Stubborn Sets for Time Petri Nets

HANIFA BOUCHENEB, Laboratoire VeriForm, Department of Computer Engineering, École Polytechnique de Montréal
KAMEL BARKAOUI, Laboratoire CEDRIC, Conservatoire National des Arts et Métiers

The main limitation of the verification approaches based on state enumeration is the state explosion problem. The partial order reduction techniques aim at attenuating this problem by reducing the number of transitions to be fired from each state while preserving properties of interest. Among the reduction techniques proposed in the literature, this article considers the stubborn set method of Petri nets and investigates its extension to time Petri nets. It establishes some useful sufficient conditions for stubborn sets, which preserve deadlocks and k-boundedness of places.

Categories and Subject Descriptors: D.2.4 [**Software Engineering**]: Software/Program Verification

General Terms: Verification, Partial Order Techniques

Additional Key Words and Phrases: Time Petri nets, stubborn set method, deadlock property

ACM Reference Format:
Hanifa Boucheneb and Kamel Barkaoui. 2015. Stubborn sets for time petri nets. ACM Trans. Embedd. Comput. Syst. 14, 1, Article 11 (January 2015), 25 pages.
DOI: http://dx.doi.org/10.1145/2680541

1. INTRODUCTION

Time Petri nets (TPNs) extend Petri nets by associating a firing interval with each transition, allowing one, at once, to specify different kinds of time constraints and to take them into account during the verification process. They offer a good compromise between modeling power and verification capabilities. The verification techniques of TPNs, such as reachability analysis, are, in general, based on the so-called state space abstraction, where states reachable by the same firing sequence, but at different dates, are grouped in the same set and considered modulo some relation of equivalence (state classes, state zones, or abstract states) [Berthomieu and Diaz 1991; Berthomieu and Vernadat 2003; Boucheneb et al. 2009; Boucheneb and Rakkay 2008; Boucheneb and Hadjidj 2006; Yoneda and Ryuba 1998]. However, for highly concurrent systems, these verification techniques face a severe problem of state-space explosion. Several partial order approaches are proposed in the literature for (time) Petri nets to alleviate this problem, such as partial order unfolding [Chatain and Jard 2013, 2006; Delfieu et al. 2007; Semenov and Yakovlev 1996; Sogbohossou and Delfeu 2008] and partial order reduction [Godefroid 1996; Peled and Wilke 1997; Peled 1993; Valmari and Hansen 2010; Valmari 1992, 1998; Yoneda and Schlingloff 1997].

Authors' addresses: H. Boucheneb, Laboratoire VeriForm, Computer Engineering Department, École Polytechnique de Montréal, Campus de l'Université de Montréal, 2900, boulevard Édouard-Montpetit, 2500, chemin de Polytechnique, Montréal (Québec), Canada, H3T 1J4; K. Barkaoui, Laboratoire CEDRIC, Conservatoire National des Arts et Métiers, 292, Rue St-Martin, 75141 Paris Cedex 03, France.

The idea of the unfolding techniques is to translate a TPN model into an acyclic Petri net with firing time constraints, respecting the partial order of the original model. The available unfolding techniques are, however, limited to 1-safe TPNs[1] or even free-choice 1-safe TPNs[2] [Chatain and Jard 2013].

The common characteristics of the partial order reduction methods are that they explore a subset of firing sequences (representative firing sequences) from each (abstract) state. These subsets are sufficient to verify the properties of interest. In almost all partial order reduction techniques, the selection of the representative firing sequences is based on an independency relation over transitions. Intuitively, two transitions are independent if they can neither disable nor enable each other and their firings in both orders lead to the same (abstract) state. If a transition is selected to be fired from a state, then all its dependent and firable transitions are selected too. Various sufficient conditions, guaranteeing an effective selection of an overapproximation of dependent transitions, are proposed in the literature, such as the persistent set [Godefroid 1996], the ample set [Peled and Wilke 1997; Peled 1993], and the stubborn set [Valmari and Hansen 2010; Valmari 1992]. However, in the context of the TPN state-space abstractions, the different interleavings of the same set of transitions lead, in general, to different abstract states and then the relation of independency is difficult to meet. To overcome this limitation, two main techniques are used in the literature: the local time semantics [Bengtsson et al. 1998; Hakansson and Pettersson 2007; Minea 1999] and partially ordered sets (POSETs) of transitions or events [Belluomini and Myers 2000; Lilius 1998; Lugiez et al. 2005; Mercer et al. 2001; Yoneda et al. 1993].

The local time semantics-based reduction approaches suppose that the model consists of a set of components and each component, represented by a timed model (timed automaton, time Petri net, etc.), has, in addition to the clocks of the component, a reference clock. The reference clocks evolve asynchronously and are synchronized when needed (i.e., when actions of synchronization are executed). Such approaches need additional clocks and the differences between reference clocks may diverge, leading to infinite state space [Lugiez et al. 2005].

The partial order reduction approaches based on POSETs aim to force the independency relation by fixing partially the firing order of transitions or events [Belluomini and Myers 2000; Lilius 1998; Lugiez et al. 2005; Mercer et al. 2001; Yoneda et al. 1993]. This principle is applied for timed automata in Lugiez et al. [2005], 1-safe TPNs in Lilius [1998] and Yoneda and Schlingloff [1997], and 1-safe P-TPNs[3] in Lilius [1998] and Yoneda et al. [1993]. In Lugiez et al. [2005], the classical clock zones of timed automata are replaced with event zones, used to compute the convex hull of zones reachable by some equivalent sequences of transitions. The correctness of this approach is ensured by the fact that the union of zones reached by equivalent sequences of transitions (with no shared clocks) is a zone, in the context of timed automata [Salah et al. 2006]. However, this nice feature does not hold in the context of TPNs, including 1-safe TPNs [Boucheneb and Barkaoui 2013]. In Yoneda and Schlingloff [1997], to force this feature, the authors use the notion of *truth parents* and compute the union of zones reachable by equivalent sequences obtained by permuting some independent transitions (in the sense of stubborn sets). The notion of parents involves keeping, in each abstract state, in addition to time constraints of the enabled transitions, those of their parents. In Lilius [1998], the authors have defined a state-space abstraction where the firing order constraints between nonrelated transitions[4] are totally ignored

[1]A 1-safe time Petri net is a 1-bounded time Petri net (i.e., each place can contain at most one token).

[2]A free-choice 1-safe TPN is a 1-safe TPN where for any pair of transitions, their sets of input places are either equal or disjoint.

[3]A 1-safe P-TPN is a 1-safe Petri net, where a time interval is associated with each place.

[4]Transitions are nonrelated if no one is enabled by the others.

when computing successors. The subset of transitions explored from each abstract state is a persistent set [Lilius 1998]. However, as we will show in Section 3.1, the state-space abstraction proposed in Lilius [1998] does not preserve deadlocks. In Belluomini and Myers [2000], the reachability algorithm, proposed for P-time Petri nets, operates on two zones to determine the state of the model: one contains time constraints over token ages (place timer zone) and the other contains firing delay constraints between transitions (transition zone). When a transition is fired, both zones are updated to compute a partial order successor. This approach is improved, in Mercer et al. [2001], by eliminating the place timer zone and keeping in the transition zone clocks of the already fired transitions until firing all transitions they have enabled.

In this article, we consider time Petri nets and the stubborn set method (the partial order reduction proposed in Valmari [1992] and Valmari and Hansen [2010]). We revisit this method, in the context of POSETs (partial order successors). The intuitive idea is to relax the firing rule of sequences of transitions by ignoring some firing order constraints of transitions. This relaxation aims at computing, by exploring only one sequence, the union of abstract states reachable by several equivalent sequences. We establish some useful sufficient conditions allowing one to compute a stubborn set for each TPN abstract state. This set determines the subset of firable transitions to explore from the abstract state and also the firing order constraints to be relaxed. We show that the resulting reduced graph preserves deadlocks of the TPN and also the k-boundedness of places. Note that the ideas and results presented here can be adapted to other methods (ample sets, persistent sets). Moreover, the extension of the verification approach, proposed here, to LTL_{-X}[5] properties over markings could be achieved as shown in Valmari and Hansen [2010].

Section 2 is devoted to time Petri nets, its semantics, and its state-space abstractions. Section 3 defines the notions of partial order successor and reduced state space. Then, it proposes, using the notion of partial order successor, a revisited definition of stubborn sets and shows that they preserve deadlocks. Section 4 establishes some practical sufficient conditions for stubborn sets, which are useful to compute reduced state class graphs (RSCGs). Then, we prove that the resulting RSCGs preserve the k-boundedness of places of the TPN models. Section 5 reports some experimental results. Section 6 is devoted to the related work. Finally, the conclusion is presented in Section 7.

2. TIME PETRI NETS

2.1. Definition and Semantics

Let P be a nonempty set. A multiset over P is a function $M : P \longrightarrow \mathbb{N}$, $\mathbb{N}$ being the set of natural numbers, defined also by the formal sum: $\sum_{p\in P} M(p) \bullet p$[6]. We denote P_{MS} and 0 the set of all multisets over P and the empty multiset, respectively. Let $M_1 \in P_{MS}$, $M_2 \in P_{MS}$, and $\prec \in \{\leq, =, <, >, \geq\}$. Operations on multisets are defined as usual:

1) $\forall p \in P,\ p \in M_1$ iff $M_1(p) > 0$
2) $M_1 \prec M_2$ iff $\forall p \in P, M_1(p) \prec M_2(p)$
3) $M_1 + M_2 = \sum_{p\in P}(M_1(p) + M_2(p)) \bullet p$
4) $M_1 \times M_2 = \sum_{p\in P} Min(M_1(p), M_2(p)) \bullet p$
5) If $M_1 \leq M_2$ then $M_2 - M_1 = \sum_{p\in P}(M_2(p) - M_1(p)) \bullet p$.

Let $\mathbb{Q}^+$ and $\mathbb{R}^+$ be the sets of nonnegative rational and real numbers, respectively, and $INT_{\mathbb{X}} = \{[a, b] | (a, b) \in \mathbb{X} \times (\mathbb{X} \cup \{\infty\})\}$, for $\mathbb{X} \in \{\mathbb{Q}^+, \mathbb{R}^+\}$, the set of intervals whose lower and upper bounds are in $\mathbb{X}$ and $\mathbb{X} \cup \{\infty\}$, respectively.

[5]LTL_{-X} properties are LTL properties where the next operator X is forbidden.
[6]The symbol $\bullet$ is used here as a separator between the elements of M and their occurrence numbers. This separator, however, can be omitted.

Definition 2.1. A time Petri net is a tuple $\mathcal{N} = (P, T, pre, post, M_0, Is)$ where:

—P and T are finite and nonempty sets of places and transitions s.t. $P \cap T = \emptyset$.
—pre and $post$ are the backward and forward incidence functions ($pre, post : T \longrightarrow P_{MS}$).
—$M_0 \in P_{MS}$ is the initial marking.
—Is is the static firing function ($Is : T \rightarrow INT_{Q^+}$). $\downarrow Is(t)$ and $\uparrow Is(t)$ denote the lower and the upper bounds of the static firing interval of transition t.

For $t \in T$, ${}^\circ t = \{p \in P | pre(t)(p) > 0\}$, and $t^\circ = \{p \in P | post(t)(p) > 0\}$, denote the sets of input and output places of t, respectively. For $p \in P$, ${}^\circ p = \{t \in T | post(t)(p) > 0\}$, and $p^\circ = \{t \in T | pre(t)(p) > 0\}$, denote the sets of input and output transitions of p, respectively.
We denote $({}^\circ t)^\circ = \bigcup_{p \in {}^\circ t} p^\circ$ the set of transitions structurally in conflict with t.

The sets ${}^\circ({}^\circ t) = \bigcup_{p \in {}^\circ t} {}^\circ p$ and $(t^\circ)^\circ = \bigcup_{p \in t^\circ} p^\circ$ are called the input and output transitions of t, respectively.

Each marking of $\mathcal{N}$ is a multiset over P. Let M be a marking of $\mathcal{N}$ and $t \in T$ a transition. The transition t is enabled in M, denoted $M[t >$ iff all required tokens for firing t are present in M, that is, $M \geq pre(t)$. In case t is enabled in M, its firing leads to the marking $M' = M - pre(t) + post(t)$. The notation $M[t > M'$ means that t is enabled in M and M' is the marking reached from M by t. We denote $En(M)$ the set of transitions enabled for M, that is, $En(M) = \{t \in T \mid M \geq pre(t)\}$.

For $t \in En(M)$, we denote $CF(M, t)$ the set of transitions enabled in M but in conflict with t, that is, $CF(M, t) = \{t' \in En(M) \mid t' = t \vee M \not\geq pre(t) + pre(t')\}$, where $M \not\geq pre(t) + pre(t')$ is equivalent to $\exists p \in P, M(p) < pre(t)(p) + pre(t')(p)$.

For any sequence $t_1 t_2 \ldots t_n \in T^+$, the usual notation $M[t_1 t_2 \ldots t_n >$ means that there are markings $M_1, \ldots M_n$ so that $M_1 = M$ and $M_i[t_i > M_{i+1}$, for $i \in [1, n-1]$ and $M_n[t_n >$. The notation $M[t_1 t_2 \ldots t_n > M'$ gives, in addition, the marking reached by the sequence from M.

Several semantics are proposed in the literature for the TPNs [Bérard et al. 2013; Boucheneb et al. 2013; Boyer and Diaz 2001]. An overview and a classification of the different semantics can be found in Boucheneb et al. [2013]. We consider here the classical and widely used semantics (i.e., the threshold, intermediate and single-server semantics). It differs from the others in the interpretation of the notion of newly enabled transition, the characterization of states, and the server policy. In the classical semantics, the notion of newly enabled transitions refers to the intermediate markings (markings resulting from the consummation of tokens). The firing of a transition is then supposed not atomic w.r.t. markings, whereas, in the atomic semantics, the firing is supposed atomic w.r.t. markings [Bérard et al. 2013]. In the classical semantics, the timing information is associated with transitions, which is either represented by clocks or delays (threshold semantics). In other semantics, referred to as age semantics, the timing information is associated with tokens and represented by clocks giving their ages [Boyer and Diaz 2001]. Threshold and age semantics are both relevant, depending of the context [Boyer and Diaz 2001]. The service policy specifies whether several enabling instances of the same transition may be handled simultaneously (multiple-server semantics) or not (single-server semantics). For the single-server semantics, the multienabledness is not ambiguous since only one enabling instance of each transition is considered at each state (i.e., sequential management). However, different interpretations can be defined for multiple-server semantics [Boucheneb et al. 2013].

Let M' be the successor marking of M by t. We denote $Nw(M, t)$ the set of transitions newly enabled in the marking M' reached from M by firing t. Formally, for the TPN classical semantics, $Nw(M,t)$ contains t, if t is enabled in M', and also all transitions enabled in the marking M but not enabled in the intermediate marking $M - pre(t)$, that is, $Nw(M,t) = \{t' \in En(M') \mid t' = t \vee M - pre(t) \not\geq pre(t')\}$.

Starting from the initial marking M_0, $\mathcal{N}$ evolves by firing transitions and time progressions. When a transition t becomes enabled, its firing interval is set to its static firing interval. Bounds of its interval decrease synchronously with time until it is fired or disabled by a conflicting firing. Transition t is firable, if the lower bound of its firing interval reaches 0. It must fire immediately, without any additional delay, when the upper bound of its firing interval reaches 0, unless it is disabled by another firing. The firing of a transition takes no time.

Formally, the state of $\mathcal{N}$ is defined as a pair $s = (M, I)$, where M is a marking and I is a firing interval function ($I : En(M) \to INT_{R^+}$). The initial state of $\mathcal{N}$ is $s_0 = (M_0, I_0)$, where $I_0(t) = Is(t)$, for all $t \in En(M_0)$.

Let $\mathcal{S}$ be the set of all syntactically correct states, $s = (M, I)$ and $s' = (M', I')$ two states of $\mathcal{S}$, $dh \in \mathbb{R}^+$ a nonnegative real number, $t \in T$ a transition, and $\to$ the transition relation defined by:

—$s \overset{dh}{\to} s'$ (s' is also denoted $s + dh$) iff the state s' is reachable from state s by dh time units, that is,

$$\forall t \in En(M), dh \leq \uparrow I(t), M' = M \text{ and}$$

$$\forall t' \in En(M'), I'(t') = [Max(0, \downarrow I(t') - dh), \uparrow I(t') - dh].$$

—$s \overset{t}{\to} s'$ iff t is immediately firable from s and its firing leads to s', that is,

$$t \in En(M), \quad \downarrow I(t) = 0, \quad M' = M - pre(t) + post(t), \text{ and}$$

$$\forall t' \in En(M'), I'(t') = \begin{cases} Is(t') & \text{if } t' \in Nw(M,t) \\ I(t') & \text{otherwise.} \end{cases}$$

The semantics of $\mathcal{N}$ is defined by the transition system $(S, \to, s_0)$, where $S \subseteq \mathcal{S}$ is the set of all states reachable from s_0 by $\overset{*}{\to}$ (the reflexive and transitive closure of $\to$).

A *run* in $(S, \to, s_0)$, starting from a state s_1 of S, is a maximal sequence[7] $\rho = s_1 \overset{dh_1}{\to} s_1 + dh_1 \overset{t_1}{\to} s_2 \overset{dh_2}{\to} s_2 + dh_2 \overset{t_2}{\to} s_3 \ldots$. By convention, for any state s_i, relation $s_i \overset{0}{\to} s_i$ holds. Sequences $dh_1 t_1 dh_2 t_2 \ldots$ and $t_1 t_2 \ldots$ are called the timed trace and firing sequence (untimed trace) of ρ, respectively. Runs of $\mathcal{N}$ are all runs of the initial state s_0. The *timed language* of $\mathcal{N}$ is the set of its timed traces.

A marking M is reachable in $\mathcal{N}$ iff $\exists s \in S$ s.t. the marking of s is M. Let $\mathcal{M}$ be the set of all reachable markings of $\mathcal{N}$. A place p is k-bounded iff $\forall M \in \mathcal{M}, M(p) \leq k$. The TPN $\mathcal{N}$ is k-bounded iff $\forall p \in P$, p is k-bounded. It is bounded iff $\exists k \in \mathbb{N}, \mathcal{N}$ is k-bounded.

2.2. TPN State-Space Abstractions

Several abstractions of the TPN state space have been proposed in the literature. The well known are the *State Class Graph (SCG)* [Berthomieu and Diaz 1991], the *Contracted State Class Graph (CSCG)* [Boucheneb and Rakkay 2008], the *Geometric Region Graph (GRG)* [Yoneda and Ryuba 1998], the *Strong State Class Graph (SSCG)* [Berthomieu and Vernadat 2003], the *Zone-Based Graph (ZBG)* [Boucheneb et al. 2009], and the *Atomic State Class Graphs (ASCGs)* [Yoneda and Ryuba 1998; Berthomieu and

[7]The sequence is either infinite or finite ending up in a terminal (i.e., deadlock) state (i.e., a state with no enabled transitions).

Vernadat 2003; Boucheneb and Hadjidj 2006]. These abstractions differ in the way that states are defined and computed. There are two categories: interval-based abstractions (SCG, CSCG) and clock-based abstractions (GRG, SSCG, ZBG, ASCG). In the interval-based abstractions, a state is defined by a marking and firing intervals of the enabled transitions (the definition of state considered here). In the clock-based abstractions, a state is defined by a marking and a valuation of clocks giving for each enabled transition its enabling duration.

In such abstractions, all states reachable by the same firing sequence, but at different dates, are grouped in the same set. The grouped states are then considered modulo some relation of equivalence. States of each class of equivalence, called here abstract states, share the same marking and the union of their time domains is represented by a consistent conjunction of atomic constraints. An atomic constraint is either a simple constraint of the form $x \prec c, -x \prec c$, or a triangular constraint of the form $x - y \prec c$, where x, y are real-valued variables, $\prec \in \{<, =, \leq, \geq, >\}$, and $c \in \mathbb{Q} \cup \{\infty, -\infty\}$ is a rational number. From the practical point of view, every conjunction of atomic constraints is represented by means of a *Difference Bound Matrix* (DBM) [Bengtsson 2002]. Although the same nonempty domain may be encoded by a different conjunction of atomic constraints, their DBMs have a canonical form. The canonical form of a DBM is the representation with the tightest bounds on all differences between variables, computed by propagating the effect of each entry through the DBM. Two conjunctions of atomic constraints are equivalent (i.e., represent the same domain) iff their DBMs have the same canonical form. Canonical forms make operations over formulas much simpler [Bengtsson 2002].

2.2.1. Definitions of Abstract State and State-Space Abstraction. Let $\mathcal{N} = (P, T, pre, post, M_0, Is)$ be a TPN. Syntactically, an abstract state of $\mathcal{N}$ is defined as a pair $\alpha = (M, F)$, where M is a marking and F is a consistent conjunction of atomic constraints over clocks or delays of transitions enabled in M. The formula F characterizes the union of firing time domains of all states within α. A state $s' = (M', I')$ belongs to α iff $M = M'$ and its firing time domain is included in the firing time domain of α (i.e., $s' = (M', I') \in \alpha$ iff $M = M'$ and $F \equiv (F \wedge \bigwedge_{t \in En(M)} \downarrow I'(t) \leq t \leq \uparrow I'(t))$).

Let $\mathcal{A}$ be the set of all syntactically correct abstract states and *succ* a function from $\mathcal{A} \times T$ to $\mathcal{A} \cup \{\emptyset\}$, called the successor function, such that for any transition $t \in T$ and abstract state $\alpha = (M, F) \in \mathcal{A}$, $succ(\alpha, t) \neq \emptyset$ iff there is at least a state s in α from which t is firable after possibly some delay, that is, $\exists s \in \alpha, \exists dh \in \mathbb{R}^+, \exists s' \in \mathcal{S}, s \overset{dh}{\rightarrow} s + dh \overset{t}{\rightarrow} s'$. In such a case, t is said to be firable from α.

Let $\alpha = (M, F)$ be an abstract state of $\mathcal{A}$. We denote $Fr(\alpha) = \{t \in T \mid succ(\alpha, t) \neq \emptyset\}$ the set of transitions firable from α. α is a deadlock iff there is no firable transition from α (i.e., $Fr(\alpha) = \emptyset$). Note that for TPNs, it holds that $Fr(\alpha) = \emptyset \Leftrightarrow En(M) = \emptyset$.

Let α be an abstract state of $\mathcal{A}$ and $\omega \in T^*$ a sequence of transitions. The successor function is extended to sequences of transitions as follows: $succ(\alpha, \omega) = succ\,(succ(\alpha, \omega_1), \omega_2)$, where $\omega = \omega_1\omega_2$ and, by convention, $succ(\alpha, \epsilon) = \alpha$, ϵ being the empty sequence.

A state-space abstraction of $\mathcal{N}$ is a structure $\mathbb{A} = (\mathcal{A}_C, succ, \alpha_0)$, where α_0 is the initial abstract state of $\mathcal{N}$ and $\mathcal{A}_C$ is the set of abstract states accessible from α_0 by applying repeatedly the successor function *succ*, that is, $\mathcal{A}_C = \{\alpha \in \mathcal{A} | \exists \omega \in T^*, \alpha = succ(\alpha_0, \omega)\}$.

For $\alpha, \alpha' \in \mathcal{A}$, $t \in T$, we denote $\alpha \xrightarrow{t} \alpha'$ iff $\alpha' = succ(\alpha, t)$. A *run* starting from α is a maximal sequence $\eta = \alpha \xrightarrow{t_1} \alpha_2 \xrightarrow{t_2} \alpha_3 \xrightarrow{t_3} \ldots$. The firing sequence of η is $t_1t_2t_3 \ldots$. The runs and firing sequences of $\mathbb{A}$ are the runs and firing sequences of α_0, respectively.

A state-space abstraction $\mathbb{A}$ is sound w.r.t. the markings and firing sequences of the TPN iff $\mathbb{A}$ includes all markings and firing sequences of the TPN, that is,

$$\forall s, s' \in \mathcal{S}, \forall t \in T, \forall dh \in \mathbb{R}^+, s \overset{dh}{\to} s + dh \overset{t}{\to} s'$$
$$\Rightarrow \forall \alpha \in \mathcal{A}_C \text{ s.t. } s \in \alpha, \exists \alpha' \in \mathcal{A}_C \text{ s.t. } s' \in \alpha' \wedge \alpha \xrightarrow{t} \alpha'.$$

$\mathbb{A}$ is complete w.r.t. the markings and firing sequences of the TPN iff the reachable markings and firing sequences of the TPN include those of $\mathbb{A}$, that is, $\forall \alpha, \alpha' \in \mathcal{A}_C, t \in T$,

$$\alpha \xrightarrow{t} \alpha' \Rightarrow \exists s \in \alpha, \exists s' \in \alpha', \exists dh \in \mathbb{R}^+, s \overset{dh}{\to} s + dh \overset{t}{\to} s'.$$

We consider here sound and complete abstractions w.r.t. the markings and firing sequences.

Let $\omega, \omega' \in T^+$ be two sequences of transitions. ω and ω' are equivalent (denoted $\omega \equiv \omega'$) iff ω' can be obtained from ω by successive permutations of its transitions. By convention, it holds that $\omega \equiv \omega$. We denote $||\omega|| \subseteq T$ the set of transitions appearing in ω.

2.2.2. Contracted State Class Graph. As an example of a sound and complete state-space abstraction, we consider here the CSCG. We report in the following the definitions of its abstract state (called state class) and the successor function *succ*.

The CSCG state class is an abstract state $\alpha = (M, F) \in \mathcal{A}$ s.t. F is a conjunction of triangular atomic constraints over delays of the enabled transitions in M. By convention, $F = \mathit{true}$ if the number of enabled transitions in M is less than 2 (i.e., there is no triangular atomic constraint in F). The CSCG initial state class is $\alpha_0 = (M_0, F_0)$, where

$$F_0 = \bigwedge_{t,t' \in En(M_0) \text{ s.t. } t \neq t'} t - t' \leq \uparrow Is(t) - \downarrow Is(t')),$$

t and t' being real-valued variables representing firing delays of transitions t and t', respectively.

Let α be a CSCG state class and $t_f \in T$. The transition t_f is firable from α (i.e., $succ(\alpha, t_f) \neq \emptyset$) iff $t_f \in En(M)$ and the following formula is consistent (its domain is not empty): $F \wedge (\bigwedge_{t \in En(M)} t_f - t \leq 0)$.

Intuitively, this formula, called the CSCG firing time condition of t_f from α, means that t_f is firable before all other transitions enabled in M. If $succ(\alpha, t_f) \neq \emptyset$, then $succ(\alpha, t_f) = (M', F')$, where $M' = M - pre(t_f) + post(t_f)$ and F' is computed in three steps:

1) Set F' to $F \wedge \bigwedge_{t \in En(M)} t_f - t \leq 0 \ \wedge \ \bigwedge_{t' \in Nw(M, t_f)} \downarrow Is(t') \leq t'^n - t_f \leq \uparrow Is(t')$..
2) Put F' in canonical form and eliminate all transitions of $CF(M, t_f)$.
3) Rename each t'^n in t'. The notation t'^n allows one to deal with the situation where t' is enabled before firing t_f and newly enabled by t_f (i.e., $t' \in CF(M, t_f) \cap Nw(M, t_f)$). The new instance of t' is temporally represented by t'^n, in step 1.

The CSCG of a TPN is finite iff the TPN is bounded (i.e., has a finite number of reachable markings). The boundedness is not decidable, but there are some useful sufficient conditions that ensure boundedness [Berthomieu and Diaz 1991].

Example 2.2. Consider the model *TPN*1 at Figure 1. Its CSCG initial state class is $\alpha_0 = (p_1 + p_2, -2 \leq t_1 - t_2 \leq 0)$. There are two enabled transitions t_1 and t_2, which are also firable from α_0, since their firing conditions $-2 \leq t_1 - t_2 \leq 0 \wedge t_1 \leq t_2$ and $-2 \leq t_1 - t_2 \leq 0 \wedge t_2 \leq t_1$ are consistent. For instance, let us compute the successor of α_0 by t_1. The firing of t_1 leads to the CSCG state class $\alpha_1 = (p_2 + p_3, -2 \leq t_2 - t_3 \leq 0)$. Its marking is computed as usual. Its formula is computed in three steps:

1) Set the formula to the firing condition of t_1 from α_0 augmented with time constraints of transition t_3 newly enabled by t_1: $-2 \leq t_1 - t_2 \leq 0 \wedge t_1 \leq t_2 \wedge t_3^n - t_1 = 2$.

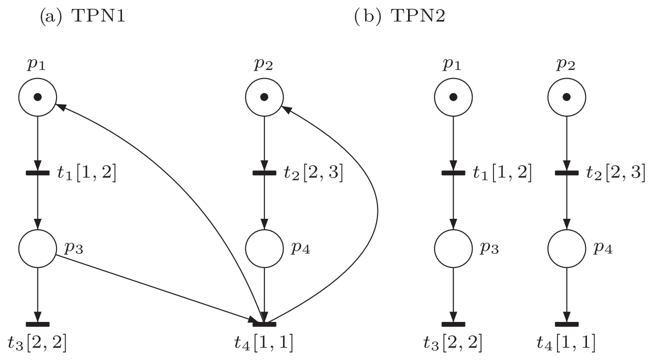

Fig. 1. Two simple TPNs.

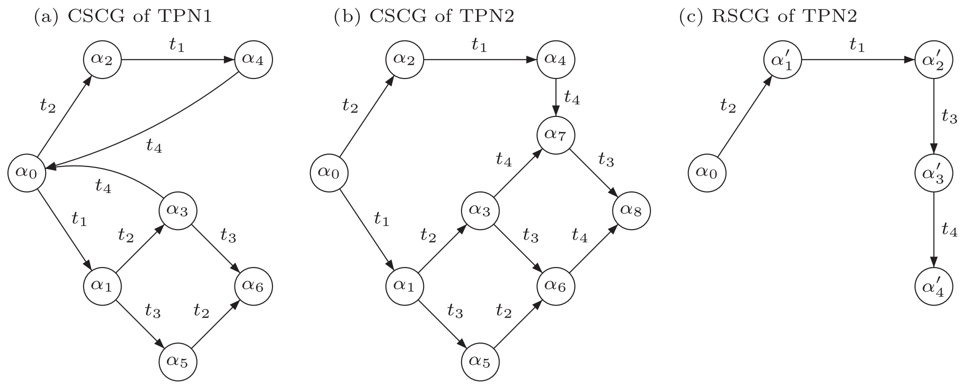

Fig. 2. CSCG of TPN1, CSCG of TPN2, and RSCG of TPN2.

2) Put the formula in canonical form and eliminate t_1: $-2 \leq t_2 - t_3^n \leq 0$.
3) Rename t_3^n in t_3: $-2 \leq t_2 - t_3 \leq 0$.

We report in Figure 2 and Table I the CSCGs and their state classes of the models at Figure 1.

3. PARTIAL ORDER REDUCTION FOR TIME PETRI NETS

Combining a partial order reduction technique with a state space abstraction aims at selecting, from each abstract state, a subset of firable transitions (representative transitions), which is sufficient to verify the properties of interest. We can find, in the literature, different selection procedures of the representative transitions such as the persistent set [Godefroid 1996], the ample set [Peled and Wilke 1997; Peled 1993], and the stubborn set [Valmari and Hansen 2010; Valmari 1998, 1992]. Among these methods, we consider here the stubborn set method and propose to revisit it in the context of the TPN abstractions.

3.1. Classical Stubborn Set Method

Valmari and Hansen [2010] have considered Petri nets and established some sufficient conditions to be met by the set of transitions selected from each marking so as to preserve deadlocks. Such sets are simply called here stubborn sets. They correspond to the strongly dynamic stubborn sets of Valmari and Hansen [2010]. In the context of

Table I. State Classes of TPN1 and TPN2

CSCG State Classes of TPN1	CSCG State Classes of TPN2	RSCG State Classes of TPN2
$\alpha_0 = (p_1 + p_2, -2 \le t_1 - t_2 \le 0)$	$\alpha_0 = (p_1 + p_2, -2 \le t_1 - t_2 \le 0)$	$\alpha_0 = (p_1 + p_2, -2 \le t_1 - t_2 \le 0)$
$\alpha_1 = (p_2 + p_3, -2 \le t_2 - t_3 \le 0)$	$\alpha_1 = (p_2 + p_3, -2 \le t_2 - t_3 \le 0)$	$\alpha'_1 = (p_1 + p_4, -3 \le t_1 - t_4 \le -1)$
$\alpha_2 = (p_1 + p_4, true)$	$\alpha_2 = (p_1 + p_4, t_1 - t_4 = -1)$	$\alpha'_2 = (p_3 + p_4, -1 \le t_3 - t_4 \le 1)$
$\alpha_3 = (p_3 + p_4, -1 \le t_3 - t_4 \le 1)$	$\alpha_3 = (p_3 + p_4, -1 \le t_3 - t_4 \le 1)$	$\alpha'_3 = (p_4, true)$
$\alpha_4 = (p_3 + p_4, t_3 - t_4 = 1)$	$\alpha_4 = (p_3 + p_4, t_3 - t_4 = 1)$	$\alpha'_4 = (0, true)$
$\alpha_5 = (p_2, true)$	$\alpha_5 = (p_2, true)$	
$\alpha_6 = (p_4, true)$	$\alpha_6 = (p_4, true)$	
	$\alpha_7 = (p_3, true)$	
	$\alpha_8 = (0, true)$	

the TPN state-space abstractions, a stubborn set of an abstract state α, denoted *stub*, is a subset of transitions, which satisfies the following conditions:

$D0$: $Fr(\alpha) \neq \emptyset \;\Leftrightarrow\; stub \cap Fr(\alpha) \neq \emptyset$.
$D1$: $\exists t \in stub \cap Fr(\alpha), \forall \omega \in (T - stub)^+, succ(\alpha, \omega) \neq \emptyset \;\Rightarrow\; succ(\alpha, \omega t) \neq \emptyset$.
$D2$: $\forall t \in stub, \forall \omega \in (T - stub)^+, succ(\alpha, \omega t) \neq \emptyset \;\Rightarrow\; succ(\alpha, t\omega) = succ(\alpha, \omega t)$.

Intuitively, condition $D0$ ensures that the stubborn set of any abstract state α is empty iff α is a deadlock. Conditions $D1$ and $D2$ mean that the firing of transitions outside *stub* has no effect on the transitions of *stub*. Moreover, the firing from α of sequences obtained by permuting transitions of *stub* with transitions outside *stub* leads to the same abstract state.

In the context of marking graphs of Petri nets, the marking reached by firing a set of transitions is independent of their firing order. However, in the TPN state-space abstractions, the abstract states reached by different interleavings of the same set of transitions are, in general, different. Condition $D2$ is then difficult to satisfy. Indeed, even if both sequences ωt and $t\omega$ are firable from the abstract state α, they lead, in general, to two different abstract states, which, in addition, may have different behaviors.

Example 3.1. Consider the TPN shown in Figure 1(a). From the initial abstract state α_0, transitions t_1 and t_2 are firable in both orders leading to the same marking $p_3 + p_4$ but to two different state classes with different behaviors (see Figure 2(a)). Indeed, in case t_2 is fired before t_1, there is no delay between their firings (t_1 and t_2 are fired at the same date, starting with t_2). So, in this case, transitions t_3 and t_4 are enabled at the same date but t_4 is eventually fired first and then disables t_3. In case t_1 is fired before t_2, both transitions t_3 and t_4 are firable. The firing of t_3 disables t_4 and leads to a deadlock. So, we will not preserve deadlocks if we decide to relax Condition $D2$ as follows: $D2'$: if $succ(\alpha, \omega t) \neq \emptyset$, then $succ(\alpha, t\omega) \neq \emptyset$. In such a case, $stub_0 = \{t_2\}$ satisfies $D0$, $D1$, and $D2'$. Exploring only t_2 from the initial abstract state will not detect the deadlock reachable by the sequence $t_1t_2t_3$.

To overcome the limitation of applying condition $D2$ in the context of TPN, we use the notion of partial order successors (POSETs) [Belluomini and Myers 2000; Lilius 1998; Lugiez et al. 2005; Mercer et al. 2001; Yoneda et al. 1993]. The intuitive idea is to relax the firing rule by ignoring some firing order constraints of transitions. This relaxation aims at computing, by exploring only one sequence, the union of abstract states reachable by several equivalent sequences and then rewriting condition $D2$ using partial order successors. Let us introduce by means of an example the idea of the partial order successor.

Example 3.2. Consider the TPN at Figure 1(b). Transitions t_1 and t_2 are firable from the initial abstract state α_0, in both orders. The idea is to compute the union of the abstract states $succ(\alpha_0, t_1t_2)$ and $succ(\alpha_0, t_2t_1)$ by exploring only one sequence t_1t_2 or t_2t_1. Only one successor of α_0 by t_1 (or t_2) is computed, without imposing any firing order between transitions t_1 and t_2. Let $succ_G(\alpha_0, t_1)$ be the computed abstract state. In $succ_G(\alpha_0, t_1)$, there are two enabled and firable transitions: t_2 and t_3. These transitions are firable in both orders. For $succ_G(\alpha_0, t_1)$, only one successor by t_2 (or t_3) is computed, without imposing any firing order between t_2 and t_3, and so on. Doing so, we handle concisely all sequences of $t_1t_3||t_2t_4$.

To use the notion of partial order successors, we must establish some sufficient condition allowing one to select which firing order constraints of transitions are to be ignored so as to handle the largest sets of equivalent sequences while preserving properties of interest. The following example shows that limiting this choice to nonconflicting transitions is not sufficient.

Example 3.3. Consider the TPN at Figure 1(a). Transitions t_1 and t_2 are firable from the initial abstract state α_0, in both orders, and are not structurally in conflict. However, the union of the abstract states $succ(\alpha_0, t_1t_2)$ and $succ(\alpha_0, t_2t_1)$ cannot be computed by exploring the sequence t_2t_1 and abstracting away the firing order constraints between t_1 and t_2. Indeed, $succ_G(\alpha_0, t_2) = (p_1 + p_4, true)$ and $succ_G(\alpha_0, t_2t_1) = (p_3 + p_4, t_3 - t_4 = 1)$ are not equal to $succ(\alpha_0, t_1t_2) \cup succ(\alpha_0, t_2t_1)$ (see Figure 2(a) and Table I). Intuitively, the reason of this failure is that t_3 and t_4 are either enabled by the same transition or different transitions, independently of the firing order of t_1 and t_2. For the sequence t_2t_1, both transitions t_3 and t_4 are enabled (at the same date) by t_1. For the sequence t_1t_2, transitions t_3 and t_4 are enabled by t_1 and t_2, respectively.

3.2. Reduced State Spaces

The definition of the reduced state space is inspired from Valmari and Hansen [2010] but based on the notion of partial order successor.

Definition 3.4. Partial order successors and reduced spaces
Let $\mathbb{A} = (\mathcal{A}_C, succ, \alpha_0)$ be a sound and complete state-space abstraction of some TPN $\mathcal{N}$.

—Let G be a function from $\mathcal{A}_C$ to 2^T called a partial order generator. The partial order successor function $succ_G$ is a function from $\mathcal{A}_C \times T$ defined by:

$$\forall \alpha = (M, F) \in \mathcal{A}_C, \forall t_f \in T, succ_G(\alpha, t_f) \neq \emptyset \text{ iff } succ(\alpha, t_f) \neq \emptyset.$$

If $succ_G(\alpha, t_f) \neq \emptyset$, then the abstract state $\alpha' = succ_G(\alpha, t_f)$ is computed as $succ(\alpha, t_f)$, except that the used firing time condition is $F \wedge \bigwedge_{t_i \in En(M) \cap G(\alpha)} t_f \leq t_i$.

Formally, if $succ_G(\alpha, t_f) \neq \emptyset$, then $succ_G(\alpha, t_f) = (M', F')$, where $M' = M - pre(t_f) + post(t_f)$ and F' is computed in three steps:

1) Set F' to $F \wedge \bigwedge_{t \in En(M) \cap G(\alpha)} t_f - t \leq 0 \ \wedge \bigwedge_{t' \in Nw(M,t_f)} \downarrow Is(t') \leq t'^n - t_f \leq \uparrow Is(t')$.
2) Put F' in canonical form and eliminate all transitions of $CF(M, t_f)$.
3) Rename each t'^n in t'.

This firing time condition does not impose any firing order between t_f and transitions outside $G(\alpha)$. Therefore, it holds that $\forall t_f \in T, succ(\alpha, t_f) \subseteq succ_G(\alpha, t_f)$.

—The reduced state space generated by G is the tuple $\mathbb{R} = (G, \mathcal{A}_G, succ_G, \alpha_0)$, where $\mathcal{A}_G = \{\alpha | \alpha_0 \xrightarrow{*}_G \alpha\}$ is the set of reachable abstract states in $\mathbb{R}$ and $\longrightarrow_G$ is the transition relation defined by $\forall \alpha, \alpha' \in \mathcal{A}_C, \forall t_f \in T$,

$$\alpha \xrightarrow{t_f}_G \alpha' \text{ iff } t_f \in G(\alpha) \wedge succ(\alpha, t_f) \neq \emptyset \wedge \alpha' = succ_G(\alpha, t_f).$$

—Let $\alpha \in \mathcal{A}_G$ and $\omega = t_1t_2 \dots t_n$ be a sequence of transitions. We write $\alpha \xrightarrow{\omega}_G \alpha_n$ iff $\exists \alpha_1, \alpha_2, \dots, \alpha_n \in \mathcal{A}_C$ s.t. $\alpha \xrightarrow{t_1}_G \alpha_1 \xrightarrow{t_2}_G \alpha_2 \dots \xrightarrow{t_n}_G \alpha_n$.

For the rest of the article, we fix a TPN $\mathcal{N}$, a sound and complete state-space abstraction $\mathbb{A} = (\mathcal{A}_C, succ, \alpha_0)$, and a reduced state space $\mathbb{R} = (G, \mathcal{A}_G, succ_G, \alpha_0)$.

3.3. Revisited Stubborn Set Method

We propose in the following to adapt the definition of stubborn sets to the context of the TPN state-space abstractions. The revisited definition is based on the notion of partial order successor defined earlier.

Definition 3.5. Let $\alpha \in \mathcal{A}_C$ be an abstract state. $G(\alpha)$ is a stubborn set iff it satisfies the following conditions:

C0: $Fr(\alpha) = \emptyset \Leftrightarrow G(\alpha) \cap Fr(\alpha) = \emptyset$.
C1: $\exists t_f \in G(\alpha) \cap Fr(\alpha), \forall \omega \in (T - G(\alpha))^+,\ succ(\alpha, \omega) \neq \emptyset \ \Rightarrow\ succ(\alpha, \omega t_f) \neq \emptyset$.
C2: $\forall t_f \in G(\alpha), \forall \omega \in (T - G(\alpha))^+$,

$$succ(\alpha, \omega t_f) \neq \emptyset \Rightarrow succ(\alpha, \omega t_f) \subseteq succ(succ_G(\alpha, t_f), \omega).$$

C3: $\forall t_f \in G(\alpha), \forall \omega \in T^+$,

$$succ(succ_G(\alpha, t_f), \omega) \neq \emptyset \ \Rightarrow \exists \omega' \in T^+, \omega' \equiv t_f\omega \wedge succ(\alpha, \omega') \neq \emptyset.$$

Conditions $C0$ and $C1$ are identical to $D0$ and $D1$. Intuitively, $C2$ means that the abstract state reachable by any sequence ωt_f of α can be covered by computing the successor, by ω, of the partial order successor of α by t_f. Condition $C3$ ensures that for any sequence ω firable from $succ_G(\alpha, t_f)$, α has at least a firing sequence, which is equivalent to $t_f\omega$.

We say that $\mathbb{R}$ satisfies $C0$, $C1$, $C2$, and $C3$ (i.e., $\mathbb{R} \models C0 \wedge C1 \wedge C2 \wedge C3$) iff $\forall \alpha \in \mathcal{A}_G, G(\alpha)$ is a stubborn set. The following theorem establishes that stubborn sets preserve deadlocks. In other words, conditions $C0$, $C1$, $C2$, and $C3$ ensure that the reduced state-space $\mathbb{R}$ preserves deadlocks of $\mathbb{A}$ (i.e., of the TPN as $\mathbb{A}$ is supposed to be sound and complete).

THEOREM 3.6. *If* ($\mathbb{R} \models C0 \wedge C1 \wedge C2 \wedge C3$), *then* ($\mathcal{A}_C$ *is deadlock free iff* $\mathcal{A}_G$ *is deadlock free*).

PROOF. Since, from any marking, equivalent firing sequences lead to identical markings, it suffices to show that the following statements (i) and (ii) hold:
(i) $\forall \omega \in T^*, \forall \alpha \in \mathcal{A}$,

$$(\alpha = succ(\alpha_0, \omega) \wedge Fr(\alpha) = \emptyset) \ \Rightarrow\ (\exists \omega' \in T^*, \omega' \equiv \omega \wedge \alpha_0 \xrightarrow{\omega'}_G succ_G(\alpha_0, \omega')).$$

(ii) $\forall \omega \in T^*, \forall \alpha \in \mathcal{A}$,

$$(\alpha_0 \xrightarrow{\omega}_G \alpha \wedge Fr(\alpha) = \emptyset) \ \Rightarrow\ (\exists \omega' \in T^*, \omega' \equiv \omega \wedge succ(\alpha_0, \omega') \neq \emptyset).$$

The proofs of (i) and (ii) are immediate for $\omega = \epsilon$, since both $\mathbb{A}$ and $\mathbb{R}$ have the same initial abstract state $\alpha_0 = succ(\alpha_0, \epsilon) = succ_G(\alpha_0, \epsilon)$.

(1) The proof of (i) is by induction on the length of ω.
  a) For $\omega = t_1$, by assumption, $succ(\alpha_0, t_1)$ is a deadlock and α_0 is not a deadlock.
    —Suppose that $t_1 \in Fr(\alpha_0) - G(\alpha_0)$. According to $C0$ and $C1$, $G(\alpha_0) \cap Fr(\alpha_0) \neq \emptyset$ and $\exists t_2 \in G(\alpha_0) \cap Fr(\alpha_0), succ(\alpha_0, t_1t_2) \neq \emptyset$. The state class $succ(\alpha_0, t_1)$ is not a deadlock, which contradicts the assumption. Therefore, $t_1 \in G(\alpha_0) \cap Fr(\alpha_0)$. Since $succ(\alpha_0, t_1) \neq \emptyset$, it follows that $\alpha_0 \xrightarrow{t_1}_G succ_G(\alpha_0, t_1)$ and then

$succ_G(\alpha_0, t_1) \in \mathcal{A}_G$. The abstract state $succ_G(\alpha_0, t_1)$ is a deadlock, since $succ(\alpha_0, t_1)$ and $succ_G(\alpha_0, t_1)$ share the same marking.

b) Suppose that, for some $n > 1$, the claim (i) is valid for any sequence s.t. its length is less than n and let us show that it is still valid for any sequence ω of length n.

If $\omega \in (T - G(\alpha_0))^+$, then according to $C0$ and $C1$, $G(\alpha_0) \cap Fr(\alpha_0) \neq \emptyset$ and $\exists t_f \in G(\alpha_0) \cap Fr(\alpha_0), succ(\alpha_0, \omega t_f) \neq \emptyset$. It follows that $\alpha = succ(\alpha_0, \omega)$ is not a deadlock, which is in contradiction with the fact that α is a deadlock.

Therefore, ω contains at least a transition of $G(\alpha_0)$, that is, $\omega = \omega_1 t_f \omega_2$ s.t. $\omega_1 \in (T - G(\alpha_0))^+$ and $t_f \in G(\alpha_0)$. Since $succ(\alpha_0, \omega_1 t_f) \neq \emptyset$, using $C2$, we can state that $succ(\alpha_0, \omega_1 t_f) \subseteq succ(succ_G(\alpha_0, t_f), \omega_1) \neq \emptyset$. By assumption, $succ(\alpha_0, \omega_1 t_f \omega_2) \neq \emptyset$. It follows that $succ(\alpha_0, \omega_1 t_f \omega_2) \subseteq succ(succ_G(\alpha_0, t_f), \omega_1\omega_2) \neq \emptyset$. With the fact that $t_f \in G(\alpha_0) \cap Fr(\alpha_0)$, we can state that $succ_G(\alpha_0, t_f)$ is an abstract state of $\mathbb{R}$.

Let $\alpha_f = succ_G(\alpha_0, t_f)$. To achieve the proof, it suffices to show that there exists a sequence ω' equivalent to $\omega_1\omega_2$ and firable in $\mathbb{R}$ from α_f. Since the length of $\omega_1\omega_2$ is $n-1$, the assumption allows us to state that such a sequence exists. Moreover, the marking of the abstract state $succ_G(\alpha_0, \omega')$ of $\mathcal{A}_G$ is M. Since $En(M) = \emptyset$, it follows that $succ_G(\alpha_0, \omega')$ is a deadlock.

(2) a) For $\omega = t_1$, $\alpha_0 \xrightarrow{t_1}_G \alpha$ implies that $\alpha = succ_G(\alpha_0, t_1) \neq \emptyset$. By definition, $succ_G(\alpha_0, t_1) \neq \emptyset$ iff $succ(\alpha_0, t_1) \neq \emptyset$.

b) Suppose that, for some $n > 1$, the claim (ii) is valid for any sequence s.t. its length is less than n and let us show that it is still valid for any sequence ω of length n.

For $\omega = t_1 ... t_{n-1} t_n$, $\alpha_0 \xrightarrow{t_1...t_{n-1}.t_n}_G \alpha$ implies that $\alpha = succ_G(\alpha_0, t_1..t_{n-1}.t_n) \neq \emptyset$. By definition, $succ_G(\alpha_0, t_1 ... t_{n-1} t_n) \neq \emptyset$ iff $succ(succ_G(\alpha_0, t_1 ... t_{n-1}), t_n) \neq \emptyset$.

Let $\alpha_{n-2} = succ_G(\alpha_0, t_1..t_{n-2})$. We rewrite $succ(succ_G(\alpha_0, t_1..t_{n-2}t_{n-1}), t_n) \neq \emptyset$ as follows: $succ(succ_G(\alpha_{n-2}, t_{n-1}), t_n) \neq \emptyset$. By $C3$, we get that $succ(\alpha_{n-2}, t_{n-1}t_n) \neq \emptyset$ or $succ(\alpha_{n-2}, t_n t_{n-1}) \neq \emptyset$. If $n = 2$, then the proof is completed. Otherwise, we repeat the same process by replacing α_{n-2} with $succ_G(\alpha_{n-3}, t_{n-2})$ and so on, until all terms are of the form $succ(\alpha_0, \omega') \neq \emptyset$, where $\omega' \equiv \omega$.

Recall that by assumption, $\alpha = succ_G(\alpha_0, \omega) \neq \emptyset$ is a deadlock. We have shown that $\alpha = succ_G(\alpha_0, \omega) \neq \emptyset$ implies that $\exists \omega' \equiv \omega, succ(\alpha_0, \omega') \neq \emptyset$. Since $succ_G(\alpha_0, \omega')$ and $succ(\alpha_0, \omega)$ have the same marking, it follows that $succ(\alpha_0, \omega')$ is a deadlock.

4. COMPUTING STUBBORN SETS

From the practical point of view, conditions $C1$, $C2$, and $C3$ are not useful to compute G. We need more simple conditions, which allow computing $G(\alpha)$, without exploring beforehand its firing sequences.

Conditions $C0$ and $C1$ are identical to $D0$ and $D1$ established by Valmari [1998] for untimed Petri nets. Several algorithms have been proposed in the literature to compute G holding $D0$, $D1$, and $D2$ in the context of untimed Petri nets. In general, these algorithms infer G from the static structure of the model being verified. We consider here one of the algorithms given in Valmari [1998] and show that it is also valid, in the context of TPN, for $C1$. So, it can be used as a starting point to derive simple algorithms for computing G holding $C0$, $C1$, $C2$, and $C3$.

4.1. Sufficient Condition for $C1$

Valmari and Hansen [2010], and Valmari [1992, 1998] have proposed some useful sufficient conditions for $D1$ and $D2$ in the context of untimed Petri nets. We consider here the strongest one given in Valmari [1998], referred to here as $SC1$. For $\mathbb{R}$, $SC1$ can be stated as follows: $\mathbb{R} \models SC1$ iff: $\forall \alpha = (M, F) \in \mathcal{A}_G, \forall t \in G(\alpha)$,

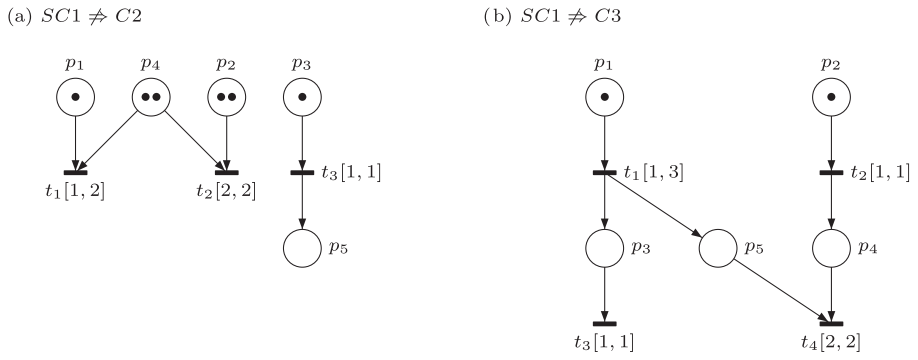

Fig. 3. Counterexamples for $SC1 \Rightarrow C2$ and $SC1 \Rightarrow C3$.

1: $t \notin En(M) \Rightarrow \exists p \in {}^{\circ}t, M(p) < pre(t)(p) \wedge {}^{\circ}p \subseteq G(\alpha)$
2: $t \in En(M) \Rightarrow ({}^{\circ}t)^{\circ} \subseteq G(\alpha)$.

Intuitively, part 1 of $SC1$ ensures that the disabled transitions of $G(\alpha)$ cannot be enabled before firing some transition of $G(\alpha)$. Part 2 of $SC1$ means that if $G(\alpha)$ includes an enabled transition t, then it also includes its structurally conflicting transitions. Lemma 4.1 states that $SC1$ implies $C1$ in $\mathbb{R}$.

LEMMA 4.1. $\mathbb{R} \models SC1 \Rightarrow \mathbb{R} \models C1$.

PROOF. Suppose that $SC1$ holds in $\mathbb{R}$ but $C1$ does not hold in $\mathbb{R}$, that is,

$$\exists \alpha \in \mathcal{A}_G, \forall t_f \in G(\alpha) \cap Fr(\alpha), \exists \omega \in (T - G(\alpha))^+, succ(\alpha, \omega) \neq \emptyset \ \wedge \ succ(\alpha, \omega t_f) = \emptyset.$$

It means that there is at least a transition t in conflict with t_f (i.e., $t \in ({}^{\circ}t_f)^{\circ}$). This transition is either in ω or enabled by some transition of ω. Since $t_f \in G(\alpha) \cap Fr(\alpha)$, according to part 2 of $SC1$, $t \in G(\alpha)$. If t belongs to ω, then $t \in T - G(\alpha)$, which is in contradiction with $t \in G(\alpha)$. Otherwise, $t \in G(\alpha) - En(M)$ is enabled directly or indirectly by some transitions of ω. According to part 1 of $SC1$, t cannot be enabled before firing some transition of $G(\alpha)$. It means that there is at least a transition in ω belonging to $G(\alpha)$, which is in contradiction with $\omega \in (T - G(\alpha))^+$. □

Let us now show, by means of two examples, that $SC1$ implies neither $C2$ nor $C3$ in $\mathbb{R}$.

LEMMA 4.2. 1) $\mathbb{R} \models SC1 \not\Rightarrow \mathbb{R} \models C2$ *and* 2) $\mathbb{R} \models SC1 \not\Rightarrow \mathbb{R} \models C3$.

PROOF. The proof of 1) and 2) consists of two counterexamples. We use here the successor function of the CSCG to process the counterexamples. Note, however, that the counterexamples are also valid for other sound and complete state-space abstractions such as the ZBG, the SCG, and the SSCG.

1) Consider the TPN in Figure 3(a) and its initial abstract state $\alpha_0 = (p_1 + 2p_2 + p_3 + 2p_4, -1 \leq t_1 - t_2 \leq 0 \wedge 0 \leq t_1 - t_3 \leq 1 \wedge t_2 - t_3 = 1)$. The set $G(\alpha_0) = \{t_1, t_2\}$ satisfies $SC1$, since $t_1, t_2 \in En(M_0)$ and $({}^{\circ}t_1)^{\circ} = \{t_1, t_2\} = ({}^{\circ}t_2)^{\circ}$. But $C2$ does not hold for $G(\alpha_0)$. Indeed, for $t_3 \notin G(\alpha_0)$, $t_2 \in G(\alpha_0)$, it holds that $succ(\alpha_0, t_3t_2) = (p_1 + p_2 + p_4 + p_5, t_1 - t_2 = -2)$, $succ(\alpha_0, t_2) = \emptyset$ (the transition t_3 must fire before t_2 from α_0 $(0 < t_2 - t_3 = 1)$), and then, by definition, $succ_G(\alpha_0, t_2) = \emptyset$. It follows that $succ(succ_G(\alpha_0, t_2), t_3) = \emptyset$ and $succ(\alpha_0, t_3t_2) \not\subseteq succ(succ_G(\alpha_0, t_2), t_3)$.

2) Consider the TPN in Figure 3(b) and its initial abstract state $\alpha_0 = (p_1 + p_2, 0 \leq t_1 - t_2 \leq 2)$. Suppose that $G(\alpha_0) = \{t_1\}$. Condition $SC1$ holds for $G(\alpha_0)$, since $t_1 \in En(M_0)$ and $({}^{\circ}t_1)^{\circ} = \{t_1\} = G(\alpha_0)$. But $C3$ does not hold for $G(\alpha_0)$, because $succ(succ_G(\alpha_0, t_1), t_2t_4) \neq \emptyset$ and α_0 has no firable sequence equivalent to $t_1t_2t_4$. Indeed, t_4 is not enabled in

α_0, $succ(\alpha_0, t_2)$, and $succ(\alpha_0, t_1)$. It is enabled but not firable from $succ(\alpha_0, t_1t_2)$ and $succ(\alpha_0, t_2t_1)$. For the sequence t_1t_2, transitions t_3 and t_4 are enabled by t_1 and t_2, respectively. Its firing leads to $succ(\alpha_0, t_1t_2) = (p_3 + p_4 + p_5, t_4 - t_3 = 1 > 0)$, from which t_4 cannot fire before t_3, that is, $succ(\alpha_0, t_1t_2t_4) = \emptyset$. For the sequence t_2t_1, transitions t_3 and t_4 are both enabled by t_1. Its firing leads to $succ(\alpha_0, t_2t_1) = (p_3 + p_4 + p_5, t_4 - t_3 = 1)$. However, it holds that $succ(succ_G(\alpha_0, t_1), t_2) = (p_3 + p_4 + p_5, -1 \leq t_4 - t_3 \leq 1)$ and $succ(succ_G(\alpha_0, t_1), t_2t_4) = (p_3, true) \neq \emptyset$. $\square$

4.2. Sufficient Conditions for *C*1, *C*2, and *C*3

We propose to strengthen the condition $SC1$ so as to meet $C2$ and $C3$. The strengthened condition, denoted $SC2$, must at least prevent the counterexamples for $C2$ and $C3$, given in the proof of Lemma 4.2. For $C2$, we enforce $SC1$ by imposing that each enabled transition of $G(\alpha)$ is firable before all transitions outside $G(\alpha)$. The aim of this extra condition is to prevent situations where a sequence ωt_f is firable from $\alpha = (M, F)$ but the sequence $t_f\omega$ is not firable from α, ω being any sequence of $(T - G(\alpha))^+$ and $t_f \in G(\alpha) \cap En(M)$. For $C3$, we enforce $SC1$ so that the effect[8] of transitions of $G(\alpha)$ is not altered by firing any transition outside $G(\alpha)$ and vice versa. So, we impose in $SC2$ that the transitions of $G(\alpha)$ can neither be enabled nor disabled by any transition outside $G(\alpha)$, and also the input and output transitions of each firable transition of $G(\alpha)$ are inside $G(\alpha)$.

Formally, $SC2$ is defined by $\mathbb{R} \models SC2$ iff $\forall \alpha = (M, F) \in \mathcal{A}_G, \forall t \in G(\alpha)$,

1: $\forall p \in {}^\circ t, M(p) < pre(t)(p) \wedge {}^\circ p \subseteq G(\alpha)$.
2: $\forall p \in {}^\circ t, M(p) \geq pre(t)(p) \wedge p^\circ \subseteq G(\alpha)$.
3: $t \in En(M) - Fr(\alpha) \Rightarrow \forall t' \in Fr(\alpha)$ s.t. $(F \wedge t' < t) \equiv F, t' \in G(\alpha)$.
4: $t \in Fr(\alpha) \Rightarrow ({}^\circ({}^\circ t) \cup (t^\circ)^\circ) \subseteq G(\alpha)$.

We first establish, in Lemma 4.3, some useful relations induced by $SC2$. Afterward, we prove, in Lemma 4.4, that $SC2$ is a sufficient condition for $SC1$, $C2$, and $C3$. Condition $SC2$ is then useful to compute TPN reduced graphs, preserving deadlocks.

LEMMA 4.3. *Let $\alpha = (M, F) \in \mathcal{A}_G$, $t_f \in G(\alpha)$, and $\omega = t_1...t_n \in (T - G(\alpha))^+$ s.t. $succ(succ_G(\alpha, t_f), \omega) \neq \emptyset$. Then, $G(\alpha)$ satisfies $SC2$ and implies that:*

1: $\forall i \in [1, n] \cup \{f\}, (CF(M_i, t_f) \cup Nw(M_i, t_f)) \subseteq G(\alpha)$,
2: $\forall i \in [1, n], G(\alpha) \cap (CF(M_i, t_i) \cup Nw(M_i, t_i)) = \emptyset$,
3: $\forall i \in [2, n] \cup \{f\}$, 1) $CF(M, t_f) = CF(M_i, t_f)$ and 2) $Nw(M, t_f) = Nw(M_i, t_f)$,
4: $\forall i \in [1, n]$, 1) $CF(M_i, t_i) = CF(M_{f_i}, t_i)$ and 2) $Nw(M_i, t_i) = Nw(M_{f_i}, t_i)$,
5: $\forall i \in [1, n] \cup \{f\}, En(M) \cap G(\alpha) = En(M_i) \cap G(\alpha)$,

where $\alpha = \alpha_1 = (M_1, F_1)$,
$\alpha_i = (M_i, F_i) = succ(\alpha_{i-1}, t_{i-1})$, for $i \in [2, n]$, and $\alpha_f = (M_f, F_f) = succ(\alpha_n, t_n)$ and $\alpha_{f_1} = (M_{f_1}, F_{f_1}) = succ(\alpha, t_f)$, $\alpha_{f_i} = (M_{f_i}, F_{f_i}) = succ(\alpha_{f_i - 1}, t_{i-1})$, for $i \in [2, n]$.

PROOF. By assumption, (i) $G(\alpha)$ satisfies $SC2$, (ii) $t_f \in G(\alpha) \cap Fr(\alpha)$, (iii) $\omega \in (T - G(\alpha))^+$, and (iv) $succ(succ_G(\alpha, t_f), \omega) \neq \emptyset$.

1: The proof of $(CF(M_i, t_f) \cup Nw(M_i, t_f)) \subseteq G(\alpha)$ is immediate from the fact that $t_f \in G(\alpha) \cap Fr(\alpha)$ and $(CF(M_i, t_f) \cup Nw(M_i, t_f)) \subseteq ({}^\circ t_f)^\circ \cup (t_f^\circ)^\circ \subseteq G(\alpha)$.
2: Suppose that $\exists t_k \in G(\alpha)$, s.t. $t_k \in CF(M_i, t_i)$, which implies that $t_i \in ({}^\circ t_k)^\circ$. If $t_k \in En(M)$, then by $SC2$, $t_i \in G(\alpha)$, which contradicts (iii). Otherwise, by $SC2$, t_k cannot be enabled before firing at least a transition of $G(\alpha)$. In other words, the sequence

[8]The effect of a transition t is defined by two sets: the set of transitions disabled by t and the set of transitions enabled by t.

$t_1 ... t_{i-1}$ contains at least a transition from $G(\alpha)$, which contradicts (iii). Suppose now that $\exists t_k \in G(\alpha), t_k \in Nw(M_i, t_i)$, which implies that $t_i \in {}^\circ({}^\circ t_k)$. If $\exists t_l \in ||t_1 ... t_i||, \exists p \in {}^\circ t_k \cap t_l^\circ, M(p) < pre(t_k)(p)$, then by $SC2$, $t_l \in G(\alpha)$, which contradicts (iii). Otherwise, as $t_k \in Nw(M_i, t_i)$, it follows that there is at least a transition t_l in $t_1 ... t_i$ s.t. $\exists p \in {}^\circ t_l \cap {}^\circ t_k \cap t_i^\circ$. By $SC2$, t_l or t_i belongs to $G(\alpha)$, which contradicts (iii).

3: If $CF(M, t_f) \neq CF(M_i, t_f)$ or $Nw(M, t_f) \neq Nw(M_i, t_f)$, then it implies that some transition t_l of $t_1 ... t_{i-1}$ will add or remove at least a transition t_k from $CF(M, t_f)$ or $Nw(M, t_f)$. Therefore, $t_k \in ({}^\circ t_f)^\circ \cup (t_f^\circ)^\circ$ and at least one of the following relations holds:
a) $\exists p \in {}^\circ t_k \cap {}^\circ t_l, M(p) \geq pre(t_k)(p) \vee p \in {}^\circ t_f \cup t_f^\circ$ or
b) $\exists p \in {}^\circ t_k \cap t_l^\circ, M(p) < pre(t_k)(p) \vee p \in {}^\circ t_f \cup t_f^\circ$.
Since $t_k \in ({}^\circ t_f)^\circ \cup (t_f^\circ)^\circ$ and $t_f \in G(\alpha) \cap Fr(\alpha)$, by $SC2$, it follows that $t_k \in ({}^\circ t_f)^\circ \cup (t_f^\circ)^\circ \subseteq G(\alpha)$. With $t_k \in G(\alpha)$, each of the relations a) and b) implies, by $SC2$, that $t_l \in G(\alpha)$, which contradicts (iii).

4: It holds that $M_{f_i}(p) = M_i(p) + post(t_f)(p) - pre(t_f)(p)$ and $M_{f_i} = M_{i_f}$, M_{i_f} being the marking reached by the sequence $t_1 ... t_{i-1} t_f$. Therefore, $CF(M_{f_i}, t_i) = CF(M_{i_f}, t_i)$ and $Nw(M_{f_i}, t_i) = Nw(M_{i_f}, t_i)$. Suppose that $CF(M_i, t_i) \neq CF(M_{i_f}, t_i)$ or $Nw(M_i, t_i) \neq Nw(M_{i_f}, t_i)$. It implies that t_f will add or remove at least a transition t_k from $CF(M_i, t_i)$ or $Nw(M_i, t_i)$. Therefore, $t_k \in ({}^\circ t_f)^\circ \cup (t_f^\circ)^\circ$ and at least one of the following relations holds:
a) $\exists p \in {}^\circ t_k \cap {}^\circ t_i, M(p) \geq pre(t_k)(p) \vee p \in {}^\circ t_f \cup t_f^\circ$ or
b) $\exists p \in {}^\circ t_k \cap t_i^\circ, M(p) < pre(t_k)(p) \vee p \in {}^\circ t_f \cup t_f^\circ$.
Relations $t_k \in ({}^\circ t_f)^\circ \cup (t_f^\circ)^\circ$ and $t_f \in G(\alpha) \cap Fr(\alpha)$ imply, by $SC2$, that $t_k \in ({}^\circ t_f)^\circ \cup (t_f^\circ)^\circ \subseteq G(\alpha)$. With $t_k \in G(\alpha)$, each of the relations a) and b) implies, by $SC2$, that $t_i \in G(\alpha)$, which contradicts (iii).

5: The proof is immediate from 2 : $\forall i \in [1, n], G(\alpha) \cap (CF(M_i, t_i) \cup Nw(M_i, t_i)) = \emptyset$, which means that all transitions enabled or disabled by any transition outside $G(\alpha)$ do not belong to $G(\alpha)$, and the fact that $En(M_i) = (En(M_{i-1}) - CF(M_{i-1}, t_{i-1})) + Nw(M_{i-1}, t_{i-1})$, for $i \in [1, n]$. □

LEMMA 4.4. *1)* $\mathbb{R} \models SC2 \Rightarrow \mathbb{R} \models SC1$. *and 2)* $\mathbb{R} \models SC2 \Rightarrow \mathbb{R} \models C2 \wedge C3$.

PROOF. 1) Let $\alpha = (M, F) \in \mathcal{A}_C$ be a state class and $t \in T$ a transition. To achieve the proof, it suffices to show that if $SC2$ holds for α and t, then $SC1$ holds for α and t too. We consider two cases: $t \in En(M)$ and $t \notin En(M)$.
a) If $t \in En(M)$, then by definition $\forall p \in^\circ t, M(p) \geq pre(t)(p)$. In this case, Condition 2 of $SC2$ implies Condition 2 of $SC1$.
b) Otherwise, it holds that $\exists p \in {}^\circ t, M(p) < pre(t)(p)$ and then (by Condition 1 of $SC2$), ${}^\circ p \subseteq G(\alpha)$. In other terms, t satisfies condition 1 of $SC1$. Therefore, $\mathbb{R} \models SC2 \Rightarrow \mathbb{R} \models SC1$.
2) Let us first recall the definition of $C1$ and $C2$:

$$C2 : \forall t_f \in G(\alpha), \forall \omega \in (T - G(\alpha))^+, succ(\alpha, \omega t_f) \neq \emptyset \Rightarrow succ(\alpha, \omega t_f) \subseteq succ(succ_G(\alpha, t_f), \omega).$$

$$C3 : \forall t_f \in G(\alpha), \forall \omega \in T^+, succ(succ_G(\alpha, t_f), \omega) \neq \emptyset \Rightarrow \exists \omega' \equiv t_f \omega, succ(\alpha, \omega') \neq \emptyset.$$

Without loss of generality, for economy of notations, we suppose that all transitions of ω are different. By definition, the firing time condition of $succ(\alpha, \omega t_f)$ from α, denoted ψ, is

$$F \wedge \bigwedge_{i \in [1,n] \cap \{f\}} \left[\bigwedge_{t_j \in En(M_i)} t_i \leq t_j \wedge \bigwedge_{t_k \in Nw(M_i, t_i)} \downarrow Is(t_k) \leq t_k^i - t_i \leq \uparrow Is(t_k) \right],$$

where for $i \in [1, n] \cup \{f\}$, t_k^i denotes the instance of t_k enabled by firing t_i from M_i.

By definition, for $i \in [1, n-1]$, $En(M_{i+1}) = (En(M_i) - CF(M_i, t_i)) + Nw(M_i, t_i)$ and $En(M_f) = (En(M_n) - CF(M_n, t_n)) + Nw(M_n, t_n)$. Therefore, ψ implies that $\forall i \in [1, n-1]$, $\forall t_j \in En(M_{i+1})$, $\forall t_k \in Nw(M_i, t_i)$, $t_i \leq t_{i+1} \leq t_j \wedge 0 \leq \downarrow Is(t_k) \leq t_k^i - t_i$. Consequently, there are in ψ some redundant constraints and the elimination of some of these redundant constraints will not affect the domain of ψ. So, ψ is equivalent to the following formula:

$$F \wedge t_1 \leq t_2 \leq \cdots \leq t_n \leq t_f$$
$$\wedge \bigwedge_{i \in [1,n] \cup \{f\}} \left[\bigwedge_{t_j \in CF(M_i, t_i)} t_i \leq t_j \;\wedge \bigwedge_{t_k \in Nw(M_i, t_i)} \downarrow Is(t_k) \leq t_k^i - t_i \leq \uparrow Is(t_k) \right]$$
$$\wedge \bigwedge_{t_l \in En(M_{\omega t_f})} t_n \leq t_l \wedge \bigwedge_{t_l \in En(M_{\omega t_f})} t_f \leq t_l,$$

where $M_{\omega t_f}$ is the successor marking of M by ωt_f.

By assumption, $t_f \in G(\alpha)$, $\omega \in (T - G(\alpha))^+$ and $succ(\alpha, \omega t_f) \neq \emptyset$ (which implies that ψ is consistent). According to part 3 of $SC2$, t_f is firable before all transitions of ω from α. Since the first transition of ω is firable from α, it follows that t_f is firable from α, that is, $succ(\alpha, t_f) \neq \emptyset$. Let $\omega = \omega_1\omega_2$, with $\omega_1, \omega_2 \in (T - G(\alpha))^*$. We denote $M_{\omega_1 t_f \omega_2}$ the successor marking of M by $\omega_1 t_f \omega_2$. It holds that $M_{\omega t_f} = M_{\omega_1 t_f \omega_2}$ and $En(M_{\omega t_f}) = En(M_{\omega_1 t_f \omega_2})$.

According to Lemma 4.3, the effect of each transition of ωt_f is the same for all sequences $\omega_1 t_f \omega_2$. It follows that the formula ϕ obtained from ψ by eliminating the constraint $t_n \leq t_f$ is consistent and equivalent to the firing time condition of

$$\beta = \bigcup_{\omega_1, \omega_2 \in (T - G(\alpha))^* \text{ s.t. } \omega = \omega_1\omega_2} succ(\alpha, \omega_1 t_f \omega_2).$$

According to Lemma 4.3, for $i \in [1, n] \cup \{f\}$, $En(M_i) \cap G(\alpha) = En(M) \cap G(\alpha)$, $(CF(M_i, t_i) \cup Nw(M_i, t_i)) \cap G(\alpha) = \emptyset$, $(CF(M, t_f) \cup Nw(M, t_f)) \subseteq G(\alpha)$, $CF(M, t_f) = CF(M_i, t_f)$, and $Nw(M, t_f) = Nw(M_i, t_f)$. Therefore, $En(M) \cap G(\alpha) = (En(M_{\omega t_f}) \cap G(\alpha)) \cup CF(M, t_f) \cup Nw(M, t_f)$. The formula φ derived from ϕ by eliminating the constraints $\bigwedge_{t_l \in En(M_{\omega t_f}) - G(\alpha)} t_f \leq t_l$ is the firing time condition of $succ(succ_G(\alpha, t_f), \omega)$:

$$F \wedge t_1 \leq t_2 \leq \cdots \leq t_n$$
$$\wedge \bigwedge_{i \in [1,n] \cup \{f\}} \left[\bigwedge_{t_j \in CF(M_i, t_i)} t_i \leq t_j \;\wedge \bigwedge_{t_k \in Nw(M_i, t_i)} \downarrow Is(t_k) \leq t_k^i - t_i \leq \uparrow Is(t_k) \right]$$
$$\wedge \bigwedge_{t_l \in En(M_{\omega t_f})} t_n \leq t_l \wedge \bigwedge_{t_l \in En(M_{\omega t_f}) \cap G(\alpha)} t_f \leq t_l.$$

It follows that $succ(\alpha, \omega t_f) \subseteq \beta \subseteq succ(succ_G(\alpha, t_f), \omega)$ and then $\mathbb{R} \models SC2 \Rightarrow \mathbb{R} \models C2$. Let us now show that $\mathbb{R} \models SC2 \Rightarrow \mathbb{R} \models C3$. We consider three cases: (a) $\omega = \epsilon$, (b) $\omega \in (T - G(\alpha))^+$, and (c) $\omega t_k \omega'$ with $\omega \in (T - G(\alpha))^*$, $t_k \in G(\alpha)$, and $\omega' \in T^*$.
a) For $\omega = \epsilon$, $succ(succ_G(\alpha, t_f), \omega) = succ_G(\alpha, t_f)$. By definition, $succ_G(\alpha, t_f) \neq \emptyset$ iff $succ(\alpha, t_f) \neq \emptyset$.
b) For $\omega \in (T - G(\alpha))^+$, consider the firing time condition ϕ of β given previously and suppose that $\beta = \emptyset$ and $succ(succ_G(\alpha, t_f), \omega) \neq \emptyset$ (φ is consistent but ϕ is inconsistent). It means that $succ(\alpha, t_f) \neq \emptyset$ and also some transition t_k of $En(M_{\omega t_f}) - G(\alpha)$ must fire before t_f (i.e., $succ(\alpha, t_f) = \emptyset$), which is not possible. Therefore, that φ is consistent implies that ϕ is consistent and then for $t_f \in G(\alpha)$, $\omega \in (T - G(\alpha))^+$, $succ(succ_G(\alpha, t_f), \omega) \neq \emptyset \Rightarrow \exists \omega_1, \omega_2 \in (T - G(\alpha))^*$ s.t. $\omega = \omega_1\omega_2 \wedge succ(\alpha, \omega_1 t_f \omega_2) \neq \emptyset$.

ALGORITHM 1: Computation of $G(\alpha)$ holding $C0 \wedge SC2$

```
Input: A TPN $\mathcal{N} = (P, T, pre, post, Is, M_0)$ and a state class $\alpha = (M, F)$.
Output: The set $G(\alpha)$.
ChooseRandomly $t$ in $Fr(\alpha)$;
$X = \{t\}$;
repeat
    $Y = X$;
    for each transition $t \in Y$ do
        for each place $p \in {}^\circ t$ do
            if $(M(p) < pre(t)(p))$ then
                $X = X \cup {}^\circ p$;
            end
            if $(M(p) \geq pre(t)(p))$ then
                $X = X \cup p^\circ$;
            end
        end
        if $(t \in En(M))$ then
            for each transition $t' \in Fr(\alpha) - X s.t.\ (F \wedge t' < t \equiv F)$ do
                $X = X \cup \{t'\}$;
            end
        end
        if $(t \in Fr(\alpha))$ then
            for each place $p \in t^\circ$ do
                $X = X \cup p^\circ$;
            end
            for each place $p \in {}^\circ t$ do
                $X = X \cup {}^\circ p$;
            end
        end
    end
until $X = Y$;
$G(\alpha) = X$;
```

c) For $\omega t_k \omega'$, with $\omega \in (T - G(\alpha))^+$, $t_k \in G(\alpha)$, and $\omega' \in T^*$, by assumption, $succ(succ_G(\alpha, t_f), \omega t_k \omega') \neq \emptyset$ and then $succ(succ_G(\alpha, t_f), \omega t_k) \neq \emptyset$. The firing time condition γ of $succ(succ_G(\alpha, t_f), \omega t_k)$ is equivalent to $\varphi \wedge \bigwedge_{t_l \in En(M_{\omega t_f})} t_k \leq t_l$. Since $t_k \in G(\alpha)$, it follows that φ implies $t_f \leq t_k$ and then adding redundant constraints $\bigwedge_{t_l \in En(M_{\omega t_f}) - G(\alpha)} t_f \leq t_l$ to γ will not affect its domain. Therefore, γ is equivalent to $\varphi \wedge \bigwedge_{t_l \in En(M_{\omega t_f})} t_k \leq t_l \wedge \bigwedge_{t_l \in En(M_{\omega t_f}) - G(\alpha)} t_f \leq t_l$, which is, in turn, equivalent to $\phi \wedge \bigwedge_{t_l \in En(M_{\omega t_f})} t_k \leq t_l$. Consequently, γ is equivalent to the firing time condition of $succ(\beta, t_k)$. It follows that $succ(succ_G(\alpha, t_f), \omega t_k) = succ(\beta, t_k)$ and then $succ(succ_G(\alpha, t_f), \omega t_k \omega') \neq \emptyset \Rightarrow \exists \omega'' \equiv t_f \omega t_k \omega', succ(\alpha, \omega'') \neq \emptyset$. $\square$

We report in Algorithm 1 a computing procedure of $G(\alpha)$, which satisfies $SC2$ and $C0$. Algorithm 2 verifies on the fly whether a given TPN is deadlock free or not. It computes the state classes of the RSCG until a deadlock is detected or all the RSCG state classes are processed.

Example 4.5. Consider the model shown at Figure 1(b) and its initial state class $\alpha_0 = (p_1 + p_2, -2 \leq t_1 - t_2 \leq 0)$. The set $G(\alpha_0)$ depends on the firable transition chosen first. If $G(\alpha_0)$ is initially set to t_2, applying recursively rules 1, 2, 3, and 4 of $SC2$ will add only the transition t_4 to $G(\alpha_0)$, that is, $G(\alpha_0) = \{t_2, t_4\}$. The transition t_2 is the only

ALGORITHM 2: Verification of $\mathcal{N} \models AG\, not\ deadlock$

```
Input: A TPN $\mathcal{N} = (P, T, pre, post, Is, M_0)$.
Output: $true$ for deadlock free, $false$ otherwise
$\alpha_0 = (M_0, \bigwedge_{t,t' \in En(M_0)} t - t' \leq \uparrow Is(t) - \downarrow Is(t'))$;
if ($\alpha_0$ is a deadlock) then
    return false;
end
$\mathcal{W} = \{\alpha_0\}$;
$A = \emptyset$;
while $W \neq \emptyset$ do
    $\alpha$ = getStateClass(W);
    $W = W - \{\alpha\}$;
    $A = A \cup \{\alpha\}$;
    for each transition $t$ in $G(\alpha) \cap Fr(\alpha)$ do
        $\alpha' = succ_G(\alpha, t)$;
        if ($\alpha'$ is a deadlock) then
            return false;
        end
        if ($\alpha' \notin W \cup A$) then
            $W = W \cup \{\alpha'\}$;
        end
    end
    return true;
end
```

transition of $G(\alpha_0)$, which is firable from α_0. The partial order successor of α_0 by t_2 is $\alpha'_1 = succ_G(\alpha_0, t_2) = (p_1 + p_4, -3 \leq t_1 - t_4 \leq -1)$.

For α'_1, if $G(\alpha'_1)$ is initially set to t_1, then rules 1, 2, 3, and 4 of $SC2$ will add only the transition t_3 to $G(\alpha'_1)$, that is, $G(\alpha'_1) = \{t_1, t_3\}$, where only t_1 is firable from α'_1. The partial order successor of α'_1 by t_1 is $\alpha'_2 = succ_G(\alpha'_1, t_1) = (p_3 + p_4, -1 \leq t_3 - t_4 \leq 1)$.

For α'_2, if $G(\alpha'_2)$ is initially set to t_3, then rules 1, 2, 3, and 4 of $SC2$ will add no transition, that is, $G(\alpha'_2) = \{t_3\}$. The partial order successor of α'_2 by t_3 is $\alpha'_3 = (p_4, true)$. From α'_3, the firing of the sole enabled and firable transition t_4 leads to $\alpha'_4 = (0, true)$. The RSCG is shown at Figure 2(c).

4.3. RSCG Preserves k-Boundedness of Places

Let us first show, by means of an example, that the k-boundedness of places is not preserved by $SC1$. Consider the model at Figure 4 and the run $\alpha_0 \xrightarrow{t_1} \alpha_1 \xrightarrow{t_3} \alpha_2 \xrightarrow{t_2} \alpha_3 \xrightarrow{t_4} \alpha_4$ feasible from its initial state class $\alpha_0 = (p_1 + p_2 + p_5, 0 \leq t_1 - t_2)$. It holds that $G(\alpha_0) = \{t_1, t_3\}$, $G(\alpha_1) = \{t_1, t_3\}$, $G(\alpha_2) = \{t_2\}$, and $G(\alpha_3) = \{t_4\}$ satisfy $SC1$. This run constitutes a reduced graph w.r.t. $SC1$. The place p_5 is 1-bounded in the reduced graph, whereas it is 2-bounded in the model, since the marking $p_1 + p_4 + 2p_5$ is reachable from the initial marking by t_2.

We establish in Theorem 4.6 that the k-boundedness of each place of $\mathcal{N}$ is preserved in the RSCG.

THEOREM 4.6. *The RSCG preserves k-boundedness of each place.*

PROOF (SKETCH OF THE PROOF). In the proof of Theorem 3.6, it is shown that the reduced state spaces of a TPN, holding conditions $C0$, $C1$, $C2$, and $C3$, preserve nonequivalent firing sequences of the TPN (i.e., for any maximal firing sequence in the TPN state space, there is an equivalent maximal firing sequence in the reduced state space and

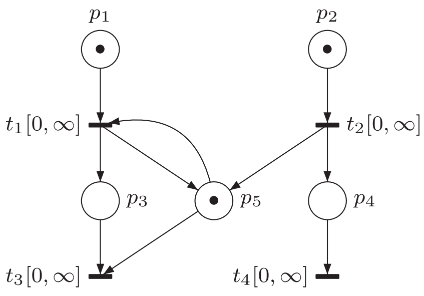

Fig. 4. $SC1$ does not preserve k-boundedness of each place.

vice versa). Recall that the RSCG satisfies $C0 \wedge SC2$. Since $SC2$ is a sufficient condition for $C1 \wedge C2 \wedge C3$, the RSCG preserves nonequivalent firing sequences of the TPN. In other words, for any maximal sequence ω of the CSCG, there is a maximal sequence ω' in the RSCG, which can be obtained from ω by a series of permutations of transitions and vice versa. Moreover, by construction of the CSCG, for each pair of such permutable transitions (t_1, t_2), there is at least a state class α reachable by a prefix of ω s.t. $t_1 \in G(\alpha) \cap Fr(\alpha)$ and $t_2 \notin G(\alpha)$. The firing order between t_1 of $G(\alpha)$ and any other transition of $G(\alpha)$ is fixed in $succ_G(\alpha, t_1)$, but no firing order between t_1 and t_2 is fixed. By $SC2$, it holds that ${}^\circ t_1 \cap ({}^\circ t_2 \cup t_2^\circ) = \emptyset$ and ${}^\circ t_2 \cap (t_1^\circ \cup {}^\circ t_1) = \emptyset$.

Therefore, the firing of t_1 (resp. t_2) from any marking M will not decrease the markings of the input and output places of t_2 (resp. t_1), that is,
$\forall p \in {}^\circ t_2 \cup t_2^\circ, M(p) \leq M_1(p) = M(p) + post(t_1)(p), M'(p) = M_2(p) + post(t_1)(p)$ and
$\forall p \in {}^\circ t_1 \cup t_1^\circ, M(p) \leq M_2(p) = M(p) + post(t_2)(p), M'(p) = M_1(p) + post(t_2)(p)$, where M_1, M_2, and M' are the successor markings of M by t_1, t_2, and t_1t_2, respectively. Moreover, $\forall p \in P, (post(t_2)(p) > 0 \Rightarrow pre(t_1)(p) = 0)$ and $(post(t_1)(p) > 0 \Rightarrow pre(t_2)(p) = 0)$. Consequently, $\forall p \in P, Max(M(p), M_1(p), M'(p)) = Max(M(p), M_2(p), M'(p))$. So, the bound of each place can be retrieved from the RSCG. □

5. EXPERIMENTAL RESULTS

We have implemented and tested the partial order technique proposed here on several small TPNs, the model SP at Figure 5, and two models taken from the MCC (Model Checking Contest) held within Petri Nets 2013:[9] HouseConstruction (HC in short) and FMS. Table II reports the sizes and computing times of the RSCG and CSCG for different markings and variants and of HC, FMS, and SP. More precisely, for each tested model, Table II shows the number of state classes (NSC), the number of computed state classes (NCSC), and the CPU time in seconds, obtained for the RSCG and the CSCG, respectively. An interrogation mark indicates a situation where the computation has not been completed after 1 hour.

The model $HC(n)$ is a free-choice and connected TPN, where n is the initial marking of the source place p_1.[10] The model $FMS(n)$ is a strongly connected TPN,[11] where n is the initial marking of places p_1, p_2, and p_3. The model $SP(n, m, o)$, shown in Figure 5, is a connected TPN with conflicting transitions, synchronization, and shared resources, representing the production system studied in Delgadillo and Llano [2006]. In the first variant (var. 1) of HC, FMS, and SP, the upper bounds of all the static firing intervals are set to ∞, which implies that the number of reachable state classes is equal to the number of reachable markings. It allows one to evaluate the gain in terms

[9]http://mcc.lip6.fr.
[10]In a connected TPN, there is a undirected path between every two nodes (places or transitions).
[11]In a strongly connected TPN, there is a directed path between every two nodes (places or transitions).

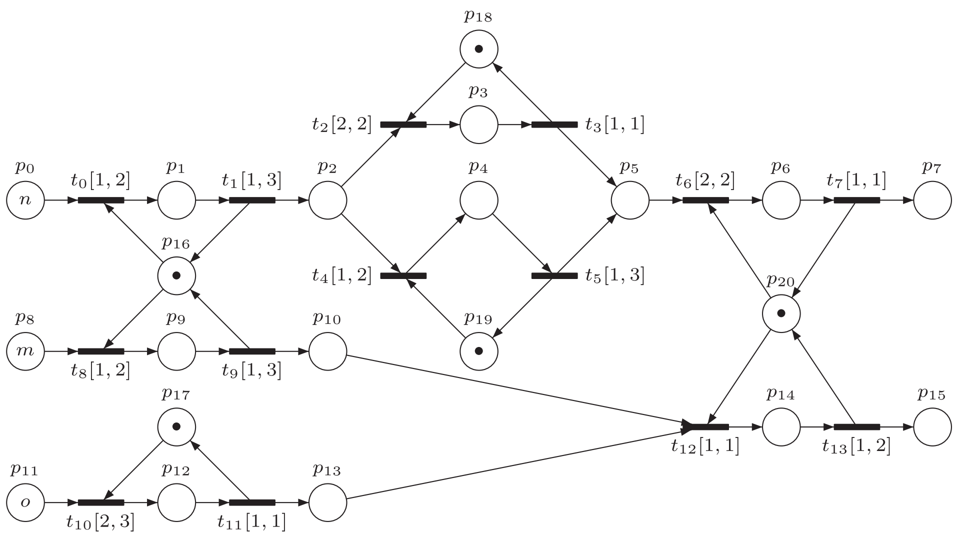

Fig. 5. The model SP(n,m,o) taken from Delgadillo and Llano [2006].

of markings. In the second variant (var. 2), the static firing intervals of transitions are all bounded. Note that $FMS(n)$ is deadlock free but $HC(n)$ and $SP(n, m, o)$ are not deadlock free. For these models, the verification of the deadlock property, using a breadth-first exploration, results in computing all the CSCG or RSCG state classes. Therefore, the results reported in Table II are also valid, in case the exploration is stopped as soon as a deadlock state is detected (i.e., Algorithm 2). Finally, we tested another variant (var. 3) of HC obtained from its second one by adding a transition t_{19} as an output of place p_6 ($pre(t_{19}) = p_6$, $post(t_{19}) = 0$, $Is(t_{19}) = [1, 1]$). We report at the end of Table II the number of computed state classes before reaching a deadlock state class.

RSCG allows a significant reduction in time and number of computed state classes compared to the CSCG (the NCSC of the RSCG is up to 20 times smaller than the NCSC of the CSCG in some cases). As for the partial order techniques of untimed Petri nets, the gain (in time and space) of the RSCG over the CSCG is much more significant for the TPNs with a higher number of independent transitions. Indeed, the connected TPNs ($HC(n)$ and $SP(n, m, o)$) show better reduction than the strongly connected TPN ($FMS(n)$). Furthermore, we obtain further reduction when we increase the marking, as it results in increasing the number of concurrent enabled transitions.

6. RELATED WORK

In Lilius [1998], Yoneda and Schlingloff [1997], and Yoneda et al. [1993], the authors have proposed two partial order reduction techniques based on POSETs for 1-safe time Petri nets. However, as we will show, the state-space abstraction used in Lilius [1998] does not preserve deadlocks, whereas the approach proposed in Yoneda and Schlingloff [1997] and Yoneda et al. [1993], is not appropriate in the context of the CSCG.

6.1. Lilius's Approach

The partial order technique proposed in Lilius [1998] for 1-safe time Petri nets consists of exploring a persistent set from each abstract state of the state-space abstraction defined by the following:

—The initial abstract state is $\beta_0 = (M_0, \bigwedge_{t\in En(M_0)} \downarrow Is(t) \leq t - o \leq \uparrow Is(t))$, o being the starting time of the system.

Table II. Some Experimental Results

TPN	RSCG	CSCG	TPN	RSCG	CSCG
HC(1) var. 1			HC(2) var. 1		
NSC	25	66	NSC	192	1,501
NCSC	30	120	NCSC	337	4,780
CPU (s)	0	0	CPU (s)	0	0
HC(3) var. 1			HC(4) var. 1		
NSC	3,618	19,486	NSC	43,758	? >131,991
NCSC	6,426	83,440	NCSC	87,756	? > 681,385
CPU (s)	1	44	CPU (s)	150	? > 3,600
HC(1) var. 2			HC(2) var. 2		
NSC	23	44	NSC	289	889
NCSC	24	60	NCSC	403	2,021
CPU (s)	0	0	CPU (s)	0	0
HC(3) var. 2			HC(4) var. 2		
NSC	3,542	10,654	NSC	36,262	102,265
NCSC	6,788	35,554	NCSC	93,219	446,242
CPU (s)	0	8	CPU (s)	134	2,183
FMS(1) var. 1			FMS(2) var. 1		
NSC	120	138	NSC	4,140	4,470
NCSC	312	381	NCSC	15,910	20,379
CPU (s)	0	0	CPU (s)	1	1
FMS(3) var. 1			FMS(4) var. 1		
NSC	65,080	70,190	NSC	? > 147,745	? > 135,980
NCSC	306,036	414,432	NCSC	? > 644,885	? > 763,840
CPU (s)	722	1,048	CPU (s)	? > 3,600	? > 3,600
FMS(1) var. 2			FMS(2) var. 2		
NSC	201	203	NSC	7,332	7,824
NCSC	334	349	NCSC	18,038	21,254
CPU (s)	0	0	CPU (s)	2	2
FMS(3) var. 2			FMS(4) var. 2		
NSC	19,528	29,235	NSC	76387	117,316
NCSC	49,502	90,017	NCSC	211220	39 6,066
CPU (s)	21	75	CPU (s)	615	1,800
SP(1,4,4) var. 1			SP(2,4,4) var. 1		
NSC	1,005	1,804	NSC	3,756	6,686
NCSC	1,885	5,620	NCSC	7,906	24,496
CPU (s)	0	0	CPU (s)	0	2
SP(3,4,4) var. 1			SP(4,4,4) var. 1		
NSC	9,930	17,416	NSC	21,019	36,610
NCSC	22,227	70,260	NCSC	48,532	156,800
CPU (s)	3	22	CPU (s)	18	173
SP(1,4,4) var. 2			SP(2,4,4) var. 2		
NSC	1,569	12,534	NSC	6,614	81,353
NCSC	2,617	32,857	NCSC	11,888	245,110
CPU (s)	0	8	CPU (s)	1	711
SP(3,4,4) var. 2			SP(4,4,4) var. 2		
NSC	17,052	? >184,768	NSC	31,981	? > 190,670
NCSC	31,905	? >555,730	NCSC	62,178	? > 548,771
CPU (s)	18	? > 3,600	CPU (s)	57	? > 3,600
HC(4) var. 3			HC(5) var. 3		
NCSC before Deadlock	2,301	2,853	NCSC before Deadlock	13,520	17,512
CPU (s)	0	0	CPU (s)	2	2
HC(6) var. 3			HC(7) var. 3		
NCSC before Deadlock	41,190	53,068	NCSC before Deadlock	183,064	264,922
CPU (s)	12	14	CPU (s)	404	669

—The successor abstract states are computed by ignoring firing order constraints between unrelated transitions (partial order successors). Let $\beta = (M, F)$ be an abstract state and $t_f \in T$ a transition. Transition t_f is firable from β iff $t_f \in En(M)$ and $F \wedge \bigwedge_{t \in En(M)} t_f \leq t$ is consistent. If t_f is firable from β, its successor abstract state $\beta' = (M', F')$ is given by $M' = M - pre(t_f) + post(t_f)$ and F' is computed in two steps:

1) Set F' to $F \wedge \bigwedge_{t \in Nw(M,t_f)} low(^{\circ\circ}t, F) + \downarrow Is(t) \leq t - o \leq up(^{\circ\circ}t, F) + \uparrow Is(t)$.
1) Put F' in canonical form and then eliminate transitions in conflict with t_f in M and those of $^{\circ\circ}t_f$,

where $low(^{\circ\circ}t, F)$ and $up(^{\circ\circ}t, F)$ are the upper bounds on the earliest and latest possible occurrence times of transitions in $^{\circ\circ}t$ (i.e., the input transitions of t), respectively.
However, this state-space abstraction is not complete (i.e., there is at least a sequence feasible in the state-space abstraction but not feasible in the model) and does not preserve deadlocks.

As an example, consider a variant of the TPN in Figure 1(a), where the static firing intervals of t_1 and t_2 are replaced with $[1, 2]$ and $[1, 1]$, respectively. Note that this TPN is deadlock free. For the state-space abstraction of Lilius, the initial abstract state is $\beta_0 = (p_1 + p_2, 1 \leq t_1 - o \leq 2 \wedge 1 \leq t_2 - o \leq 1)$. According to the firing rule, t_1 and t_2 are firable and not in conflict. Both firing orders of t_1 and t_2 lead to the same abstract state $\beta_{12} = (p_3 + p_4, 1 \leq t_1 - o \leq 2 \wedge 1 \leq t_2 - o \leq 1 \wedge 3 \leq t_3 - o \leq 4 \wedge 2 \leq t_4 - o \leq 3)$. Transitions t_3 and t_4 are firable from β_{12}. Therefore, sequences $t_1t_2t_3$ and $t_2t_1t_3$ are feasible in the state-space abstraction and lead to a deadlock abstract state. The approach of Lilius will conclude wrongly that the model is not deadlock free. In fact, sequences $t_1t_2t_3$ and $t_2t_1t_3$ are not feasible in the model. Indeed, for the sequence t_2t_1, both transitions t_3 and t_4 are enabled by t_1. So, transition t_4 will reach the upper bound 1 of its firing interval before reaching the lower bound 2 of the firing interval of t_3, for the sequence t_1t_2, t_2 is immediately fired after t_1. As there is no delay between firings of transitions t_1 and t_2, it means that there is no delay between the enabling dates of t_3 and t_4 and then t_4 will reach the upper bound of its firing interval before reaching the lower bound of the firing interval of t_3. Therefore, after t_1t_2 or t_2t_1, transition t_4 will eventually fire first and disable t_3. Sequences $t_1t_2t_3$ and $t_2t_1t_3$ are then not feasible in the model.

6.2. Yoneda's Approach

The partial order approach of Yoneda et al. is based on the stubborn set method. Let $\alpha = (M, F)$ be an abstract state and $t \in Fr(\alpha)$. The subset of transitions $Ready(\alpha)$ to explore from α, called the ready set of α, is computed as follows [Yoneda and Schlingloff 1997]:

```
1: Initialize the ready set ready with some transition firable from $\alpha$.
2: Iterate $ready := ready \cup (dep(t, M) \cap Fr(\alpha))$ for some $t \in ready$ until a fixed point is
   reached.
3: If $\exists t \in ready$ and $t' \in dep(t, M)$ s.t. $t'$ can fire before all transitions in ready but
   $t' \notin ready$, then add some firable $t''$ s.t. $F = F \wedge t'' < t'$ to ready and go to step 2.
```

The sets $dep(t, M)$, $nec^*(t, M)$, and $nec(t, M)$ are defined as follows:

—$dep(t, M) = \bigcup_{t' \in (^\circ t)^\circ} nec^*(t', M) \cap En(M)$.
—$nec^*(t, M)$ is the smallest set s.t.

$$t \in nec^*(t, M) \ \wedge \ \forall t' \in T, (t' \in nec^*(t, M) \Rightarrow nec(t, M) \subseteq nec^*(t, M)).$$

—$nec(t, M) = \begin{cases} \{t' \in T \mid \exists p \in t'^\circ, pre(t)(p) > M(p)\} & \text{if t} \notin \text{En(M)} \\ \emptyset & \text{otherwise.} \end{cases}$

Intuitively, $dep(t, M)$ is the subset of transitions enabled in M that are structurally in conflict with t or may enable directly or indirectly a transition structurally in conflict with t (i.e., all transitions of $En(M)$ that are necessary for transitions structurally in conflict with t). The ready set ensures the inclusion of all transitions structurally in conflict with its enabled transitions and also all transitions that may enable directly or indirectly its disabled transitions, which, intuitively, corresponds to the condition $SC1$. Moreover, there is no transition outside the ready set that must be fired before all its transitions. It is then clear that there is no condition equivalent to part 4 of $SC2$ (a sufficient condition of $C2$). Let us show by means of an example that the ready set does not satisfy $C2$. Consider the model in Figure 1(a). The ready set of the initial abstract state may be either $\{t_1\}$ or $\{t_2\}$. Condition $C2$ is not satisfied for the ready set $\{t_2\}$, as $succ(\alpha_0, t_1t_2) \not\subseteq succ(succ_G(\alpha_0, t_2), t_1)$. Indeed, in $succ(\alpha_0, t_1t_2) = (p_3+p_4, -1 \leq t_3-t_4 \leq 1)$, both transitions t_3 and t_4 are firable, whereas in $succ(succ_G(\alpha_0, t_2), t_1) = (p_3+p_4, t_3-t_4 = 1)$, only t_4 is firable. It means that using the ready set $\{t_2\}$, the model will be declared as deadlock free, which is not correct. The reason for this failure is that t_4 may be enabled by t_1 or t_2 dependently of their firing order (i.e., the parent of t_4 is independent of the firing order of t_1 and t_2).

To deal with this issue, in Yoneda and Schlingloff [1997], the computation of successor abstract states considers all possible *truth parents* of the newly enabled transitions. More precisely, let β be an abstract state and t_f a firable transition from β. The firing of t_f may enable new transitions. If the transitions fired so far, including t_f, are partially ordered, the newly enabled transitions may have different parents. In such a case, a successor abstract state of β by t_f is computed for each combination of truth parents of the newly enabled transitions. Each successor is computed by ignoring the blocking time constraints of transitions outside the ready set $Ready(\beta)$ (partial order successor).

For the previous example, firing t_2 from the initial abstract state will not enable any new transition. Its firing, without imposing any firing order between t_2 and t_1, implicitly includes that the case of t_1 is fired before t_2. Afterward, the firing of t_1 will enable two new transitions t_3 and t_4, but the parent of t_4 may be t_2 as, in fact, the firing order between t_1 and t_2 is not fixed. Two successors by t_1 are computed, one for each possible parent of t_4.

To determine the truth parents of enabled transitions, each abstract state keeps, in addition to the time constraints of all enabled transitions, those of their parents [Yoneda and Schlingloff 1997]. Let us explain the principle of this approach using the TPN in Figure 1(a) and the interval-based timing information (instead of the clock-based timing information used in Yoneda and Schlingloff [1997]). The initial abstract state is $\beta_0 = (p_1 + p_2, -2 \leq t_1 - t_2 \leq 0)$. The ready set of β_0 may be either $\{t_1\}$ or $\{t_2\}$. If $Ready(\beta_0) = \{t_2\}$, the firing of t_2 leads to the abstract state $\beta_1 = (p_1+p_4, -2 \leq t_1-t_2 \leq 0)$. The firing of t_1 from β_1 will enable two new transitions t_3 and t_4, where t_4 has two possible parents t_1 and t_2. So, β_2 has two successors by t_1: $\beta_2 = (p_3 + p_4, -2 \leq t_1 - t_2 \leq 0 \wedge t_1 \leq t_2 \wedge t_3 - t_1 = 2 \wedge t_4 - t_2 = 1)$ and $\beta_3 = (p_3 + p_4, -2 \leq t_1 - t_2 \leq 0 \wedge t_2 \leq t_1 \wedge t_3 - t_1 = 2 \wedge t_4 - t_1 = 1)$. From β_2, both transitions t_3 and t_4 are firable, while from β_3, only t_4 is firable.

However, determining all possible truth parents of the newly enabled transition is costly for large POSETs and even more complicated for nonsafe TPNs. Moreover, to achieve more contractions (coarser abstractions), in the CSCG state classes, the simple time constraints of the enabled transitions and time constraints of their parents are not represented. Extending the CSCG state classes with the time constraints of the parents may offset its coarseness.

7. CONCLUSION

We have proposed a revisited stubborn set method, which is more appropriate in the context of the TPN model. We have shown that the resulting reduced state space preserves deadlocks. From the practical point of view, we have established a sufficient condition $SC2$, which allows one to derive simple algorithms for computing stubborn sets and reduced state class graphs (RSCGs). We have proved that the resulting RSCGs preserve the k-boundedness of each place of the TPN. Moreover, according to the proof of Theorem 3.6, the RSCGs preserve nonequivalent firing sequences of the TPN. So, the extension of the proposed verification approach to LTL_{-X} properties is straightforward using the well-established methods based on the stuttering equivalent sequences [Godefroid 1996; Peled and Wilke 1997; Yoneda and Schlingloff 1997; Valmari and Hansen 2010].

As a future work, we will investigate the timed properties preserved by the RSCGs.

REFERENCES

W. Belluomini and C. J. Myers. 2000. Timed state space exploration using POSETs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits* 19, 5 (2000), 501–520.

J. Bengtsson. 2002. *Clocks, DBMs and States in Timed Systems*. Ph.D. thesis, Dept. of Information Technology, Uppsala University.

J. Bengtsson, B. Jonsson, J. Lilius, and W. Yi. 1998. Partial order reductions for timed systems. In *9th International Conference on Concurrency Theory (CONCUR'98)*. Lecture Notes in Computer Science, Vol. 1466. Springer, Berlin, 485–500.

B. Bérard, F. Cassez, S. Haddad, D. Lime, and O. H. Roux. 2013. The expressive power of time Petri nets. *Theoretical Computer Science* 474 (2013), 1–20.

B. Berthomieu and M. Diaz. 1991. Modeling and verification of time dependent systems using time petri nets. *IEEE Transactions on Software Engineering* 17, 3 (1991), 259–273.

B. Berthomieu and F. Vernadat. 2003. State class constructions for branching analysis of time Petri nets. In *9th International Conference of Tools and Algorithms for the Construction and Analysis of Systems.* Lecture Notes in Computer Science, Vol. 2619. Springer, 442–457.

H. Boucheneb and K. Barkaoui. 2013. Reducing interleaving semantics redundancy in reachability analysis of time Petri nets. *ACM Transactions on Embedded Computing Systems* 12, 1 (2013), 259–273.

H. Boucheneb, G. Gardey, and O. H. Roux. 2009. TCTL model checking of time Petri nets. *Logic and Computation* 19, 6 (2009), 1509–1540.

H. Boucheneb and R. Hadjidj. 2006. CTL* model checking for time Petri nets. *Theoretical Computer Science* 353, 1–3 (2006), 208227.

H. Boucheneb, D. Lime, and O. H. Roux. 2013. On multi-enabledness in time Petri nets. In *34th International Conference on Application and Theory of Petri Nets and Other Models of Concurrency (ICATPN'13).* Lecture Notes in Computer Science, Vol. 7927. Springer, 130–149.

H. Boucheneb and H. Rakkay. 2008. A more efficient time Petri net state space abstraction useful to model checking timed linear properties. *Fundamenta Informaticae* 88, 4 (2008), 469–495.

M. Boyer and M. Diaz. 2001. Multiple-enabledness of transitions in time petri nets. In *9th IEEE International Workshop on Petri Nets and Performance Models*. IEEE Computer Society, 219–228.

T. Chatain and C. Jard. 2006. Complete finite prefixes of symbolic unfoldings of safe time Petri nets. In *27th International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency (ICATPN'06).* Lecture Notes in Computer Science, Vol. 4024. Springer, 125–145.

T. Chatain and C. Jard. 2013. Back in time petri net. In *11th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'13).* Lecture Notes in Computer Science, Vol. 8053. Springer, 91–105.

D. Delfieu, M. Sogbohossou, L. M. Traonouez, and S. Revol. 2007. Parameterized study of a time Petri net. In *Cybernetics and Information Technologies, Systems and Applications: CITSA*. 89–90.

G. M. Delgadillo and S. P. Llano. 2006. Scheduling application using Petri nets: A case study: Intergráficas s.a. In *9th International Conference on Production Research*. 1–6.

P. Godefroid. 1996. *An Approach to the State-Explosion Problem*. Lecture Notes in Computer Science, Vol. 1032. Springer-Verlag, New York.

J. Hakansson and P. Pettersson. 2007. Partial order reduction for verification of real-time components. In *5th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS'07).* Lecture Notes in Computer Science. Springer, 211–226.

J. Lilius. 1998. Efficient state space search for time Petri nets. In *MFCSÕ98 Workshop on Concurrency Algorithms and Tools*. Electronic Notes of Theoretical Science, Vol. 8. Elsevier Science B.V, Brno, Czech Republic, 113–133.

D. Lugiez, P. Niebert, and S. Zennou. 2005. A partial order semantics approach to the clock explosion problem of timed automata. *Theoretical Computer Science* 345, 1 (2005), 27–59.

E. G. Mercer, C. J. Myers, and T. Yoneda. 2001. Improved POSET timing analysis in timed Petri nets. In *International Workshop on Synthesis and System Integration of Mixed Technologies*. 1–6.

M. Minea. 1999. Partial order reduction for model checking of timed automata. In *10th International Conference on Concurrency Theory (CONCUR'99)*. Lecture Notes in Computer Science, Vol. 1664. Springer, Berlin, 431–446.

D. Peled. 1993. All from one, one for all: On model checking using representatives. In *Computer Aided Verification*, Costas Courcoubetis (Ed.). Lecture Notes in Computer Science, Vol. 697. Springer, Berlin, 409–423.

D. Peled and T. Wilke. 1997. Stutter invariant temporal properties are expressible without the next-time operator. *Information Processing Letters* 63, 5 (1997), 243–246.

D. Pradubsuwun, T. Yoneda, and C. Myers. 2005. Partial order reduction for detecting safety and timing failures of timed circuits. *IEICE - Transactions on Information and Systems*. E99-D(7) (2005), 1646–1661.

R. Ben Salah, M. Bozga, and O. Maler. 2006. On interleaving in timed automata. In *17th International Conference on Concurrency Theory (CONCUR'06)*. Lecture Notes in Computer Science, Vol. 4137. Springer, 465 –476.

A. Semenov and A. Yakovlev. 1996. Verification of asynchronous circuits using time petri net unfolding. In *33rd Annual Conference on Design Automation (DAC'96)*. 59–62.

M. Sogbohossou and D. Delfeu. 2008. Temporal reduction in time petri net. In *3rd International Workshop on Design and Test (IDT'08)*. IEEE, 260–265.

A. Valmari. 1992. A stubborn attack on state explosion. *Formal Methods in System Design* 1, 4 (1992), 297–322.

A. Valmari. 1998. *The State Explosion Problem*. Lecture Notes in Computer Science, Vol. 1491. Springer, Berlin, 429–528 pages.

A. Valmari and H. Hansen. 2010. Can stubborn set be optimal. In *31st International Conference on Application and Theory of Petri Nets and Concurrency (Petri Nets'10)*. Lecture Notes in Computer Science, Vol. 6128. Springer, Berlin, 43–62.

T. Yoneda and H. Ryuba. 1998. CTL model checking of time petri nets using geometric regions. *IEICE - Transactions on Information and Systems*. E99-D(3) (1998), 297–306.

T. Yoneda and B. H. Schlingloff. 1997. Efficient verification of parallel real-time systems. *Formal Methods in System Design* 11, 2 (1997), 187–215.

T. Yoneda, A. Shibayama, B. H. Schlingloff, and E. M. Clarke. 1993. Efficient verification of parallel real-time systems. In *Computer Aided Verification*, Costas Courcoubetis (Ed.). Lecture Notes in Computer Science, Vol. 697. Springer, Berlin, 321–332.

Received November 2013; revised March 2014; accepted September 2014