ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 1, No. 1, 2006, *pp*. 37-46

Component-Level Reduction Rules for Time Petri Nets Based on DTPN

Jinlong Jiang[1 +], Xianzhong Zhou[2] and Yongcheng Sun[1]

[1]Department of Automation, Nanjing University of Science and Technology, Nanjing 210094, China
[2]School of Management and Engineering, Nanjing university, Nanjing 210093, China

(*Received Oct. 04, 2005, accepted Nov. 25, 2005*)

Abstract. Time Petri Nets (TPNs) are a popular Petri net model for specification and verification of real-time systems. A widely applied method for analyzing Petri nets is component-level reduction analysis. The existing technique for component-level reduction analysis transforms a TPN component to a constant size of simple one while maintains the net external observable timing properties, but it neglects the internal properties of component such as synchronization, conflict and concurrency. Based on Delay Time Petri Net (DTPN), the paper transforms a TPN component to DTPN model in order to preserve such properties as synchronization, conflict and concurrency during the reduction. For the sake of analyzing the DTPN model, the paper proposes new schedule analysis method. Finally, reduction rules based on DTPN are applied to the TPN model analysis in the command and control (C2) system.

Keywords: component-level reduction rules; DTPN model; new DTPN' schedule analysis method; C2 system.

1. Introduction

TPNs are a most widely used model for real-time system specification and verification [1,2,3,4]. A fundamental and most widely applied method for analyzing Petri net's model is reachability analysis. It can enumerate all the reachable state then gets the state transition graph [2]. This representation makes explicit such properties as deadlock freedom and reachability. For a complex or even middle-sized TPN, however, it is difficult to enumerate its reachable state, which is commonly referred to state-explosion problem. Sloan et al. developed reduction rules about place fusion and transition fusion, which works at individual place and transition level [5]. But these reduction rules contain such defects as inefficiency of verification and frequency of operation. It is necessary to reduce TPN model at a coarse grained level so as to make TPN model expedite constringency.

A set of component-level reduction rules for TPN model is proposed in [4]. Each of reduction rules transforms a TPN component to a constant size of simple one while maintains the net's external observable timing properties. It works at a much coarser level than those developed by Sloan et al., and fewer applications of these rules are needed to reduce the size of the TPN. These rules reduce the complexity of TPN, however, neglect the internal properties of component such as conflict, synchronization and concurrency. It leads to the occurrence of events that shouldn't be taken place. In figure 1 (a), $t4$ should happen before $t6$, so place $p9$ cannot get one token. But $p7$ and $p9$ have the same chance to get one token in figure 1 (b). It is obvious that TPN model after transformation is not consistent with TPN model before transformation. This paper introduces DTPN [6] and transformations a TPN component to DTPN. It preserves the external observable timing properties while maintains such properties as conflict, synchronization and concurrency.

A state of DTPN is a pair S = （M， Θ） [6],where:

- $M = (pi[EDPi, LDPi], \cdots)$;

+ Corresponding author.
E-mail address: jjlong88@126.com

- Θ is a global (dynamic) time. It indicates that one token arrives at Pi in the dynamic interval $[EDPi+\theta, LDPi+\theta]$.

At run time, a *set of dynamic intervals* EA is associated with each state S. $EA=(TOKij(pj,ti)[\theta Eai, LEai],\cdots)$, which indicated that token $TOKij$ in place pj is able to enable transition ti after a time in the dynamic interval $[\theta Eai,\theta Lai]$. A *set of dynamic transition firing intervals* FT is associated with each state S. $FT=(ti[\theta ETi,\theta LTi]\ \{TOKi1,\cdots,TOKin\},\cdots)$, which indicated that transition ti must fire in the dynamic interval $[\theta ETi,\theta LTi]$ by using the token set $\{TOKi1,\cdots,TOKin\}$ if no token in this token set is removed before the firing.

Assume the current state is S. When a transition ti in FT fires at dynamic time $\theta 1$, a new state S' is obtained by:

① Changing the dynamic time θ into $\theta 1$.

② Removing $pj[EDPj,LDPj]$ from M , where pj is an input place of transition ti . Adding $pk[EDPk,LDPk]$ into M , where pk is an output place of transition ti . Modifying $pn[EDPn,LDPn]$, where pn is corresponding to a token not used for transition ti' firing, $EDPn'=\max(0,EDPn+\theta-\theta 1)$, $LDPn'=\max(LDPn+\theta-\theta 1)$.

③ Updating EA. Removing $TOKmj(pj,tm)[\theta EAm,\theta LAm]$, where tm is the output transition of place pj. Adding $TOKsk(pk,ts)[\theta EAks,\theta LAks]$, where ts is the output transition of place pk. $[\theta EAks,\theta LAks]=[\theta 1,\theta 1]+[EDAik,LDAik]+[EDAks,EDAks]$, where $[EDAik,LDAik]$ is the static interval on $arc(ti,pk)$, $[EDAks,LDAks]$ is the static interval on $arc(pk,ts)$.

④ Updating FT. $ts[\theta ETs,\theta LTs]\{TOKs1,\cdots,TOKsl,\cdots,TOKsn\}$ is added into FT if there exists an element $TOKsl(pl,ts)\in EA_{k+1}, l=1,\cdots,n$ in EA for each input place pl of transition ts , where $[\theta ETs,\theta LTS]=[ETs,LTs]+MAX_{pl}([\theta EAls,\theta LAls])$, $[ETs,LTs]$ is the static interval of ts , $[\theta EAls,\theta LAls]$ is the dynamic interval for $TOKsl(pl,ts)$ in the updated EA.

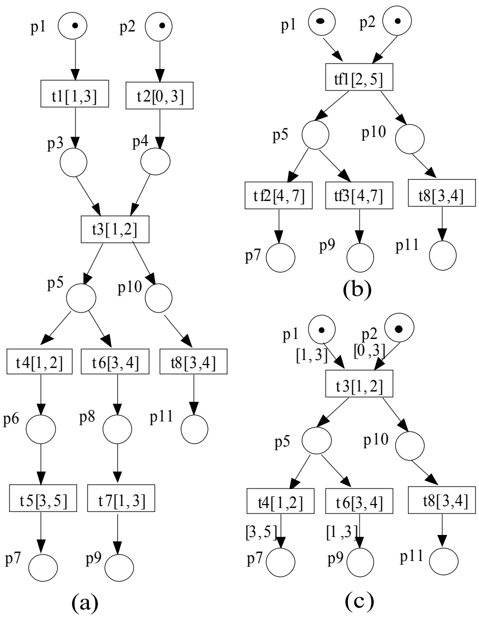


Figure 1 (a) TPN model (b) TPN model after reduction (c) DTPN model after reduction.

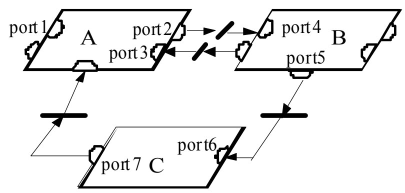


Figure 2 Framework of compositional TPN model

As shows in figure (1c), assume that the global time starts from θ . The initial state is $S0=(p1[0,0],p2[0,0],\theta)$. State $S1=(P5[0,0],P10[0,0],\theta+5)$ is reachable through the firing of $t3$. The delay between $S1$ and $S0$ is $(\theta+5)-\theta=5$. The global time is the value. It is unreasonable in TPN model [1]. We use the interval instead of the value in order to improve the representation. At the same time it lacks the specific schedule analysis method according to concurrency, synchronization and conflict.

Our main contributions include: ① this paper presents several component-level reduction rules based on DTPN, ② an new schedule analysis approach for DTPN which integrates static interval and dynamic interval is proposed.

2. New Schedule Analysis method

There are two changes about the state definition of new method: ① the global time ST is a interval; ② the definition of FT: $(ti[\theta ETi,\theta LTi][R\theta ETi,R\theta LTi]\{TOKi1,\cdots,TOKin\},\cdots)$, which associates each transitions with both static firing interval $[R\theta ETi,R\theta LTi]$ and dynamic firing interval $[\theta ETi,\theta LTi]$. Assume the current state is S_k and the global time is ST_k. The procedure of new method is as follow:

① If $ti \notin FT_k$, then transition ti is not nonschedulable;

② Let $MR\theta LT_k = \min(R\theta LTj / tj \in FT_k)$, $M\theta LT_k = \min\{\theta LTj / Tj \in FT_k\}$. If $R\theta ETi \le MR\theta LT_k$, then ti is schedulable at marking M_k. The static firing interval of ti is $[R\theta ETi, MR\theta LT_k]$ and the dynamic firing interval of ti is $[\theta ETi, M\theta LT_k]$;

③ $ST_{k+1} = [\theta ETi, M\theta LT_k]$;

④ M' change is the same as above.

⑤ Updating EA_k: Removing $TOKmj(pj,tm)$ $[\theta EAm,\theta LAm]$ from EA_k, where tm is the output transition of place pj. Adding $TOKsk(pk,ts)[\theta EAs,\theta LAs]$ into EA_k, where ts is the output transition of place pk and $[\theta EAs,\theta LAs] = ST_{k+1} + [EDAik,LDAik] + [EDAks,EDAks]$, $[EDAik,LDAik]$ is the static interval on $arc(ti,pk)$ and $[EDAks,LDAks]$ is the static interval on $arc(pk.ts)$.

⑥ Updating FT_k: Removing $tm[\theta ETm,\theta LTm]$ $[R\theta ETm,R\theta LTm]$ $\{TOKm1,\cdots,TOKmn\}$ from FT_k, where $TOKmj \in \{TOKi1,\cdots,TOKin\}$. $ts[\theta ETs,\theta LTs]$ $[R\theta ETs,R\theta LTs]$ $\{TOKs1,\cdots,TOKsl,\cdots,TOKsn\}$ is added into FT_k, where $TOKsl(pl,ts) \in EA_{k+1}$, $l = 1,\cdots,n$. Dynamic firing interval of transition ts is $[\theta ETs,\theta LTS] = [ETs,LTs] + MAX_{pl}([\theta EAls,\theta LAls])$, where $[ETs,LTs]$ is the static firing interval of transition ts and $[\theta EAls,\theta LAls]$ is the dynamic firing interval on $TOKsl(pl,ts)$ in EA_{k+1}. Relative firing interval of transition ts is $[R\theta Els,R\theta Lls] = [EDAhl,LDAhl] + [EDAls,LDAls,]$, where $[EDAhl,LDAhl]$ is the static firing interval on $arc(th,pl)$ and $[EDAls,LDAls]$ is the static firing interval on $arc(pl,ts)$, th is a input transition of place pl. If $ta[\theta ETa,\theta LTa][R\theta ETa,R\theta LTa]$ $\{TOKa1,\cdots,TOKan\}$ is an inherited friable transition (enabled by both M_k and M_{k+1}), then modifies its relative firing interval and absolute firing interval $[\theta ETa',\theta LaS'] = [\max(\theta ETa,\theta ETi),\theta LTa]$, $[R\theta ETa',R\theta LTa']$ $= [\max(0,R\theta ETa - RMLT_k), R\theta LTa - R\theta ETi]$.

As shows in figure (1c), the initial state $S_0 = (p1[0,0], p2[0,0],[0,0])$. If $t3$ fires, then $S_1 = (p5[0,0],[2,5])$ and $FT_1 = (t4[3,7][1,2], t6[5,9][3,4])$. Analyzing static firing intervals of two friable transitions, it appears that $t4$ happens before $t6$. At the same time, there is a conflict between $t4$ and $t6$, so $t6$ cannot fire. Hence the final state is $S2 = (p10[3,5],[3,7])$, that is $S2 = (p10[0,0],[6,12])$. It is obvious that the time delay between $S2$ and $S0$ is $[6,12]$.

3. Component-level Reduction Rules Based on DTPN

The building blocks of a compositional TPN are component [7]. A component is a coarse grained subnet of a TPN. A compositional TPN consists of two basic elements: component TPN and inter-component connections. Figure 2 shows an example of a compositional TPN model. The model has three components – A, B and C. Each component has two parts: ① communication ports (denoted by half circles), including input ports (e.g. port6) and output ports (e.g. port7).

3.1. Reduction Rules

The following definition is useful in the proofs of the following theorems.

Definition 1. Component Nc is transformed into Nc'. Nc satisfies: when all the input ports contain a token, the output port can receive token. If Nc' also satisfies the above condition, we can say Nc' preserves synchronization property. In figure 1 (a), when $p1$ and $p2$ both contain a token, $p5$ can receive a token. The same happen in (b) and (c). So (b) and (c) preserve the synchronization property of (a).

Definition 2. Component Nc is transformed into Nc'. Nc satisfies: when the input port contains a token, all the output ports can receive token. If Nc' also satisfies the above condition, we can say Nc' preserves concurrency property. After the firing of $t3$, $p5$ and $p10$ can receive a token in figure 1 (a). The same happen in (b) and (c). So (b) and (c) preserve the concurrency property of (a).

Definition 3. Component Nc is transformed into Nc'. Nc satisfies: when all the input ports contain a token, only one of all the output ports can receive token. If Nc' also satisfies the above condition, we can say Nc' preserves conflict property. In figure 1 (a), when $p5$ contains a token, $p7$ receives a token. But in (b), $p7$ and $p8$ can have a opportunity to obtain a token. In (c), only $p7$ can obtains a token. So (c) preserves the conflict property of (a), but (b) don't. If we change the time delay on $t6$ into $[1,3]$, (b) also can preserve the conflict property of (a).

Definition 4. There are two component TPN model C and C'. C' is the refined model of C. If two condition are satisfied: ① C and C' have the same input ports and output ports, ② C' is subject to all the constrains of C, then we say that C' is the correct refined model of C.

Component-Level Reduction Rule 1

Let N be the TPN model of a system, and Nc the TPN model of a component in the system. $\text{C.PORT\_IN} = \{\text{port1}\}$, $\text{C.PORT\_OUT} = \{\text{port2}\}$. The component has no enabled transition under the initial marking of N. If

① Whenever both $port1$ receives a token, $port2$ is guaranteed to receive a token in the future, and

② $port1$ can't receive another token until $port2$ received a token.

Then we can reduce N into N' by replacing Nc with a simple net which is compose of three places: $port1$, $port2$ and one transition: t, such that

① $port1^{*}={}^{*}port2=\{t\}$, ${}^{*}t=\{port1\}$ $t^{*}=\{port\}$, which ${}^{*}port1$ and $port2^{*}$ remain unchanged, and transition t also remains the same delay interval, and

② $SI(arc(port1, port2)) = SI\_(t)$,. $SI(arc(port1, port2))$ is the delay interval from $port1$ to $port2$. $SI\_(t)$ is the delay interval on t.

Theorem 1. The component-level reduction rule 1 is timing property preserving. (The proof of theorem 1 can be found in [4])

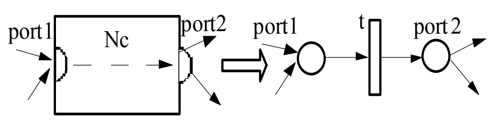


Figure3 Illustration of reduction rule 1

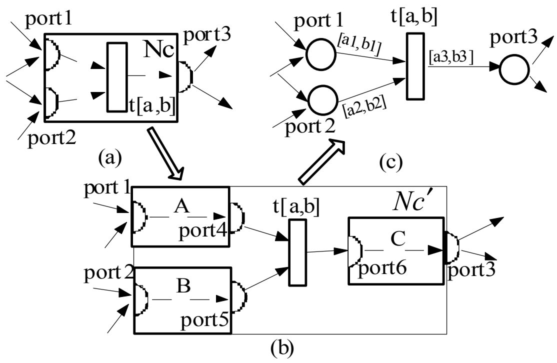


Figure 4 Illustration of reduction rule 2

Component-Level Reduction Rule 2

Let N be the TPN model of a system, and Nc the TPN model of a component in the system. $\text{C.PORT\_IN} = \{\text{port1,port2}\}$, $\text{C.PORT\_OUT} = \{\text{port3}\}$. The component has no enabled transition under the initial marking of N. If

① Whenever both $port1$ and $port2$ receive a token, $port3$ is guaranteed to receive a token in the future, and

② At least one of $port1$ and $port2$ can't receive another token until $port3$ received a token.

Then we can reduce N into N' by replacing Nc with a simple net which is compose of three places: $port1$, $port2$ and $port3$, and one transition: t, such that

① $port1^{*} = port2^{*}={}^{*}port3 = \{t\}$, ${}^{*}t = \{port1, port2\}$ $t^{*} = \{port3\}$, which ${}^{*}port1$, ${}^{*}port2$ and $port3^{*}$ remain unchanged, and transition t also remains the same delay interval, and

② $SI(arc(port1,t))$ = $SI\_(port1,{}^{*}t)$, $SI(arc(port2,t))$ = $SI(port2.{}^{*}t)$, $SI(arc(t,port3))$ = $SI\_(t^{*},port3)$. $SI(arc(port1,t))$ is the delay interval on the arc from $port1$ to t . $SI(arc(t,port3))$ is the delay interval on the arc from t to $port2$.

Theorem 2. The component-level reduction rule 2 is timing property preserving and preserves the synchronization property.

***Proof*:** In figure 4, Nc in (a) is refined into Nc' in (b). Hence we first prove Nc' is the correct refined model of Nc .

$Nc.PORT\_IN = \{port1, port2\} = Nc'.PORT\_IN$, $Nc.PORT\_OUT = \{port3\} = Nc'.PORT\_OUT$. It indicates that Nc and Nc' have the same input ports and output ports.

From preconditions ① and ② of Nc , it must contain a synchronizing transition t , which synchronizes tokens sent by $port1$ and $port2$, then new token is received by $port3$.

The principle of constructing Nc' : maintaining the synchronizing transition t and replacing the middle model of $port1 \to t$, $port2 \to t$ and $t \to port3$ by sub-component A, B and C. We can see that Nc' preserves synchronization property of Nc .

Timing constrains are maintained : $SI\_(port1, port4) = SI\_(port1,{}^{*}t)$, $SI\_(port2, port5) = SI\_(port2,{}^{*}t)$, $SI\_(port6, port3) = SI\_(t^{*}, port3)$.

According to the definition 4, Nc' is the correct refined model of Nc .

Then sub-component A, B and C in (b) are reduced by reduced rule 1, respectively. DTPN model in (c) can be obtained.

So the component-level reduction rule 2 is timing property preserving The proof of theorem 1 can be found in [4].

Component-Level Reduction Rule 3

Let N be the TPN model of a system, and Nc the TPN model of a component in the system. C.PORT_IN = {port1} , C.PORT_OUT = {port2,port3} . The component has no enabled transition under the initial marking of N . If

① Whenever $port1$ receives a token, $port2$ and $port3$ are guaranteed to receive a token in the future, and

② $port1$ can't receive another token until one of $port2$ and $port3$.

Then we can reduce N into N' by replacing Nc with a simple net which is compose of four places: p , $port1$, $port2$ and $port3$, and three transitions: t , $t1$ and $t2$, such that

① $port1^{*} = \{t\}$, ${}^{*}port2 = \{t1\}$, ${}^{*}port3 = \{t2\}$, ${}^{*}p = \{t\}$
$p^{*} = \{t1, t2\}$, $t^{*} = \{p\}$, ${}^{*}t = \{port1\}$, $t1^{*} = \{port1\}$,

${}^{*}t1 = \{p\}$, ${}^{*}t2 = \{p\}$ ${}^{*}t2 = \{port2\}$, which p , ${}^{*}port1$, $port2^{*}$ and $port3^{*}$ remain unchanged, and $t1$ and $t2$ also remain the same delay interval, and

② $SI(arc(port1,t) = SI\_(port1, p)$, $SI(arc(t1, port2)) = SI\_(t1^{*}, port2)$, $SI(arc(t2, port3)) = SI\_(t2^{*}, port3)$.

Theorem 3. The component-level reduction rule 3 is timing property preserving, also preserves conflict property.

***Proof*:** In figure 5, Nc in (a) is refined into Nc' in (b). Hence first proves Nc' is the correct refined model of Nc .

$Nc.PORT\_IN = \{port1\} = Nc'.PORT\_IN$, $Nc.PORT\_OUT = \{port2, port3\} = Nc'.PORT\_OUT$. It indicates that Nc and Nc' have the same input ports and output ports.

From preconditions ① and ② of Nc model, it must contain a conflict-fork place p that there is a conflict among its output transitions , two forking transitions $t1$ and $t2$ in the inside of component.

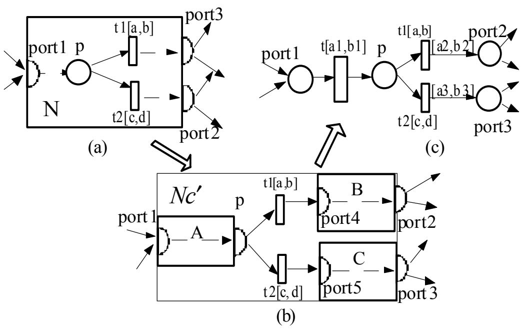
The principle of constructing Nc' : maintaining p , $t1$ and $t2$, replacing the middle model of $port1 \to p$, $t1^{*} \to port2$ and $t2^{*} \to port3$ by sub-component A, B and C. We can see that Nc' preserves conflict property of Nc .

Timing constrains are maintained: $SI(A) = SI\_(port1, p)$, which $SI(A)$ is the delay interval on component A. $SI\_(port4, port2) = SI\_(t1^*, port2)$, $SI\_(port5, port3) = SI\_(t2^*, port3)$.

According to the definition 4, Nc' is the correct refined model of Nc.

Then sub-component A, B and C in figure (b) are reduced by rule 1, respectively. DTPN model in figure (c) can be obtained.

So the component-level reduction rule 2 is timing property preserving. The proof of theorem 1 can be found in [4].



Figrue 5 Illustration of reduction rule 3

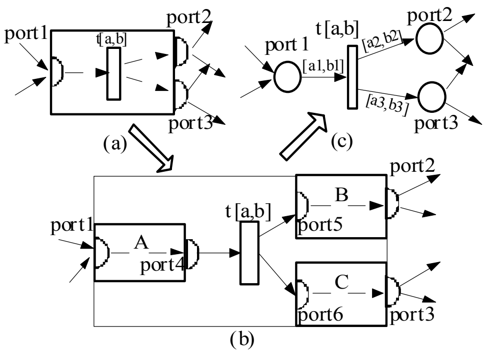


Figure 6 Illustration of reduction rule 4

Component-Level Reduction Rule 4

Let N be the TPN model of a system, and Nc the TPN model of a component in the system. $\text{C.PORT\_IN} = \{\text{port1}\}$, $\text{C.PORT\_OUT} = \{\text{port2,port3}\}$. The component has no enabled transition under the initial marking of N. If

① henever $port1$ receives a token, both $port2$ and $port3$ are guaranteed to receive a token in the future, and

② $port1$ can't receive another token until both of $port2$ and $port3$ received a token.

Then we can reduce N into N' by replacing Nc with a simple net which is compose of three places: $port1$, $port2$ and $port3$, and one transition: t, such that

① $port1^* = {}^*port2 = {}^*port3 = \{t\}$, ${}^*t = \{port1\}$, $t^* = \{port2, port3\}$, which ${}^*port1$, $port2^*$ and $port3^*$ remain unchanged, and transition t also remains the same delay interval, and

② $SI(arc(port1, t)) = SI\_(port1, {}^*t)$, $SI(arc(t, port2)) =$
$SI\_(t^*, port2)$, $SI(arc(t, port3)) = SI\_(t^*, port3)$.

Theorem 4. The component-level reduction rule 4 is timing property preserving, also preserves concurrency property.

Proof: In figure 6, Nc in (a) is refined into Nc' in (b). Hence first proves Nc' is the correct refined model of Nc.

$Nc.PORT\_IN = \{port1\} = Nc'.PORT\_IN$,

$Nc.PORT\_OUT = \{port2, port3\} = Nc'.PORT\_OUT$. It indicates that Nc and Nc' have the same input ports and output ports.

From preconditions ① and ② of Nc model, it must contain a concurrency-fork transition t in the inside of component. Through firing finite numbers a token received by $port1$ forked by transition t, then there are two concurrent forks, which are causally independent, one to $port2$ and the other to $port3$.

The principle of constructing Nc': maintaining the concurrency-fork transition t, replacing the middle model of $port1 \to t$, $t \to port2$ and $t \to port3$ by sub-component A, B and C. We can see that Nc' preserves conflict property of Nc.

Timing constrains are maintained:

$SI\_(port1, port4) = SI\_(port1, {}^*t)$, $SI\_(port5, port2) = SI\_(t^*, port2)$, $SI\_(port6, port3) = SI\_(t^*, port3)$.

According to the definition 1, Nc' is the correct refined model of Nc.

Then sub-component A, B and C in (b) are reduced by rule 1, respectively. DTPN model in (c) can be obtained.

So the component-level reduction rule 2 is timing property preserving . The proof of theorem 1 can be found in [4].

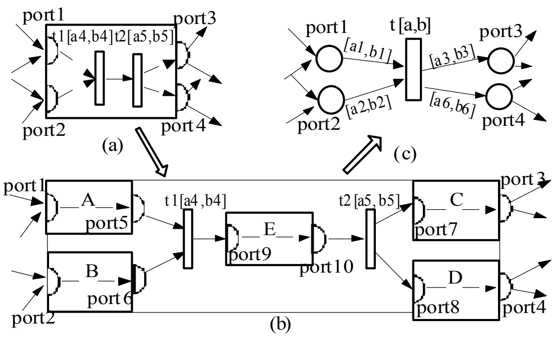


Figure 7 Illustration of reduction rules 5

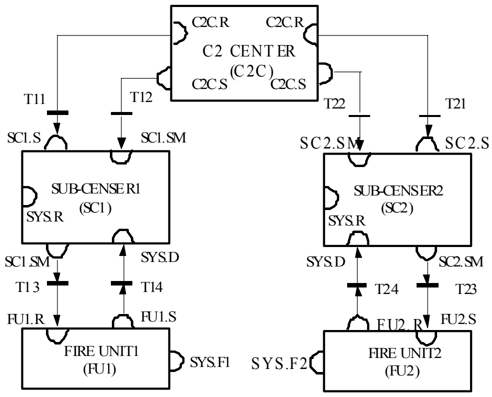


Figure 8 Composition interface of the C2 system

Component-Level Reduction Rule 5

Let N be the TPN model of a system, and Nc the TPN model of a component in the system. $\text{C.PORT\_IN} = \{\text{port1}, \text{port2}\}$, $\text{C.PORT\_OUT} = \{\text{port3}, \text{port4}\}$. The component has no enabled transition under the initial marking of N . If

① Whenever both $port1$ and $port2$ receive a token, both of $port3$ and $port4$ are guaranteed to receive a token in the future, and

② At least one of $port1$ and $port2$ cannot receive another token until both of $port3$ and $port4$ received a token.

Then we can reduce N into N' by replacing Nc with a simple net which is composed of three places: $port1$, $port2$, $port3$ and $port4$, and one transition: t , such that

① $port1^* = port2^* = {}^*port3 = {}^*port4 = \{t\}$, ${}^*t = \{port1, port2\}$, $t^* = \{port3, port4\}$, which ${}^*port1$, $port2^*$, $port3^*$ and $port4^*$ remain unchanged, and transition t also remains the same delay interval, and

② $SI(arc(port1,t)) = SI\_(port1, {}^*t1)$, $SI(arc(port2,t)) = SI\_(port2, {}^*t1)$, $SI(t) = SI(t1) + SI(t1^*, {}^*t2) + SI(t2)$, $SI(arc(t, port3)) = SI\_(t2^*, port3)$, $SI(arc(t, port4)) = SI\_(t2^*, port4)$.

Theorem 5. The component-level reduction rule 5 is timing property preserving, .also preserves synchronization and concurrency property.

Proof: As shows in figure 7, we can see that Nc in reduction rule 5 actually is the compositional model integrating Nc in reduction rule 2 with Nc in reduction rule 4.

Hence, the proof of theorem 5 can refer to the proofs of theorem 2 and theorem 4.

It should noted that:

① The component-level reduction rules are developed based on not only the external observable input-output patterns of component bet also the internal properties preserving.

② A component may be analyzed by reachability analysis method [8],. This is a fundamental and most widely applied method for analyzing TPNs. If necessary and possible, we can use some individual transition level reduction rules given in [5] to reduce the component before reachability analysis. In case a component is very complicated, we can also use simulation or test to obtain the timing parameters required by its reduced net.

4. Reduction Analysis for C2 system

In this section, we show the application of the improving reduction rules to the verification of timing properties of a command and control (C2) system. Figure 8 is the CTPN model of a anti-air system, which consists of two level command and control centers: one is C2 center ($C2C$), the other is SUB_CENTER. We focus on the verification of requirements on the time delays in the execution of the system functions.

Suppose that for a specific C2 system with the structure of Figure 8 there are following timing requirements:

(C1) The reaction time of the system must be less or equal to 40 time units.

(C2) The time delay from a detailed firing assignment scheme made by a SUB_CENTER to the result of the damage assessment referred to the execution of this scheme received by the same SUB_CENTER must be less than or equal to 20 time units.

(C3) Since the bottleneck for information processing is often located in the $C2C$, the center is always asked to respond quickly. This is captured by the requirement that the whole processing time for a group of messages from the two SUB_CENTER must be less than or equal to 15 time units.

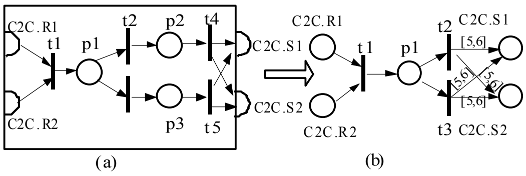


Figure 9 The TPN model of component C2C and reduction model
Firing times: t1: [1, 2], t2: [3, 5], t3: [3, 5], t4,t5: [5, 6]

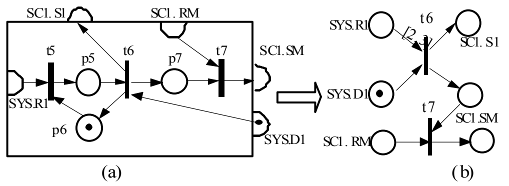


Figure 10 The TPN model of component Sub-Center 1 and reduction model. Firing times: t5: [2, 3], t6: [1, 2], t7: [4, 6]

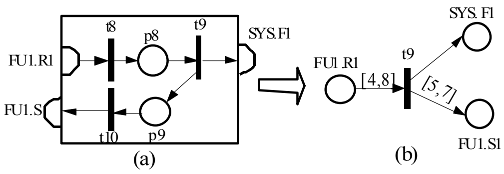


Figure 11 The TPN model of component Fire Unit 1 and reduction model.Firing times:
t8: [4, 6], t9: [1, 2], t10: [5, 7]

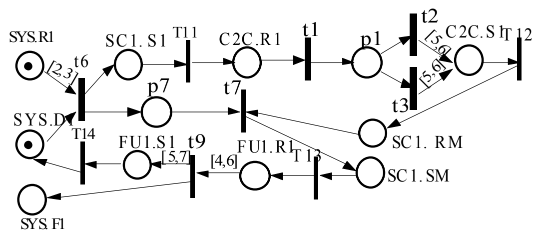


Figure 12 The DTPN model of the system

Figure 9 (a), figure 10 (a) and figure11 (a) show the internal structure of each TPN component. For detailed representation of each component see [3].

As show in figure 9 (b), figure 10 (b) and figure 11 (b), we apply reduction rules to reduce three components. Through analyzing figure 9(b), we can obtain: $SI(t2)=[3,5]<[6,7]=SI(t3)$, $t3$, that is intelligence seat 2, can't work. But $t2$, that is intelligence seat 1, work forever. It is appear that resource is assigned unreasonable. It can be resolved by changing the delay of $t3: SI(t3)=[3,5]$. Then DTPN model of the whole system can be obtained in figure 12. Through analyzing, we can compute: $SI(t_{(C2C.R,C2C.S)})$ =[9,13]<15, $SI(SC1.SM,SYS.D1)$ =[12,17]<20 and $SI(SYS.R1,SYS.F1)$ =[24,35]<40. It appears that **(C1)**, **(C2)** and **(C3)** are satisfied.

Port/Transaction	Description	C2CRI	C2C received message from Sub_Center1
SYS.R1	A message from Air Radar Group I arrived	C2CS1	C2C send command to Sub_Center1
SYS.F1	A combat command to Fire Unit1 sent	T11	Sub-Center 1 sends information to C2C, Fire time: [1,1]
SC1.S1	Sub_Center1 ready to send intelligence to C2C	T12	C2C sends command to Sub-Center 1, Fire time: [1, 1]
SC1.RM	Sub_Center1 received result C2C	T13	Sub_Center1 sends command to Fire Unit 1, Fire time: [1, 1]
SC1.SM	Sub_Center1redy to send command to Fire Unit1	T14	Fire Unit1 sends result of loss assessment to Sub-Center 1, Fire time [1, 1]
SC1.D1	Sub_Center1 received result of damage assessment from Fire Unit1	FU1.R1	Fire Unit 1 received command from Sub-Center1
FU1.S1	Fire Unit1 ready to send result of damage assessment to Sub_Center1		

Table 1 Legends of partial ports in Figure 7

5. Conclusion

This paper improves the existing component-level reduction rules. During the process of transformation new rules not only preserve timing properties, but also maintain such internal properties of component as conflict and concurrency. At the same time this paper proposes new schedule analysis method based on DTPN model, which offsets the limitation of the existing schedule analysis method [6]. Finally, this paper shows how to apply this reduction rules to timing property verification of the TPN model of a C2 system.

6. Reference

[1] B Berthomieu, D Diaz. Modeling and Verification of Time Dependent Systems Using Time Petri Nets, IEEE Transaction on Software Engineering, 1991,17(3), 259-275.

[2] G Bucci, E Vivario, Compositional Validation of Time-Critical Systems Using Communicating Time Petri Nets, IEEE Transactions on Software Engineering, 21(1995)12, 969-992.

[3] Jiacun Wang, Yi Deng, Guang Xu, Reachability Analysis of Real-Time System Using Petri Nets, IEEE Transactions on System, Man and Cybernetics, 30(2000)5, 725-736.

[4] Jiacun Wang, Yi Deng, Men Chun Zhou, Compositional Time Petri Nets and Reduction Rules, IEEE Transactions on System, Man and Cybernetics, 30(2000)4, 562-572.

[5] R Sloan, U Buy, Reduction Rules for Time Petri Nets, Acta Information, 33(1996), 687-706.

[6] E Juan, J Tsai, T Murata, Y Zhou, Reduction Rules for Real-Time Systems Using Delay Time Petri Nets, IEEE Transactions on Software Engineering, 27(2001)5, 422-448.

[7] Y Deng, J Wang, R Sinha, Integrated Architecture Modeling of Real-Time Concurrent Systems with Applications in FMS, Proceeding of the 10th International Conference on Software Engineering and Knowledge Engineering, San Francisco Bay, USA June 1998, 18-20.

[8] Dianxiang Xu, Xudong He, Yi Deng, Compositional Schedulability Analysis of Real-Time Systems Using Time P*etri Nets*. IEEE Transactions on Software Engineering, 28(2002)10, 984-996.