Scheduling of Hybrid System: A New Approach Integrating Decomposition of THPN and Multi-core Cluster

1, 2 Wei-Zhi Liao
1 College of Mathematics Physics and Information Engineering, Jiaxing University, Jiaxing 314001, China
2 Guangxi Key laboratory of hybrid computation and IC design analysis, Nanning 530006, China
1 Tel.: (86)057383647003
E-mail: weizhiliao2002@aliyun.com

Received: 1 January 2014 /Accepted: 7 March 2014 /Published: 30 April 2014

Abstract: Effective scheduling of hybrid system in industry has a high economic return. However, a hybrid system contains continuous dynamics and discrete dynamics so that the modeling and scheduling of hybrid system become more and more complex and difficult. This paper presented a new approach integrating decomposition of timed hybrid Petri net (THPN) and multi-core cluster to scheduled hybrid system. Firstly, a decomposition method for THPN which is used to model hybrid system is proposed. A timed hybrid Petri net is decomposed to several subnets and each subnet is a T-net, in which any place has no more than one input or one output transition. Secondly, each subnet of THPN is mapped to a processor of multi-core cluster and scheduling model of master-slave is used to realize parallel scheduling of hybrid system. Finally, the effectiveness of the proposed method is demonstrated by experiment results. Copyright © 2014 IFSA Publishing, S. L.

Keywords: Hybrid system, Hybrid Petri nets, Decomposition, Parallel scheduling, Multi-core cluster.

1. Introduction

Modeling and scheduling of hybrid system provide an important guidance to the process automation, automatic scheduling, robot control, computer communications and a series of engineering problems. The hybrid system contains continuous dynamics and discrete dynamics so that the modeling and scheduling of hybrid system become more and more complex and difficult. Even today, the optimization and scheduling of the hybrid system is still lack the effective and feasible methods. Even for a small-sized system, it’s very difficult to find an optimal solution or even the local optimal solution. Hybrid Petri net is one of the most effective tools for the simulation and analysis of the hybrid system, which can evaluate the dynamic behavior of the hybrid system combined with graphics and analysis. It was first developed by R. David and H. Alla. With the definition of the hybrid Petri nets, discrete event systems can be described by discrete Petri nets, and the continuous variable system can be described by the continuous Petri nets, the interaction between them is achieved by defining appropriate transition [1]. In order to enhance the description ability of hybrid Petri net, related extended hybrid Petri nets were presented, such as Batch Petri Nets [2], Hybrid Flow Nets [3], Differential Petri

Article number P_1985
Nets [4], First-Order Hybrid Petri Nets [5], General Hybrid Petri Nets [6] and Timed Hybrid Petri Nets [7]. These extended hybrid Petri nets are used to implement a variety of hybrid system in modeling, analysis, and scheduling. However, with the increase of the node number in a hybrid Petri net, its structure will be more complex so it is more difficult to schedule hybrid system which is modeled by hybrid Petri net. Traditionally, to overcome similar difficulty in discrete Petri nets, some solutions such as reduction, composition and decomposition technique were developed. Decomposition of Petri nets is one of very useful methods for analysis of structure-complex systems [8-10]. However, the aforementioned decompositions are only concerned with discrete Petri nets. In this paper, the decomposition technique for Timed Hybrid Petri Nets (THPN) is developed and the presented technique is investigated for scheduling of structure-complex hybrid systems. In the other hand, multi-core machines has become the mainstream of high performance computing platform with the development of multi-core machine and parallel technique, which bring new opportunities and challenges for hybrid system. Therefore, how to use multi-core machines to realize analysis and scheduling of hybrid system is an important problem.

2. Decompositions of THPN Based on Transition Attribution

For a more comprehensive introduction to hybrid Petri net, see [1]. The common notation and semantics for timed hybrid Petri net (THPN) can be found in [7]. In this section, the definitions and the properties of subnet of THPN were proposed firstly. After that, the decomposition algorithm of THPN was investigated.

Definition 1.
Let $\langle \text{THPN}, M(0) \rangle = (P, T, F, W, S, d, s, m(0))$ is a THPN, the function $f: T \rightarrow \{1, 2, \ldots, k\}$ is said to be the attribution function on the transitions of THPN if only if $f$ satisfies: for any two transitions $t_i, t_j \in T$, if there exists a place $p \in P$, such that $\{t_i, t_j\} \subseteq p$ or $\{t_i, t_j\} \subseteq p^*$, then $f(t_i) \neq f(t_j)$.

Definition 2.
Let $\langle \text{THPN}, M(0) \rangle = (P, T, F, W, S, d, s, m(0))$, and $f: P \rightarrow \{1, 2, \ldots, k\}$ is the attribution function on the transitions of THPN, $\text{THPN} = (P, T, F, W, S, d, s, m(0))$, $(S, d, s, m(0)) \in \{1, 2, \ldots, k\}$ is said to be a subnet of THPN based on transition attribution when $\text{THPN} \in \{1, 2, \ldots, k\}$ satisfy the following conditions:

$\bullet P^i = \{p \in P | \exists t_i \in T, p \in \text{Pre}(t_i)\}$;

$\bullet T^i = \{t_i \in T | f(t_i) = i\}$;

$\bullet F^i = (T^i \cup (T \times P)) \cap F$;

$\bullet M(0) \ni \Gamma_{p \rightarrow t}^i M(0)$;

$\bullet W^i): (P \times T^i) \cup (T^i \times P) \rightarrow Q^*$ is a weight function;

$\bullet (S^i): T^i \rightarrow R^*$ specifies the firing speeds associated to continuous transitions THPN.

$\bullet \Gamma_{p \rightarrow t}^i M(0)$ is the projection of $M(0)$ such that: $\forall p \in P \ni \Gamma_{p \rightarrow t}^i M(0)(p) = M(0)(p)$.

Theorem 1. Let THPN $\equiv (P, T, F, W, S, d, s, m(0))$, $M(0), \ x \in \{1, 2, \ldots, k\}$ is the subnet of $\langle \text{THPN}, M(0) \rangle = (P, T, F, W, S, d, s, m(0))$ based on transition attribution, we have:

For any $i, j \in \{1, 2, 3, \ldots, k\}, i \neq j$, then $T_i \cap T_j = \emptyset$.

Proof: We assume that there exists $i, j \in \{1, 2, 3, \ldots, k\}$ and $i \neq j$ such that $T_i \cap T_j = \emptyset$, then $T_i \cap T_j = \emptyset$. We assume that $\forall i \in \{1, 2, 3, \ldots, k\}$, $T_i \cap T_j = \emptyset$. Hence, there exists a place $P$ such that $P \cap P = \emptyset$, where $i \neq j$.

Theorem 2. Let THPN $\equiv (P, T, F, W, S, d, s, m(0))$, $i \in \{1, 2, 3, \ldots, k\}$ is the subnet of $\langle \text{THPN}, M(0) \rangle = (P, T, F, W, S, d, s, m(0))$ based on transition attribution, we have:

If the number of subnet (i.e. $k$) is bigger than 1, for any $i \in \{1, 2, 3, \ldots, k\}$, there exists $j \in \{1, 2, 3, \ldots, k\}$ such that $P \cap P = \emptyset$, where $i \neq j$.

Proof: We assume that there exists $i \in \{1, 2, 3, \ldots, k\}$, for any $j \in \{1, 2, 3, \ldots, k\}$ always satisfies $P \cap P = \emptyset$, where $i \neq j$. According to Theorem 1, we have $T_i \cap T_j = \emptyset$. It means that there exists a subnet THPN, does not connect with any other subnet. In fact, it is not difficult to know each subnet of THPN is a part of THPN, so it must be at least connected with one of the other subnet. Hence, the assumption is not correct.

According to Theorem 2, it is easy to know that $P \cap P = \emptyset$, so we have $P \cap P = \emptyset$. We assume that $P \cap P = \emptyset$ and $P \cap P = \emptyset$. Hence, there exists a place $p \in P$ such that for any $P_i$ (i.e. $P_i \in \{1, 2, 3, \ldots, k\}$) $\Rightarrow p \in P_i$. Thus, place $p$ does not belong to subnet $\langle \text{THPN}, M(0) \rangle \in \{1, 2, 3, \ldots, k\}$ and belongs to other subnet of THPN. It means that the number of subnet of THPN is 1.
is more than \( k \). Hence, the assumption is not correct. The theorem is proved.

**Definition 3.** A hybrid Petri net \(<THPN, M(0)> = (P, T, F, W, S_x, S_y, M(0))\) is a T-net if \( \forall p \in P, \) then \(|p| \leq 1\) and \(|p^*| \leq 1\).

**Theorem 3.** Let \( THPN=(P_i,T_i,F_i,W_i,S_{x_i},S_{y_i},M_i(0)), i \in \{1,2,\ldots,k\}\) be the decomposition subnet of \(<THPN, M(0)> = (P, T, F, W, S_x, S_y, M(0))\) based on transition attribution, then THPN is a T-net.

Proof: In the decomposition \( THPN_i = (P_i,T_i,F_i,W_i, (S_{x_i},S_{y_i},M_i(0))) \) \( i \in \{1,2,\ldots,k\}\), we assume that there is at least one net is not a T-net. Without loss of generality, let \( THPN_i=(P_i,T_i,F_i,W_i, (S_{x_i},S_{y_i},M_i(0))) \) \( i \in \{1,2,\ldots,k\}\) be not a T-net. Since THPN is not a T-net, there is at least one place \( p \notin P \) such that \(|p| > 1\) or \(|p^*| > 1\). In the case \(|p| > 1\), the place \( p \) has at least two input transitions. We can assume the two input transitions are \( t_1 \) and \( t_2 \), so we have \( t_1 \cap t_2 \notin \emptyset \). According to Definition 1, \( f(t_1) \neq f(t_2) \). Thus, \( t_1 \) and \( t_2 \) should be decomposed into different subnet. It means that \( \{t_1,t_2\} \) can not be the subset of \( p \). Hence, the assumption is not correct. In the case \(|p^*| > 1\), the place \( p \) has at least two output transitions. We can assume the two output transitions are \( t_1 \) and \( t_2 \), so we have \( t_1 \cap t_2 \notin \emptyset \). According to Definition 1, \( f(t_1) \neq f(t_2) \). Thus, \( t_1 \) and \( t_2 \) should be decomposed into different subnet. It means that \( \{t_1,t_2\} \) can not be the subset of \( p^* \). Hence, the assumption is not correct. The theorem is proved.

**Definition 4.** Let \( THPN=(P_i,T_i,F_i,W_i, (S_{x_i},S_{y_i},M_i(0))) (i \in \{1,2,\ldots,k\}) \) be a subnet of THPN, if \( THPN_i=(P_i,T_i,F_i,W_i, (S_{x_i},S_{y_i},M_i(0))) \) \( i \in \{1,2,\ldots,k\}\) satisfy the following conditions:

- \( P = P_1 \cup P_2 \), \( P_1 \cap P_2 \neq \emptyset \);
- \( T = T_1 \cup T_2 \), \( T_1 \cap T_2 \neq \emptyset \);
- \( F = F_1 \cup F_2 \);
- \( M(0)(p) = \begin{cases} M_1(0)(p) & p \in P_1 \\ M_2(0)(p) & p \in P_2 \end{cases} \)

then the THPN is said to be the sharing synthetic net of THPN1 and THPN2, denoted by \( THPN\equiv THPN_1 \cup THPN_2 \).

**Definition 5.** Let \( THPN=(P_i,T_i,F_i,W_i, (S_{x_i},S_{y_i},M_i(0))) (i \in \{1,2,\ldots,k\}) \) be a subnet of THPN, if \( THPN_i=(P_i,T_i,F_i,W_i, (S_{x_i},S_{y_i},M_i(0))) \) \( i \in \{1,2,\ldots,k\}\) satisfy the following conditions:

- \( P = \bigcup_{i=1}^{k} P_i \) and for any \( i \in \{1,2,\ldots,k\} \), there exists \( j \in \{1,2,\ldots,k\} \) such that \( i \neq j \) and \( P_i \cap P_j \neq \emptyset \);
- \( T = \bigcup_{i=1}^{k} T_i \) and for any \( i,j \in \{1,2,\ldots,k\} \), if \( i \neq j \), then \( T_i \cap T_j = \emptyset \);
- \( F = \bigcup_{i=1}^{k} F_i \);
- \( \text{If} p \in P \in \{1,2,\ldots,k\} \) then \( M(0)(p) = M_i(0)(p) \);

then the THPN is said to be the sharing synthetic net of \( THPN_1,THPN_2,\ldots,THPN_k \), denoted by \( THPN=\bigcup_{i=1}^{k} THPN_i \).

**Definition 6.** Let \( THPN=(P_i,T_i,F_i,W_i, (S_{x_i},S_{y_i},M_i(0))) (i \in \{1,2,\ldots,k\}) \) be the T-net, the attribution of transition \( t \) based on \( X \) is denoted by \( \gamma(t)_X \), where \( \gamma(t)_X = \{f(t')_X | \forall t' \in X \} \).

According to the above definitions and theorems, the decomposition algorithm of THPN was developed in this section. Firstly, the method to find the function of transition attribution for THPN was developed. After that, the details how to decompose THPN to several subnets based on the function of transition attribution was described. The details of algorithm can be seen in Algorithm 1.

**Algorithm 1.** Decomposition algorithm of THPN

*Input* \(<THPN,M(0)> = (P, T, F, W, S_x, S_y, M(0))\)

*Output* all subnets of THPN

1. **Step 1** Let \( X = T \), \( Y = \emptyset \), \( k = 1 \);
2. **Step 2** If \( X = \emptyset \), then go to Step 5, else for any transition \( t \in X \), compute \( \gamma(t)_X \);
3. **Step 3** If \( \forall t \in X \), \( \gamma(t)_X = \emptyset \), then go to step 4, else get a transition \( t' \) with \( \gamma(t')_X = \emptyset \); \( X = X - \{t'\} \), \( X = X - \emptyset \), go to Step 2;
4. **Step 4** Let \( X = X \cdot Y = Y \), \( k = k + 1 \), then go to step 2;
5. **Step 5** If \( X = \emptyset \) and \( Y = \emptyset \), then the subnet of THPN is itself and go to Step 10; else for any \( p \in P \), if \( p \in X \), then let \( f(p) = k \);
6. **Step 6** Let \( i = 1 \);
7. **Step 7** If \( i > k \), then go to step 10;
8. **Step 8** For any transition \( t_j \in X_i \), construct the subnet \( THPN_j \) which contains \( t_j \) as follows:
   1. **Initialization:** \( THPN_j \equiv \emptyset \);
   2. **All places and arcs associated with \( t_j \) in the THPN would be included in \( THPN_j \);
   3. **Let \( X_i^* = X_i - \{t_j\} \)**;
   4. **If \( X_i^* \) is empty, then go to (7); otherwise choose any transition \( t_n \) from \( X_i^* \);
   5. **All places and arcs associated with \( t_n \) in the THPN were included in \( THPN_j \) and a new net is obtained, denoted by \( THPN_j \equiv THPN_j \cup THPN_n \);
   6. **Let \( X_i^* = X_i^* - \{t_n\} \) then go to (4);
   7. **Output THPN_j**;
   8. **Step 9** Let \( i = i + 1 \), go to step 7;
   9. **Step 10** end.

3. **Parallel Scheduling of Hybrid System Based on Multi-core Cluster**

According to the proposed algorithm 1, we can decompose a complex THPN into several subnets. Since for any place \( p \) of any subnet of THPN such that \(|p| \leq 1\) and \(|p^*| \leq 1\), the structure of each subnet is simple. In this section, we would investigate the
parallel scheduling methods of hybrid system through subnet of THPN based on multi-core cluster. Unlike traditional work on hybrid system scheduling based on THPN, the scheduling method of hybrid system in this study is processed by many processors but only one processor, i.e., a hybrid system would be decomposed into several subsystems and the scheduling of subsystem would be processed in different processor at same time. The main idea of the proposed method as follows: Firstly, we should construct a THPN model for hybrid system and compute all subnets of THPN according to Algorithm 1. Secondly, each subnet of THPN is reflected as a thread and each thread is mapped to a processor of multi-core cluster, also any place which is shared by different subnet is mapped to a processor of multi-core cluster too. Thirdly, the parallel model of master-slave is used to realize parallel scheduling in this study. Each subsystem (i.e., subnet or thread) is scheduled in the respective processor, and the message between the threads is passed through the corresponding server threads and the synchronization between the threads is coordinated by the main thread.

**Algorithm 2** parallel scheduling algorithm for hybrid system based on multi-core cluster

*Input* a hybrid system

*Output* scheduling results of hybrid system

*Step 1.* Construct the THPN model for the input hybrid system;  
*Step 2.* Compute all subnets of THPN based on the decomposition algorithm of THPN (as algorithm 1), and each subnet corresponds to a thread;  
*Step 3.* Find any common place between two threads and put each common place as a server thread;  
*Step 4.* All threads are mapped to different processors of multi-core cluster;  
*Step 5.* Input time horizon TH, and let \( k=0, \ z=0; \)  
*Step 6.* Initialize subnet of THPN according to Step1 of algorithm 6.1 in the literature [7];  
*Step 7.* Firstly, each thread calculate the end time of the current IB state according to Step2 to Step12 of algorithm 6.1 in the literature [7]; Secondly, all results are sent to the main thread; Thirdly, the main thread compute the minimum of the end time (denoted by min_endtime); Finally, the minimum of the end time is distributed to each thread and let \( \tau_{k+1}=\text{min}_{\text{endtime}}; \)  
*Step 8.* Compute the marking of \( p_i \) according to the follow formula, where \( p_i \) is any unshared place in every thread.

\[
\begin{bmatrix}
 m^0(\tau_{k+1}) \\
 m^1(\tau_{k+1})
\end{bmatrix} = \begin{bmatrix}
 m^0(\tau_k) \\
 m^1(\tau_k)
\end{bmatrix}
\begin{bmatrix}
 W^0 \\
 W^1
\end{bmatrix}
\begin{bmatrix}
 n^0(\tau_{k+1}) \\
 n^1(\tau_{k+1})
\end{bmatrix}
+ \begin{bmatrix}
 0 \\
 0
\end{bmatrix}
\begin{bmatrix}
 i_{k+1} \\
 v^1(\tau_k)
\end{bmatrix}
\]

*Step 9.* Compute \( B(\text{SUBNET}_{\tau_k}) (\tau_{k+1} - \tau_k) \) for any shared place \( p_i \) and upload the result to its server thread, where \( B(\text{SUBNET}_{\tau_k}) \) is the dynamic marking balance of place \( p_i \) in SUBNET at time \( \tau_k \).

*Step 10.* Compute the marking of shared place \( p_i \) at time \( \tau_{k+1} \) according to the follow formula:

\[
 m(\tau_{k+1}) = m(\tau_k) + \sum_{\text{SERVICE}_{u}} (m(\tau_k) + m(\tau_{k+1} + 1) - m(\tau_k) - m(\tau_{k+1})).
\]

where \( \text{SERVICE}_{u} \) is the subnet which contains \( p_i \).

*Step 11.* If \( \text{TH} > \tau_{k+1} \), then \( k \leftarrow k+1 \) and go to Step7, else output the result.

4. Case Study

One case study is presented in this section to show the correctness and efficiency of the proposed approach in scheduling of hybrid system by experiments. We consider a production system presented in literature [7], where the production system is a typical hybrid system. The THPN of the production system is presented in Fig. 1, which contains discrete-transitions synchronized on external events \( t_1, t_3, t_4, t_8, t_9, t_{11}, t_{12}, \) and \( t_{16} \), discrete-transitions with constant timings \( t_2, t_6, t_{10}, \) and \( t_{14} \), and immediate discrete-transitions \( t_5, t_7, t_{13} \) and \( t_{15} \) for which neither an event nor a delay is written in the figure. A constant flow rate is associated with every continuous-transition. In our experiment, the scheduling process (i.e. the behavior evolution) of hybrid system is concerned. Experiments are performed on a machine with AMD 6272 64-cores with 2.1 GHz CPUS with 128 G of RAM, software environment with Windows server 2008 and compiling environment with Visual Studio 2010.

Firstly, compute all subnets of the THPN based on the algorithm 1. It is not difficult to obtain all subnets of THPN, given in Fig. 2. Obviously, any subnet of THPN is \( T \)-net.

Secondly, each subnet of THPN is mapped to one processor of multi-core cluster and is scheduled paralleled by the parallel model of master-slave. Tables 1 to 6 are the results of scheduling of subnet 1 to subnet 6 respectively, where the time horizon (i.e. the time length of scheduling) is equal to 30. The table lists the marking of discrete place, the initial marking of continuous place of IB state, running time interval and the events to change IB state, where the symbol \( t_i(j) \) means that IB state is changed by transition \( t_i \) and transition \( t_j \) is the transition of the \( j \)-th subnet of THPN. The result of the above parallel scheduling(PS) method is consistent with the result of the non-parallel scheduling(NPS) method which is proposed in literature [7]. It means that the parallel scheduling method in this study provides a correct way for the analysis and scheduling of large-scale hybrid systems by decomposition THPN with multi-core cluster.
Fig. 1. THPN model of a hybrid system.

Fig. 2 (a). Subnet 1.

Fig. 2 (b). Subnet 2.

Fig. 2 (c). Subnet 3.
Finally, to illustrate the efficiency of the proposed method in this paper, the time cost of computer process of PS and the time cost of computer process of NPS is compared. Fig. 3 presented the time cost of computer process of PS and NPS in the case which the time horizon is equal to 30, $3 \times 10^2$, $3 \times 10^3$, $3 \times 10^4$, $3 \times 10^5$, $3 \times 10^6$ and $3 \times 10^7$ respectively, where the time unit is $10^{-6}$ s. According the data from Fig. 3, there is no evident difference between the time cost of computer process of PS and NPS when the time horizon is short, for example, when the time horizon is equal to 30 and $3 \times 10^2$. However, since the PS method makes full use of the different machines of the multi-core processors to achieve the schedule of each subnet, the time cost of computer process of PS is obviously less than the NPS' with the increase of time horizon.

Table 1. Scheduling process for subnet1 (TH=30).

<table>
<thead>
<tr>
<th>State</th>
<th>Marking</th>
<th>Time interval</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 0 1 1 1</td>
<td>[0,4)</td>
<td>$t_1(1)$</td>
</tr>
<tr>
<td>2</td>
<td>2 0 1 1 1</td>
<td>[4,10)</td>
<td>$t_1(1)$</td>
</tr>
<tr>
<td>3</td>
<td>2 0 1 1 1</td>
<td>[10,15)</td>
<td>$t_1(5)$</td>
</tr>
<tr>
<td>4</td>
<td>2 0 1 1 1</td>
<td>[15,18.5)</td>
<td>$t_1(2)$</td>
</tr>
<tr>
<td>5</td>
<td>2 0 1 1 1</td>
<td>[18.5,20)</td>
<td>$t_1(5)$</td>
</tr>
<tr>
<td>6</td>
<td>2 0 1 1 1</td>
<td>[20,22.5)</td>
<td>$t_1(2)$</td>
</tr>
<tr>
<td>7</td>
<td>2 0 1 1 1</td>
<td>[22.5,30)</td>
<td>$t_1(3)$</td>
</tr>
</tbody>
</table>

Table 2. Scheduling process for subnet2 (TH=30).

<table>
<thead>
<tr>
<th>State</th>
<th>Marking</th>
<th>Time interval</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 0</td>
<td>[0,4)</td>
<td>$t_1(1)$</td>
</tr>
<tr>
<td>2</td>
<td>1 0</td>
<td>[4,10)</td>
<td>$t_1(1)$</td>
</tr>
<tr>
<td>3</td>
<td>1 0</td>
<td>[10,15)</td>
<td>$t_1(5)$</td>
</tr>
<tr>
<td>4</td>
<td>1 0</td>
<td>[15,18.5)</td>
<td>$t_1(2)$</td>
</tr>
<tr>
<td>5</td>
<td>1 0</td>
<td>[18.5,20)</td>
<td>$t_1(5)$</td>
</tr>
<tr>
<td>6</td>
<td>1 0</td>
<td>[20,22.5)</td>
<td>$t_1(2)$</td>
</tr>
<tr>
<td>7</td>
<td>1 0</td>
<td>[22.5,30)</td>
<td>$t_1(3)$</td>
</tr>
</tbody>
</table>

Table 3. Scheduling process for subnet3 (TH=30).

<table>
<thead>
<tr>
<th>State</th>
<th>Marking of Discrete Place</th>
<th>Marking of Continuous Place</th>
<th>Firing Speed</th>
<th>Time Interval</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>m_1 m_2 m_3 m_9 m_10 m_17 m_18 m_19 m_20 m_21 v_17 v_18</td>
<td>0 800 0 0 0 0 0</td>
<td>[0,4)</td>
<td>$t_1(1)$</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1 1 2 0 1 0 800 0 0 0 0 0 0</td>
<td>(4,10)</td>
<td>$t_1(1)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1 0 2 0 1 500 300 0 0 0 0 0 0</td>
<td>[10,15)</td>
<td>$t_1(5)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1 0 2 0 1 400 400 0 0 0 0 0 0</td>
<td>20 0</td>
<td>$t_1(2)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1 0 2 0 1 330 470 70 0 0 0 0 0</td>
<td>20 0</td>
<td>$t_1(5)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1 0 2 1 1 300 500 100 0 0 0 0 0</td>
<td>20 20</td>
<td>$t_1(2)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1 0 2 1 1 250 550 100 50 50 50 20 20</td>
<td>[22.5,30)</td>
<td>$t_1(3)$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4. Scheduling process for subnet4 (TH=30).

<table>
<thead>
<tr>
<th>State</th>
<th>Marking of discrete place</th>
<th>Marking of continuous place</th>
<th>Firing speed</th>
<th>Time interval</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>m11 m12 m13 m16</td>
<td>m22 m23 m24 m25 v19 v20</td>
<td></td>
<td>[0,4)</td>
<td>t1(1)</td>
</tr>
<tr>
<td>1</td>
<td>1 0 1 0 0 0 0 0 0 0</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1 0 1 0 0 0 0 0 0 0</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td></td>
<td>[4,10)</td>
<td>t2(1)</td>
</tr>
<tr>
<td>3</td>
<td>1 0 1 0 0 0 0 0 0 0</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td></td>
<td>[10,15)</td>
<td>t5(5)</td>
</tr>
<tr>
<td>4</td>
<td>1 0 1 0 0 0 0 0 0 0</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td></td>
<td>[15,18.5)</td>
<td>t6(2)</td>
</tr>
<tr>
<td>5</td>
<td>1 1 1 0 0 0 0 0 0 0</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td></td>
<td>[18,5,20)</td>
<td>t6(5)</td>
</tr>
<tr>
<td>6</td>
<td>1 1 1 0 0 0 0 0 0 0</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td></td>
<td>[20,22.5)</td>
<td>t6(2)</td>
</tr>
<tr>
<td>7</td>
<td>1 0 1 0 700 0 0 0 0 0 10 0</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td></td>
<td>[22,5,30)</td>
<td>t6(3)</td>
</tr>
</tbody>
</table>

Table 5. Scheduling process for subnet5 (TH=30).

<table>
<thead>
<tr>
<th>State</th>
<th>Marking of discrete place</th>
<th>Marking of continuous place</th>
<th>Time interval</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>m7 m8 m9 m19 m21</td>
<td></td>
<td>[0,4)</td>
<td>t1(1)</td>
</tr>
<tr>
<td>1</td>
<td>1 0 0 0 0</td>
<td></td>
<td>0 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1 0 0 0 0</td>
<td></td>
<td>[4,10)</td>
<td>t2(1)</td>
</tr>
<tr>
<td>3</td>
<td>1 0 0 0 0</td>
<td></td>
<td>[10,15)</td>
<td>t5(5)</td>
</tr>
<tr>
<td>4</td>
<td>0 1 0 0 0</td>
<td></td>
<td>[15,18.5)</td>
<td>t6(2)</td>
</tr>
<tr>
<td>5</td>
<td>0 1 0 0 0</td>
<td></td>
<td>[18,5,20)</td>
<td>t6(5)</td>
</tr>
<tr>
<td>6</td>
<td>0 0 1 100 0</td>
<td></td>
<td>[20,22.5)</td>
<td>t6(2)</td>
</tr>
<tr>
<td>7</td>
<td>0 0 1 100 50</td>
<td></td>
<td>[22,5,30)</td>
<td>t6(3)</td>
</tr>
</tbody>
</table>

Table 6. Scheduling process for subnet6 (TH=30).

<table>
<thead>
<tr>
<th>State</th>
<th>Marking of discrete place</th>
<th>Marking of continuous place</th>
<th>Time interval</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>m7 m15 m16 m23 m25</td>
<td></td>
<td>[0,4)</td>
<td>t1(1)</td>
</tr>
<tr>
<td>1</td>
<td>1 0 0 0 0</td>
<td></td>
<td>0 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1 0 0 0 0</td>
<td></td>
<td>[4,10)</td>
<td>t2(1)</td>
</tr>
<tr>
<td>3</td>
<td>1 0 0 0 0</td>
<td></td>
<td>[10,15)</td>
<td>t5(5)</td>
</tr>
<tr>
<td>4</td>
<td>0 0 0 0 0</td>
<td></td>
<td>[15,18.5)</td>
<td>t6(2)</td>
</tr>
<tr>
<td>5</td>
<td>0 0 0 70 0</td>
<td></td>
<td>[18,5,20)</td>
<td>t6(5)</td>
</tr>
<tr>
<td>6</td>
<td>0 0 0 0 0</td>
<td></td>
<td>[20,22.5)</td>
<td>t6(2)</td>
</tr>
<tr>
<td>7</td>
<td>0 0 0 0 0</td>
<td></td>
<td>[22,5,30)</td>
<td>t6(3)</td>
</tr>
</tbody>
</table>

Fig. 3. The comparison of running time.

5. Conclusions

With the deep research of hybrid Petri nets, hybrid Petri net has become an effective tool for modeling and scheduling of the hybrid system. This study represents a new way to schedule complex hybrid system based on decomposition of timed hybrid Petri nets (THPN). A THPN model of
A complex hybrid system could be decomposed to several simple subnets, where each subnet is a T-net. Hence, the complexity of scheduling of hybrid systems can be decreased effectively. Further, scheduling of hybrid system is developed with parallel model of master-slave based on multi-core clusters. It is verified that the proposed approach is an effective method of scheduling of complex hybrid system.

Acknowledgements

The project was supported by open fund of Guangxi Key laboratory of hybrid computation and IC design analysis [2012HCIC01], and by Guangxi Natural Science Foundation [2012GXNSFAA053218], and by Guangxi Colleges and Universities Scientific Research Projects [2013ZD040].

References