Continuous flow Systems and Control Methodology Using Hybrid Petri nets

Latéfa Ghomri*, Hassane Alla**

**MELT, University of Tlemcen, Algeria,*
ghomri@mail.univ-tlemcen.dz
*** GIPSA-LAB, University of Grenoble, France,*
Hassane.alla@gipsa-lab.grenoble-inp.fr

Abstract: In this paper, we consider the controller synthesis for continuous flow systems. These lasts are a sub-class of hybrid dynamic systems. Their main characteristics are positiveness and linearity. Transport, manufacturing, communication and biological systems are examples of continuous flow systems. Numerous tools and techniques exist in the literature for modelling and analyzing such systems. As positiveness is a hard constraint, an appropriate tool integrating naturally this constraint is strongly needed. Hybrid Petri Nets are an elegant modeling tool of positive systems, while Hybrid Automata are a powerful tool giving formally the reachable dynamic space. Combining these two tools aim to a sound approach for control synthesis of continuous flow systems. We start by considering the process to control and compute its reachable state space using specialized software like PHAVer. Algebraic inequalities define this reachable state space. The constrained behaviour is obtained by restricting this state space into a smaller desired space. This reduction is expressed in term of linear constraints only over the continuous variables; while the control is given by the discrete transitions (occurrence dates of controllable events). The controller synthesis methodology is based on the control of a hybrid system modelled by a D-elementary hybrid Petri Net. The control consists in modifying the guard of the controllable transitions so as the reachable controlled state space is maximally permissive.

Keywords: Hybrid Petri Nets, Hybrid Automata, controller synthesis.

1. INTRODUCTION

Modelling and control of physical systems are crucial issues. In this work we are interested in a particular class of systems, such as transport, manufacturing, communication and biological systems. These systems have in common that they are positive dynamic systems, i.e., the state variables are positive. Often continuous and discrete event dynamics interact in these systems, so they can be considered as positive hybrid dynamic systems. This class of systems requires for their description, the use of continuous time models like differential equations, and discrete event models like finite state automata or Petri nets (PNs). In general, the state of a hybrid system is given by the discrete mode and the values of the continuous variables. This state may change either continuously, according to a differential equation or discretely by an instantaneous change of the discrete control mode.

The concept of controller synthesis, considered here, has for origin the work of (Ramadge and Wonham, 1989). The latter have introduced controller synthesis for discrete event systems. The process is a discrete event system described by a finite state automaton. The constraints imposed on its behaviour are modelled by any regular language. Both models (process and constraints) allow synthesizing a controller whose role is to prohibit some controllable events in order to always satisfy the specifications. The theory of (Ramadge and Wonham, 1989) has been extended in several directions; one of the major extensions of this theory is the controller synthesis for timed systems. In this case the process model considers time in an explicit manner. Time can either be discretized, in this case the process and its supervisor are modelled by finite state machines equipped with a discrete clock (Brandin and Wonham, 1997); or dense, and in this case the process and its supervisor are modelled by timed automata (Alur and Dill, 1994). Several research studies have been devoted to the controller synthesis of timed systems (Altisen and Tripakis, 1999; Asarin and Maler, 1997; Asarin et al., 1998). The controller synthesis theory is well established for discrete event systems and timed systems. However, it has not yet an explicit solution for hybrid systems, although some studies have been devoted to this field (Wong-Toi, 1997).The difficulty of analysis, in general, and of controller synthesis, in particular, of hybrid systems, is due to the fact that restrictions are needed on the dynamics in order to have an algebraic characterization of the reachable state space. Another difficulty comes from the fact that the computation is not decidable in the general case.

Our aim is to develop a controller synthesis technique for hybrid dynamic systems modelled by a D-elementary Hybrid Petri Nets (HPNs). It is a formal tool with large description capacities and is well appropriate for positive hybrid systems modelling. Its dynamic analysis and performance calculation have been studied in (Ghomri and Alla, 2007). The continuous part of a D-elementary HPN evolves in a

piecewise linear manner, according to differential equations of the form: $\dot{x} = k$, where k is constant. This continuous part is controlled by the discrete part which evolves in an independent manner (i.e. the discrete part evolution is completely independent from the continuous part evolution). Because of the strong interaction between the discrete and the continuous parts and the lack of analysis tools (software) for D-elementary HPNs, we have proposed to translate it into a linear hybrid automaton, which has a great analysis power. In this way, we associate the modelling power of HPNs to the analysis power of hybrid automata. This paper is focused on the control synthesis, for more details on HPNs and hybrid automata the reader may refer to the basic papers (David and Alla, 2010; Alur et al., 1995).

The rest of the paper is organized as follows: Section 2 gives an intuitive presentation of the ideas developed in this paper. Section 3 will be devoted to modelling tools; namely: D-elementary HPNs and hybrid automata, as well as their use in the process description. Section 4 corresponds to the main contribution on this paper; it is divided into two parts. In the first part we will show how to model specifications imposed on the system, and in the second part we will explain the proposed approach to solve the control problem. The problem is formally solved for a location and some directions are given for the synthesis of the whole controller. Finally a conclusion and some perspectives will be given in Section 5.

2. INTUITIVE PRESENTATION

For the control approach that we propose here we start by considering a system whose abstraction is a hybrid dynamic system, and we want to restrict its continuous dynamics by acting on the discrete variables. These dynamics are expressed in term of constraints (also called specifications) over the continuous variables. Restricting the reachable state space needs variables such that the control of these variables will prohibit the state space to reach any undesirable value.

We set here the fundamental hypothesis: 1) the control points are the discrete variables, more precisely the occurrence dates of controllable events, and 2) there is no control in the continuous part. The controller synthesis will take advantage on the coupling between the discrete and continuous dynamics. This is illustrated in the following example.

Example: Consider a producer consumer system composed of a machine that supplies a buffer with a production rate of 20 parts/mn (Figure 1.a). The buffer is used to satisfy a demand of 13 parts/mn. Stop or start the machine is effective after a delay of 2 mns. Initially the buffer contains 50 parts. Figure 1.b shows the D-elementary hybrid PN modelling this system. The discrete part is represented by simple lines and the continuous part by double lines. Transitions T_3 and T_4 represent respectively the discrete events stop and start of the machine. The main advantage of this modelling tool is to show explicitly the physical elements of the process. For example the buffer is only modelled by place P_1 while, as we will see, in the automaton model, this information will be less explicit.

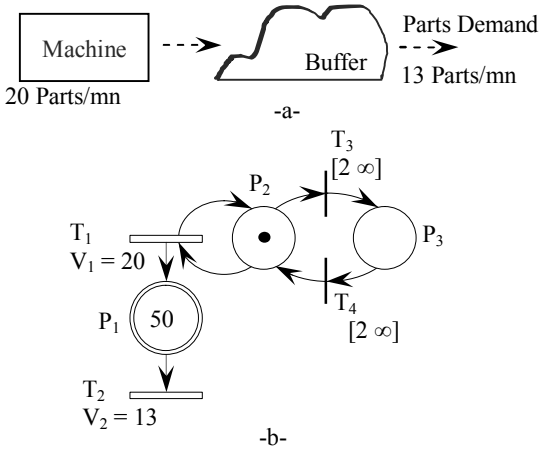


Fig. 1.a. A producer consumer system; b. The D-elementary hybrid PN model of the producer consumer system.

In this example, it is obvious that the number of parts in the buffer can be infinite, i.e. the firing of T_3 is infinitely delayed. Let us suppose that we want to impose to the buffer level to never exceed 100 parts (specification). In order to control the buffer level we must act on the stop date of the machine. Even for this simple case it is difficult to calculate the firing instants of the discrete transitions so that the specification given above is always verified, for any location. The specification presented in this example comes to limit the state space reached by the continuous variable. The translation of a D-elementary HPN in linear hybrid automata allows calculating the reachable space using specialized software like PHAVer (Figure 2.a). In this paper our control objective is to determine formally the new guards of discrete transitions, so that specifications are met. This control must be maximally permissive as we will show in the sequel; This is indicated in the unfolded hybrid automaton in Figure 2.b.

Our control synthesis approach is based on the following three steps:

Modelling the system without constraints by a D-elementary HPN;

Translating the D-elementary HPN in a linear hybrid automaton;

Modelling the specifications and computing the new transitions guards that ensure the specifications compliance;

These three steps are summarized in Figure 3 below. Each block in this Figure corresponds to a step. The two first steps correspond to previous results and the contribution of this paper corresponds to the third step.

3. MODELING OF POSITIVE HYBRID DYNAMIC SYSTEMS

This section corresponds to the two first steps of our approach, i.e. blocks 1 and 2 in Figure 3. As previously mentioned, we are interested here with positive hybrid dynamic systems. We use D-elementary HPNs for modelling

these systems. Not only this formalism inherits all the advantages of classical PNs, but also it is well suitable for the class of systems considered here.

The continuous dynamics are linear in the sense of the linear hybrid automata. HPN is a generic name for a set of formalisms used for modelling hybrid systems. This set has in common the integration of a discrete PN and a continuous PN. D-elementary HPN is a specific HPN which integrates a T-time PN that controls a constant speed continuous PN.

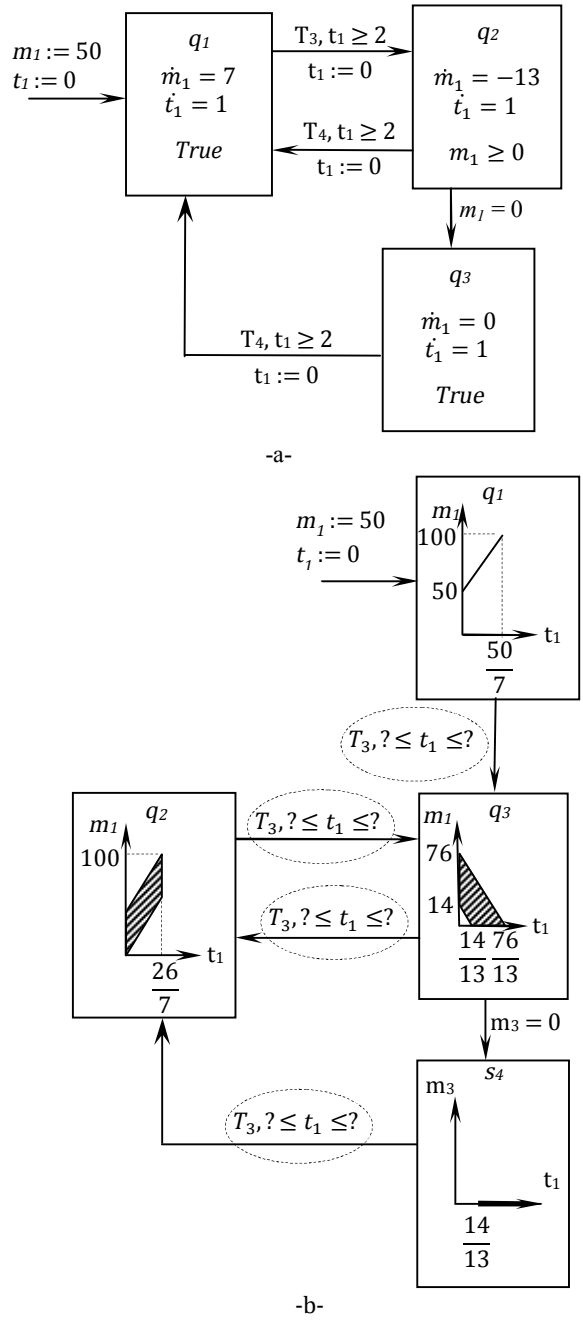


Fig. 2. Hybrid automaton modelling the producer consumer system; a) before control; b) After control.

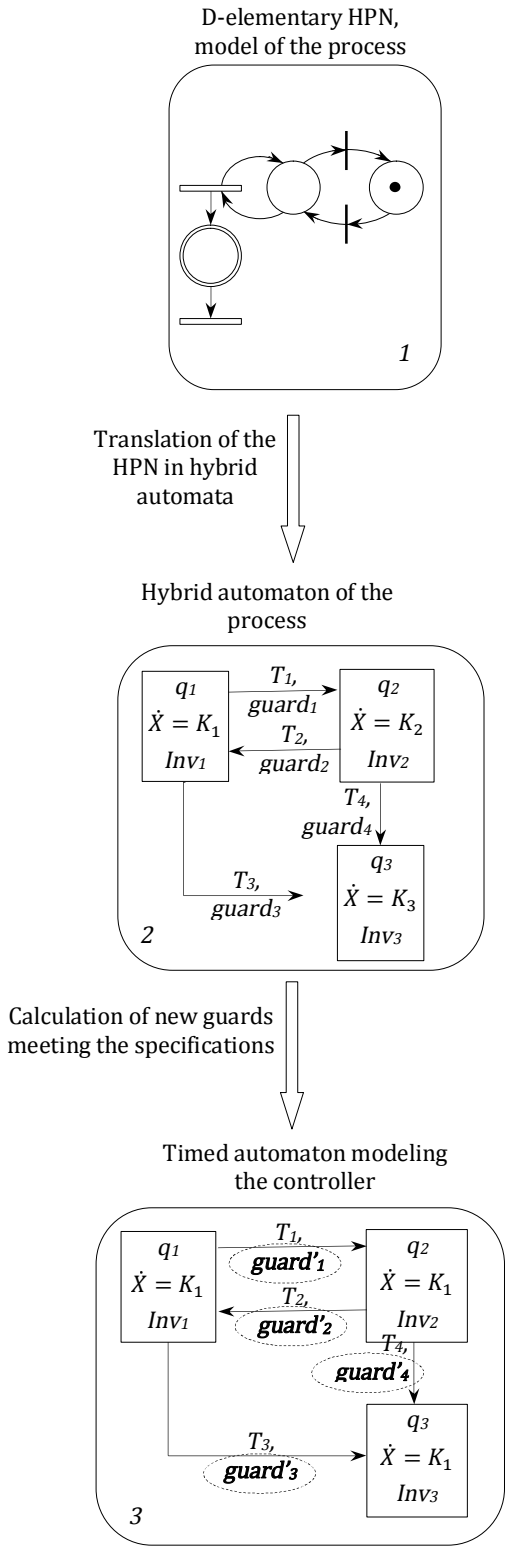


Fig. 3. Controller synthesis approach.

We have supposed that all the discrete transitions are associated with controllable events, and the continuous part is uncontrollable. This represents the current cases met in real life systems, however taking into account uncontrollable events in the discrete part is a direct extension of the proposed approach and constitutes a future research direction.

Definition 1 (D-elementary hybrid Petri nets):

A D-elementary HPN is a structure PN = (P, T, **h**, E, Σ, *Pre*, *Post*, U, V, $\mathbf{m_0}$) such that:

1. $P = \{P_1, P_2, \ldots, P_n\}$ is a finite set of n places; $P = P^C \cup P^D$, such that $P^C = \{P_1, P_2, \ldots, P_{nc}\}$ is the set of n_C continuous places, and $P^D = \{P_{nC+1}, P_{nc+2}, \ldots, P_n\}$ is the set of discrete places.
2. $T = \{T_1, T_2, \ldots, T_m\}$ is a finite set of m transitions; $T = T^C \cup T^D$, such that $T^C = \{T_1, T_2, \ldots, T_{mc}\}$ is the set of m_C continuous transitions, and $T^D = \{T_{mc+1}, T_{mC+2}, \ldots, T_m\}$ is the set of discrete transitions.
3. **h**: P∪T→{C, D} defines the set of continuous nodes, (h(x) = C) and discrete nodes, (h(x) = D).
4. E is a set of events;
5. Σ: $T^D \rightarrow E$ is a function that associates to each discrete transition an event from E;
6. *Pre*: P x $T \rightarrow \mathcal{N}$ and *Post*: P x $T \rightarrow \mathcal{N}$ are the backward and forward incidence mappings. These mappings are such that:

$$\forall (P_i, T_j) \in P^C x T^D, Pre(P_i, T_j) = Post(P_i, T_j) = 0 \quad (1)$$

$$\forall (P_i, T_j) \in P^D x T^C, Pre(P_i, T_j) = Post(P_i, T_j) \quad (2)$$

7. U: $E \rightarrow \mathbb{R}^+$ x $(\mathbb{R}^+ \cup \{\infty\})$ associates to each event e_j its firing interval $[\boldsymbol{\alpha_j}, \boldsymbol{\beta_j}]$.
8. V: $T^C \rightarrow \mathbb{R}^+$ associates a maximal firing speed V_j to each C-transition T_j.
9. $\mathbf{m_0} = \begin{pmatrix} \mathbf{m_{0C}} \\ \mathbf{m_{0D}} \end{pmatrix}$ is the initial marking vector; it is the concatenation of $\mathbf{m_{0C}}$, the real-valued continuous palaces initial marking, and $\mathbf{m_{0D}}$ the natural-valued discrete places initial marking.

The condition (1) of point 6, in the above definition means that no arcs connect continuous places to discrete transitions. Physically, that means that the continuous dynamics has no influence on the discrete dynamics. This last has a fully independent behaviour. The condition (2) in the same point means and if an arc connects a discrete Place P_i to a continuous transition T_j, the arc connecting T_j to P_i must exist. This appears graphically as loops connecting discrete places to continuous transitions. Physically this means that the discrete dynamics controls the evolution of the continuous dynamics. In a D-elementary hybrid Petri net, there is no transformation of tokens, neither from the continuous state to the discrete state, nor in the other sense.

A HPN combines a discrete and a continuous PN, its state at time θ is given by the state of the two models. This strong coupling makes it hard to analyse the hybrid model. The translation of HPNs in hybrid automata allows using formal tools developed for the latter; and it is possible thereby to combine the description power of HPNs to the analysis power of hybrid automata.

Hybrid automata were introduced by (Alur et al., 1995). They are the most general formalism to model hybrid dynamic systems, in the sense that they can model the largest range of this class of systems. They combine: differential equations, which are the basic description model of continuous dynamic systems, and finite state automata that represent the basic model for describing discrete event systems. This is why the hybrid automata are the most used formalism in model-checking and controller synthesis algorithms for hybrid dynamic systems.

To be able to take advantage both from of HPNs and hybrid automata, a translation approach was developed in (Ghomri and Alla, 2007). This approach allows translating D-elementary HPNs into hybrid automata. The translation approach has been proven formally in (Ghomri and Alla, 2013). It is presented in appendix A.

The translation of D-elementary HPN in figure 1.b gives the hybrid automaton of figure 4, that we label elementary hybrid automaton in this paper. It has particular properties, as detailed in the following definition. For a general definition of a linear hybrid automaton, the reader may refer to (Henzinger, 1996).

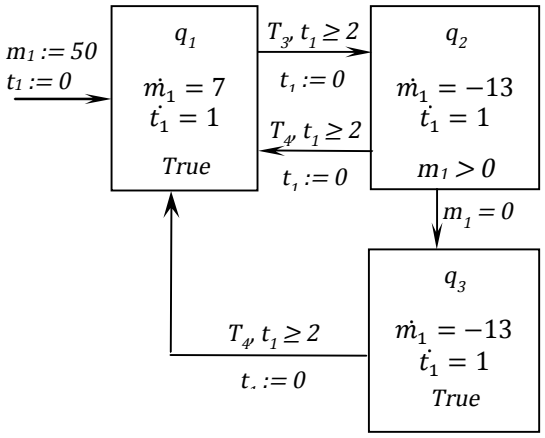


Fig. 4. Hybrid automaton resulting from the translation of the D-elementary HPN in figure 1.b.

Definition 2 (Elementary hybrid automata): An elementary hybrid automaton A = (**Loc**, **X**, E, **δ**, F, Inv) such that:

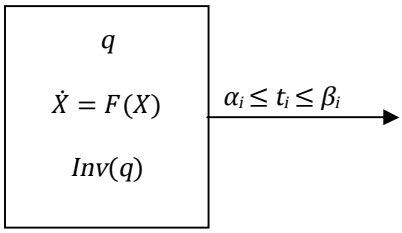
1. Loc = $\{q_1, q_2, \ldots\}$ is a finite set of locations;
2. $X = \begin{pmatrix} X_C \\ X_D \end{pmatrix}$ is the continuous state space. It is composed of two vectors: $X_C = (m_1, m_2, \ldots, m_{nc})^T$ is the vector of nc real-valued variables modelling the continuous places marking; and $X_D = \{t_1, t_2, \ldots, t_k\}$ is the vector of clocks corresponding to enabled transitions. A valuation **v** is a function that assigns a real-valued $v(x) \in \mathbb{R}^n$ to each variable $x \in X$.
3. E is a set of events;

4. δ is a finite set of transitions, each transition is a quintuple T = (q, a, g, γ, q') such that:
   - q ∈ Loc is the source location;
   - a ∈ *E* is the event associated to the firing of T;
   - g is the transition guard, it is a linear predicate on X; a transition can be fired whenever its guard is satisfied.
   - *Init* is a reset function that affects a linear expression to variables of X when taking the corresponding transition;
   - q'∈ Loc is the target location;
5. F is a function that assigns to each location a continuous linear vector field on X. While in discrete location q, the continuous variables $m_i \in \mathbf{m_C}$ evolve according to a differential equation of the form $\dot{m}_i = B_i$, where B_i is a the dynamic balance of the continuous place P_i. and the clocks $t_j \in T_D$ evolves according to the differential equation $\dot{t}_j = 1$.
6. *Inv* is a function that affects to each location q, a linear predicate Inv(q) that must be satisfied by the continuous variables in order to stay in the location q.

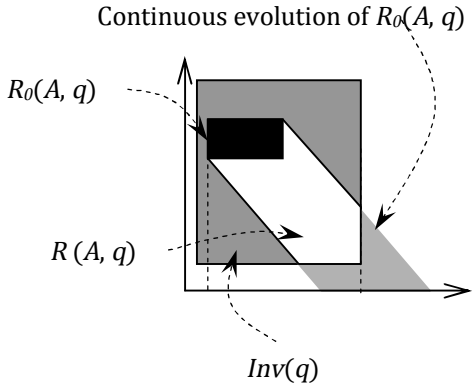
A state of a HA is a pair (q, v) consisting of a location q and a valuation of continuous variables v. This state can evolve in two manners: either continuously; in this case the continuous variables evolves according to the vector field and the discrete location remains unchanged; or discretely by firing a discrete transition. Several problems, related to analysis of HA properties, could be expressed as a reachability problem. Note that this problem is generally undecidable unless restrictions are added to the basic model, to obtain special sub-classes of HA (Henzinger et al., 1995). Our models are often well formed and the translation algorithm always terminates. A future research will consist in characterizing formally the sub class of HPNs such that the translation algorithm terminates always. It is an interesting modelling problem which must be solved.

Remark 1: In the elementary hybrid automaton the transitions guards are function of only one variable since a controllable transition correspond to the firing of a discrete transition in the HPN. They have guards of the form $\alpha_i \le t_i \le \beta_i$; where t_i is a clock and α_i, β_i are real constants. Uncontrollable transitions correspond to emptying a continuous place marking. They have guards of the form $m_i = 0$.

The controller synthesis consists first in solving the problem for a location q and then to iterate the procedure for the whole elementary hybrid automaton. Figure 5.a gives the structural representation of a location; we suppose, in a first time, that q has only one output transition, which is controllable. The guard of a controllable transition is only function of one clock t_i. This guard is of the form: $\alpha_i \le t_i \le \beta_i$, where α_i and β_i are real constants. Figure 5.b schematizes R(A, q) the reachable state space in location q, which depends on the input state space R_0(A, q), the vector Field F(q) and on the invariant Inv(q). We will resume the location q throughout this paper to explain the controller synthesis procedure and to indicate the consequences of each step of the procedure on the reachable state space in q.



-a-



-b-

Fig. 5.a) a location q from the elementary hybrid automaton, b) reachable state space in location q.

4. CONTROLLER SYNTHESIS

The controller synthesis consists in realizing the last step described in block 3 of Figure 3. It will be solved in this paper for a location; therefore we will develop in this section the calculation of the new guards so that the location state space verifies the specifications. Some directions will be given to extend this hard computation problem in order to determine the final controller.

4.1 Control Specifications Modelling

In hybrid dynamic systems, specifications can be imposed on the discrete part or on the continuous part. We consider here specifications on the continuous behaviour; we think that it is the most currently met cases in real life systems. It means that specifications are only related to continuous variables m_i and never on clocks. They have the form of linear inequalities.

Definition 3 (Specification): Let $S = (s_1\ s_2\ \ldots s_{nc})^T$ be a real-valued constant vector and b be a real constant. A specification Spec on the continuous behaviour of the elementary hybrid automaton is a predicate of the form:

$$S_1 m_1 + S_2 m_2 + \cdots + S_{nc} m_{nc} \le b$$

Then $S^T.M_C \le b$

Let us recall that M_C is a vector of nc real-valued variables modelling the continuous places marking (Definition 2).

We note by Spec(q) the general form of a specification, it is the conjunction of all the specifications imposed on the state space in location q:

$Spec(q) = Spec_1 \wedge Spec_2 \wedge \ldots \wedge Speck$

A continuous specification Spec(q) is a set of constraints on the continuous space reached by the HA in any discrete location q. The automaton must stay in q if Spec(q) is satisfied, and must leave q by firing a transition before the violation of Spec(q). Of course, a specification can be different from a location to another one.

An example is given for location q_1 described in Figure 6. Thanks to PHAVer software, the computation of the reachable space in this location is obtained. It is determined by the following set of inequalities:

$$R(A, q_1) = \begin{cases} m_1 - 3t \geq 39 \\ m_2 - t \geq 50 \\ 3m_2 - m_1 \geq 108 \\ m_2 \geq 53 \\ m_1 - 3m_2 \geq 114 \\ m_1 \geq 48 \\ m_2 - t \leq 53 \\ t \geq 1 \\ m_1 - 3t \leq 48 \\ m_1 \leq 100 \end{cases}$$

Let us add the following specification: $m_1 - m_2 \leq 20$, which must be verified at any time. The state space above contains a subset of values that violate this specification. The goal of the next section is to compute the new output guard so that this specification is always verified.

4.2 Control computation

The controller is obtained by modifying the guards on transitions of the elementary hybrid automaton such as the specification are verified in the maximally permissive way. This corresponds to block 3 of Figure 3. This calculation is made using the reachable state space of automaton A.

Let us consider the general case of a location q from A (Figure 7.a) and the controllable output transition T_j whose guard is function of the clock t_i: $\alpha_i \leq t_i \leq \beta_i$. The addressed problem is to calculate the largest interval $[\alpha'_i, \beta'_i] \subseteq [\alpha_i, \beta_i]$ in the dynamic behaviour of the elementary hybrid automaton. i.e. the guard $\alpha'_i \leq t_i \leq \beta'_i$ that allows to meet the specifications.

Each marking evolves according to a linear function when in a specific location since its time derivative is constant, and there is an indeterminism in the initial values of time and marking. This can be formalized as below.

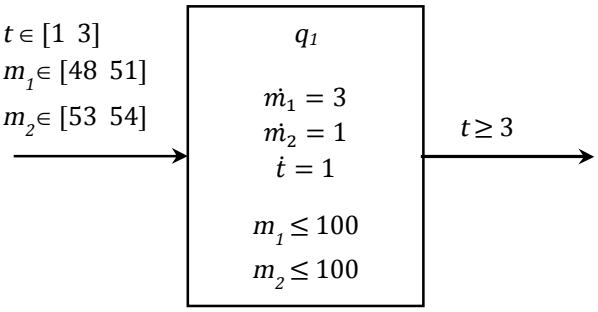
Property 1: In a location q, each continuous variable m_i can be written as:

$$m_i = c_i(t - t_0) + d_{i0}$$

where $t_0 \in [t_{0min}\ t_{0max}]$ and $d_{i0} \in [d_{0imin}\ d_{0imax}]$ are time and marking initial values.

$[t_{0min}\ t_{0max}]$ and $[d_{0imin}\ d_{0imax}]$ are convex intervals given by the input space $R_0(A, q)$,

With $t_{0min}, c_i, d_{0imin} \in \Re^+$ and $t_{0max}, d_{0imax} \in \Re^+ \cup \{\infty\}$

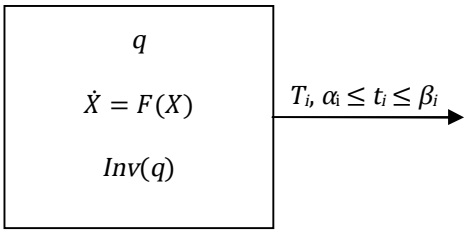


-a-

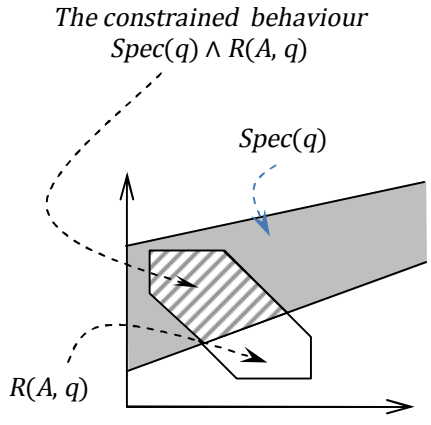
$m_1 - m_2 \leq 20$

-b-

Fig. 6. a) Autonomous behaviour, b) Specification.



-a-



-b-

Fig. 7.a) Location q in A, b) Constrained behaviour.

For any location q in A, We note by:

$C = (c_1\ c_2\ \dots\ c_{nC})^T$ the vector of the m_i variables slopes (dynamics of the markings),

$D_{0min} = (d_{01min}\ d_{02min}\ \dots\ d_{0nCmin})^T$ the vector of minimal input values in $R_0(A,q)$,

$D_{0max} = (d_{01max}\ d_{02max}\ \dots\ d_{0nCmax})^T$ the vector of maximal input values in $R_0(A, q)$,

$$S^- = \begin{pmatrix} \min(s_1, 0) \\ \min(s_2, 0) \\ \vdots \\ \min(s_{nC}, 0) \end{pmatrix} \text{ and } S^+ = \begin{pmatrix} \max(s_1, 0) \\ \max(s_2, 0) \\ \vdots \\ \max(s_{nC}, 0) \end{pmatrix}$$

Where S is the vector of specification decomposed into 2 sub vectors S^- and S^+

Such that: $S = S^- + S^+$;

$d_{\max}$ is the maximal sojourn time in $R(A, q)$.

Theorem 1: The maximal permissive control of a discrete controllable transition T_i meeting the specification: $S^T.M_C \le b$ on the continuous part is given by the new guard: $\alpha'_i \le t_i \le \beta'_i$

Such that:

$\alpha'_i = \max(\alpha_i, t_{imin})$

And

$\beta'_i = \min(\beta_i, t_{imax})$

Where: $t_{imin} = t_{0min}$

And $t_{i\,max}$ is calculated as follows:

- If $S^T.C > 0$ $t_u = \frac{b + S^T C.t_{0\,min} - S^{+T} D_{0max} - S^{-T} D_{0min}}{S^T C}$
  - i. If $t_u < \alpha'_i$ then q is forbidden;
  - ii. If $t_u \ge \alpha'_i$ then $t_{i\,max} = t_u$
- if $S^T.C < 0$ $t_u = \frac{b + S^T C.t_{0\,min} - S^{+T} D_{0min} - S^{-T} D_{0max}}{S^T C}$
  - i. If $t_u > t_{0min}$ then q is forbidden.
  - ii. If $t_u \le t_{0min}$ then $t_{imax} = d_{max}$
- if $S^T.C = 0$,

i. If $S^T D_0 \le b$, the specification is always verified;

ii. If $S^T D_0 > b$, Location q is forbidden.

Proof:

The value of t_{imin} is obvious; it is given by the reachable space from location q in A.

The value of t_{imax} is derived from the specification:

$S^T.M_C \le b$

As (from property 1):

$M_C = C(t - t_0) + D_0$

We can write:

$S^T.(C(t - t_0) + D_0) \le b$

The sign of the scalar $S^T.C$ is very important in the calculation of the maximal value of the time upper bound. It combines the weights in the specification with the slopes of the marking. Depending on the sign of $S^T.C$, three cases must be distinguished:

1st case: $S^T.C > 0 \Rightarrow t \le \frac{b + S^T.C.t_0 - S^T.D_0}{S^T.C}$

is the condition verifying the specification.

And the more constraining bound for t is:

$t_u = \frac{b + S^T C.t_{0min} - S^{+T} D_{0max} - S^{-T} D_{0min}}{S^T.C}$

- If $t_u < \alpha'_i$, then the new guard is empty and location q is forbidden.
- If $t_u \ge \alpha'_i$, then $t_{imax} = t_u$ and $\beta'_i = \min(\beta_i, t_{imax})$

2nd case: $S^T.C < 0$ $\Rightarrow t \ge \frac{b + S^T.C.t_0 - S^T.D_0}{S^T.C}$

is the condition verifying the specification.

And the more constraining bound for t is:

$t_u = \frac{b + S^T.C.t_{0min} - S^{+T}.D_{0min} - S^{-T} D_{0max}}{S^T.C}$

- If $t_u > t_{0\,min}$ then the specification is not verified for the entrance space and location q is forbidden.
- If $t_u \le t_{0\,min}$ then $t_{i\,max} = d_{max}$ (maximal sojourn time for location q in A)

and $\beta'_i = \min(\beta_i, t_{imax})$

3rd case: $S^T.C = 0$

- If $S^T.D_0 \le b$, then the specification is always verified
- If $S^T.D_0 > b$, then the specification is not verified for some values of the entrance marking in location q which is forbidden.

Figure 7 illustrates the two first cases of Theorem 1. In Figure 8.a, the specification:

$m_1 - m_2 \le 20$ gives

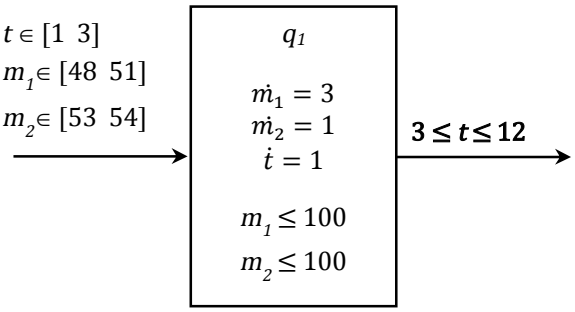
$S^T = (1\ -1)$, and $S^T.C = (1\ -1).\binom{3}{1} = 2$

Then $t_{i\ max} = \frac{20+2-51+53}{2} = 12$ and the new guard is [3, 12].

In Figure 8.b, the specification $m_2 - m_1 \le 20$ gives

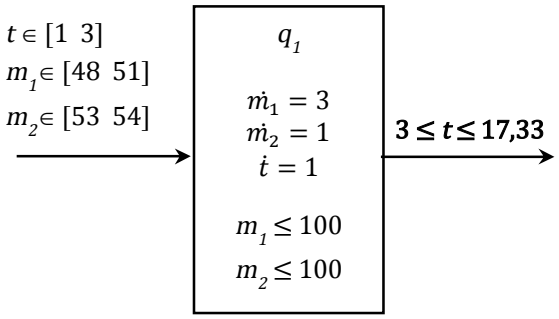
S^T = (-1 1), and $S^T.C$ = (-1 1).$\begin{pmatrix}3\\1\end{pmatrix}$ = -2

Then $t_{i\ max} = \frac{20-2-53+51}{-2}$ = -8; In that case, it is necessary to compute the maximal sojourn time in the location, d_{max}= Min $\left(\frac{100-48}{3}, 100-53\right)$ = 17.33. The new guard is [3, 17.33].



Specification: $m_1 - m_2 \leq 20$

-a-



Specification: $m_2 - m_1 \leq 20$

-b-

Fig. 8. Control computation: a) $S^T.C > 0$, b) $S^T.C < 0$.

Remark 2:

1) In the case of a general specification of the form:

$$Spec(q_i) = Spec_{i1} \wedge Spec_{i2} \wedge \ \ldots \wedge Spec_{ik}$$

Then each $Spec_{ik}$ is studied alone with Theorem 1. The final solution consists in considering the conjunction of the different guards $[\alpha'_{ik}\ \beta'_{ik}]$ in order to obtain the final guard. It can be noticed that the computation of the reachable space of Automaton A is done only once.

2) When a location has several output transitions, each guard is computed as it is alone. There will be several output concurrent transitions.

Our approach is powerful since it gives the control thanks to an algebraic computation. This allows considering any kind of specifications, which can be different from a location to another one. However it needs the knowledge of the intervals of the different variables of the entrance space and in some cases the maximal sojourn time in a location. These intervals are given using linear programming on the reachable spaces.

We have soled formally the problem for a location. In order to have the complete controller, it is necessary to iterate the formal procedure for the whole automaton, leading to a decidability problem. It is well known that for a hybrid automaton the computation of the reachability spaces does not terminate in general. However, real life systems own good properties that make this computation decidable. Then it will be interesting to characterize the set of systems for which the controller synthesis is possible. This constitutes our future research.

Example:

Let us consider again the producer consumer system and the specification that imposes to never exceed 100 parts in the buffer. The controller guarantying to meet this specification is schematized in Figure 9. This controller is built location by location applying Theorem 1. A control is computed for one input state pace, called a visit. It is why he final controller given in Figure 9 is an unfolded automaton. This model is only timed since the control points are given by the discrete timed transitions. The continuous dynamic does not appear, its influence is taken into account in the computation of the guard. By acting on the discrete controls Start and Stop in the control timed intervals, the respect of the specification is guaranteed.

5. CONCLUSION

In this paper, we have presented a controller synthesis technique for hybrid dynamic systems modelled by D-elementary HPNs. This model is first translated in a hybrid automaton before controller synthesis. We have highlighted our contribution on the computation of the control for a location. Specifications are added to the hybrid automaton in order to limit its reachable state space to a desired one.

This is obtained thanks to the computation of new guards associated with the controllable discrete transitions. We have determined algebraic formulas giving the new time bounds that allow respecting the specifications. This computation needs the bounds of the different variables for the input state space, which are obtained by linear programming. The controller is a timed automaton; it is optimal in the sense that it gives the most permissive state space of the system guarantying the specifications. This optimality is obtained via an algebraic calculation.

The obtained timed automaton is the model of the closed loop system. It represents the system coupled to its controller, and has a nondeterministic behaviour. It is possible to choose one deterministic behaviour from an infinite number. It is also possible to calculate an optimal behaviour according to a given criterion.

Our future work consists in: 1) establishing a general algorithm for the automatic determination of the controller,

starting from any D-elementary HPN; 2) Taking into account uncontrollable events in the discrete part; and 3) characterizing formally the sub class of HPNs such that the translation algorithm terminates always.

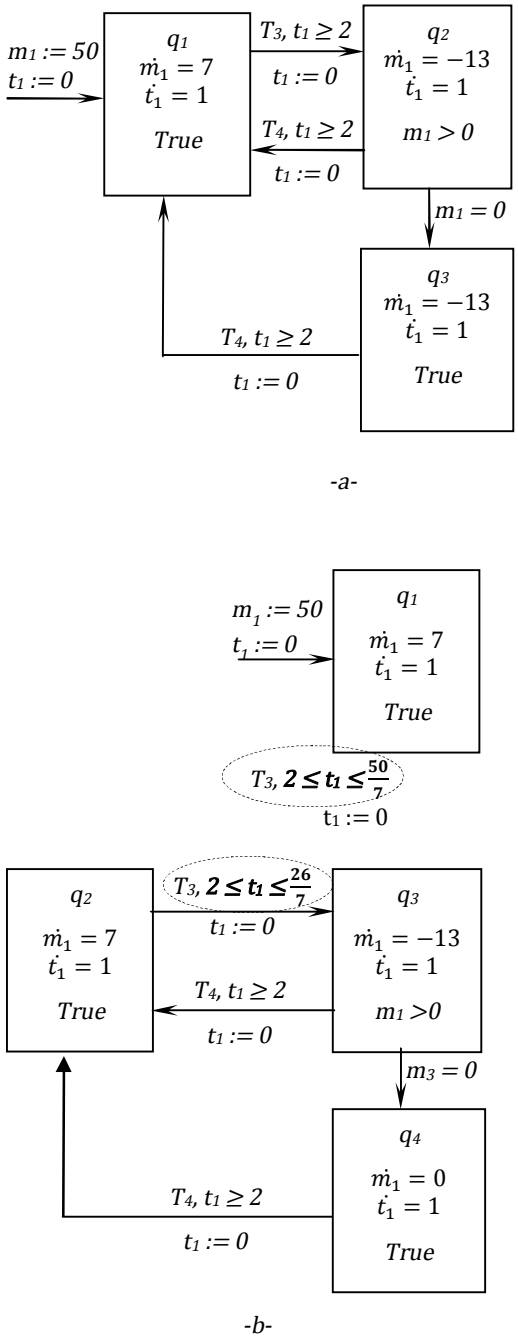


Fig. 9. Controller of the consumer producer system.

REFERENCES

Altisen.K. and Tripakis. S., (1999), On-the-Fly Controller Synthesis for Discrete and Dense-Time Systems, *Proceedings of the World Congress on Formal Methods in the Development of Computing System* (FM'99), 1708 de Lecture Notes in Computer Science, p. 233-252,

Alur, R., and D. L. Dill, A theory of timed automata, Theoretical Computer Sciences, 126: 183-235, 1994.

Alur, R., C. Courcoubetis, N. Halbwchs, T.A. Henzinger, P.H.Ho, X. Nicolin, A. Olivero, J. Sifakis and S. Yovine, The AlgorithmicAnalysis of Hybrid Systems, Theoretical computer science, Vol. 138,pp.3-34, 1995.

Alur. R., Courcoubetis. C., Henzinger. T.A. and Ho. P.H., Hybrid Automata: An algorithmic approach to the Specification and verification of of hybrid systems. In Hybrid Systems, LNCS 736, pp. 209 – 229, 1003;

Asarin E., Maler O., (1999)*As Soon as Possible: Time Optimal Control for Timed Automata*, Proceedings of the 2nd International Workshop Hybrid Systems, Computation and Control (HSCC'99), vol. 1569 de Lecture Notes in Computer Science, p. 19-30, Springer, 1999.

Asarin E., Maler O., Pnueli A.andSifakis J., (1998), Controller Synthesis for Timed Automata, *Proceedings of IFAC Symposium on System Structure and Control*, p. 469-474, Elsevier Science, 1998.

Berthomieu.B. and Diaz. M. (1991) Modelling and verification of time dependant systems using time Petri nets, IEEE transactions on software engineering 17(3), 259 – 273;

Brandin.B. and Wonham. M. W., (1994), Supervisory control of timed discrete event systems*, IEEE, transactions on automatic control*, 39(2), 329 – 341.

Cassez, F.; & Roux, O.H. (2003) Traduction Structurelle des Réseaux de Petri Temporel vers les Automates temporisés, Proceedings of 4ième colloque Francophone sur la modélisation des systèmes réactifs, (MSR'03), Metz, France

David. R. and Alla. H., Discrete, Continuous and Hybrid Petri Nets, second edition, 2010, Springer, Heidelberg Germany;

Ghomri. L. and Alla. H., Using D-elementary hybrid Petri nets and linear hybrid automata for modelling manufacturing systems, MCPL, Fortaleza, Ceará, Brazil, September 11-13, 2013.

Ghomri.L. and Alla. H., Modelling and analysis using hybrid Petri nets, Non linear analysis: hybrid systems, 1(2), 141 – 153;

Henzinger, T. A., The theory of hybrid automata.In Annual Symposium on Logic in Computer Science, pages 278{292. IEEE Computer Society Press, 1996.

Henzinger, T.A., Kopke. P.W.,Puri. A, and P. Varaiya. What's decidableabout hybrid automata? In Proceedings of the 27th Annual Symposium onTheory of Computing, pages 373{382.ACM Press, 1995.

Ramadge. J. G. and Wonham. W. M., (1989),The Control of Discrete Event Systems,*Proceedings of the IEEE,* 77(1),81-97.

Wong-Toi, H. (1997), The synthesis of controllers for linear hybrid automata. Proceedings of the 36th Conference on Decision and Control. IEEE Computer Society Press, 4607 – 4612

APPENDIX A: Translation approach.

The translation approach can be summarized in the following three steps:

1. Isolate the discrete PN of the hybrid model and construct its equivalent timed automaton: Several algorithms permitting the translation of a Time PN in a timed automaton exist in the literature. We reuse the existing algorithms for this first step of our approach. Locations of the resulting timed automaton are said macro-locations in the following.

2. Construct the hybrid automaton corresponding to each macro-location of the timed automaton resulting from the previous step: Indeed, a macro-location represents a marking of the discrete PN, and therefore a configuration (set of locations) of the system. In this second step we determine the continuous PN corresponding to each macro-location and we translate it in a hybrid automaton. At the end of this second step we have a hierarchical model containing macro-locations, each of which comprises a hybrid automaton.

3. Replace transitions between macro-locations by transitions between internal locations: This last step transforms the hierarchical model of the previous step in a linear hybrid automaton.

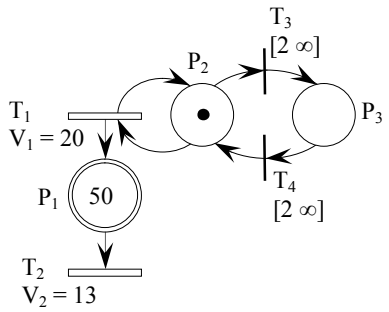
To illustrate the different steps of the translation approach, let us consider again the D-elementary HPN representing the producer consumer system in Figure A1.a.

In Figure A1.b below we isolate the discrete part of the HPN of Figure A1.a. The time PN obtained has two reachable markings and only one transition enabled at a time. It is modelled by a timed automaton with two locations, that represent the markings, and one clock, representing the time during which a transition is enabled (Figure A1.c).

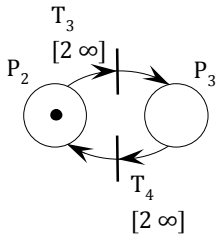
To each marking of the discrete part (time Petri net) corresponds a configuration of the system. Each configuration corresponds to one or several locations. The running of a constant speed continuous PN can be modelled by a linear hybrid automaton where the state variables are the continuous places marking.

Only one type of events can change the continuous variables derivative, this autonomous event is the emptiness of a continuous place marking. This is always an uncontrollable event. For the HPN in Figure A1.a, the discrete part has 2 reachable markings, then 2 configurations for the continuous part. Figure A2 represents the hierarchical form of the hybrid automaton.

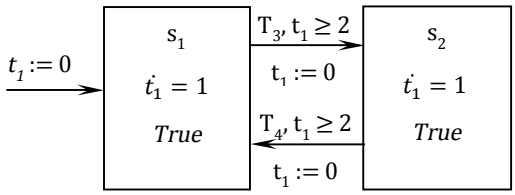
The final form of the hybrid automaton is obtained by replacing each transition between macro-locations by transitions between internal locations. A reachability analysis can be useful to eliminate the unreachable locations and non fireable transitions. The hybrid automaton resulting from the translation of the HPN in figure A1.a is represented in Figure A4.



-a-



-b-



-c-

Fig. A1.a. HPN of the producer-consumer system; b. T-Time Petri net of the discrete part; c. Equivalent hybrid automaton.

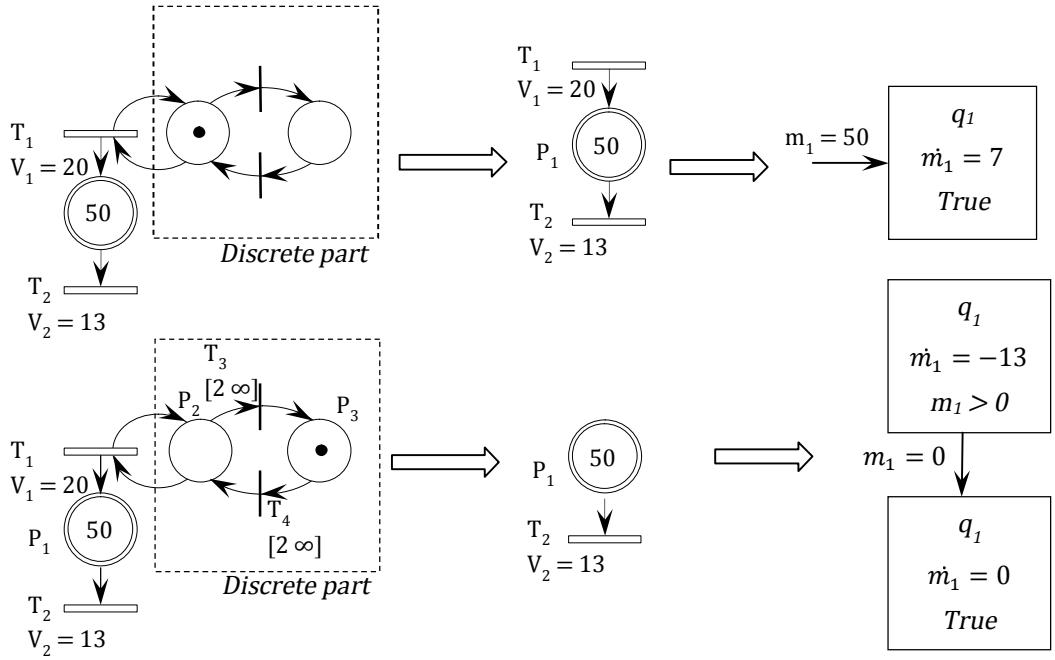


Fig. A2. Continuous behaviour corresponding to each discrete marking.

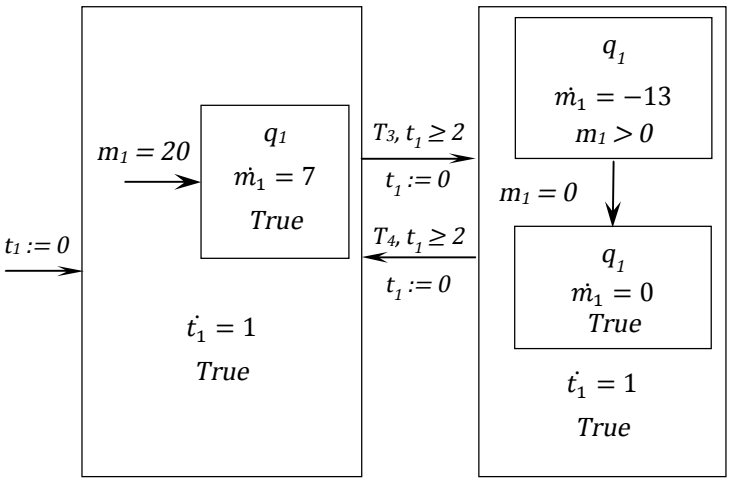


Fig. A3. Hierarchical form of the hybrid automaton.

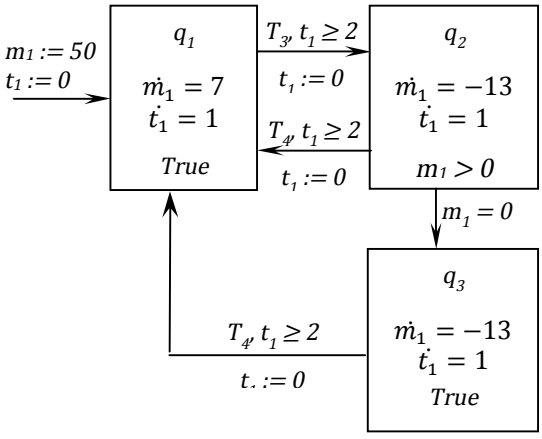


Fig. A4. Hybrid automaton resulting from the translation approach.