COMPOSITIONAL SCHEDULABILITY ANALYSIS OF WORKFLOW USING TIMING CONSTRAINT PETRI NETS

Peng Li, Qing Gu, Keqiang Cao, Daoxu Chen
State Key Laboratory of Novel Software Technology
Nanjing University
Nanjing, 210093, P.R. China
lipeng@dislab.nju.edu.cn

Jiangmin Zhu
Department of Computer Science
University of Texas at Dallas
Dallas, Texas, USA
jxz041000@utdallas.edu

ABSTRACT
The schedulability analysis is important in workflow modeling. This paper presents a new approach to the schedulability analysis of individual transition or transition sequences in Time constraint Petri nets (TCPNs). The reachability of markings can also be checked based on the schedulability analysis. If a specific transition sequence is schedulable, the corresponding task sequence can complete its execution successfully; otherwise, nonschedulable transitions should be pinpointed to help adjust timing constraints. A technique for compositional timing analysis is also proposed to deal with complex transition sequences, which improves efficiency of analysis. We have also constructed an automated tool: TCPN-PIPE2 to facilitate the schedulability analysis of workflow.

KEY WORDS
Workflow modeling, Timing constraints Petri nets, Reachability, Compositional schedulability, TCPN-PIPE2

1 Introduction
Managing time is important in designing and managing workflow, a workflow model requires to deal with time issues. The existing models for managing time are mainly based on workflow graphs and Petri nets. Based on Workflow graphs, Eder et al. [3], Marjanovic [4] and Zhuge Hai et al. [5] proposed their own workflow models dealing with time issues.

Petri nets have gained many applications in the field of workflow modeling [6, 7], since basic Petri nets lack the temporal description, several extended models of Petri nets considering time were proposed, such as Timed Petri nets (Timed PNs) [8] and Time Petri nets (Time PNs) [2, 9].

Our work studies schedulability analysis of workflow modeled in Timing Constraint Petri nets (TCPNs) which were proposed by Tsai et al. [1]. As a visual model, TCPNs are very expressive to depict time constraints; Furthermore, TCPNs follow the weak firing rule [1], which is more appropriate for modeling workflow than strong firing mode that Time (Timed) PNs use. Though TCPNs have some advantages for modeling workflow, the formulas of EFBT and LFET proposed in [1] are inconsistent with the meanings of timing constraints; moreover, the approach to schedulability analysis introduced by Tsai et al. is insufficient to determine schedulability of individual transition. These two shortcomings, in some respects, limit the usefulness of TCPNs.

According to new formulas of EFBT and LFET different from those presented in [1], our main results include: 1) a new approach to schedulability analysis of individual transition and transition sequences, 2) a compositional strategy for complex transition sequences, 3) an automated tool for building, simulating and analyzing TCPN models. This paper is organized as follows: Section 2 gives a brief introduction to TCPNs. Section 3 presents the schedulability analysis of individual transition and transition sequences. Section 4 describes the compositional schedulability analysis. Section 5 introduces the automated tool TCPN-PIPE2 and its application. Conclusions are drawn in Section 6.

2 Timing Constraints Petri Nets and Workflow

2.1 Timing Constraint Petri Nets

Definition 1. [1] A timing constraint Petri net is a 6-tuple \((P, T, F, C, D, M_0)\) where:

- \(P\) is a finite set of places.
- \(T\) is a finite set of transitions.
- \(F\) is a set of arcs which connect places and transitions.
- \(C\) is a set of integer pairs, \((\text{TC}_{\text{min}}(pt_j), \text{TC}_{\text{max}}(pt_j))\), where \(pt_j\) is either a place or a transition.
- \(D\) is a set of integer pairs, \([\text{FIRE}_{\text{dus}}(t_j)]\), where \(t_j\) is a transition.
- \(M_0\) is the initial marking function.

Without considering time constraints, \(P, T, F, M_0\) together define a Petri net, which could be denoted as the underlying net \(\text{UN}=\{P, T, F, M_0\}\). Given a marking \(M\) and a place \(p\), \(M(p)\) denote the number of tokens contained in \(p\) under the marking \(M\).

In TCPN, \(\text{TC}_{\text{min}}(p)/\text{TC}_{\text{max}}(p)\) are the minimum/maximum elapsed time intervals between the
token arrival time of \( p \) and the beginning/ending firing time of \( p \)'s output transitions. A transition \( t \) with a time pair, \((TC_{min}(t), TC_{max}(t))\), is said to be enabled under marking \( M \) if \((p \in \bullet t) M(p) \geq 1\). A transition \( t \), which is enabled at time \( T_0 \), is said to be firable during the time interval \([T_0 + TC_{min}(t), T_0 + TC_{max}(t)]\). Here what we should pay much attention on is that there is no guarantee that a firable transition \( t \) can complete the firing successfully in that the firing should take a period of time \( [FIRE_{dur}(t)] \). For any place or transition which has non-explicit timing constraints, the default value of a time pair is \((0, \infty)\) and the default of transition duration is 0. Fig. 1 shows time constraints in TCPN.

In comparison of other existing time-related Petri nets such as Time or Timed PNs, TCPNs have three main advantages for modeling workflow: 1) TCPNs include more expressive time constraints, in TCPN, given a workflow schema, a workflow designer can assign execution duration \( FIRE_{dur}(t) \) as well as relative executable interval \((TC_{min}(t), TC_{max}(t))\) for each task or transition. Whereas Timed PNs lack time pairs to restrict the execution of a transition, Time PNs can merely depict the transition which is instantaneous. In some cases, the resources or conditions modeled by places should also be limited under a time interval, the place time pair \((TC_{min}(p), TC_{max}(p))\) in TCPN can fulfill this requirement. 2) Different from Timed and Time PNs, TCPNs follow the weak firing mode. According to [1], the strong firing mode may cause two conflict transitions to fire simultaneously, which contradicts the definition of conflict structures; it also might be prone to cause dead transitions. 3) TCPNs are most suitable for systems with conflict structures.

### 3 Schedulability Analysis Using TCPN

#### 3.1 Basic Concepts

Here, we defined some other time constraints, \( \text{TOKEN}_{\text{arr}}(p) \) is an absolute time representing the time when the token arrives at place \( p \); \( FIRE_{\text{begin}}(t)/FIRE_{\text{end}}(t) \) is the time when transition \( t \) begins/ends the firing; \( EFBT(t)/LFET(t) \) denote the earliest firing beginning time and the latest firing ending time of transition \( t \) after \( t \) is enabled; \( IT(p)/OT(p) \) is a set of input/output transitions of \( p \); \( IP(t)/OP(t) \) is a set of input/output places of \( t \).

**Definition 2.** A transition \( t_j \) is said to be strongly firable if each of the input places of \( t_j \) currently has at least one token at the same time with the consideration of the actual arrival time of tokens in each input place. That is, \( t_j \) is strongly firable if and only if:

\[
EFBT(t) = \max\{\text{TOKEN}_{\text{arr}}(p) + TC_{\text{min}}(p)\}, \quad \max\{\text{TOKEN}_{\text{arr}}(p_j) : p_j \in IP(t_j)\} + TC_{\text{min}}(t_j)\}
\]

\[
LFET(t) = \min\{\text{TOKEN}_{\text{arr}}(p) + TC_{\text{max}}(p)\}, \quad \max\{\text{TOKEN}_{\text{arr}}(p_j) : p_j \in IP(t_j)\} + TC_{\text{max}}(t_j)\}
\]

and

\[
LFET(t_j) - EFBT(t_j) \geq FIRE_{\text{dur}}(t_j).
\]

**Definition 3.** A transition \( t_j \) is strongly schedulable if \( t_j \) is strongly firable and can complete firing successfully with considering the arrival time of tokens in each input place. That is, \( t_j \) is strongly schedulable if and only if:

\[
LFET(t_j) - EFBT(t_j) \geq FIRE_{\text{dur}}(t_j).
\]

The formulas (3.1) and (3.2) were proposed by Dianxiang Xu et al. [2], they described that \( EFBT(t_j) \) and \( LFET(t_j) \) in [1] are inconsistent with the meanings of time constraints in TCPN.

#### 3.2 Schedulability Analysis of A Transition

**Theorem 1.** If a transition \( t \) is strongly schedulable and needs to complete the firing successfully, then \( EFBT(t) \leq FIRE_{\text{begin}}(t) \leq LFBT(t) \). Where \( LFBT(t) \) represents the latest beginning firing time of \( t \), \( LFBT(t) = LFET(t) - FIRE_{\text{dur}}(t) \).

**Proof.** If a transition \( t \) is strongly schedulable, obviously, \( LFET(t) - EFBT(t) \geq FIRE_{\text{dur}}(t) \), and if \( FIRE_{\text{begin}}(t) > LFBT(t) \), then \( FIRE_{\text{end}}(t) = FIRE_{\text{begin}}(t) + FIRE_{\text{dur}}(t) > LFET(t) \), which means that \( t \) can not complete the firing, thus we can directly draw the conclusion.

Based on \( FIRE_{\text{begin}}(t), FIRE_{\text{end}}(t) \in [EFBT(t), LFET(t)] \), where \( EFBT(t) \) represents the earliest ending firing time of \( t \), and \( EFET(t) = EFBT(t) + FIRE_{\text{dur}}(t) \). The time interval \([EFBT(t), LFET(t)] \) can be considered as the decision span of a transition at run-time, the relation of these time constraints can be illustrated via Fig. 2.

**Theorem 2.** For \( \forall p_j \in OP(t_i) \), if \( t_i \) completes the firing successfully, then \( \text{TOKEN}_{\text{arr}}(p_j) = FIRE_{\text{end}}(t_i) \in [EFBT(t_i), LFET(t_i)] \).
Proof. Because TOKEN\textsubscript{arr}(p_j) = FIRE\textsubscript{end}(t_j), if \( t_j \) completes its firing successfully, we can directly get this theorem. \( \square \)

**Definition 4.** For \( \forall p_j \in IP(t_j) \), \( \text{Min}(\text{TOKEN}\textsubscript{arr}(p_j)) \) and \( \text{Max}(\text{TOKEN}\textsubscript{arr}(p_j)) \) can be used to represent the lower and upper bound of \( \text{TOKEN}\textsubscript{arr}(p_j) \). According to formulas (3.1) and (3.2), we can define \( \text{EFBT}'(t_j) \), \( \text{LFET}'(t_j) \), \( \text{EFBT}''(t_j) \) and \( \text{LFET}''(t_j) \) as follows:

- \[ \text{EFBT}'(t_j) = \text{Max}\{\text{Min}(\text{TOKEN}\textsubscript{arr}(p_j)) + \text{TC}_{\text{min}}(p_j) \}, \text{Max}\{\text{Min}(\text{TOKEN}\textsubscript{arr}(p_j)) + \text{TC}_{\text{min}}(t_j) \} \] (3.3)
- \[ \text{LFET}'(t_j) = \text{Min}\{\text{Max}(\text{TOKEN}\textsubscript{arr}(p_j)) + \text{TC}_{\text{max}}(p_j) \}, \text{Max}\{\text{Max}(\text{TOKEN}\textsubscript{arr}(p_j)) + \text{TC}_{\text{max}}(t_j) \} \] (3.3)
- \[ \text{EFBT}''(t_j) = \text{Max}\{\text{Max}(\text{TOKEN}\textsubscript{arr}(p_j)) + \text{TC}_{\text{max}}(p_j) \}, \text{Max}\{\text{Max}(\text{TOKEN}\textsubscript{arr}(p_j)) + \text{TC}_{\text{max}}(t_j) \} \] (3.4)
- \[ \text{LFET}''(t_j) = \text{Min}\{\text{Min}(\text{TOKEN}\textsubscript{arr}(p_j)) + \text{TC}_{\text{max}}(p_j) \}, \text{Max}\{\text{Min}(\text{TOKEN}\textsubscript{arr}(p_j)) + \text{TC}_{\text{max}}(t_j) \} \] (3.5)

and

\[ \text{EFBT'}(t_j) \leq \text{EFBT}(t_j) \leq \text{EFBT}''(t_j); \]
\[ \text{LFET'}(t_j) \leq \text{LFET}(t_j) \leq \text{LFET}''(t_j). \]

The schedulability of individual transition can be checked as following three cases:

**Case 1:** For a transition \( t_j \) with one input place \( p_j \):

\[ \text{EFBT}(t_j) = \text{TOKEN}\textsubscript{arr}(p_j) + \text{Max}\{\text{TC}_{\text{min}}(p_j), \text{TC}_{\text{min}}(t_j) \}; \]
\[ \text{LFET}(t_j) = \text{TOKEN}\textsubscript{arr}(p_j) + \text{Min}\{\text{TC}_{\text{max}}(p_j), \text{TC}_{\text{max}}(t_j) \}; \]

So the result of \( (\text{LFET}(t_j) - \text{EFBT}(t_j) )\) is not influenced by the value of \( \text{TOKEN}\textsubscript{arr}(p_j) \).

**Case 2:** For a transition \( t_j \) with several input places \( p_j \) (\( j = 1, \ldots, k \)):

Based on Definition 4, we can get \( \text{LFET}''(t_j) - \text{EFBT}'(t_j) \leq \text{LFET}(t_j) - \text{EFBT}(t_j) \leq \text{LFET}'(t_j) - \text{EFBT}'(t_j) \).

- If \( \text{LFET}''(t_j) - \text{EFBT}'(t_j) \geq \text{FIRE}_{\text{dur}}(t_j) \) \( \Rightarrow \) \( \text{LFET}(t_j) - \text{EFBT}(t_j) \geq \text{FIRE}_{\text{dur}}(t_j) \), then \( t_j \) is strongly schedulable.

- If \( \text{LFET}''(t_j) - \text{EFBT}'(t_j) < \text{FIRE}_{\text{dur}}(t_j) \) and \( \text{LFET}'(t_j) - \text{EFBT}'(t_j) \geq \text{FIRE}_{\text{dur}}(t_j) \), then \( t_j \) is perhaps strongly schedulable or not. In this case, \( t_j \) is in an unsafe state.

- If \( \text{LFET}(t_j) - \text{EFBT}'(t_j) < \text{FIRE}_{\text{dur}}(t_j) \) \( \Rightarrow \) \( \text{LFET}(t_j) - \text{EFBT}(t_j) < \text{FIRE}_{\text{dur}}(t_j) \), then \( t_j \) can not be strongly schedulable.

**Case 3:** For a transition \( t_j \) (\( j = 1, 2, \ldots, k \)) in a conflict structure:

The strength of TCPNs over other time-related Petri nets is in the modeling and analysis of conflict structures; the handling of any transition in a conflict structure is same as that of a transition with one input place.

### 3.3 Schedulability Analysis of a Transition Sequence

In TCPN, a marking \( M_n \) is said to be reachable if there exists a firing sequence \( \sigma = (M_0, t_1, M_1, \ldots, t_i, M_i, \ldots, t_n, M_n) \), or simply transition sequence \( \delta = (t_1, \ldots, t_n) \) that transforms \( M_0 \) to \( M_n \), and \( \delta \) is schedulable with respect to imposed time constraints.

**Definition 5.** A transition sequence \( \delta = (t_1, \ldots, t_n) \) is schedulable if and only if all transitions including in \( \delta \) are strongly schedulable.

In schedulability analysis of a transition sequence, there exists a Root\_Time for each transition \( t_i \), it means that the token arrival times of input places of \( t_i \) can be represented by Root\_Time uniformly. The schedulability of \( \delta = (t_1, \ldots, t_n) \) will be checked as follows:

**Step1:** Determine the initial marking \( M_0 \) and suppose \( \text{TOKEN}\textsubscript{arr}(M_0) \) is \( T_0 \).

**Step2:** According to the sequence \( \delta \), find the input places of \( t_i \) (\( i = 1, \ldots, n \)), identify the Root\_Time of \( t_i \) and token arrival time of input places, and then check the schedulability of \( t_i \). If \( t_i \) is strongly schedulable, then do Step 3; else, the unreasonable timing constraints should be modified to make \( t_i \) re-schedulable.

**Step3:** \( \text{FIRE}_{\text{end}}(t_i) \) should be computed for \( t_i \).

- For transition \( t_i \) with one input place:
  \[ \text{FIRE}_{\text{end}}(t_i) \in [\text{EFBT}(t_i) + \text{FIRE}_{\text{dur}}(t_i), \text{LFET}(t_i)]; \]
- For transition \( t_i \) with several input places:
  \[ \text{FIRE}_{\text{end}}(t_i) \in [\text{EFBT}'(t_i) + \text{FIRE}_{\text{dur}}(t_i), \text{LFET}'(t_i)]; \]

In Fig. 3, the schedulability of \( \delta = (t_1, t_2, \ldots, t_n) \) is checked as follows: Obviously, \( t_1, t_2, t_3 \) and \( t_4 \) are strongly schedulable. The Root\_Time of \( t_5 \) is \( T_2 \), which represents the firing end time of \( t_5 \), token arrive times of \( p_5 \) and \( p_6 \) are \( T_2 + 6, T_2 + 7 \) and \( T_2 + 8, T_2 + 9 \) respectively. Based on schedulability of transition with several input places, \( t_5 \) is also strongly schedulable, thus \( \delta \) is schedulable based on imposed time, the marking \( M_5 \) = \{ \{p_9 \} \} is reachable in TCPN. Suppose \( T_0 \) be the beginning time of total process, \( \text{FIRE}_{\text{end}}(t_i) \) based on \( T_0 \) can be computed, the ending time
of task sequence $\delta$.TOKEN$_{arr}(p_0)$ is $[T_0 + 24, T_0 + 29]$, which means that if $\delta$ could complete its execution successfully, it can’t end before $T_0 + 24$ or after $T_0 + 29$.

In a workflow modeling with TCPN, transition sequence $\delta$ can represent a task sequence. If $\delta$ is schedulable, for every task or transition $t_i \in \delta$, LFET($t_i$) - EFBT($t_i$) $\geq$ FIRE$_{dur}$(1), there must exist a decision span for each task at run-time, thus the total task sequence $\delta$ can complete the execution successfully.

4 Compositional Analysis of Schedulability

In this section, we describe how to conduct schedulability analysis by decomposing a firing sequence in UN into a number of subsequences. We use EN($M$) to denote the set of transitions enabled under marking $M$. According to [2], we can get the Definition 6:

**Definition 6.** Let $\sigma_1 = (M_{10}t_{11}M_{11} \cdots t_{1m}M_{1m})$ ($m \geq 1$) and $\sigma_2 = (M_{20}t_{21}M_{21} \cdots t_{2n}M_{2n})$ ($n \geq 1$) be two sequences in UN, where $M_{10}$ and $M_{20}$ are reachable from $M_0$, $\sigma_2$ is composable with $\sigma_1$ if and only if $M_{1m} = M_{20}$ and EN($M_{1m}$) $\cap$ EN($M_{1m-1}$) = $\emptyset$. The composition of $\sigma_2$ with $\sigma_1$, denoted as $\sigma_1 \cdot \sigma_2$, is

$$(M_{10}t_{11}M_{11} \cdots t_{1m}M_{1m}t_{21}M_{2m}
\cdots t_{2n}M_{2n})$$

Obviously, according to Definition 6, it is incorrect to separate concurrent transitions while decomposing a sequence.

**Theorem 3.** Let $\sigma_2$ be composable with $\sigma_1$. $\delta_1 \delta_2$ is schedulable if and only if both $\delta_1$ and $\delta_2$ are schedulable.

**Proof.** Let $\sigma_1 = (M_{10}t_{11}M_{11} \cdots t_{1m}M_{1m})$
$\sigma_2 = (M_{20}t_{21}M_{21} \cdots t_{2n}M_{2n})$
$\delta_1 = (t_{11} \cdots t_{1m})$
$\delta_2 = (t_{21} \cdots t_{2n})$
$\delta_1 \delta_2 = (t_{11} \cdots t_{1m}t_{21} \cdots t_{2n})$
AD$_i = [EFBT(t_i), LFET(t_i)]$

1. Suppose both $\delta_1$ and $\delta_2$ are schedulable. So there exist two sequences of absolute firing domains for checking the schedulability of $\delta_1$ and $\delta_2$, say, $(AD_{i1} \cdots AD_{i1} \cdots AD_{i1m})$, $(AD_{21} \cdots AD_{22} \cdots AD_{2n})$, and for each AD$_i$, there is: $LFET(t_i) - EFBT(t_i) \geq FIRE_{dur}(t_i)$. Since $\delta_2$ is composable with $\delta_1$, so $M_{1m} = M_{20}$, $(AD_{i1} \cdots AD_{i1} \cdots AD_{i1m}AD_{21} \cdots AD_{22} \cdots AD_{2n})$ is exactly the sequence of absolute firing domains for checking the schedulability of $\delta_1 \delta_2$. Thus $\delta_1 \delta_2$ is schedulable.

2. Suppose $\delta_1 \delta_2$ is schedulable. There exists a sequence of absolute firing domains for checking the schedulability of $\delta_1 \delta_2$, say, $(AD_{i1} \cdots AD_{i1} \cdots AD_{i1m}AD_{21} \cdots AD_{22} \cdots AD_{2n})$. Obviously, $(AD_{i1} \cdots AD_{i1} \cdots AD_{i1m})$ and $(AD_{21} \cdots AD_{22} \cdots AD_{2n})$ are sequences of absolute firing domains for checking $\delta_1$ and $\delta_2$, respectively. So, both $\delta_1$ and $\delta_2$ are schedulable.

**Theorem 4.** Let $\sigma_i$ ($1 \leq i \leq k$) be sequences, and $\sigma_i$ ($2 \leq i \leq k$) be composable with $\sigma_{i-1}$. $\delta_1 \cdots \delta_k$ is schedulable if and only if $\delta_i$ ($1 \leq i \leq k$) are all schedulable.

**Proof.** It is obvious if $k = 2$; if $k = 3$, then $\delta_1 \delta_2 \delta_3 = (\delta_1 \delta_2) \delta_3$. Let $\delta = $ $\delta_1 \delta_2 \delta_3 \delta_4 \delta_5 \delta_6 \delta_7$, $\delta_3$ is schedulable iff $\delta$ and $\delta_3$ are schedulable iff $\delta_1$, $\delta_2$, and $\delta_3$ are schedulable. Suppose $\delta = $ $\delta_1 \cdots \delta_{k-1}$ is schedulable iff $\delta_1$ ($1 \leq i \leq k - 1$) are all schedulable. $\delta_1 \cdots \delta_k$ is schedulable iff $\delta_1$ ($1 \leq i \leq k$) are all schedulable.

**Theorem 5.** Suppose sequence $\sigma_2$ is composable with itself (called self-composable) and with sequence $\sigma_1$ and sequence $\sigma_3$ is composable with $\sigma_2$. Let $\delta = (\delta_2)^k = \delta_2 \cdots \delta_2 \cdots \delta_2$, where the number of $\delta_2$ is $k$ ($k > 0$). $\delta_1 \delta_3$ is schedulable if and only if $\delta_1 \delta_2 \delta_3$ is schedulable.

**Proof.** If $\sigma_2 = (M_{20}t_{21}M_{21} \cdots t_{2n}M_{2n})$ is a self-composable sequence, then $M_{20} = M_{2n}$. For each $t_i$ in $\sigma_2$, the schedulability of $t_i$ does not change with times of execution of $\sigma_2$ increasing, thus the schedulability of $(\delta_2)^k$ is same as the schedulability of $\delta_3$, and according to Theorem 5, we can easily conclude this theorem.

Theorem 4 and Theorem 5 can simplify the schedulability analysis of sequences containing loops. Note that Theorems 3-5 are useful only if sequences can be decomposed, i.e., no concurrency is present at decomposition markings.

5 Schedulability Analysis of a Workflow Example Using TCPN-PIPE2

5.1 Description of TCPN-PIPE2

This subsection gives a brief description of TCPN-PIPE2, which is developed based on an open-source software: Platform Independent Petri Net Editor2(PIPE2). We have...
applied the ideas of compositional schedulability analysis to TCPN-PIPE2, the intent of which is to build, simulate, and analyze TCPN models.

TCPN Editor: Supports direct model construction in the TCPN formalism. The components in a TCPN model can be directly modeled in this graphical user interface, and automatically generates TCPN models encoded in XML format.

TCPN Simulator: Supports the simulation of TCPN models by moving the tokens on the firing of transitions, and performs simulation in step-by-step mode. The result of simulation is a list of simulation data such as: schedulability of transitions, Fire_End of transitions.

TCPN Analyzer: It has the duty of analyzing the simulation data, and determining the schedulability of specified transition sequence. If sequence is not schedulable, the Simulator can’t proceed until the time constraints are modified.

5.2 Comparisons

In [2], xu et al. proposed the compositional technique, but this technique is based upon a totally different model: TPNs. Compared with TPNs, TCPNs have distinctive time constraints and transition firing rules; Moreover, the schedulability analysis of TCPNs is different from that of TPNs.

There are many other famous time-related tools, such as TimeNET and F-net for stochastic Petri nets (SPNs), ORIS, TINA and Opera for TPNs and so on. Here we take TimeNET [10] and ORIS [11] for example, ORIS is a tool for analyzing extended TPN models, named Preemptive TPNs; TimeNET provides a unified framework for modeling and performance evaluation of non-Markovian stochastic Petri nets, they are both based upon different time-related Petri nets models from TCPNs.

Compared with tools listed or unlisted above, the functionality of TCPN-PIPE2 is somewhat limited. i.e., TCPN-PIPE2 can’t generate reachability graph, and can only implement the state space analysis of basic Petri nets.

Although the functionality is limited, the expressive semantics of TCPNs gives TCPN-PIPE2 a expressive power, which permits the treatment of practical schedulability problems. This tool provides an experiment platform for future research about TCPNs, we are currently extending the capabilities of TCPN-PIPE2 to analyze TCPN models with more complicated behaviors, even the SPNs and TPNs models.

5.3 Schedulability Analysis Using TCPN-PIPE2

Through TCPN-PIPE2, we modeled various time-dependent workflow architectures. Here, we should consider a famous workflow example: the processing of complaints [6].

The TCPN model of this workflow is built in TCPN Editor and shown in Fig. 5, in TCPN-PIPE2, we use interval (0,1000) to represent (0,∞). Without consideration of timing constraints, the functional requirements of above cases are easily analyzed. For example, δ = (t1,γ1,γ2,γ3)k (t11t12) γ4 = (t2,t3,t5,τ7) or (t2,t5,t3,τ7) or (t3,t2,t5,τ7) or (t3,t5,t2,τ7), γ = (t1,γ1,γ2,γ3)

is one of the four basic transition sequences in UN that model the functional behaviors of correspondent workflow. These four sequences can transform M0 to M∞ = {p11}. Next, we will present the schedulability analysis of δ, according to Theorem 5, because γ is self-composable, to check the schedulability of δ, we only need to check the schedulability of δ0 = (t1,γ1,γ2,γ3)k (t11t12) (k = 0) and δ1 = (t1,γ1,γ2,γ3)k (t11t12) (k = 1), suppose γ4 = (t2,t3,t5,τ7).

After modifying timing constraints of γ0 and γ1 from (1,5)][2,5][0] to (1,8)][2,7][0], δ0 and δ1 are both schedulable, then δ is schedulable.

Table 1. FIRE_END of transitions in δ

<table>
<thead>
<tr>
<th>Transition</th>
<th>Condition</th>
<th>FIRE_END</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>null</td>
<td>[t0 + 2, t6 + 8]</td>
</tr>
<tr>
<td>t2</td>
<td>null</td>
<td>[t0 + 4, t6 + 8]</td>
</tr>
<tr>
<td>t3</td>
<td>null</td>
<td>[t0 + 7, t6 + 10]</td>
</tr>
<tr>
<td>t5</td>
<td>null</td>
<td>[t0 + 2, t6 + 10]</td>
</tr>
<tr>
<td>t6</td>
<td>k ≥ 0</td>
<td>[t0 + 10 + 5 × k, t6 + 18 + 15 × k]</td>
</tr>
<tr>
<td>t7</td>
<td>k ≥ 0</td>
<td>[t0 + 11 + 5 × k, t6 + 21 + 15 × k]</td>
</tr>
<tr>
<td>t10</td>
<td>k ≥ 1</td>
<td>[t0 + 12 + 5 × (k - 1), t6 + 25 + 15 × (k - 1)]</td>
</tr>
<tr>
<td>t11</td>
<td>k ≥ 0</td>
<td>[t0 + 13 + 5 × k, t6 + 20 + 15 × k]</td>
</tr>
<tr>
<td>t12</td>
<td>k ≥ 0</td>
<td>[t0 + 14 + 5 × (k + 1), t6 + 23 + 15 × (k + 1)]</td>
</tr>
</tbody>
</table>

Table 1 shows the FIRE_END of transitions in δ based on modified time constraints. k is the times that γ executes, the value shown in table 1 is validated by TCPN-PIPE2. Similarly, other three sequences are also schedulable with respect to newly-modified timing constraints, the
final marking $M_n = \{p_{11}\}$ is reachable.

6 Conclusions and Future Work

The contributions of this paper include:

1. A detailed approach to schedulability analysis of individual transition and transition sequences.
2. A compositional technique to analyze the schedulability of sequences with loops.
3. An analyzing tool for TCPN models: TCPN-PIPE2

An interesting research problem is how the compositional strategy can be extended to analyze nets with irregular structures. Since there exist some weaknesses in TCPN-PIPE2, extending the capabilities of this tool is our on-going research task.

Acknowledgement

The work is partly supported by China NSF grant (No. 60402027), Jiangsu Provincial NSF grant (No. BK2006115), China 973 project (No. 2002CB312002).

References


