Supervisory Control for Rectangular Hybrid Automata

Michael P. Spathopoulos

System Dynamics and Control
Depart. of Mechanical Eng.,
University of Strathclyde
75 Montrose Street
Glasgow G1 1XJ
Scotland
e-mail:mps@mecheng.strath.ac.uk

Abstract: We consider the problem of supervisory control for compact rectangular automata with uniform rectangular activity, i.e. initialised. The supervisory controller is state feedback and can disable only discrete-event transitions in order to solve the non-blocking forbidden state problem. The non-blocking problem is defined under both strong and weak conditions. For the latter maximally permissive solutions that are computable on a finite quotient space characterised by language equivalence are derived.

1. Introduction.

This paper concerns a language optimisation problem defined on Rectangular Hybrid Automata that was originally stated for finite-state automata [5] and then extended to timed automata in [2], [4]. A *hybrid automaton* is a model of a system that contains both continuous and discrete components. The control actions considered in this paper are exercised by preventing discrete events from occurring at particular states. The set of states of a hybrid automaton A is defined to be S_A and the discrete event transitions are labelled by the finite set Σ_A. We define the *supervisor* $\tau : S_A \times \Sigma_A \to \{0,1\}$; an event $\sigma \in \Sigma_A$ is permitted by τ to occur at the state $s \in S_A$ if and only if $\tau(s,\sigma) = 1$. A state of A is called τ*-reachable* if it can be reached from an initial state via a path all of whose events are permitted by τ. The problem definition involves partitioning the event set Σ_A into two subsets, $\Sigma_A = \Sigma_c \cup \Sigma_u$, the *controllable* and *uncontrollable* events respectively. The controllable events are characterised by the fact that they can at any point be prevented from occurring, whereas the uncontrollable events cannot be prevented from occurring; that is, always $\tau(s,\sigma) = 1$ if $\sigma \in \Sigma_u$. This definition was used in [5] and in this respect differs from the controller synthesis problem stated in [3, Def. 2.1.34] where all events except one at each state are prevented from occurring. Also a control problem under the hybrid games formalism has been considered in [6].

The supervisory control problem is defined as follows: A compact rectangular hybrid automaton A with fixed activity rectangle and two sets of locations F,T of A are given**.** Compute a supervisor τ such that in the *controlled hybrid automaton* (A,τ) a) the locations in F are inaccessible, b) a location in T can be reached from every τ-reachable state of A (this latter condition we call the nonblocking condition) and c) the language accepted by (A,τ) is maximal for all supervisors with this property (this statement will be made precise later).

Our main result is that the supervisory problem is decidable and it is possible to compute the supervisor τ. We partition the state set into finitely many equivalence classes, based on language equivalence, and we show that if any two transitions with controllable events lead to equivalent states, then both or neither of these transitions should be permitted. The solution follows from the fact that the reachability problem, for the class of rectangular hybrid automata we are considering, can be solved by replacing the hybrid automaton by a timed automaton, in which the corresponding supervisory control problem can be solved. In this respect the controller synthesis is implemented by an *exact* (accurate) computational algorithm that terminates. Initialised rectangular automata therefore are, together with timed automata, classes of hybrid automata for which the controller synthesis procedure terminates. Beyond that approximating schemes are needed, see [7]. If the hybrid automaton can be partitioned into finitely *bisimulation classes* (see [3, 2.1.18] for a definition and discussion of this term), like a timed automaton (see [1]), then the supervisory control problem is decidable and the supervisor computable, (see also [3, 2.1.36]. However for the class of rectangular hybrid automata that we consider, there are uncountably many bisimulation classes, except for some low-dimensional special cases (see [3, chapter 6]).

This paper is organised as follows. In part 2 we give the essential definitions and a precise statement of the problems that we are attempting to solve. In parts 3 and 4 we prove the main results and in part 5 we state our conclusions.

2. Definitions

A rectangular hybrid automaton (RHA) is defined as in [3, 2.2] but with several necessary restrictions. Let $\mathbf{R}$ be the set of real numbers. We define B_n to be the set of all compact rectangles of dimension n; that is, an element of B_n is a set

$\prod_{i=1}^{n}[L_i,U_i]\subseteq \mathbf{R}^n$ with each L_i,U_i an integer. In order to prove our results, we assume that the RHA A is compact (that is, all rectangles given in the definition of an RHA are compact). A compact RHA A of *dimension* n is defined by the following sets;

1) a finite alphabet Σ_A;
2) a finite directed graph (V_A, E_A). Elements of V_A are called *vertices* (or locations)*;* elements of E_A are *edges;*
3) a function $inv_A : V_A \to \mathrm{B}_n$. We call $inv_A(v)$ the *invariant set* of the vertex v. A *state* of A is a pair $s=(v,x)$ with $v\in V_A$, $x\in inv_A(v)$. We say x is the *continuous part,* and v the *discrete part,* of the state (v,x). We write $v = discrete((v,x)), x = cts((v,x))$. The invariant set specifies the allowable continuous states at each location. The set of states is S_A. We also define $S_A(v)$ to be the set of states with discrete part $v\in V_A$;
4) an *activity rectangle* $act_A \in \mathrm{B}_n$. The activity rectangle defines the rates at which the continuous part of a state changes. In [3] the general definition of a rectangular automaton assumes an activity rectangle for each vertex, but here we assume that each vertex has the same activity rectangle. Basically we assume that the rectangular automata are *initialised* (see [3, 3.4]). Also, we need to assume that no face of act_A has a zero co-ordinate; that is, each $L_i \neq 0, U_i \neq 0$ when act_A is expressed as given above. These restrictions are essential for proving our results*;*
5) an *initial state function* $I_A : V_A \to inv_A(v)$. We need not to assume that for every vertex $v\in V_A$, the set $I_A(v)$ is a rectangle in order to prove our results, although in [3] this set is considered rectangular. We define $init_A = \{(v,x)\in S_A : x\in I_A(v)\}$;
6) the functions $preguard_A : E_A \to \mathrm{B}_n, update_A : E_A \to 2^{\{1,\ldots,n\}}, postguard_A : E_A \to \mathrm{B}_n$;
7) a function $event_A : E_A \to \Sigma_A$

Functions 6) and 7) define when a discrete event can take place.

The following set of binary relations is associated with A:

For each element $\sigma\in\Sigma\cup\mathbf{R}_{\geq 0}$ there is a relation $\xrightarrow[A]{\sigma}\in S_A\times S_A$. We write $s_1\xrightarrow[A]{\sigma}s_2$ to mean $(s_1,s_2)\in\xrightarrow[A]{\sigma}$.

Next the relation; $(v,x)\xrightarrow[A]{t}(v,y)$ holds if there is a differentiable function $f:[0,t]\to inv_A(v)$ with derivative in act_A and $f(0)=x, f(t)=y$, $t\geq 0$. This is equivalent to the condition that either $x=y$ or there is a function f with *constant* derivative satisfying the conditions stated. Note that $(v,x)\xrightarrow[A]{t_1}(v,y)$, $(v,y)\xrightarrow[A]{t_2}(v,z)$ implies $(v,x)\xrightarrow[A]{t_1+t_2}(v,z)$. We also write $\xrightarrow[A]{time}=\bigcup_{t\geq 0}\xrightarrow[A]{t}$. The relation $(v,x)\xrightarrow[A]{\sigma}(w,y)$ holds if there is an edge $e\in E_A$ from v to w with $\sigma = event_A(e)$, $x\in preguard_A(e)$, $y\in postguard_A(e)$ and for all $i\notin update_A(e)$, the ith component of x is equal to the ith component of y. We write $s_1\xrightarrow[delayed,A]{\sigma}s_2$ if $s_1\xrightarrow[A]{t}s_1'$ and $s_1'\xrightarrow[A]{\sigma}s_2$ for some $t\geq 0$ and $s_1'\in S_A$. For $Q\subseteq S_A$ we write $pre^{delayed,\sigma}(Q)$ to be the set of states s satisfying $s\xrightarrow[delayed,A]{\sigma}s_1\in Q$ and $pre^{v,delayed,\sigma}(Q) = pre^{delayed,\sigma}\cap S_A(v)$. The language $L_A(s)\subseteq\Sigma_A^*$ is defined as follows; if $s_1\xrightarrow[delayed,A]{\sigma_1}s_2\cdots\cdots s_r\xrightarrow[delayed,A]{\sigma_r}s_{r+1}$ then $\sigma_1\cdots\sigma_r\in L_A(s_1)$. We also define $L_A^t(s)\subseteq(\Sigma_A\cup\mathbf{R}_{\geq 0})^*$ if $s_0\xrightarrow[A]{t_1}s_1\xrightarrow[A]{\sigma_1}\bar{s}_1\cdots\cdots s_m\xrightarrow[A]{\sigma_m}\bar{s}_m$ then $t_1\sigma_1\ldots t_m\sigma_m\in L_A^t(s_1)$. We define $L_A=\bigcup_{s\in I_A}L_A(s)$ and define L_A^t similarly.

Next we define the notion of *language equivalence* used below. Given a compact rectangle $R\in\mathrm{B}_n$ and $x,y\in\mathbf{R}^n$, we say that x and y are *language equivalent* for R if for every compact rectangular hybrid automaton A' with activity rectangle $act_{A'}=R$ and *every* location $v\in V_{A'}$ we have $L_{A'}(v,x)=L_{A'}(v,y)$. Two states of a hybrid automaton are *language equivalent* for R if they have the same discrete part and their continuous parts are language equivalent. We now define an equivalence relation $\cong_A$ on $\mathbf{R}^n$ which is known to be at least as fine as language equivalence for act_A. Given $x=(x_1,\ldots x_n), x'=(x_1',\ldots x_n')\in\mathbf{R}^n$, assume that $act_A=\prod_{i=1}^{n}[L_i,U_i]\subseteq\mathbf{R}^n$ and let λ be the least common multiple of the integers in the set $\{L_1,U_1,\ldots,L_n,U_n\}$ and define $x\cong_A x'$ if

- $\forall i\leq n, \mathrm{int}(x_i\frac{\lambda}{L_i}+1)=\mathrm{int}(x_i'\frac{\lambda}{L_i}+1)$ and

- $\forall i, j \le n, \text{int}(x_j \frac{\lambda}{U_j} - x_i \frac{\lambda}{L_i}) =$
- $\text{int}(x'_j \frac{\lambda}{U_j} - x'_i \frac{\lambda}{L_i})$.

Note that owing to our hypotheses, there are no zero denominators. This is the only point in the paper at which every $U_i \neq 0, L_i \neq 0$ needs to be assumed. It should be observed that in every compact rectangle there are finitely many $\cong_A$-classes. It is proved in [3, Corollary 7.3.2] that $\cong_A$-equivalence implies language equivalence (sufficient condition). Given two states $(v_1, x_1), (v_2, x_2) \in S_A$ we write $((v_1, x_1), (v_2, x_2)) \in \cong_A$ to mean $(x_1, x_2) \in \cong_A$ and $v_1 = v_2$.

Let the event alphabet Σ_A be partitioned as $\Sigma_A = \Sigma_c \cup \Sigma_u$, the *controllable* and *uncontrollable* events respectively. Also, and in order to define clearly the supervisor, we require one further condition on A; if $\sigma \in \Sigma_c$ and $s \xrightarrow[A]{\sigma} s'$, and $s \xrightarrow[A]{\sigma} s''$ then $s' = s''$. Then we will say that A is Σ_c-*deterministic.* This condition can be enforced by assuming that $postguard_A(e)$ is a singleton if the edge e has label in Σ_c and that no two edges with the same label in Σ_c have the same starting vertex.

Definition of the supervisory controller: We consider the supervisory controller to be the (static) state-feedback map $\tau : S_A \to \Gamma$ where Γ is the set of control patterns defined to be $\Gamma := \{\gamma \subseteq \Sigma_A / \gamma \supseteq \Sigma_u\}$. Thus a supervisor for A is a function $\tau : V_A \times \mathbf{R}^n \times \Sigma_A \to \{0,1\}$ satisfying $\tau(v, x, \sigma) = 1$ if $\sigma \in \Sigma_u$. We write $s_1 \xrightarrow[\tau, delayed, A]{\sigma} s_2$ if $s_1 \xrightarrow[A]{t} s_1'$ and $s_1' \xrightarrow[A]{\sigma} s_2$ for some $t \ge 0$ and $s_1' \in S_A$ and $\tau(s_1', \sigma) = 1$; also, we write $s_1 \xrightarrow[\tau, A]{\sigma} s_2$ if $s_1 \xrightarrow[A]{\sigma} s_2$ and $\tau(s_1, \sigma) = 1$. We define $L(A, \tau) \subseteq L_A, L^t(A, \tau) \subseteq L_A^t$ by only considering transitions allowed by τ. For any subset $\Sigma' \subseteq \Sigma_A$, the relation $\xrightarrow[delayed, A]{\Sigma'}$ is the union of the relations $\xrightarrow[delayed, A]{\sigma}$ for $\sigma \in \Sigma'$; also, $\xrightarrow[\tau, delayed, A]{\Sigma'}$ is defined similarly.

For any relation $\to$, the relation $\to^*$ is the transitive closure of the union of $\to$ and equality. If $s_0 (\xrightarrow[\tau, delayed, A]{\Sigma_A})^* s$ with $s_0 \in init_A$ then we say that s is τ-***reachable*** and if $s_0 (\xrightarrow[\tau, delayed, A]{\Sigma_A})^* s \xrightarrow[A]{time} s'$ then s' is ***almost τ-reachable.***

The main problem of this paper is the following; assume that the sets $F, T \subseteq V_A$ are given. The assumption that these sets are 'whole' vertex sets (and not 'part' of them) is withought loss of generality.

Problem definition: Non-blocking Forbidden State Problem for Rectangular Hybrid Automata

The supervisory control problem takes the following form: Find a supervisor τ for A satisfying:

1. $\neg s (\xrightarrow[\tau, delayed, A]{\Sigma_A})^* s'$, *for all* $s \in init_A$ *and* $discrete(s') \in F$ (that is, τ avoids F. This defines the ***forbidden state problem*** or a safety property problem) *and*

2. if $s \in S_A$ is any τ-reachable state, then $s (\xrightarrow[\tau, delayed, A]{\Sigma_A})^* s'$ for $discrete(s') \in T$ (that is, τ avoids blocking. This defines an eventuality property problem or the ***non-blocking controller***).

We then say that τ solves the *weak non-blocking forbidden state problem* for the vertex sets F, T. We also define the *strong non-blocking forbidden problem,* in which the hypothesis on $s \in S_A$ is merely that s is *almost* τ-*reachable,* and which in other respects is the same as the weak non-blocking problem. The strong reachability problem appears to be more difficult to solve than the weak non-blocking forbidden problem. Recall that disabling of only discrete-event transitions is used and no assumptions concerning non-blocking behaviour have been assumed for the hybrid automaton A. In this paper we concentrate mostly on the weak non-blocking forbidden problem. It is not guaranteed that a solution to either problem exists. We prove that if a solution to the weak nonblocking problem exists, then there exists a solution $\tau_{\max}$, decidable and computable, which is *maximally permissive*; that is, if any supervisor τ also solves the weak nonblocking problem and for some $s_0 \in init_A$, $t_i \ge 0$ and $\sigma_i \in \Sigma_A$ we have $s_0 \xrightarrow[\mu, A]{t_1} s_1 \xrightarrow[\mu, A]{\sigma_1} \bar{s}_1 \cdots\cdots s_m \xrightarrow[\mu, A]{\sigma_m} \bar{s}_m$ for $\mu = \tau$ then this also holds for $\mu = \tau_{\max}$. Then $L^t(A, \tau) \subseteq L^t(A, \tau_{\max})$. We show that if $s_1 \xrightarrow[A]{\sigma_1} s', s_2 \xrightarrow[A]{\sigma_2} s''$ with $\sigma_1, \sigma_2 \in \Sigma_c$ and s', s'' are $\cong_A$-equivalent then $\tau_{\max}(s_1, \sigma_1) = \tau_{\max}(s_2, \sigma_2)$. Based on this we then show that given a $\cong_A$-class r then $\tau_{\max}(s_1, \sigma)$ is computable for any $s_1 \in S_A, s_1 \xrightarrow[A]{\sigma} s' \in r$.

3. Main results

3.1 Finite partition of the rectangular automaton

In order to compute the required supervisor, it suffices to be able to construct a *labelled* transition system defined over a

finite state set which enables us to establish, given a state $s_1 \in S_A$, whether $s_1(\underset{delayed,A}{\overset{\Sigma_u}{\rightarrow}})^* s'$ for any $s' \in F$. It is this that motivates the following results in this section. First we consider the following lemma concerning language equivalence.

Lemma 1. If $(v_1,x_1) \underset{delayed,A}{\overset{\sigma_1}{\rightarrow}} (v_2,x_2)\cdots \cdots \underset{delayed,A}{\overset{\sigma_m}{\rightarrow}} (v_{m+1},x_{m+1})$ and $x_1 \cong_A x_1'$ then $(v_1,x_1') \underset{delayed,A}{\overset{\sigma_1}{\rightarrow}} (v_2,x_2')\cdots \cdots \underset{delayed,A}{\overset{\sigma_m}{\rightarrow}} (v_{m+1},x_{m+1}')$ for elements $x_i' \in \mathbf{R}^n$. (Note that we do not claim $x_i \cong_A x_i'$ for $i>1$.)

Proof. Since $x_1 \cong_A x_1'$ we must have $(v_1,x_1') \underset{delayed,A}{\overset{\sigma_1}{\rightarrow}} (v_2',x_2')\cdots \cdots \underset{delayed,A}{\overset{\sigma_m}{\rightarrow}} (v_{m+1}',x_{m+1}')$. To prove that each $v_i = v_i'$, construct a hybrid automaton $\overline{A}$ without loops in the directed graph defined by its vertex and edge sets, of dimension n with activity rectangle act_A and containing only the locations and edges of A appearing in the statement $(v_1,x_1) \underset{delayed,A}{\overset{\sigma_1}{\rightarrow}} (v_2,x_2)\cdots \cdots \underset{delayed,A}{\overset{\sigma_m}{\rightarrow}} (v_{m+1},x_{m+1})$ (if this path passes through the same vertex more than once then duplicate vertices will have to be defined). The result now follows from the fact that $x_1 \cong_A x_1'$ implies $x_1 \cong_{\overline{A}} x_1'$ and so $L_{\overline{A}}(v,x_1) = L_{\overline{A}}(v,x_1')$.

Definition 2. A statement $(v_1,x_1) \underset{delayed,A}{\overset{\sigma_1}{\rightarrow}} (v_2,x_2)\cdots \cdots \underset{delayed,A}{\overset{\sigma_m}{\rightarrow}} (v_{m+1},x_{m+1})$ is called a *path* over the *vertex-sequence* $v_1,\ldots,v_{m+1}$.

Using language equivalence define a finite partitioning of $\mathbf{R}^n$ and let $r_{\cong}$ be the corresponding function from $\mathbf{R}^n$ into the set $\mathbf{R}^n / \cong_A$ of equivalence classes.

Definition 3. We define the labelled transition system on the finite set $V_A \times \mathbf{R}^n / \cong$ as follows. Given states $(v_1,x_1),(v_2,x_2)$ we define $(v_1, r_{\cong}(x_1)) \overset{\sigma}{\Longrightarrow} (v_2, r_{\cong}(x_2))$ if $(v_1,x_1) \underset{delayed,A}{\overset{\sigma}{\rightarrow}} (v_2,x_2)$.

Theorem 4. *Let* $r_{\cong}(x_1) = r_1$. *Then* $(v_1,r_1) \overset{\sigma_1}{\Longrightarrow} (v_2,r_2)\cdots\cdots \overset{\sigma_m}{\Longrightarrow} (v_{m+1},r_{m+1})$ *for sets* $r_i \subseteq \mathbf{R}^n$ *iff* there exists a path $(v_1,x_1) \underset{delayed,A}{\overset{\sigma_1}{\rightarrow}} (v_2,x_2)\cdots \underset{delayed,A}{\overset{\sigma_m}{\rightarrow}} (v_{m+1},x_{m+1})$ *for* $x_i \in \mathbf{R}^n$.

Proof. That the second assertion implies the first follows immediately from the definition of $\overset{\sigma}{\Rightarrow}$.

We assume the first assertion and prove the second using induction on m. Thus $(v_1,x_1') \underset{delayed,A}{\overset{\sigma_1}{\rightarrow}} (v_2,x_2')$ for some $x_1' \in r_1$, $x_2' \in r_2$ by the definition of $\overset{\sigma}{\Rightarrow}$. Also $(v_2,x_2') \underset{delayed,A}{\overset{\sigma_2}{\rightarrow}} \cdots\cdots \underset{delayed,A}{\overset{\sigma_m}{\rightarrow}} (v_{m+1},x_{m+1}')$ for $x_i' \in \mathbf{R}^n$ follows from the inductive hypothesis. The result now follows from Lemma 1 and the fact that $x_1' \in r_1$ implies $x_1 \cong_A x_1'$.

Remark*:* Note that $r_{\cong}(x_i) \neq r_i$ *for* $i>1$. This would be true if we had bisimulation equivalence.

3.2 Forbidden state problem

Next we consider the forbidden state problem (whithought involving non-blocking) and we show that this is decidable. We state this problem briefly in order to give some intuition and to clarify that the non-blocking forbidden problem is more involved. The latter is stated afterwards in 3.3

Theorem 5 Assume that $\neg\left(s_0(\underset{delayed,A}{\overset{\Sigma_u}{\rightarrow}})^* s'\right)$ for $s_0 \in init_A$, $s' \in F$ (otherwise no solution to the problem could exist). Define the supervisor τ as follows; $\tau(s_1,\sigma) = 0$ if there exists a state $s \in S_A$ with $s_1 \underset{A}{\overset{\sigma}{\rightarrow}} s$ and $s(\underset{delayed,A}{\overset{\Sigma u}{\longrightarrow}})^* s'$ with the discrete part of s' in F and $\sigma \in \Sigma_c$. In all other cases let $\tau(s_1,\sigma) = 1$. Given any supervisor $\overline{\tau}$ that avoids the vertex set F, we have $L^t(A,\tau) \supseteq L^t(A,\overline{\tau})$.

Also, if $s_1 \underset{A}{\overset{\sigma_1}{\rightarrow}} (v,x_1)$ and $s_2 \underset{A}{\overset{\sigma_2}{\rightarrow}} (v,x_2)$ with each $\sigma_i \in \Sigma_c$ and $x_1 \cong x_2$ then $\tau(s_1,\sigma_1) = \tau(s_2,\sigma_2)$, and this means that *if there exists a supervisor solving the forbidden state problem then this is constant on transitions mapping into the same* $\cong_A$ *class.*

Proof. Assume first that the defined supervisor does not solve the problem i.e. there exists a path $s_0(\underset{\tau,delayed,A}{\overset{\Sigma}{\rightarrow}})^* s'$ with $s_0 \in init_A$ and $s' \in F$. Then since $\neg\left(s_0(\underset{delayed,A}{\overset{\Sigma_u}{\rightarrow}})^* s'\right)$

by the hypothesis, we infer that $s_1' \xrightarrow[\tau,A]{\sigma} s'' (\xrightarrow[\tau,delayed,A]{\Sigma_u})^* s'$ with $\sigma \in \Sigma_c$. However this is impossible, since we then have $s'' (\xrightarrow[\tau,delayed,A]{\Sigma_u})^* s'$ and thus $\tau(s_1', \sigma) = 0$ by definition.

Next assume that $\bar{\tau}$ is a supervisor and that $s_0 \xrightarrow[\mu,A]{t_1} s_1 \xrightarrow[\mu,A]{\sigma_1} \cdots s_m \xrightarrow[\mu,A]{\sigma_m} \bar{s}_m$ holds for $\mu = \bar{\tau}$ but not for $\mu = \tau$ and m is minimal such that this condition is satisfied. Thus $\sigma_m \in \Sigma_c$ and $\bar{s}_m (\xrightarrow[delayed,A]{\Sigma_u})^* s'$ for some $s' \in F$, by the definition of τ, and this implies that a location in F is accessible from $s_0 \in init_A$ under $\bar{\tau}$, hence $\bar{\tau}$ is not a solution to the forbidden state problem.

The last part of the theorem follows from lemma 1. This is an important property for the involved partition of $\cong_A$ classes.

We next show that it is decidable whether we have $\tau(s_1, \sigma) = 1$ or 0 for a transition $s_1 \xrightarrow[A]{\sigma} (v, x)$, where $\sigma \in \Sigma_c$, r is a $\cong_A$-class, $x \in r$ and $v \in V_A$. Following the last part of Theorem 5, for each $s_1 \in S_A$ and event σ the transition $s_1 \xrightarrow[A]{\sigma} (v, x)$ mapping into the $\cong_A$ class $(v, r) = (v, r_{\cong}(x))$ can be established as disabled or not, depending whether there exists a sequence of uncontrollable or not labelled transitions from the $\cong_A$ class to a forbidden state. The decidability follows from Theorem 6 showing that the condition $(v, r_{\cong}(x))(\xRightarrow{\Sigma_u})^* s'$ for some $s' \in F$ can be checked since the set $V_A \times \mathbf{R}^n / \cong$ is finite.

Theorem 6. It is decidable whether, given a $\cong_A$-class r and $v \in V_A$ we have $s(\xrightarrow[delayed,A]{\Sigma_u})^* s'$ with $s' \in F$ for any (and hence all) states s whose continuous part lies in r and whose discrete part is v.

Proof (sketch). Choose any vector $x \in r$ with rational components. We may assume that the components of $x \in r$ are in fact integers; otherwise it is necessary to scale up all the dimensions of A. Let A' be the rectangular hybrid automaton that is identical to A except that its initial state set is $\{v, x\}$ and all controllable edges are deleted. In [3, Def. 3.2.2] a $2n$ dimensional timed automaton $N_{A'}$ with vertex set V_A is constructed whose initial state set is $\{v, x, x\}$ such that $(v, x)(\xrightarrow[delayed,A']{\Sigma_u})^* s'$ with $s' \in F$ if and only if $(v, x, x)(\xrightarrow[delayed,N_{A'}]{\Sigma_u})^* s'$ with $x \in r$ The latter is a decidable reachability problem that can be checked on the (finite) region equivalence quotient space of the timed automaton using, for example, the untiming construction, see [1], [3].

3.3 Weak Non-blocking Forbidden State Problem

Our main result is the computation of the non-blocking supervisor $\tau_{\max}$ for each equivalent class $\cong_A$ of A.

Intuitively due to the existence of the uncontrollable events we need to identify possible deadlock states and remove them. State set backpropagation is needed and for decidability reasons we introduce the notion of the so-called *good sets*. Since the considered quotient space based on language equivalence is more 'subtle' than bisimilation, we require some additional properties in order to solve the non-blocking forbidden state problem.

Definition 7. A set of states $Q \subseteq S_A$, all with the same discrete part $v \in V_A$, is called *elementary* if there are two sets of vertex-sequences, U', U'', all of whose members have the same first vertex, and such that given $x \in S_A(v)$ we have $x \in Q$ ***if and only if*** for every sequence $u \in U'$ there is a path over u starting at the state x and for every sequence $u \in U''$ there is no path over u starting at x. We write $Q = \Theta_A(U', U'')$ if this holds. If the state sets $Q_1, \ldots, Q_m$ are all elementary and all elements of $Q_1 \cup \ldots \cup Q_m$ have the same discrete part, then this set is called *good*.

By lemma 1, an elementary set is a union of language equivalence classes (and hence $\cong_A$-classes) of states, thus there are finitely many elementary (and good) sets at each vertex.

Lemma 8. Let $v \in V_A$ and let Q_1, Q_2 be good sets at v. Then $Q_1 \cup Q_2, Q_1 \cap Q_2$ and $S_A(v) - Q_1$ are all good sets.

Proof. Clearly $Q_1 \cup Q_2$ is a good set at v. To show that $Q_1 \cap Q_2$ is a good set, assume first that Q_1, Q_2 are elementary. Thus there are vertex-sequences U_i', U_i'' such that $Q_i = \Theta_A(U_i', U_i'')$. Then $Q_1 \cap Q_2 = \Theta_A(U_1' \cup U_2', U_1'' \cup U_2'')$ which is elementary. The general case follows from the fact that unions of good sets are good, and the fact that $\cap$ is distributive over $\cup$.

To show that $S_A(v) - Q_1$ is a good set, we assume first that $Q_1 = \Theta_A(U', U'')$. Then $S_A(v) - Q_1 = \bigcup_{u \in U'} \Theta_A(\varnothing, \{u\}) \cup \bigcup_{u \in U''} \Theta_A(\{u\}, \varnothing)$ whi ch is good. The general case follows from the fact that intersections of good sets are good.

Lemma 9. Let $Q \subseteq S_A$ be a good set with discrete part $w \in V_A$ and assume there is an edge $v \xrightarrow{e} w$, with $event_A(e) = \sigma$. Then $pre^{v,delayed,\sigma}(Q)$ is good.

Proof. We may assume that Q is in fact elementary, since $pre^{v,delayed,\sigma}$ commutes with the union operator; thus let $Q = \Theta_A(U', U'')$. Let $\overline{U'}$ be the set of vertex-sequences

obtained by preceding those in U' with v and define $\overline{U''}$ using U'' similarly. Then $pre^{v,delayed,\sigma}(Q) = \Theta_A(\overline{U'} \cup \{(v)\}, \overline{U''})$ which is elementary, as required.

The next result is essential to show that the algorithm below can, in fact, be computed.

Lemma 10. Given a good set Q, and an edge $v \xrightarrow{e} w$, with $event_A(e) = \sigma$, it is possible to compute $pre^{v,delayed,\sigma}(Q)$ as unions of $\cong_A$-classes.

Proof. This proof is sketchy since is using involved notions developed in [3]. We may assume that Q is elementary; $Q = \Theta_A(U', U'')$. In [3, Def. 3.2.2] a $2n$-dimensional timed automaton N_A is defined with the same vertex and edge sets and a computable mapping ξ_A from $2^{S_{N_A}}$ to 2^{S_A} which preserves unions. The timed automaton N_A is an integral one-sided timed automaton with attractors, see [3, 3.1.10 and 7.2.3]. Then $\xi_A \Theta_{N_A}(U', U'') = \Theta_A(U', U'')$ and $\xi_A pre^{v,delayed,\sigma} \Theta_{N_A}(U', U'') = pre^{v,delayed,\sigma} \xi_A \Theta_{N_A}(U', U'')$; both these statements follow from [3, Lemma 3.2.8]. Furthermore, $\Theta_{N_A}(U', U'')$ is a union of language equivalence classes, and by [3, Theorem 7.2.18], a language equivalence class is also an asynchronous simulation class. This equivalence relation, called one-sided region equivalence, is considerably coarser than the known bisimilation relation called region equivalence. Following lemma 9 its preimage $pre^{v,delayed,\sigma} \Theta_{N_A}(U', U'')$ is also a union of asynchronous simulation classes, therefore this set can be computed. Consequently the set $pre^{v,delayed,\sigma} \xi_A \Theta_{N_A}(U', U'')$ can be computed as well. Clearly this set is computed as unions of $\cong_A$-classes.

The pre-image of a bisimulation class under a transition is necessarily a union of bisimulation classes, whereas the equivalence classes we consider here do not satisfy this property, and this leads to decidability problems when the non-blocking forbidden problem is considered. The result above is not true if Q is merely assumed to be a union of $\cong_A$-classes; for in that case there is no guarantee that $pre^{v,delayed,\sigma}(Q)$ is a union of $\cong_A$-classes since a finite asynchronous bisimulation quotient does not exist, in general, for rectangular automata (see [3, Theorem 6.2.4]. Therefore the introduction of good sets of states is necessary. Using these sets we show the termination of the algorithm below in a finite number of steps.

Computation of the non-blocking supervisor $\tau_{\max}$

The main algorithm proceeds as follows. It labels at each stage unions of $\cong_A$-classes as *acceptable*, *forbidden* or *blank*. These unions are all unions of good sets, since for any vertex $v \in V_A$ the set $S_A(v) = \Theta_A(\{(v)\}, \varnothing)$ is elementary and therefore steps 1, 2 and 3 below (using lemmas 8 and 9) label only unions of good sets. The notion of elementary and goods set is not an assumption for the stated problem but a property used to show decidability of the involved algorithm.

During the algorithm a $\cong_A$-class which has been labelled may be relabelled; however a $\cong_A$-class which has been labelled *forbidden* will not subsequently change this label.

1. Label all the $\equiv_A$-classes with discrete part in F *forbidden* and label all other $\cong_A$-classes *blank*.
2. Label all the $\equiv_A$-classes with discrete part in T *acceptable* unless they have been labelled *forbidden*.
3. Do steps (a) and (b) below until doing either step leaves unchanged the labelling of all $\equiv_A$-classes.

(a) If there is an edge $v \xrightarrow[A]{e} w$, with $event_A(e) = \sigma \in \Sigma_u$, and $Q \subseteq S_A$ is the union of $\equiv_A$-classes at w which have been labelled *forbidden*, then label those in $pre^{v,delayed,\sigma}(Q)$ *forbidden*.

(b) If there is an edge $v \xrightarrow{e} w$, with $event_A(e) = \sigma$, and $Q \subseteq S_A$ is the union of $\cong_A$-classes at w which have been labelled *acceptable*, and $\sigma \in \Sigma_u$, let $P \subseteq S_A$ be the union of $\cong_A$-classes at w which have been labelled *forbidden*. Then label *acceptable* the $\cong_A$-classes in $pre^{v,delayed,\sigma}(Q - P)$. If on the other hand $\sigma \in \Sigma_c$ then label *acceptable* the $\cong_A$-classes in $pre^{v,delayed,\sigma}(Q)$.

4. Relabel *forbidden* all $\cong_A$-classes which are labelled *blank*, and relabel *blank* all $\cong_A$-classes which are labelled *acceptable*.
5. Repeat steps 2,3,4 (in that order) repeatedly until there are no further changes in labelling.

There are finitely many $\cong_A$-classes and so all steps of the algorithm except step 5 obviously terminate. Step 5 either increases the number of $\cong_A$-classes labelled *forbidden* or keeps this number constant; and in the latter case step 5 has no effect; thus the algorithm eventually terminates.

We then define the supervisor $\tau_{\max} : V_A \times \mathbf{R}^n \times \Sigma_A \to \{0,1\}$ by $\tau_{\max}(s, \sigma) = 0$ if $\sigma \in \Sigma_c$ and $s \xrightarrow[A]{\sigma} s'$ and s' lies in a *forbidden* $\cong_A$-class at the end of the algorithm and $\tau_{\max}(s, \sigma) = 1$ otherwise. Since A is Σ_c-deterministic, $\tau_{\max}$ is well defined. This is the only point in the paper at which Σ_c-determinism is needed.

Corollary 11 If there are transitions $s_1 \xrightarrow[A]{\sigma_1} s', s_2 \xrightarrow[A]{\sigma_2} s''$ with $\sigma_1, \sigma_2 \in \Sigma_c$ and s', s'' are $\cong_A$-equivalent then $\tau_{\max}(s_1, \sigma_1) = \tau_{\max}(s_2, \sigma_2)$.

Proof. Follows from lemma 1.

Following the algorithm above and corollary 11 we conclude that given a $\cong_A$-class r then the derivation of $\tau_{\max}(s_1, \sigma)$ is decidable for any $s_1 \in S_A$, $s_1 \xrightarrow[A]{\sigma} (v, x)$, $x \in r$.

Theorem 12. If a solution to the weak non-blocking forbidden state problem exists then $\tau_{\max}$ is a maximally permissive solution to this problem.

Proof. We first prove that $\tau_{\max}$ is a solution to the weak non-blocking forbidden state problem.

Assume first that $s_0 \xrightarrow[\tau_{\max},A]{t_1} s_1 \xrightarrow[\tau_{\max},A]{\sigma_1} \bar{s}_1 \cdots\cdots s_m \xrightarrow[\tau_{\max},A]{\sigma_m} \bar{s}_m$ with $s_0 \in init_A$ and $discrete(\bar{s}_m) \in F$. At least one $\sigma_i \in \Sigma_c$, otherwise no solution to the weak non-blocking forbidden state problem could exist. Assume that i is maximal with this property; then the $\cong_A$-class containing $\bar{s}_i$ would become *forbidden* after repeated applications of step 3a in the algorithm, contradicting $s_i \xrightarrow[\tau_{\max},A]{\sigma_i} \bar{s}_i$. Next assume that $s_0 \xrightarrow[\tau_{\max},A]{t_1} s_1 \xrightarrow[\tau_{\max},A]{\sigma_1} \bar{s}_1 \cdots\cdots s_m \xrightarrow[\tau_{\max},A]{\sigma_m} \bar{s}_m$ with $s_0 \in init_A$, and there is no path $\bar{s}_m \xrightarrow[\tau_{\max},A]{t_1} \cdots\cdots \xrightarrow[\tau_{\max},A]{\sigma} s$ with $discrete(s) \in T$. This implies that the $\cong_A$-class containing $\bar{s}_m$ would have become *forbidden* after repeated applications of step 5. Assume, in this case, that $i \le m$ is maximal with $\sigma_i \in \Sigma_c$. (If no such $i \le m$ exists, then no solution to the weak non-blocking forbidden problem exists.) Thus the $\cong_A$-class containing $\bar{s}_i$ would have become *forbidden* after repeated applications of step 3a, once the $\cong_A$-class containing $\bar{s}_m$ had become *forbidden*. This contradicts $s_i \xrightarrow[\tau_{\max},A]{\sigma_i} \bar{s}_i$. Thus $\tau_{\max}$ solves the weak non-blocking forbidden problem. The maximality assertion follows from the fact that in the computation of $\tau_{\max}$ we only exclude states that are necessarily inaccessible.

4. On the Strong Non-blocking Forbidden State Problem

This appears to be more difficult since the identification and removing of possible blocking states may become undecidable. It is worth mentioning one case in which a solution to the weak non-blocking problem also solves the strong non-blocking forbidden problem.

Theorem 13. Assume that $0 \in \mathbf{R}^n$ is an interior point of the activity rectangle act_A. Then if the supervisor τ solves the weak non-blocking forbidden problem, it also solves the strong non-blocking forbidden problem.

Proof. This follows from the fact that if $s_1 \xrightarrow[A]{time} s_2$ with $cts(s_i) = x_i$ then $x_2 = x_1 + at$ with $a \in act_A, t \ge 0$. Choose $\lambda > 0$ small enough so that $-\lambda a \in act_A$; since $x_1 = x_2 + (-\lambda a)(\frac{t}{\lambda})$, we have $s_2 \xrightarrow[A]{time} s_1$. Assume that a state s' is almost τ-reachable; that is, that $s_0 (\xrightarrow[\tau,delayed,A]{\Sigma_A})^* s \xrightarrow[A]{time} s'$ for some $s_0 \in init_A$. Thus s is τ-reachable and so $s(\xrightarrow[\tau,delayed,A]{\Sigma_A})^* s''$ *for* $discrete(s'') \in T$. Since $s' \xrightarrow[A]{time} s$, we have $s'(\xrightarrow[\tau,delayed,A]{\Sigma_A})^* s''$, as required.

5. Conclusions

We have defined an algorithm that solves a language optimisation problem. There are several ways in which the results here could be extended. The compactness condition on A was introduced in order to be able to employ the results in [3] in Lemma 10. This condition can almost certainly be relaxed, albeit using more difficult arguments for the proof of Lemma 10. Also, the Σ_c-determinism condition may be unnecessary.

References

[1] R Alur, D Dill, A theory of timed automata, Theoreticcal Computer Science, vol. 126, pp. 183-235, 1994.

[2] G Hoffman, H Wong-Toi, The Control of Dense Real-time Discrete Event Systems; Technical report, University of Stanford, CS-1411-1992.

[3] P W Kopke, Theory of Rectangular Automata, PhD Thesis, Cornell University, 1996.

[4] M R Laurence, M P Spathopoulos, Forbidden State Problems in Timed Automata, Proceedings of WODES98, pp. 15-23, Cagliari, 1998.

[5] P Ramadge, W H Wonham, Supervisory control of a class of discrete event processes, SIAM Journal of Control and Optimisation, 25(1), 1202-1218, 1987.

[6] T A Henzinger, B Horowitz and R Majumdar, Rectangular Hybrid Games, Proc. CONCUR'99, LNCC, vol. 1664, pp.320-335, 1999.

[7] E Asarin, O Bournez, T Dang, O Maler, A Pnueli, Effective Synthesis of Switching Controllers for Linear Systems, Proceeding of the IEEE. To appear.