Timed control for rectangular hybrid systems

Sonia Batis, Hassane Alla

Timed control for rectangular hybrid systems

Sonia Batis*. Hassane Alla**

**Gipsa-lab, Grenoble, France, (e-mail : sonia.batis@gipsa-lab.grenoble-inp.fr)*

***Gipsa-lab, Grenoble, France, (e-mail : hassane.alle@gipsa-lab.grenoble-inp.fr)*

Abstract: In this work we develop an approach for the synthesis of the timed control of a class of hybrid systems modeled by rectangular hybrid automata. Our approach is within an automatic control view and is based on the reachable state spaces computation for any location. It consists of two steps, the first one being the computation of the control and the second being a step of the control propagation.

Keywords: Timed controller synthesis, rectangular hybrid automaton, forward analysis, reachable state space.

1. INTRODUCTION

In this paper, we study the timed control of a class of systems broadly known as hybrid processes. This type of process characterizes most of the real life automated systems.

This work is in the extension of the Ramadge and Wonham theory (Ramadge et al, 1987). In this theory, Ramadge and Wonham introduced an approach for the control of discrete event systems. These systems are modeled as generators of formal languages; the adjunction of a control structure allows limiting the language generated by the system by accordingly enabling and disabling events. Brandin and Wonham (Brandin et al, 1992) have then extended the theory to the timed discrete event systems by adding discrete timing features to the initial systems.

Maler (Maler et al, 1995) introduced an extension to the Ramadge and Wonham theory, working in the timed automaton framework (suggested by Alur and Dill (Alur et al, 1994)) and defining the notion of timed games. Later, Cassez (Cassez et al, 2005) used this notion to introduce an efficient on-the-fly algorithm for the analysis of timed automata.

Lots of automated systems evolve according to continuous sub processes which are started and stopped by discrete state orders. Therefore, processes have rarely a purely discrete or continuous behavior but a mixture between both of them. These dynamical systems with a double behavioral component are called hybrid systems and can be modeled by many tools among which there is the rectangular hybrid automaton (Henzinger, 1996). This model can be considered as a generalization of the timed automaton model (Alur et al, 1994), where the continuous evolution is no longer represented by clocks but by differential equations on the continuous variables of the system.

Henzinger (Henzinger et al, 1999) has studied the control problem of rectangular hybrid automata by introducing the hybrid games. The game proceeds in an infinite sequence of rounds and produces an ω-sequence of states. In each round, both players independently choose enabled moves; the pair of chosen moves either results in a discrete state change, or in a passage of time during which the continuous state evolves. In the special case of a rectangular game, the enabling condition of each move is a rectangular region of continuous state, and when time advances, then the derivative of each continuous variable is governed by a constant differential inclusion. The control problem for hybrid games asks: giving a hybrid game B and a formula φ over the discrete states of B, is there a strategy for player 1 so that all possible outcomes of the game satisfy φ?

Spathopoulos (Spathopoulos, 2000) considered the problem of supervisory control for rectangular hybrid automata and established a supervisory controller that can disable only discrete-event transitions in order to solve the non-blocking forbidden state problem.

All these approaches are within a computer science point of view. Our work is within an automatic control point view, and as researchers in this domain, we need to synthesize an effectively implementable control. This requires realistic assumptions.

In this article, we present an approach for the synthesis of the timed control of a class of hybrid systems modeled by rectangular hybrid automata. The control is obtained via a global clock and is based on the reachable state spaces computation for any location. Often, in this research field the formal ideas are complex. It is why we start this paper with an intuitive presentation.

It is organized as follows. In section 2, we present an intuitive example to introduce our ideas. In section 3, we define the model of interest and introduce the control specifications.

Section 4 develops our approach of control. The conclusion is given in section 5.

2. INTUITIVE PRESENTATION

We consider a traffic section that can tolerate a maximal number of 200 cars (we suppose that the average distance between two cars is 4 meters, L=4m). The cars arrive in the section with an average speed of 36 km/h and leave it with an average speed of 42 km/h.

The example is shown in Fig. 1.

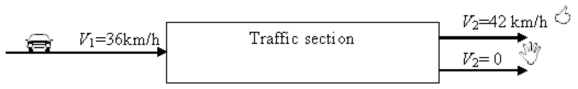

Fig. 1. Example of the traffic section

We suppose that there is a regulation of the traffic that can be done either by a traffic light or by a policeman. Two situations are possible: authorized circulation (V_2=42 km/h) and unauthorized circulation (V_2= 0). The transition from one situation to another can be done at any time, hence we use a clock, denoted by ‘h’ and expressed in minutes. Initially, the number of cars is equal to 80. Then, the example can be represented by the automaton shown in Fig. 2., taking as state variable x_1 the number of cars in the section, and considering the flow as given in the expression: $\dot{x_1} = \frac{V_1 - V_2}{L}$.

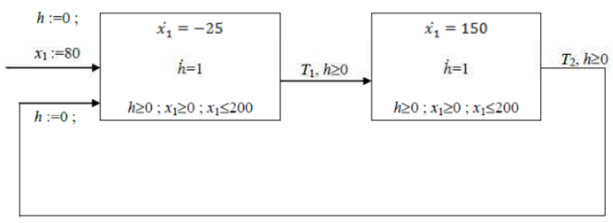

Fig. 2. The traffic section automaton model

Each automaton location is defined by: the variables (x_1 et h), the dynamics of the variables (ex: $\dot{x_1} = -25$) and the invariants (ex: x_1≤200: maximal capacity of the section).

The dynamical behavior of the system can be obtained by a forward analysis of the automaton using the software PHAVer (Frehse, 2008). This will be detailed later. In our case, we have four state spaces, which can be explained by the fact that we have in the system, a transient and a permanent behavior.

The dynamics and the invariants constrain the commutation intervals of the clock. They are computed by optimization and state space computations.

Fig. 3. summarizes the clock intervals obtained in our case:

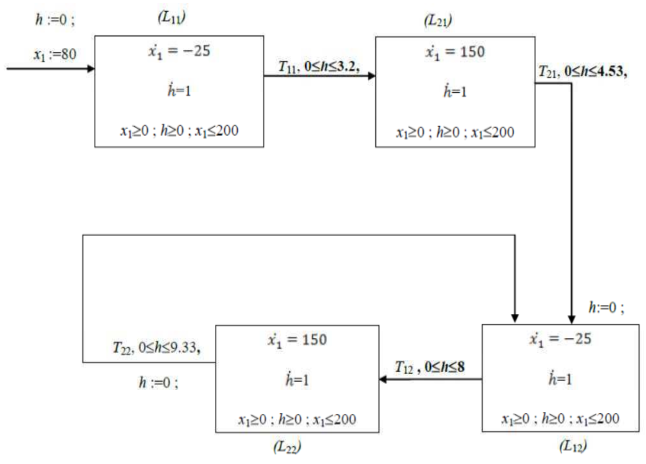

Fig. 3. Dynamical behavior of the system without control

For example, h=3.2 minutes in the transition T_{11} corresponds to the date when the section is completely emptied.

Now, suppose that we need to add a control in order to limit for example the number of cars in the section. We then consider that a policeman wants that: 1) when the circulation is authorized, the section contains at most the quarter of its capacity (x_1≤50), and 2) when the circulation is forbidden, the number of cars is between 30 and 40 (30≤x_1≤40).

All specifications are added to the transitions automaton in italic as shown in Fig.4. below:

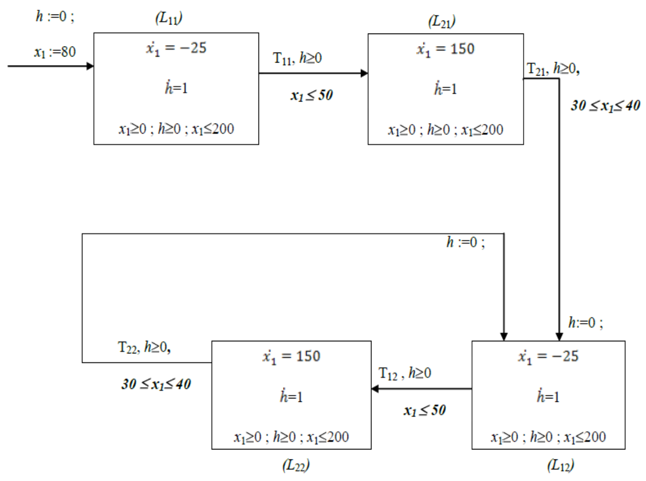

Fig. 4. Constrained automaton model of the traffic section

It is possible to have different specifications then, different guards are added to the automaton. For example, the control strategy specifies that the commutation between locations L_{11} and L_{21} must occur at any time so that the number of cars is less than 50.

The problem is to find the clock intervals of the commutations between the different locations of the system, and therefore, to reduce the intervals found in the study of the uncontrolled behavior given in Fig. 2. Consequently, we want to determine the maximal permissive behavior.

We first compute the new reachable state spaces of the constrained automaton. Then, we consider location L_{21} since there is a clock reset just after this location so it cannot be influenced by the downstream structure. We determine the new clock values of this location by using optimization programs. For example, the guard between the location L_{21} and L_{12} becomes $1.6 \leq h \leq 3.47$.

It remains one last problem: the non-feasibility of the constraint, i.e., the constraint may not be achievable by the dynamic and the proof of its non-feasibility will be translated by the computation of the minimal duration of stay which can be negative.

For example, at Location L_{21}, the maximal input value of x_1 is 50 and the maximal output value is 40. This contradiction will be modeled by a minimal duration of stay, denoted δ, which is negative here ($\delta_{21} = \frac{40-50}{300} < 0$). This needs to modify the input transition guard of L_{21}, which will be reduced to $x_1 \leq 40$. This gives the final control concerning Location L_{21}, it is represented as in Fig. 5. All these ideas are formalized in the sequel.

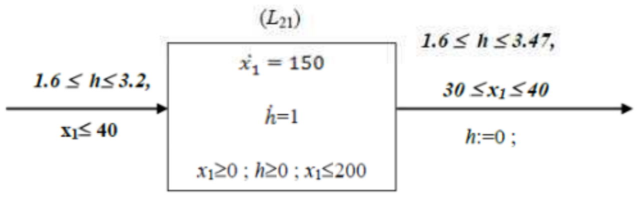

Fig. 5. Final control for L_{21}

3. PROCESS AND SPECIFICATION MODELING

In this section, we first define the rectangular hybrid automaton, then, we introduce the control specifications.

3.1 The Rectangular Hybrid Automaton

The model chosen here is the Rectangular Hybrid Automaton (RHA) as shown in the intuitive presentation. This model can be considered as a generalization of the Timed Automaton model. In the case of a RHA, the continuous evolution is no longer represented by clocks, but by differential equations on the continuous variables of the system.

We define the considered RHA model based on the definition of Henzinger (Henzinger, 1996).

Definition 1: **The general model**

A Rectangular Hybrid Automaton is defined by the tuple $H=(Q, X \cup \{h\}, \Sigma, E, inv, flow, init)$, where:

- $Q=\{q_1,\ldots,q_k\}$ is a finite set of locations representing the discrete states of the system;
- $X \cup \{h\}$ is a finite set of real variables and h is a global clock.
- Σ is a set of events;
- $E \subseteq Q \times \Sigma \times Rect\,(X \cup \{h\}) \times Rect\,(X \cup \{h\}) \times 2^X \times Q$ is a finite set of transitions, where $Rect\,(X \cup \{h\})$ being the set of rectangular predicates on $X \cup \{h\}$).
- *inv*: $Q \rightarrow Rect\,(X \cup \{h\})$ is a function which associates with each location $q \in Q$ a rectangular constraint for any variable $x_i \in X$ and for the clock h. The system can stay in a location as long as the invariant of the location is satisfied.
- *flow*: $Q \rightarrow Rect\,(\dot{X} \cup \{\dot{h} = 1\})$ is the function that assign to each location a representation for the continuous evolution.
- *init* $\subseteq Q \times Rect(X \cup \{h\})$, is the initial condition of the automaton
- Each loop of the automaton has at least a transition guard that contains a reset of the global clock.

❑

Remark 1: The last point of the definition will guarantee the termination of the computation algorithm of the reachable state spaces. It is an assumption that we have always encountered in cases studies.

At each time, the state of a RHA is given by a pair (q, $\mathbf{v}$) corresponding to the association of a discrete state of the system q and a vector $\mathbf{v} \in \mathbb{R}^n$ indicating the current value of each variable.

Example: The system shown in Fig 3 is modeled by a rectangular hybrid automaton with a continuous variable x_1 and the global clock h. The example fully meets the definition. In fact, there is a clock reset in the loop.

3.2 Control specification

The goal of the control is to impose some desired constraints, indicated on the transition guards. Consequently, it is necessary to search the date at which the commutations between the states have to be made. This control must be maximal permissive.

While imposing certain constraints to the location, the system commutes at a certain moment that reduces the state space (Fig. 6.). This joins the idea of forbidden states, introduced by Ramadge (Ramadge et al, 1987) in the classical supervisory control.

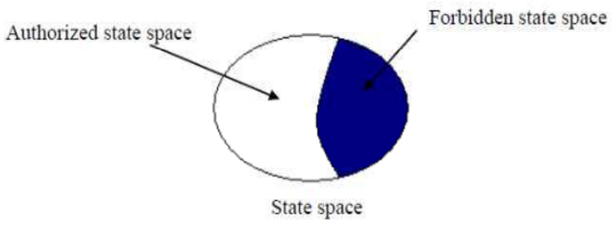

Fig. 6.State space structure

Definition 2: Constraints specification: Given a set of variables X, the set of constraints on X, denoted $C(X)$, is defined by the grammar:

$$\psi ::= x \sim c \,|x - y \sim c|\, \psi \wedge \psi |\neg \psi|\ \text{true}$$

Where $x, y \in X$, $c \in \mathbb{Z}$, $\sim\ \in \{<, \leq, =, \geq, >\}$.

❑

In the sequel, the automaton to which are added the desired constraints in the transition guards will be called the constrained automaton.

Definition 3: Automaton region: A region (or symbolic state) of a RHA is represented by a pair (q, z), where q corresponds to a location of an automaton and z corresponds to a region of the continuous state space, represented by a polyhedron.

❑

Definition 4: Continuous successor: The set of reachable states from any state $(q, \mathbf{v}) \in (q, z)$ letting the time pass while staying in the same location is called the continuous successor of the region (q, z) and is computed using the operator $Post_c$, defined as follows:

$$Post_c((q, z))=\{ (q, \mathbf{v'})|\ (q, \mathbf{v})\xrightarrow{\delta}(q, \mathbf{v'}),\ \mathbf{v}\in z,\ \delta \in \mathbb{R}_+\}$$

❑

Definition 5: Discrete successor: The set of reachable states from any state $(q, \mathbf{v}) \in (q, z)$ firing the transition $e \in E$ is called the discrete successor of the region (q, z) and is computed using the operator $Post_d$, defined as follows:

$$Post_d\,((q, z), e) =\{((q', \mathbf{v'})|\ (q, \mathbf{v})\xrightarrow{e}(q', \mathbf{v'}),\ \mathbf{v}\in z\}$$

❑

In order to perform the control, we first determine the dynamical behavior of the constrained automaton. We realize a forward analysis on the automaton using the software PHAVer. This software provides commands for computing reachable sets of states and simulation relations plus a number of commands for the manipulation and output of data structure. Its language is as user friendly as possible.

The forward analysis operators are used in order to compute all possible trajectories of the system. This leads to compute the state spaces associated withe the stays of the system in each location of the model. One location can be reached with different continuous state spaces at its entrance, especially when the model contains cycles. Consequently, the state space associated with a location is equal to the conjunction of the continuous state spaces of all possible visits to the location.

The forward analysis will provide the reachable state spaces of the constrained automaton. Each state space is characterized by a certain number of inequalities giving the relationships between the different state variables of the system. These relations respect the invariants and specification constraints of each location. The obtained state spaces are smaller than the initial state spaces of the uncontrolled automaton (in Fig 6, it is shown as the authorized state space).

Example: The state spaces obtained from the forward analysis of the constrained automaton shown in Fig. 4. are given in Fig. 7. below. The constraints are shown in italic. For example, in the transition between Locations L_{21} and L_{12}, we need to authorize the traffic when the number of cars is between 30 and 40. This interval corresponds here to a tolerance.

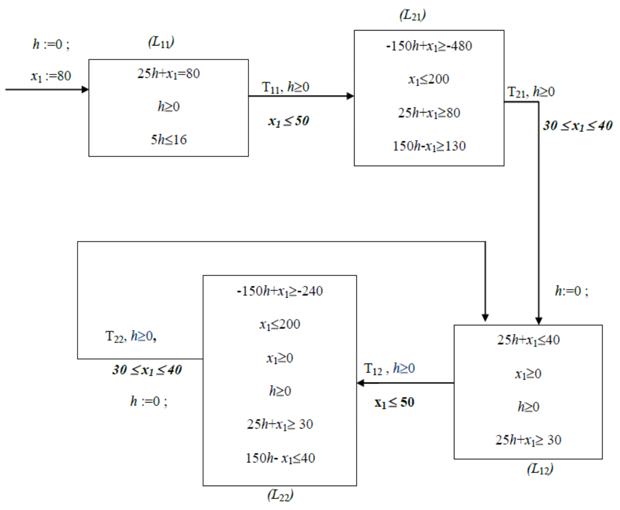

Fig. 7. Forward analysis of the constrained automaton

For example, Location L_{21} is characterized by the following inequalities:

$$150h\text{-}x_1\leq 480;\ x_1\leq 200;\ 25h+x_1\geq 80;\ 150h\text{-}x_1\geq 130$$

They express the linear constraints between the variable x_1 and the global clock h. They provide a bijective characterization of the state space and this formalization will allow us to compute the values of the control clock.

4. COMPUTATION OF THE CONTROL

Finding the maximal permissive controller consists in finding all maximal guards of commutation of the clocks that respect both the invariants and the constraints. The base of the computation consists in solving the general problem for one location of the automaton, which model is shown in Fig. 8.:

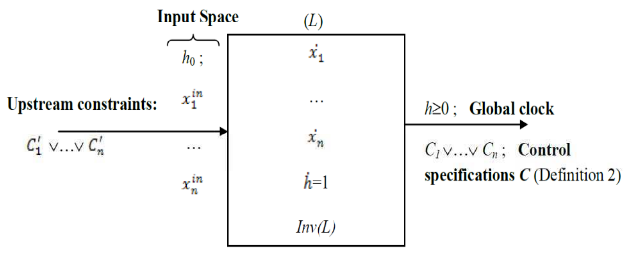

Fig. 8. Location control

Where, for each $i \in \{1,\ldots,n\}$:

- x_i : the i^{th} variable of the system
- h : the global clock
- h^{in}: the initial value of the global clock at the entrance of Location L
- x_i^{in} : the initial entrance state space of Location L for the i^{th} variable. $x_i^{in} \in [\min x_i^{in}; \max x_i^{in}]$.
- C_i: the desired constraint for the variable x_i downstream the location (L). $C_i \in [\min C_i ; \max C_i]$. The set of all constraints corresponding to Location L is denoted C.
- C_i': the constraint for the variable x_i upstream the location (L). $C_i' \in [\min C_i' ; \max C_i']$
- $\dot{x}_i$: the flow. $\dot{x}_i \in [\min \dot{x}_i; \max \dot{x}_i]$
- $inv\ (L)$: location invariants

It is obvious that $min\ (\dot{x}_i)$ and $max\ (\dot{x}_i)$ are with the same sign. This characterization corresponds to the behavior of most real-life systems. In the following, we will denote by E the state space of a location L. E is obtained by the forward analysis defined in 3.2.

Our control approach is composed of two major steps. The first one consists in the computation of the commutation interval respecting the constraints and the second one will go back the structure to modify the constraints given by the modeling.

4.1 Step 1: Computation of the value of the clock control

Proposition 1: The clock commutation interval from location L is $h=[h_{min}; h_{max}]$. It is computed using the linear programming:

- $\boldsymbol{h_{min}}$ **= Minimum** $\boldsymbol{h}$ **With the constraints:** $\boldsymbol{E \wedge C}$
- $\boldsymbol{h_{max}}$ **= Maximum** $\boldsymbol{h}$ **With the constraints:** $\boldsymbol{E \wedge C}$

❑

Example: for the computation of the clock values related to Location L_{21} which state space is shown in Fig. 7., the corresponding linear program is as follows:

- h_{min} = Minimum h with the constraints: $E=\{150h-x_1\leq 480; x_1\leq 200; 25h+x_1\geq 80; 150h-x_1\geq 130\} \wedge C=\{\boldsymbol{x_1\leq 40; x_1\geq 30}\}$
- h_{max} = Maximum h with the constraints: $E=\{150h - x_1\leq 480; x_1\leq 200; 25h+x_1\geq 80; 150h-x_1\geq 130\} \wedge C=\{\boldsymbol{x_1\leq 40; x_1\geq 30}\}$

We find $\boldsymbol{1.6\leq h\leq 3.47}$

For example, the maximal bound h=3.47 corresponds to the maximal constraint value x_1=40. This bound is derived only from the first inequality ($150h - x_1\leq 480$).

4.2 Step 2: Upstream control

The constraints are defined regardless of the process. There is then no guarantee on their achievability. In fact, they can be inconsistent with the dynamic of the location. This amounts to calculate the minimal tolerated duration of stay in the location in question, denoted as δ_i, for each variable x_i. This value has to be positive, since it is a time value. It can happen that an input value of the variable is greater than the output value (with a positive flow in the location), then $\delta_i<0$. In this case, we must modify the constraints upstream the location.

It follows that, for the computation of δ_i, two cases are possible, depending on the sign of the flow in the location. If we obtain at least one $\delta_i<0$, the new constraint on variable x_i upstream the location must be modified as it is given in the following theorem.

Theorem 1: The new desired constraint C_i' for each variable x_i ($i\in\{1,\ldots,n\}$) is:

- $\max(C_i')=\max(C_i)$ if $\dot{x}_i>0$ and $\delta_i = \frac{\max(C_i) - \max(x_i^{in})}{\max(\dot{x}_i)} <0$
- $\min(C_i')= \min(C_i)$ if $\dot{x}_i<0$ and $\delta_i = \frac{\min(C_i) - \min(x_i^{in})}{\min(\dot{x}_i)} <0$
- C_i' remains unchanged otherwise

❑

Intuitive proof: If $\delta_i<0$, there is at least one value of the initial state space such that the constraint is not feasible. In order to obtain a maximal permissive control, It is then necessary and sufficient to remove all the initial space values that make $\delta_i<0$. We are currently working on the formal proof.

Remark 2: The functions max (maximum) that appear in the computation of δ_i (in case we have a positive flow), correspond to the strongest constraints. For example, $\max(C_i)$ corresponds to the maximal tolerated constraint. This is a modeling problem.

Example: The computation of the duration of stay in Location L_{21} of the constrained automaton shown in Fig 4 is as follows, the flow ($\dot{x}_1 = 150$) being positive:

$$\delta_{21} = \frac{40-50}{150} = -0.07 < 0$$

In this case, we replace the constraint $x_1\leq 50$ upstream the location by $x_1\leq 40$

4.3 Concluding remarks

Our main objective is to find a controller for the system that respects in a maximal permissive way the constraints imposed by the operator. We must then be able to repeat what was established for one location to the whole automaton.

That is why we ask an important question which is the starting point of the general algorithm. We must choose a point where there is no possibility of modifying the clock computation. The only possible point is the guard where the global clock is reset. This specification has led us to impose the last point of Definition 1.

This work is in progress in a formal way. However, we determined informally the controller of the traffic section example, which is shown in Fig. 9.:

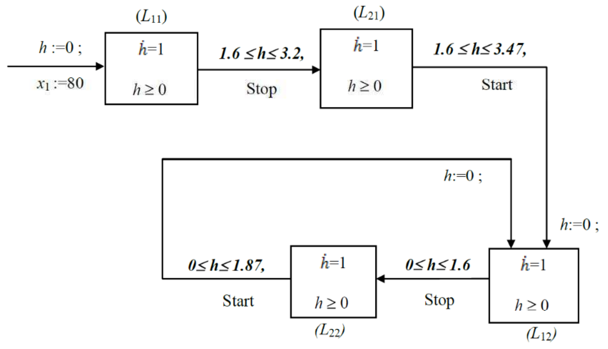

Fig. 9.a. Timed controller of the traffic section

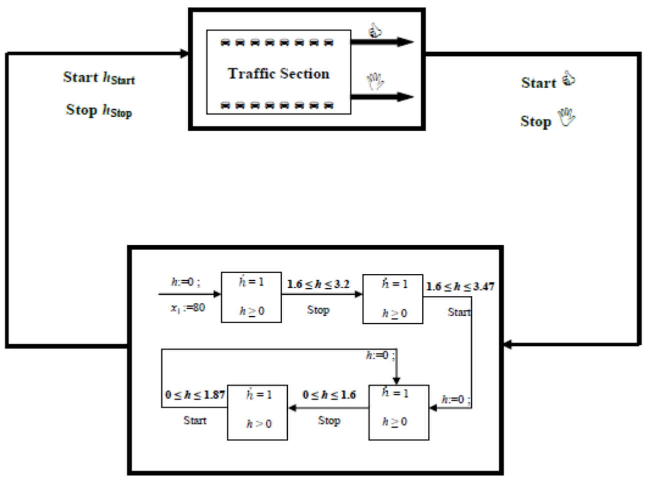

Fig. 9.b. Control structure

This controller is timed; the commutations between two locations are piloted by the clock. For example, the event “Start” will be fired between Locations L_{21} and L_{12} at any moment when the global clock h is between 1.6 and 3.47 minutes.

Remark 3: The final controller model is non-deterministic. There is infinity of solutions that respect the control specification constraints. The implemented controller must be deterministic, then we should find a criterion in order to have one solution. For example, here we can choose as an optimization criterion a minimal downtime of the cars or a balanced transit and downtime.

5. CONCLUSIONS

In this article, we have introduced an approach for the control of a class of hybrid systems modeled by rectangular hybrid automata and characterized by a single global clock. This class of hybrid systems is interesting since it can represent a great class of real life automated systems. The control is based on the computation of the clocks commutation intervals respecting the specifications constraints. They are deduced from the reachable spaces of the process and the constrained behavior. We have focused our presentation on the control in one location since it contains the main original ideas. It consists in two major steps, the first one being the computation of the value of the clock due to the desired constraints imposed by the operator, and the second being of control propagation.

In future papers, we shall develop in a formal way the control approach for the whole system modeled by a rectangular hybrid automaton. The main difficulty will come from the fact that the controller construction must be decidable. Another farther perspective consists in making a combined synthesis between the control and the diagnosis of these systems.

REFERENCES

Alur, R., Dill, D. (1994). A theory of timed automata, *Theoretical Computer Science*, 126(2), 183-235.

Asarin, E., Maler, O., Pnueli, A., Sifakis, J. (1998). Controller Synthesis for Timed Automata, In Proc. IFAC Symp. *On System Structure & Control*, 469-474. Elsevier Science.

Bradin, B.A., Wonham, W.M. (1994). The supervisory control of timed discrete-event systems, *IEEE Transactions on Automatic Control*, 39, 3357-3362.

Cassez, F., David, A., Fleury, E., Larsen, K.G, Lime, D. (2005). Efficient On-the-Fly Algorithms for the Analysis of Timed Games, *CONCUR2005*, 66-80

Frehse, G. (2008). PHAVer: Algorithmic Verification of Hybrid Systems past Hytech, *International journal on software tools for technology transfer (STTT)*, volume 10 number 3.

Henzinger, T.A. (1996). The theory of hybrid automata, *Hybrid Systems II*, LNCS, vol.999, 278-292.

Henzinger, T.A., Kopke, P.W. (1997). Discrete-time control for rectangular hybrid automata, In Proceedings of the 24th International Colloquium on Automata, *Languages and Programming*, Lecture Notes in Computer Science 1256, Spring-Verlag, 582-593.

Henzinger, T.A., Horowitz, B., Majumdar, R. (1999). Rectangular hybrid games, In Proceedings of the Tenth International Conference on *Concurrency Theory*, Lecture Notes in Computer Science 1664, 320-335, Springer-Verlag.

Maler, O., Pnueli, A., Sifakis, J. (1995). On the synthesis of discrete controllers for timed systems, In Proc. 12th Symp. On *Theoretical Aspects of Computer Science* (STACS’95), vomume 900, 229-242. Springer.

Ramadge, P., Wonham, W.M. (1987) Supervisory control of a class of discrete event systems, SIAM, J.*Control and Optimisation*, vol.25, No. 1, 206-230.

Spathopoulos, M.P (2000). Supervisory Control for Rectangular Hybrid Automata, In Proceedings of the 39th IEEE Conference on *Decision and Control*, Sydney, Australia, 35-41.