International Journal of Control and Automation
Vol.9, No.3 (2016), pp.89-100
http://dx.doi.org/10.14257/ijca.2016.9.3.09

The Synthesis Design and Application of Hybrid Systems

Hai-bin Zhang

School of Computer Science and Technology, Xidian University, Xi'an City, P. R. China
h_b_zhang@126.com

Abstract

In this paper, a computational model for hybrid systems is defined. Based on this model, a partial order relation of hybrid systems is given and a lattice of hybrid systems is formalized. Armed with these notations, synthesis of hybrid systems is discussed. The synthesis can be seen as the operation of solving the least upper bound of members of the lattice. An example is given to illustrate the approach.

Keywords*: hybrid systems, synthesis, temporal logic*

1. Introduction

A hybrid system is a dynamical system with both discrete and continuous state changes [1-8].The continuous activities are usually governed by differential equations or algebraic equations, and the discrete events are expressed by discrete variables. Most safety critical systems such as aircraft landing system and the system controlling fast trains, as well as some manufacturing systems such as the nuclear plant control system, are typical hybrid systems .To model, simulate, verify and control such a system, various kinds of formalisms based on finite state machines, e.g. phase transition systems [9], hybrid automata [10], hybrid graphs [11], state charts [12], and integration graphs [13-14], as well as formalisms based on logics, e.g. Duration calculus [15], TLA+[16], and ERTL [17] have been used with success in recent years. It seems that the machine-based notations admit a computational model for hybrid systems based on a two-phase step assumption. That is, a computation run of a hybrid system is assumed to be a sequence of two-phase steps. The first phase of a step corresponds to a continuous state evolution usually described by differential equations; in the second phase, the state is submitted to a discrete change taking zero time.

This paper focuses on the synthesis (or composition) of hybrid systems. In some cases of physical environments, a complex hybrid system can be viewed as a composition or synthesis of a number of small sub-hybrid systems. For instance, a steel plant system consists of several soaking pit furnaces, the ingots cooling bank, and transport systems such as cranes and chariots [4]. To model and verify such systems, it is important for us to investigate theory and technique for dealing with synthesis of hybrid systems. In this paper, an approach based on lattice for synthesis of hybrid systems is presented. With this method, a continuous activity is modeled as a macro-state, and a discrete event is modeled as a micro-state which occurs between two macro-states. Thus, a hybrid system is modeled as a sequence of continues activity and discrete events, that is, an interleaving sequence of macro-states and microstates. This sequence is also called an interval. To manipulate synthesis of hybrid systems, a partial order relation $\leq$ over intervals is defined. All hybrid systems under this partial order relation become a lattice since two arbitrary hybrid systems have the least upper bound and the great lower bound. Thus, the synthesis of two hybrid systems can be viewed as the operation of solving the least upper bound of the two systems. This enables us to use theory of lattices for hybrid systems.

ISSN: 2005-4297 IJCA

This paper is organized as follows. The next section presents a computational model. In Section 3, the replacement of variables in hybrid systems is described. Section 4 discusses the synthesis of hybrid systems. A synthesis algorithm is given in details. Section 5 provides with an example of synthesis of two hybrid systems. Conclusions are drawn in Section 6.

2. Computational Model

In this section, a computational model for hybrid systems is formalized. Based on this model, micro-states, macro-states and intervals are defined. These notations are those presented in [3] with some minor changes.

We model time by the non-negative real line $R^{\oplus}$. A time interval is a left and right closed subinterval of $R^{\oplus}$ if it is finite; or it is a left closed and right unbounded sub-interval if the interval is infinite. The left end-point of a time interval I is denoted by l_I and the right end-point is r_I. Two intervals I_1 and I_2 are adjacent if $r_{I_1} = l_{I_2}$. A time interval sequence $\bar{I} = I_0\ I_1 \ldots$ is a finite or infinite sequence of time intervals that partitions $R^{\oplus}$.

- Any two intervals I_i and I_{i+1} are adjacent.
- For the first time interval I_0, $l_{I_0} = 0$, and for the final time interval of any finite interval sequence, it is unbounded.

Let $\mathit{\Pi}$ be the set of propositions, and V, the set of dynamic and static variables. A micro-state s is an interpretation of all the variables in V and all the propositions in $\mathit{\Pi}$. We write Σ_s for the set of micro-states.

In the following, at any time instant t, for each variable $x \in V$, we use $x^-(t)$ to denote the left limit of x in t, $x^+(t)$ the right limit. s^- represents the interpretation of all left limits of variables while s^+ represents the interpretation of all right limits of variables.

A macro-state m_i is a tuple, (s_i, I_i, g_i), which models a continuous activity, where I_i is a time interval, and g_i is a family of continuous functions $g_{ix} : I_i^0 \rightarrow$ R, where $I_i^0 = (l_{I_i}, r_{I_i})$. s_i denotes the right micro-state s^+ at time point l_{I_i}. s_i is called the left end micro-state and $g_i^-(r_{I_i})$ is called the right end micro-state of the macro-state m_i.

Definition 1 (Intervals). An interval $\sigma =< m_0, m_1, \ldots >$ is a non-empty sequence of macro-states, such that $\forall\ m_i, m_{i+1}$ in σ, I_i, I_{i+1} are adjacent. The length of σ, denoted by $|\sigma|$, is defined as ω if σ is infinite; otherwise it is the number of macro-states in σ minus 1.For $0 \le i, j \le |\sigma|$, we will use $\sigma(i..j)$ to denote the subinterval starting at the macro-state m_i and ending at the macro-state m_j, and $\sigma^{(k)}$ to denote the interval $< m_k, \ldots >$.

3. Replacements of Variables in Hybrid Systems

To model hybrid systems, we use variables to denote activities. The variable name is not sensitive since an activity can be specified by an arbitrary variable. Therefore two

hybrid systems are equivalent if one can be derived from another by replacing some variables.

Definition 2. Given a hybrid system σ, if x is a variable used in σ and t is a variable not appearing in it. Then σ[t/x] denotes the result of simultaneous replacement of all occurrences of x by t. The replacement is called compatible if either x and t are static or x is dynamic.

In the following, we write σ(x) to imply that σ has an occurrence of variable x. The replacement σ[t/x] is called admissible for σ(x) if it is compatible and t has no occurrence in σ. We claim that, in the following sections, two arbitrary hybrid systems don't have the same variables except for shared variables (this can be achieved by replacements of variables).

4. Synthesis of Hybrid Systems

Let $\sum_\sigma = \{\sigma_i \mid i \in N\}$ be the set of hybrid systems, and $\leq$ over $\sum_\sigma$ is defined as follows:

Definition 3. ($\leq$) For $\forall \sigma_i$, $\sigma_j \in \sum_\sigma$, here $\sigma_i =< m_0^i, m_1^i, \cdots >$ and $\sigma_j =< m_0^j, m_1^j, \cdots >$, $\sigma_i \leq \sigma_j$ if and only if

(1) $\exists h, l, 0 \leq h \leq l \leq |\sigma_j|$ such that $l_{I_0^i} = l_{I_n^j}$, $r_{I^i_{|\sigma_i|}} = r_{I_l^j}$ [1] and

(2) $\forall k, h \leq k \leq l$, $\exists k', 0 \leq k' \leq |\sigma_i|$ such that $I_k^j \subseteq I_{k'}^i$, $g_i^+(l_{I_k^j}) \subseteq s_k^j$ and $g_{k'}^i \subseteq g_k^j$.

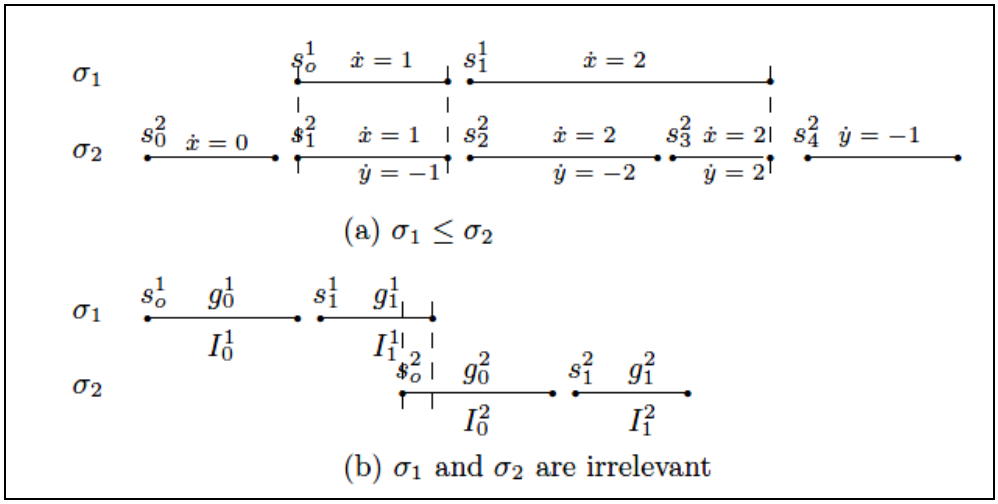


Figure 1. Relations of Intervals

Example 1. In Figure 1(a) $s_0^1 = \{x = 0\}$, $s_1^1 = \{x = 3\}$; $s_1^2 = \{x = 0, y = 0\}$, $s_2^2 = \{x = 3, y = 2\}$, $s_3^2 = \{x = 7, y = 6\}$; $I_1^1 = [3,6]$; $I_2^2 = [3,5]$, $I_3^2 = [5,6]$.

Theorem 1. Let $\sum_\sigma = \{\sigma_i \mid i \in N\}$ be the set of hybrid systems, and $\leq$ be the binary relation defined in Definition 3. Then $(\sum_\sigma, \leq)$ is a partial order.

Proof.

(1) Reflexivity: it is obvious.

ISSN: 2005-4297 IJCA

(2) Antisymmetry: for $\forall\sigma_i,\sigma_j$, if $\sigma_i \le \sigma_j$ and $\sigma_j \le \sigma_i$, by the definition of $\le$, $[l_{I_O^i}, r_{I^i_{|\sigma_i|}}] \subseteq [l_{I_o^j}, r_{I^j_{|\sigma_j|}}]$ and $[l_{I_O^j}, r_{I^j_{|\sigma_j|}}] \subseteq [l_{I_o^i}, r_{I^i_{|\sigma_i|}}]$, so $l_{I_o^i} = l_{I_O^j}$, and $r_{I^i_{|\sigma_i|}} = r_{I^j_{|\sigma_j|}}$. For $\forall m_k^j$ in σ_j, $\exists m_r^i$ in σ_i, such that $I_k^j \subseteq I_r^i$, $g_i^+(l_{I_r^j}) \subseteq s_k^j$, $g_r^i \subseteq g_k^j$, for this m_r^i, $\exists m_s^j$ such that $I_r^i \subseteq I_s^j$, $g_j^+(l_{I_r^i}) \subseteq s_r^i$, $g_s^j \subseteq g_r^i$, then $I_k^j \subseteq I_s^j$. By the definition of interval, m_k^j and m_s^j are the same macro-states of σ_j, so $I_k^j \subseteq I_r^i$, $I_r^i \subseteq I_k^j$ and $s_k^j \subseteq s_r^i$, $s_r^i \subseteq s_k^j$, $g_k^j \subseteq g_r^i$, $g_r^i \subseteq g_k^j$. Hence, $m_k^j = m_r^i$. Conversely, in the same way we can prove that for $\forall m_t^i$ in σ_i, $\exists m_{t'}^j$ in σ_j, such that $m_t^i = m_{t'}^j$. Therefore $\sigma_i = \sigma_j$.

(3) Transitivity: for $\forall\sigma_i,\sigma_j,\sigma_k$, if $\sigma_i \le \sigma_j$ and $\sigma_j \le \sigma_k$, by the definition of $\le$, $\exists h,l,h',l'$, $0 \le h \le l \le |\sigma_j|$, $0 \le h' \le l' \le |\sigma_k|$, Such that $l_{I_0^i} = l_{I_h^j}$, $r_{I^i_{|\sigma_i|}} = r_{I_l^j} l_{I_0^j} = l_{I_{h'}^k}$, $r_{I^j_{|\sigma_j|}} = r_{I_{l'}^k}$, for $\forall m, h' \le m \le l'$, $\exists m', 0 \le m' \le |\sigma_j|$ and $I_m^k \subseteq I_{m'}^j$. Thus, there exist a and b, $h' \le a \le b \le l'$, such that $l_{I_h^j} = l_{I_a^k}$, $r_{I_l^j} = r_{I_b^k}$, leading to $l_{I_0^i} = l_{I_a^k}$ and $r_{I^i_{|\sigma_i|}} = r_{I_b^k}$. In addition, for $\forall r, a \le r \le b$, $\exists s, h \le s \le l$, such that $I_r^k \subseteq I_s^j$, $g_j^+(l_{I_r^k}) \subseteq s_r^k$, $g_s^j \subseteq g_r^k$, for this s, $\exists t, 0 \le t \le |\sigma_i|$, $I_j^s \subseteq I_t^i$, $g_i^+(l_{I_s^j}) \subseteq s_s^j$, $g_t^i \subseteq g_s^j$, hence, $I_r^k \subseteq I_t^i$, $g_i^+(l_{I_r^k}) \subseteq s_r^k$, $g_t^i \subseteq g_r^k$. Therefore $\sigma_i \le \sigma_k$.

We extend $\sum_\sigma$ to $\sum$ by adding ε (empty interval), i.e. $\sum = \sum_\sigma \cup \{\varepsilon\}$, it is obvious that $(\sum, \le)$ is still a partial order.

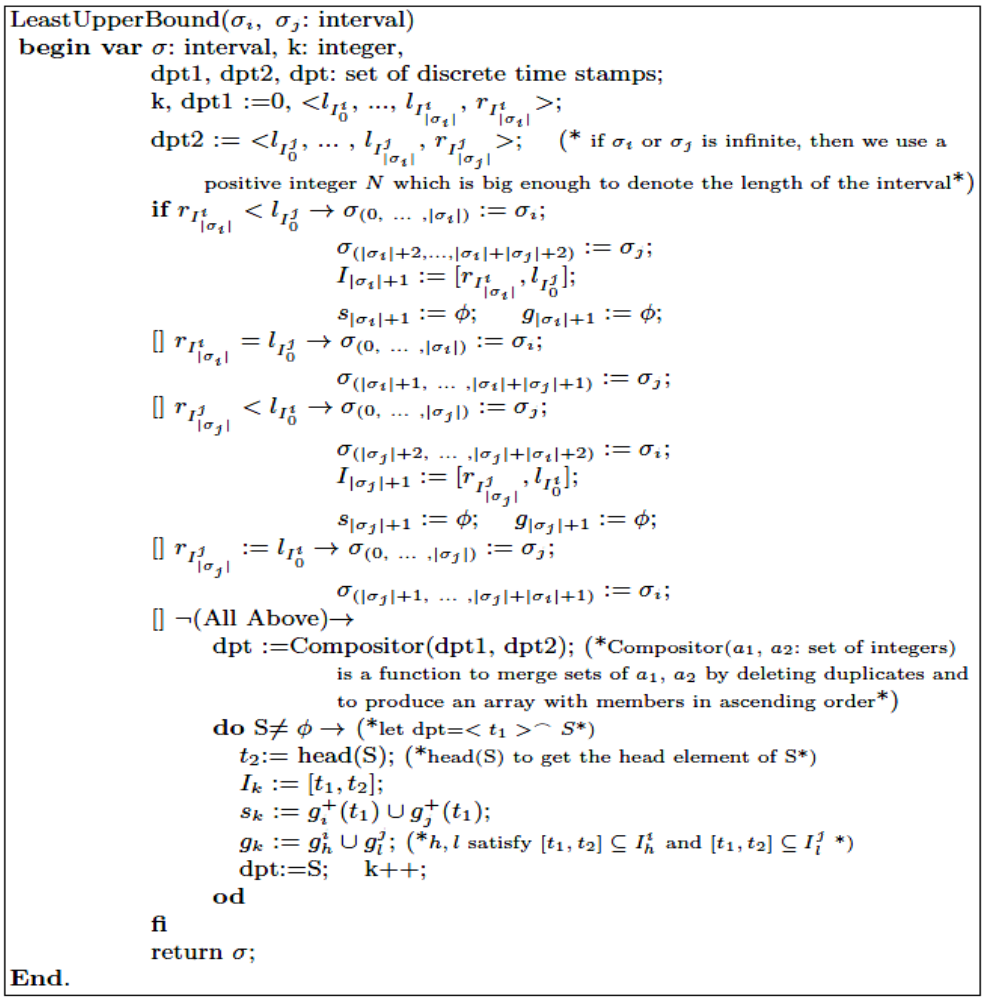


Figure 2. Algorithm 1 for the Least Upper Bound

Theorem 2. For the partial order $(\sum, \leq)$, $\forall \sigma_i, \sigma_j \in \sum$, there exist the least upper bound and the great lower bound.

Before the proof, we present two algorithms. One is for solving the least upper bound of two hybrid systems, and the other is for solving the great lower bound. The algorithms use Dijkstra's guarded command language (Dijkstra,1976)[18].

Example 2. In figure 3, s_0^1 = {x = 5, y = 0}, s_1^1 = {x = 0, y = 5}; s_0^2 ={z = 0}, s_1^2 = {x = 2, z =0 }; s_0^L = {x = 5, y = 0}, g_0^L = { $\dot{x} = -1$, $\dot{y}$ = 1}, s_1^L = { x = 3, y = 2, z = 0}, g_1^L = { $\dot{x}$ = -1, $\dot{y}$ = 1, $\dot{z}$ = 1}, s_2^L = { x = 0, y = 5, z = 2}, g_2^L = { $\dot{x}$ = 1, $\dot{y} = -1$, $\dot{z}$ = 1}, s_3^L = {x = 2, y = 3, z = 0}, g_3^L = { $\dot{x}$ = 1, $\dot{y}$ = -1, $\dot{z}$ = -1}, s_4^L = {x = 4,z = -2}, g_4^L ={ $\dot{x}$ = 1, $\dot{z}$ = -1}.

ISSN: 2005-4297 IJCA

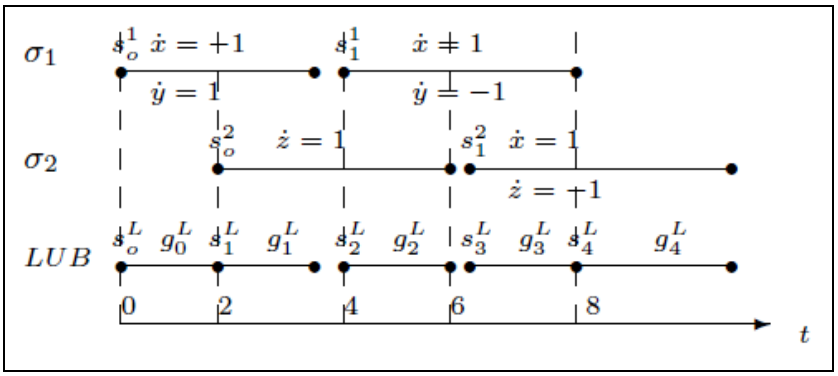


Figure 3. LUB Of σ_1, σ_2 Generated by Algorithm 1

```
GreatLowerBound(σ_i, σ_j: interval)
 begin var σ: interval, k: integer,
          dpt1, dpt2, dpt: set of discrete time instants;
          k, dpt1 :=0, <l_{I_0^i}, ..., l_{I^i_{|σ_i|}}, r_{I^i_{|σ_i|}}>;
          dpt2 := <l_{I_0^j}, ... , l_{I^j_{|σ_i|}}, r_{I^j_{|σ_j|}}>;
          if (r_{I^i_{|σ_i|}} ≤ l_{I_0^j} ∨ r_{I^i_{|σ_j|}} ≤ l_{I_0^i}) → σ := ε;
            [] ¬ ((r_{I^i_{|σ_i|}} ≤ l_{I_0^j} ∨ r_{I^i_{|σ_j|}} ≤ l_{I_0^i}))→
               dpt :=IdenticalMember(dpt1,dpt2); (*IdenticalMember(a_1, a_2: set
                of integers) is a function to search for identical members in sets of a_1, a_2*)
               do S≠ φ →(*let dpt=< t_1 >⌢ S*)
                  t_2:=head(S); (*head(S) to get the head element of S*)
                  I_k := [t_1, t_2];
                  if g_i' ≠ φ ∨ g_j' ≠ φ →    (*g_i' and g_j' are respectively the maximal
                                subsets of g_i and g_j, the restrictions of which into the time
                                duration [t_1, t_2] are continuous*)
                     s_k := g_i^+(t_1) ∩ g_j^+(t_1);
                     g_k := g_i^{k'} ∩ g_j^{k'};   (*g_i^{k'} and g_j^{k'} are respectively restrictions of g_i'
                                                    and g_j' into the time duration I_k*)
                  fi
                  dpt:=S;    k++;
               od
          fi
          return σ;
End.
```

Figure 4. Algorithm 2 for the Great Lower Bound

Example 3. In Fig. 5, s_1^1 = {x = 0, y = 0}, s_2^1 = {x = 2, y = -2}; g_2^1 ={ $\dot{x} = -1$, $\dot{y}$ = 1}, s_3^1 = {x = 0, y =2 }; g_3^1 = { $\dot{x}$ = -1, $\dot{y}$ = -1}, s_4^1 = { x = -2, y = 0}, g_4^1 = { $\dot{x}$ = -1, $\dot{y}$ = 0}, s_1^2 = { x = 0, z = 4}, s_2^2 = {x = 2, z = 0}, s_3^2 = {x = 2,z = -4}, s_0^G = {x = 0}, s_1^G = {x = 2}.

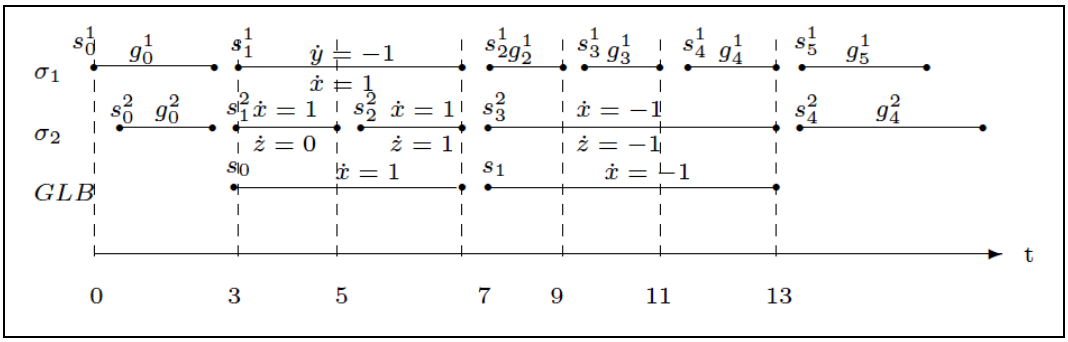


Figure 5. GLB Of σ_1, σ_2 Generated By Algorithm 2

proof (Theorem 2). (1): σ derived from Algorithm 1 being the least upper bound of σ_i, σ_j is proved as follows.

Firstly, by the definition of $\leq$ and Algorithm 1, it is easy to prove than σ is an upper bound of σ_i, σ_j.

Secondly we prove that $\forall \sigma' = < m_0', ... >$, if σ' is an upper bound of $\sigma_i, \sigma_j, \sigma \leq \sigma'$. Since σ' is an upper bound of $\sigma_i, \sigma_j, \exists h, l, h', l', 0 \leq h, l, h', l' \leq |\sigma'|$, such that $l_{I_0^i} = l_{I_h'}$, $r_{I^i_{|\sigma_i|}} = r_{I_l'}$ and $l_{I_0^j} = l_{I_{h'}'}$,

$r_{I^j_{|\sigma_j|}} = r_{I_{l'}'}$. Since an upper bound of $\sigma_i, \sigma_j, \exists h, l, h', l', 0 \leq h, l, h', l' \leq |\sigma'|$, such that $l_{I_0^i} = l_{I_h'}$, $r_{I^i_{|\sigma_i|}} = r_{I_i'}$ and $l_{I_0^j} = l_{I_{h'}'}$, $r_{I^j_{|\sigma_j|}} = r_{I_i'}$. Since $l_{I_0} = \min(l^i_{I_0}, l^j_{I_0})$, $r_{I_{|\sigma|}} = \max(r_{I^i_{|\sigma_i|}}, r_{I^j_{|\sigma_j|}})$, there exist a, b, $a = \min(h, h')$, $b = \max(l, l')$, such that $0 \leq a \leq b \leq |\sigma'|$, $l_{I_0} = l_{I_a'}$, $r_{I|\sigma|} = r_{I_b'}$. For $\forall k, a \leq k \leq b$, by the definition of $\leq$ and Algorithm 1, m_k' satisfies one of the following conditions:

Case 1: $I_k' \subseteq [l_{I_0^i}, r_{I^i_{|\sigma_i|}}]$ and $I_k' \not\subseteq [l_{I_0^j}, r_{I^j_{|\sigma_j|}}]$. Then $\exists r, s, 0 \leq r \leq |\sigma_i|$ and $0 \leq s \leq |\sigma|$, such that $I_k \subseteq I_r^i$, $g_i^+(l_{I_k'}) \subseteq s_k'$, $g_r^i \subseteq g_k'$ and $m_r^i = m_s$, hence, $I_k' \subseteq I_s$, $g^+(l_{I_k}) \subseteq s_k'$, $g_s \subseteq g_k'$.

Case 2: $I_k' \subseteq [l_{I_0^j}, r_{I^j_{|\sigma_j|}}]$ and $I_k' \not\subseteq [l_{I_0^i}, r_{I^i_{|\sigma_i|}}]$. In the same way we can prove that $\exists s, 0 \leq s \leq |\sigma|$ such that $I_k' \subseteq I_s$, $g^+(l_{I_k'}) \subseteq s_k'$, $g_s \subseteq g_k'$.

Case 3: $I_k' \subseteq [l_{I_0^i}, r_{I^i_{|\sigma_i|}}] \cap [l_{I_0^j}, r_{I^j_{|\sigma_j|}}]$. Then $\exists s$, t $0 \leq s \leq |\sigma_i|$ and $0 \leq t \leq |\sigma_j|$, such that $I_k' \subseteq I_s^i$, $g_i^+(l_{I_k'}) \subseteq s_k'$, $g_s^i \subseteq g_k'$ and $I_k' \subseteq I_t^j$, $g_j^+(l_{I_k'}) \subseteq s_k'$, $g_t^j \subseteq g_k'$, thus, $I_k' \subseteq [\max(l_{I_s^i}, l_{I_t^j}), \min(r_{I_s^i}, r_{I_t^j})]$, and $(g_i^+(l_{I_k'}) \cup g_j^+(l_{I_k'})) \subseteq s_k'$, $(g_s^i \cup g_t^j) \subseteq g_k'$. Therefore, $\exists r, 0 \leq r \leq |\sigma|$, $I_r = [\max(l_{I_s^i}, l_{I_t^j}), \min(r_{I_s^i}, r_{I_t^j})]$, such that $I_k' \subseteq I_r$, $g^+(l_{I_k'}) = (g_i^+(l_{I_k'}) \cup g_j^+(l_{I_k'})) \subseteq s_k'$, and $g_s = (g_s^i \cup g_t^j) \subseteq g_k'$.

Case 4: $I_k' \not\subseteq [l_{I_0^i}, r_{I^i_{|\sigma_i|}}]$ and $I_k \not\subseteq [l_{I_0^j}, r_{I^j_{|\sigma_j|}}]$. Since σ' is an upper bound of σ_i, σ_j, $\exists r$, $I_r = [r_{I^i_{|\sigma_i|}}, l_{I_0^j}]$ or $I_r = [r_{I^j_{|\sigma_j|}}, l_{I_0^i}]$ such that $I_k' \subseteq I_r$, $g^+(l_{I_k'}) = \varphi \subseteq s_k'$ and $g_r = \varphi \subseteq g_k'$.

(2) We can similarly prove that σ derived by Algorithm 2 is the great lower bound of σ_i, σ_j.

We use the binary operator $\vee$ and $\wedge$ on $\sum$ respectively as the operation for solving the least upper bound and the great lower bound of two hybrid systems.

Theorem 3. The algebra $(\sum, \vee, \wedge)$ is a lattice.

proof. By Theorem 2, it is straightforward.

Thus, the operator $\vee$ can act as the synthesis of hybrid systems.

ISSN: 2005-4297 IJCA

5. An Example of Synthesis of Hybrid Systems

In this section, an example, a water-level monitor [21] is given to illustrate how two different hybrid systems can be synthesized using our approach.

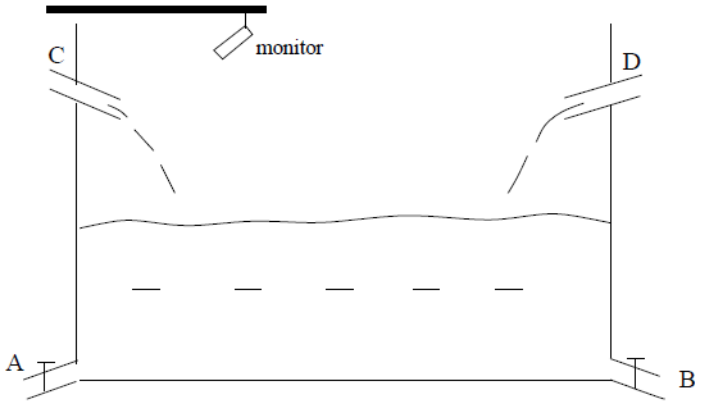


Figure 6. The Water-Level Monitor

The water level in a tank is controlled by two valves (A, B) used for draining and two pumps (C,D) used for inpouring as well as a monitor employed for sending signals to switch on or off valves and/or pumps (Figure 6).

When we use only valve A and pump C to control the water level, the system is modeled by a hybrid machine [2, 3, 5, 19, 20] (see the appendix for details) shown in figure 7 (a). When valve A is open and pump C is off, the water level, denoted by variable y1 falls by 3 inches per second; whereas when valve A is closed and pump C is on, the water level rises by 3 inches per second. Suppose that, initially, the water level (y1) is 40 inches and valve A is open and pump C is off. Whenever the water level falls to 22 inches or raises to 40 inches the monitor sends signals to change the status of both the valve and pump.

When we use only valve B and pump D, the water level changes as shown in figure 7(b). When valve B is open and pump D is off, the water level, denoted by variable y2 falls by 3 inches per second; whereas when valve B is open and pump D is on, the water level rises by 1 inch per second. Suppose that, initially, the water level (y2) is 40 inches, and valve B is open and pump D is off. Whenever the water level falls to 22 inches or rises to 40 inches, the monitor sends signals to change the status of the pump only.

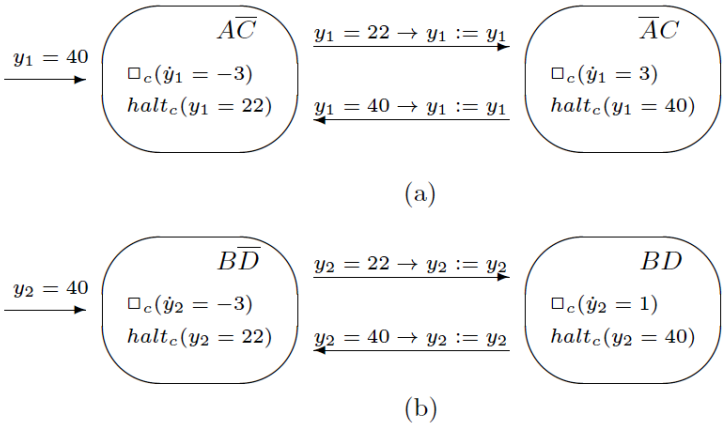
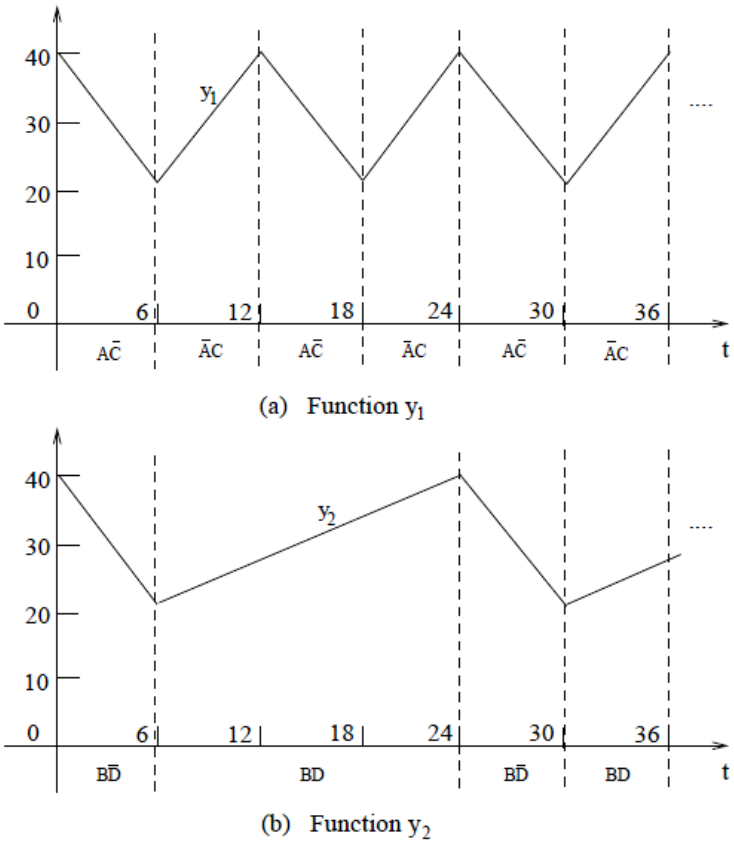


Figure 7. Two Water Level Control Systems



(a) Function y_1

(b) Function y_2

Figure 8. Function Y1 and Y2

In Figure 7, $A\bar{C}$ denotes a type of macro-states— valve A is open and pump C is off. Similarly, $\bar{A}C$, denotes that A is closed and C is on; $B\bar{D}$ means B is open and D is off; and BD tells us that B is open and D is on.

We now model the hybrid systems shown in Figure 7 within HPTL [3] formulas.

$$p_1 \stackrel{def}{=\!=} \Box_c(\dot{y}_1 = -3) \wedge halt_c(y_1 = 22)$$

$$p_2 \stackrel{def}{=\!=} \Box_c(\dot{y}_1 = 3) \wedge halt_c(y_1 = 40)$$

$$q \stackrel{def}{=\!=} y_1 = 40 \wedge s_1 = 0 \wedge \Box((s_1 = 0 \rightarrow O(s_1 = 1 \wedge y_1 = 22)) \wedge (s_1 = 1 \rightarrow O(s_1 = 0 \wedge y_1 = 40)))$$

Then the formula,

$$(p_1, p_2)^{(+)}\ prj\ q \vee (p_1, (p_2, p_1)^{(+)})\ prj\ q$$

Model of the hybrid system is shown in Figure 8 (a). And

$$p_1 \stackrel{def}{=\!=} \Box_c(\dot{y}_2 = -3) \wedge halt_c(y_2 = 22)$$

$$p_2 \stackrel{def}{=\!=} \Box_c(\dot{y}_2 = 1) \wedge halt_c(y_2 = 40)$$

$$q \stackrel{def}{=\!=} y_2 = 40 \wedge s_2 = 0 \wedge \Box((s_2 = 0 \rightarrow O(s_2 = 1 \wedge y_2 = 22)) \wedge (s_2 = 1 \rightarrow O(s_2 = 0 \wedge y_2 = 40)))$$

ISSN: 2005-4297 IJCA

Then the formula,

$$(p_1, p_2)^{(+)}\ prj\ q \vee (p_1, (p_2, p_1)^{(+)})\ prj\ q$$

Models the hybrid system is shown in Figure 8 (b). s1 and s2 are two discrete variables. The system shown in Figure 8 (a) locates at state $A\bar{C}$ when s1=0 and at $\bar{A}C$ whens1=1 ; whereas the system shown in Figure 8 (b) locates at state $B\bar{D}$ when s2=0 and at BD when s2=1.

The two systems in Figure 7 can be modeled using intervals as shown in Figure 8.

Using Algorithm 1, we can derive the synthesis of the two hybrid systems in Figure 8. To show the intuitive result of the synthesized system, a variable y is added to show that, in every macro-state of the synthesized system, the derivative of y, $\dot{y}$ is equal to $\dot{y}_1 + \dot{y}_2$ (Figure 9).

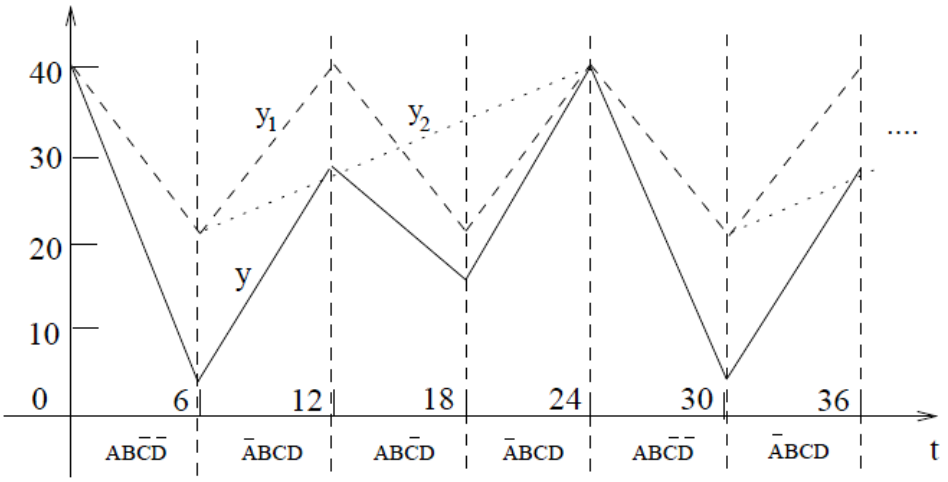


Figure 9. The Synthesized Water-Level Monitor

In Figure 9, the solid line denotes the synthesized variable y, the dashed line denotes variable y1 while the dotted line denotes variable y2. We can conclude that, there are three types of macro-states in the synthesized water-level monitor: $AB\bar{C}\bar{D}$, $\bar{A}BCD$, $AB\bar{C}D$ and $(\bar{A}BCD)'$ Taking $AB\bar{C}D$ for example, it is synthesized by $A\bar{C}$ in Figure 8 (a) and BD in Figure 8 (b), expressing that valve A and B are open for draining, and pump D is on for inpouring but C is off. In this type of macro-states, the water level falls by 2 inches per second.

The synthesized water-level monitor can be modeled equivalently by a hybrid machine shown in Figure10 or a HPTL formula as follows:

$$(p_1, p_2, p_3, p_4)^{(+)}\ prj\ q \vee (p_1, (p_2, p_3, p_4, p_1)^{(+)})\ prj\ q$$

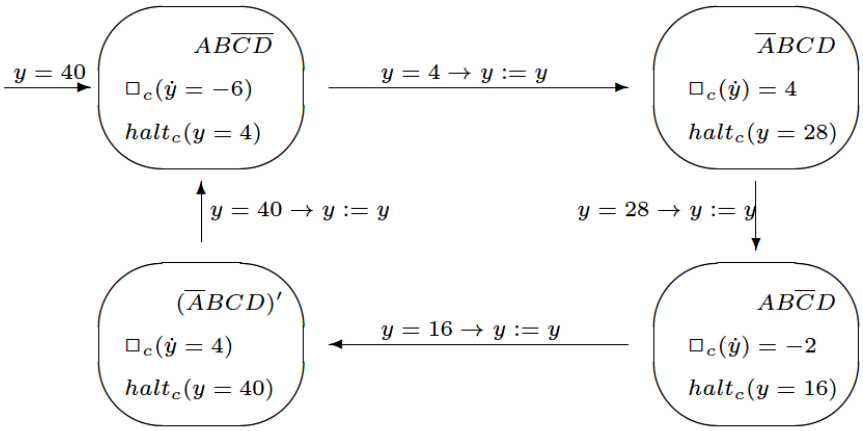


Figure 10. The Synthesized Water-Level Monitor

Where

$$p_1 \overset{def}{=\!=} \Box_c(\dot{y} = -6) \wedge halt_c(y = 4)$$

$$p_2 \overset{def}{=\!=} \Box_c(\dot{y} = 4) \wedge halt_c(y = 28)$$

$$p_3 \overset{def}{=\!=} \Box_c(\dot{y} = -2) \wedge halt_c(y = 16)$$

$$p_4 \overset{def}{=\!=} \Box_c(\dot{y} = 4) \wedge halt_c(y = 40)$$

$$q \overset{def}{=\!=} y = 40 \wedge s = 0 \wedge \Box((s = 0 \to O(s = 1 \wedge y = 4)) \wedge (s = 1 \to O(s = 2 \wedge y = 28)) \wedge$$

$$(s = 2 \to O(s = 3 \wedge y = 16)) \wedge (s = 3 \to O(s = 0 \wedge y = 40)))$$

And s is a discrete variable. The water-level monitor system locates at state $A\overline{B}\overline{C}D$ when s =0, and at $\overline{A}BCD$ when s =1, and at $AB\overline{C}D$ when s =2, and at $(\overline{A}BCD)'$ when s =3.

6. Conclusion

In this paper, we have outlined a partial order relation $\leq$ on hybrid systems and proved that under this relation the set of hybrid systems is a lattice. The algorithm for the least upper bound and the great lower bound are also given in details. The synthesis of hybrid systems can be seen as an operation $\vee$ in the lattice. An example is given to show the synthesis of hybrid systems using our approach in this paper. However, the relationship between synthesized system and the subsystems are needed to be investigated. Moreover, how can verification of synthesized system be done by means of subsystems is another issue in our future research.

ISSN: 2005-4297 IJCA

Acknowledgements

This research is supported by the NSFC under Grant No. 61373043 and 61003079.

References

[1] Z. Duan, M. Holcombe and A. Bell, "Bio. Systems", vol. 55, **(2000)**, pp. 93-105.
[2] Z. Duan, "Modeling of hybrid systems", Department of Computer Science, Ph.D thesis, University of Sheffield, UK, February, **(1997)**.
[3] Z. Duan, "Modelling and Analysis of Hybrid Systems", Science Press, Beijing, **(2004)**.
[4] Z. Duan, "Holcombe M., and Linkens A", Modelling of a soaking pit furnace in hybrid machines, In Proceeding of the IMACS symposium on systems analysis and simulation, Berlin, **(1995)**.
[5] E.D. Francis, N. Eswara Prasad, C.h. Ratnam, P.S. Kumar and V.V. Kumar, International Journal of Advance Science and Technology, vol. 27, **(2011)**, pp. 35-44.
[6] R. Alur, T.A. Henzinger, G. Lafierriere and G.J. Pappas, "Discrete Abstractions of Hybrid Systems", Proceedings of the IEEE vol. 88, no. 7, **(2000).**
[7] T.A. Henzinger and P.W. Kopke, "Theory on Computer. Science", vol. 221, **(1999)**, pp. 369-392.
[8] K.L. Lu, W. Yan, Q.Y. Ding and C. Wang, „International Journal of Hybrid Information Technology, vol. 6, no. 3, **(2013)**, pp. 33-44.
[9] S. Umamaheswari, Dr. V. Palanisamy and Dr. M. Chidambaram, International Journal of Cont. and Auto, vol. 3, no. 2, **(2010)**, pp. 1-8.
[10] R. Alur, C. Courcoubetis and T.A. Henzinger, "Ho P-H., LNCS", vol. 736, **(1993)**, pp. 209-229.
[11] X. Nicollin, A. Olivero , J. Sifakis and S. Yovine, "LNCS", vol. 736, **(1993)**, pp. 149-178.
[12] R. Aloui and N.B. Braiek, International Journal of Cont. and Automobile, vol. 5, no. 3, **(2012)**, pp. 65-78.
[13] D. Harel, Science of Computer Programming, vol. 8, **(1987)**, pp. 231-274.
[14] Y. Keten, A. Pnueli, J. Sifakis and S. Yovine, "LNCS", vol. 736, **(1993)**.
[15] C.C. Zhou, A.P. Ravn and M.R. Hansen, "LNCS", vol. 736, **(1993)**.
[16] S.K. Shome, S.R.K. Vadali, U. Datta, S. Sen and A. Mukherjee, International Journal of Signal Proceeding, Image Proceeding and Patt. Reco, vol. 5, no. 3, **(2012)**, pp. 75-92.
[17] R.D. Lemons and J.G. Hall, Hybrid System, vol. 3, **(1995)**.
[18] J.P. Katoen, "Lecture Note of the Court", **(1998)**, pp. 79-81.
[19] Eilenberg, "Automata Languages and Machiness", Academic press, vol. A, **(1974)**.
[20] N. Patel and M. Zaveri, International Journal of Computer Grap, vol. 1, no. 1, **(2010)**, pp. 1-18.
[21] R. Alury, C. Courcoubetisz, N. Halbwachsx, T.A. Henzinger, P.-H. Hox, X. Nicollinz, A. Oliv-eroz, J. Sifakisz and S. Yovinez, "Theory of Computer Science", vol. 138, no. 1, **(1995)**, pp. 3-34.

