Non-blocking supervisory control for initialised rectangular 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
tel:+441415482326, fax:+441415525105

Abstract

We consider the problem of supervisory control for a class of rectangular automata and more specifically for compact rectangular automata with uniform rectangular activity, i.e. initialised. The supervisory controller is state feedback and disables 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.

Key words: Rectangular automata, supervisory control, non-blocking supervisor.

1. Introduction

This paper concerns a language optimisation problem defined on a class of rectangular automata (linear hybrid systems) that was originally stated for finite-state automata [5] and then extended to timed automata in [2], [4]. A *rectangular* (*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 and the discrete event transitions are labelled by the finite set Σ . We define the *supervisor* $\tau : S \times \Sigma \to \{0,1\}$; an event $\sigma \in \Sigma$ is permitted by τ to occur at the state $s \in S$ 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 into two subsets, $\Sigma = \Sigma_c \cup \Sigma_u$, 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 automaton A with fixed activity rectangle and two sets of locations (states) 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 supervisor τ is *maximally permissive* (this statement will be made precise later).

Our main result is that the non-blocking 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. Moreover, the partition of the state space in equivalence classes allows us to derive the *non-blocking supervisor* by removing the blocking states. The non-blocking supervisory algorithm is decidable and computable and we derive the appropriate algorithm using the partition. In this respect the controller synthesis is computed by an *exact* (accurate) algorithm that terminates.

Initialised rectangular automata 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 a finite number of *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 is an uncountable number of bisimulation classes, except for some low-dimensional special cases (see [3, chapter 6]). Thus, although the reachability problem is decidable is not known whether the non-blocking supervisory problem is decidable as well.

Decidability issues associated to hybrid automata have been studied in [9], [10] where the reader is referred for more details. In [9] initialised rectangular automata are stated as the boundary between decidability and undecidability of the reachability problem for hybrid automata. In [11] the problem of synthesis of controllers for generalised rectangular automata, named linear hybrid systems, has been addressed. Reachability was shown to be *semidecidable* for this class of systems and studies for zeno behaviour, time diverging and partial observability have been considered. The synthesis of controllers for hybrid systems, in general, and for *safety* and *eventuality* specifications, using optimal control and games theory , have been studied in [15] and [14] respectively.

The main contribution of the paper is the study of the supervisory control problem for rectangular automata under the Ramadge-Wonham methodology, [5]. In relation to [3] our original contribution includes the introduction of uncontrollable events in the model and the detailed study, derivation and exact computation of the non-blocking (dead-lock free) maximum permissive supervisor using the partition of the state space in language equivalence classes. In particular, the effect of the synchronous (timed automata) and asynchronous (rectangular automata) semantics for the derivation of the non-blocking supervisor is demonstrated.
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) A 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;
2) a finite directed graph (V,E). Elements of V are called *vertices* (or locations)*;* elements of E are *edges;*
3) a function $inv:V\rightarrow \mathrm{B}_n$. We call $inv\ (v)$ the *invariant set* of the vertex v. A *state* of A is a pair $s=(v,x)$ with $v\in V$, $x\in inv\ (v)$. 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. We also define $S\ (v)$ to be the set of states with discrete part $v\in V$;
4) an *activity rectangle* $act\in \mathrm{B}_n$. 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. In [3, 3.4] it was shown that basically the class of initialised rectangular automata is essentially equivalent to the class of rectangular automata with uniform dynamics. A rectangular automaton is called *initialised* if whenever the continuous dynamics of a variable x_i change, due to a change in location, the value of x_i is nondeterministically reinitialised. Also, we need to assume that no face of act has a zero co-ordinate; that is, each $L_i\neq 0, U_i\neq 0$ when act is expressed as given above. These restrictions are essential for proving our results*;*
5) an *initial state function* $I:V\rightarrow inv(v)$. We define $init=\{(v,x)\in S : x\in I(v)\}$;
6) the functions
$preguard : E\rightarrow \mathrm{B}_n, update : E\rightarrow 2^{\{1,\dots,n\}}, postguard : E\rightarrow \mathrm{B}_n$;

7) a function $event : E \to \Sigma$

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 $\overset{\sigma}{\to} \in S \times S$. We write $s_1 \overset{\sigma}{\to} s_2$ to mean $(s_1, s_2) \in \overset{\sigma}{\to}$. Next the relation; $(v,x) \overset{t}{\to} (v,y)$ holds if there is a differentiable function $f : [0,t] \to inv(v)$ with derivative in act 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 $(v,x) \overset{t_1}{\to} (v,y)$, $(v,y) \overset{t_2}{\to} (v,z)$ implies $(v,x) \overset{t_1+t_2}{\to} (v,z)$. We also write $\overset{time}{\to} = \bigcup_{t \geq 0} \overset{t}{\to}$. The relation $(v,x) \overset{\sigma}{\to} (w,y)$ holds if there is an edge $e \in E$ from v to w with $\sigma = event(e)$, $x \in preguard(e)$, $y \in postguard(e)$ and for all $i \notin update(e)$, the ith component of x is equal to the ith component of y. We write $s_1 \underset{delayed}{\overset{\sigma}{\longrightarrow}} s_2$ if $s_1 \overset{t}{\to} s_1'$ and $s_1' \overset{\sigma}{\longrightarrow} s_2$ for some $t \geq 0$ and $s_1' \in S$. For $Q \subseteq S$ we write $pre^{delayed,\sigma}(Q)$ to be the set of all states s satisfying $s \underset{delayed}{\overset{\sigma}{\longrightarrow}} s_1 \in Q$ and $pre^{v,delayed,\sigma}(Q) = pre^{delayed,\sigma} \cap S(v)$.

The language $L(s) \subseteq \Sigma^*$ (where Σ^* indicates the set of finite strings from Σ) is defined as follows; if $s_1 \underset{delayed}{\overset{\sigma_1}{\longrightarrow}} s_2 \cdots \cdots s_r \underset{delayed}{\overset{\sigma_r}{\longrightarrow}} s_{r+1}$ then $\sigma_1 \cdots \sigma_r \in L(s_1)$.

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 $act = R$ and *every* location $v \in V$ we have $L(v,x) = L(v,y)$. 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 $\cong$ on $\mathbf{R}^n$ which is known to be at least as fine as language equivalence for act. Given $x = (x_1, \dots x_n), x' = (x_1', \dots x_n') \in \mathbf{R}^n$, assume that $act = \prod_{i=1}^{n} [L_i, U_i] \subseteq \mathbf{R}^n$ and let λ be the least common multiple of the integers in the set $\{L_1, U_1, \dots, L_n, U_n\}$ and define $x \cong x'$ if

- $\forall i \leq n, \uparrow(x_i \frac{\lambda}{L_i}) = \uparrow(x_i' \frac{\lambda}{L_i})$ and
- $\forall i, j \leq n, i \neq j, \downarrow(x_j \frac{\lambda}{U_j} - x_i \frac{\lambda}{L_i}) =$
  $\downarrow(x_j' \frac{\lambda}{U_j} - x_i' \frac{\lambda}{L_i})$.

Here by $\uparrow$ we denote the *ceiling* and by $\downarrow$ the *floor* of $x \in \mathbf{R}^n$.

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 $\cong$ -classes. It is proved in [3, Corollary 7.3.2] that $\cong$ -equivalence implies language equivalence (sufficient condition). Given two states $(v_1, x_1), (v_2, x_2) \in S$ we write $((v_1, x_1), (v_2, x_2)) \in \cong$ to mean $(x_1, x_2) \in \cong$ and $v_1 = v_2$.

Example 1: The partition of the continuous state space for a two dimensional rectangular automaton, based on language equivalence, as defined above is derived. This partition is used in section 4 as well, in order to clarify the notion of the so-called good sets. Consider the rectangular automaton where the continuous dynamics at each location are given by

$\dot{x} = [1\ 2]$ and $\dot{y} = [1\ 2]$. The partition of the state space based on language equivalence classes is shown in the figure 1. The families of sloping lines are given by the equations $y = U_2 / L_1 * x - h * U_2 / \lambda$ and $y = L_2 / U_1 * x + k * L_2 / \lambda$ where h, k are integers. The family of the horizontal lines is given by the equations $y = n * L_2 / \lambda$ and the family of the vertical lines by $x = m * L_1 / \lambda$ where m, n are integers.

Let the event alphabet Σ be partitioned as $\Sigma = \Sigma_c \cup \Sigma_u$, 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 $s \overset{\sigma}{\rightarrow} s'$ and $s \overset{\sigma}{\rightarrow} s''$ then $s' = s''$. Then we will say that A is Σ_c*-deterministic.* This condition can be enforced by assuming that *postguard* (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 $\tau : S \rightarrow \Gamma$ where Γ is the set of control patterns defined to be $\Gamma := \{\gamma \subseteq \Sigma \ : \gamma \supseteq \Sigma_u\}$. Thus a supervisor for A is a function $\tau : V \times \mathbf{R}^n \times \Sigma \rightarrow \{0,1\}$ satisfying $\tau(v, x, \sigma) = 1$ if $\sigma \in \Sigma_u$. We write $s_1 \underset{\tau,delayed}{\overset{\sigma}{\longrightarrow}} s_2$ if $s_1 \overset{t}{\rightarrow} s_1'$ and $s_1' \overset{\sigma}{\longrightarrow} s_2$ for some $t \geq 0$ and $s_1' \in S$ and $\tau(s_1', \sigma) = 1$; also, we write $s_1 \underset{\tau}{\overset{\sigma}{\longrightarrow}} s_2$ if $s_1 \overset{\sigma}{\longrightarrow} s_2$ and $\tau(s_1, \sigma) = 1$. For any subset $\Sigma' \subseteq \Sigma$, the relation $\underset{delayed}{\overset{\Sigma'}{\rightarrow}}$ is the union of the relations $\underset{delayed}{\overset{\sigma}{\rightarrow}}$ for $\sigma \in \Sigma'$; also, $\underset{\tau,delayed}{\overset{\Sigma'}{\rightarrow}}$ is defined similarly.

For any relation $\rightarrow$, the relation $\rightarrow^*$ is the transitive closure of the union of $\rightarrow$ and equality. If $s_0 (\underset{\tau,delayed}{\overset{\Sigma}{\rightarrow}})^* s$ with $s_0 \in init$ then we say that s is ***τ-reachable*** and if $s_0 (\underset{\tau,delayed}{\overset{\Sigma}{\rightarrow}})^* s \overset{time}{\rightarrow} s'$ then s' is ***almost τ-reachable.***

The main problem of this paper is the following; assume that the sets $F, T \subseteq V$ are given. The assumption that these sets are 'whole' vertex sets (and not 'part' of them) is without 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. $\neg s (\underset{\tau,delayed}{\overset{\Sigma}{\longrightarrow}})^* s'$ *for all* $s \in init$ *and* $discrete(s') \in F$ *(*that is, τ avoids F. This defines the ***forbidden state problem*** or a safety property problem) *and*

2. if $s \in S$ is any τ-reachable state, then $s (\underset{\tau,delayed}{\overset{\Sigma}{\rightarrow}})^* s'$ 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 $s \in S$ is merely that s is *almost* τ*-reachable*, and which in other respects is the same as the weak non-blocking problem. The latter problem appears to be more difficult to solve than the weak non-blocking forbidden problem, see section 4 below. 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 $s_0 \in init$, $t_i \geq 0$ and $\sigma_i \in \Sigma$ we have $s_0 \underset{\mu}{\overset{t_1}{\longrightarrow}} s_1 \underset{\mu}{\overset{\sigma_1}{\longrightarrow}} \bar{s}_1 \cdots\cdots s_m \underset{\mu}{\overset{\sigma_m}{\longrightarrow}} \bar{s}_m$ for $\mu = \tau$ then this also holds for $\mu = \tau_{\max}$.

We show that if $s_1 \xrightarrow{\sigma_1} s', s_2 \xrightarrow{\sigma_2} s''$ with $\sigma_1, \sigma_2 \in \Sigma_c$ and s', s'' are $\cong$ -equivalent then $\tau_{\max}(s_1, \sigma_1) = \tau_{\max}(s_2, \sigma_2)$. Based on this we then show that given a $\cong$ -class r then $\tau_{\max}(s_1, \sigma)$ is computable for any $s_1 \in S$,$s_1 \xrightarrow{\sigma} s' \in r$.

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 $s_1 \in S$, whether $s_1 (\xrightarrow[delayed]{\Sigma_u})^* s'$ 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 $(v_1, x_1) \xrightarrow[delayed]{\sigma_1} (v_2, x_2) \cdots \cdots \xrightarrow[delayed]{\sigma_m} (v_{m+1}, x_{m+1})$ and $x_1 \cong x_1'$ then $(v_1, x_1') \xrightarrow[delayed]{\sigma_1} (v_2, x_2') \cdots \cdots \xrightarrow[delayed]{\sigma_m} (v_{m+1}, x_{m+1}')$ for elements $x_i' \in \mathbf{R}^n$. (Note that we do not claim $x_i \cong x_i'$ for $i > 1$.)

Proof. Since $x_1 \cong x_1'$ we must have $(v_1, x_1') \xrightarrow[delayed]{\sigma_1} (v_2', x_2') \cdots \cdots \xrightarrow[delayed]{\sigma_m} (v_{m+1}', x_{m+1}')$. 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* and containing only the locations and edges of A appearing in the statement $(v_1, x_1) \xrightarrow[delayed]{\sigma_1} (v_2, x_2) \cdots \cdots \xrightarrow[delayed]{\sigma_m} (v_{m+1}, x_{m+1})$ (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 $x_1 \cong x_1'$ in A, implies $x_1 \cong x_1'$ in $\overline{A}$, and so for the automaton $\overline{A}$ $L(v, x_1) = L(v, x_1')$.

Definition 2. A statement $(v_1, x_1) \xrightarrow[delayed]{\sigma_1} (v_2, x_2) \cdots \cdots \xrightarrow[delayed]{\sigma_m} (v_{m+1}, x_{m+1})$ 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 $\mathbf{R}^n / \cong$ of equivalence classes.

Definition 3. We define the labelled transition system on the finite set $V \times \mathbf{R}^n / \cong$ 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

$(v_1, x_1) \xrightarrow[delayed]{\sigma} (v_2, x_2)$.

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 $(v_1, x_1) \xrightarrow[delayed]{\sigma_1} (v_2, x_2) \cdots \xrightarrow[delayed]{\sigma_m} (v_{m+1}, x_{m+1})$ *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. For m=1, $(v_1, x_1') \underset{delayed}{\overset{\sigma_1}{\longrightarrow}} (v_2, x_2')$ for some $x_1' \in r_1$, $x_2' \in r_2$ by the definition of $\overset{\sigma}{\Longrightarrow}$. Let $(v_2, x_2') \underset{delayed}{\overset{\sigma_2}{\longrightarrow}} \cdots\cdots \underset{delayed}{\overset{\sigma_m}{\longrightarrow}} (v_{m+1}, x_{m+1}')$ for $x_i' \in \mathbf{R}^n$ following the inductive hypothesis. The result now follows from Lemma 1 and the fact that $x_1' \in r_1$ implies $x_1 \cong x_1'$.

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

3.2 Forbidden state problem

Next we consider the forbidden state problem (without involving non-blocking) and we show that this is decidable. This follows from the decidability of the associated reachability problem. 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 $\neg(s_0(\underset{delayed}{\overset{\Sigma_u}{\longrightarrow}})^* s')$ for $s_0 \in init$, $s' \in F$ (otherwise no solution to the problem could exist). Define the supervisor τ as follows; $\tau(s_1, \sigma) = 0$ if there exists a state $s \in S$ with $s_1 \overset{\sigma}{\longrightarrow} s$ and $s(\underset{delayed}{\overset{\Sigma_u}{\longrightarrow}})^* s'$ with the discrete part of s' in F and $\sigma \in \Sigma_c$. In all other cases let $\tau(s_1, \sigma) = 1$. Then given any other supervisor $\overline{\tau}$ that avoids the vertex set F, $\overline{\tau}$ is not maximally permissive.

Also, if $s_1 \overset{\sigma_1}{\longrightarrow} (v, x_1)$ and $s_2 \overset{\sigma_2}{\longrightarrow} (v, x_2)$ 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 $\cong$ class.*

Proof. Assume first that the defined supervisor does not solve the problem i.e. there exists a path $s_0(\underset{\tau,delayed}{\overset{\Sigma}{\longrightarrow}})^* s'$ with $s_0 \in init$ and $s' \in F$. Then since $\neg(s_0(\underset{delayed}{\overset{\Sigma_u}{\longrightarrow}})^* s')$ by the hypothesis, we infer that $s_1' \underset{\tau,A}{\overset{\sigma}{\longrightarrow}} s''(\underset{\tau,delayed}{\overset{\Sigma_u}{\longrightarrow}})^* s'$ with $\sigma \in \Sigma_c$. However this is impossible, since we then have $s''(\underset{\tau,delayed}{\overset{\Sigma_u}{\longrightarrow}})^* s'$ and thus $\tau(s_1', \sigma) = 0$ by definition.

Next assume that $\overline{\tau}$ is a supervisor and that $s_0 \underset{\mu}{\overset{t_1}{\longrightarrow}} s_1 \underset{\mu}{\overset{\sigma_1}{\longrightarrow}} \overline{s}_1 \cdots\cdots s_m \underset{\mu}{\overset{\sigma_m}{\longrightarrow}} \overline{s}_m$ holds for $\mu = \overline{\tau}$ but not for $\mu = \tau$ and m is minimal such that this condition is satisfied. Thus $\sigma_m \in \Sigma_c$ and $\overline{s}_m(\underset{delayed}{\overset{\Sigma_u}{\longrightarrow}})^* s'$ for some $s' \in F$, by the definition of τ, and this implies that a location in F is accessible from $s_0 \in init$ under $\overline{\tau}$, hence $\overline{\tau}$ 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 $\cong$ classes.

We next show that it is decidable whether we have $\tau(s_1, \sigma) = 1$ or 0 for a transition $s_1 \overset{\sigma}{\longrightarrow} (v, x)$, where $\sigma \in \Sigma_c$, r is a $\cong$ -class, $x \in r$ and $v \in V$. Following the last part of Theorem 5, for each $s_1 \in S$ and event σ the transition $s_1 \overset{\sigma}{\longrightarrow} (v, x)$ mapping into the $\cong$ 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 $\cong$ class to a forbidden state. The decidability follows from Theorem 6 showing that the condition $(v, r_{\cong}(x))(\overset{\Sigma_u}{\Longrightarrow})^* s'$ for some $s' \in F$ can be checked since the set $V \times \mathbf{R}^n / \cong$ is finite.

Theorem 6. It is decidable whether, given a $\cong$ -class r and $v \in V$ we have $s(\underset{delayed}{\overset{\Sigma_u}{\longrightarrow}})^* s'$ 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 is constructed whose initial state set is $\{v, x, x\}$ such that the path $(v,x)(\xrightarrow[delayed]{\Sigma_u})^* s'$, with $s' \in F$, in A' exists if and only if the path $(v,x,x)(\xrightarrow[delayed]{\Sigma_u})^* s'$ in $N_{A'}$ exists, 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].

Remark : In both [3] and [10] it is shown how to construct the timed automaton such that the transition systems of the timed automaton and the given initialised rectangular automaton are isomorphic. Then the timed automaton contains all the reachability information about the rectangular automaton. Here, given the finite partition on the rectangular automaton, the fact that the reachability problem is decidable may follow directly from the above arguments without using the timed automaton.

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 $\cong$ of A.

Intuitively due to the existence of the uncontrollable events we need to identify possible deadlock (blocking) states and remove them. State set backpropagation is needed and for decidability and computability reasons the notion of the so-called *good sets* is introduced. The reason for this is that the considered quotient space based on language equivalence is more 'subtle' than bisimilation and thus some additional properties for this partition are required in order to solve the non-blocking forbidden state problem.

Definition 7. A set of states $Q \subseteq S$, all with the same discrete part $v \in V$, 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 $x \in S\ (v)$ we have $(v, x) \in Q$ ***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 $\cong$ -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 $S\ (v) - Q_1$ 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 $S\ (v) - Q_1$ is a good set, we assume first that $Q_1 = \Theta_A(U', U'')$. Then $S\ (v) - Q_1 = \bigcup_{u \in U'} \Theta_A(\varnothing, \{u\}) \cup \bigcup_{u \in U''} \Theta_A(\{u\}, \varnothing)$ which is good. The general case follows from the fact that intersections of good sets are good.

In the following lemma we show that the operator $pre^{v,delayed,\sigma}$ commutes with the union. For simplicity we consider the continuous part of the operator in one discrete location, $pre^{v,delayed}$, since clearly the result can be extended for the whole operation.

Given $P = \bigcup_{i=1}^{n} P_i$, where P_i are polyhedral sets then

Lemma 9. The predecessor operator of *P* can by computed by $pre^{v,delayed}(P) = \bigcup_{i=1}^{n} pre^{v,delayed}(P_i)$

Proof: $pre^{v,delayed}(P) = pre^{v,delayed}(\bigcup_{i=1}^{n} P_i) = \{x / \exists u \in act\ \exists t, x + ut \in \bigcup_{i=1}^{n} P_i\}$

$= \{x / \exists u \in act\ \exists t, x + ut \in P_1 \vee \ldots\ldots \vee \exists u \in act\ \exists t, x + ut \in P_n\} = \bigcup_{i=1}^{n} pre^{v,delayed}(P_i)$

Lemma 10. Let $Q \subseteq S$ be a good set with discrete part $w \in V$ and assume there is an edge $v \xrightarrow{e} w$, with $event\ (e) = \sigma$. Then $pre^{v,delayed,\sigma}(Q)$ is good.

Proof. We may assume that Q is in fact elementary, since, following Lemma 9, $pre^{v,delayed,\sigma}$ 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. This result can be directly derived from the lemma 10 above and the fact that we have a finite partition. However, for completeness and in order to relate it with theorem 6 we prove it involving the timed automaton.

Lemma 11. Given a good set Q, and an edge $v \xrightarrow{e} w$, with $event\ (e) = \sigma$, it is possible to compute $pre^{v,delayed,\sigma}(Q)$ as unions of $\cong$ -classes.

Proof. This proof is sketchy since is using involved notions developed in [3]. We may assume that Q is elementary i.e. $Q = \Theta_A(U', U'')$. 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 from $2^{S_{N_A}}$ to 2^S 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 $\xi \Theta_{N_A}(U', U'') = \Theta_A(U', U'')$ and $\xi\ pre^{v,delayed,\sigma} \Theta_{N_A}(U', U'') = pre^{v,delayed,\sigma} \xi\ \Theta_{N_A}(U', U'')$; 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 (in which time is invisible) simulation class. This equivalence relation, called one-sided region equivalence (one-sided means that each continuous variable is either exclusively bounded from above, or exclusively bounded from below, by preguards), is considerably coarser than the known bisimilation relation, called region equivalence, that is a synchronous (time is visible) bisimulation. Following lemma 10 its preimage $pre^{v,delayed,\sigma} \Theta_{N_A}(U', U'')$ is also a union of asynchronous simulation classes and this set can be computed. Consequently the set $pre^{v,delayed,\sigma} \xi\ \Theta_{N_A}(U', U'')$ can be computed as union of $\cong$ -classes as well.

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 and computability 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 $\cong$ -classes; for in that case there is no guarantee that $pre^{v,delayed,\sigma}(Q)$ is a union of $\cong$ -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 the fact that the backpropagation of these sets can be computed as a union of equivalent classes we can show the termination of the algorithm below in a finite number of steps.

Example 2. In figure 1 it is shown some good sets and some not good sets (using dashed lines) that are unions of equivalence classes. The significance of introducing the good or elementary sets can clearly be demonstrated. Given the language equivalence partition, if the sets are not good or elementary further partition of the state space in order to calculate the backpropagated sets is necessary. Then, since more splitting is required, the decidability and computability of the involved algorithm cannot be shown. In figure 3 the set $pre^{v,delayed}(Q)$ is derived in one location for the rectangular automaton considered in example 1, see [12]. This set is shown using the full black lines for the rectangular set Q.

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

The main algorithm proceeds as follows. It labels at each stage unions of $\cong$ -classes as *acceptable*, *forbidden* or *blank*. These unions are all unions of good sets, since for any vertex $v \in V$ the set $S\ (v) = \Theta_A(\{(v)\}, \varnothing)$ is elementary and therefore steps 1, 2 and 3 below (using lemmas 8, 10 and 11) 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 computability of the involved algorithm in a finite number of steps.
During the algorithm a $\cong$ -class which has been labelled may be relabelled; however a $\cong$ -class which has been labelled *forbidden* will not subsequently change this label.

1. Label all the $\cong$ -classes with discrete part in F *forbidden* and label all other $\cong$ -classes *blank* .
2. Label all the $\cong$ -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 $\cong$ -classes.

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

4. Relabel *forbidden* all $\cong$ -classes which are labelled *blank* , and relabel *blank* all $\cong$ -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 $\cong$ -classes and so all steps of the algorithm except step 5 obviously terminate. Step 5 either increases the number of $\cong$ -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 $\tau_{\max} : V \times \mathbf{R}^n \times \Sigma \to \{0,1\}$ by $\tau_{\max}(s,\sigma) = 0$ if $\sigma \in \Sigma_c$ and $s \xrightarrow{\sigma} s'$ and s' lies in a *forbidden* $\cong$ -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 12. If there are transitions $s_1 \xrightarrow{\sigma_1} s', s_2 \xrightarrow{\sigma_2} s''$ with $\sigma_1, \sigma_2 \in \Sigma_c$ and s', s'' are $\cong$ -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 12 we conclude that given a $\cong$ -class r then the derivation of $\tau_{\max}(s_1, \sigma)$ is decidable for any $s_1 \in S$, $s_1 \xrightarrow{\sigma} (v, x),\ x \in r$.

Theorem 13. 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 $s_0 \xrightarrow[\tau_{\max}]{t_1} s_1 \xrightarrow[\tau_{\max}]{\sigma_1} \bar{s}_1 \cdots\cdots s_m \xrightarrow[\tau_{\max}]{\sigma_m} \bar{s}_m$ with $s_0 \in init$ 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 $\cong$ -class containing $\bar{s}_i$ would become *forbidden* after repeated applications of step 3a in the algorithm, contradicting $s_i \xrightarrow[\tau_{\max}]{\sigma_i} \bar{s}_i$. Next assume that $s_0 \xrightarrow[\tau_{\max}]{t_1} s_1 \xrightarrow[\tau_{\max}]{\sigma_1} \bar{s}_1 \cdots\cdots s_m \xrightarrow[\tau_{\max}]{\sigma_m} \bar{s}_m$ with $s_0 \in init$, and there is no path $\bar{s}_m \xrightarrow[\tau_{\max}]{t_1} \ldots\ldots \xrightarrow[\tau_{\max}]{\sigma} s$ with $discrete(s) \in T$. This implies that the $\cong$ -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 $\cong$ -class containing $\bar{s}_i$ would have become *forbidden* after repeated applications of step 3a, once the $\cong$ -class containing $\bar{s}_m$ had become *forbidden*. This contradicts $s_i \underset{\tau_{\max}}{\overset{\sigma_i}{\longrightarrow}} \bar{s}_i$. 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 is more difficult since the *exact* computation of such blocking states, using the introduced partition, becomes undecidable and removing them may occur on the expense of losing the least-restrictive controlled behaviour. The finite quotient based on language equivalence is defined on the asynchronous transition system where time is treated as a 'silent' or invisible action. However, in timed automata, a synchronous transition system is used, time is *visible* and is modelled as the event {time}, that subsumes all real-valued time durations. This event can be *controllable* or *uncontrollable*, see [8]. The *time-abstract* transition system (where the time it takes to reach one discrete state from another is ignored) has the event set $\Sigma \cup \{time\}$ and involves on the region equivalence quotient space where there exists a finite number of bisimulation classes. It follows that in timed automata the (strong) non-blocking forbidden problem is solvable, see [8], [2], [4].
To explain this further we introduce an illustrative example shown in Figure 3, where within a discrete location some blocking states are shown, two of then are τ*-reachable* and one is *almost* τ*-reachable.* It is clear that the discrete transitions that lead to the first two states can be associated and these transitions can be disabled. The third state is *almost* τ*-reachable* and following the fact that the transition system is asynchronous this state cannot be associated with any discrete transition. Thus, if we need to avoid this deadlock state, we should disable all transitions leading to the discrete location. However, this will result in a conservative (not maximal permissive) controlled behaviour.
It is worth mentioning one case in which a solution to the weak non-blocking problem also solves the strong non-blocking forbidden problem for initialised rectangular automata.

Theorem 14. Assume that $0 \in \mathbf{R}^n$ is an interior point of the activity rectangle act . 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 $s_1 \overset{time}{\longrightarrow} s_2$ with $cts(s_i) = x_i$ then $x_2 = x_1 + at$ with $a \in act$,$t \ge 0$. Choose $\lambda > 0$ small enough so that $-\lambda a \in act$; since $x_1 = x_2 + (-\lambda a)(\frac{t}{\lambda})$, we have $s_2 \overset{time}{\longrightarrow} s_1$. Assume that a state s' is almost τ-reachable; that is, that $s_0 (\underset{\tau,delayed}{\overset{\Sigma}{\longrightarrow}})^* s \overset{time}{\longrightarrow} s'$ for some $s_0 \in init$. Thus s is τ-reachable and so $s (\underset{\tau,delayed}{\overset{\Sigma}{\longrightarrow}})^* s''$ *for* $discrete(s'') \in T$. Since $s' \overset{time}{\longrightarrow} s$, we have $s' (\underset{\tau,delayed}{\overset{\Sigma}{\longrightarrow}})^* s''$, as required.

5. Conclusions

We have derived an algorithm that solves a language optimisation problem. The non-blocking supervisor satisfies safety specifications and is computed using the language equivalence partition for initialised rectangular automata under weak and strong blocking conditions. The compactness condition on A was introduced in order to employ the results in [3] for Theorem 6 and Lemma 11. This condition can be relaxed, see [3, 3.3]. Also, the Σ_c-determinism condition may be unnecessary.
In [16] the supervisory control problem for timed automata has been studied without the use of the bisimulation partition. Here, the supervisory algorithm is derived using a partition. However, having shown its decidability and computability, it is expected that it can be derived without the partition, similar to [16]. Also, the backpropagated sets can be computed using associated techniques involving the software environment of Hytech, see [13], [12].

References

[1] R Alur, D Dill, A theory of timed automata, Theoretical 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, LNCS, 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, October 2000.
[8] M. P. Spathopoulos'On supervisory control for timed automata using urgency' Proceedings of the 9th IEEE International Conference on Methods and Models in Automation and Robotics, Miedzyzdroje, Poland, pp. 863-869, August 2003. ISBN: 83-88764-82-9.
[9] T. A. Henzinger, P. W. Kopke, A. Puri, P. Varaiya 'What's decidable about Hybrid Automata?', Journal of Computer and System Sciences 57, pp. 94-124, 1998.
[10] A. Puri 'Theory of Hybrid systems and Discrete event systems' PhD Thesis, University of California at Berkeley, 1995
[11] H. Wong-Toi 'The synthesis of controllers for linear Hybrid Automata' Proceedings of the 36th IEEE Conference in Decision and Control, pp. 4607-4612, San Diego, 1997.
[12] R. Alur, T. A. Henzinger, P. Ho 'Automatic Symbolic Verification of Embedded Systems', IEEE Transactions on Software Engineering, 22(3), pp. 181-201, 1996.
[13] Hytech. Department of Electrical Engineering and Computer Science, University of California, Berkeley, URL:http://www-cad.eecs.berkeley.edu/~tah/Hytech/
[14] A. Trontis and M. P. Spathopoulos 'Hybrid control synthesis for eventuality specifications using level set methods' International Journal of Control, vol.76, No 16, pp.1599-1627, 2003
[15] J. Lygeros, C. Tomlin, S. Sastry, 'Controllers for reachability specifications for hybrid systems' Automatica: Special issue on hybrid systems, 35, pp. 349-370, 1999.
[16] E. Asarin, O. Maler, A. Pnueli, J. Sifakis, Controller synthesis for timed automata, Proc. IFAC Symposium on System Structure and Control, pp. 469-474, 1998

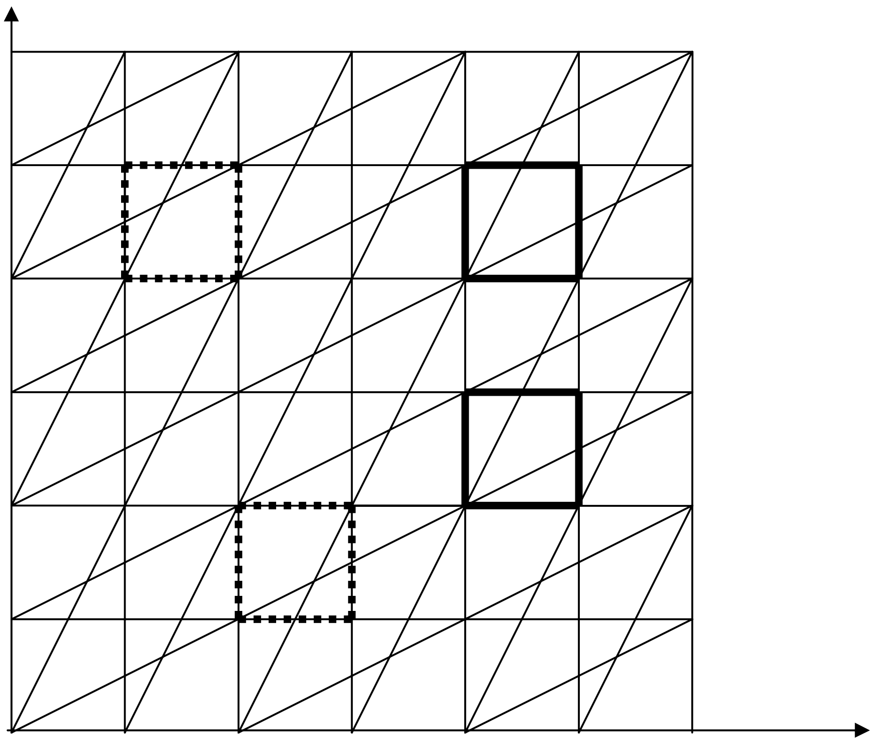

Figure 1: A partition with some good and not good (dashed lines) sets

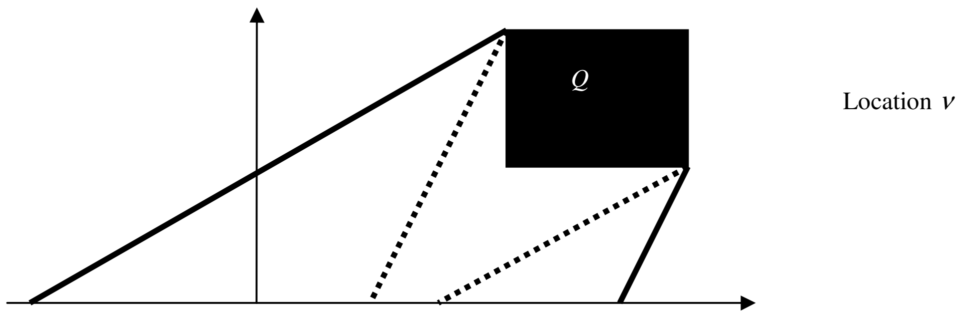

Figure 2: Computation of $pre^{v,delayed}(Q)$

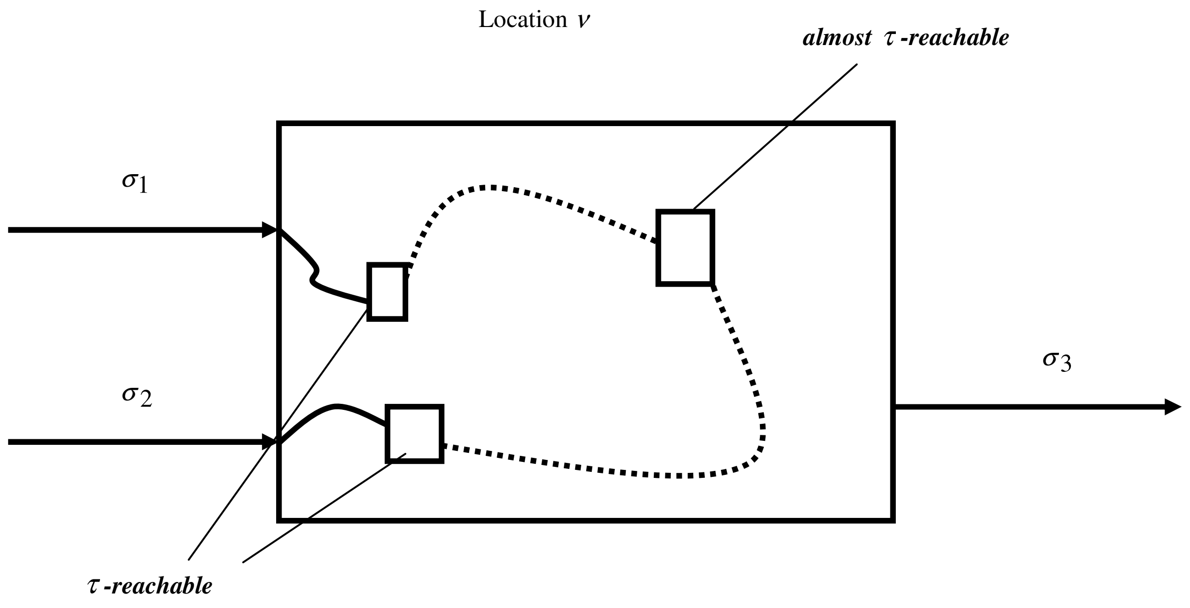

Figure 3