Bounded model checking of Time Petri Nets using SAT solver

Tomoyuki Yokogawa^{1a)}, Masafumi Kondo², Hisashi Miyazaki²,
Sousuke Amasaki¹, Yoichiro Sato¹, and Kazutami Arimoto¹

¹ Okayama Prefectural University, 111 Kuboki, Soja, Okayama 719–1197, Japan

² Kawasaki University of Medical Welfare,

288 Matsushima, Kurashiki, Okayama 701–0193, Japan

a) t-yokoga@cse.oka-pu.ac.jp

Abstract: To carry out performance evaluation of an asynchronous system, the system is modeled as Time Petri Net (TPN) and an iteration of Petri net simulations produces its performance index. The TPN model needs to satisfy required properties such as deadlock freeness. We proposed a symbolic representation of TPN for SAT-based bounded model checking. In the proposed encoding scheme, firing of transitions and elapsing of place delays are expressed as boolean formulas discretely. Our representation can work with relaxed $\exists$ -step semantics which enables to perform each step by two or more transitions. We applied the encoding to example TPN models and checked the deadlock freeness using SAT solver. The results of experiments demonstrated the effectiveness of the proposed representation.

Keywords: symbolic model checking, bounded model checking, Time Petri Net, SAT solver

Classification: Integrated circuits

References

- [1] P. M. Merlin and D. J. Faber: IEEE Trans. Softw. Eng. **24** [9] (1976) 1936.
- [2] A. Xie and P. A. Beerel: Hardware Design and Petri Nets (1999) 239.
- [3] A. Biere, A. Cimatti, E. M. Clarke and Y. Zhu: TACAS '99 (1999) 193.
- [4] S. Ogata, T. Tsuchiya and T. Kikuno: ATVA '04 **3299** (2004) 79.
- [5] M. Sorea: Electron. Notes Theor. Comput. Sci. **68** [5] (2003) 116. DOI:10.1016/S1571-0661(04)80523-1
- [6] J. Dubrovin, T. Junttila and K. Heljanko: Sci. Comput. Program. **77** (2012) 1095. DOI:10.1016/j.scico.2011.07.005
- [7] J. Sifakis: Net Theory and Applications **84** (1980) 307. DOI:10.1007/3-540-10001-6_30
- [8] B. Duterte and L. De Moura: CAV '06 **4144** (2006) 81.
- [9] UPPAAL: <http://www.uppaal.com/>.
- [10] Y. Miwa, T. Yokogawa, H. Miyazaki, M. Kondo and Y. Sato: FIT2011 (2011) (in Japanese).

1 Introduction

Time Petri Nets (TPN) [1] is a major system modeling method in performance evaluation of a large-scale asynchronous system. The evaluation assigns stochastic distributions to time restrictions on TPN, iterates a Petri net simulation on the model, and then produces its performance index [2]. It is necessary that the TPN model satisfies required properties such as deadlock freeness or absence of undesirable behaviors to carry out an iteration of petri net simulations for performance evaluation. The existence of deadlock or the reachability to undesirable states indicates incorrect modeling or an error in the original asynchronous system. While model checking can verify those properties on TPN in theory, the increase of TPN size brings state explosion and makes hard the verification in practice.

This paper proposed a symbolic representation of TPN for SAT-based model checking [3]. SAT-based symbolic model checking can substantially reduce the verification cost on Petri Nets (PN) [4]. The proposed symbolic representation encodes firing of transitions and elapsing place delays to boolean formulas discretely in a similar manner of encoding of timed systems in [5]. This representation can work with *relaxed $\exists$ -step semantics* [4, 6], which enables to perform each step by two or more transitions. An experiment demonstrated the effectiveness of the proposed representation.

The main contributions of this paper are:

1. symbolic representation of TPN for bounded model checking, and
2. empirical demonstration of the effectiveness of the representation on verifying large-scale asynchronous circuits.

2 Time Petri Nets

PN is defined as 5-tuple $PN = (P, T, F, F_{in}, M_0)$, where P is a set of *places*, T is a set of *transitions*, $F \subseteq (P \times T) \cup (T \times P)$ is a set of *arcs*, $F_{in} \subset F$ is a set of *inhibitor arcs* and $M_0 \subseteq P$ is the initial marking. A directed arrow connects a place and a transition. A place can have *tokens*, and a transition fires when all of its input places have a token, and the tokens move to its output places through the transition. We focus on a *safe* PN where each place has at most one token, and arc weight is not considered. The initial marking represents the places that have a token at the initial state. An inhibitor arc, which is described as a circle-headed arc, represents a condition of inhibiting a transition firing. The transitions which have input places which connected with an inhibitor arc can fire only if those places do not have tokens. Input and output places of a transition $t \in T$ are described as $\bullet t$ and $t\bullet$, and those of a place $p \in P$ are described as $\bullet p$ and $p\bullet$, respectively. The places with an inhibitor arc to transition t are represented as $\circ t$.

We introduce P-Time Petri Nets [7] which is a subclass of TPN and extends PN so that it can represent *place delay*. P-TPN is defined as 6-tuple $P\text{-TPN} = (P, T, F, F_{in}, M_0, X)$ and $X : P \rightarrow (\mathbb{Z}^+) \times (\mathbb{Z}^+ \cup +\infty)$ is a function mapping the place delays to places ($\mathbb{Z}^+$ denotes a set of integers which are greater than or equal to zero). The place delay represents the time required to enable a token of the place and is described as a form of (l_i, u_i) , where l_i and u_i denote lower and upper bound of place delays on a place $p_i \in P$, respectively, and satisfy $l_i < u_i$. A token in place

p_i can be enabled after l_i elapsed and must be enabled until u_i elapsed. A transition fires when all of its input places have an *enabled* token.

We consider a firing of each transition and time elapsing is carried out discretely, that is, the state of TPN changes by either time elapsing or firing of one transition. Since a state transition of TPN is generally performed by firing of multiple transitions, this assumption may introduce unexpected states in original behaviors of TPN. However, it never eliminates reachable states in original behaviors and there is no problem to carry out reachability analysis of TPN using bounded model checking.

Fig. 1 shows an example of a TPN. The initial marking of the TPN is $\{p_1\}$. Since the place delay $[0, 0]$ is assigned to p_1 , its token is enabled immediately and then t_1 fires. By firing of t_1 , p_2 and p_3 get tokens. When the token of p_2 is enabled earlier than p_3 , p_4 gets a token, and it is enabled immediately. Since t_3 is inhibited by p_4 , t_3 can not be fired even if the token of p_3 is enabled. p_5 can not get a token until the token of p_4 is removed by firing of t_4 . While t_4 requires that p_5 gets an enabled token. Thus this TPN falls into a deadlock.

Fig. 1. An example of a TPN.

3 Bounded model checking of TPN

3.1 Bounded model checking

A state s of TPN which has l places ($l = |P|$) is defined by two l -vectors $\mathbf{m} = (m_1, \dots, m_l)$ and $\mathbf{x} = (x_1, \dots, x_l)$ for the vector of places $\mathbf{p} = (p_1, \dots, p_l)$ where $p_i \in P$. m_i is a boolean variable which evaluates to true when p_i has a token and x_i represents the elapsed time of the token in p_i . $s \xrightarrow{t} s'$ denotes that a state s changes to s' by firing of a transition t , and $s \xrightarrow{x} s'$ denotes that a state s changes to s' by elapsing of time interval x . A boolean function over variables of s which holds iff a state s belongs to a state set S is called a *characteristic function* of S . Similarly, a transition relation can be specified as a characteristic function over s and s' which holds iff s can change to s' .

To carry out SAT-based bounded model checking, it is necessary to encode two characteristic function $\mathcal{N}_k$ and $\mathcal{R}_k$. $\mathcal{N}_k(s_0, \dots, s_k)$ denotes that the initial state s_0 can reach s_k by k -steps through $s_1, s_2, \dots, s_{k-1}$. $\mathcal{R}_k(s_0, \dots, s_k)$ denotes that the system satisfies a desirable property to be checked in any one of the states $s_0, \dots, s_k$. Here, a *step* represents the changing of states caused by time elapsing and firing of a transition. That is, we define a step as a changing of states s and s' , where $s \xrightarrow{x} s'$

and $s'' \xrightarrow{t} s'$ for some t, x and s'' . Bounded model checking can check whether the system can satisfy the desirable property within k -steps by determining the satisfiability of $\mathcal{N}_k \wedge \mathcal{R}_k$ using SAT-solver. When $\mathcal{N}_k \wedge \mathcal{R}_k$ is satisfiable, the property can be satisfied in k -steps from the initial state. Then we can obtain the fact that the system satisfies the reachability of the states where the property holds and the verification can be carried out successfully. On the other hand, if $\mathcal{N}_k \wedge \mathcal{R}_k$ can not be satisfied, it can be shown that the system does not satisfy the property at least within k -steps.

3.2 Symbolic representation

3.2.1 Symbolic representation of steps

In our encoding, 1-step of TPN is performed by *delay step* and *firing step* discretely. When $C(s, d, s')$ and $F(s, s')$ denote characteristic functions of a delay step and a firing step, a characteristic function of 1-step $T(s, d, s')$ can be defined as follows,

$$T(s, d, s') \stackrel{\text{def}}{=} C(s, d, s'') \wedge F(s'', s')$$

where d denotes elapsing place delay and s'' is an intermediate state between delay and firing steps. In order to represent place delays, we extend the characteristic function of k -steps as $\mathcal{N}_k(s_0, \dots, s_k, d_1, \dots, d_k)$ which denotes that s_k can be reached from the initial states by k -steps and each place delay between s_{i-1} and s_i is d_i .

A transition t can fire when all $p \in \bullet t$ have enabled token and all $p \in \circ t$ do not have enabled token. Thus characteristic function $En_t(s)$ which denotes t can fire in a state s can be defined as follows,

$$En_t(s) \stackrel{\text{def}}{=} \bigwedge_{p_i \in \bullet t} (m_i \wedge u_i \leq x_i) \wedge \bigwedge_{p_i \in \circ t} \neg(m_i \wedge l_i \leq x_i).$$

On the other hand, t can not fire when some $p \in \bullet t$ do not have enabled token or some $p \in \circ t$ have enabled token, thus $Ds_t(s)$ which denotes t can not fire in a state s can be defined as follows,

$$Ds_t(s) \stackrel{\text{def}}{=} \bigvee_{p_i \in \bullet t} \neg(m_i \wedge l_i \leq x_i) \vee \bigvee_{p_i \in \circ t} (m_i \wedge u_i \leq x_i).$$

Note that $En_t(s)$ and $Ds_t(s)$ are not exclusive. It may be impossible to decide whether t can fire or not if $l_i \leq x_i < u_i$ for some places $p_i \in \bullet t$ or $p_i \in \circ t$. In such cases, whether t can fire or not is decided non-deterministically, and then neither $En_t(s)$ nor $Ds_t(s)$ holds.

A delay step is not performed when there exist some transition which can fire. Thus $C(s, d, s')$ can be defined as follows,

$$\begin{aligned} C(s, d, s') &\stackrel{\text{def}}{=} \bigwedge_{t \in T} \neg En_t(s) \wedge \bigwedge_{p_i \in P} (x'_i = x_i + d \wedge m'_i \leftrightarrow m_i) \wedge d > 0 \\ &\vee \bigwedge_{p_i \in P} (x'_i = x_i \wedge m'_i \leftrightarrow m_i) \wedge d = 0 \end{aligned}$$

where m'_i and x'_i represent the value of m_i and x_i in s' .

Here $F_t(s, s')$ denotes a characteristic function which holds iff $s \xrightarrow{t} s'$ or $s = s'$. Then $F(s, s')$ of TPN which has n transitions ($n = |T|$) can be defined as follows,

$$F(s, s') \stackrel{\text{def}}{=} F_{t_1}(s, s_1) \wedge F_{t_2}(s_1, s_2) \wedge \cdots \wedge F_{t_n}(s_{n-1}, s').$$

It is clear that $F(s, s')$ holds if s can reach s' by a sequence of transitions following the order in $t_1, \dots, t_n$. $F_t(s, s')$ can be defined as follows,

$$\begin{aligned} F_t(s, s') &\stackrel{\text{def}}{=} \neg Ds_t(s) \wedge \bigwedge_{p_i \in t \bullet} (m'_i \wedge x'_i = 0) \wedge \bigwedge_{p_i \in \bullet t \setminus t \bullet} (\neg m'_i \wedge x'_i = 0) \\ &\quad \wedge \bigwedge_{p_i \in P \setminus (\bullet t \cup t \bullet)} (m'_i \leftrightarrow m_i \wedge x'_i = x_i) \\ &\quad \vee \bigwedge_{p_i \in P} (m'_i \leftrightarrow m_i \wedge x'_i = x_i). \end{aligned}$$

Note that if neither $En_t(s)$ nor $Ds_t(s)$ holds, both $C(s, d, s')$ and $F_t(s, s')$ can be hold and then it is decided non-deterministically whether t can fire or not.

$\mathcal{N}_k$ which represents k -steps is obtained as conjunction of $T(s, d, s')$ as follows,

$$\begin{aligned} \mathcal{N}_k &\stackrel{\text{def}}{=} I(s_0) \wedge T(s_0, d_1, s_{n+1}) \wedge T(s_{n+1}, d_2, s_{2(n+1)}) \\ &\quad \wedge \cdots \wedge T(s_{(k-1)(n+1)}, d_k, s_{k(n+1)}) \\ &= I(s_0) \wedge C(s_0, d_1, s_1) \wedge F_{t_1}(s_1, s_2) \wedge \cdots \wedge F_{t_n}(s_n, s_{n+1}) \\ &\quad \wedge \cdots \wedge C(s_{(k-1)(n+1)}, d_k, s_{(k-1)(n+1)+1}) \\ &\quad \wedge F_{t_1}(s_{(k-1)(n+1)+1}, s_{(k-1)(n+1)+2}) \wedge \cdots \wedge F_{t_n}(s_{k(n+1)-1}, s_{k(n+1)}) \end{aligned}$$

where $I(s)$ denotes a characteristic function of initial marking.

3.2.2 Symbolic representation of properties

Here $R(s)$ denotes a characteristic function of the states where the given property is satisfied. When $\mathcal{N}_k$ holds and $R(s_i)$ holds in some state s_i , $\mathcal{N}_k$ also holds if s_i is assigned to $s_{i+1}, \dots, s_{k(n+1)}$ because $C(s, d, s')$ and $F_t(s, s')$ hold if $s = s'$ and $d = 0$. Thus $\mathcal{N}_k \wedge R(s_{k(n+1)})$ holds. On the other hand, if $\mathcal{N}_k \wedge R(s_{k(n+1)})$ holds, the given property is satisfied in k -steps. Therefore, we obtain $\mathcal{R}_k \stackrel{\text{def}}{=} R(s_{k(n+1)})$.

In this paper, we check deadlock freeness of TPN. When all tokens of places are enabled and no transition can fire, deadlock occurs. Thus a characteristic function $R_d(s)$ which denotes deadlock states can be defined as follows,

$$R_d(s) \stackrel{\text{def}}{=} \bigwedge_{t \in T} \neg En_t(s) \wedge \bigwedge_{p_i \in P} (m_i \rightarrow u_i \leq x_i).$$

3.3 Reducing verification cost by transforming formula

3.3.1 Expanding search space by ordering transitions

$F(s, s')$ evaluates to true when both $s \xrightarrow{t_i} s''$ and $s'' \xrightarrow{t_j} s'$ are satisfied for some state s'' and some transitions t_i and t_j where $i < j$. Therefore, by solving satisfiability of $F(s, s')$, we can search the path from s to s' by firing of multiple transitions which follow the order in $F(s, s')$. For example, in the TPN of Fig. 1 t_1 and t_2 can fire successively without time elapsing. Then $\mathcal{N}_1$ can search the path from the initial state to the marking reached by firing of t_1 and t_2 . By the similar manner in [4], we decide the order of transitions by depth first search of PN from the initial marking.

3.3.2 Reducing formula size by replacing variables

Since only a few places are involved in firing of a transition, most of variables do not change by firing of the transition. Such unchanged variables are represented as terms formed by $x' = x$. Since that terms are connected as conjunction in T , the satisfiability of T is preserved even if all variables x' in T are replaced by x and the terms $x' = x$ are removed. This substitution can reduce the size of formula considerably.

Table I. The size of example TPNs.

Model	# of p	# of t	deadlock
m1	140	159	exist
m2	154	143	not exist
m3	374	389	not exist
m4	1030	1193	not exist

4 Experiments

In this section, we show the effectiveness of our encoding through the comparative experiments. We conducted experiments on the computer with Fedora 15 OS (64 bit), Intel Core i3 3.1 GHz CPU and 16 GB Memory. We use yices [8] as SAT solver. Table I shows the size of TPNs which model data transfer process of asynchronous systems. It is already known that only model m1 has a deadlock.

We first compared the time to detect the deadlock by our method and that by UPPAAL [9]. Input of UPPAAL is generated by the translation in [10]. As a result of experiments, the time for verification of model m1 by UPPAAL is 15,824 (sec.) and that by our method is 22 (sec.).

We also show the relation between the effect of variable replacing and the size of TPN. We apply the proposed encoding to the TPNs m2, m3 and m4 and check the deadlock freeness by yices. The number of steps is increased from 1 to 16 within the range that the verification time is less than 10,000 (sec.). The results are shown in Fig. 2. As shown in the figure, the verification time are greatly reduced by

Fig. 2. Relation between number of steps and verification time.

variable replacing. In addition, the effect of variable replacing increases as the size of TPN gets large. Although the size of m4 is almost three times of that of m3, the time required for verification of m4 is nearly equal to that of m3. This seems to be because variables which change by firing of each transition does not increase even if the number of places dramatically increases.

5 Conclusion

We proposed a symbolic representation of TPN for bounded model checking. In the proposed encoding scheme, firing of transitions and elapsing delays are represented as boolean formulas discretely. Our encoding can check two more firing of transitions by the formula for 1-step. We showed the experimental result of deadlock detection for example TPN. The effectiveness of our encoding on verification for large scale asynchronous system is also shown.