Verification of Infinite-State Dynamic Systems Using Approximate Quotient Transition Systems

Alongkrit Chutinan and Bruce H. Krogh, *Fellow, IEEE*

***Abstract*—This paper concerns computational methods for verifying properties of labeled infinite-state transition systems (e.g., *hybrid systems*) using *quotient transition system (QTS)*. A QTS is a conservative approximation to the infinite-state transition system based on a finite partition of the infinite state space. For universal specifications, positive verification for a QTS implies the specification is true for the infinite-state transition system. Since in most applications exact reachability mappings required to compute the QTS cannot be represented or computed, we introduce the *approximate quotient transition system (AQTS)*. The AQTS is an approximation to the QTS obtained from a conservative approximation to the reachability mapping. The paper presents a sufficient condition for an AQTS to be a *bisimulation* of the infinite state transition system. An AQTS bisimulation is essentially equivalent to the infinite-state system for the purposes of verification. It is well known, however, that finite-state bisimulations do not exist for most hybrid systems of practical interest. Therefore, the use of the AQTS for verification of universal specifications is proposed and illustrated with an example. This approach has been implemented in a tool for computer-aided verification of a general class of hybrid systems.**

***Index Terms*—Bisimulation, hybrid systems, reachability, temporal logic, verification.**

I. Introduction

MANY complex dynamic systems can be described as *transition systems*, where the system dynamics are interpreted as a *transition relation* on the state space. This approach has been used, for example, to analyze *hybrid systems* [1]–[3]. This paper concerns formal verification of the infinite-state transition system resulting from such an interpretation, or *semantics*, for a dynamic system. The general verification problem can be stated as follows: Given a desired property, called a *specification*, we would like to guarantee that all of the system behaviors satisfy the specification. This is a very important problem in the validation of the system design, especially for safety-critical applications [4], [5].

Previous work on formal verification dealt mainly with finite-state systems [6]–[8]. The system under consideration is typically described as a finite graph with nodes representing the system states and arcs representing possible state transitions. Specifications are written using formalisms such as *computation tree logic* (CTL) [6], [8].

CTL uses the notion of the *computation tree*, which is the infinite tree obtained by unfolding the state transition graph along all possible state-transition sequences starting from a designated initial state. A CTL specification describes the system evolution along paths in the computation tree and can be verified through symbolic reachability analysis of the state transition graph. Verification of finite-state systems using this approach has enjoyed considerable success [6].

The CTL formalism can be extended to infinite-state transition systems. Conceptually, the computation trees, which in general contain an uncountable number of paths and an uncountable number of states along each path, can be obtained in the same manner and the same reachability analysis can be applied to verify a CTL formula. The cardinality of the state spaces of such transition systems makes direct reachability analysis impossible, however. A standard approach to verifying properties of an infinite-state transition system is to find a *bisimulation*, which is a finite-state transition system that is equivalent to the original transition system with respect to the verification problem. To find a bisimulation, a so-called *quotient transition system (QTS)* is constructed from a partition of the state space of the original transition system. The QTS is then checked to see if it is a bisimulation of the original transition system. If not, the state space partition is refined, a new QTS is constructed, and the procedure is repeated until a bisimulation is found [9].

This paper addresses two principal difficulties with the bisimulation approach to verification of hybrid dynamic systems. First, finite-state bisimulations exist for only very limited classes of hybrid systems [9]–[12]. Consequently, one cannot guarantee the procedure for computing a bisimulation will terminate for most hybrid systems of practical interest. We avoid this problem by performing the verification on the QTS in each iteration of the bisimulation procedure, rather than waiting for the bisimulation procedure to terminate. Certain specifications can be verified on a QTS, which is a conservative approximation of the infinite-state transition system. By "conservative approximation" we mean that in addition to representing all valid transition sequences for the original infinite-state transition system, the QTS may admit some additional transition sequences that are not valid for the infinite-state transition system. A conservative approximation is called a *simulation* in the transition system literature. As the verification may be inconclusive due to the coarseness of the approximation, the QTS can be refined to yield a more accurate approximation. Although it cannot be guaranteed that the verification question can be resolved using QTSs, there

Manuscript received October 20, 1999; revised October 6, 2000, November 24, 2000, and February 6, 2001. Recommended by Associate Editor K. Rudie. This work was supported in part by DARPA under Contract F33615-97-C-1012 and in part by the Ford Motor Company.

A. Chutinan is with the Ford Research Laboratory, Dearborn, MI 48121-2053 USA (e-mail: achutina@ford.com).

B. H. Krogh is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213-3890 USA (e-mail: krogh@ece.cmu.edu).

Publisher Item Identifier S 0018-9286(01)08801-8.

are cases where verification can be accomplished with this approach even when a finite-state bisimulation does not exist [13]–[15].

Second, any procedure for computing a bisimulation requires the computation of *reachable sets* which are used to define transitions for the QTS. This may not be possible to implement in a computer since finite representations may not exist for arbitrary uncountable sets of states. One may consider using approximations to the reachable sets in the QTS computations. Introducing reachability approximations leads to *approximate quotient transition systems (AQTSs)*. We discuss the implication of using AQTSs in the bisimulation procedure as well as in our verification procedure.

This paper is organized as follows. Section II introduces the concepts of transition systems, quotient transition systems, and bisimulation. Section III discusses the verification of transition systems against a CTL specification using bisimulation and compares the method to our approach which uses QTSs. Section IV discusses the issues that arise when reachability approximation is incorporated into the quotient system verification procedure. The use of the AQTS for verification of hybrid systems is illustrated with an example in Section V. The concluding section summarizes the contributions of this paper and points to a realization of the proposed approach to verification for a class of hybrid systems in a MATLAB-based tool called *CheckMate*.

II. Preliminaries

This section introduces labeled transition systems and summarizes the theory of bisimulation using quotient transition systems for verification of labeled transition systems. The material in this section is adapted from [9], [16]. In our development, *event* labels are dropped from the transition system definition and the simulation relation definition is extended from a single marking to a more general labeling function.

A. Labeled Transition Systems and Bisimulations

Definition 1: A *labeled transition system* T is a tuple $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$ where Q is the set *states*, $\rightarrow \subseteq Q \times Q$ is the set of *transitions*, $Q_0 \subseteq Q$ is the set of *initial states*, $\mathcal{L}$ is a countable set of *labels*, and $L : Q \rightarrow \mathcal{L}$ is the *labeling function*.

Given q and $q' \in Q$, the notation $q \rightarrow q'$ indicates that $(q, q') \in \rightarrow$. We assume that the transition relation $\rightarrow$ is *total*, that is, for every state $q \in Q$ there exists a state $q' \in Q$ such that $q \rightarrow q'$. This assumption allows us to consider only the infinite paths in the labeled transition system. The notations Q^* and Q^ω denote the sets of strings of elements in Q that are finite (but arbitrary) length and infinite length, respectively.

Definition 2: Given a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$, a sequence of states $\pi = q_0 q_1 \cdots \in Q^* \cup Q^\omega$ is called a *path* of T if $q_i \rightarrow q_{i+1}$ for all $i \geq 0$. A path π is called a *run* of T if $q_0 \in Q_0$.

Definition 3: Given a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$, and a set $P \subseteq Q$, the *pre-condition* of P, denoted $Pre(P)$, is defined as $Pre(P) = \{q \in Q \mid \exists p \in P, q \rightarrow p\}$, and the *post-condition* of P, denoted $Post(P)$, is defined as $Post(P) = \{q \in Q \mid \exists p \in P, p \rightarrow q\}$.

Given two labeled transitions systems T_1 and T_2 with the same set of labels, T_2 is said to *simulate* T_1 if for every path of T_1, there is a corresponding path in T_2. This concept of simulation is formalized in the following definition.

Definition 4 [16]: Let $T_1 = (Q_1, \rightarrow_1, Q_{01}, \mathcal{L}, L_1)$ and $T_2 = (Q_2, \rightarrow_2, Q_{02}, \mathcal{L}, L_2)$ be labeled transition systems. A *simulation relation* of T_1 by T_2 is a binary relation $\psi \subseteq Q_1 \times Q_2$ such that:

i) for every $q_1 \in Q_1$, there exists $q_2 \in Q_2$ such that $(q_1, q_2) \in \psi$;
ii) if $(q_1, q_2) \in \psi$ and $q_1 \rightarrow_1 q_1'$, then there exists q_2' such that $q_2 \rightarrow_2 q_2'$ and $(q_1', q_2') \in \psi$;
iii) if $(q_1, q_2) \in \psi$, then $q_1 \in Q_{01} \iff q_2 \in Q_{02}$;
iv) if $(q_1, q_2) \in \psi$, then $L_1(q_1) = L_2(q_2)$.

We say T_2 simulates T_1 by ψ, denoted $T_1 \preceq_\psi T_2$, if ψ is a simulation relation of T_1 by T_2. T_2 is also called a ψ-simulation of T_1. Although Condition iii) may be viewed as a special case of labeling where some states are labeled as "initial states," we choose to keep a separate definition for the initial states to emphasize the fact that a run of the transition system must originate from one of these states. Given a relation $\psi = \{(q_1, q_2) \mid (q_1, q_2) \in Q_1 \times Q_2\}$, we write $\psi^{-1} = \{(q_2, q_1) \mid (q_1, q_2) \in \psi\}$ to denote the inverse relation of ψ.

Definition 5: Let $T_1 = (Q_1, \rightarrow_1, Q_{01}, \mathcal{L}, L_1)$ and $T_2 = (Q_2, \rightarrow_2, Q_{02}, \mathcal{L}, L_2)$ be labeled transition systems. A *bisimulation relation* between T_1 and T_2 is a binary relation $\psi \subseteq Q_1 \times Q_2$ such that $T_1 \preceq_\psi T_2$ and $T_2 \preceq_{\psi^{-1}} T_1$.

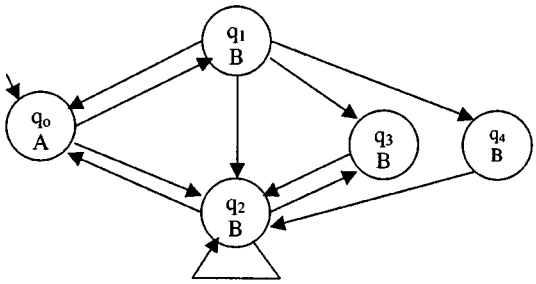
We say the transition systems T_1 and T_2 bisimulate each other by ψ, denoted $T_1 \equiv_\psi T_2$, if ψ is a bisimulation relation between T_1 and T_2.

Example: The definitions above are illustrated by the three labeled transition systems in Fig. 1, T, T_1, and T_2, with labels from $\mathcal{L} = \{A, B\}$ assigned to each state. States of T are associated with states of T_1 and T_2 through relations ψ_1 and ψ_2, respectively, as indicated in the figure. It can be verified that $T \preceq_{\psi_1} T_1$: the two transitions from q_0 (the only state associated with P_0) are to states associated with P_1, and for each state in Q associated with P_1 there is a transition to either another state associated with P_1 (corresponding to the transition $P_1 \rightarrow P_1$ in T_1) or a transition to q_0 (corresponding to the transition $P_1 \rightarrow P_0$ in T_1). The relation ψ_1 is not a bisimulation relation, however, because transition system T does not simulate T_1 under the relation ψ_1^{-1}. This is demonstrated by the pair of states $(P_1, q_4) \in \psi_1^{-1}$: although there is a transition $P_1 \rightarrow P_0$ in T_1, there is no transition from q_4 to q_0 in T. On the other hand, $T \equiv_{\psi_2} T_2$, since it can be shown easily that $T \preceq_{\psi_2} T_2$ and $T_2 \preceq_{\psi_2^{-1}} T$.

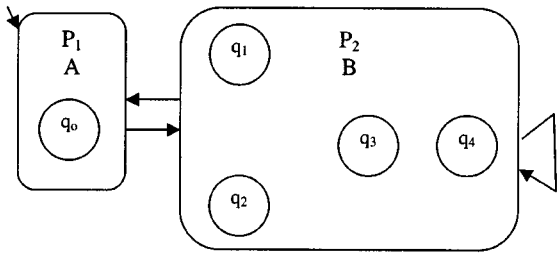
The following definition will be used in the discussion of computational procedures for generating finite-state simulations of a given infinite-state transition system.

Definition 6: If $T_1 \preceq_\psi T_2$, the relation ψ is said to be a *strict simulation relation* of T_1 by T_2, denoted $T_1 \prec_\psi T_2$, if ψ is not a bisimulation relation between T_1 and T_2.

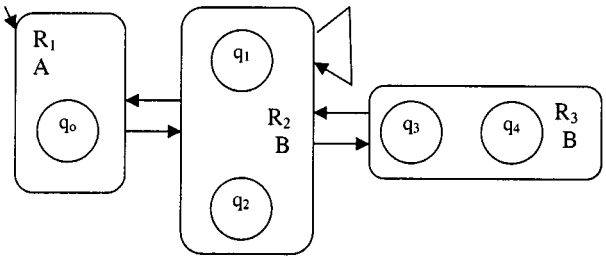
For completeness we include the following two lemmas concerning properties of simulations and bisimulations of transition systems.



(a) Labeled transition system T with states $Q = \{q_0, \ldots, q_4\}$.



(b) Labeled transition system T_1 with states $Q_1 = \{P_0, P_1\}$.



(c) Labeled transition system T_2 with states $Q_2 = \{R_1, R_2, R_3\}$.

Fig. 1. Transition systems (a) T, (b) T_1, (c) T_2. $T \preceq_{\psi_1} T_1$ and $T \equiv_{\psi_2} T_2$, where relations ψ_1, ψ_2 are defined by the partitions of Q indicated in (b) and (c), respectively.

Lemma 1: Let $T_1 = (Q_1, \rightarrow_1, Q_{01}, \mathcal{L}, L_1)$ and $T_2 = (Q_2, \rightarrow_2, Q_{02}, \mathcal{L}, L_2)$ be labeled transition systems and $\psi \subseteq Q_1 \times Q_2$. If $T_1 \preceq_\psi T_2$, then for all $(q, q') \in \psi$, if $\pi = q_0 q_1 \ldots$ is a path of T_1 with $q = q_0$, then there exists a path $\pi' = q'_0 q'_1 \ldots$ of T_2 with $q' = q'_0$ such that $(q_i, q'_i) \in \psi$, $\forall i \geq 0$.

Proof: Straightforward by induction on the length of the path π. ∎

Lemma 2: Let $T_1 = (Q_1, \rightarrow_1, Q_{01}, \mathcal{L}, L_1)$ and $T_2 = (Q_2, \rightarrow_2, Q_{02}, \mathcal{L}, L_2)$ be labeled transition systems and $\psi \subseteq Q_1 \times Q_2$. If $T_1 \equiv_\psi T_2$, then for all $(q, q') \in \psi$, the following conditions hold:

i) if $\pi = q_0 q_1 \ldots$ is a path of T_1 with $q = q_0$, there exists a path $\pi' = q'_0 q'_1 \ldots$ of T_2 with $q' = q'_0$ such that $(q_i, q'_i) \in \psi$, $\forall i \geq 0$;
ii) if $\pi' = q'_0 q'_1 \ldots$ is a path of T_2 with $q' = q'_0$, there exists a path $\pi = q_0 q_1 \ldots$ of T_1 with $q = q_0$ such that $(q_i, q'_i) \in \psi$, $\forall i \geq 0$.

Proof: Follows from Lemma 1. ∎

B. Quotient Transition Systems

An approach to finding a bisimulation of a labeled transition system uses QTSs. A QTS is constructed from a partition of the state space of the underlying transition system. The partition must be consistent with the labeling function and set of initial states in the following sense.

Definition 7: Given a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$, a partition $\mathcal{P}$ of Q is said to be *consistent* if and only if for all $P \in \mathcal{P}$ and for all $q, q' \in P$, $L(q) = L(q')$ and $q \in Q_0 \iff q' \in Q_0$.

In words, all states in the same set in the partition must have the same labels, including the special "initial state" labels.

Definition 8 (Quotient Transition System) [9], [10]: Given a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$ and a consistent partition $\mathcal{P}$ of Q, the *quotient transition system* of T is defined as $T/\mathcal{P} = (\mathcal{P}, \rightarrow_\mathcal{P}, Q_0/\mathcal{P}, \mathcal{L}, L_\mathcal{P})$, where

i) for all $P, P' \in \mathcal{P}$, $P \rightarrow_\mathcal{P} P'$ iff there exist $q \in P$ and $q' \in P'$ such that $q \rightarrow q'$, or equivalently $P \rightarrow_\mathcal{P} P' \iff Post(P) \cap P' \neq \emptyset \iff P \cap Pre(P') \neq \emptyset$;
ii) $Q_0/\mathcal{P} = \{P \in \mathcal{P} \mid P \subseteq Q_0\}$;
iii) for all $P \in \mathcal{P}$, $L_\mathcal{P}(P) = L(q)$ for some $q \in P$.

In words, the a partition state in the QTS can make a transition to another partition state if one of its element state can make a transition to another element state in the destination partition state.

It is easy to see that $T \preceq_\psi T/\mathcal{P}$, where $\psi = \{(q, P) \in Q \times \mathcal{P} \mid q \in P\}$. Since we will refer to the relation $\{(q, P) \in Q \times \mathcal{P} \mid q \in P\}$ quite often, we denote it with a special notation $\Gamma_{Q,\mathcal{P}} = \{(q, P) \in Q \times \mathcal{P} \mid q \in P\}$. The following proposition gives a necessary and sufficient condition for $T/\mathcal{P}$ to be a bisimulation of T with the relation $\Gamma_{Q,\mathcal{P}}$.

Proposition 1: Given a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$ and a consistent partition $\mathcal{P}$ of Q, $T \equiv_{\Gamma_{Q,\mathcal{P}}} T/\mathcal{P}$, if and only if the partition $\mathcal{P}$ satisfies

$$\text{for all } P, P' \in \mathcal{P}, \text{ either } P \cap Pre(P') = \emptyset \text{ or } P \cap Pre(P') = P. \quad (1)$$

Proof: This is a standard result regarding the construction of bisimulations of basic transition systems (see, e.g., [10]). ∎

In words, (1) states that all states in each $P \in \mathcal{P}$ behave uniformly: given another $P' \in \mathcal{P}$, either all states or no state in P can reach some state in P' in one transition. Condition (1)

leads to the general procedure, BP, for computing a bisimulation of a transition system using QTSs shown in Fig. 2.

Procedure BP uses the notion of partition refinement, defined below, to generate a sequence of QTSs that are increasingly less conservative approximations to the behavior of the original system.

Definition 9: Given two partitions $\mathcal{P}_1, \mathcal{P}_2$ of Q, $\mathcal{P}_2$ is said to be a *refinement* of $\mathcal{P}_1$ if and only if for all $P_2 \in \mathcal{P}_2$, there exists $P_1 \in \mathcal{P}_1$ such that $P_2 \subseteq P_1$.

Note that the termination condition of BP in Fig. 2 is precisely condition (1) of Proposition 1. In each iteration of BP, the partition refinement scheme uses the information obtained from $Pre(P')$ to split P into the part that can reach P' and the part that cannot. Also note that the $Pre(\cdot)$ operation in BP is with respect to the transition relation $\rightarrow$ of T, not the transition relation $\rightarrow_{\mathcal{P}}$ of the QTS $T/\mathcal{P}$.

Condition (1) involves only the precondition sets. Using the post-condition sets, the following proposition gives a sufficient condition for $T/\mathcal{P}$ to be a bisimulation of T with the relation $\Gamma_{Q,\mathcal{P}}$.

Proposition 2: Given a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$ and a consistent partition $\mathcal{P}$ of Q, $T \equiv_{\Gamma_{Q,\mathcal{P}}} T/\mathcal{P}$ if the partition $\mathcal{P}$ satisfies

$$\text{for all } P, P' \in \mathcal{P}, \text{ either } Post(P) \cap P' = \emptyset \text{ or } Post(P) \cap P' = Post(P). \quad (2)$$

Proof: Since

i) $Post(P) \cap P' = \emptyset \Rightarrow P \cap Pre(P') = \emptyset$;

ii) $Post(P) \cap P' = Post(P) \Rightarrow Post(P) \subseteq P' \Rightarrow P \subseteq Pre(P') \Rightarrow P \cap Pre(P') = P$, the proposition follows from Proposition 1.

■

Example: The transition systems in Fig. 1 illustrate the definitions and propositions above regarding quotient transition systems. Identifying the states of transition systems T_1 and T_2 with subsets of Q, as indicated in the figure, defines two partitions of Q, $\mathcal{P} = \{\mathcal{P}_\infty, \mathcal{P}_\in\}$ and $\mathcal{R} = \{\mathcal{R}_\infty, \mathcal{R}_\in, \mathcal{R}_\ni\}$. Thus, $T_1 = T/\mathcal{P}$ and $T_2 = T/\mathcal{R}$. Moreover, partiation $\mathcal{R}$ is a refinement of $\mathcal{P}$ that can be obtained by applying procedure BP to $\mathcal{P}$. Also, it can be verified easily that paritition $\mathcal{R}$ satisfies the conditions in Propositions 1 and 2, implying $T \equiv_{\Gamma_{Q,\mathcal{R}}} T/\mathcal{R}$.

Bisimulation relations are important for verification because for labeled transition systems related by a bisimulation relation all properties for the labeled behaviors are identical. Therefore, verification results for the two systems will be identical. This is particularly significant for infinite-state transition systems because, if a finite-state bisimulation exists, procedure BP will terminate with a finite-state transition system that can be used to verify properties of the infinite-state transition system using standard model checking tools.

Bisimulation Procedure (BP) [9],[10]:
set $\mathcal{P} = \mathcal{P}_0$
% check termination condition
while $\exists P, P' \in \mathcal{P}$ such that $\emptyset \neq P \cap Pre(P') \neq P$ {
  % partition refinement
  split P into $P_1 = P \cap Pre(P')$ and $P_2 = P \setminus Pre(P')$
  set $\mathcal{P} = (\mathcal{P} \setminus \{P\}) \cup \{P_1, P_2\}$
}

Fig. 2. Bisimulation procedure.

III. Verification of Labeled Transition Systems

A. Computation Tree Logic

CTL is a well established specification formalism for finite-state transition systems [5], [6]. The specification is formulated by attaching a finite set of *atomic propositions* AP whose individual values are either true or false to each state in the transition system. Thus, we consider a labeled transition system of the form $T = (Q, \rightarrow, Q_0, 2^{AP}, L)$ where $L : Q \rightarrow 2^{AP}$ is the labeling function which assigns to each state in Q a set of atomic propositions which are true for that state. In the CTL literature, the labeled transition system is often called a *Kripke structure* [5].

By selecting a state in the labeled transition system as the initial state and unfolding the state transition graph of the transition system, we obtain an infinite tree, called a *computation tree*, with the selected initial state at the root [6]. A CTL formula specifies the system evolutions in terms of the atomic propositions along some or all paths in the computation tree of the labeled transition system.

In this paper, we consider a class of CTL formulae called ACTL [7], which allows only universal assertions, i.e., the assertions that must hold for *all* paths in the computation tree. An ACTL formula consists of *temporal operators* and the *universal path quantifier* $\mathbf{A}$. Temporal operators specify the system evolution along a single path of the computation tree. There are four basic temporal operators, $\mathbf{X}$, $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{U}$. The operator $\mathbf{X}$ ("next time") asserts that a property holds in the next state in the path. The operator $\mathbf{F}$ ("in the future") asserts that a property holds some future state along the path (including the current state). The operator $\mathbf{G}$ ("globally") asserts that a property holds globally, i.e., at every state along the path. The operator $\mathbf{U}$ ("until") involves two properties: the assertion $f\ \mathbf{U}\ g$ requires that g holds as some state in the path (including the current state) and that f holds at every state along the path prior to the occurrence of g. The universal path quantifier $\mathbf{A}$ specifies that the path assertion holds along *all* paths in the computation tree. The syntax of an ACTL formula f, in *positive normal form* is given by the grammar

$$f \rightarrow ap \mid \neg ap \mid f \vee f \mid f \wedge f \mid \mathbf{AX}f \mid \mathbf{AF}f \mid \mathbf{AG}f \mid \mathbf{A}f\mathbf{U}f.$$

where ap denotes an atomic proposition. We write $T, q \models f$ to mean that the ACTL formula f holds at state q in T. T is usually dropped from this notation when the labeled transition system is clear from the context.

An ACTL formula for a labeled transition system corresponds to a region (a set of states) in the state space of the transition system for which the ACTL formula holds. It has been shown that the region corresponding to an ACTL formula can be computed using fixed-point iterations in [6]. For a finite-state labeled transition system, these fixed-point computations are guaranteed to terminate. The labeled transition system satisfies the ACTL specification if all initial states are included in the region corresponding to the ACTL specification.

ACTL formulas can be used to specify properties of the system such as

- **AG** safe. The system is safe at all time;
- **AG** (**AF** reset). The system is reset infinitely often;
- **AF** (**AG** safe). The system will eventually reach a safe region and stay in the region forever;

Using the CTL framework, the *verification problem* can be stated as follows. Given a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$ and a CTL specification f, determine whether or not $T, q \models f$ for all $q \in Q_0$.

B. Verification Using QTSs

In the bisimulation approach to transition system verification, the procedure BP in Fig. 2 is applied to find a partition of the state space that will give a finite-state bisimulation of the underlying infinite-state transition system. In each iteration of BP, a QTS is constructed from the current partition. BP tests if the QTS is a bisimulation of the underlying transition system. If so, BP stops, otherwise the partition is refined and BP repeats until a bisimulation is found. The approach requires that a bisimulation be found before the verification step can be performed. This is illustrated in Fig. 3(a).

Since QTSs are, in general, simulations of the underlying transition system, one could attempt to perform verification on the QTS in any iteration of BP rather than waiting for BP to terminate. If the property is verified, there would be no need to refine the QTS further. If not, another refinement iteration can be executed and verification can be attempted on the new QTS. Continuing this process, it is sometimes possible to conclude whether or not the underlying transition system satisfies the desired property before a bisimulation is achieved or even if a bisimulation does not exist [13]–[15]. Fig. 3(b) depicts this alternative verification approach using QTSs. The restriction here is that since QTSs are conservative approximations of the underlying transition system, only *universal* properties can be verified. In the context of CTL, a universal property is a property that can be specified as an ACTL formula. This is formalized in the following proposition.

Proposition 3: Let $T_1 = (Q_1, \rightarrow_1, Q_{01}, 2^{AP}, L_1)$ and $T_2 = (Q_2, \rightarrow_2, Q_{02}, 2^{AP}, L_2)$ be labeled transition systems and f be an ACTL formula. If $T_1 \preceq_\psi T_2$, then for all $(q, q') \in \psi, (T_2, q' \models f) \Rightarrow (T_1, q \models f)$.

Proof: We prove this by induction on the length of the ACTL formula f.

Basis Step: ACTL formulas of the shortest length are ap and $\neg ap$ where $ap \in AP$. Since $(q, q') \in \psi$, we have that $L_1(q) = L_2(q')$ by Condition iv) of Definition 4. Thus, $T_2, q' \models ap \iff T_1, q \models ap$ and the proposition holds for both ap and $\neg ap$.

Induction Step: Suppose the proposition holds for all subformulae g and h, we show that the proposition also holds for the formula f produced by any production rules in the ACTL grammar.

i) $f = g \vee h$. Suppose that $T_2, q' \models f$. Then $T_2, q' \models g$ or $T_2, q' \models h$. By the inductive assumption, we have that $T_1, q \models g$ or $T_1, q \models h$. Consequently, $T_1, q \models f$.

ii) $f = g \wedge h$. This follows by a similar argument to Case i).

For the rest of the production rules for f, suppose that $T_2, q' \models f$. For any path $q_0 q_1 \ldots$ of T_1 starting with $q_0 = q$, there exists a path $q'_0 q'_1 \ldots$ in T_2 starting with $q'_0 = q'$ such that $(q_i, q'_i) \in \psi, \forall i \geq 0$ by Lemma 1.

iii) $f = \mathbf{A} g \mathbf{U} h$. Since $T_2, q'_0 \models f$, there exists a $k \geq 0$ such that $T_2, q'_k \models h$ and $T_2, q'_j \models g$ for all $0 \leq j < k$. By the inductive assumption, we have that $T_1, q_k \models h$ and $T_1, q_j \models g$ for all $0 \leq j < k$. Thus, $T_1, q_0 \models f$.

iv) $f = \mathbf{AF} g$. This follows because $f = \mathbf{A}\ \mathit{true}\ \mathbf{U} g$ and, therefore, is a special case of iii).

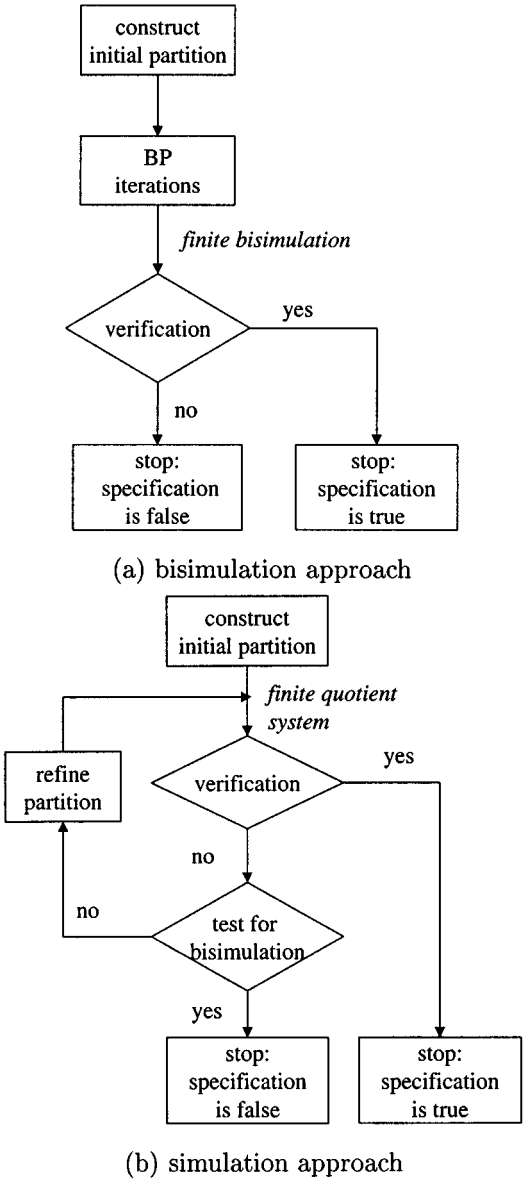


Fig. 3. Verification approaches using quotient transition systems that are (a) bisimulations and (b) simulations of the original system.

v) $f = \mathbf{AG}g$. Since $T_2, q_0' \models f$, we have that $T_2, q_k' \models g$ for all $k \geq 0$. By the inductive assumption, $T_1, q_k \models g$ for all $k \geq 0$. Thus, $T_1, q_0 \models f$.

vi) $f = \mathbf{AX}g$. Since $T_2, q_0' \models f$, we have that $T_2, q_1' \models g$. By the inductive assumption, $T_1, q_1 \models g$. Thus, $T_1, q_0 \models f$.

■

It follows from Proposition 3 that given $T/\mathcal{P}$, if $P \models f$ for all $P \in Q_0/\mathcal{P}$, then $T, q \models f$ for all $q \in Q_0$. Thus, if $T/\mathcal{P}$ satisfies the ACTL specification f, we can conclude that T also satisfies the ACTL specification. However, the converse is not true unless $T/\mathcal{P}$ is a bisimulation of T.

Fig. 4 shows the *quotient transition system procedure (QTSP)* based on the approach in Fig. 3(b). The procedure is basically a modified version of BP with the verification step moved inside each iteration of BP. The relocation of the verification step is not the only change made to BP, however, as three other aspects of BP are also relaxed in QTSP.

First, as noted in Section II-B, the termination condition (1) serves to guarantee that the QTS is a bisimulation of the underlying transition system. In practice, it might not be convenient to test (1), for example, if the exact pre-condition sets are not computable. Thus, one may consider replacing (1) with some weaker condition that also implies the QTS is a bisimulation but only relies on the approximation of the pre-condition and the post-condition sets. (An example of such condition is given in Section IV.) The generic condition which guarantees a bisimulation is referred to in QTSP as the *bisimulation_termination_condition*.

Second, we observe that any refinement method, referred to in QTSP generically as the *refinement_method*, can be used to refine the QTS. Different refinement methods may result in different rates of convergence to a bisimulation (if a bisimulation exists). The refinement method used in BP has the desirable property that it produces a QTS that is strictly better than the previous QTS. This result is stated and proved formally as Lemma 3 at the end of this section. It may be difficult or even impossible, however, to implement the BP refinement method as one has to represent and manipulate arbitrary sets of infinite states. Thus, it may be necessary to use a simpler alternative for refining the QTS when the verification fails.

Third, since the complexity of the QTS computation depends largely on the size of the system itself, one should also try to limit the growth in the number of states resulting from the partition refinement in each iteration of the bisimulation procedure. To slow down the state explosion resulting from the partition refinement, we refine only the states in the current QTS that are relevant to the ACTL specification. Recall that in the process of an ACTL verification, one obtains the set of initial states in the current QTS that satisfy the ACTL specification. Since the specification holds for all computation paths, the verification result cannot be improved by refining these states and their descendants. Thus, one should only refine the initial states that do not satisfy the ACTL specification and their descendants. In QTSP, *TBR* is the set of states "to be refined" in each iteration. It consists of the set *REFINE*, the states that should be refined from the bisimulation requirement, and the set *reach* $((\mathcal{P} \setminus SPEC) \cap Q_0/\mathcal{P}_N)$, the states that should be refined from the ACTL specification requirement.

Quotient Transition System Procedure (QTSP):

```
initialize N = 0 and P_N = P_0
repeat for the partition P_N
    compute T/P_N (or approximate, see Section IV)
    compute SPEC = {P ∈ P_N | P satisfies ACTL spec for T/P_N}
    if Q_0/P_N ⊆ SPEC
        stop = 1 % specification is satisfied
    else
        compute REFINE = {P ∈ P_N | ∃P' ∈ P_N such that (P, P')
            violates bisimulation_termination_condition}
        if REFINE == ∅
            stop = 1 % bisimulation obtained and specification is false
        else
            compute
                TBR = reach((P_N \ SPEC) ∩ Q_0/P_N) ∩ REFINE
            if TBR == ∅
                stop = 1 % no state worth refining and spec is false
            else
                stop = 0 % refine partition
                P_{N+1} = P_N
                for each P ∈ TBR
                    split P using refinement_method into P_1, P_2
                        such that P_1 ∪ P_2 = P and P_1 ∩ P_2 = ∅
                    set P_{N+1} = (P_{N+1} \ {P}) ∪ {P_1, P_2}
                endfor
                N = N + 1
            endif
        endif
    endif
until (stop == 1)
```

Fig. 4. Verification procedure using QTSs.

The following lemma demonstrates that the refinement procedure in BP always generates a QTS that is strictly less conservative than the previous QTS. This lemma uses notation for a relation similar to the relation $\Gamma_{Q,\mathcal{P}}$. In particular, given partitions $\mathcal{P}_2$ and $\mathcal{P}_1$ where $\mathcal{P}_2$ is a refinement of $\mathcal{P}_1$, we define $\Gamma_{\mathcal{P}_2,\mathcal{P}_1} = \{(P_2, P_1) \in \mathcal{P}_2 \times \mathcal{P}_1 \mid P_2 \subseteq P_1\}$.

Lemma 3: Suppose $\mathcal{P}_N$ is a partition of Q and there exist $P, P' \in \mathcal{P}_N$ such that $\emptyset \neq P \cap Pre(P') \neq P$. If $\mathcal{P}_{N+1} = [\mathcal{P}_N \setminus \{P\}] \cup \{P_1, P_2\}$ where $P_1 = P \setminus Pre(P')$ and $P_2 = P \cap Pre(P')$, then $T/\mathcal{P}_{N+1} \prec_{\Gamma_{\mathcal{P}_{N+1},\mathcal{P}_N}} T/\mathcal{P}_N$.

Proof: Since $\mathcal{P}_{N+1}$ is a refinement of $\mathcal{P}_N$, we have that $T/\mathcal{P}_{N+1} \preceq_{\Gamma_{\mathcal{P}_{N+1},\mathcal{P}_N}} T/\mathcal{P}_N$. Thus, we only need to show that $\Gamma^{-1}_{\mathcal{P}_{N+1},\mathcal{P}_N}$ is not a simulation relation of $T/\mathcal{P}_N$ by $T/\mathcal{P}_{N+1}$. Consider the pair (P, P_1) as defined in the lemma. The assumptions imply that $(P, P_1) \in \Gamma^{-1}_{\mathcal{P}_{N+1},\mathcal{P}_N}$ and $P \rightarrow_{\mathcal{P}_N} P'$. We make two observations. First, since $P_1 \cap Pre(P') = \emptyset$, we have $P_1 \not\rightarrow P'$. Second, since P' is not refined, there is no $\hat{P} \in \mathcal{P}_{N+1}$ other than P' such that $(P', \hat{P}) \in \Gamma^{-1}_{\mathcal{P}_{N+1},\mathcal{P}_N}$. These two observations imply is no $\hat{P} \in \mathcal{P}_{N+1}$ such that $P_1 \rightarrow_{\mathcal{P}_{N+1}} \hat{P}$ and $(P', \hat{P}) \in \Gamma^{-1}_{\mathcal{P}_{N+1},\mathcal{P}_N}$. ■

IV. Approximate QTSPs

The major obstacle toward applying the quotient transition system procedure (QTSP) or the bisimulation procedure (BP) to the transition system representing any real dynamic system is the lack of effective methods for computing and representing the pre- and post-condition sets. This section concerns the problems

that arise when these sets can only be approximated. We will refer to any method M used to compute and represent pre- and post-condition sets for a transition system as a *reachability approximation method*, and denote the pre- and post-condition sets for a set of states P computed using method M by $Pre^M(P)$ and $Post^M(P)$, respectively.

Definition 10: A reachability approximation method M for a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$ is called *conservative* if for all $P \subseteq Q$, $Pre(P) \subseteq Pre^M(P)$ and $Post(P) \subseteq Post^M(P)$.

The development of effective reachability approximation methods has been a major issue in recent hybrid systems research [17]–[21]. Without exact pre- and post-condition sets, one can only approximate the $T/\mathcal{P}$ for a partition $\mathcal{P}$. The approximate pre- and post-condition sets or their combinations can be used to define a variety of the *approximate quotient transition systems (AQTSs)*. We give two examples of the AQTSs in the following definitions.

Definition 11: For a reachability approximation method M for a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$, the *post-$M$ approximate quotient transition system* for T given a consistent partition $\mathcal{P}$ of Q is defined as

$$\frac{T^M}{\mathcal{P}} = \left(\mathcal{P}, \rightarrow_{\mathcal{P}}^M, \frac{Q_0}{\mathcal{P}}, \mathcal{L}, L_{\mathcal{P}}\right)$$

where $Q_0/\mathcal{P}$ and $L_{\mathcal{P}}$ are defined as in Definition 8 and $\rightarrow_{\mathcal{P}}^M$ is defined as follows: for all $P, P' \in \mathcal{P}$, $P \rightarrow_{\mathcal{P}}^M P'$ iff $Post^M(P) \cap P' \neq \emptyset$.

Definition 12: For a reachability approximation method M for a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$, the *pre-$M$ approximate quotient transition system* for T given a consistent partition $\mathcal{P}$ of Q is defined as

$$\frac{{}^M T}{\mathcal{P}} = \left(\mathcal{P}, \rightarrow_{\mathcal{P}}^M, \frac{Q_0}{\mathcal{P}}, \mathcal{L}, L_{\mathcal{P}}\right)$$

where $Q_0/\mathcal{P}$ and $L_{\mathcal{P}}$ are defined as in Definition 8 and $\rightarrow_{\mathcal{P}}^M$ is defined as follows: for all $P, P' \in \mathcal{P}$, $P \rightarrow_{\mathcal{P}}^M P'$ iff $P \cap Pre^M(P') \neq \emptyset$.

The following lemma is easily proved.

Lemma 4: Given a conservative reachability approximation method M, $T \preceq_{\Gamma_{Q,\mathcal{P}}} T^M/\mathcal{P}$ and $T \preceq_{\Gamma_{Q,\mathcal{P}}} {}^M T/\mathcal{P}$.

The above lemma follows from the fact that $T/\mathcal{P} \preceq_{\Gamma_{\mathcal{P},\mathcal{P}}} T^M/\mathcal{P}$ and $T/\mathcal{P} \preceq_{\Gamma_{\mathcal{P},\mathcal{P}}} {}^M T/\mathcal{P}$ when M is conservative and that $T \preceq_{\Gamma_{Q,\mathcal{P}}} T/\mathcal{P}$. Thus, when M is conservative, we can use either $T^M/\mathcal{P}$ or ${}^M T/\mathcal{P}$ to verify ACTL specifications as shown in Proposition 3. In this paper, however, we only consider using the post-M AQTS.

Lemma 5: Given a conservative reachability approximation method M, if $T \prec_{\Gamma_{Q,\mathcal{P}}} T/\mathcal{P}$, then $T \prec_{\Gamma_{Q,\mathcal{P}}} T^M/\mathcal{P}$ and $T \prec_{\Gamma_{Q,\mathcal{P}}} {}^M T/\mathcal{P}$.

Proof: Since T/P is not a bisimulation of T with respect to $\Gamma_{Q,\mathcal{P}}$, condition (1) implies that there exists $P, P' \in \mathcal{P}$ such that $\emptyset \neq Pre(P') \cap P \neq P$. We show for this case that $\Gamma_{Q,\mathcal{P}}^{-1}$ is not a simulation relation of $T^M/\mathcal{P}$ by T. From $Pre(P') \cap P \neq P$, we have that there exists $q \in P$ such that $q \notin Pre(P')$, i.e., there is no $q' \in P'$ such that $q \rightarrow q'$. From $Pre(P') \cap P \neq \emptyset$, we have that $Post(P) \cap P' \neq \emptyset$, which implies that $Post^M(P) \cap P' \neq \emptyset$ and, consequently, $P \rightarrow_{\mathcal{P}}^M P'$. Recalling that $(P, q) \in \Gamma_{Q,\mathcal{P}}^{-1} \iff q \in P$, we have from the above results that Condition (ii) of Definition 4 is violated that because $(P, q) \in \Gamma_{Q,\mathcal{P}}^{-1}$ and $P \rightarrow_{\mathcal{P}}^M P'$ but there is no q' such that $q \rightarrow q'$ and $(P', q') \in \Gamma_{Q,\mathcal{P}}^{-1}$. Similar argument follows for ${}^M T/\mathcal{P}$. ■

In words, the above lemma states that if a QTS $T/\mathcal{P}$ is not a bisimulation of T, then neither is any conservative approximation of $T/\mathcal{P}$. This result makes intuitive sense because conservativeness generally leads to reduced approximation accuracy.

Replacing the computation of the pre-condition sets, post-condition sets, and QTS in BP by their corresponding approximations in the method M, the bisimulation condition (1) becomes

$$\text{for all } P, P' \in \mathcal{P}, \text{ either } P \cap Pre^M(P') = \emptyset \text{ or } P \cap Pre^M(P') = P. \tag{3}$$

Note that condition (3) does not imply that the post-M QTS $T^M/\mathcal{P}$ is a bisimulation of T relative to relation $\Gamma_{Q,\mathcal{P}}$ even when M is conservative. To see this, consider the following counter example for finite-state transition system. Let T be a transition system with $Q = \{a, b, c\}$ and $\rightarrow = \{(a, c), (b, a), (b, b), (c, c)\}$. Partition Q by $\mathcal{P} = \{P_1, P_2\}$ where $P_1 = \{a, b\}$ and $P_2 = \{c\}$. Consequently, $\emptyset \neq P_1 \cap Pre(P_2) = \{a, b\} \cap \{a, c\} = \{a\} \neq P_1$. Clearly, the partition $\mathcal{P}$ violates condition (1) and, therefore, $T \prec_{\Gamma_{Q,\mathcal{P}}} T/\mathcal{P}$. It then follows from Lemma 5 that $T^M/\mathcal{P}$ is not a bisimulation of T independent of any conservative approximation method M.

If the approximation method M possesses the property in the following proposition, it is possible to obtain a sufficient condition for the post-M QTS to be a bisimulation T.

Proposition 4: Given a conservative reachability approximation method M for a labeled transition system $T = (Q, \rightarrow, Q_0, \mathcal{L}, L)$, the condition

$$\text{for all } P, P' \in \mathcal{P}, \text{ either } Post^M(P) \cap P' = \emptyset \text{ or } Post^M(P) \cap P' = Post^M(P) \tag{4}$$

for the partition $\mathcal{P}$ of Q implies $T \equiv_{\Gamma_{Q,\mathcal{P}}} T^M/\mathcal{P}$.

Proof: Since $T \preceq_{\Gamma_{Q,\mathcal{P}}} T^M/\mathcal{P}$ for the conservative approximation method M, it only remains to show that given condition (4), $T^M/\mathcal{P} \preceq_{\Gamma_{Q,\mathcal{P}}^{-1}} T$. Conditions i), iii), and iv) of Definition 4, follow from the fact that $\mathcal{P}$ is a partition of Q and that $\mathcal{P}$ consistent. We show that Condition ii) is satisfied as follows.

Suppose $(P, q) \in \Gamma_{Q,\mathcal{P}}^{-1}$ and $P \rightarrow_{\mathcal{P}}^M P'$. Thus, we have that $q \in P$ and $Post^M(P) \cap P' \neq \emptyset$. It then follows from (4) that $Post^M(P) \cap P' = Post^M(P)$, which implies that $Post^M(P) \subseteq P'$. Consequently, $Post(P) \subseteq P'$. Since $q \in P$ and $\rightarrow$ is total, there exists $q' \in P'$ such that $q \rightarrow q'$. Thus, we have the desired result that $q \rightarrow q'$ and $(P', q') \in \Gamma_{Q,\mathcal{P}}^{-1}$. ■

Proposition 4 suggests that we may replace (1) with (4) and use the AQTS in BP or QTSP. Since approximations are used, there may be cases where the procedure with the exact QTS terminates but the procedure with the approximate QTS does not. This can be seen from the following example. Let T be a transition system with

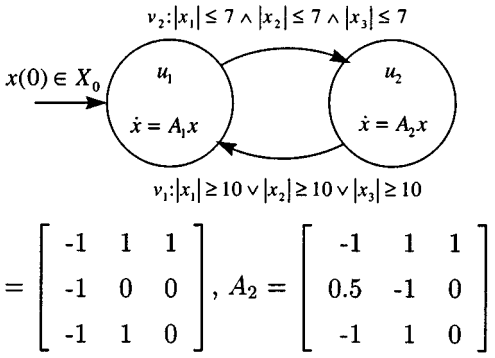


Fig. 5. Example hybrid system.

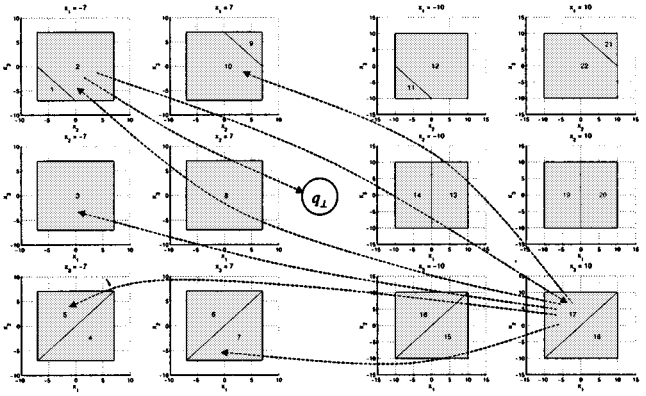


Fig. 6. Initial partition.

$Q = \{a, b, c\}$ and $\rightarrow = \{(a,b),(a,c),(b,a),(b,c),(c,c)\}$. Let $\mathcal{P} = \{P_1, P_2\}$ where $P_1 = \{a, b\}$ and $P_2 = \{c\}$. Since $P_1 \cap Pre(P_2) = \{a, b\} \cap \{a, b, c\} = P_1$ and $P_2 \cap Pre(P_1) = \{c\} \cap \{a, b\} = \emptyset$, condition (1) is satisfied and therefore a procedure with exact QTS terminates with $\mathcal{P}$. For AQTS, suppose the approximation method yields $Post^M(P_1) = \{a, b, c\}$. We have $\emptyset \neq Post^M(P_1) \cap P_2 = \{a, b, c\} \cap \{c\} = \{c\} \neq Post^M(P_1)$. Therefore, condition (4) is not satisfied and the procedure with AQTS does not terminate with $\mathcal{P}$ (the procedure will always terminate when every element in the state partition is a singleton set for our finite-state example, of course). Despite the above drawback, we can at least guarantee that the AQTS $T^M/\mathcal{P}$ that we have is indeed a bisimulation of T upon termination of the procedure. Note that (4) simply states that each state P of the AQTS $T^M/\mathcal{P}$ has at most one successor state.

V. Example

This section demonstrates the AQTS computations for the verification of a simple hybrid system (a *hybrid automaton*) shown in Fig. 5. General hybrid automata are defined in [1] (see also [3]), and the particular class of hybrid automata from which this example is taken, namely, *polyhedral invariant hybrid systems*, is defined in [22]. The definition of the hybrid automaton for this example can be explained informally by referring to Fig. 5. The possible values for the discrete state of the hybrid automaton are represented by the circles labeled u_i, $i = 1, 2$, often referred to as *locations* in the hybrid systems literature. The continuous state $x \in R^3$ evolves according to the linear dynamic state equations in each location, $\dot{x} = A_i x$, $i = 1, 2$. A discrete-state transition occurs immediately when the continuous state satisfies the condition (called the *guard*) on the outgoing arc from the current location. In this case the guards, labeled ν_1, ν_2, are defined by simple inequalities on the components of the continuous state. The continuous state retains its value when a discrete-state transition occurs. The initial continuous state is assumed to be selected from the set $X_0 \subseteq R^3$ and the initial discrete state is u_1.

We are interested in the behavior of the system as it is characterized by the values of the discrete and continuous state when discrete state transitions occur. Therefore, the state space Q for the associated transition system in this example is defined as $Q = X_o \cup X_{\text{entry}} \cup \{q_{u_1}^\perp, q_{u_2}^\perp\}$, where X_{entry} is the set of continuous states on the boundaries of the guard sets and $q_{u_1}^\perp, q_{u_2}^\perp$ are special states introduced to indicate that the system remains in the respective location indefinitely. The transitions relation for the transition system is defined by the mappings between states in Q defined by the continuous trajectories that connect the system states from one discrete-state transition to the next discrete-state transition. The transition system defined by the state space and transition relation described above is an example of a *discrete-trace transition system* defined formally in [22] for polyhedral invariant hybrid systems.

The objective is to verify that the hybrid automaton eventually enters and remains in the location u_2 indefinitely from any initial state. The ACTL expression for this specification is $\mathbf{AF}(\mathbf{AG}\ (u == u_2))$ where u denotes the discrete state (the location) of the hybrid system. We note that it is not possible to apply procedure BP to this example because there are no computational tools available for computing or representing the exact reachable sets required to construct QTSs for hybrid systems with linear continuous dynamics. In other words, only reachability approximation methods are available for such systems.

The AQTSs are constructed from the partitions of the switching surfaces, the 3-D cubes $\|x\|_\infty = 7$ and $\|x\|_\infty = 10$. The initial partition is shown in Fig. 6. The state transitions in the initial AQTS are computed using the reachability approximations method called the *flow pipe approximations* [15], [23]. A transition to one of the states $q_{u_1}^\perp, q_{u_2}^\perp$ is defined if the flow pipe converges to the equilibrium $x = 0$ without going through any switching surfaces completely. This corresponds to the situation where the flow pipe segment at a certain time can be contained in a Lyapunov stability ellipsoid which lies inside all the switching surfaces. It is clear that we can stop the computation at this point since all future state trajectories will remain inside the ellipsoid and will not trigger any future discrete transition.

Fig. 6 shows the partial transitions in the initial AQTS. As seen in Fig. 6, the initial AQTS violates the specification as it contains cycles, indicating the possibility of not entering the location u_2. The AQTS is then refined by bisecting every polytope in the partition that violates the specification by being a part of a cycle. After 3 refinements, we have the partition shown in Fig. 7 and the AQTS satisfies the specification since the AQTS eventually reach the equilibrium state $q_\perp$ from any initial state. For example, all paths from state 2 eventually ends in $q_\perp$ as shown by the partial transitions in Fig. 7.

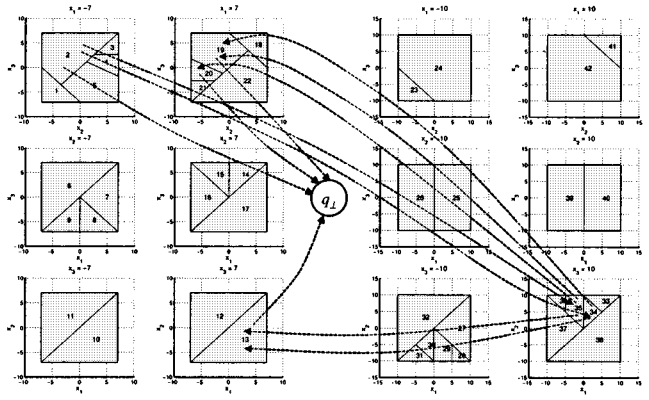


Fig. 7. Partition after 3 refinements.

VI. Discussion

This paper suggests the use of quotient transition systems as an alternative to bisimulations for verification of infinite-state transition systems. This approach has been applied successfully to some case studies of *hybrid systems* [13]–[15], which may be view as infinite-state transition systems, where finite-state bisimulations clearly do not exist. The paper also addresses the issues that arise when exact reachability computation is not possible for the computation of the QTSs and discusses the implication of the reachability approximations in the verification process. We obtain a sufficient condition for an approximate quotient transition system to be a bisimulation of the underlying transition system. The interested reader is referred to [15], [23] for more information on the computation techniques for reachability approximation for hybrid systems and their recent extensions.

The verification approach and the theoretical results presented in this paper serve as the foundation for *CheckMate*, a MATLAB-based verification tool for hybrid systems [24]. Information on *CheckMate* can be found at the web site [25]. Implementations of the complete verification algorithms and examples are also available at this web site.

Acknowledgment

The authors thank the reviewers for several helpful comments and suggestions.

References

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

[2] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi, "Algorithmic analysis of nonlinear hybrid systems," *IEEE Trans. Automat. Contr.*, vol. 43, pp. 540–554, Apr. 1998.

[3] R. Alur, T. A. Henzinger, and P.-H. Ho, "Automatic symbolic verification of embedded systems," *IEEE Trans. Software Eng.*, vol. 22, pp. 181–201, Mar. 1996.

[4] Z. Manna and A. Pnueli, *Temporal Verification of Reactive Systems*. New York: Springer-Verlag, 1995.

[5] E. M. Clarke, O. Grumberg, and D. Peled, *Model Checking*. Cambridge, MA: MIT Press, 2000.

[6] E. M. Clarke, O. Grumberg, and D. Long, "Verification tools for finite-state concurrent systems," in *Proc. Decade Concurrency: Reflections Perspectives*, June 1994.

[7] O. Grumberg and D. E. Long, "Model checking and modular verification," *ACM Trans. Prog. Lang. Syst.*, vol. 16, no. 3, pp. 843–871, May 1994.

[8] E. M. Clarke, E. A. Emerson, and A. P. Sistla, "Automatic verification of finite-state concurrent systems using temporal logic pecifications," *ACM Trans. Prog. Lang. Syst.*, vol. 8, no. 2, pp. 244–263, Apr. 1986.

[9] G. Lafferriere, G. J. Pappas, and S. Yovine, "A new class of decidable hybrid systems," in *Hybrid Systems: Computation Control, 2nd Int. Workshop, HSCC'99*, F. W. Vaandrager and J. H. Van Schuppen, Eds., 1999, pp. 137–151.

[10] T. A. Henzinger, "Hybrid automata with finite bisimulations," in *ICALP 95: Automata, Languages, and Programming*, Z. Fülöp and F. Gécseg, Eds., 1995, pp. 324–335.

[11] J. S. Miller, "Decidability and complexity results for timed automata and semi-linear hybrid automata," in *Hybrid Systems: Computation and Control*, N. Lynch and B. H. Krogh, Eds. New York: Springer-Verlag, 2000, vol. 1790, LNCS, pp. 296–309.

[12] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?," *J. Comput. Sci.*, vol. 57, no. 1, pp. 94–124, 1998.

[13] A. Chutinan and B. H. Krogh, "Computing approximating automata for a class of hybrid systems," *Math. Modeling Syst.: Special Issue Discrete Event Models Continuous Syst.*, vol. 6, pp. 30–50, Mar. 2000.

[14] ——, "Computing approximating automata for a class of linear hybrid systems," in *Hybrid Systems V*. New York: Springer-Verlag, 1998, Lecture Notes in Computer Science.

[15] ——, "Computing polyhedral approximations to dynamic flow pipes," in *Proc. 37th IEEE Conf. Decision Control*, 1998.

[16] G. Barrett and S. Lafortune, "Bisimulation, the Supervisory Control Problem, and Strong Model Matching for Finite State Machines," System Science and Engineering Division, Department of Electrical Engineering and Computer Science, The University of Michigan, CGR-97-05, 1997.

[17] A. Puri, P. Varaiya, and V. Borkar, "ϵ-Approximation of differential inclusions," in *Hybrid Systems III: Verification and Control*, R. Alur, T. A. Henzinger, and E. D. Sontag, Eds: Springer-Verlag, 1996, pp. 362–376.

[18] J. Preußig, O. Stursberg, and S. Kowalewski, "Reachability analysis of a class of switched continuous systems by integrating rectangular approximation and rectangular analysis," in *Hybrid Systems: Computation and Control, 2nd International Workshop, HSCC'99*, F. W. Vaandrager and J. H. Van Schuppen, Eds. Berg en Dal, The Netherlands: Springer-Verlag, 1999, pp. 209–222.

[19] T. Dang and O. Maler, "Reachability analysis via face lifting," in *Hybrid Systems: Computation and Control*. New York: Springer-Verlag, 1998.

[20] A. B. Kurzhanski and P. Varaiya, "Ellipsoidal techniques for reachability analysis," in *Hybrid Systems: Computation and Control-HSCC'00*, N. Lynch and B. H. Krogh, Eds: Springer, 2000, vol. 1790, LNCS, pp. 202–214.

[21] M. R. Greenstreet and I. Mitchell, "Reachability analysis using polygonal projections," in *Hybrid Systems: Computation and Control*: Springer, 1999, vol. 1569, LNCS, pp. 103–116.

[22] A. Chutinan and B. H. Krogh, "Verification of polyhedral-invariant hybrid automata using polygonal flow pipe approximations," in *Hybrid Systems: Computation Control, 2nd Int. Workshop, HSCC'99*, F. W. Vaandrager and J. H. Van Schuppen, Eds., 1999, pp. 76–90.

[23] ——, "New results on computational techniques for hybrid system verification," IEEE Trans. Automat. Contr., to be published.

[24] I. Silva and B. H. Krogh, "Formal verification of hybrid systems using CheckMate: A case study," in *2000 Amer. Control Conf.*, June 2000.

[25] , http://www.ece.cmu.edu/~krogh/CheckMate/main.htm.

Alongkrit Chutinan received the B.S., M.S., and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 1995, 1996, and 1999, respectively. His Ph.D. research topic was in the area of dynamical systems and control with emphasis on *hybrid systems verification*.

After graduation, he joined Ford Research Laboratory, Dearborn, MI, where he developed and applied computer tools to model-based analysis and design of automotive powertrain systems. He is currently with Emmeskay, Inc., Plymouth, MI, developing software for clients in the automotive industry.

Bruce H. Krogh (S'82–M'82–SM'92–F'98) received the B.S. degree in mathematics and physics from Wheaton College, Wheaton, IL, and the Ph.D. degree in electrical engineering from the University of Illinois, Urbana, in 1975 and 1983, respectively.

In 1983, he joined Carnegie Mellon University, Pittsburgh, PA, where he is currently a Professor of electrical and computer. He served on the editorial board of *Discrete Event Dynamic Systems: Theory and Applications*. His research interests include synthesis and verification of control algorithms and software for discrete event and hybrid dynamic systems.

Dr. Krogh served on the editorial board of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, and was founding Editor-in-Chief of the IEEE TRANSACTIONS ON CONTROL SYSTEMS.