of the 4th Annual Workshop on Computer-Aided Verification*, Lecture Notes in Computer Science 663, pages 269–300. Springer-Verlag, 1992.

[CH78] P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In *Proceedings of the 5th Annual Symposium on Principles of Programming Languages*, ACM Press, 1978.

[CHR91] Z. Chaochen, C. A. R. Hoare, and A. P. Ravn. A calculus of durations. *Information Processing Letters*, 40(5):269–276, 1991.

[Hal93] N. Halbwachs. Delay analysis in synchronous programs. In C. Courcoubetis, editor, *Proceedings of the 5th Annual Conference on Computer-Aided Verification*, Lecture Notes in Computer Science 697, pages 333–346. Springer-Verlag, 1993.

[HH94] T.A. Henzinger and P.-H. Ho. Model-checking strategies for hybrid systems. Presented at the Seventh International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, May 1994. Available as Technical Report CSD-TR-94-1437, Cornell University, July 1994.

[HNSY94] T.A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. *Information and Computation*, 111(2):193–244, 1994.

[HPR94] N. Halbwachs, Y.-E. Proy, and P. Raymond. Verification of linear hybrid systems by means of convex approximations. In *Proceedings of the International Symposium on Static Analysis*, Lecture Notes in Computer Science, to appear. Springer-Verlag, 1994.

[JLHM91] M. Jaffe, N. Leveson, M. Heimdahl, and B. Melhart. Software requirements analysis for real-time process-control systems. *IEEE Transactions on Software Engineering*, 17(3):241–258, 1991.

[KPSY93] Y. Kesten, A. Pnueli, J. Sifakis, and S. Yovine. Integration graphs: a class of decidable hybrid systems. In R.L. Grossman, A. Nerode, A.P. Ravn, and H. Rischel, editors, *Workshop on Theory of Hybrid Systems*, Lecture Notes in Computer Science 736, pages 179–208. Springer-Verlag, 1993.

[Lam87] L. Lamport. A fast mutual-exclusion algorithm. *ACM Transactions on Computer Systems*, 5(1):1–11, 1987.

[LeV92] H. LeVerge. A note on Chernikova's algorithm. Research Report 635, IRISA, February 1992.

[LY92] D. Lee and M. Yannakakis. Online minimization of transition systems. In *Proceedings of the 24th Annual Symposium on Theory of Computing*, pages 264–274. ACM Press, 1992.

[MMP92] O. Maler, Z. Manna, and A. Pnueli. From timed to hybrid systems. In J.W. de Bakker, K. Huizing, W.-P. de Roever, and G. Rozenberg, editors, *Proceedings of the REX Workshop "Real-Time: Theory in Practice"*, Lecture Notes in Computer Science 600, pages 447–484. Springer-Verlag, 1992.

[NOSY93] X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. An approach to the description and analysis of hybrid systems. In R.L. Grossman, A. Nerode, A.P. Ravn, and H. Rischel, editors, *Workshop on Theory of Hybrid Systems*, Lecture Notes in Computer Science 736, pages 149–178. Springer-Verlag, 1993.

[NSY92] X. Nicollin, J. Sifakis, and S. Yovine. Compiling real-time specifications into extended automata. *IEEE TSE Special Issue on Real-Time Systems*, 18(9):794–804, September 1992.

[NSY93] X. Nicollin, J. Sifakis, and S. Yovine. From ATP to timed graphs and hybrid systems. *Acta Informatica*, 30:181–202, 1993.

[OSY94] A. Olivero, J. Sifakis, and S. Yovine. Using abstractions for the verification of linear hybrid systems. In D. Dill, editor, *Proceedings of the 6th Annual Conference on Computer-Aided Verification*, Lecture Notes in Computer Science 818, pages 81–94. Springer-Verlag, 1994.

[PV94] A. Puri and P. Varaiya. Decidability of hybrid systems with rectangular differential inclusions. In D. Dill, editor, *Proceedings of the 6th Annual Conference on Computer-Aided Verification*, Lecture Notes in Computer Science 818, pages 95–104. Springer-Verlag, 1994.