Distributed processing based on Timed Petri Nets

xianwen Fang\textsuperscript{1}\textsuperscript{*}, zhicai Xu\textsuperscript{2}, zhixiang Yin\textsuperscript{1}

\textsuperscript{1}Maths and Physics Department, Anhui University of Science and Technology, Huainan 232001, China; xwenfang@tom.com
\textsuperscript{2} Chuzhou University, Anhui Chuzhou, 239012, China, xwliu@aust.edu.cn

Abstract

At present, in the design and analysis of discrete event systems, the parallel or distributed simulation mechanism is adopted to make these models occur concurrently, then the problem will be easy to be solved, the validation of their models is often addressed via simulation. Lookahead computation is a good method that improves the distributed simulation performance of Timed Transition Petri Nets (TTPN). Extended Timed Petri Nets (ETTPN) is extended in time based on TTPN, it meet Lookahead requirements, and use the Lookahead to analyze the ETTPN model for finding the concurrency and blocking structure, thereby making certain the logical process (LP). Based on the specialties of Lookahead, the Lookahead was applied in parallel simulation, and acquire the sufficient condition about the ETTPN model existing concurrency. In order to reduce error rate,a improved mapping algorithm is presented.

Keywords: Timed Petri Nets; Lookahead; Distributed Processing; Mapping; Logical Process

1. Introduction

For analyzing the big system model, adopting serial discrete incident simulation has seemed insufficiently. At present, we mainly carry out the way, that is adopting the parallel or distributed simulation mechanism and corresponding software tool to make these models occur concurrently, then the problem will be easy to be solved. In the distributed simulation in order to make submodels carry out asynchronously and concurrently, the space decomposing technique correlating with time is introduced. Here, we also call the submodel for the logical process (LP). Lookahead is an important concept in the distributed simulation time management protocol; every simulation entity has time mark of the event oneself producing through Lookahead to notify other entities earlier, in order to accelerate the operation of the procedure. In the High Level Architecture (HLA) standard the explanation of Lookahead function is:” guaranteeing all members of the union not to produce the event whose time mark is smaller than presently the union member of time plus the Lookahead”. In literature [8], the Lookahead conception of Timed Petri Nets in distributed simulation has been proposed. We use the Lookahead to analyze the TTPN model for finding the concurrency and blocking structure, thereby making certain the logical process (LP). The model can be classified into some submodels (logical process) according to the characteristics of the model and the logical processes are performed on the nodes of the parallel machine, which can result in the goodness of the performance and treatment. In literature [9],the Lookahead based on the timed place petri net is presented, and applied to distributed processing. The mapping of the process on the processor relate to the executing time of distributed processing, so the performance analysis of the distributed processing is important. Extended Timed Petri Nets (ETTPN) is extended in time based on TTPN, it meet Lookahead requirements, in this paper, we use the Lookahead to analyze the ETTPN model for finding the concurrency and blocking structure, and acquire the sufficient condition about the ETTPN model existing concurrency, then give out the partition algorithm about distributed processing.

2. Based conception
Here, we only introduce several conceptions correlating with the paper close, other Petri Nets terms in the literature [1].

Definition 1[1]. A 6-tuple $N=(P; T; F; K; W; M)$ is called a place/transition net (P/T-net) iff

1. $(P; T; F)$ is a net;
2. $(K) : P \rightarrow N_0 \ (N_0$ is natural number set, gives a (possibly unlimited) capacity for each place;
3. $(W) : F \rightarrow N_0 - \{0\}$, attaches a weight to each of the net;
4. $M : P \rightarrow N_0$ is the initial marking, respecting the capacities, i.e. $\forall p \in P : M(p) \leq K(p)$
5. $(A transition t \in T$ is enabled iff
   $\forall p \in t^- : M(p) \geq W(p, t)$
   $\forall p \in t^+ : M(p)+W(t-t) \leq K(p)$
   If $M[t>M$ (a enabled transition $t$ may yield a follower marking $M'$ of $M$), for $\forall p \in P :$
   $M'(p)=
   \begin{cases} 
   M(p)-W(p, t) & \text{if } p \in t^- \text{ and } t^*
   
   M(p)+W(t-t) & \text{if } p \in t^- \text{ and } t^*
   
   M(p)-W(p, t)+W(t-t) & \text{if } p \in t^- \text{ and } t^* \cup t^*
   
   M(p) & \text{if } p \in t^- \text{ and } t^*
   \end{cases}$

Definition 2. A 6-tuple $ETTPN=(P; T; F; \tau; W; M)$ is called extended timed transition petri net (ETTPN), iff

$P=\{p_1, p_2, \cdots, p_n\} (n \geq 0)$ is a finite set of place;
$T=\{t_1, t_2, \cdots, t_m\} (m \geq 0)$ is a finite set of transition,
and $P \cap T = \Phi$;
$F \subseteq (P \times T) \cup (T \times P)$ is a directed arcs set;
$\tau : T \rightarrow \{0, 1, 2, 3, \cdots\}$ is a time mapping function; when $\tau = 0$ is instantaneous transition, otherwise is timed transition.
$W : F \rightarrow \{1, 2, 3, \cdots\}$ is called weight function;
$M : P \rightarrow I (I = \{0, 1, 2, \cdots\})$ is net marking.

The transition enabled rules of extended timed transition petri net is similar to the P/T net, but in the ETTPN, the mark $M$ enabled $t$ firing, and the transition $t$ can fire completely after $\tau$ time.

3. The Lookahead based on ETTPN in the distributed simulation

The Lookahead computing of the ETTPN is based on the structure of each subnet and the last state of the net running; it is linked to the end place and the path from the source place to the end place. From the source place to the end place the basic forms about ETTPN as follows(Fig.1):
graph, then according to the corresponding Lookahead computing of the five kinds of basic structures, we calculate the Lookahead of the whole prediction graph. Calculating the having definite initial marking Lookahead is correlated with the initial marking, that looking for the initial marking place as the source place to calculate Lookahead, their solving methods are similar elementally.

4. The Lookahead analyzing in distributed processing based on ETTPN

In the parallel simulation of petri net, the general method is to resolve the complicated network structure into several sub nets (logic process), then let these sub nets be assigned to each node of the parallel machine to deal with, thus improve the simulation result. In literature [7], A.Chiola provides the method that cuts apart of models according to the structure of the net, the core is separating the concurrent structure of the net, make it different logic processes, and then assign to the parallel nodes, in order to reduce time and communication expenses; conflicts or the blocking structure that waits for will be put on the same logical process in the information transmission, in order to reduce the communication expenses mainly.

Definition 3. Provided t (a) means the part simulation time of transition a, given transition x and y, if t (x) < t (y), then it is called as transition x happened prior to transition y, written as x \( \succ \) y.

Definition 4. If z happens, it must be x \( \succ \) z, then called as x schedules y.

Definition 5. Provided LP means a logic process, x is the transition of Lpi, y and z are the transitions of Lpj, if x can schedule z and z \( \succ \) y, then we call transition x can influence transition y. Else transition x can’t influence transition y.

Definition 6. Provided x and y are the transitions of Lpi and Lpj respectively, if x can’t influence y and y can’t influence x, then we call x and y are concurrent.

When analyzing the timed transition petri net model by using Lookahead, we may partition the model into several logical processes at the beginning, let each Lp only includes a transition, then analyze the relation among each Lp, combine them lastly. In addition, we will put the transition of conflict structure on one Lp directly according to the method of Chiola, and other structure needing to be put on the same LP is done well in advance (As for many transitions being put on the same LP, we can design an event table by time stamp). Then analyze these LPs with theirs Lookahead, so that we can know the blocking and concurrency of these Lps.

Theorem 1. Provided x and y are the unique transition of Lpi and Lpj respectively, Laij is the having definite initial marking Lookahead of connecting Lpi with Lpj, if t (x) + Laij < t (y), then x influences y. if t (x) + Laij \( \succ \) t (y), then x can’t influence y.

Theorem 2. Provided x and y are the first transition of the event tables of Lpi and Lpj respectively, Laij is the having definite initial marking Lookahead of connecting Lpi with Lpj, Laji is the having definite initial marking Lookahead of connecting Lpj with Lpi, if t (x) + Laij \( \geq \) t (y) and t (y) + Laji \( \geq \) t (x), then x and y are concurrent.

According to the concept of Lookahead and the Theorem 2, when we simulate the complicated timed petri net structure on the parallel machine, we can confirm the logical process through analyzing which events are concurrent and which events may be blocked. In this way we can reduce the simulation expenses greatly, and improve simulation performance.

5. The improved partitioning algorithm on the issue of process-processor mapping

The study about process-processor mapping has lots of methods, the process – processor mapping issue can be developed in literature[2], the most disadvantage of this method is the neglection of influence of communication spending. Nicol and Mao raised a method based on time petri net and nodes combination mechanism [3]. The main concept of the method is the nodes combination and developing line links, nodes which having big communication spending come to be adjacent or at least near from each other. Nandy and Loucks developed an iterative arrange task method [4] which can reduce the communication spending with the mean of keeping every processors loading balance and optimize operation time.

For the case of process N is bigger, it is complex for task graph method. We can denote the task graph by following matrix, and then give the best mapping plan by analyzing the matrix[8], a mapping algorithm is presented in the literature[8], the algorithm is avail by simulation, but mapping error rate is big. So, a improved algorithm is presented in order to reduce error rate.

\[
A = \begin{bmatrix}
1 & a_{10} & a_{11} & a_{12} & \cdots & a_{1n} \\
2 & a_{20} & a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
N & a_{n0} & a_{n1} & a_{n2} & \cdots & a_{nn}
\end{bmatrix}
\]

Corresponding process

A improved algorithm partitioning algorithm about
process-processor mapping as follows:

1. Plot the task graph or find communication time between every process and executing time of each process.

2. We write an \(n \times (n+1)\) matrix \(A\), where \(a_{i0} = t_i\), \(a_{ij} = c_{ij}\) (\(1 \leq i \leq n, 1 \leq j \leq n\)), \(a_{ii} = 0\), \(a_{ij} = a_{ji}\).

3. Select the minimum element from the zeroth column in matrix \(A\), using the notation \(a_{m0} = \min\{a_{ij}\}\) i.e. the element of the \(m\) row and the 0 column, and then select the maximum value of the elements denoted by \(a_{mk}\) from the \(m\) row, i.e. \(a_{mj}\) (\(1 \leq j \leq n\)).

4. If \(a_{mk} \geq a_{m0}\) let \(a_{k0} = a_{k0} + a_{m0}\), \(a_{kn} = a_{kn} + a_{m0}\), \(a_{kj} = a_{kj} + a_{mj}\), \(a_{mk} = 0\), \(a_{jk} = a_{kj}\) (\(1 \leq j \leq n, j \neq m\)). If \(a_{mk} < a_{m0}\) and \(a_{m0} + a_{k0} - \max\{a_{ij}\} > a_{mk}\) then select the \(m\) from the matrix \(A\) but do not take the row having minimum computing time into account, until \(a_{m0} + a_{k0} - \max\{a_{ij}\} < a_{mk}\).

5. Delete all the elements of the \(m\) row and the \(m\) column of the matrix, accordingly, delete the corresponding process \(m\), and process corresponding to the \(k\)-th row is denoted by \(k+m\).

6. Repeat the above procedure until the matrix \(A\) coming to be \(P \times (P+1)\) matrix. Then there are \(p\) notations corresponding to the processes, every element in the notations is the process mapped on the same processor. Now the total operating time of the system is denoted by:

\[
(1 \leq i \leq p, 1 \leq j \leq p)
\]

The improved algorithm is on the basis of the algorithm presented in literature[8], a mount of experimental data verifies that the algorithm is quite effective. The method described above has been applied to the analysis of 10 processes mapping on 4 processors, and the computing time of the process is random produced. We define the rate between time of communication and of computing as the notation \(comm/comp=\) communication time between two processes/the sum of computing time of two processes. Then the communication time can be determined by the time of computing and \(comm/comp\). Assume error rate=(simulating time – optimizing time)/optimizing time.

We get the two methods error rate as shown in Fig.3 by giving different \(comm/comp\) value. It can be seen that the improved algorithm is better than the basic algorithm.

Example 1 let us consider 6 processes mapping on 3 processors. The computing time is 7, 4, 8, 5, 3, and 6 respectively. The communication time is given in Table 1 and the data in the table is communication time of two processes.

Table 1. The communication time of interact processes

<table>
<thead>
<tr>
<th>process</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>1.5</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0.6</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>1.5</td>
<td>1</td>
<td>0.5</td>
<td>0.6</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>8</td>
<td>3</td>
<td>7</td>
<td>0.4</td>
</tr>
</tbody>
</table>

If we adopt mapping algorithm in literature[8], it is no avail, the apply of improved mapping method as follows:

\[
A = \begin{bmatrix}
7 & 0 & 1 & 3 & 4 & 1.5 & 2 \\
4 & 1 & 0 & 5 & 3 & 1 & 8 \\
8 & 3 & 5 & 0 & 2 & 0.5 & 3 \\
5 & 4 & 3 & 2 & 0 & 0.6 & 7 \\
3 & 1.5 & 1 & 0.5 & 0.6 & 0 & 0.4 \\
6 & 2 & 8 & 3 & 7 & 0.4 & 0
\end{bmatrix}
\]

(\(\text{the operating time of the system is } 8+42=50\))

\[
A = \begin{bmatrix}
1 & 7 & 0 & 3 & 4 & 1.5 & 3 \\
3 & 8 & 3 & 0 & 2 & 0.5 & 8 \\
4 & 5 & 4 & 2 & 0 & 0.6 & 10 \\
5 & 3 & 1.5 & 0.5 & 0.6 & 0 & 1.4 \\
6 & 2 & 10 & 3 & 8 & 10 & 1.4 & 0
\end{bmatrix}
\]

(\(\text{the operating time of the system is } 10+34=44\))
The operating time of the system is \(15+24=39\).

Optimal partition is \(\{3\}, \{5\}, \{1,2, 4, 6\}\), and the corresponding time is 39.

6. Conclusions

In the conservative distributed simulation, calculating Lookahead is an effective method of promoting the parallel simulation performance, it lets us analyze the structure of net firstly, to reduce the communication expenses for blindly partitioning the net, and let the concurrent logical process operate in the same node of the parallel machine, thus improve simulation performance. In the paper we present the improved partitioning algorithm about the process-processor mapping, this will reduce error rate, and promote the performance of the parallel simulation.

References


Author(s) Biography

xianwen Fang was born in 1975 in Henan province, China. He is a Ph.D. student. His research interests are parallel computing and Petri nets.

zhicai Xu is a Ph.D, professor. He was born in Anhui province in 1958. His research interest is Petri nets and system simulation.

zhixiang Yin is a Ph.D, professor. He was born in Anhui province in 1966. His research interest is system simulation.