IEICE TRANS. INF. & SYST., VOL.E92–D, NO.3 MARCH 2009

PAPER

Reachability Analysis of Variants of Communication-Free Petri Nets

Chien-Liang CHEN[†a)], ***Member***, **Suey WANG**[††b)], ***and*** **Hsu-Chun YEN**[†,†††*c)], ***Nonmembers***

SUMMARY *Communication-free Petri nets* provide a net semantics for *Basic Parallel Processes*, which form a subclass of Milner's Calculus of Communicating Systems (CCS) a process calculus for the description and algebraic manipulation of concurrent communicating systems. It is known that the reachability problem for communication-free Petri nets is NP-complete. Lacking the synchronization mechanism, the expressive power of communication-free Petri nets is somewhat limited. It is therefore importance to see whether the power of communication-free Petri nets can be enhanced without sacrificing their analytical capabilities. As a first step towards this line of research, in this paper our main concern is to investigate, from the decidability/complexity viewpoint, the reachability problem for a number of variants of *communication-free Petri nets*, including communication-free Petri nets augmented with '*static priorities*,' '*dynamic priorities*,' '*states*,' '*inhibitor arcs*,' and '*timing constraints*.'

key words: *reachability, communication-free Petri nets*

1. Introduction

Petri nets are one of the most popular tools for modeling concurrent systems. In spite of their popularity, conventional Petri nets are very difficult to analyze. Over the years, various subclasses of Petri nets have been defined and investigated in the literature, hoping that by imposing certain structural or behavioral constraints on the basic model of Petri nets, such restricted classes become easier to analyze while retaining sufficient expressive power. A *communication-free Petri net* is a Petri net in which each transition has exactly one input place, and the firing of a transition removes exactly one token from its input place. As a modeling tool, the computational power of communication-free Petri nets is somewhat limited. The limitation is a direct consequence of the inability for communication-free Petri nets to model 'synchronization' actions, which require places to synchronize through transition firings.

With respect to the *reachability problem*, it has been shown in [7] that the problem is NP-complete for communication-free Petri nets (equivalently *commutative context-free grammars*). Considering the difficulty of analyzing conventional Petri nets [13], communication-free Petri nets exhibit a 'relatively low' complexity, which is attractive from an algorithmic viewpoint. On the negative side, however, the absence of synchronization mechanisms makes such a Petri net class a bit too weak from a practical viewpoint. Consequently, it is interesting to see how the complexity of checking reachability is affected when such Petri nets are extended with the notions of, say, 'priority', 'time,' among others. As a first step towards this line of investigation, this paper is concerned with the study the reachability problem from a decidability/complexity viewpoint for a number of *extended communication-free Petri nets*. In our setting, the basic model of communication-free Petri nets are augmented with '*static priorities*,' '*dynamic priorities*,' '*states*,' '*inhibitor arcs*,' and '*timing constraints*.'

In *static-priority* communication-free Petri nets, priorities are assigned to transitions statically, and at any instant, only transitions among the highest priority class are allowed to fire. For *dynamic-priority* communication-free Petri nets, the assignment of priorities to transitions varies from marking to marking, as opposed to a fixed assignment in the static-priority case. *State-extended* communication-free Petri nets can be viewed as a restricted class of *vector addition systems with states* whose *addition vectors* have at most one '-1' position (i.e., each vector can subtract one from a single position at most). For communication-free Petri nets with *inhibitor arcs*, the test-for-zero capabilities are explicitly allowed. Several timed versions of Petri nets have been proposed in the literature to introduce the timing information (either implicitly or explicitly) into the Petri net model (see, e.g., [14], [16]). A common approach is to associate each of the time-related transitions with an upper and lower bounds, defining the interval in which the transition must fire. In a more recent article [6] (see also [4]), a new model of 'timed' Petri nets for which transitions are annotated by clock constraints was introduced. Such a model can be thought of as a generalization of the so-called *timed automata* model [1]. To model real-time systems, real-value clocks are incorporated into the model of Petri nets in such a way that clocks measure the advances of time, and, with the help of the associated clock constraints, timing requirements for transition firings can be enforced. (To distinguish from those timed versions of Petri nets previously defined in the literature, such a model will be coined "*clocked Petri nets*", throughout the remainder of this paper.) As opposed to conventional timed Petri nets for which enabled transitions must

---

Manuscript received April 7, 2008.
Manuscript revised October 26, 2008.

[†] The authors are with the Dept. of Electrical Engineering, National Taiwan University, Taipei, Taiwan 106, ROC.
[††] The author is with Technology Department of Morgan Stanley Japan Securities Co., Ltd., Tokyo, 150–6008 Japan.
[†††] The author is with the Dept. of Computer Science, Kainan University, Taoyuan, Taiwan 338, ROC.
[*] Corresponding author.
a) E-mail: clyde@cobra.ee.ntu.edu.tw
b) E-mail: Eliza.Wang@morganstanley.com
c) E-mail: yen@cc.ee.ntu.edu.tw

DOI: 10.1587/transinf.E92.D.377

Table 1 Decidability/complexity results of the reachability problem for a variety of communication-free PNs. (NP-c denotes NP-complete; NEXPTIME = $\bigcup_{i \geq 0} NTIME(2^{n^i})$.)

	+ priority		*+ inhibitor arc*		*+ clock*		*+ priority + clock*
	static	dynamic	general	w/o inhibitor cycle	general	alter. RQ	
basic	NP-c	undecid.	?	NEXPTIME NP-hard	decidable	NP-c	undecid.
state-extended	undecid.	undecid.	undecid.	undecid.	decidable	decidable	undecid.

fire within their time bounds, clocked Petri nets (of [6]) allow time to elapse, causing enabled transitions to become disabled without being fired. In fact, this sort of 'lazy firing' semantics of clocked Petri nets is the key behind the decidability of the reachability problem. In this paper, we define the so-called *alternating RQ communication-free Petri nets* where R denotes the resets and Q denotes the queries — a restricted class of *clocked* Petri nets in which, aside from being communication-free structurally, resets and queries of a clock along any computation must appear alternatively. Our definition of alternating RQ communication-free Petri nets is mainly motivated by the work of [11] in which the so-called *alternating RQ timed automata* were defined. As it turns out, alternating RQ timed automata admit more efficient verification algorithms, in comparison with that for the general model of timed automata [11] (see also [12]). As indicated in [11], [12], many practical examples in the literature meet the alternating RQ constraints. Inspired by [4], [11], [12], we tailor the alternating RQ conditions defined in [11], [12] to our clocked communication-free Petri net model. Our results are summarized in Table 1.

2. Definitions

Let Z (resp., N and R) denote the set of integers (resp., nonnegative integers and reals). A *Petri net* (PN) is a triple (P, T, φ), where P is a finite set of *places*, T is a finite set of *transitions*, and φ is a *flow function* $\varphi : (P \times T) \cup (T \times P) \rightarrow N$. A *marking* (or *configuration*) is a mapping $\mu : P \rightarrow N$. A transition $t \in T$ is *enabled* at a marking μ iff for every $p \in P$, $\varphi(p, t) \leq \mu(p)$. A transition t may *fire* at a marking μ if t is enabled at μ. We then write $\mu \xrightarrow{t} \mu'$, where $\mu'(p) = \mu(p) - \varphi(p, t) + \varphi(t, p)$ for all $p \in P$. A sequence of transitions $\sigma = t_1 \ldots t_n$ is a *firing sequence* from μ_0 iff $\mu_0 \xrightarrow{t_1} \mu_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} \mu_n$ for some sequence of markings $\mu_1, \ldots, \mu_n$. (We also write '$\mu_0 \xrightarrow{\sigma} \mu_n$'.) We write '$\mu_0 \xrightarrow{\sigma}$' to denote that σ is enabled and can be fired from μ_0, i.e., $\mu_0 \xrightarrow{\sigma}$ iff there exists a marking μ such that $\mu_0 \xrightarrow{\sigma} \mu$. A *marked* PN is a pair $((P, T, \varphi), \mu_0)$, where (P, T, φ) is a PN, and μ_0 is a marking called the *initial marking*. Let $\mathcal{P} = ((P, T, \varphi), \mu_0)$ be a marked PN. The *reachability set* of $\mathcal{P}$ is the set $\mathrm{R}(\mathcal{P}) = \{\mu \mid \mu_0 \xrightarrow{\sigma} \mu$ for some $\sigma \in T^*\}$. The *reachability problem* is that of determining, given a marked PN $\mathcal{P}$ and a marking μ, whether $\mu \in \mathrm{R}(\mathcal{P})$.

For ease of expression, the following notations will be used extensively throughout the rest of this paper. (Let σ be a transition sequence, p be a place, and t be a transition.)

- $\#_\sigma(t)$ represents the number of occurrences of t in σ.
- $\Delta(\sigma)$: the *displacement* of σ is defined as $\Delta(\sigma) = \mu' - \mu$, provided that $\mu \xrightarrow{\sigma} \mu'$.
- $Tr(\sigma) = \{t | t \in T, \#_\sigma(t) > 0\}$, denoting the set of transitions used in σ.
- $p^\bullet = \{t | \varphi(p, t) \geq 1, t \in T\}$; $t^\bullet = \{p | \varphi(t, p) \geq 1, p \in P\}$; ${}^\bullet p = \{t | \varphi(t, p) \geq 1, t \in T\}$; ${}^\bullet t = \{p | \varphi(p, t) \geq 1, p \in P\}$.

Given a computation $\mu \xrightarrow{\sigma} \mu'$, a sequence σ' is said to be a *rearrangement* of σ if $\#_\sigma = \#_{\sigma'}$ and $\mu \xrightarrow{\sigma'} \mu'$. Given a directed graph $G = (V, E)$, we write $u \hookrightarrow v$ to denote a path from nodes u to v.

A marked PN $((P, T, \varphi), \mu_0)$ is said to be *communication-free* (abbreviated as *cf*) [5] if

(1). $\forall t \in T, |{}^\bullet t| = 1$, (i.e., every transition has exactly one input place), and
(2). $\forall p \in P, t \in T, \varphi(p, t) \leq 1$ (i.e., every arc going from a place to a transition has weight 1).

3. Variants of Communication-Free Petri Nets

In this section we recall the definitions of variants of cf-PNs. The cf-PNs is special class of Petri nets in which each transition has exactly one input place, and the firing of a transition removes exactly one token from its input place. A motivation comes from cf-PNs exhibit a relatively low complexity from an algorithmic viewpoint, as opposed to general Petri nets. It is worth to take a look and see how the complexity of checking complexity is affected when such Petri nets are extended with the notations of priority, time, among others.

- **State-extended cf-PNs**

A *state-extended cf-PN* is a $(S, s_0, (P, T, \varphi), \delta)$, where S is a finite set of states, $s_0 \in S$ is the initial state, (P, T, φ) is a cf-PN, and $\delta \subseteq S \times S \times T$ defines the transition relation. A *configuration* is a pair (p, x), where p is in S and x is a vector in N^k. (s_0, v_0) is the *initial configuration*, where v_0 is the initial marking. The transition (p, q, t) can be applied to the configuration (p, v) and yields the configuration $(q, v + \Delta(t))$, provided that t is fireable in marking v in PN (P, T, φ). In this case, $(q, v + \Delta(t))$ is said to *follow* (p, v). Let σ_0 and σ_t be two configurations. Then σ_t is said to be *reachable* from σ_0 iff $\sigma_0 = \sigma_t$ or there exist configurations $\sigma_1, \cdots, \sigma_{t-1}$ such that σ_{r+1} follows σ_r for $r = 0, \ldots, t-1$.

Some equivalence issues of state-extended BPP (equivalently, state-extended cf-PNs) have been studied in, e.g., [9], [10]. In particular, bisimulation equivalence has been shown to be undecidable for state-extended BPP.

• Priority cf-PNs

A *static-priority PN* is a tuple $((P, T, \varphi), \rho)$, where (P, T, φ) is a PN, and ρ ($\subseteq T \times T$) defines the so-called *priority relation* over T. Intuitively, $(t_1, t_2) \in \rho$ means that t_2 takes precedence over t_1 in transition firing. Let $\bar{\rho}$ denote $\{(t, t')|(t, t') \notin \rho \text{ and } (t', t) \notin \rho\}$ (i.e., $\bar{\rho}$ is the set of pairs of incomparable transitions). In this paper (so is in [2]), ρ is assumed to be irreflexive, asymmetric, and transitive, and $\bar{\rho}$ be an equivalence relation. Given a ρ, transition t is *ρ-enabled* at μ iff t is enabled at μ and for every t' ($\in T$) enabled at μ, $(t, t') \notin \rho$ (i.e., t is of the highest priority among those enabled at μ). In a static-priority PN $((P, T, \varphi), \rho)$, a transition t may *fire* at a marking μ if t is ρ-enabled at μ; we then write $\mu \overset{t}{\Rightarrow} \mu'$, where $\mu'(p) = \mu(p) - \varphi(p, t) + \varphi(t, p)$ for all $p \in P$. By replacing $\rightarrow$ (defined for conventional PNs in Sect. 2) with $\Longrightarrow$, the definitions of *firing sequences* and *reachable markings* apply to static-priority PNs as well. Since $\bar{\rho}$ is assumed to be an equivalence relation, $\bar{\rho}$ partitions T into a number of *equivalence classes* $\Omega_1, \ldots, \Omega_d$, for some d ($d \leq |T|$). Intuitively, each Ω_i, $1 \leq i \leq d$, represents a set of transitions having the same priority. Since ρ is also assumed to be irreflexive, asymmetric, and transitive, for every $t \in \Omega_i$ and $t' \in \Omega_j$ (where $1 \leq i, j \leq d$ and $i \neq j$), either $(t, t') \in \rho$ or $(t', t) \in \rho$ (but not both); we write $\Omega_i < \Omega_j$ (resp., $\Omega_j < \Omega_i$) if $(t, t') \in \rho$ (resp., $(t', t) \in \rho$). Without loss of generality, we assume that $\Omega_1, \ldots, \Omega_d$ be enumerated in increasing priority throughout the rest of this paper. Given a $t \in T$ and a ρ, we let $class(t) = i$ if $t \in \Omega_i$ (i.e., $class(t)$ is the index of the equivalence class containing t).

A *dynamic-priority PN* is a tuple $((P, T, \varphi), \rho)$, where (P, T, φ) is a PN, and $\rho : N^k \rightarrow 2^{T \times T}$ defines the so-called *dynamic priority relation* over T. The main difference between static and dynamic priorities is that the priority relation is fixed for the former, whereas it is a function of the marking for the latter. More precisely, for a marking μ, $\rho(\mu)$ ($\subseteq (T \times T)$) defines the priority relation at marking μ. For dynamic-priority PNs, the notions of enabledness as well as reachability are analogous to that in the static priority case. See [3] for more about PNs with dynamic priorities.

For cf-PNs, it is assumed that for arbitrary transitions t and t', if ${}^\bullet t = {}^\bullet t'$ (i.e., they share the same input place), then t and t' have the same priority. (If this is not the case, the lower priority one can never be fired.)

• Cf-PNs with inhibitor arcs

A PN with *inhibitor arcs* is a tuple (P, T, φ, I), where P, T, and φ are the same as that in conventional PNs defined earlier, and $I \subseteq P \times T$ defines the set of *inhibitor arcs*. (We assume $(p, t) \in I \Longrightarrow \varphi(p, t) = 0$.) A transition t may fire at a marking μ if for every $(p, t) \in \varphi$, $\varphi(p, t) \leq \mu(p)$ and for every $(p, t) \in I$, $\mu(p) = 0$. For convenience, a transition t is said to be an *o-transition* if $(p, t) \in I$, for some place p. As its name suggests, a *cf-PN with inhibitor arcs* is a PN with inhibitor arcs such that the structure of the PN is communication-free.

• Clocked cf-PNs

Given a set $X = \{x_1, x_2, \ldots, x_n\}$ of *clock variables*, the set $\Phi(X)$ of *clock constraints* δ is defined inductively by

$$\delta := x \leq c \mid c \leq x \mid \neg\delta \mid \delta \wedge \delta,$$

where x is a clock in X and c is a constant in $\mathbf{N}$. A *clock reading* is a mapping $\nu : X \rightarrow \mathbf{R}$ which assigns each clock a real value. For $\eta \in \mathbf{R}$, we write $\nu + \eta$ to denote the clock reading which maps every clock x to the value $\nu(x) + \eta$. A clock reading ν for X *satisfies* a clock constraint δ over X, denoted by $\delta(\nu) \equiv$ **true** , iff δ evaluates to **true** using the values given by ν.

A *clocked PN* is a tuple $\mathcal{N} = ((P, T, \varphi), X, r, q)$, where (P, T, φ) is a PN, X is a finite set of real-value clock variables, $r : T \longrightarrow 2^X$ is a labeling function assigning clocks to transitions, and $q : T \longrightarrow \Phi(X) \cup \{\lambda\}$ is a labeling function assigning clock constraints to transitions. Intuitively, $r(t)$ contains those clock variables which are reset when transition t is fired, and $q(t)$ is a 'guard' governing the condition under which the clock reading must meet in order for t to fire. Notice that when $r(t) = \emptyset$ and $q(t) = \lambda$, transition t behaves just like a transition in an ordinary PN. A transition is said to be *clocked* it is associated with either resets or queries. In our subsequent discussion, when a transition t in a PN graph is annotated by $R(x)$, it means $x \in r(t)$. A *configuration* of a clocked PN consists of a *marking* μ, the *global time* η and the present *clock reading* ν. We use the 3-tuple (μ, η, ν) to denote such a configuration. Note that the clock reading ν is continuously being updated as η, the global time, advances. Hence, ν and η are not completely independent. Given a configuration (μ, η, ν) of a clocked PN $\mathcal{P}$, a transition t is *enabled* iff $\mu \xrightarrow{t}$, and $q(t)(\nu) \equiv$ **true** . Let μ be the marking and ν the clock reading at time η. Then t *may* fire at η if t is enabled in the marking μ with the clock reading ν. We then write $(\mu, \nu) \overset{(t,\eta)}{\rightarrow} (\mu', \nu')$, where $\mu' = \mu + \Delta(t)$, $\nu'(x) = 0$ (for all $x \in r(t)$) and $\nu'(x) = \nu(x)$ (for all $x \notin r(t)$). Note that the global time remains unchanged as a result of firing t. That is, the firing of a transition is assumed to let no time elapse at all. The global clock will start moving immediately after the firing of a transition is completed. A *marked clocked PN* is a pair $(\mathcal{N}, \mu_0)$, where $\mathcal{N}$ is a clocked PN and μ_0 is the initial marking. Initially, we assume the initial global time η_0 and clock reading ν_0 to be $\eta_0 = 0$ and $\nu_0(x) = 0$ ($\forall x \in X$), respectively.

It is important to point out that in the execution semantics defined above, *enabledness* is necessary but not sufficient for transition firing. Unlike the case in the strong firing semantics of *timed* (or *time*) PNs ([16]), it is not required to fire all the enabled transitions at any point in time during the course of a computation. The *reachability problem* for clocked PNs is to decide, given a marked clocked PN $(\mathcal{N}, \mu_0)$ and a marking μ, deciding whether $\mu_0 \overset{\omega}{\rightarrow} \mu$, for some $\omega = (\sigma, \tau)$, where σ and τ are transition and time sequences, respectively.

4. Complexity and Decidability Analysis of the Reachability Problem

4.1 Priority Communication-Free Petri Nets

A 2-counter machine consists of a finite-state control equipped with two *counters*, in which a transition from state p to state q (operating on counter C) is of one of the following three: *increment* ($p \xrightarrow{C+} q$), *decrement* ($p \xrightarrow{C-} q$), and *test-for-zero* ($p \xrightarrow{C=0} q$). *2-counter machines* are computationally equivalent to Turing machines.

Theorem 4.1: The reachability problem for dynamic-priority cf-PNs is undecidable.

Proof: It suffices to show that given a 2-counter machine M, a dynamic-priority cf-PN $\mathcal{P}$ and a marking μ can be constructed in such a way that μ is reachable in $\mathcal{P}$ iff M halts.

Consider $M's$ transitions $p \xrightarrow{C-} q$ and $p \xrightarrow{C=0} q'$, we use the cf-PN fragment shown in Fig. 1 to simulate these two transitions. Here place c simulates Counter C and places a, d, f simulate states p, q', q, respectively. (The simulation of $p \xrightarrow{C+} q$ is rather straightforward, and hence, is left to the reader.) Before the simulation begins, it is the case that $\mu(a) = 1$ (indicating p being the current state), $\mu(c)$ equals the counter $C's$ value, and the remaining places are empty. In our setting, only two priority classes, namely, $\{High, Low\}$, are needed to make the simulation work. More precisely,

(1) $High = \{t_1\}$ and $Low = \{t_2, t_3, t_4, t_5\}$, when $\mu(a) = 1 \wedge \mu(c) = 0$,
(2) $High = \{t_2\}$ and $Low = \{t_1, t_3, t_4, t_5\}$, when $\mu(a) = 1 \wedge \mu(c) > 0 \wedge \mu(e) = 0$,
(3) $High = \{t_3\}$ and $Low = \{t_1, t_2, t_4, t_5\}$, when $\mu(b) = 1 \wedge \mu(c) > 0 \wedge \mu(e) = 0$,
(4) $High = \{t_4\}$ and $Low = \{t_1, t_2, t_3, t_5\}$, when $\mu(b) = 1 \wedge \mu(e) = 1$,
(5) $High = \{t_5\}$ and $Low = \{t_1, t_2, t_3, t_4\}$, when $\mu(b) = 0 \wedge \mu(e) = 1$ and
(6) the remaining transitions belong to class 'Low'.

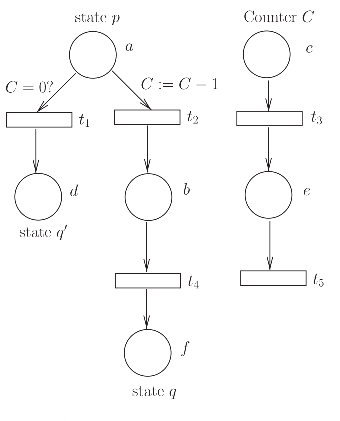

Fig. 1 Simulation of a 2-counter machine using a dynamic-priority cf-PN.

$p \xrightarrow{C=0} q'$ is simulated through the firing of transition t_1, whose feasibility is guaranteed by Case (1) above. $p \xrightarrow{C-} q$, on the other hand, requires the execution of a sequence of transitions $t_2t_3t_4t_5$, which is feasible as the result of the priorities assigned in Cases (2)–(5) above. ■

By reducing from the halting problem of 2-counter machines, the following is not difficult to show.

Theorem 4.2: The reachability problem is undecidable for state-extended cf-PNs with static priorities.

For cf-PNs with static priorities, we are able to show the reachability problem to be NP-complete. We require the following lemma, whose proof is similar in style to a result in [2] concerning the so-called EQUAL-conflict *P/T nets*.

Lemma 4.3: Given a cf-PN (P, T, φ) and a priority relation ρ, if $\mu \xrightarrow{\sigma} \mu'$ and for every t enabled at μ', $t \in \Omega_1$ (i.e., t is in the lowest priority class induced by $\bar{\rho}$), then $\mu \overset{\sigma'}{\Rightarrow} \mu'$, for some permutation σ' of σ.

Proof: Consider a path from μ to μ' along which μ_1 is the leftmost marking at which the priority requirement is violated, and t_1 be the transition fired at μ_1. We claim that one of the highest priority transitions enabled at μ_1 must fire in between μ_1 and μ', for only transitions belonging to the class Ω_1 are enabled at μ'. We let $\mu_2 \xrightarrow{t_2}$ be the marking and its associated transition such that μ_2 is the nearest location (following μ_1) with t_2 among the highest priority transitions enabled at μ_1. (More precisely, every transition occurring between μ_1 and μ_2 has its priority lower than t_2.) Since t_1 and t_2 (which have different priorities) do not share a common input place, firing t_2 at μ_1 followed by t_1 remains a valid path. By repeatedly applying such a rearrangement to the remaining path, a path meeting the priority requirement can be constructed. ■

The following lemma (shown in [17]) suggests that the reachability problem for cf-PNs can be characterized as an *Integer Linear Programming* problem.

Lemma 4.4: (from Theorem 4 in [17]) Given a cf-PN $\mathcal{P} = (P, T, \varphi)$, a system of linear inequalities $ILP(\mathcal{P}, \mu, \mu')$ can be constructed in NP in such a way that μ' is reachable from μ iff $ILP(\mathcal{P}, \mu, \mu')$ has a solution over the integers.

With the above two lemmas, we are ready to derive the complexity of the reachability problem for static-priority cf-PNs. The proof closely parallels that of a recent result concerning priority conflict-free PNs [18].

Theorem 4.5: The reachability problem for static-priority cf-PNs is NP-complete.

4.2 Communication-Free Petri Nets with Inhibitor Arcs

A cf-PN with inhibitor arcs $\mathcal{P} = (P, T, \varphi, I)$ is said to be *cycle-free* if it is the case that for every cycle l (i.e., a closed loop) in the PN graph of $\mathcal{P}$, there exists a transition t along l such that the following condition holds:

$$\text{(Condition A):} \\ \forall p \in t^{\bullet}, \forall t' \in T, (p, t') \notin I$$

(i.e., the firing of t does not deposit tokens to a place from which an inhibitor arc emanates).

To analyze the reachability problem of cf-PNs with cycle-free inhibitor arcs, we require the following lemma, which demonstrates the possibility of rearranging a path into some sort of a 'canonical form'.

Lemma 4.6: Given a cf-PN with cycle-free inhibitor arcs $\mathcal{P} = (P, T, \varphi, I)$, suppose $\mu \overset{\sigma_1 t \sigma_2}{\rightarrow} \mu'$ is a computation in $\mathcal{P}$ such that t is an o-transition, $t \notin Tr(\sigma_1)$, and t satisfies Condition A above, then σ can be rearranged into $\sigma_1 \overbrace{t \cdots t}^{i} \sigma_3$, for some i, such that $t \notin Tr(\sigma_3)$. (That is, all the occurrences of t in σ can be fired all at once.)

Proof: Since none of t's output places is the input of an o-transition (Condition A), firing t earlier (compared to the original firing position) does not affect the fireability of the remaining transitions. The rest is then clear. ■

By repeatedly applying the above lemma, we have the following:

Lemma 4.7: Let $\mathcal{P} = (P, T, \varphi, I)$ be a cf-PN with cycle-free inhibitor arcs, and $\mu \overset{\sigma}{\rightarrow} \mu'$ be a computation in $\mathcal{P}$. The σ can be rearranged into $\sigma_1 \overbrace{t_1 \cdots t_1}^{n_1} \sigma_2 \overbrace{t_2 \cdots t_2}^{n_2} \cdots \sigma_m \overbrace{t_m \cdots t_m}^{n_m} \sigma_{m+1}$ such that

1. $\forall 1 \le i \le m$, t_i is an o-transition,
2. $\forall 1 \le i \le m + 1$, none of the transitions in σ_i is an o-transition, and
3. $m \le 2^k$, where k denotes the number of o-transitions in T.

(Notice that $t_i, 1 \le i \le m$, do not have to be distinct.)

Proof: According to Lemma 4.6, any computation $\mu \overset{\sigma}{\rightarrow} \mu'$ can be rearranged into $\mu \overset{\sigma_1 \overbrace{t \cdots t}^{i} \sigma_3}{\longrightarrow}$, for some o-transiton $t \notin Tr(\sigma_1) \cup Tr(\sigma_3)$. As a result, the number of o-transitions in σ_1 and σ_3, respectively, is at most $k - 1$. By recursively applying Lemma 4.6 to σ_1 and σ_3 (with respect to PN $(P, T - \{t\}, \varphi|_{T-\{t\}}, I|_{T-\{t\}})$), our lemma follows. ■

Theorem 4.8: The reachability problem for cf-PNs with cycle-free inhibitor arcs is solvable in NEXPTIME.

Proof: From Lemma 4.7, any computation $\mu \overset{\sigma}{\rightarrow} \mu'$ can be rearranged into $\mu \overset{\sigma_1}{\rightarrow} \mu_1 \overset{\overbrace{t_1 \cdots t_1}^{n_1}}{\longrightarrow} \mu'_1 \overset{\sigma_2}{\rightarrow} \mu_2 \overset{\overbrace{t_2 \cdots t_2}^{n_2}}{\longrightarrow} \mu'_2 \cdots \overset{\sigma_m}{\rightarrow} \mu_m \overset{\overbrace{t_m \cdots t_m}^{n_m}}{\longrightarrow} \mu'_m \overset{\sigma_{m+1}}{\rightarrow} \mu'$, for some markings $\mu_1, \mu'_1, \ldots, \mu_m, \mu'_m$, and positive integers $n_1, \ldots, n_m$. As each $\sigma_i, 1 \le i \le m$, does not contain any o-transitions, its computation can be captured by an instance of integer linear programming, as Lemma 4.4 suggests. By 'guessing' those o-transitions $t_1, t_2, \ldots, t_m$, checking whether μ' is reachable from μ in $\mathcal{P}$ can be formulated as

$$\begin{cases} x_0 = \mu \\ \forall 1 \le i \le m + 1 \\ \qquad ILP((P, T', \varphi), x_{i-1}, x_i) \\ \qquad x_i({}^{\bullet}t_i) = 0 \\ \qquad x'_i = x_i + n_i \Delta(t_i) \ge \mathbf{0} \\ x_{m+1} = \mu \end{cases}$$

where T' is the set $\{t \mid t \in T, t \text{ is not an o-transition}\}$, vector variables x_0 and x_{m+1} represent markings μ and μ', respectively, $x_i, x'_i, 1 \le i \le m$, are vector variables representing markings μ_i, μ'_i, respectively, and $n_i, 1 \le i \le m$, are scalar variables. As m is bounded by an exponential function in the size of $\mathcal{P}$, the above instance of integer linear programming is solvable in NEXPTIME. ■

The reachability problem for cf-PNs with cycle-free inhibitor arcs is clearly NP-hard, as the problem for original cf-PNs is NP-complete. It would be interesting to see whether the lower bound can be improved.

By reducing from the halting problem of 2-counter machines, we can show the following:

Theorem 4.9: The reachability problem is undecidable for state-extended cf-PNs with cycle-free inhibitor arcs.

4.3 Alternating RQ Communication-Free Petri Nets

For general clocked PNs, we are able to show the following result using a standard technique based on the concept of *clock regions* [1]. The idea is to reduce the reachability problem for clocked Petri nets to that for *vector addition systems with states* (by associating each clock region with a state), which are known to be computationally equivalent to Petri nets.

Theorem 4.10: The reachability problem is decidable for general clocked PNs.

In view of the above, together with the fact that checking reachability in cf-PNs is easier to solve than in general PNs (the former is in NP whereas the latter is EXPSPACE-hard), a natural question to ask is how difficult the reachability problem is for clocked cf-PNs. As one might expect, the property of 'communication-freedom' is crucial in making cf-PNs easier to analyze (from a computational complexity viewpoint) than their general counterparts. The introduction of timing constraints to cf-PNs, however, renders (in an implicit way) the net not communication-free behaviorally. In fact, for static-priority cf-PNs with clocks, the following result shows the reachability problem to be undecidable. (Recall that for static-priority cf-PNs without clocks, the problem was shown to be solvable in NP in the previous section.)

Theorem 4.11: The reachability problem is undecidable for static-priority clocked cf-PNs.

Proof: Following Theorem 4.2, it suffices to show that any static-priority state-extended cf-PN can be simulated by a static-priority clocked cf-PN. The idea of the simulation is depicted in Fig. 2. Figure 2 (1) shows a fragment of a state-extended cf-PN in which the firing of PN transition t is controlled by $p \rightarrow q$. In our construction we associate two distinct clocks, namely x and y in Fig. 2 (2), with each transition t. Clocks x and y are used to 'synchronize' the actions between 'moving a token from places p to q' (which corresponds to the 'state' portion of the state-extended cf-PN) and firing transition t itself. We assume the clock values of x and y to be greater than 1 initially; hence, they have to be reset before t becomes fireable. (This ensures that only the transition associated with the 'current state', namely, p, has the right to fire.) It is easy to see that the sequence utv is fireable. In addition, upon firing v, the values of x and y are 2 and 3, respectively, satisfying our earlier assumption that the values of x and y are greater than 1. ■

In view of the above and Theorem 4.5, cf-PNs with clocks seem to be harder to analyze than their unclocked counterparts. As a first step towards a complexity analysis of clocked cf-PNs, in what follows we lower our expectations by considering a restricted class of clocked cf-PNs called *alternating RQ cf-PNs*. As the structure of the so-called *circuits* (or *cycles*) in the PN graph plays an important role in our subsequent discussion, some definitions are needed first.

A *circuit* of a PN is a 'simple' closed path in the PN graph. (By 'simple' we mean all nodes are distinct along the closed path.) It is important to note that every circuit $c = p_1t_1p_2t_2 \cdots p_nt_np_1$ in a cf-PN must have ${}^\bullet t_i = \{p_i\}$, for every $i, 1 \leq i \leq n$. (Notice that the firing of a transition may deposit more than one token into a place. Given a circuit $c = p_1t_1p_2t_2 \cdots p_nt_np_1$, let $P_c = \{p_1, p_2, \cdots, p_n\}$ denote the set of places in c. We define the token count of circuit c in marking μ to be $\mu(c) = \sum_{p \in P_c} \mu(p)$. A circuit c is said to be *token-free* in μ iff $\mu(c) = 0$. A set of circuits $C = \{c_1, c_2, \ldots, c_n\}$ is said to be *a circuit collection* iff for every i, j, $1 \leq i, j \leq n$, there exist $1 \leq h_1, h_2, \ldots, h_r \leq n$, for some r, such that $h_1 = i$, $h_r = j$, and for every $1 \leq l < r$, $P_{c_{h_l}} \cap P_{c_{h_{l+1}}} \neq \emptyset$. In words, every pair of neighboring circuits in the sequence $c_{h_1}, c_{h_2}, \ldots, c_{h_r}$ share at least one place. For a simple circuit c, we also use $\#_c$ to denote the vector count of transitions used in c, i.e., $\#_c(i) = 1$ if t_i is in c; $\#_c(i) = 0$, otherwise. A circuit is called a *clocked circuit* if it contains at least one clocked transition. A sequence σ is said to *cover* circuit c if $\#_c \leq \#_\sigma$, i.e., every transition of c appears in σ.

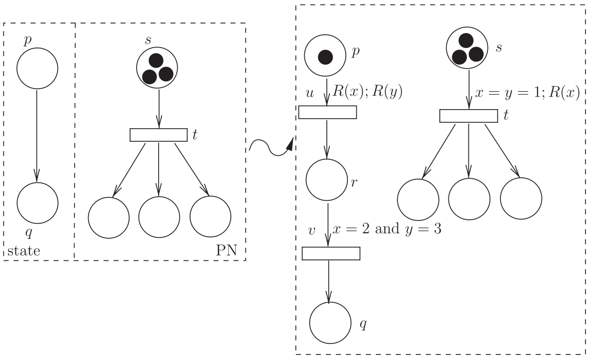

Fig. 2 Transformation of a state-extended cf-PN into an equivalent clocked cf-PN.

Our definition of *alternating RQ cf-PNs* is mainly motivated by the work of [11] in which the so-called *alternating RQ timed automata* were defined. Intuitively, a timed automaton is said to be an *alternating RQ timed automaton* if given an arbitrary computation σ and a clock x, the sequence of 'resets' and 'queries' regarding x must appear alternatively. As it turns out, alternating RQ timed automata admit more efficient verification algorithms, in comparison with that for the general model of timed automata [11] (see also [12]). What makes alternating RQ timed automata easier to analyze, compared to their general counterparts, seems to lie in the so-called *simple path property*, suggesting that if a state q is reachable from p in a timed automaton, then q can be reached from p through a path without cycles, and if a cycle is traversable once, it can be traversed an arbitrary (finite of infinite) number of times. The intuition behind the justification of imposing the alternating RQ constraints is that, in many cases, if we want to inquire about a timing status at a certain point during a computation via a query, we often reset the associated clock before the query. Furthermore, if the same clock is to be inquired twice, different clocks can be used for each query. As indicated in [11], [12], many practical examples in the literature meet the alternating RQ constraints. Inspired by [4], [11], [12], in our subsequent discussion we tailor the alternating RQ conditions defined in [11], [12] to our clocked PN model.

Given a sequence σ of transitions, the *RQ sequence* of σ, denoted by $\Gamma(\sigma)$, is the sequence of resets and queries encountered along σ. Given an RQ sequence $\Gamma(\sigma)$, the RQ sequence with respect to clock x, denoted by $\Gamma(\sigma)|_x$, is obtained from $\Gamma(\sigma)$ by deleting all the resets and queries that do not involve x. An RQ sequence $\Gamma(\sigma)$ is *alternating*, if, for each clock x, $\Gamma|_x$ is of the form $R(x)Q(x)R(x)Q(x) \cdots$, where $R(x)$ and $Q(x)$ denote reset and query operations, respectively, with respect to clock x.

An *alternating RQ cf-PN* is a clocked cf-PN satisfying the following constraints:

(1) For each clock x, there is only one pair of reset and query associated with x in the PN.
(2) For each computation $\omega = (\sigma, \tau)$ starting from the initial configuration, $\Gamma(\sigma)$ is *alternating*, i.e., for each x, resets and queries in $\Gamma(\sigma)|_x$ appear in an alternating fashion.
(3) Suppose (μ, η, ν) is a reachable configuration with $\mu(p_1) \geq 1$ and $p_1t_1p_2t_2 \cdots p_nt_n$ is an arbitrary simple path in the PN graph. Then there exists a time sequence τ such that $(\mu, \eta, \nu) \overset{(t_1t_2\cdots t_n, \tau)}{\rightarrow}$ is fireable.
(4) Each strongly connected component in the PN graph

contains at most one clocked circuit.

The reason why multiple clocked circuits are disallowed (as Condition (4) indicates) is that the time delay induced by traversing a clocked circuit might invalidate the clock constraints associated with another clocked circuit (more precisely, the number of times a clocked circuit can be traversed hinders on another clocked circuit) — violating the *simple path property* mentioned earlier. As a consequence, at the current stage of our investigation we only consider cf-PNs which exhibit 'simple' circuit structures, although we surmise that a more general condition also works. Although not discussed in this paper, it is worthy of investigating the complexity of deciding whether a given PN is an alternating RQ cf-PN or not.

Given Condition (1) in the above definition, for convenience we define two mappings $R : X \rightarrow T$ and $Q : X \rightarrow T$ which indicate the transitions at which resets and queries, respectively, reside. $R(x) = t$ indicates that clock x is reset whenever t is fired. Likewise, $Q(x)$ represents the transition at which a query concerning clock x resides. A transition (resp., place) is said to be *timed* if and only if there exists a path in the PN graph from the transition (resp., place) to a clocked transition. Let T_{timed} and P_{timed} be the sets of timed transitions and places, respectively, and $T_{untimed} = T - T_{timed}$, $P_{untimed} = P - P_{timed}$. The following result shows an important feature regarding the behavior of alternating RQ cf-PNs.

Lemma 4.12: *Let $\mathcal{P} = (\mathcal{N}, \mu_0)$ be an alternating RQ cf-PN. If $p_1 \hookrightarrow p_2 \hookrightarrow t$ is a path in the PN graph and t is clocked, then $\mu(p_1) + \mu(p_2) \leq 1$, for every reachable configuration (μ, η, ν). (Here $p_1 \hookrightarrow p_2$ can be null; in this case, $p_1 = p_2$.)*

Proof: Suppose t is the first clocked transition along the path $\hookrightarrow p_2 \hookrightarrow t$. Let the transition sequences (excluding t) in $p_1 \hookrightarrow p_2$ and $p_2 \hookrightarrow t$ be σ_1 and σ_2, respectively. Consider the following cases:

(Case 1): $\mu(p_2) \geq 2$. Due to the property of being an alternating RQ cf-PN, there exists a time sequence η such that $\mu \xrightarrow{(\sigma_2 t, \eta)} \mu'$, for some μ'. As none of the transitions in σ_2 is clocked, $\mu \xrightarrow{(\sigma_2 t, \eta)\|(\sigma_2 t, \eta)} \mu''$, for some μ'' — violating the RQ alternating property as t being fired twice consecutively.

(Case 2): $\mu(p_1) \geq 1$ and $\mu(p_2) \geq 1$. In this case there exist computations $\mu \xrightarrow{(\sigma_1 \sigma_2 t, \eta)} \mu_1$, $\mu \xrightarrow{(\sigma_2 t, \eta')} \mu_2$, for some μ_1 and μ_2. It then follows that $\mu \xrightarrow{(\sigma_1 \sigma_2 t, \eta)\|(\sigma_2 t, \eta')} \mu_3$, for some μ_3 — violating the RQ alternating property as t being fired twice consecutively.

(Case 3): $\mu(p_1) \geq 2$. Similar to Case 1.

If t is not the first clocked transition in $p_2 \hookrightarrow t$, choose the first such transition, and then apply the above argument. ■

To simplify our subsequent discussion, we require the following result whose proof is not difficult.

Lemma 4.13: *Given an arbitrary alternating RQ cf-PN and a marking μ, we can construct in polynomial time an 'equivalent' (as far as reaching μ is concerned) alternating RQ cf-PN $\mathcal{P}$ satisfying the following two properties:*

(1) $\varphi(t, p) \leq 1$, $\forall\, t \in T_{timed}$ (i.e., firing any timed transition puts only one token into each of its output places),

(2) for every $p \in P_{untimed}$, there is at most one timed transition in ${}^{\bullet}p$.

It is clear from the definitions of timed and untimed transitions and places that in any circuit c, if there is an untimed transition or place, then all the transitions and places in c must be untimed as well. Such an observation applies to circuit collections. Hence, a circuit c is called a *timed circuit* of $\mathcal{P}$ if $\forall\, p \in c$ and $t \in c$, $p \in P_{timed}$ and $t \in T_{timed}$. A circuit collection S is called a *timed circuit collection* if for every $c \in S$, c is a timed circuit. Note that a clocked circuit is also a timed circuit, but the converse does not necessarily hold.

Let $\mathcal{N} = ((P, T, \varphi), X, r, q)$ be an alternating RQ cf-PN. Suppose φ_t and φ_u are the restrictions of φ to timed and untimed transitions, respectively. The *timed subnet* of $\mathcal{N}$, denoted by $\mathcal{N}_T$, is also an alternating RQ cf-PN where $\mathcal{N}_T = ((P, T_{timed}, \varphi_t), X, r, q)$. The *untimed subnet* of $\mathcal{N}$, denoted by $\mathcal{N}_U$, is a cf-PN where $\mathcal{N}_U = (P, T_{untimed}, \varphi_u)$.

Our strategy of showing the reachability problem for alternating RQ cf-PNs to be in NP is, given an alternating RQ cf-PN $\mathcal{P}$ and a marking μ, to construct a system of linear inequalities $\mathcal{L}$ in such a way that μ is reachable in $\mathcal{P}$ iff $\mathcal{L}$ has an integer solution. As *integer linear programming* is known to be solvable in NP, our result follows.

To give the reader a better feel for how the main proof goes, in what follows we first present the key idea in a high-level and intuitive fashion. The details will be filled in as our discussion progresses.

To begin with, we first guess the set of transitions (possibly) witnessing reachability. (Note that we have the luxury of doing so (i.e., guessing) since NP is what we are aiming for.) Our next step towards the complexity analysis is to further divide the (guessed) structure of the net into *timed* and *untimed* subnets in a way described in the previous section. For tokens in the timed subnet, their behaviors are restricted by the timing constraints. On the other hand, tokens can move more freely if they are in the untimed subnet. Such a disparity in token behavior motivates a lemma (namely, Lemma 4.14, which will be proven later) which proves the legitimacy of separating the computation of the two subnets when dealing with the reachability problem. More precisely, we are able to show that any computation $\mu_0 \xrightarrow{\sigma} \mu$ in $\mathcal{P}$ can be rearranged into $\mu_0 \xrightarrow{(\sigma_1, \tau)} \mu' \xrightarrow{\sigma_2} \mu$, for some μ' and time sequence τ, so that σ_1 (resp., σ_2) uses only timed (resp., untimed) transitions. Following Lemma 4.4, a system of linear inequalities can be set up for $\mu' \xrightarrow{\sigma_2} \mu$, since σ_2 involves only untimed transitions. (Note that in such a system of linear inequalities, μ' represents a vector variable.)

It remains to show how a similar (albeit much more complicated) system of linear inequalities can be derived for $\mu_0 \xrightarrow{(\sigma_1, \tau)} \mu'$. To this end, we argue that any 'timed' compu-

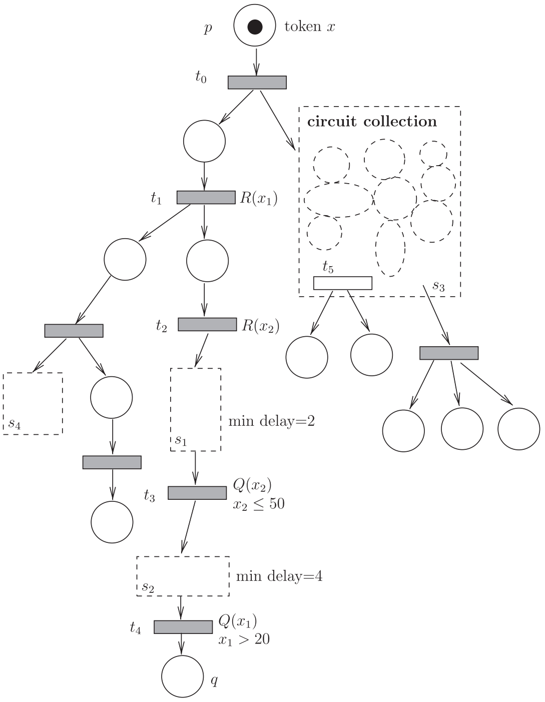

Fig. 3 Computational structure of an alternating RQ cf-PN.

tation in an alternating RQ cf-PN must behave in a rather regular fashion. It is important to point out that such a regularity is exhibited in the PN not only structurally but also behaviorally. Take Fig. 3 as an example for illustrating what we mean by 'regularity.' Figure 3 represents the life cycle of a token, say, x, in existence initially, along with those tokens generated by x directly or indirectly as x progresses. In a graph-theoretic sense, the portion of the PN involved in the course of token x's computation consists of circuit collections (see S_1 and S_2 in Fig. 3) interleaved by line segments (see transitions t_1 - t_4 in Fig. 3). In a line segment, token x can only move downwards, while possibly generating new token (to either timed or untimed portions of the PN) as it progresses. Once x moves inside a circuit collection, it is entitled to circle around an arbitrary number of times (provided the timing requirement is also met) while spinning off new tokens. There are several key features (which will be proven later) regarding a circuit collection:

- At any instant, at most one token is present in a circuit collection.
- Tokens can only be injected into untimed places as a result of spin-off (see t_5 in Fig. 3).
- To exit from a circuit collection, a token must leave from a unique exit place.
- Tokens in the timed portion can never interfere with one another.

With the exception of generating new tokens (or being generated by another token), a token, during its entire life span, never interferes with the rest of the tokens in existence in the PN.

Enough for the intuition, we are in a position to embark on the derivation of the NP upper bound. To do so, a few lemmas are in order.

Lemma 4.14: *Let $\mathcal{P} = ((P, T, \varphi), X, r, q)$ be an alternating RQ cf-PN with initial marking μ_0. For every reachable marking μ of $\mathcal{P}$, there exists a 'canonical' computation $\omega = (\sigma, \tau) = (\sigma_a\sigma_b, \tau_a\tau_b)$, where $\sigma_a \in (T_{timed})^*$ and $\sigma_b \in (T_{untimed})^*$, such that $\mu_0 \overset{(\sigma_a,\tau_a)}{\rightarrow} \mu' \overset{(\sigma_b,\tau_b)}{\rightarrow} \mu$, and for the* intermediate marking *μ', $\mu'(p) = \mu(p)$, $\forall\, p \in P_{timed}$ and $p^\bullet \subseteq T_{timed}$.*

Proof: Given a reachable marking μ, there exists a computation ω_0 such that $\mu_0 \overset{(\omega_0,\tau')}{\rightarrow} \mu$, for some time sequence τ'. Let t_1 and t_2 be the first pair of transitions fired consecutively in ω_0, with $t_1 \in T_{untimed}$, $t_2 \in T_{timed}$ and $\eta_1 \leq \eta_2$, where

$$\mu_0 \overset{(\sigma_1,\tau_1)}{\rightarrow} \mu_1 \overset{(t_1,\eta_1)}{\rightarrow} \mu_2 \overset{(t_2,\eta_2)}{\rightarrow} \mu_3 \overset{(\sigma_2,\tau_2)}{\rightarrow} \mu.$$

Let $\{p_1\} = {}^\bullet t_1$, $\{p_2\} = {}^\bullet t_2$. Obviously, p_1 and p_2 are marked in μ_1 and μ_2, respectively. (Hence, $\mu_1(p_1) \geq 1$ and $\mu_2(p_2) \geq 1$.) By definition, p_2 is a timed place; hence, $\mu_2(p_2) = 1$ (Lemma 4.12). We also have $t_1 \not\mapsto t_2$, since t_1 is untimed while t_2 is timed. Therefore $p_2 \notin t_1^\bullet$ and $\mu_1(p_2) \geq \mu_2(p_2) = 1$. Also from Lemma 4.12, $p_1 \neq p_2$. Since t_1 is an untimed transition, none of the clock variables is reset upon firing t_1. Let ν_1 and ν_2 be the clock readings at time η_1 and η_2, respectively. Clearly $\nu_2 = \nu_1 + (\eta_2 - \eta_1)$, regardless of the firing of t_1. By skipping t_1 and firing t_2 in μ_1 at time η_2, we have $\mu_0 \overset{(\sigma_1,\tau_1)}{\rightarrow} \mu_1 \overset{(t_2,\eta_2)}{\rightarrow} \mu'_2$, for some μ'_2, and $1 \leq \mu_1(p_1) \leq \mu'_2(p_1)$. Since t_1 is an untimed transition, it is still enabled in μ'_2 at time η_2, in spite of being delayed. Hence we have

$$\mu_0 \overset{(\sigma_1,\tau_1)}{\rightarrow} \mu_1 \overset{(t_2,\eta_2)}{\rightarrow} \mu'_2 \overset{(t_1,\eta_2)}{\rightarrow} \mu'_3 \overset{(\sigma_2,\tau_2)}{\rightarrow} \mu,$$

for some μ'_3.

By repeatedly applying the above to every pair of neighboring transitions in which the untimed one precedes the timed, a canonical rearrangement $\mu_0 \overset{(\sigma_a,\tau_a)}{\rightarrow} \mu' \overset{(\sigma_b,\tau_b)}{\rightarrow} \mu$ with all the timed (resp., untimed) transitions appear in σ_a (resp., σ_b) is obtained. It remains to show that $\mu'(p) = \mu(p)$, $\forall\, p \in P_{timed}$ and $p^\bullet \subseteq T_{timed}$. Let $p_1 \in P_{timed}$ and $p_1^\bullet \subseteq T_{timed}$. Since $p_1 \in P_{timed}$, all the input transitions of p_1 are timed transitions which will not be fired in σ_b. In other words, no tokens are put into p_1 during the course of firing σ_b; so $\mu'(p_1) \leq \mu(p_1)$. On the other hand, by assumption all the output transitions of p_1 are also timed, so that they won't be fired in σ_b. Since no tokens leave p_1 during σ_b, $\mu'(p_1) \geq \mu(p_1)$. Therefore, $\mu'(p_1) = \mu(p_1)$. ■

In our subsequent discussion, we simply use $\mathcal{P}_T$ to denote the marked timed subnet $((P, T_{timed}, \varphi_t), X, r, q)$ (with initial marking μ_0) of a marked alternating RQ cf-PN $\mathcal{P}$. Note that $\mathcal{P}_T$ itself is an alternating RQ cf-PN and all the transitions in $\mathcal{P}_T$ are timed transitions. So there are only "timed" circuits and circuit collections in $\mathcal{P}_T$.

Lemma 4.15: *Let $\omega = (\sigma, \tau)$ be a computation of $\mathcal{P}_T$*

where $\mu_0 \overset{\omega}{\rightarrow} \mu$. *For every transition t in* $\mathcal{P}_T$, *if t is not on any circuit, then t can be fired at most once in* ω, *i.e.,* $\#_\sigma(t) \leq 1$.

Proof: Suppose $\#_\sigma(t) \geq 2$ and let $\mu_0 \overset{(\sigma_1,\tau_1)}{\rightarrow} \mu_1 \overset{(t,\eta_1)}{\rightarrow} \mu_1' \overset{(\sigma_2,\tau_2)}{\rightarrow} \mu_2 \overset{(t,\eta_2)}{\rightarrow} \mu_2' \overset{(\sigma_3,\tau_3)}{\rightarrow} \mu$. Let ${}^\bullet t = \{p\}$, then we have $\mu_1(p) = 1$, $\mu_1'(p) = 0$ and $\mu_2(p) = 1$. So there must be a path χ (in the PN graph) which starts from a marked place p_a in μ_1' and ends in p, through which a token is deposited to p before time η_2. (Let the firing sequence corresponding to path χ be σ_a.) Since p_a is a timed place, $\mu_1'(p_a) = 1$ (Lemma 4.12). Furthermore, no transition in σ_a ever removes a token from p.

Since transitions of σ_a are carried out in σ_2 (perhaps interleaved with other transitions), i.e. $\#_{\sigma_a} \leq \#_{\sigma_2}$. Hence $\mu_1' \overset{(\sigma_a,\tau_a)}{\rightarrow} \mu_1'' \overset{(t,\eta_2')}{\rightarrow} \mu_2''$ is a legal firing sequence, for some $\tau_a, \eta_2', \mu_1'', \mu_2''$. If $p_a \notin t^\bullet$, $\mu_1(p_a) = 1$. $\mu_1''(p) = 2$ — contradicting Lemma 4.12. Hence, $t \hookrightarrow p_a \hookrightarrow t$, contradicting the assumption. So $\mu_1(p_a) = 0$. Since $\mu_1 \overset{(t,\eta_1)}{\rightarrow} \mu_1'$ and $\mu_1'(p_a) = 1$, it means $p_a \in t^\bullet$ and p_a is then reachable from t. This, together with the fact that $p_a \hookrightarrow t$, results in a circuit containing t — a contradiction. This completes the proof of the Lemma. ■

Let $c = p_1 t_1 p_2 \cdots p_n t_n p_1$ be a circuit in an alternating RQ cf-PN $\mathcal{P}$. A place p_i in c is called an *entry* of c if $\exists\, t \in {}^\bullet p_i$, $t \notin c$. A place p_i in c is called an *exit* of c if $\exists\, t \in p_i^\bullet$, $t \notin c$. A transition t_i in c is called a *leak* of c if $\exists\, p \in t_i^\bullet$, $p \notin c$. In short, an *entry* of a circuit is a place through which tokens are injected into the circuit. An *exit*, in contrast, is a place from which tokens leave the circuit. A *leak* is a transition which spins off tokens whenever the circuit is traversed.

The following is easy to show.

Lemma 4.16: *Let c be a clocked circuit and* μ *be a reachable marking with* $\mu(c) \geq 1$. *Then for arbitrary* $a \in N$ *there exists a computation* $\omega = (\sigma, \tau)$ *such that* $\mu \overset{\omega}{\rightarrow}$ *and* $\#_\sigma = a \cdot \#_c$, *for some* τ. *(That is, c can be traversed a times in* μ.*)*

Corollary 4.17: *Let c be a circuit that is traversed during the course of the computation. If t is one of its leaks, then* $\forall\, p \in t^\bullet$, $p \notin c$ *implies p is an untimed place.*

We can also show the following:

Lemma 4.18: Given a clocked circuit c and a number $d \in N$, we can decide in NP whether traversing circuit c once (if it is traversable) needs time $<$ or $\leq d$.

Given a computation $\omega = (\sigma, \tau)$ of $\mathcal{P}_T$, the *traversed net of* $\mathcal{P}_T$ *associated with* ω, denoted by $\hat{\mathcal{P}}(\omega)$, is the subnet consisting of places and transitions involved in ω. Notice that some places in $\mathcal{P}_T$ may not be timed.

From Lemma 4.12, we have

Lemma 4.19: *Let* $s = \{c_1, c_2, \ldots, c_r\}$ *be a circuit collection in the traversed net* $\hat{\mathcal{P}}(\omega)$. *For any reachable marking* μ *of* $\hat{\mathcal{P}}(\omega)$, *we must have* $\sum_{i=1}^{r} \mu(c_i) \leq 1$. *In other words, there can be at most one token within a circuit collection in* $\hat{\mathcal{P}}(\omega)$ *at any time during the computation* ω.

Lemma 4.20: *Let p be a place in* $\hat{\mathcal{P}}(\omega)$. *If p does not belong to any circuit, then p has at most one input (and output) transition in* $\hat{\mathcal{P}}(\omega)$.

Proof: First consider the *input* case. According to Lemma 4.13, if p is an untimed place, p will have at most one timed input transition visible in $\hat{\mathcal{P}}(\omega)$. Therefore the above statement is true. Next, we consider the case when p is a timed place.

Assume that p has two input transitions t_1 and t_2 in $\hat{\mathcal{P}}(\omega)$. Let $\{p_1\} = {}^\bullet t_1$ and $\{p_2\} = {}^\bullet t_2$. Without loss of generality, we assume that t_1 is fired earlier than t_2 during the computation $\omega = (\sigma, \tau)$, where

$$\mu_0 \overset{(\sigma_1,\tau_1)}{\rightarrow} \mu_1 \overset{(t_1,\eta_1)}{\rightarrow} \mu_1' \overset{(\sigma_2,\tau_2)}{\rightarrow} \mu_2 \overset{(t_2,\eta_2)}{\rightarrow} \mu_2'.$$

So, $\mu_1(p_1) = 1$ and $\mu_1'(p_1) = \mu_1(p_1) - 1 = 0$. Since $1 \geq \mu_1'(p) = \mu_1(p) + 1$, we will have $\mu_1'(p) = 1$. That is, in μ_1', the place p has one token.

Since $\mu_2(p_2) = 1$, there exists a place p' marked in μ_1' such that $p' \hookrightarrow p_2$. Since p is not on any circuits and $p_2 \overset{t_2}{\rightarrow} p$, $p \not\hookrightarrow p_2$ (the assumption of the lemma). Therefore $p' \neq p$. Now we have at μ_1', $\mu_1'(p) = \mu_1'(p') = 1$ and $p' \hookrightarrow p \hookrightarrow t$, for some clocked transition t — violating Lemma 4.12.

The proof of the *output* case is similar, and hence, the details are left to the reader. ■

Let $s = \{c_1, c_2, \ldots, c_r\}$ be a circuit collection in $\hat{\mathcal{P}}(\omega)$ and t be a transition which is not in s. (That is, $t \notin c_i\ \forall\ 1 \leq i \leq r$.) t is said to be an *input* of s if $\exists\, p \in c_j$ for some j where $1 \leq j \leq r$, such that $p \in t^\bullet$, and t is an *output* of s if $\exists\, p \in c_j$ for some j where $1 \leq j \leq r$, such that $p \in {}^\bullet t$.

Corollary 4.21: *If s is a circuit collection in* $\hat{\mathcal{P}}(\omega)$, *then s has at most one input and one output.*

The following lemma is from [17].

Lemma 4.22: (From Lemma 1 in [17]) Let $C = \{c_1, c_2, \ldots, c_n\}$ be a set of connected circuits in a cf-PN $\mathcal{P}$ and μ be a marking with $\mu(c_i) > 0$, for some i. For arbitrary integers $a_1, a_2, \ldots, a_n > 0$, there exists a sequence σ such that $\mu \overset{\sigma}{\rightarrow}$ and $\#_\sigma = \sum_{j=1}^{n} a_j(\#_{c_j})$. (In words, from μ there exists a fireable sequence σ utilizing circuit c_j exactly a_j times, for every j.)

We are in a position to put all the pieces together, in order to devise a system of linear inequalities to capture $\mu_0 \overset{(\sigma_a,\tau_a)}{\rightarrow} \mu' \overset{(\sigma_b,\tau_b)}{\rightarrow} \mu$ (where $\sigma_a \in T_{timed}^*$ and $\sigma_b \in T_{untimed}^*$) for alternating RQ cf-PNs. (Assume the set of timed transitions T_{timed} and the set of timed places P_{timed} of $\mathcal{P}$ are precomputed in advance.)

1. Guess $T' \subseteq T_{timed}$ such that $T' = Tr(\sigma_a)$, the portion actually involved in the computation.
2. Guess the tree structure of T' discussed earlier (see

Fig. 3), and then verify its validity. Let $S = \{s_1, s_2, \ldots, s_g\}$ be circuit collections in T', and $s_i = \{c^i_1, \ldots, c^i_{r_1}\}$ be the set of circuits of s_i, for some r_i. We also guess $d_1, d_2, \ldots, d_g \in N$. Then the following are verified.

   a. for each s_i, check the connectivity condition of $c^i_1, \ldots, c^i_{r_1}$ (doable in polynomial time (Lemma 4.4)),
   b. for each circuit collection s_i, verify the condition of Lemma 4.21 (i.e., whether there is at most one input and output for each s_i),
   c. for any timed place p not in s_i check if p has at most one input (and output) transition (Lemma 4.20),
   d. verify that the clocked circuit in $s_i, 1 \leq i \leq g$, can be traversed in $<$ (or $\leq$) d_i time (Lemma 4.18).

3. Disregarding timing constraints, a system of linear inequalities can be set up (guaranteed by Lemma 4.4) to capture the essence of a marking μ' reachable from the initial marking. (Here μ' is a vector variable.) In setting up such a system of inequalities, the number of times a circuit is traversed in a circuit collection is kept as a scalar variable (see Lemma 4.4). Now we focus on how timing requirements can be enforced by adding additional inequalities to the system. To give the reader a better feel for how this is done, consider the example given in Fig. 3. As discussed earlier, for each transition t in T' not in any circuit collection, t is fired exactly once during the course of the computation. We therefore assign a variable d_t to denote the global time at which t is fired. Since we have allocated variables to represent the numbers of times circuits in T' (including clocked circuits) are traversed, linear inequalities with respect to variables d_t can easily be set up, taking into account the number of times clocked circuits are traversed. Take the path from t_0 to t_4 for example (assuming that the minimum delays (which can be decided in NP as Lemma 4.18 suggests) of circuit collections S_1 and S_2 are 2 and 4 time units, respectively). The associated inequalities for timing constraints along this path are:

$$\begin{cases} d_{t_i} \leq d_{t_{i+1}}, \forall 1 \leq i \leq 3 \\ n_1 * 2 \leq d_{t_3} - d_{t_2} \leq 50 \\ n_2 * 4 + (d_{t_3} - d_{t_1}) \leq d_{t_4} - d_{t_1} \\ 20 < d_{t_4} - d_{t_1} \end{cases}$$

   where n_1 and n_2 (which are scalar variables) represent the numbers of times the clocked circuits in S_1 and S_2, respectively, are executed. The details for the general case are simply a matter of technicality, and hence, are left to the reader.
4. To see whether marking μ (given as part of the problem's instance) can be reached from μ' (which is a vector variable) using only transitions belonging to the untimed portion of the PN, Lemma 4.4 again can be used for setting up a system of linear inequalities.

Based on the above idea, we have the following result.

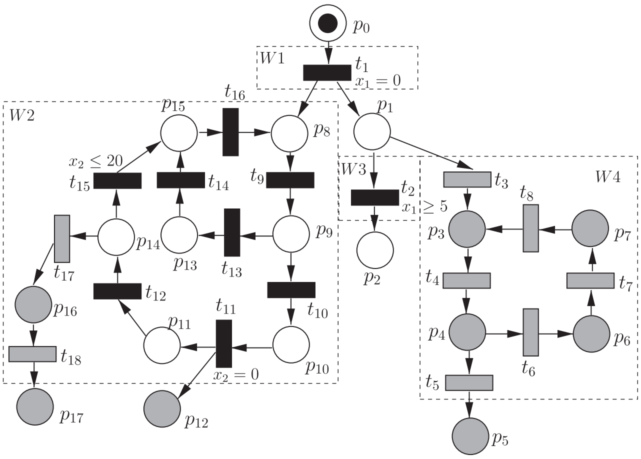

Fig. 4 A flexible manufacturing system with timing constraints.

Theorem 4.23: *The reachability problem for alternating RQ cf-PNs is NP-complete.*

Example 4.1: A flexible manufacturing system with time constraint is specified as follows. This machine disassembles one component to provide finished or unfinished products. Figure 4 is the *alternating RQ cf-PN* model of an assembly machine which manufactures four types of products denoted by p_2, p_5, p_{12} and p_{17} using one type of component denoted by p_0. Workstation 1 divides Material A into two Parts B and C. Then, Part B needs to be processed by Workstation 2 with a portion of rework by the same workstation; and Part C needs to be processed by either Workstation 3 or 4. Workstation 2 with a timing constraint produces two types of products p_{12} and p_{17}. Workstation 3 with a time constraint produces one type of product p_2. Workstation 4 produces one type of product p_5.

Let $\mathcal{P} = ((P, T, \varphi), X, r, q)$ be an alternating RQ cf-PN with initial marking $\mu_0 = (1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ as shown in Fig. 4 where the shaded place and transitions are denoted by untimed places and transitions. $\mathcal{P}$ can be divided into timed subnet $\mathcal{P}_T = ((P, T_{timed}, \varphi_t), r, q)$ and untimed subnet $\mathcal{P}_U = (P, T_u, \varphi_u)$. Given a marking $\mu = (0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 1)$, we want to know whether marking μ can be reached from μ_0. To answer this question, the idea is to construct a system of linear inequalities $\mathcal{L}$ that μ is reachable in $\mathcal{P}$ iff $\mathcal{L}$ has an integer solution.

According to Lemma 4.14, there exists a computation $\omega = (\sigma, \tau) = (\sigma_a\sigma_b, \tau_a\tau_b)$, where $\sigma_a \in (T_{timed})^*$ and $\sigma_b \in (T_{untimed})^*$, such that $\mu_0 \overset{(\sigma_a,\tau_a)}{\rightarrow} \mu' \overset{(\sigma_b,\tau_b)}{\rightarrow} \mu$. We can determine whether marking μ of the alternating RQ cf-PN $\mathcal{P}$ is reachable by the following two steps. First, find an intermediate marking μ' which is reachable from the initial marking μ_0 by firing only the timed transitions. Next, we have to determine if μ is reachable in the untimed subnet from the intermediate marking μ'.

1. Consider $\mu_0 \overset{(\sigma_a,\tau_a)}{\rightarrow} \mu'$, disregarding timing constraints, a

system of linear inequalities can be set up (guaranteed by Lemma 4.4) to capture the essence of a marking μ' reachable from the initial marking as follows:

$$\mu' = \mu_0 + A_1\bar{x}_1$$

in which A_1 represents the adjacency matrix associated with subnet $\mathcal{P}_T$ and is of dimension 18×18, $\bar{x}_1$ is an 18×1 column vector of nonnegative integers which is called the firing count vector. Now we focus on how timing requirements can be enforced by adding additional inequalities to the system. We assign a variable d_t to denote the global time at which transition t is fired and the linear inequalities with respect to variables d_t can be set up. The associated inequalities for timing constraints along the path are

$$\begin{cases} d_{t_1} \le d_{t_2} \\ 5 \le (d_{t_2} - d_{t_1}) \\ d_{t_{11}} \le d_{t_{15}} \\ (d_{t_{15}} - d_{t_{11}}) \le 20 \end{cases}$$

2. Consider $\mu' \overset{(\sigma_b,\tau_b)}{\rightarrow} \mu$, Lemma 4.4 again can be used for setting up a system of linear inequalities as follows:

$$\mu = \mu' + A_2\bar{x}_2$$

in which A_2 represents the adjacency matrix associated with subnet $\mathcal{P}_U$ and is of dimension 18×18, $\bar{x}_2$ is an 18×1 column vector of nonnegative integers which is called the firing count vector.

According to the above results, we can construct a system of linear inequalities $\mathcal{L}$ such that μ is reachable in $\mathcal{P}$ iff $\mathcal{L}$ has an integer solution.

$$\mathcal{L} = \begin{cases} \mu' = \mu_0 + A_1\bar{x}_1 \\ \mu = \mu' + A_2\bar{x}_2 \\ d_{t_1} \le d_{t_2} \\ 5 \le (d_{t_2} - d_{t_1}) \\ d_{t_{11}} \le d_{t_{15}} \\ (d_{t_{15}} - d_{t_{11}}) \le 20 \end{cases}$$

In this example, the $\mathcal{L}$ has an integer solution $\bar{x}_1 = (1,0,0,0,0,0,0,0,3,2,2,2,1,1,1,2,1,1)$, $\bar{x}_2 = (0,0,1,3,1,2,2,2,0,0,0,0,0,0,0,0,0,0)$, $d_{t_1} = 1$, $d_{t_2} = 6$, $d_{t_{15}} = 10$ and $d_{t_{11}} = 5$. ■

References

[1] R. Alur and D. Dill, "A theory of timed automata," Theor. Comput. Sci., vol.126, pp.183–235, 1994.

[2] F. Bause, "On the analysis of Petri net with static priorities," Acta Inform., vol.33, pp.669–685, 1996.

[3] F. Bause, "Analysis of Petri nets with a dynamic priority method," Proc. 18th International Conference on Application and Theory of Petri Nets, Toulouse, France, June 1997.

[4] B. Berard, "Untiming timed languages," Inf. Process. Lett., vol.55, pp.129–135, 1995.

[5] J. Esparza, "Petri nets, commutative context-free grammars, and basic parallel processes," Fundamenta Informaticae, vol.31, pp.13–26, 1997.

[6] R. Gorrieri and G. Siliprandi, "Real-time system verification using P/T nets," CAV'94, LNCS 818, pp.15–26, 1994.

[7] D. Huynh, "Commutative grammars: The complexity of uniform word problems," Inf. Comput., vol.57, pp.21–39, 1983.

[8] D. Huynh, "The complexity of equivalence problems for commutative grammars," Inf. Comput., vol.66, pp.103–121, 1985.

[9] P. Jančar, A. Kučera, and R. Mayr, "Deciding bisimulation-like equivalences with finite-state processes," Theor. Comput. Sci., vol.258, pp.409–433, 2001.

[10] P. Jančar and F. Moller, "Techniques for decidability and undecidability of bisimilarity," Proc. CONCUR'99, vol.1664 of LNCS, pp.30–45, 1999.

[11] W. Lam and R. Brayton, "Alternating RQ timed automata," CAV'93, LNCS 697, pp.237–252, 1993.

[12] W. Lam and R. Brayton, "Criteria for the simple path property in timed automata," CAV'94, LNCS 818, pp.27–40, 1994.

[13] R. Lipton, "The reachability problem requires exponential space," Technical Report 62, Yale University, Dept. of CS., Jan. 1976.

[14] M. Marsan and G. Chiola, "On Petri nets with deterministic and exponential transition firing times," Proc. 7th European Workshop on Application and Theory of Petri Nets, pp.151–165, 1986.

[15] E. Mayr, "An algorithm for the general Petri net reachability problem," SIAM J. Comput., vol.13, no.3, pp.441–460, 1984.

[16] C. Ramchandani, "Analysis of asynchronous concurrent systems by Petri nets," Tech Report MAC TR-120, Massachusetts Institute of Technology, 1974.

[17] H. Yen, "On reachability equivalence for BPP-nets," Theor. Comput. Sci., vol.179, pp.301–317, 1997.

[18] H. Yen, "Priority conflict-free Petri nets," Acta Inform., vol.35, pp.673–688, 1998.

Chien-Liang Chen received the B.S. degree in Computer Science and Information Engineering from Tamkang University, Taiwan, in 1995, the M.S. degree in Electrical Engineering from National Taiwan University, Taiwan, in 2000. He is a Ph.D. candidate in the Department of Electrical Engineering at Nation Taiwan University, Taiwan. His current research interests include Petri net theory, formal methods, discrete event systems and manufacturing systems.

Suey Wang was born in Taiwan, Republic of China, on September 22, 1971. She received the B.S. degree in physics and the M.S. degree in electrical engineering from National Taiwan University, Taiwan, in 1994 and 1996, respectively. She received the M.S. degree in information networking from Carnegie Mellon University, U.S.A., in 1998. She joined Morgan Stanley in 1998. She worked in the New York office for six years before transferring to Tokyo in 2004. She is currently a vice president in Technology department of Morgan Stanley Japan Securities Co., Ltd.

Hsu-Chun Yen was born in Taiwan, Republic of China, on May 29, 1958. He received the B.S. degree in electrical engineering from National Taiwan University, Taiwan, in 1980, the M.S. degree in computer engineering from National Chiao-Tung University, Taiwan, in 1982, and the Ph.D. degree in computer science form the University of Texas at Austin, U.S.A., in 1986. He is presently a Professor of Electrical Engineering at National Taiwan University, where he initially joined in August 1990. Since August 2007, he has served as Dean of School of Informatics at Kainan University, Taiwan. From August 1986 to July 1990, he was an Assistant Professor of Computer Science at Iowa State University, Ames, Iowa, U.S.A. His current research interests include Petri net theory, formal methods, design and analysis of algorithms, and complexity theory. Dr. Yen is an editor of International Journal of Foundations of Computer Science (IJFCS, Worlds Scientific Publisher).