Fuzzy-Timing Petri Nets with Choice Probabilities for Response Time Analysis

Jaegeol Yim and Kye-Young Lee

Dept. of Computer and Multimedia Dongguk University at GyeongJu, GyeongBuk, 780-714
Republic of Korea
{yim, lky}@dongguk.ac.kr

Abstract. The system response time is one of the most critical user's requirements. Therefore, a response time analysis must be done at the design stage. This paper proposes a response time analysis method based on 'Fuzzy-Timing Petri net with choice probabilities (FTCP)'. FTCP is a modified version of timed net: A delay time associated with a transition is not a crisp number but a possibility distribution function and a transition of FTCP is also associated with a choice probability. We are proposing a response time analysis procedure for FTCP. It is similar to the minimum cycle time analysis for Timed Nets. The difference is that we take account of choice probabilities of the transitions in the process of generating the T-invariant. The other feature of our analysis method is that it can handle temporal uncertainties. This paper finally demonstrates the usefulness of the method by applying the method on a Web Service System.

1 Introduction

Petri net has been widely used in the field of computer system analysis. Modeling and performance analysis of message-oriented middleware using generalized stochastic Petri net is introduced in [1]. Petri net has also been used in steel making process [2]. Murata introduced Fuzzy-Timing High-Level Petri Nets for time-critical applications [3]. Integrating Fuzzy Timing Petri net and Time Petri net, Zhou et al made up a new temporal Petri net. Using the new temporal Petri net, they built a model of distributed multimedia synchronization and analysis methods [4]. The new temporal Petri net has also been used in modeling and performance analysis of networked virtual environments [5].

A Timed Petri net is a modified Petri net where transitions are associated with delay times. The minimum cycle time analysis method of Timed Petri net has been introduced in [6]. The method has been applied to a Petri net model of a multi-stage production system for performance evaluation of the system. Dasdan and Gupta introduced a more efficient algorithm for minimum cycle time analysis [7]. An algorithm to find the minimum cycle time of a marked graph, a subclass of Petri nets, is introduced in [8]. Given both a timed net and a limit of cycle time, finding the minimum initial marking for which the minimum cycle time is less than or equal to the limit is called the minimum initial marking problem. An answer for this problem is given in [9].

One of the advantages of Fuzzy Timing Petri net is that it can handle uncertain time whereas one of its disadvantages is the lack of efficient analysis methods. Although reduction methods and simulation methods are introduced in [5] as analysis methods for Fuzzy Timing Petri net, they are essentially time consuming exhaustive methods.

One of the advantages of minimum cycle time method is its efficiency. Once we find S-invariants and T-invariants, we can perform minimum cycle time analysis in a sequence of matrix multiplications and divisions. With the result of minimum cycle time we can find the bottleneck of the system. The weak points of minimum cycle time method include the following two points: 1) It can handle only constant times 2) it does not consider the choice probabilities of events. In the real world, events are associated with not only delay time but also probabilities of choices. For example, considering a read event and a write event in a database, not only delay time for read is different from delay time for write, but also the choice probability of read event is different from the choice probability of write event. However, the existing Timed net only considers delay time.

This paper proposes ‘Fuzzy-Timing Petri Nets with Choice Probabilities (FTCP)’ where transitions are associated with not only delay time but also probability of choice. The delay time associated with a transition can be a fuzzy time function, or possibility distribution. As an analysis method for FTCP, we propose a modified version of minimum cycle time method. With the method, we can find the response time of the underlying computer system.

2 Related Works

We are assuming that readers are familiar with Petri net terminologies, such as transition, place, arc, weight, initial marking, T-invariant, S-invariant, support, and so on. For introduction to Petri net theory and its applications, please refer to one of the references [10-13]. Among the related works, Fuzzy-Timing Petri nets [3] and minimum cycle time method [10] are the most closely related to this paper.

An example of Fuzzy-Timing High-Level Petri net (FTHN) is shown in Fig. 1 [3]. The FTHN represents a job shop. A job arrival is represented as a token at place P_{in} . Two tokens ‘a’ and ‘b’ in place P_{in} implies that two jobs named ‘a’ and ‘b’ have been arrived for process. There is a machine which can be in either ready or busy state. A job arrived in P_{in} leaves the job shop via place P_{out} after being processed by the machine. Place P_{busy} is corresponding to the state in which the machine is busy operating

on a job. A job in P_{in} can enter P_{busy} if there is a machine in P_{ready} .

A token in the FTHN is associated with a timestamp. A timestamp is a possibility distribution function. As examples of timestamps, let

Fig. 1. A FTHN model of a job shop

$\pi_{0a}(\tau) = (1 \ 3 \ 3 \ 5)$, $\pi_{0b}(\tau) = (3 \ 5 \ 5 \ 7)$ and $\pi_0(\tau) = (0 \ 0 \ 0 \ 0)$ be the timestamps associated with tokens 'a' and 'b' at place P_{in} , and the token at place P_{ready} , respectively. An arc from a transition t to place p , (t, p) is associated with a delay time. Let the arcs (t_1, P_{busy}) , (t_2, P_{ready}) and (t_2, P_{out}) be associated with $d_1(\tau) = (0,0,0,0)$ and $d_2(\tau) = d_3(\tau) = (4,5,7,9)$, respectively, in Fig. 1. Reference [3] introduces an analysis method with which we can figure out the possibility for the jobs can be finished within certain given times. However, the method is essentially enumerating all the possible firing sequences and too time consuming.

In reference [10], a cycle time is defined as the time to complete a firing sequence from the initial marking back to the same marking after firing each transition at least once on a Timed Net. A mathematical cycle time analysis method is also introduced in [10] as follows, where, y_k is an S-invariant, $A^- = [a_{ij}^-]_{n \times m}$ and $a_{ij}^- = w(p_j, t_i)$, D is the diagonal matrix of delay d_i , $i=1, 2, \dots, n$, and M_0 is the initial marking.

$$\tau_{\min} = \text{Max}_k \{ y_k^T (A^-)^T D x / y_k^T M_0 \} \quad (1)$$

However, a Timed Net is not equipped with any facility of specifying the choice probabilities for transitions in conflict.

3 Fuzzy-Timing Petri Nets with Choice Probabilities

The main objective of this paper is to propose the Fuzzy-Timing Petri Net with Choice Probabilities (FTCP). We are introducing an example of it, then a formal definition of it, followed by analysis method in this section. An example FTCP is shown in Fig. 2. It is FTCP model of a readers-writers system. In the real world, the frequency of read operation is greater than the frequency of write operation. However, the frequencies are not considered in the timed net. The frequency of read operation, C_1 , is labeled on t_1 and the frequency of write operation, C_2 , is on t_2 in Fig. 2. The labels for delay times (d_i s) on Fig. 2 are possibility distribution functions.

For the Petri net in Fig. 2, there are two minimal support T-invariants:

Fig. 2. An example Fuzzy-Timing Petri net with Choice Probabilities

$$x_1 = (1 \ 0 \ 1 \ 0)^T \text{ and } x_2 = (0 \ 1 \ 0 \ 1)^T.$$

Let the choice probabilities of C_1 and C_2 be 0.9 and 0.1, respectively. Considering the choice probabilities, we can find a T-invariant $x = (9 \ 1 \ 9 \ 1)$. Furthermore, if we want to find the time required for the k processes to perform either read or write operation once each, we should use $x = (0.9k \ 0.1k \ 0.9k \ 0.1k)$.

If we use $x = (0.9k \ 0.1k \ 0.9k \ 0.1k)$, then we obtain

$$(A^-)^T D x = \begin{bmatrix} 0.9kd_1 + 0.1kd_2 & 0.9kd_3 & 0.9kd_1 + 0.1k^2d_2 & 0.1kd_4 \end{bmatrix}^T$$

Thus, application of (1) yields the following minimum cycle time:

$$\begin{aligned}\tau_{\min} &= \text{Max}\{(0.9kd_1 + 0.1kd_2 + 0.9kd_3 + 0.1kd_4)/k, [0.1k^2(d_2 + d_4) + 0.9k(d_1 + d_3)]/k\} \\ &= 0.1k(d_2 + d_4) + 0.9(d_1 + d_3).\end{aligned}$$

An interpretation of the minimum cycle time is that 0.1 k out of k processes will perform 'write operation' spending $(d_2 + d_4)$ time each, i.e. 0.1 k processes will spend $0.1k * (d_2 + d_4)$ time units. On the other hand, 0.9 k processes will perform 'read operation'. Since read operation can be done in parallel, it takes only $(d_1 + d_3)$ time units for all k processes to perform read-operation. The minimum cycle time counts only portion of $(d_1 + d_3)$, i.e. $0.9(d_1 + d_3)$ time units because only 0.9 k processes perform 'read-operation'. In total, the readers-writers system with k processes needs $0.1k(d_2 + d_4) + 0.9(d_1 + d_3)$ time units in order for k processes to perform either read or write operation once each. It is worth of noting that the time needed for k processes to perform either read or write operation once each is the system response time if k is the average number of processes using the DB concurrently.

As a numerical example, let $d_1 = d_2 = (0, 0.5, 0.5, 1.5)$, $d_3 = (1, 2, 2, 4)$, $d_4 = (3, 4, 5, 7)$, and $k = 10$. Then the response time is $(0, 0.5, 0.5, 1.5) \oplus (3, 4, 5, 7) \oplus 0.9((0, 0.5, 0.5, 1.5) \oplus (1, 2, 2, 4)) = (3, 4.5, 5.5, 8.5) \oplus (0.9, 2.25, 2.25, 4.95) = (3.9, 6.75, 7.75, 13.45)$. From the result, we can conclude that the system meets the following fuzzy statements: The system mostly responses within 7.8 time units.

For the formal definition of FTCP, we are defining a few terminologies.

[Def. 1] If transitions t_a and t_b are sharing an input place, they are said to be in choice relation, and denoted by ' t_a choice t_b ' or $\text{choice}(t_a, t_b)$.

[Def. 2] If $\text{choice}(t_a, t_b)$, then these transitions should be associated with probabilities of choice. The probability of choice for t_a is denoted by $\text{Prob}(t_a)$.

For the simplicity we denote the output/input vertices of a vertex v $v \bullet / \bullet v$. For example, $p_1 \bullet = \{t_1, t_2\}$ in the case of Fig. 1. Sum of all the choice probabilities of transitions sharing one input place should be 1, i.e. the following holds:

$$\sum_{t_j \in p_i \bullet} \text{Prob}(t_j) = 1$$

A definition of FTCP is given in Table 1.

Since a transition can be associated with a choice probability, a T-invariant can also be associated with a choice probability. In the definition of the choice probability of a T-invariant, we are using the terminology, group, defined in [Def. 3].

Table 1. A definition of FTCP

A FTCP is 7-tuple, $\text{FTCP} = (P, T, F, W, M_0, DT, \text{Prob},)$, where:
 (P, T, F, W, M_0) is a Petri net,
 $DT: T \rightarrow$ fuzzy timestamps, is a function determining delay time of a transition,
 $\text{Prob}: T \rightarrow (0, 1)$ is a function determining choice probability, where, $\sum_{t_j \in p_i \bullet} \text{Prob}(t_j) = 1$.

[Def. 3] Given a T-invariant x , a group regarding $\text{choice}(t_1, t_2, \dots, t_k)$ is the set of transitions which are in the choice relation and are also members of the support of x .

[Def. 4] Given a group $g_i = \{t_1, t_2, \dots, t_k\}$, the choice probability of g_i is defined by $\text{Prob}(g_i) = \text{Prob}(t_1) + \text{Prob}(t_2) + \dots + \text{Prob}(t_k)$.

[Def. 5] Given a T-invariant x , the choice probability of x , $\text{Prob}(x)$, is defined by the probability obtained by performing the following steps:

Step 1: We find all groups considering all choices.

Step 2: For all the groups found in Step 1, we find choice probabilities of them.

Step 3: We multiply all the choice probabilities of the groups to find the choice probability of x , i.e. the choice probability of x is defined by:

$$\text{Prob}(x) = \prod_{g_i \in \|x\|} \text{Prob}(g_i).$$

The response time is the time needed to finish a certain process. A process consists of many steps including the start step. The place corresponding to the start step is called Start Place. If we designate a different place as Start Place then response time is also changed. Therefore, we can conclude that we have to designate the Start Place properly considering the semantics of the FTCP. We define a FTCP which is consistent and has a Start Place a well-formed FTCP.

Given a well-formed FTCP, the algorithm shown in Table 2 finds the response time of the computer system modeled as the FTCP.

Table 2. Algorithm for response time analysis of FTCP

Input:

N: A well-formed FTCP,

n: the number of tokens at Start Place.

Step 1: We find all minimal support T-invariants of N.

Step 2: We perform the following steps to form maximal minimal T-invSPs

2.1 Construct a sub-FTCP for each minimal support T-invariant

2.2. While there exists a sub-FTCP_i whose covering T-invariant is not a T-invSP

2.2.1. Merge sub-FTCP_i to its adjacent sub-FTCP.

2.3. For each sub-FTCP, we obtain its covering T-invariant (* All of them are maximal minimal T-invSP *)

Step 3: For each T-invariant x_i obtained at Step 2.3, we obtain its choice probability prob_i .

Step 4: For each T-invariant x_i obtained at Step 2.3, we obtain $\text{prob}_i * n * x_i$.

Step 5: We obtain a positive T-invariant x by summing up all $\text{prob}_i * n * x_i$.

Step 6: Apply equation (1) $\tau_{\min} = M a x \{ y_k^T (A^-)^T D x / y_k^T M_0 \}.$

[Theorem] Sum of all the choice probabilities of all the T-invariants obtained at Step 2 is 1.

<Proof> We prove this theorem with mathematical induction.

Base Step: Since the given FTCP is well-formed, it has at least one T-invariant. If it has exactly one T-invariant, then for all $p_i \in P, |p_i| = 1$. $|p_i| = 1$ implies that for all $t_i \in T$, $\text{Prob}(t_i) = 1$. Therefore, the choice probability of x is $1 * 1 * \dots * 1 = 1$.

Induction Hypothesis: If there are k T-invSPs, then the sum of choice probabilities of them is 1.

Induction Step: We have to prove that if one more T-invSP is added to existing k T-invSPs then the sum of choice probabilities is still 1. Let the k existing T-invariants be $x_1, x_2, \dots, x_k$ and $\text{Prob}(x_1) + \text{Prob}(x_2) + \dots + \text{Prob}(x_k) = 1$. According to Theorem 1, addition of $k+1^{\text{th}}$ T-invSP requires one addition of choice relation. Let $\text{choice}_i(t_i, t_j)$ be

the additional choice relation. Then, either t_i or t_j must be a member of the support of one of the existing k T-invariants and the other one must be a member of the support of $k+1^{\text{th}}$ T-invSP. Let $t_i \in \llbracket x_h \rrbracket$ and $t_j \in \llbracket x_{k+1} \rrbracket$. Adding choice $_{(t_i, t_j)}$ changes the $\text{prob}(t_i)$. Therefore, we name the x_h with new choice probability of t_i new x_h .

Let $\{t_{i1}, t_{i2}, \dots, t_i, t_{i+1}, \dots, t_{iq}\} = \llbracket \text{new}x_h \rrbracket$, $\{t_{j1}, t_{j2}, \dots, t_j, t_{j+1}, \dots, t_{jr}\} = \llbracket x_{k+1} \rrbracket$, and the transitions are listed in the order of distance from Start Place. Then, the following must be true. Otherwise, the number of T-invSP is increased by more than one.

$t_{i1} = t_{j1}, t_{i2} = t_{j2}, \dots, t_{i-1} = t_{j-1}$ and $\text{prob}(t_{j+1})=1, \dots, \text{prob}(t_{iq})=1, \text{prob}(t_{j+1})=1, \dots, \text{prob}(t_{jr})=1$.

Therefore, $\text{Prob}(\text{new}x_h) = \text{Prob}(x_h) * \text{prob}(t_i)$, and $\text{Prob}(x_{k+1}) = \text{Prob}(x_h) * \text{prob}(t_j)$. Now, $\text{Prob}(\text{new}x_h) + \text{Prob}(x_{k+1}) = \text{Prob}(x_h) * \text{prob}(t_i) + \text{Prob}(x_h) * \text{prob}(t_j) = \text{Prob}(x_h) * \{\text{prob}(t_i) + \text{prob}(t_j)\}$. But, $\text{prob}(t_i) + \text{prob}(t_j) = 1$. Therefore, $\text{Prob}(x_h) * \{\text{prob}(t_i) + \text{prob}(t_j)\} = \text{Prob}(x_h) * 1 = \text{Prob}(x_h)$.

4 Application of FTCP on Web Service System

An FTCP model of a web service system is shown in Fig. 3.

At Step 1 of the algorithm shown in Table 2, we find the following minimal support T-invariants:

$$x_1 = (1 \ 1 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0)^T,$$

$$x_2 = (1 \ 1 \ 0 \ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 2)^T,$$

$$x_3 = (1 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 2)^T.$$

At Step 2, we find that the minimal support T-invariants are also maximal minimal T-invSPs.

At Step 3, we find the probability of choice for $x_1 = 1 * 1 * 0.8 = 0.8$, for $x_2 = 1 * 1 * 0.2 * 0.7 * 1 \dots = 0.14$, and for $x_3 = 0.2 * 0.3 = 0.06$.

As the result of Step 4 and 5, considering the probabilities of choice and the number of users, n , we obtain the following (2) positive T-invariant. A sequence of firing t_2 n times, t_1 n times, t_3 $0.8n$ times, and so on is possible and returns back to the initial marking.

$$x = (n \ n \ 0.8n \ 0.2n \ 0.06n \ 0.14n \ 0.06n \ 0.2n \ 0.2n \ 0.4n)^T \quad (2)$$

Step 6 is the existing minimum cycle time method, and we can obtain the response time as follows:

$$\text{Max} \{nd_1/n, 0.2n(d_2+d_5)/r, 0.06nd_3/m, 0.2nd_4/k, (nd_1+0.2nd_2+0.06nd_3+0.2nd_4+0.2nd_5)/n\} \quad (3)$$

For a numerical example, consider the following possibility distribution functions and numbers of resources:

$d_1 = (59,900, 60,000, 60,000, 60,100)$, $d_2 = (90, 100, 100, 110)$, $d_3 = (490, 500, 500, 530)$, $d_4 = (990, 1,000, 1,000, 1200)$, $d_5 = (99, 100, 100, 120)$, $n = 1,000$, $k = 1$, $m=1$, $r=1$.

The response time is:

$\text{Max}\{(59,900, 60,000, 60,000, 60,100), (37,800, 40,000, 40,000, 46,000), (29,400, 30,000, 30,000, 31,800), (198,000, 200,000, 200,000, 240,000), (60,165.2, 60,270, 60,270, 60,417.8)\} = (198,000, 200,000, 200,000, 240,000)$.

From the response time, we can conclude that the following statements hold:

- 1) The system mostly responses within 220,000 time units.
- 2) The system definitely responses within 240,000 time units.
- 3) The system unlikely responses within 200,000 time units.
- 4) The system never responses within 198,000 time units.

The result implies that the system's response time is dominated by the content provider. That is, a user of the system will waste time for waiting response from the LBS system. In order to solve this problem, the designer has to do one of the following: 1) reduce subscribers, 2) speed up the content provider 3) increase the number of processors of the content provider.

As another numerical example, let us change the number of subscribers to 50 and the other factors leave the same as the above example. Then the minimum cycle time will be determined by d1, the time interval needed by a user to choose a menu. In this case, the system's performance is better than necessary and we can do one of the followings: 1) reduce implementation cost by using less expensive resources, or 2) serve more subscribers.

5 Conclusions

In the real world, the duration time from the start of an event to the end of the event is not a crisp number but a fuzzy number. The events are usually in conflict in that the occurrence of event A prevents event B from occurrence. However, the existing minimum cycle time method which deals with timed nets considers neither temporal fuzziness nor choice probabilities of events. On the other hand, Fuzzy-timing Petri net can handle temporal uncertainty but the analysis of the net is time consuming.

We proposed Fuzzy-Timing Petri net with Choice Probability (FTCP) in this paper. FTCP is a new version of timed net in which a transition can be associated with both possibility distribution function and choice probability. Therefore, FTCP has both Fuzzy-Timing Petri net's advantage and Timed net's advantage, namely the response time analysis method of FTCP is as efficient as the minimum cycle time method and the result of response time analysis is not a crisp number but a possibility distribution function.

We have built a FTCP model of a Web Service System and analyzed it with our response time analysis algorithm. The analysis result showed that FTCP indeed possesses both the advantages of Fuzzy-Timing Petri net and of Timed Net.

References

- [1] S. Fernandes, W. Silva, M. Silva, N. Rosa, P. Maciel, and D. Sadok, "On the generalised stochastic Petri net modeling of message-oriented middleware systems," 2004 IEEE International Conference on Performance, Computing, and Communications, Cidade Univ., Recife, Brazil, 2004, pp. 783–788
- [2] H. Gao, J. Zeng, G. Sun, Y. Wu, "New matrix method for analyzing steel making process," 2002 International Conference on Machine Learning and Cybernetics, Vol. 2, Nov. 4-5, 2002, pp. 1070-1074.
- [3] T. Murata, "Temporal Uncertainty and Fuzzy-Timing High-Level Petri Nets," Application and Theory of Petri Nets 1996, pp.11-28.
- [4] Y. Zhou and T. Murata, "Modeling and Analysis of Distributed Multimedia Synchronization by Extended Fuzzy-Timing Petri Nets," Transactions of the SDPS (Society for Design and Process Science), Vol. 5, No. 4, Dec. 2001, pp. 23-37.
- [5] Y. Zhou, T. Murata, and T. DeFanti: Modeling and Performance Analysis Using Extended Fuzzy-Timing Petri Nets for Networked Virtual Environments. IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, Vol. 30, No. 5, October 2000, pp.737-756.
- [6] Hervé P. Hillion, "Timed Petri Nets and Application to Multi-Stage Production Systems.", Lecture Notes in Computer Science, Vol. 424; Advances in Petri Nets 1989, Berlin, Germany, Springer-Verlag, 1990, pp. 281-305.
- [7] A. Dasdan, R.K. Gupta, "Faster maximum and minimum mean cycle algorithms for system performance analysis," IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol. 17, No. 10, 1998, pp. 889-899.
- [8] M. Nakamura, M. Silva, "Cycle time computation in deterministically timed weighted marked graphs," Proceedings of the 7th IEEE International Conference on Emerging Technologies and Factory Automation, Vol. 2, Oct. 18-21, 1999, pp. 1037-1046.
- [9] J. Rodriguez-Beltran, A. Ramfrez-Trevino, "Minimum initial marking in timed marked graphs," 2000 IEEE International Conference on Systems, Man, and Cybernetics, Vol. 4, Oct. 8-11, 2000, pp. 3004-3008.
- [10] T. Murata, "Petri nets: Properties, analysis and applications," Proceedings of the IEEE, Vol. 77. no. 4, April 1989, pp. 541-580
- [11] J. L. Peterson, Petri Net Theory and the Modeling of Systems, Prentice-Hall, N.J., 1981, ISBN: 0-13-661983-5.
- [12] W. Reisig, Petri Nets, An Introduction, EATCS, Monographs on Theoretical Computer Science, W.Brauer, G. Rozenberg, A. Salomaa (Eds.), Springer Verlag, Berlin, 1985.
- [13] Jörg Desel, Wolfgang Reisig, Grzegorz Rozenberg (Eds.), Lectures on Concurrency and Petri Nets, Advances in Petri Nets, Lecture Notes in Computer Science, vol. 3098, Springer-Verlag, 2004, ISBN: 3-540-22261-8.