DISCRETE-TIME CONTROL FOR RECTANGULAR HYBRID AUTOMATA

by

Thomas A. Henzinger and Peter W. Kopke

Memorandum No. UCB/ERL M97/29

15 April 1997

DISCRETE-TIME CONTROL FOR RECTANGULAR HYBRID AUTOMATA

by

Thomas A. Henzinger and Peter W. Kopke

Memorandum No. UCB/ERL M97/29

15 April 1997

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Discrete-Time Control for Rectangular Hybrid Automata*,**

Thomas A. Henzinger[1] Peter W. Kopke[2]

[1] University of California, Berkeley, CA. Email: tah@eecs.berkeley.edu
[2] William H. Kopke, Jr. Inc., Lake Success, NY. Email: 75467.2651@compuserve.com

Abstract. Rectangular hybrid automata model digital control programs of analog plant environments. We study rectangular hybrid automata where the plant state evolves continuously in real-numbered time, and the controller samples the plant state and changes the control state discretely, only at the integer points in time. We prove that rectangular hybrid automata have finite bisimilarity quotients when all control transitions happen at integer times, even if the constraints on the derivatives of the variables vary between control states. This is sharply in contrast with the conventional model where control transitions may happen at any real time, and already the reachability problem is undecidable. Based on the finite bisimilarity quotients, we give an exponential algorithm for the symbolic sampling-controller synthesis of rectangular automata. We show our algorithm to be optimal by proving the problem to be EXPTIME-hard. We also show that rectangular automata form a maximal class of systems for which the sampling-controller synthesis problem can be solved algorithmically.

1 Introduction

Hybrid systems are dynamical systems with both discrete and continuous components. A paradigmatic example of a hybrid system is a digital control program for an analog plant environment, like a furnace or an airplane: the controller state moves discretely between control modes, and in each control mode, the plant state evolves continuously according to physical laws. A natural mathematical model for hybrid systems is the *hybrid automaton*, which represents discrete components using finite-state machines and continuous components using real-numbered variables [ACH+95]. A particularly important subclass of hybrid automata are the *rectangular automata*, where in each control mode v, the given n variables follow a nondeterministic differential equation of the form $\frac{d\mathbf{x}}{dt} \in B(v)$, for an n-dimensional rectangle $B(v) \subset \mathbb{R}^n$ [HKPV95]. Rectangular automata are useful as (1) they can be made to approximate, arbitrarily closely, complex continuous behavior using lower and upper bounds on derivatives [HH95], and (2) they can be analyzed automatically using (semi)algorithms based on symbolic execution, such as those implemented in HYTECH [HHW97].

For systems that can be executed symbolically, verification and control yield to a (semi)algorithmic approach even if the state space is infinite [Hen96]. For such systems, a temporal formula can be verified automatically and a controller can be synthesized automatically by computing, using iterative approximation, a fixpoint of an operator on state sets [BCM+92, MPS95]. The fixpoint computation is guaranteed to terminate in the presence of a suitable finite quotient space. For example, symbolically-executable systems with finite bisimilarity quotients allow symbolic LTL and CTL model checking, and symbolic safety controller synthesis. While rectangular automata can be executed symbolically, they do not necessarily have finite bisimilarity quotients, and simple reachability questions are undecidable [HKPV95]. A noted subclass of rectangular automata

* To appear in the *Proceedings of the 24th International Colloquium on Automata, Languages, and Programming* (ICALP), *Lecture Notes in Computer Science*, Springer-Verlag, 1997.

** This research was supported in part by the ONR YIP award N00014-95-1-0520, by the NSF CAREER award CCR-9501708, by the NSF grant CCR-9504469, by the AFOSR contract F49620-93-1-0056, by the ARO MURI contract DAAH-04-96-1-0341, by the ARO contract DAAL03-91-C-0027 through the MSI at Cornell University, by the ARPA grant NAG2-892, and by the SRC contract 95-DC-324.036.

with finite bisimilarity quotients are *timed automata*, where all variables are clocks with derivative 1 [AD94]. As a consequence, the symbolic model checking and controller synthesis problems have been solved for timed automata [HNSY94, MPS95].

While previous results on timed and hybrid automata allow edge transitions (i.e., control switches) to occur at any real-numbered points in time, this is not necessarily a natural assumption for controller synthesis, as it permits controllers that, in a single time unit, can interact with the plant an unbounded number of times (even infinitely often, if no special care is taken [AH97]). By contrast, we study the control problem under the assumption that while the plant evolves continuously, the controller samples the plant state discretely, at the integer points in time only.[3] This leads to the following formulation of the *sampling-controller synthesis problem* for rectangular automata: given a continuous-time rectangular automaton, is there a discrete-time controller that samples the automaton state at integer times and switches the control mode accordingly so that the resulting closed-loop system satisfies a given invariant?

To solve this problem, we study the *discrete-time transition systems* of timed and rectangular automata, where all time transitions have unit duration. It should be noticed that all variables still evolve continuously, in real-numbered time; only edge transitions are restricted to discrete time. We prove that unlike in the case of dense time, the discrete-time transition system of every rectangular automaton has a finite bisimilarity quotient.[4] As a corollary, we conclude that the standard approaches to symbolic model checking and controller synthesis are guaranteed to terminate when all control switches must occur at integer times. The running times of the verification and control algorithms depend on the number of bisimilarity equivalence classes, which, while exponential in the description of the automaton, is less by a multiplicative exponential factor than the number of region equivalence classes used for the dense-time verification and control of timed automata. Thus, the often more realistic sampling-controller synthesis problem can be solved for a wider class of hybrid systems than dense-time control (rectangular vs. timed), at a smaller cost.

We prove that our sampling-control algorithm is optimal, by giving lower bounds on the control problem for timed and hybrid systems: we show that the safety control decision problem (does there exist a controller that maintains an invariant?) is complete for EXPTIME already in the restricted case of discrete-time timed automata. We also identify the boundary of sampling controllability by proving that several generalizations of rectangular automata lead to an undecidable reachability problem, even in discrete time. The undecidability of dense-time reachability for rectangular automata has led [PV94] to consider the restriction that the flow rectangle $B(v)$ must be the same for each control mode v. For the resulting class of *initialized* rectangular automata, reachability is decidable [HKPV95]. Our work can be viewed as pointing out an orthogonal restriction of rectangularity, namely, that the flow rectangle may change only at integer points in time. Unlike initialization, our restriction guarantees not only a finite language equivalence quotient but a finite bisimilarity quotient on the infinite state space of a rectangular automaton.

2 Definitions and Previous Results

2.1 Labeled Transition Systems

Definition 2.1 [Transition system] A *transition system* $S = (Q, \Sigma, \rightarrow, Q_I, \Pi, \models)$ consists of a set Q of *states*, a finite set Σ of *events*, a multiset $\rightarrow \subset Q \times \Sigma \times Q$ called the *transition relation*, a set $Q_I \subset Q$ of *initial states*, a set Π of *propositions*, and a *satisfaction relation* $\models \subset Q \times \Pi$. We write $q \xrightarrow{\sigma} q'$ instead of $(q, \sigma, q') \in \rightarrow$, and $q \models \pi$ instead of $(q, \pi) \in \models$. The transition system S

[3] The sampling rate of the controller may be any rational, but without loss of generality we assume it to be 1.

[4] Under the technical restriction that either the invariant and flow rectangles are positive, or the automaton state stays within a bounded region.

is *finite* if Q is finite. We assume for simplicity that S is deadlock-free; that is, for each state $q \in Q$, there exists an event $\sigma \in \Sigma$ and a state $r \in Q$ such that $q \xrightarrow{\sigma} r$. A *region* is a subset of Q. Given a proposition $\pi \in \Pi$, we write $R_\pi = \{q \in Q \mid q \models \pi\}$ for the region of states that satisfy π. ■

Verification as reachability

Definition 2.2 [Weakest precondition] Let S be a transition system. For each event $\sigma \in \Sigma$, the *σ-predecessor operator* $Pre_\sigma : 2^Q \to 2^Q$ is defined by $Pre_\sigma(R) = \{q \in Q \mid \exists r \in R.\, q \xrightarrow{\sigma} r\}$. In particular, $Pre_\sigma(Q)$ is the set of states in which the event σ is enabled. Define $Pre : 2^Q \to 2^Q$ by $Pre(R) = \bigcup_{\sigma \in \Sigma} Pre_\sigma(R)$. A region $R \subset Q$ is *reachable* in S if $Q_I \cap Pre^k(R) \neq \emptyset$ for some $k \in \mathbb{N}$. ■

The basic verification problem for transition systems asks whether an unsafe state is unreachable.

Definition 2.3 [Safety verification] Let $\mathcal{C}$ be a class of transition systems. The *safety verification problem* for $\mathcal{C}$ is stated in the following way: given a transition system $S \in \mathcal{C}$ and a proposition $\pi \in \Pi$, determine whether the region R_π is not reachable in S. ■

For finite transition systems, the safety verification problem is the complement of graph reachability, which can be solved in linear time and is complete for NLOGSPACE. The safety verification problem can be generalized to the safety control problem.

Control as alternating reachability We use the following model for control: for each state q of a transition system, a (memory-free) controller chooses an enabled event σ so that in state q, the controlled system always proceeds via event σ. Since q may have several σ-successors, the controlled system may still be nondeterministic. Alternative models for memory-free control are equivalent.

Definition 2.4 [Control map] Let S be a transition system. A *control map* for S is a function $\kappa : Q \to \Sigma$ such that for each state $q \in Q$, there exists a state $r \in Q$ with $q \xrightarrow{\kappa(q)} r$. The *closed-loop system* $\kappa(S)$ is the transition system $(Q, \Sigma, \Rightarrow, Q_I, \Pi, \models)$, where $q \overset{\sigma}{\Rightarrow} q'$ iff $q \xrightarrow{\sigma} q'$ and $\kappa(q) = \sigma$. ■

The basic control problem for transition systems asks whether an unsafe state is avoidable by applying some control map.

Definition 2.5 [Safety control] Let $\mathcal{C}$ be a class of transition systems. The *safety control decision problem* for $\mathcal{C}$ is stated in the following way: given a transition system $S \in \mathcal{C}$ and a proposition $\pi \in \Pi$, determine whether there exists a control map κ such that the region R_π is not reachable in the closed-loop system $\kappa(S)$. If so, then we say π is *avoidable* in S. The *safety controller synthesis problem* requires the construction of a witnessing control map κ when π is avoidable. ■

For finite transition systems, the safety control decision problem is the complement of AND-OR graph reachability, which can be solved in quadratic time and is complete for PTIME.

Definition 2.6 [Alternating reachability] An *AND-OR graph* $G = (V_A, V_O, V_I, \to)$ consists of a finite set $V = V_A \cup V_O$ of vertices that is partitioned into a set V_A of AND vertices and a set V_O of OR vertices, a set $V_I \subset V$ of initial vertices, and a multiset $\to \subset V \times V$ of edges. We assume deadlock freedom, namely, that for each vertex $v \in V$, there exists a vertex $w \in V$ such that $v \to w$. The *controllable predecessor operator* $CPre : 2^V \to 2^V$ is defined by $CPre(R) = \{q \in V_O \mid \exists r \in R.\, q \to r\} \cup \{q \in V_A \mid \forall r \in V.\, q \to r \text{ implies } r \in R\}$. A set $R \subset V$ of vertices is *alternating reachable* in G if $V_I \cap CPre^k(R) \neq \emptyset$ for some $k \in \mathbb{N}$. The *alternating reachability problem* asks whether a given set of vertices is alternating reachable in a given AND-OR graph. ■

Theorem 2.1 [Imm81] *The alternating reachability problem is complete for PTIME.*

There is a simple correspondence between safety control and alternating reachability. Let S be a finite transition system and let π be a proposition. Define an AND-OR graph G_S as follows: let $V_A = Q$ and $V_O = Q \times \Sigma$ and $V_I = Q_I$; for each vertex $q \in V_A$ and each event $\sigma \in \Sigma$, let $q \rightarrow (q, \sigma)$ in G_S iff $q \in Pre_\sigma(Q)$ in S; and for each vertex $(q, \sigma) \in V_O$, let $(q, \sigma) \rightarrow r$ in G_S iff $q \xrightarrow{\sigma} r$ in S. Then the proposition π is avoidable in S iff the set R_π of AND vertices is not alternating reachable in G_S.

Corollary 2.1 *The safety control decision problem for finite transition systems is complete for PTIME.*

Moreover, a byproduct of a negative alternating reachability computation is a control map that avoids π. Note that for each set $R \subset Q$ of AND vertices, $CPre^2(R) = \bigcap_{\sigma \in \Sigma}(Pre_\sigma(R) \cup (Q \setminus Pre_\sigma(Q)))$. Thus the region $CPre^2(R)$ is the set of all states that no control map can keep out of R at the next transition. Let $R_F = CPre^{2|Q|}(R_\pi)$. Then π is avoidable in S iff $Q_I \cap R_F = \emptyset$. Each application of $CPre^2$ can be computed in linear time, so R_F can be computed in quadratic time. If π is indeed avoidable, then a witnessing control map may be constructed by choosing for each state $q \in Q \setminus R_F$ an event σ such that $q \in Pre_\sigma(Q) \setminus Pre_\sigma(R_F)$.

Theorem 2.2 [RW87] *The safety controller synthesis problem for finite transition systems can be solved in quadratic time.*

Effectively-presented transition systems with finite bisimilarity quotients The safety controller synthesis problem can be solved not only for finite transition systems, but also for effectively-presented transition systems with finite bisimilarity quotients.

Definition 2.7 [Effective presentation] A *symbolic execution theory* for the transition system S consists of a set $\mathcal{F}$ of *formulas*, a formula $\phi_I \in \mathcal{F}$, and a map $[\![\cdot]\!] : \mathcal{F} \rightarrow 2^Q$ such that (1) every proposition $\pi \in \Pi$ is a formula: $[\![\pi]\!] = R_\pi$; (2) for all formulas $\phi_1, \phi_2 \in \mathcal{F}$, the three expressions $\phi_1 \wedge \phi_2$ and $\phi_1 \vee \phi_2$ and $\neg\phi_1$ are formulas: $[\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]$ and $[\![\phi_1 \vee \phi_2]\!] = [\![\phi_1]\!] \cup [\![\phi_2]\!]$ and $[\![\neg\phi_1]\!] = Q \setminus [\![\phi_1]\!]$; (3) $[\![\phi_I]\!] = Q_I$; (4) the set $\{\phi \in \mathcal{F} \mid [\![\phi]\!] = \emptyset\}$ is recursive; and (5) for each event $\sigma \in \Sigma$, there is a computable map $Pre_\sigma : \mathcal{F} \rightarrow \mathcal{F}$ such that $[\![Pre_\sigma(\phi)]\!] = Pre_\sigma([\![\phi]\!])$ for all formulas $\phi \in \mathcal{F}$. An *effectively-presented transition system* consists of a transition system S together with a symbolic execution theory for S. ■

Definition 2.8 [Bisimilarity] A *bisimulation* on the transition system S is an equivalence relation $\cong$ on the state set Q such that (1) if $q \cong r$ then for all propositions $\pi \in \Pi$, we have $q \models \pi$ iff $r \models \pi$, and (2) if $q \cong r$ and $q \xrightarrow{\sigma} q'$, then there exists a state $r' \in Q$ such that $r \xrightarrow{\sigma} r'$ and $q' \cong r'$. The largest bisimulation on S is denoted by $\equiv$. The *bisimilarity quotient* $S/_\equiv$ is the transition system $(Q/_\equiv, \Sigma, \rightarrow_\exists, Q_\exists, \Pi, \models_\exists)$, where $R \xrightarrow{\sigma}_\exists R'$ iff there exist two states $q \in R$ and $q' \in R'$ such that $q \xrightarrow{\sigma} q'$, where $R \in Q_\exists$ iff $R \cap Q_I \neq \emptyset$, and where $R \models_\exists \pi$ iff $R \cap R_\pi \neq \emptyset$. ■

The controllable-predecessor operator $CPre^2$ can be computed on any effectively-presented transition system. When the bisimilarity quotient has $k \in \mathbb{N}$ equivalence classes, the R_F computation converges in at most k iterations of $CPre^2$. Synthesizing a control map is accomplished by first computing the bisimilarity quotient, and then choosing for each state in each equivalence class R disjoint from R_F, an event $\sigma \in \Sigma$ such that $R \cap Pre_\sigma(Q) \neq \emptyset$ and $R \cap Pre_\sigma(R_F) = \emptyset$.

Theorem 2.3 [Hen95] *The safety control decision problem is decidable for effectively-presented transition systems with finite bisimilarity quotients. Moreover, when a proposition is avoidable, a witnessing control map can be computed.*

This result can be generalized to liveness verification such as μ-calculus model checking, and to memory-free liveness control such as control-map synthesis for Rabin chain conditions.

2.2 Rectangular Hybrid Automata

Definition 2.9 [Rectangle] Let $X = \{x_1, \ldots, x_n\}$ be a set of real-valued variables. A *rectangular inequality* over X is a formula of the form $x_i \sim c$, where c is an integer constant, and $\sim$ is one of $<, \leq, >, \geq$. A *rectangular predicate* over X is a conjunction of rectangular inequalities. The rectangular predicate ϕ defines the set of vectors $[\![\phi]\!] = \{\mathbf{y} \in \mathbb{R}^n \mid \phi[X := \mathbf{y}] \text{ is true}\}$. A set of the form $[\![\phi]\!]$, where ϕ is a rectangular predicate, is called a *rectangle*. Given a positive integer $m \in \mathbb{N}_{>0}$, the rectangular predicate ϕ and the rectangle $[\![\phi]\!]$ are *m-definable* if $|c| \leq m$ for every conjunct $x_i \sim c$ of ϕ. The set of all rectangular predicates over X is denoted $Rect(X)$. ■

Definition 2.10 [Rectangular automaton][HKPV95] A *rectangular automaton* A consists of the following components:

Variables. A finite set $X = \{x_1, \ldots, x_n\}$ of real-valued *variables* representing the continuous component of the system. The number n is the *dimension* of A. We write $\dot{X}$ for the set $\{\dot{x}_i \mid x_i \in X\}$ of dotted variables, and X' for the set $\{x_i' \mid x_i \in X\}$ of primed variables.

Control graph. A finite directed multigraph (V, E) representing the discrete component of the system. The vertices in V are called *control modes*. The edges in E are called *control switches*.

Invariant conditions. A function $inv\colon V \to Rect(X)$ mapping each control mode to its *invariant condition*, a rectangular predicate.

Initial conditions. A function $init\colon V \to Rect(X)$ mapping each control mode to its *initial condition*, a rectangular predicate.

Jump conditions. A function *jump* mapping each control switch $e \in E$ to a predicate $jump(e)$ of the form $\phi \wedge \phi' \wedge \bigwedge_{i \notin update(e)}(x_i' = x_i)$, where $\phi \in Rect(X)$ and $\phi' \in Rect(X')$ are rectangular predicates, and $update(e) \subset \{1, \ldots, n\}$. The jump condition $jump(e)$ specifies the effect of the change in control mode on the values of the variables: each unprimed variable x_i refers to a value before the control switch e, and each primed variable x_i' refers to the corresponding value after the control switch.

Flow conditions. A function $flow\colon V \to Rect(\dot{X})$ mapping each control mode v to a *flow condition*, a rectangular predicate that constrains the behavior of the first derivatives of the variables while time passes in control mode v.

Events. A finite set Σ of *events*, and a function $event\colon E \to \Sigma$ mapping each control switch to an event.

Thus a rectangular automaton A is a tuple $(X, V, E, inv, init, jump, flow, \Sigma, event)$. The automaton A is *m-definable* if every rectangular predicate in the definition of A is m-definable. The automaton A is *positive* if for every control mode $v \in V$, the invariant rectangle $[\![inv(v)]\!]$ and the flow rectangle $[\![flow(v)]\!]$ are subsets of the positive orthant $\mathbb{R}^n_{\geq 0}$. The automaton A is *bounded* if for every control mode $v \in V$, the invariant rectangle $[\![inv(v)]\!]$ is a bounded set. ■

The state of a rectangular automaton has two parts: a discrete (or control) part, and a continuous (or plant) part. The discrete state is a control mode. The continuous state is a valuation for the variables.

Definition 2.11 [States of rectangular automata] Let A be a rectangular automaton. A *state* of A is a pair $(v, \mathbf{y})$, where $v \in V$ is a control mode and $\mathbf{y} \in [\![inv(v)]\!]$ is a vector satisfying the invariant condition of v. Thus the set of states is $Q = \{(v, \mathbf{y}) \in V \times \mathbb{R}^n \mid \mathbf{y} \in [\![inv(v)]\!]\}$. A subset of Q is called a *region* of A. A *rectangular state predicate* for A is a function ψ from V to $Rect(X)$. The rectangular state predicate ψ defines the region $[\![\psi]\!] = \{(v, \mathbf{y}) \in Q \mid \mathbf{y} \in [\![\psi(v)]\!]\}$. A region of the form $[\![\psi]\!]$, where ψ is a rectangular state predicate for A, is called a *rectangular region*. The initial condition map defines the rectangular region $Q_I = [\![init]\!]$ of *initial states*. ■

A rectangular automaton makes two types of transitions: jump (or edge, or control) transitions, and flow (or time, or plant) transitions. Jump transitions are instantaneous. They are characterized

by a change in control mode, and are accompanied by discrete modifications to the variables in accordance with the jump condition of the control switch. During flow transitions, while time elapses, the control mode remains fixed and the variables evolve continuously via a trajectory that satisfies the flow condition of the active control mode.

Definition 2.12 [Transitions of rectangular automata] Let A be a rectangular automaton. For each event $\sigma \in \Sigma$, we define the *jump relation* $\xrightarrow{\sigma} \subset Q^2$ by $(v, \mathbf{y}) \xrightarrow{\sigma} (v', \mathbf{y}')$ iff there exists a control switch $e = (v, v') \in E$ such that $event(e) = \sigma$ and $(\mathbf{y}, \mathbf{y}') \in [\![jump(e)]\!]$. For each nonnegative real $\delta \in \mathbb{R}_{\geq 0}$, we define the *flow relation* $\xrightarrow{\delta} \subset Q^2$ by $(v, \mathbf{y}) \xrightarrow{\delta} (v', \mathbf{y}')$ iff (1) $v = v'$, and (2) there exists a differentiable function $f : [0, \delta] \to [\![inv(v)]\!]$ such that $f(0) = \mathbf{y}$ and $f(\delta) = \mathbf{y}'$, and $\dot{f}(\epsilon) \in [\![flow(v)]\!]$ for all reals $\epsilon \in (0, \delta)$, where $\dot{f}$ is the first derivative of f. We say that δ is the *duration* of the flow transition. Since the rectangle $[\![inv(v)]\!]$ is a convex set, it follows that for $\delta > 0$, condition (2) is equivalent to $\frac{\mathbf{y}' - \mathbf{y}}{\delta} \in [\![flow(v)]\!]$; that is, all flows can be thought of as straight lines. ■

Every rectangular automaton defines two transition systems.

Definition 2.13 [Discrete time and dense time] Let A be a rectangular automaton. Define the binary relation $\xrightarrow{time} \subset Q^2$ by $(v, \mathbf{y}) \xrightarrow{time} (v', \mathbf{y}')$ iff $(v, \mathbf{y}) \xrightarrow{\delta} (v', \mathbf{y}')$ for some duration $\delta \in \mathbb{R}_{\geq 0}$. Define Π to be the set of rectangular state predicates for A, and for all states $(v, \mathbf{y}) \in Q$, define $(v, \mathbf{y}) \models \pi$ iff $(v, \mathbf{y}) \in [\![\pi]\!]$. The *discrete-time transition system* of A is defined by $S_A^{disc} = (Q, \Sigma \cup \{1\}, \to, Q_I, \Pi, \models)$. The *dense-time transition system* of A is defined by $S_A^{dense} = (Q, \Sigma \cup \{time\}, \to, Q_I, \Pi, \models)$. Thus all flow transitions in the discrete-time transition system are required to have duration 1, while flow transitions in the dense-time transition system can have any nonnegative real duration. We refer to the safety verification problem for transition systems of the form S_A^{disc} (resp. S_A^{dense}), for some rectangular automaton A, as the *discrete-time* (resp. *dense-time*) *safety verification problem* for rectangular automata, and similarly for the control decision and controller synthesis problems. ■

Dense-time undecidability results In dense time, the verification and control of rectangular automata cannot be fully automated.

Theorem 2.4 [ACH+95] *For positive and bounded rectangular automata, the dense-time safety verification problem (and thus the dense-time safety control decision problem) is undecidable.*

Research has therefore concentrated on subclasses of rectangular automata. In [HKPV95] it is shown that for *initialized* rectangular automata, whose flow condition map is a constant function (i.e., all control modes have the same flow condition), the dense-time safety verification problem (in fact, LTL model checking) can be decided. These automata, however, have no finite bisimilarity quotients in dense time [Hen95], and therefore further restrictions are desirable.

Timed automata An important special case of initialized rectangular automata are timed automata. All variables of a timed automaton are clocks, which advance uniformly at rate 1 while time elapses.

Definition 2.14 [Timed automaton][AD94] A *timed automaton* is a positive rectangular automaton A with the restriction that $flow(v) = \bigwedge_{i=1}^{n} (\dot{x}_i = 1)$ for every control mode v. A *triangular inequality* over a set X of variables is a formula of the form $x_i - x_j \sim c$, where $x_i, x_j \in X$ are variables, c is an integer constant, and $\sim$ is one of $<, \leq, >, \geq$. A *triangular predicate* over X is a conjunction of rectangular and triangular inequalities. A *triangular state predicate* for a timed automaton A is a function that maps every control mode of A to a triangular predicate over the variables of A. ■

The fundamental theorem for timed automata states that the dense-time transition system S_A^{dense} of a timed automaton A has a finite bisimilarity quotient and can be presented effectively using triangular state predicates.

Theorem 2.5 [AD94, HNSY94] *For every m-definable n-dimensional timed automaton A with k control modes, the dense-time transition system S_A^{dense} has a finite bisimilarity quotient with $O(k \cdot (n+1)! \cdot (2m)^n)$ many equivalence classes. Moreover, the boolean combinations of triangular state predicates for A form a symbolic execution theory for S_A^{dense}.*

Corollary 2.2 *For timed automata, the dense-time safety verification problem (in fact, LTL and CTL model checking) can be solved in PSPACE, and the dense-time safety controller synthesis problem can be solved in EXPTIME.*

As for finite transition systems, control is harder than verification. In [AD94] it is shown that the dense-time safety verification problem for timed automata is hard for PSPACE. From Theorem 3.2 below it follows that the dense-time safety control decision problem for timed automata is hard for EXPTIME.

3 Discrete-Time Rectangular Automata

3.1 Finite Bisimilarity Quotients and Effective Presentation

We show that the discrete-time transition system S_A^{disc} of a positive or bounded rectangular automaton A has a finite bisimilarity quotient and can be presented effectively using rectangular state predicates. More precisely, in discrete time, two states of a rectangular automaton are bisimilar if (1) they have the same control mode, (2) corresponding variable values agree on their integer parts, and (3) corresponding variable values agree on whether they are integral. Moreover, if an m-definable rectangular automaton is positive, then it cannot distinguish variable values greater than m. For m-definable bounded rectangular automata, the continuous part of the state is contained in the cube $[-m, m]^n$. It follows that in both the positive and the bounded case, the bisimilarity quotient is finite.

Definition 3.1 Define the equivalence relation $\approx_n$ on $\mathbb{R}^n$ by $\mathbf{y} \approx_n \mathbf{z}$ iff $\lfloor y_i \rfloor = \lfloor z_i \rfloor$ and $\lceil y_i \rceil = \lceil z_i \rceil$ for all $1 \leq i \leq n$. Given $m \in \mathbb{N}_{>0}$, define the equivalence relation $\approx_n^m$ on $\mathbb{R}^n$ by $\mathbf{y} \approx_n^m \mathbf{z}$ iff for each $1 \leq i \leq n$, either $y_i \approx_1 z_i$, or both y_i and z_i are greater than m, or both y_i and z_i are less than $-m$. For an n-dimensional rectangular automaton A, define the equivalence relations $\cong_A$ and $\cong_A^m$ on the states of A by $(v, \mathbf{y}) \cong_A (w, \mathbf{z})$ iff $v = w$ and $\mathbf{y} \approx_n \mathbf{z}$, and $(v, \mathbf{y}) \cong_A^m (w, \mathbf{z})$ iff $v = w$ and $\mathbf{y} \approx_n^m \mathbf{z}$. ■

Lemma 3.1 *Consider two vectors $\mathbf{y}, \mathbf{z} \in \mathbb{R}^n$. Then $\mathbf{y} \approx_n \mathbf{z}$ iff for every rectangle $B \subset \mathbb{R}^n$, we have $\mathbf{y} \in B$ iff $\mathbf{z} \in B$. Moreover, $\mathbf{y} \approx_n^m \mathbf{z}$ iff for every m-definable rectangle $B \subset \mathbb{R}^n$, we have $\mathbf{y} \in B$ iff $\mathbf{z} \in B$.*

Theorem 3.1 *Let A be an n-dimensional rectangular automaton with k control modes. The equivalence relation $\cong_A$ is a bisimulation on the discrete-time transition system S_A^{disc}. If A is m-definable and either positive or bounded, then $\cong_A^m$ is also a bisimulation on S_A^{disc}. The number of equivalence classes of $\cong_A^m$ is $k \cdot (4m+3)^n$.*

Proof. We argue that $\cong_A^m$ is a bisimulation for positive m-definable A; the other parts of the proof are similar. Suppose that $(v, \mathbf{y}) \cong_A^m (w, \mathbf{z})$ and $(v, \mathbf{y}) \xrightarrow{\sigma} (v', \mathbf{y}')$. We must show that there exists a state $(w', \mathbf{z}')$ such that $(w, \mathbf{z}) \xrightarrow{\sigma} (w', \mathbf{z}')$ and $(v', \mathbf{y}') \cong_A^m (w', \mathbf{z}')$. First, assume that $\sigma \in \Sigma$. In this case there exists a control switch e with source $v = w$ such that $event(e) = \sigma$ and $(\mathbf{y}, \mathbf{y}') \in [\![jump(e)]\!]$, and $y_i = y_i'$ for each $i \notin update(e)$. Define $\mathbf{z}'$ by $z_i' = z_i$ for $i \notin update(e)$,

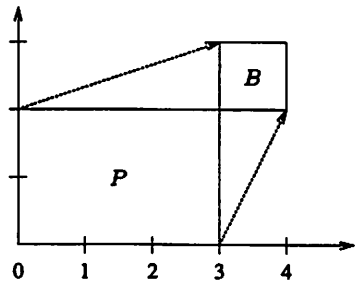

Fig. 1. Given a control mode v, consider the flow condition $\mathit{flow}(v) = (1 \leq \dot{x}_1 \leq 3 \wedge 1 \leq \dot{x}_2 \leq 2)$. Let $B = [\![3 \leq x_1 \leq 4 \wedge 2 \leq x_2 \leq 3]\!]$ and $P = [\![0 \leq x_1 \leq 3 \wedge 0 \leq x_2 \leq 2]\!]$. Then $\mathit{Pre}_1(\{v\} \times B) = \{v\} \times P$.

and $z_i' = y_i'$ for $i \in \mathit{update}(e)$. By Lemma 3.1, $(\mathbf{z}, \mathbf{z}') \in [\![\mathit{jump}(e)]\!]$ and $\mathbf{z}' \in [\![\mathit{inv}(v')]\!]$. It follows that $(w, \mathbf{z}) \xrightarrow{\sigma} (v', \mathbf{z}')$.

Second, assume that $\sigma = 1$ (cf. Fig. 1). In this case $v' = v = w$, and $\mathbf{y}' - \mathbf{y} \in [\![\mathit{flow}(v)]\!]$. We must show that there exists a vector $\mathbf{z}'$ such that $\mathbf{z}' - \mathbf{z} \in [\![\mathit{flow}(v)]\!]$ and $\mathbf{y}' \approx_n^m \mathbf{z}'$ (notice that by Lemma 3.1, $\mathbf{y}' \approx_n^m \mathbf{z}'$ implies $\mathbf{z}' \in [\![\mathit{inv}(v)]\!]$). We do this one coordinate at a time. Fix $i \in \{1, \ldots, n\}$. Suppose that $y_i > m$. It follows that $y_i' > m$ and $z_i > m$, because A is positive. Choose any $c \in [\![\mathit{flow}(v)]\!]_i$, and define $z_i' = z_i + c$. Since $c \geq 0$, we have $y_i' \approx_1^m z_i'$. Now suppose that $y_i \leq m$. If $y_i \in \mathbb{N}$ then $z_i = y_i$, because $y_i \approx_1 z_i$. Define $z_i' = y_i'$. Then $z_i' - z_i = y_i' - y_i \in [\![\mathit{flow}(v)]\!]_i$. If $y_i \notin \mathbb{N}$ then $\lfloor y_i \rfloor < y_i, z_i < \lceil y_i \rceil$. The set $[\![\mathit{flow}(v)]\!]_i$ is an interval, say, with endpoints $a, b \in \mathbb{N}$ (it is easy to extend the argument to the case $b = \infty$). Thus $[\![\mathit{flow}(v)]\!]_i$ contains the open interval (a, b), and $y_i' \in [y_i + a, y_i + b]$. We show that there exists a number $c \in (a, b)$ such that $y_i' \approx_1 z_i + c$. Since $a, b \in \mathbb{N}$ and $y_i \approx_1 z_i$, it follows that $y_i + a \approx_1 z_i + a$ and $y_i + b \approx_1 z_i + b$. Thus the closed interval $[z_i + a, z_i + b]$ intersects the same $\approx_1$-equivalence classes as does $[y_i + a, y_i + b]$. Since neither $z_i + a$ nor $z_i + b$ is an integer, the same is true for the open interval $(z_i + a, z_i + b)$. Therefore there exists a number $c \in (a, b)$ such that $y_i' \approx_1 z_i + c$. ■

Corollary 3.1 *For every rectangular automaton A, the boolean combinations of rectangular state predicates for A form a symbolic execution theory for the discrete-time transition system S_A^{disc}.*

Corollary 3.2 *For positive or bounded rectangular automata, the discrete-time safety verification problem (in fact, LTL and CTL model checking) can be solved in PSPACE, and the discrete-time safety controller synthesis problem can be solved in EXPTIME.*

The LTL and CTL parts of the corollary follow from the facts that both model-checking problems can be solved in space logarithmic in the size of the transition system and polynomial in the size of the temporal formula [Kup95]. It should be noted that while in the same complexity class, the actual running times of the discrete-time algorithms for rectangular automata are better by a multiplicative exponential factor than the running times of the corresponding dense-time algorithms for timed automata. This is because there, the number of equivalence classes of the bisimilarity quotient is $\Omega(k \cdot n! \cdot (m+1)^n)$. By providing tight lower bounds, the following theorem shows that our algorithms are optimal. The second part of the theorem follows from Theorem 3.4 below.

Theorem 3.2 *For bounded timed automata, the discrete-time safety verification problem is hard for PSPACE [AD94], and the discrete-time safety control decision problem is hard for EXPTIME.*

3.2 Sampling-Controller Synthesis

The dense-time and discrete-time control problems are not realistic, as a controller may enforce arbitrarily many (even infinitely many) consecutive instantaneous jumps. A more natural control model for hybrid systems involves a controller that samples the plant state once per time unit, and then issues a command based upon its measurement. The command may cause a switch in control mode, after which the plant state evolves continuously for one time unit, before receiving the next command. We call this model "sampling control" to distinguish it from discrete-time control. Moreover, we wish to ensure that a proposition is avoided not only at the sampling points but also between sampling points. Given a rectangular automaton A, we define a third transition system, S_A^{sample}, such that (1) any control map behaves in a sampling manner and (2) the propositional regions are "large enough" so that they cannot be entered and left by a single flow transition of duration 1. For example, if π is a rectangular state predicate that maps each control mode of A to either *true* or *false*, then R_π is large enough. If the region of unsafe states is not large enough, this may be correctable by increasing the sampling rate (i.e., by reducing the unit of time).

Definition 3.2 [Sampling control] Let A be a rectangular automaton. A rectangular state predicate $\pi \in \Pi$ is *large enough* for A if there are no three states $(v, \mathbf{y}), (v, \mathbf{y}') \notin R_\pi$ and $(v, \mathbf{y}'') \in R_\pi$ such that $(v, \mathbf{y}) \xrightarrow{\delta} (v, \mathbf{y}'')$ and $(v, \mathbf{y}'') \xrightarrow{1-\delta} (v, \mathbf{y}')$ for some real $\delta \in (0, 1)$. Define $\Pi' \subset \Pi$ to be the set of rectangular state predicates that are large enough for A, and define $((v, \mathbf{y}), \lambda) \models' \pi$ iff $(v, \mathbf{y}) \models \pi$. The *sampling-control transition system* of A is defined by $S_A^{sample} = (Q \times \{control, plant\}, \Sigma \cup \{1\}, \Rightarrow, Q_I \times \{control\}, \Pi', \models')$, where the binary relation $\Rightarrow$ is defined by: (1) for each event $\sigma \in \Sigma$, we have $((v, \mathbf{y}), control) \stackrel{\sigma}{\Rightarrow} ((v', \mathbf{y}'), plant)$ iff $(v, \mathbf{y}) \xrightarrow{\sigma} (v', \mathbf{y}')$, and (2) $((v, \mathbf{y}), plant) \stackrel{1}{\Rightarrow} ((v', \mathbf{y}'), control)$ iff $(v, \mathbf{y}) \xrightarrow{1} (v', \mathbf{y}')$. Thus in the sampling-control transition system the controller and the plant take turns: first the controller specifies a jump transition, then one time unit passes in a flow transition, and so on. We refer to the safety control decision problem for transition systems of the form S_A^{sample}, for some rectangular automaton A, as the *sampling-control decision problem* for rectangular automata, and similarly for the sampling-controller synthesis problem. ■

Theorem 3.3 *For positive or bounded rectangular automata, the sampling-controller synthesis problem can be solved in EXPTIME.*

Proof. Consider an n-dimensional positive or bounded rectangular automaton A. We reduce the sampling-control problems to discrete-time control problems by constructing a rectangular automaton $Ctrl(A)$ such that S_A^{sample} is isomorphic to $S_{Ctrl(A)}^{disc}$. Moreover, if A is positive, then $Ctrl(A)$ is positive, and if A is bounded, then $Ctrl(A)$ is bounded. Let $X_{Ctrl(A)} = X_A \cup \{x_{n+1}\}$ for a clock $x_{n+1} \notin X_A$. The control graph and events of $Ctrl(A)$ are identical to those of A. Let $inv_{Ctrl(A)}(v) = inv_A(v) \wedge 0 \leq x_{n+1} \leq 1$, let $init_{Ctrl(A)}(v) = init_A(v) \wedge x_{n+1} = 1$, let $jump_{Ctrl(A)}(e) = jump_A(e) \wedge x_{n+1} = 1 \wedge x'_{n+1} = 0$, and let $flow_{Ctrl(A)}(v) = flow_A(v) \wedge \dot{x}_{n+1} = 1$. It follows that in the discrete-time transition system $S_{Ctrl(A)}^{disc}$, jump transitions must alternate with flow transitions (of duration 1). Hence the map $f : Q_{Ctrl(A)} \to Q_A \times \{control, plant\}$, defined by $f(v, \mathbf{y}, 0) = (v, \mathbf{y}, plant)$ and $f(v, \mathbf{y}, 1) = (v, \mathbf{y}, control)$, is an isomorphism between the transition systems $S_{Ctrl(A)}^{disc}$ and S_A^{sample}. If A is m-definable with k control modes, by Theorem 3.1, the bisimilarity quotient of $S_{Ctrl(A)}^{disc}$ has no more than $k \cdot (4m + 3)^{n+1}$ equivalence classes, which is singly exponential in the size of A. ■

Lemma 3.2 *Let $G = (V_A, V_O, V_I, \rightarrow)$ be an AND-OR graph, and let R be a set of vertices of G. Define the transition system $S_G = (V_A \cup V_O, \Sigma, \rightarrow, V_I, \{\pi\}, \models)$ such that (1) $v \models \pi$ iff $v \in R$, (2) for all OR states $v \in V_O$, if $v \xrightarrow{\sigma} w$ and $v \xrightarrow{\sigma} w'$, then $\sigma = \sigma'$, and (3) for all AND states $v \in V_A$, if $v \xrightarrow{\sigma} w$ and $v \xrightarrow{\sigma} w'$ and $w \neq w'$, then $\sigma \neq \sigma'$. Then R is alternating reachable in G iff π is not avoidable in S_G.*

Theorem 3.4 *For bounded timed automata, the sampling-control decision problem is hard for EXPTIME.*

Proof sketch. We reduce the halting problem for alternating Turing machines using polynomial space [CKS81] to the sampling-control decision problem for bounded timed automata. Let M be an alternating Turing Machine with input s so that M uses space $p(|s|)$. Then M accepts s iff the unique final state u_F is alternating reachable in an AND-OR graph whose vertices are configurations of M. The set of configurations of M is $U \times \{1, \ldots, p(|s|)\} \times \Gamma^{p(|s|)}$, where U is the state set of M, the second component of the product gives the position of the tape head, and Γ is the tape alphabet. Without loss of generality, we assume that $\Gamma = \{0, 1, 2\}$, where 0 is the "blank" symbol. We first define a bounded positive rectangular automaton A whose states are configurations of M, and a proposition π_F, large enough for A, that is true exactly in the configurations containing u_F. This is done in a way consistent with Lemma 3.2, so that π_F is not avoidable in S_A^{sample} iff M accepts s. Then we turn A into a bounded timed automaton.

The automaton A uses $p(|s|)$ variables $x_1, \ldots, x_{p(|s|)}$ to store the tape contents. The set of control modes of A is $U \times \{1, \ldots, p(|s|)\}$. The invariant and flow conditions are constant functions: $inv(u, i) = \bigwedge_{j=1}^{p(|s|)}(0 \leq x_j \leq 2)$ and $flow(u, i) = \bigwedge_{j=1}^{p(|s|)}(\dot{x}_j = 0)$ for all u and i; thus flow transitions have no effect. The initial condition is defined by $init(u, i) = false$ except when u is the initial state u_I of M and $i = 1$; in that case, $init(u_I, 1) = \bigwedge_{j=1}^{|s|}(x_j = s_j) \wedge \bigwedge_{j=|s|+1}^{p(|s|)}(x_i = 0)$. Each transition t of M consists of a source state $u \in U$, a tape symbol $\gamma \in \Gamma$, and a list of triples (u_j, γ_j, d_j), where $u_j \in U$ is a target state, $\gamma_j \in \Gamma$ is written on the current tape cell, and $d_j \in \{-1, 1\}$ gives the direction moved by the tape head (there is exactly one transition for each source state u). For every transition $t = (u, \gamma, (u_j, \gamma_j, d_j)_{j \in J})$ of M, every tape position $1 \leq i \leq p(|s|)$, and every $j \in J$, we define in A a control switch $e_{t,i,j}$ with source (u, i) and target $(u_j, i + d_j)$. The jump condition $jump(e_{t,i,j})$ is $x_i = \gamma \wedge x_i' = \gamma_j \wedge \bigwedge_{k \neq i}(x_k' = x_k)$. If u is an AND state of M, then $event(e_{t,i,j}) = (u, i, j)$. If u is an OR state of M, then $event(e_{t,i,j}) = 0$. To turn A into a timed automaton, all variables are replaced by clocks, and between any two control switches of A, a sequence of $p(|s|)$ control switches is added, one for each clock, to subtract $p(|s|) + 1$ from each clock value. ■

4 Beyond Rectangular Automata

Discrete-Time Undecidability Results We show that the pleasant properties of discrete-time rectangular automata (Theorem 3.1) depend on both conditions, (1) positivity or boundedness and (2) rectangularity. If either condition is violated, then already the discrete-time safety verification problem becomes undecidable.

Definition 4.1 [Triangular automaton] A *triangular automaton* A has the same components as a rectangular automaton, except that the predicates defining A may be triangular predicates, and need not necessarily be rectangular. ■

Theorem 4.1 *The discrete-time safety verification problem (and thus the discrete-time control decision problem) is undecidable for the class of all rectangular automata, and also for the class of bounded positive triangular automata.*

Proof sketch. Both parts use a reduction from the halting problem for two-counter machines. For the first part, the reduction is simple, as counter values can be represented by variable values, as in [KPSY93]. For the second part, counter values must be encoded, so that the counter value c corresponds to the variable value $\frac{1}{2^c}$. For this purpose, the wrapping-clock technique of [HKPV95] can be modified as follows. The set $\{x_1, \ldots, x_n\}$ of dense-time clocks used for encoding counter values is simulated in discrete time by variables with the triangular flow condition $\dot{x}_1 = \cdots = \dot{x}_n$.

Then the variables are enforced to represent valid encodings at those integer times when the wrapping clock shows 0. ■

Generalized Rectangular Automata It is well-known that the pleasant properties of timed automata (Theorem 2.5) are preserved if rectangularity is relaxed to triangularity in invariant, initial, and jump conditions. We conclude with a similar observation for rectangular automata. A *generalized rectangular automaton* is a triangular automaton whose flow conditions are rectangular predicates. It follows from our arguments that for every generalized rectangular automaton A, the boolean combinations of triangular state predicates for A form a symbolic execution theory for the discrete-time transition system S_A^{disc}. Consequently, if A is a *bounded* generalized rectangular automaton, then S_A^{disc} has a finite bisimilarity quotient (which is identical to the region equivalence of timed automata [AD94], and finer by a multiplicative exponential factor than the equivalence of Theorem 3.1). For such automata, we can automatically synthesize sampling controllers that avoid triangular state predicates.

References

[ACH+95] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

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

[AH97] R. Alur, T.A. Henzinger. Modularity for timed and hybrid systems. In *CONCUR: Concurrency Theory*, LNCS. Springer, 1997.

[BCM+92] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, L.J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98:142–170, 1992.

[CKS81] A.K. Chandra, D.C. Kozen, L.J. Stockmeyer. Alternation. *J. ACM*, 28:114–133, 1981.

[Hen95] T.A. Henzinger. Hybrid automata with finite bisimulations. In *ICALP: Automata, Languages, and Programming*, LNCS 944, pp. 324–335. Springer, 1995.

[Hen96] T.A. Henzinger. The theory of hybrid automata. In *Proc. 11th Symp. Logic in Computer Science*, pp. 278–292. IEEE, 1996.

[HH95] T.A. Henzinger, P.-H. Ho. Algorithmic analysis of nonlinear hybrid systems. In *CAV: Computer-Aided Verification*, LNCS 939, pp. 225–238. Springer, 1995.

[HHW97] T.A. Henzinger, P.-H. Ho, H. Wong-Toi. HYTECH: a model checker for hybrid systems. In *CAV: Computer-Aided Verification*, LNCS. Springer, 1997.

[HKPV95] T.A. Henzinger, P.W. Kopke, A. Puri, P. Varaiya. What's decidable about hybrid automata? In *Proc. 27th Symp. Theory of Computing*, pp. 373–382. ACM, 1995.

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

[Imm81] N. Immerman. Number of quantifiers is better than number of tape cells. *J. Computer and System Sciences*, 22:384–406, 1981.

[KPSY93] Y. Kesten, A. Pnueli, J. Sifakis, S. Yovine. Integration graphs: a class of decidable hybrid systems. In *Hybrid Systems*, LNCS 736, pp. 179–208. Springer, 1993.

[Kup95] O. Kupferman. *Model Checking for Branching-Time Temporal Logics*. PhD thesis, The Technion, Haifa, Israel, 1995.

[MPS95] O. Maler, A. Pnueli, J. Sifakis. On the synthesis of discrete controllers for timed systems. In *STACS: Theoretical Aspects of Computer Science*, LNCS 900, pp. 229–242. Springer, 1995.

[PV94] A. Puri, P. Varaiya. Decidability of hybrid systems with rectangular differential inclusions. In *CAV: Computer-Aided Verification*, LNCS 818, pp. 95–104. Springer, 1994.

[RW87] P.J. Ramadge, W.M. Wonham. Supervisory control of a class of discrete-event processes. *SIAM J. Control and Optimization*, 25:206–230, 1987.