http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 9 (2): 319-324, 2010
ISSN 1812-5638

Partition and Mapping Method of Distributed System Based on Petri Nets

[1]Xiangwei Liu and [1,2]Xianwen Fang
[1]College of Economy and Management, Anhui University of Science and Technology,
Huainan Anhui, 232001, China
[2]Department of Computer Science and Technology, Tongji University, Shanghai, 201804, China

Abstract: For analyzing large discrete event systems, the validation of their models is often addressed via simulation. The study presents lookahead computation methods which is based on a part of optimism computed on the prediction time each logical process can determine for its advancement, so it is a good method that improves the distributed simulation performance based on Timed Transition Petri Nets (TTPN). Then, the partition methods of distributed system are proposed by using the specialties of lookahead, which can avoid to partition model blindly. Aimed to the deficiency of process-processor mapping algorithm, an improved algorithm is presented. Theoretical analysis and experimental results indicate that the improved algorithms are very effective.

Key words: Timed petri nets, distributed system, partition method, improved mapping algorithm

INTRODUCTION

For analyzing the big system model, adopting serial discrete incident simulation have 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 method, it introduces the space decomposing technique correlating with time and carry out asynchronously and concurrently. 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 study of Nketsa and Khalifa (2001), the lookahead conception of timed place petri nets in distributed simulation has been proposed. Joseph (2002) proposed a generalized partitioning scheme when transitions in structural conflicts have to be fired. Drawback of the approach is the great number of null messages sent when they implemented the conservative method. Janneck (1998) has decomposed the global Petri net such that an LP is created for every place and every transition resulting in a large number of LPS. The possibly tremendous amount of messages inherent to this approach prevents efficiency if simulated in distributed memory architecture.

We use the lookahead to analyze the TTPN model (Murata, 1989) for finding the concurrency and deadlock structure, thereby making certain the Logical Process (LP). The model can be partitioned into some LPS (Jin *et al*., 2008) according to the characteristics of the model and the LPS are executed on the parallel machine. Aimed to the deficiency of the mapping algorithm (Bassel *et al*., 2008). We present the denotation matrix of task graph and give out the improved algorithm that can decide quickly the process-processor mapping. Lots of experiments show that the algorithm is very effective.

THE LOOKAHEAD STUDY OF TIMED TRANSITION PETRI NET

In different distributed simulation environments and different protocols, the concept meaning of lookahead is different subtly. In the conservative distributed simulation protocol, the event handling model of logical process adopts the way of the single queue single server, in this situation conveying the time stamp of event between logical process adopts three pieces of time amount (Fig. 1): t_{cause}: the moment of sending events which entity knows at first, can be also regarded as source time mark of event; t_{commit}: the moment that the event can be committed and sent, the moment that the source event is finished; t_{effect}: as the effective moment of time mark of the sent event, or as time mark of the sent event.

Corresponding Author: Xiangwei Liu, College of Economy and Management, Anhui University of Science and Technology, Huainan Anhui, 232001, China

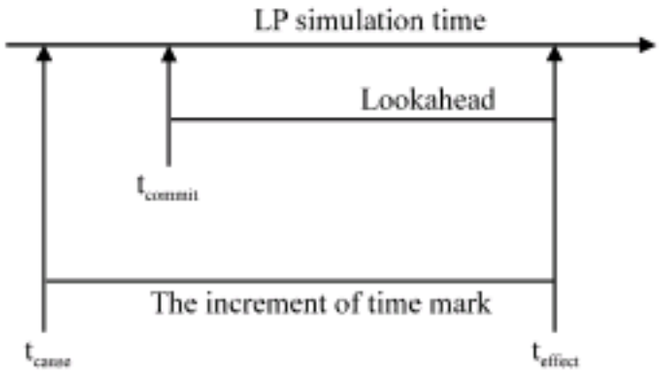

Fig. 1: The indication of lookahead

So, the definition of lookahead is adopted $\text{Min}\{t_{effect}-t_{commit}\}$, it means the minimum of the increment of time mark in the simulation course.

Definition 1: Provided t_{commit} means the moment that the event can be committed and sent, t_{effect} means the effective moment of time mark of the sent event, $La = \text{Min}\{t_{effect}-t_{commit}\}$, which means the minimum of the increment of time mark in the simulation course. Then La is called as Lookahead.

The lookahead computing of the Timed Petri net is based on the structure of each subnet and the last state of running; it is linked to the end place and the transition from the source place to the end place. From the source place to the end place the basic forms in Fig. 2.

For some complicated Petri network structure, it is difficult for us to produce lookahead directly. Next we adopt the prediction graph algorithm to turn the structure graph with loop to the prediction graph without loop. The prediction graph algorithm of lookahead as follows:

(1) The prediction graph is built according to the shifting of the end place in the opposite direction; the node of the prediction graph is made up of place and transition
(2) The initial node of the prediction graph is the end place PS, if there is many end places and then the prediction graph includes many initial nodes
(3) The next node of the initial node of the prediction graph is the preset of PS (it is a transition)
(4) Next, we find the preset of the current node of the prediction graph from Petri net structure graph and make it as the next node of the node, if the current node has many preset, then each preset element is made a branch node. In accordance with above-mentioned methods we go on to search for the preset each branch node until without preset or presenting the following situations:

- The preset of some node has appeared in the prediction graph and if connecting the preset of the node with the node of prediction graph which has been produced can form circulating structure (has direction), then we can't add the preset of the node to the prediction graph, the operations of the branch are cancelled until the place node which is added lately (if the node is a place, then we end with the node)
- If the preset of some node has appeared in the prediction graph, but it doesn't form circulation by direction, then the preset of the node isn't added (that is public)

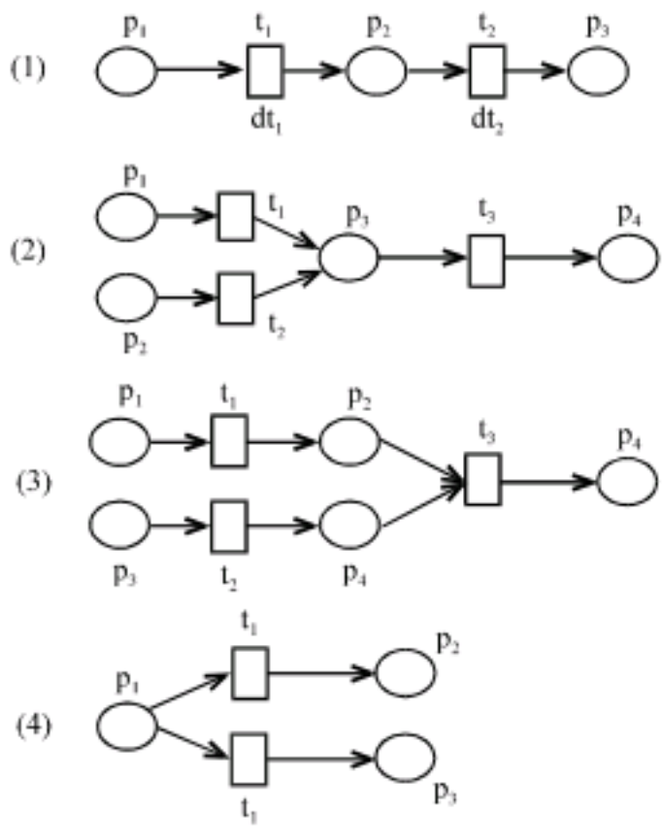

Fig. 2: The four basic structures

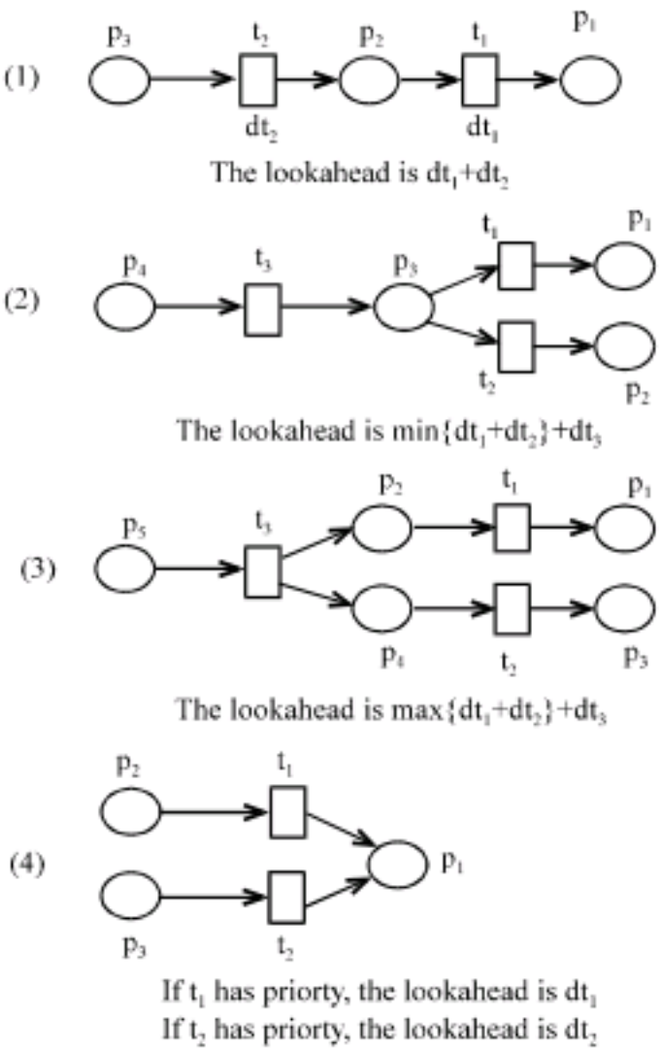

Fig. 3: The corresponding lookahead

(5) When having added the source place to the graph and other branches have ended according to the term 4, the prediction graph is completed

According to the prediction graph algorithm of solving lookahead, the four kinds of basic forms prediction graph from the source place to the end place and corresponding lookahead in Fig. 3.

THE PROCESS PARTITION ING BASED ON LOOKAHEAD

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. Nketsa and Khalifa (2001) provided 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 2: 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≻y.

Definition 3: If z happens, it must be x≻z, then called as x schedules z.

Definition 4: Provided LP means a logic process, x is the transition of L_{pi}, y and z are the transitions of L_{pj}, if x can schedule z and z≻y, then we call transition x can influence transition y,else transition x can't influence transition y.

Definition 5: Provided x and y are the transitions of L_{pi} and L_{pj}, respectively, if x can't influence y and y can't influence x, then we call x and y are concurrent.

Theorem 1: Provided x and y are the unique transition of L_{pi} and L_{pj}, respectively, La_{ij} is the having definite initial marking lookahead of connecting L_{pi} with L_{pj}. If $t(x)+La_{ij}<t(y)$, then x influences y. If $t(x)+La_{ij}\geq t(y)$, then x can't influence y.

Proof: We can produce the conclusion according to the Definition 4 and lookahead.

Theorem 2: Provided x and y are the first transition of the event tables of L_{pi} and L_{pj}, respectively, La_{ij} is the having definite initial marking lookahead of connecting L_{pi} with L_{pj}, La_{ji} is the having definite initial marking lookahead of connecting L_{pj} with L_{pi}, if $t(x)+La_{ij}\leq t(y)$ and $t(y)+La_{ij}\geq t(x)$, then x and y are concurrent.

Proof: According to the Theorem 1,

$$\text{If } t(x)+La_{ij}\geq t(y), \quad \text{then x can't influence y}$$

$$\text{If } t(y)+La_{ij}\geq t(x), \quad \text{then y can't influence x}$$

According to the Definition 5, x and y are concurrent.

Theorem 3: If a loop structure of a TTPN model is turned into prediction graph according to the item (4a) of prediction graph algorithm, then the loop structure may be livelock or deadlock, meanwhile, the loop structure is turned into livelock free or deadlock free after building up the model prediction graph.

Proof: According to the structure properties of TTPN, a loop structure may be having three statuses:

(1) Self-loop (only including a transition and a place), the loop structure is easy.
(2) A loop which portion transition (for ∃t) timestamp is minor than the $^{\bullet\bullet}$t (preset of preset of the transition) timestamp.
(3) A loop which all transitions (for ∀t) timestamp is minor than the $^{\bullet\bullet}$t (preset of preset of the transition) timestamp.

In the above, (2) loop structure may be occur livelock, (3) loop structure is deadlock. According to the prediction graph algorithm, in the item (4a), a loop structure can be turned into without loop structure based on lookahead definition. So, the loop structure is turned into livelock free or deadlock free after building up the model prediction graph.

According to the concept of lookahead and the Theorem 2, when we simulate the complicated time 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.

THE MAPPING ALGORITHM OF PROCESS

In the study, the objective of process mapping issue is to balance the processor computing spending and

reducing communication spending between processors. Based on the task graph (Lee *et al*., 2008), denotation matrix of task graph is defined. Mapping algorithms (Fang *et al*., 2007) based on the denotation matrix are also proposed. The algorithms can determine the process mapping rapidly. Lots of experimental data analyses show that the algorithms are quite effective. For comparing experiments effect, we assume the communication time is not parallel, computing time is parallel, which is the same as some relative literature.

In the study of Fang *et al*. (2007), a partitioning algorithm is presented based on Lookahead and Petri nets. Though the algorithm can deal with the process partitioning problem, some problem (such as example 1) can not solve very good. According to experiment effect, so we propose an improved algorithm to solve the problem.

For the case of process N is bigger, it is complex for task graph method because that all of the partition methods have to be found and then compute them, respectively and give the best partition by comparing the results. We can denote the task graph by following matrix and then give the best mapping plan by analyzing the matrix.

$$A=\begin{Bmatrix}1\\2\\\vdots\\n\end{Bmatrix}\begin{pmatrix}a_{10} & a_{11} & a_{12} & \cdots & a_{1n}\\ a_{20} & a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & \vdots & \vdots & \vdots\\ a_{n0} & a_{n1} & a_{n2} & \cdots & a_{nn}\end{pmatrix}$$

↑

Corresponding process

The improved mapping algorithm as follows:

(1) Find communication time between every process and executing time of each process.
(2) We write n×(n+1) matrix A, where $a_{i0} = t_i$, $a_{ij} = c_{ij}$ ($1\le i\le n$, $1\le j\le n$), $a_{ij} = 0$, $a_{ij} = a_{ji}.t_i$ is the computing time of the ith process, c_{ij} is the communication time between the ith process and the j-th process.
(3) Select the minimum element from No. zero column in matrix A, using the notation a_{m0}, 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., from a_{mj} ($1\le j\le n$).
(4) If $a_{mk}\ge a_{m0}$, let $a_{k0} = a_{k0}+a_{m0}$, $a_{km} = 0$, $a_{kj} = a_{kj}+a_{mj}$, $a_{mk} = 0$, $a_{jk} = a_{kj}$, ($1\le j\le n$, $j\ne m$). If $a_{mk}<a_{m0}$ and $a_{m0}+a_{k0}-\max\{a_j\}>a_{mk}$, then select the m from the matrix A but do not take the row having minimum computing time into account, until $a_{m0}+a_{k0}-\max\{a_j\}>a_{mk}$.

Table 1: The communication time of interact processes

	Communication time					
Process	1	2	3	4	5	6
1		1	3.0	4.0	1.5	2.0
2	1.0		5.0	3.0	1.0	8.0
3	3.0	5		2.0	0.5	3.0
4	4.0	3	2.0		0.6	7.0
5	1.5	1	0.5	0.6		0.4
6	2.0	8	3.0	7.0	0.4	

(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×(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:

$$\mathrm{Max}\{a_{i0}\}+\frac{1}{2}\sum a_{ij}\ \ (1\le i\le p, 1\le j\le p)$$

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 shown in Table 1 and the data in the table is communication time of two processes.

If we adopt mapping algorithm, it is no avail, improved mapping algorithm can deal with the problem, the application of improved mapping method as follows:

$$A=\begin{Bmatrix}1\\2\\3\\4\\5\\6\end{Bmatrix}\begin{pmatrix}7 & 0 & 1 & 3 & 4 & 1.5 & 2\\ 4 & 1 & 0 & 5 & 3 & 1 & 8\\ 8 & 3 & 5 & 0 & 2 & 0.5 & 3\\ 5 & 4 & 3 & 2 & 0 & 0.6 & 7\\ 3 & 1.5 & 1 & 0.5 & 0.6 & 0 & 0.4\\ 6 & 2 & 8 & 3 & 7 & 0.4 & 0\end{pmatrix}$$

(The operating time is 8+42 = 50)

$$\rightarrow\begin{Bmatrix}1\\3\\4\\5\\6+2\end{Bmatrix}\begin{pmatrix}7 & 0 & 3 & 4 & 1.5 & 3\\ 8 & 3 & 0 & 2 & 0.5 & 8\\ 5 & 4 & 2 & 0 & 0.6 & 10\\ 3 & 1.5 & 0.5 & 0.6 & 0 & 1.4\\ 10 & 3 & 8 & 10 & 1.4 & 0\end{pmatrix}$$

(The operating time is 10+34 = 44)

$$\rightarrow\begin{Bmatrix}1\\3\\5\\6+2+4\end{Bmatrix}\begin{pmatrix}7 & 0 & 3 & 1.5 & 7\\ 8 & 3 & 0 & 0.5 & 10\\ 3 & 1.5 & 0.5 & 0 & 2\\ 15 & 7 & 10 & 2 & 0\end{pmatrix}$$

(The operating time is 15+24 = 39)

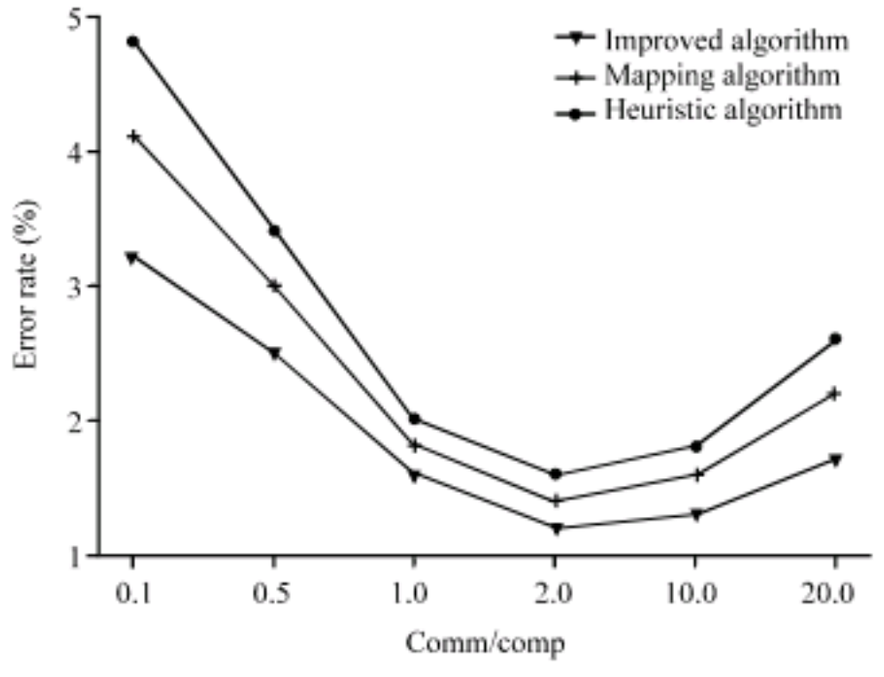

Fig. 4: The three methods error rate in different comm/comp value

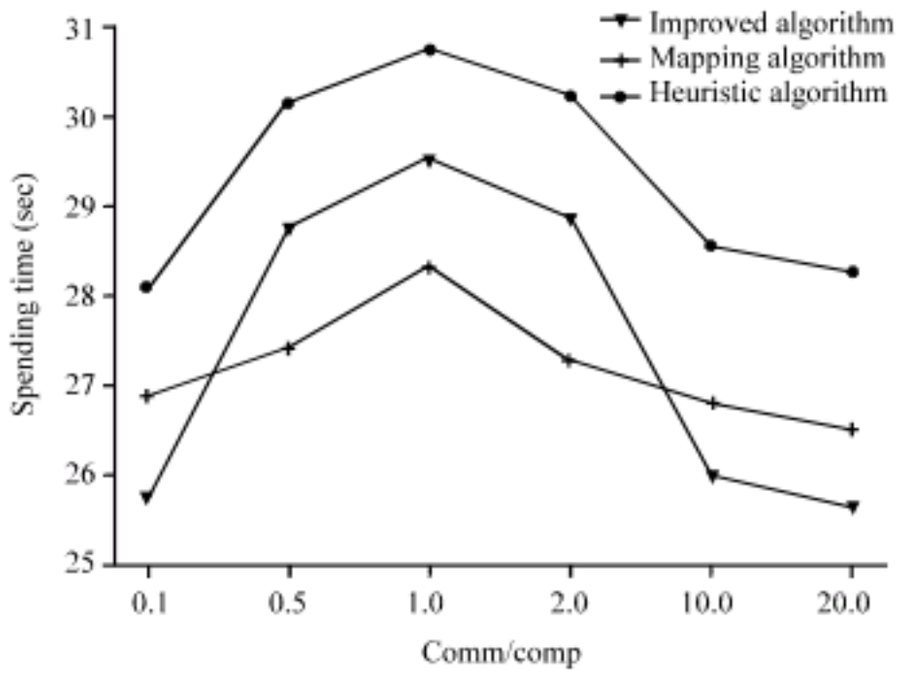

Fig. 5: The three methods spending time in different comm/comp value

$$\rightarrow \begin{Bmatrix} 3 \\ 5 \\ 6+2+4+1 \end{Bmatrix} \begin{pmatrix} 8 & 0 & 0.5 & 13 \\ 3 & 0.5 & 0 & 3.5 \\ 22 & 13 & 3.5 & 0 \end{pmatrix}$$

(The operating time is 22+17 = 39)

So, optimal partition is {{3}, {5}, {1,2, 4, 6}} and the operating time is 39.

For validating mapping algorithm, we analyze 100 processes mapping on 8 processors in computer which CPU is T2030, memory is 1G 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 compare the three methods (improved algorithm, mapping algorithm (Fang *et al*., 2007), heuristic algorithm (Juhasz and Turner, 2000) error rate as shown in Fig. 4 by giving different comm/comp. It can be seen that the improved algorithm is best than the others, the basic mapping algorithm is better than the heuristic algorithm.

In the Fig. 5, we obtain the spending time (operating time) using three methods respectively. The effect shows that the spending time is most using the heuristic algorithm. When comm/comp is between 0.5 with 10, the improved algorithm is better, the others, the basic mapping algorithm is better the improved algorithm.

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 because of blindly partitioning the net and let the concurrent logical process operate in the same node of the parallel machine, thus improve simulation performance. In this study, we propose partition methods of distributed system based on lookahead and obtain an improved partitioning algorithm, this will promote the performance of the parallel simulation. Lots of experiments show the improved algorithm is effective.

In the future, we plan to study the lookahead computing based on GA or DPSO and propose some better distributed partition algorithm.

ACKNOWLEDGMENTS

We would like to thank the support of the National Natural Science Foundation of China under Grant No. 60873144 and No. 90818023, the excellent young talents project of Anhui Province (2009SQRS045) and the humanity and society science project of Anhui Province (2009SK156).

REFERENCES

Bassel, A., K. Day and A. Touzene, 2008. A multilevel partitioning approach for efficient tasks allocation in heterogeneous distributed systems. J. Syst. Architecture, 54: 530-548.

Fang, X.W., Z.C. Xu and Z.X. Yin, 2007. Distributed processing based on timed petri nets. Proceeding of the 3nd International Conference on Natural Computation, Aug. 14-17, Haikou, pp: 287-291.

Janneck, J.W., 1998. Generalizing lookahead-behavioral prediction in distributed simulation. ACM SIGSIM Simulation Digest, 28: 12-19.

Jin, S.Y., G. Schiavone and D. Turgut, 2008. A performance study of multiprocessor task scheduling algorithms. J. Supercomputing, 43: 77-97.

Joseph, A., 2002. A concurrent processing framework for the set partitioning problem. Comput. Operat. Res., 29: 1375-1391.

Juhasz, Z. and S.J. Turner, 2000. A new heuristic for the process-processor mapping problem. Proceedings of the 3rd Austrian–Hungarian Workshop on Distributed and Parallel Systems, May 12-14, Kluwer Academic Publishers Norwell, MA, USA., pp: 91-94.

Lee, S., X.B. Zhao and A. Shendarkar, 2008. Fully dynamic epoch time synchronization method for distributed supply chain simulation. Int. J. Comput. Appl. Technol., 31: 249-262.

Murata, T., 1989. Petri nets-properties analysis and applications. Proc. IEEE, 77: 541-580.

Nketsa, A. and N.B. Khalifa, 2001. Timed Petri nets and prediction to improve the Chandy-Misra conservative-distributed simulation. Applied Mathematics Comput., 12: 235-254.