Decidable Hybrid Systems[*]

Anuj Puri and Pravin Varaiya
Department of Electrical Engineering and Computer Science,
University of California, Berkeley, CA 94720

Abstract

Hybrid systems combine differential equations with discrete event systems. We consider the reachability problem: is there a trajectory from an initial state to a target state in the hybrid system. We show that for hybrid systems with decoupled differential inclusions, the reachability problem can be decided in a finite number of steps.

1 Introduction

Complex systems that are being designed today, incorporate both differential equations to model the continuous behavior, and discrete event systems to model instantaneous state changes in response to events. System which incorporate both dynamical and discrete event models are called hybrid systems.

In this paper we study properties of hybrid automata — a formalism for specifying hybrid systems. In particular, we are interested in algorithmic methods for solving problems relating to hybrid systems. Hybrid systems in which a problem can be solved algorithmically in a finite number of steps are called decidable hybrid systems. We consider the reachability problem: is there a trajectory from an initial state s_0 to a target state. We review the work in [2, 3, 5], and show that for initialized hybrid automata with constant decoupled differential inclusions, the reachability problem can be solved in a finite number of steps.

[*]Research supported by the California PATH program and by the National Science Foundation under grant ECS9417370.

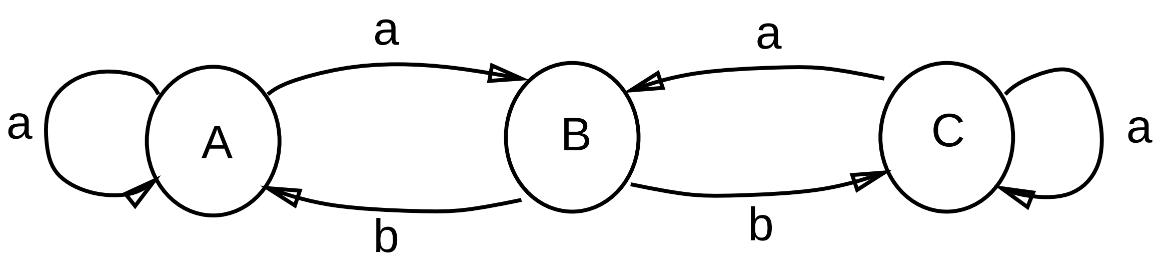

Figure 1: An Example Transition System

In Section 2, we discuss transition systems, and the relationships between them. A transition system specifies the states of the system, a set of generators, and the behavior of the system under the generators. In Section 3, we introduce hybrid automata and their transition systems. In Section 4, we discuss timed automata, a decidable class of hybrid automata. We show that initialized multirate automata are isomorphic to timed automata, and therefore also decidable. In Section 5, we extend the decidability results to rectangular automata with rectangular differential inclusions.

2 State Transition Systems

2.1 Transition Systems

Given a system with states S and a set of generators Σ, a *transition system* describes how the generators cause the state to evolve.

Example 2.1 *The example of figure 1 is a state machine with states $S = \{A, B, C\}$, and generators $\Sigma = \{a, b\}$. Generator a causes state A to move to A or B, and state C to move to C or B. Generator b causes state B to move to A or C.*

Definition 2.1 *A transition system is $\mathcal{A} = \langle S, \longrightarrow, \Sigma \rangle$ where*

- *S is a set of states.*
- *Σ is a set of generators.*

- $\longrightarrow \subset S \times \Sigma \times S$ *is the transition relation.*

We will write $(s, \sigma, s') \in \longrightarrow$ as $s \xrightarrow{\sigma} s'$. The transition relation for the system in Example 2.1 is $\longrightarrow = \{(A, a, A), (A, a, B), (B, b, A), (B, b, C), (C, a, C), (C, a, B)\}$.

Example 2.2 *The definition of a transition system is quite general. A differential equation* $\dot{x} = f(x)$ *with solution* $\phi(t)$ *defines the transition system* $\mathcal{A} = (\mathcal{R}^n, \longrightarrow, Time)$ *where* $Time = \{t | t \in \mathcal{R}^+\}$*, and the transition relation is* $x \xrightarrow{t} y$ *provided* $y = \phi(t)$ *and* $\phi(0) = x$.

Definition 2.2 *If* $\mathcal{A} = \langle S, \longrightarrow, \Sigma \rangle$ *is a transition system, then the reversed system is* $\mathcal{A}^{-1} = \langle S, \longrightarrow_R, \Sigma \rangle$ *where* $s' \xrightarrow{\sigma}_R s$ *iff* $s \xrightarrow{\sigma} s'$.

The reversed system moves backwards. For the example of figure 1, the reversed system is the same state machine with edges reversed. In Example 2.2, the reversed system $\mathcal{A}^{-1} = (\mathcal{R}^n, \longrightarrow_R, Time)$ is the transition system for the differential equation $\dot{x} = -f(x)$. The reversed system corresponds to the system obtained by reversing the time flow.

2.2 Relationship Between Transition Systems

We study relationships between transition systems. For example, two transition systems may be isomorphic. This is a very strong relationship. Weaker relationships are obtained by using the idea of simulation. A transition system simulates another when it can perform the same sequence of actions as another transition system. Using the idea of simulation, one can define bisimulation between two systems, where each system simulates the other. In our discussion of transition systems $\mathcal{A} = \langle X, \longrightarrow, \Sigma_X \rangle$ and $\mathcal{B} = \langle Y, \longrightarrow, \Sigma_Y \rangle$, we will assume there is a one-to-one correspondence between the generators Σ_X and Σ_Y. For generator $\sigma_x \in \Sigma_X$, we write σ_y for the corresponding generator in Σ_Y.

Definition 2.3 *Transition systems* $\mathcal{A} = \langle X, \longrightarrow, \Sigma_X \rangle$ *and* $\mathcal{B} = \langle Y, \longrightarrow, \Sigma_Y \rangle$ *are isomorphic provided there is a bijection* $h : X \longrightarrow Y$ *such that* $x \xrightarrow{\sigma_x} z$ *iff* $h(x) \xrightarrow{\sigma_y} h(z)$.

The transition system of the state machine of figure 1 is isomorphic to the transition system of a state machine obtained from figure 1 by relabeling of the states. A more intersting example is obtained by a change of variables in a differential equation.

Example 2.3 *For $x, y \in \mathcal{R}^2$ and differential equation $\dot{x} = (x_1, -x_2)$, consider the change of variables $y = (y_1, y_2) = h(x) = (x_1 - x_2, x_2)$. Then $\dot{y} = g(y) = (y_1 + 2y_2, y_2)$. The transition system $\mathcal{X} = \langle \mathcal{R}^2, \longrightarrow, Time \rangle$ of differential equation $\dot{x} = f(x)$ and $\mathcal{Y} = \langle \mathcal{R}^2, \longrightarrow, Time \rangle$ of differential equation $\dot{y} = g(y)$ are isomorphic.*

Definition 2.4 *Given transition systems $\mathcal{A} = \langle X, \longrightarrow, \Sigma_X \rangle$ and $\mathcal{B} = \langle Y, \longrightarrow, \Sigma_Y \rangle$, we say $\mathcal{B}$ simulates $\mathcal{A}$ with relation $R \subset X \times Y$ if $(x, y) \in R$ and $x \xrightarrow{\sigma_x} x'$ implies that there is $y' \in Y$ such that $y \xrightarrow{\sigma_y} y'$ and $(x', y') \in R$.*

The relation R in Definition 2.4 is called a simulation relation for the following reason. Whenever $(x, y) \in R$, for any sequence generated by $\mathcal{A}$ starting from state x, the corresponding sequence can be generated by $\mathcal{B}$ starting from state y. That is, $\mathcal{B}$ from state y can simulate $\mathcal{A}$ from state x when $(x, y) \in R$.

The following lemma states that the simulation property is transitive.

Lemma 2.1 *If $\mathcal{A}$ simulates $\mathcal{B}$ and $\mathcal{B}$ simulates $\mathcal{C}$, then $\mathcal{A}$ simulates $\mathcal{C}$.*

When $\mathcal{A}$ can simulate $\mathcal{B}$ and $\mathcal{B}$ can simulate $\mathcal{A}$, we say $\mathcal{A}$ and $\mathcal{B}$ are bisimilar.

Definition 2.5 *If $\mathcal{A} = \langle X, \longrightarrow, \Sigma_X \rangle$ and $\mathcal{B} = \langle Y, \longrightarrow, \Sigma_Y \rangle$ are transition systems, we say $R \subset X \times Y$ is a bisimulation provided $\mathcal{B}$ simulates $\mathcal{A}$ with R and $\mathcal{A}$ simulates $\mathcal{B}$ with $R^{-1} \subset Y \times X$ where $R^{-1} = \{(y, x) | (x, y) \in R\}$.*

A bisimulation between two transition systems indicates that the transition systems are equivalent in some sense. The relation that makes the transition systems bisimilar also indicates the states in the two systems that are equivalent. A stronger equivalence between two transition systems is obtained by requiring that $\mathcal{A}$ and $\mathcal{B}$ are bisimilar, and $\mathcal{A}^{-1}$ and $\mathcal{B}^{-1}$ are bisimilar. That is, the transition systems are bisimilar in reversed time as well.

Definition 2.6 *If $\mathcal{A} = \langle X, \longrightarrow, \Sigma_X \rangle$ and $\mathcal{B} = \langle Y, \longrightarrow, \Sigma_Y \rangle$ are transition systems, we say $R \subset X \times Y$ is a time-symmetric bisimulation provided $\mathcal{A}$ and $\mathcal{B}$ are bisimilar and $\mathcal{A}^{-1}$ and $\mathcal{B}^{-1}$ are bisimilar.*

Transition systems which are isomorphic are time-symmetric bisimilar. The next lemma states that the union of bisimulations is itself a bisimulation. This guarantees the existence of a largest bisimulation for two transition systems which are bisimilar.

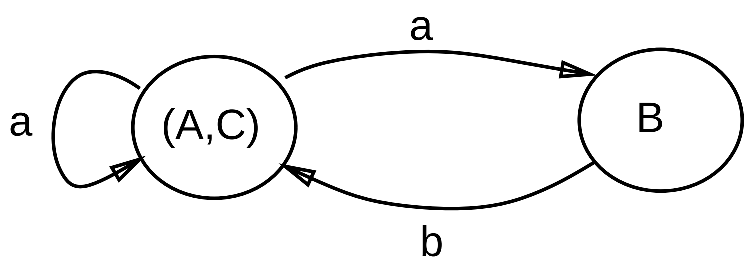

Figure 2: Quotient Transition System

Lemma 2.2 *If $\mathcal{A} = \langle X, \longrightarrow, \Sigma_X \rangle$ and $\mathcal{B} = \langle Y, \longrightarrow, \Sigma_Y \rangle$ are transition systems with bisimulation $R_i \subset X \times Y$, $i \in J$, then $R = \bigcup_{i \in J} R_i$ is also a bisimulation.*

2.3 Equivalence Relations

Given a transition system $\mathcal{A} = \langle X, \longrightarrow, \Sigma \rangle$, we are interested in identifying states of $\mathcal{A}$ that are "equivalent". We do this by considering the bisimulation of $\mathcal{A}$ with itself. The following lemma shows that a bisimulation on $\mathcal{A}$ leads to an equivalence relation.

Lemma 2.3 *If $\mathcal{A} = \langle X, \longrightarrow, \Sigma \rangle$ is a transition system and $R \subset X \times X$ is a bisimulation, then the reflexive, symmetric and transitive closure of R is also a bisimulation.*

A bisimulation on $\mathcal{A}$ creates an equivalence relation $\sim$ on the states X of $\mathcal{A}$, where all the states in the congruence class $[x]$ are bisimilar to each other. Since the identity relation $\{(x, x) | x \in X\}$ is a bisimulation, from Lemma 2.2, there is a largest bisimulation from $\mathcal{A}$ to itself. From Lemma 2.3, the largest bisimulation is an equivalence relation.

Definition 2.7 *For a transition system $\mathcal{A} = \langle X, \longrightarrow, \Sigma \rangle$, and a bisimulation $\sim \subset X \times X$ which is an equivalence relation, we define the quotient transition system $\mathcal{A}/\sim = \langle X/\sim, \longrightarrow, \Sigma \rangle$ where $X/\sim$ is the set of congruence classes, and $[x] \xrightarrow{\sigma} [y]$ provided $x \xrightarrow{\sigma} y$.*

Although we have stated lemma 2.2, lemma 2.3 and definition 2.7 for bisimulations, they hold for time-symmetric bisimulations as well. For the example in figure 1, $R = \{(A, A), (B, B), (C, C), (A, C), (C, A)\}$ is a time-symmetric bisimulation which is an equivalence relation. The quotient transition system is shown in figure 2.

2.4 Reach Set of Transition Systems

For the transition system $\mathcal{A}= (S, \longrightarrow, \Sigma)$, we define the relation $\Longrightarrow \subset S \times S$, where $\Longrightarrow = \{\xrightarrow{\sigma} | \sigma \in \Sigma\}^*$. That is, $\Longrightarrow$ is the transitive closure of the set $\{\xrightarrow{\sigma} | \sigma \in \Sigma\}$. Essentially $s \Longrightarrow w$ provided w can be reached from s by an application of a sequence of generators from Σ.

Definition 2.8 *For a transition system $\mathcal{A}= \langle S, \longrightarrow, \Sigma \rangle$, define $Reach_{\mathcal{A}}(S_0) = \{w | s \Longrightarrow w, s \in S_0\}$.*

The reach set $Reach_{\mathcal{A}}(S_0)$ is the set of states that can be reached from S_0 by a sequence of transitions. For a map $h : X \longrightarrow Y$ and $S \subset X$, define $h(S) = \{h(s) | s \in S\}$, and for a relation $R \subset X \times Y$ and $S \subset X$, define $R(S) = \{y | (s, y) \in R$ for some $s \in S\}$.

We next note that when two transition systems are equivalent, the reach set of one can be computed in terms of another.

Lemma 2.4 *If $\mathcal{A} = \langle X, \longrightarrow, \Sigma_X \rangle$ and $\mathcal{B} = \langle Y, \longrightarrow, \Sigma_Y \rangle$ are isomorphic with bijection $h : X \longrightarrow Y$ and $S_0 \subset X$, then $Reach_{\mathcal{B}}(h(S_0)) = h(Reach_{\mathcal{A}}(S_0))$.*

Lemma 2.5 *If $\mathcal{B} = \langle Y, \longrightarrow, \Sigma_Y \rangle$ simulates $\mathcal{A} = \langle X, \longrightarrow, \Sigma_X \rangle$ with relation $R \subset X \times Y$ and $Y_0 \subset Y$, then $Reach_{\mathcal{A}}(R^{-1}(Y_0)) \subset R^{-1}(Reach_{\mathcal{B}}(Y_0))$.*

Lemma 2.6 *If $\mathcal{A}^{-1} = \langle X, \longrightarrow_R, \Sigma_X \rangle$ simulates $\mathcal{B}^{-1} = \langle Y, \longrightarrow_R, \Sigma_Y \rangle$ with relation $R^{-1} \subset X \times Y$ and $Y_0 \subset Y$, then $R^{-1}(Reach_{\mathcal{B}}(Y_0)) \subset Reach_{\mathcal{A}}(R^{-1}(Y_0))$*

As a consequence of lemma 2.5 and lemma 2.6, we get the following theorem.

Theorem 2.1 *If $\mathcal{B}$ simulates $\mathcal{A}$ with $R \subset X \times Y$, $\mathcal{A}^{-1}$ simulates $\mathcal{B}^{-1}$ with R^{-1}, and $Y_0 \subset Y$, then $Reach_{\mathcal{A}}(R^{-1}(Y_0)) = R^{-1}(Reach_{\mathcal{B}}(Y_0))$.*

3 Hybrid Automata

A hybrid automaton [1, 4] models a hybrid system. It consists of control locations with edges between the control locations. Each location is labeled with a differential inclusion, and every edge is labeled with a guard, and a jump relation. The state of the hybrid automaton is the pair (l, x) where l is the control location, and $x \in \mathcal{R}^n$ is the continuous state. The hybrid automaton starts from some initial state (l_0, x_0). The trajectory evolves with the control location remaining constant and the continuous state x evolving with the differential inclusion at that location. When the continuous state satisfies the guard of an edge from location l to control location m, a jump can be made to location m. During the jump, the continuous state may get initialized to a new value y. The new state is the pair (m, y). The continuous state y now moves with the new differential inclusion, followed some time later by another jump, and so on.

3.1 Syntax

A guard is $g \subset \mathcal{R}^n$. An edge is enabled when the state $x \in g$. A jump relation is $j \subset \mathcal{R}^n \times \mathcal{R}^n$. During the jump, x is set to y provided $(x, y) \in j$. When j is the identity relation, the continuous state does not change. A differential inclusion is $\dot{x} \in f(x)$ where f is a set-valued map (i.e, $f(x) \subset \mathcal{R}^n$); in case $f(x)$ is a singleton, it is a differential equation. A solution to the differential inclusion with initial condition $x_0 \in \mathcal{R}^n$ is any differentiable function $\phi(t)$, where $\phi : \mathcal{R}^+ \to \mathcal{R}^n$ such that $\phi(0) = x_0$ and $\dot{\phi}(t) \in f(\phi(t))$. Associated with a differential inclusion $\dot{x} \in f(x)$ is the transition system $\langle \mathcal{R}^n, \longrightarrow_f, Time \rangle$ where $x \overset{t}{\longrightarrow}_f y$ provided for a solution ϕ of the differential inclusion, $\phi(0) = x$ and $\phi(t) = y$. The set of guards, jump relations and differential inclusions of interest are *Guard*, *Jump*, and *Inclusions* respectively.

A hybrid automaton is $H = (L, D, E)$ where

- L is a set of control locations.
- $D : L \longrightarrow Inclusions$ where $D(l)$ (also written D_l) is the differential inclusion at location l.
- $E \subset L \times Guard \times Jump \times L$ are the edges $-$ an edge $e = (l, g, j, m) \in E$ is an edge from location l to m with guard g and jump relation j.

Figure 3 is an example of a hybrid automaton with control locations A and B, and continuous state x. On the edge from B to A, the guard is $g = \{x | x \leq 2\}$, and x is nondeterministically assigned a value in the interval "$[2, 4]$".

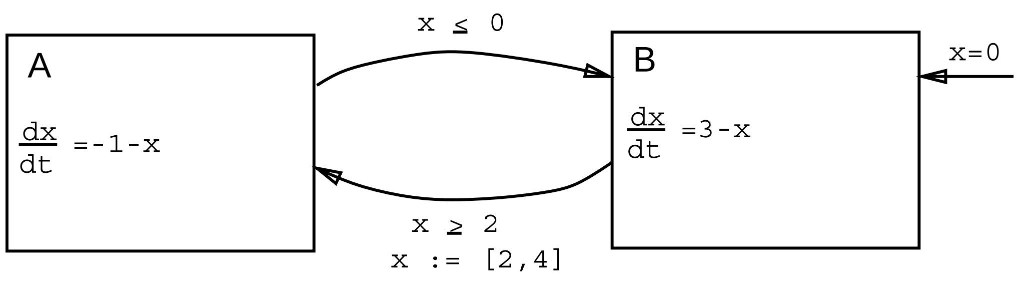

Figure 3: Hybrid Automata

3.2 Transition System of Hybrid Automata

The state space of the hybrid automaton is $Q_H \subset L \times \mathcal{R}^n$. We define the semantics of the hybrid automaton by defining its transition system. The state $(l, x) \in Q_H$ of the hybrid automaton evolves with either the control location remaining fixed and x evolving according to the inclusion at location l; or with a jump from one control location to another. The generators for the hybrid automaton $H = (L, D, E)$ are $\Sigma = Time \bigcup \{d\}$. The generator $t \in Time$ causes state (l, x) to evolve to state (l, y) in time t. The generator "d" causes a jump in the control location.

Definition 3.1 *For the hybrid automaton $H = (L, D, E)$, with state space $Q_H \subset L \times \mathcal{R}^n$, we define the transition system $H = \langle Q_H, \longrightarrow, Time \bigcup \{d\} \rangle$ where*

- *$(l, x) \xrightarrow{t} (l, y)$ provided $x \xrightarrow{t}_{D_l} y$.*
- *$(l, x) \xrightarrow{d} (m, y)$ provided $(l, g, j, m) \in E$, $x \in g$ and $(x, y) \in j$.*

We next define a τ-transition system for the hybrid automaton in which "*time*" does not play any role.

Definition 3.2 *Corresponding to the transition system $H = \langle Q_H, \longrightarrow, Time \bigcup \{d\} \rangle$ of the hybrid automaton H, define the τ-transition system $H_\tau = \langle Q_H, \longrightarrow, \{\tau\} \bigcup \{d\} \rangle$ where*

- *$(l, x) \xrightarrow{\tau} (l, y)$ in H_τ provided $(l, x) \xrightarrow{t} (l, y)$ for some t in H.*
- *$(l, x) \xrightarrow{d} (m, y)$ in H_τ provided $(l, x) \xrightarrow{d} (m, y)$ in H.*

$Reach_H(S_0)$ are all the states that can be reached in the hybrid automaton H starting from $S_0 \subset Q_H$. The next lemma states that the reach set of the transition systems H and H_τ are the same.

Lemma 3.1 *$Reach_H(S_0) = Reach_{H_\tau}(S_0)$.*

3.3 Classes of Hybrid Automata

We define some special classes of hybrid automata which are of interest.

Definition 3.3 *A n-dimensional rectangle is a set of the form $r = [l_1, u_1] \times \ldots \times [l_n, u_n]$ with $l_i, u_i \in \mathcal{Z}$. The ith component of r is $r_i = [l_i, u_i]$. The set of all n-dimensional rectangles is $Rect_n$.*

Definition 3.4 *A n-dimensional rectangular automaton $R = (L, D, E)$ is a hybrid automaton with $Inclusions = Rect_n$, $Guard = Rect_n$ and $Jump = \{j | j = j_1 \times \ldots \times j_n$ where $j_i = [l_i, u_i]$ or $j_i = id\}$. The relation $[l_i, u_i] = \{(x, y) | y \in [l_i, u_i]\}$ and id is the identity relation.*

Definition 3.5 *A n-dimensional multirate automaton is a n-dimensional rectangular automaton in which the inclusions consist of single points, i.e., $Inclusions = \{r : r \in \mathcal{Z}^n\}$.*

Figure 8 is an example of a rectangular automaton, and figure 6 shows a multirate automaton.

Definition 3.6 *A n-dimensional timed automaton is a n-dimensional multirate automaton in which the set Inclusions contains the single element "$\{1\}^n$", i.e., $\dot{x}_i = 1$ for each i at every control location.*

Since the differential equation is fixed at each location in the timed automaton, we denote the timed automaton by $T = (L, E)$.

In an *initialized* rectangular (multirate) automaton, the differential inclusion for the ith component changes only when it is initialized. That is, for locations l and m and edge $(l, g, j, m) \in E$, the inclusion for x_i can be different at l and m provided

x_i is initialized during the jump, i.e., j_i is not the identity relation. The multirate automaton in figure 6 is an initialized multirate automaton. For example, $\dot{x} = 2$ in location A and $\dot{x} = -3$ in location B, but x is initialized on the jump from A to B.

The ith component of guard g is $g_i = [l_i, u_i]$. For $k > 0$, define $\frac{g_i}{k} = [\frac{l_i}{k}, \frac{u_i}{k}]$, and for $k < 0$, define $\frac{g_i}{k} = [\frac{u_i}{k}, \frac{l_i}{k}]$. Similarly, the ith component of relation j is j_i. For relation $j_i = [l_i, u_i]$, $\frac{j_i}{k}$ is similarly defined, and for $j_i = id$, $\frac{j_i}{k} = id$.

3.4 Additional Notation

For an edge $e = (l, j, g, m)$ of a n-dimensional rectangular automaton with $j_i = [r_i, s_i]$ and $g_i = [a_i, b_i]$, we say the edge is labeled with the initialization $(x_i := j_i = [r_i, s_i])$ and the guard $(a_i \leq x_i \leq b_i)$. This information is summarized by writing a conjunctive expression $\bigwedge_{i=1}^{n}(a_i \leq x_i \leq b_i)(x_i := j_i)$. We also permit a disjunction of conjunctive expressions of the form $\bigvee_{k=1}^{m} \bigwedge_{i=1}^{n}(a_{ik} \leq x_i \leq b_{ik})(x_i := j_{ik})$. Such an expression would be represented by m edges in our definition of a rectangular automaton.

We also permit expressions of the form $((x_1 \leq 5) \longrightarrow (x_1 := 5))$, which translates to $(x_1 > 5) \vee ((x_1 \leq 5) \wedge (x_1 := 5))$, and would be represented with two edges. In general, any boolean expression can be translated into disjunction of conjunctive expressions, and represented with multiple edges.

4 Timed Automata and Initialized Multirate Automata

4.1 Timed Automata

A n-dimensional timed automaton has n clocks, $x_i, i = 1, \ldots, n$, with $\dot{x}_i = 1$ at every control location. The guards on the edges are rectangles. The jump relation either leaves the value of the clock unchanged, or sets the clock nondeterministically to a value in an interval. Figure 4 is an example of a timed automaton.

We review the work in [2] in which it was shown that timed automata have a time-symmetric bisimulation with a finite number of congruence classes.

Consider a timed automaton $T = (L, E)$ with state space Q_T. Suppose M_i is the largest integer with which x_i is compared in a guard, or assigned in a jump relation.

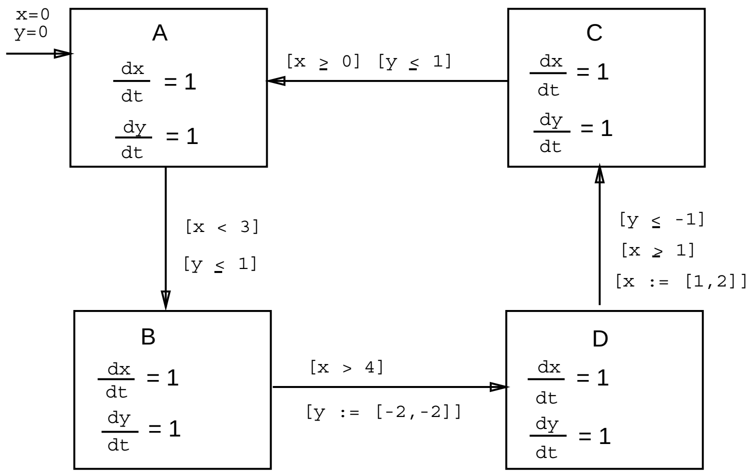

Figure 4: Timed Automaton

We will define an equivalence relation $\sim$ on $\mathcal{R}^n$, where two states will be related provided the ordering of the fractional parts of the components of the two states is the same; and the integer parts of the components of two states match, or are greater than the largest integer with which they are compared. For $z \in \mathcal{R}$, we write $\lfloor z \rfloor$ for its integer part, and $\langle z \rangle$ for its fractional part.

Definition 4.1 *We define a relation $\sim \subset \mathcal{R}^n \times \mathcal{R}^n$, where $x \sim y$ provided*

- $\langle x_i \rangle < \langle x_j \rangle$ *iff* $\langle y_i \rangle < \langle y_j \rangle$
- $\langle x_i \rangle = \langle x_j \rangle$ *iff* $\langle y_i \rangle = \langle y_j \rangle$
- $(\lfloor x_i \rfloor = \lfloor y_i \rfloor \leq M_i)$ *or* $((\lfloor x_i \rfloor > M_i)$ *and* $(\lfloor y_i \rfloor > M_i))$

It can be checked that $\sim$ is an equivalence relation.

We next describe the congruence classes of the equivalence relation $\sim$. For a vector $x \in \mathcal{R}^n$, we define $\langle x \rangle$ to be the ordering of the fractional parts of its components. For example, for $x = [0.3\ 4.7\ 3.2\ 8.3]$, $\langle x \rangle = (0 < \langle x_3 \rangle < \langle x_1 \rangle = \langle x_4 \rangle < \langle x_2 \rangle)$. Similarly, $\lfloor x \rfloor$ is a vector of integer parts of the components, or an indication that the integer part is greater than M_i. For $x = [0.3\ 4.7\ 3.2\ 8.3]$, and $M_1 = M_2 = M_3 = M_4 = 5$, $\lfloor x \rfloor = [0\ 4\ 3\ >]$.

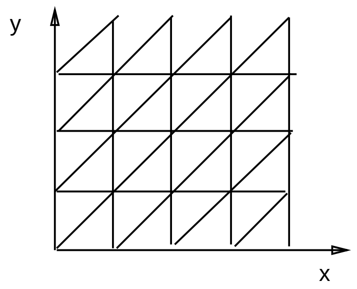

Figure 5: Congruence Classes of the Bisimulation for Timed System

Lemma 4.1 *For the equivalence relation $\sim$, the congruence class $[x] = \{y | \langle x \rangle = \langle y \rangle$ and $\lfloor x \rfloor = \lfloor y \rfloor\}$.*

Since the integers M_i are bounded a priori, the equivalence relation $\sim$ has only a finite number of congruence classes. Figure 5 shows the congruence classes for $\mathcal{R}^2$. We extend the relation $\sim$ to $Q_T \subset L \times \mathcal{R}^n$ by defining $(l, x) \sim (l, y)$ provided $x \sim y$. The relation $\sim$ is an equivalence relation on Q_T and the congruence class $[(l, x)] = \{l\} \times [x]$. Since L is finite, Q_T has a finite number of congruence classes.

Theorem 4.1 *The equivalence relation $\sim$ is a time-symmetric bisimulation for the τ-transition system $T_\tau = (Q_T, \longrightarrow, \{\tau\} \bigcup \{d\})$.*

Therefore the quotient transition system $T_\tau / \sim = (Q_T / \sim, \longrightarrow, \{\tau\} \bigcup \{d\})$ has only a finite number of states. As a consequence, we get the following result.

Lemma 4.2 *For a timed automaton $T = (L, E)$, $Reach_T(S_0)$ can be computed in a finite number of steps.*

Although, we have only considered timed automata in which the coefficients in the guards and jump relation are integers, similar results hold for timed automata with rational coefficients.

4.2 Initialized Multirate Automata

In a multirate automaton, different components of state x move at different rates. In an initialized multirate automaton, the rate at which x_i moves can change when a jump is made into another control location and x_i is initialized during the jump. We show how to construct a timed automaton such that the transition systems of the timed automaton and the initialized multirate automaton are isomorphic.

Corresponding to the initialized n-dimensional multirate automaton $M = (L, D, E)$, define the corresponding n-dimensional timed automaton $T = (L, E_T)$. The edge $(l, g, j, m) \in E$ is replaced with the edge $(l, g^T, j^T, m) \in E_T$ with $g_i^T = \frac{g_i}{v_i}$ and $j_i^T = \frac{j_i}{w_i}$ where $D_l = \{v\}$ and $D_m = \{w\}$.

Figure 6 shows an initialized multirate automaton, and figure 7 shows the corresponding timed automaton. Consider the edge from location A to location B. The guard $(x \leq 5)$ on the edge in the multirate automaton is replaced with the guard $(x \leq \frac{5}{2})$ in the timed automaton since $\dot{x} = 2$ in location A of the multirate automaton. Similarly the initialization $x := [4, 4]$ in the multirate automaton is replaced with the initialization $x := [\frac{4}{-3}, \frac{4}{-3}]$ in the timed automaton since $\dot{x} = -3$ in location B of the multirate automaton.

Define the map $h : L \times \mathcal{R}^n \longrightarrow L \times \mathcal{R}^n$ by $h(l, y) = (l, x)$ where $x = (\frac{y_1}{v_1}, \ldots, \frac{y_n}{v_n})$ and $D_l = v$. The state space of the timed automaton T is $Q_T = h(Q_M)$. The next theorem states that the transition systems of the multirate automaton and the timed automaton are isomorphic.

Theorem 4.2 *The transition system $M = (Q_M, \longrightarrow, Time \bigcup \{\tau\})$ of the initialized multirate automaton, and $T = (Q_T, \longrightarrow, Time \bigcup \{\tau\})$ of the timed automaton are isomorphic with bijection $h : Q_M \longrightarrow Q_T$.*

Lemma 4.3 *$h(Reach_M(X_0)) = Reach_T(h(X_0))$.*

Since the reach set of a timed automaton can be computed in a finite number of steps, the reach set of an initialized multirate automaton can also be computed in a finite number of steps.

5 Initialized Rectangular Automata

In this section, we discuss initialized rectangular automata. Decidability of initialized rectangular automata was shown in [5]. We follow the proof given in [3].

In a rectangular automaton, the inclusion for the ith component x_i is $\dot{x}_i \in [l_i, u_i]$. Figure 8 shows an initialized rectangular automaton. In this section we present a

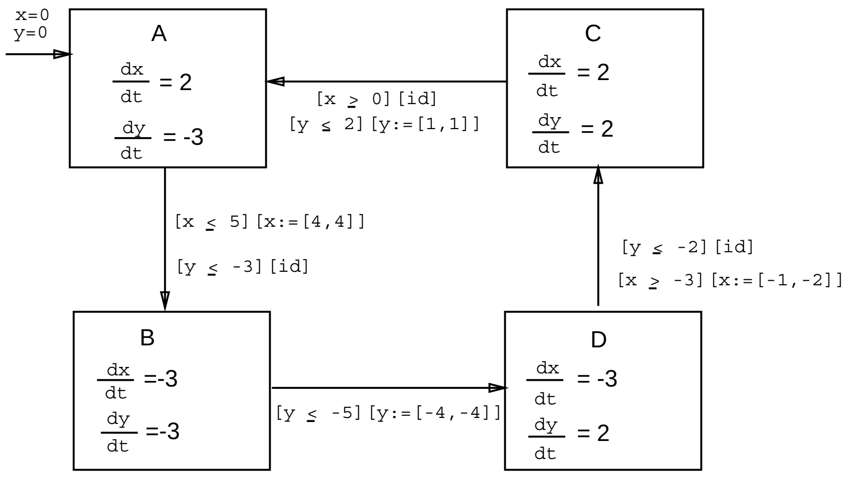

Figure 6: Multirate Automaton

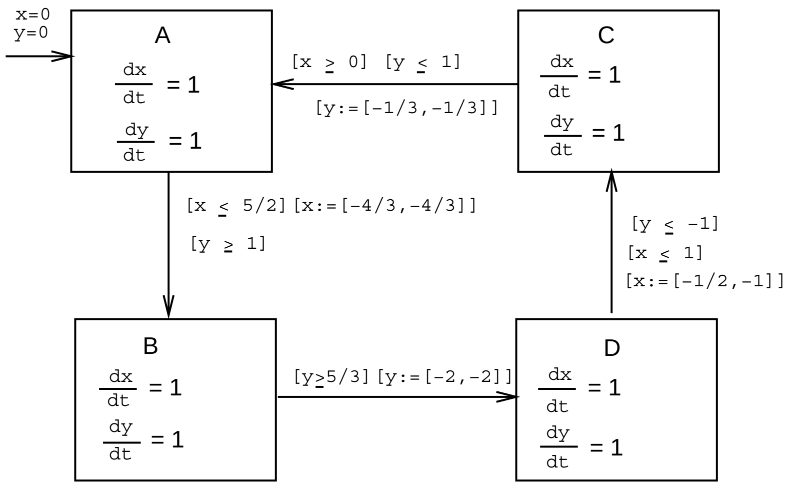

Figure 7: Translation of Multirate Automaton to Timed Automaton

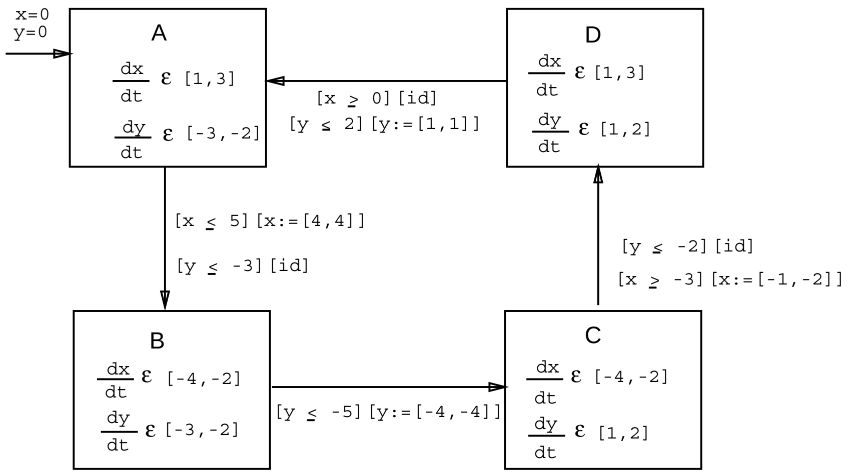

Figure 8: Initialized Rectangular Automaton

method to translate an initialized rectangular automaton into an initialized multirate automaton. Given an n-dimensional rectangular automaton R, we define a simulation relation S and a $2n$-dimensional multirate automaton M such that M simulates R, and R^{-1} simulates M^{-1}.

We do this by replacing a variable $\dot{x} \in [l, u]$ in the rectangular automaton with two variables $\dot{x}_l = l$ and $\dot{x}_u = u$ in the multirate automaton. The variable x defines an envelope in the rectangular automaton whose lower and upper boundaries are tracked by x_l and x_u in the multirate automaton (figure 9). When the test $(x \geq a)$ is made in the rectangular automaton, we make the test $(x_u \geq a)$ in the multirate automaton. After the test, the boundary of the envelope gets redefined (figure 9). This is done by checking whether $(x_l \leq a)$, and initializing x_l to a when this is the case. Hence, the lower and the upper boundary of the envelope are tracked again by x_l and x_u after the test. Similarly, when the test $(x \leq a)$ is made in the rectangular automaton, we test whether $(x_l \leq a)$ in the multirate automaton. To update the boundary, we initialize x_u to a when $(x_u \geq a)$.

More formally, corresponding to an n-dimensional rectangular automaton $R = (L, D_R, E_R)$, we define the $2n$-dimensional multirate automaton $M = (L, D_M, E_M)$. The variable y_{2i-1} in the multirate automaton tracks the lower boundary of the envelope created by x_i in rectangular automaton, and the variable y_{2i} tracks the upper boundary of the envelope. At location l, $\dot{x}_i \in [l_i, u_i]$ in the rectangular automaton is replaced with $\dot{y}_{2i-1} = l_i$ and $\dot{y}_{2i} = u_i$ in the multirate automaton. A guard $(x_i \geq a)$ in

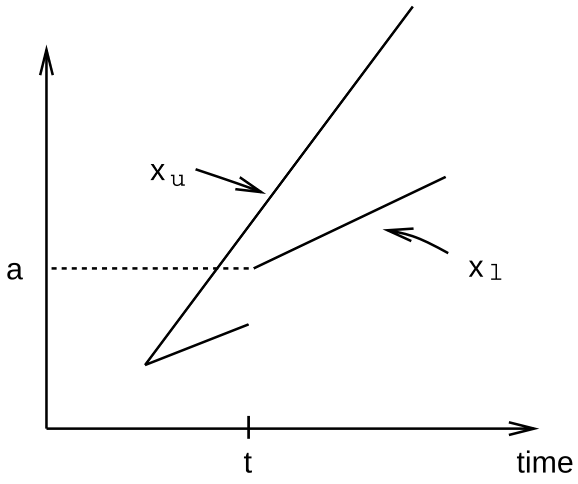

Figure 9: Envelope of differential inclusion $\dot{x} \in [l, u]$

the rectangular automaton is replaced with $(y_{2i} \geq a) \wedge ((y_{2i-1} \leq a) \longrightarrow (y_{2i-1} := a))$ on the corresponding edge in the multirate automaton. Similarly the guard $(x_i \leq a)$ in the rectangular automaton is replaced with $(y_{2i-1} \leq a) \wedge ((y_{2i} \geq a) \longrightarrow (y_{2i} := a))$ in the multirate automaton. The initialization $(x_i := [r, s])$ in the rectangular automaton is replaced with $(y_{2i-1} := r) \wedge (y_{2i} := s)$ on the corresponding edge in the multirate automaton. Figure 10 is the initialized multirate automaton obtained by translating the initialized rectangular automaton of figure 8.

Definition 5.1 *For a rectangular automaton R with state space Q_R, and the corresponding multirate automaton M with state space Q_M, define the relation $S \subset Q_M \times Q_R$ where $S = \{((l, y), (l, x)) | y_{2i-1} \leq x_i \leq y_{2i}\}$.*

The relationship between the states of the initialized rectangular automaton, and the initialized multirate automaton obtained by translation from it, is made with the relation S. The multirate automaton tracks the rectangular automaton. When the multirate automaton reaches state (l, y), the rectangular automaton can reach any state in the set $S(\{(l, y)\}) = \{l\} \times [y_1, y_2] \times \cdots \times [y_{2n-1}, y_{2n}]$.

Theorem 5.1 *If $R = (Q, \longrightarrow, Time \bigcup \{\tau\})$ is the transition system of an initialized rectangular automaton, and $M = (Q_M, \longrightarrow, Time \bigcup \{\tau\})$ is the transition system of the corresponding multirate automaton, then*

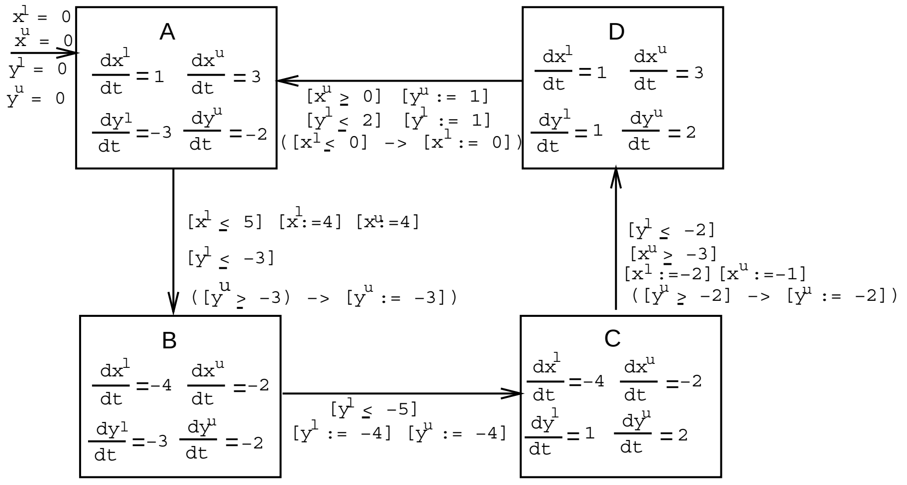

Figure 10: Translation of Rectangular Automaton to Multirate Automaton

- *M simulates R with relation S^{-1}, and*
- *R^{-1} simulates M^{-1} with relation S.*

From theorem 2.1, we get the following result.

Theorem 5.2 *$Reach_R(S(Y_0)) = S(Reach_M(Y_0))$.*

Since the reach set of an initialized multirate automaton can be computed in a finite number of steps, we can compute the reach set of an initialized rectangular automaton in a finite number of steps.

References

[1] R.Alur, C.Courcoubetis, T.A. Henzinger and P.-H. Ho, Hybrid automata: an algorithmic approach to the specification and analysis of hybrid systems,*Hybrid Systems*, LNCS 736, Springer-Verlag 1993.

[2] R. Alur and D. Dill, Automata for modeling real-time systems, *Proc. 17th ICALP*, Lecture Notes in Computer Science 443, Springer-Verlag, 1990.

[3] P.Kopke, T. Henzinger, A. Puri and P. Varaiya, What's Decidable About Hybrid Automata, *STOCS 1995*.

[4] X.Nicollin, A.Olivero, J. Sifakis and S. Yovine, An Approach to the Description and Analysis of Hybrid Systems, *Hybrid Systems*, LNCS 736, Springer-Verlag 1993.

[5] A. Puri and P. Varaiya, Decidability of hybrid systems with rectangular differential inclusions, *Proc. 6th Workshop on Computer-Aided Verification*, LNCS 818, Springer-Verlag, 1994.