Exploiting local persistency for reduced state-space generation

Kamel Barkaoui, Hanifa Boucheneb, Zhiwu Li

▶ **To cite this version:**

Kamel Barkaoui, Hanifa Boucheneb, Zhiwu Li. Exploiting local persistency for reduced state-space generation. Innovations in Systems and Software Engineering, 2020, 10.1007/s11334-020-00358-3 . hal-02590041

HAL Id: hal-02590041
https://hal.science/hal-02590041

Submitted on 7 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Noname manuscript No.
(will be inserted by the editor)

Exploiting local persistency for reduced state space generation

K. Barkaoui · H. Boucheneb · Z. Li

Received: date / Accepted: date

Abstract This paper deals with the partial order techniques of Petri nets, based on persistent sets and step graphs methods. To take advantage of the strengths of each method, it proposes the persistent step sets as a parametric combination of the both methods. The persistent step sets method allows to fix, for each marking, the set of transitions S to be covered by the selected steps and then to control their maximal length and number. Depending on the parameter S, it computes covering-steps, persistent sets, persistent-step sets or other kinds of combination of both methods. Moreover, this persistent step selective search preserves, at least, deadlocks of Petri nets.

Furthermore, this paper provides two practical computation procedures of the persistent step sets based on the strong-persistent sets and the weak-persistent sets, respectively. Finally, to achieve further reductions, it shows how to weaken the sufficient conditions used in the literature to build persistent sets.

Keywords Petri nets · reachability analysis · state explosion problem · persistent sets · partial order techniques · step graphs

Kamel Barkaoui
Laboratoire CEDRIC, Conservatoire National des Arts et Métiers,
192 rue Saint Martin, Paris Cedex 03, France
E-mail: kamel.barkaoui@cnam.fr

Hanifa Boucheneb
Laboratoire VeriForm, Department of Computer Engineering and Software Engineering,
École Polytechnique de Montréal,
P.O. Box 6079, Station Centre-ville, Montréal, Québec, Canada, H3C 3A7.
E-mail: hanifa.boucheneb@polymtl.ca

Zhiwu Li
Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau,
School of Electro-Mechanical Engineering, Xidian University, Xi'an 710071, China,
E-mail: zhwli@xidian.edu.cn

1 Introduction

The state explosion problem is the main obstacle for the verification of concurrent systems, as they are generally based on an interleaving semantics, where all possible firing orders of concurrent actions are exhaustively explored. Different techniques for fighting this problem have been proposed such as structural analysis, symmetries and partial orders.

The structural analysis attempts to find a relationship between the behaviour of the net and its structure, where the initial marking is considered as a parameter. The net structure can be studied through its associated incidence matrix and the corresponding net state equation leading mainly to the concept of place invariants [6]. It can also be studied through topological properties of the interplay between conflict and synchronisation of remarkable substructures of the net such siphons and traps. In both cases, the aim of these studies is to highlight necessary and/or sufficient structural conditions to check general behavioural properties such as liveness [4,9], for large subclasses of place/transition nets [1,2].

The second well-accepted technique to tackle combinatorial explosion in model-checking consists in exploiting symmetries over states and the transition relation [8] to build a quotient graph of equivalence classes of states, that may be exponentially smaller than the full state graph, while preserving many behavioural properties of interest.

The partial order techniques have been proven to be the most successful in practice. We distinguish two classes of partial order techniques: partial order reduction techniques [7,10,11,13–15] and step graph techniques [18]. Partial order reduction techniques, such as the ample sets [10,11], the stubborn sets [13–15] and the persistent sets [7], deal with the state explosion problem by avoiding as much as possible to explore firing sequences that are equivalent w.r.t. the properties of interest (deadlock freeness, deadlocks, reachability, liveness, or linear properties)[1]. The step graph methods explore all the transitions of the state space but some of them are fired together in atomic steps. They aim to reduce the depth of the marking graph while the purpose of the partial order reduction techniques is to reduce its breadth.

The common characteristics of all these methods is to reduce the state space to be explored, by selecting the actions or sets of actions (steps) to be executed from each state. The selection procedure of actions or steps relies on the notion of independent actions. Two actions are said to be independent, if whenever they are enabled, they can be fired in both orders and the firing of one of them does not inhibit the occurrence of the other. Moreover, their firing in both orders leads to the same state. Both of these conditions constitute the well known diamond property.

Each of the partial order reduction methods above provides sufficient conditions that ensure, at least, preservation of deadlocks markings (i.e., markings with no enabled transitions). Thus, the set ST of the selected transitions or steps is only empty for the deadlock markings. The other sufficient conditions are generally based on the structure of the model, the property to be verified and the current marking. Their aim

[1] Two firing sequences are equivalent w.r.t. some property, if they cannot be distinguished by the property.

is to ensure independency between transitions of ST and the others. Indeed, for the ample sets method [10, 11], there is no transition outside ST that is firable before all transitions of ST and, at the same time, is dependent of at least a transition of ST. For the stubborn sets method [13–15], ST contains at least an enabled transition that cannot be disabled by the transitions outside ST and each of its transitions t is independent of all transitions outside ST that are firable before t. The persistent sets method [7] is a particular case of the stubborn sets method, where all transitions of ST are enabled.

For the covering-steps graph method [18], the set of steps to be fired from each marking must cover the set of enabled transitions. To achieve more reductions, in [12], the authors have combined this technique with the persistent sets method. This combination consists to firstly compute a persistent set for the current marking and then look for firing steps within this persistent set. For these approaches, the transitions within the same step are neither in weak-conflict[2] nor in structural conflict with the partially enabled transitions.

This paper deals with the persistent sets and the step graph methods. To take advantage of the strengths of each method, it investigates their combination and proposes persistent-step sets method. Persistent step sets method is a parametric combination of persistent sets with step graphs that allows to fix, for each marking, the set of transitions S to be covered by the selected steps and then to control their maximal length and number. Depending on the parameter S, it computes covering-steps [18], persistent sets [7], persistent step sets [12] or other combinations of both methods. Furthermore, this paper establishes weaker sufficient conditions to build persistent sets.

The rest of the paper is organised as follows. Section 2 fixes some classical definitions and notations used throughout the paper. Section 3 presents the strong-persistent sets [12, 7], the weak-persistent sets (a weaker version of the strong-persistent sets) and the step graph methods, while pointing out their weaknesses. Section 4 provides a formal definition of persistent step sets and proves that they yield graphs preserving deadlocks of Petri nets. Section 5 establishes two parametric computation procedures of persistent step sets (Algorithm 1 and Algorithm 2) that are based on strong-persistent sets and weak-persistent sets, respectively. Section 6 presents some experimental results of Algorithm 1 and its comparison with the tool TINA[3]. Section 7 shows how to weaken the sufficient conditions used to build strong persistent sets, while preserving deadlocks of Petri nets. It also reports some results of experimental comparison of the algorithms proposed in this paper. Conclusions are presented in Section 8.

2 Preliminaries

Let P be a nonempty set. A multi-set over P is a function $M : P \longrightarrow \mathbb{N}$, $\mathbb{N}$ being the set of natural numbers, defined also by the linear combination over P: $\sum_{p \in P} M(p) \times p$.

[2] The (structural) weak-conflict relation is the transitive closure of the (structural) conflict relation.

[3] http://projects.laas.fr/tina//home.php.

We denote by P_{MS} and 0 the set of all multi-sets over P and the empty multi-set, respectively. Operations on multi-sets are defined as usual. Notice that any subset $X \subseteq P$ can be defined as a multi-set over P: $X = \sum_{p \in X} 1 \times p$.

For $X \in 2^T$ and $Y \subseteq T$, we define the operator $\otimes$ by:
$X \otimes Y = \{x \cup \{y\} \mid x \in X \wedge y \in Y\}$ and $\otimes Y = \emptyset \otimes Y = \{\{y\} \mid y \in Y\}$.

An ordinary Petri net (PN in short) is a tuple $PN = (P, T, pre, post)$ where:

- P and T are finite and nonempty sets of places and transitions with $P \cap T = \emptyset$,
- pre and $post$ are the backward and forward incidence functions over the set of transitions T ($pre, post : T \longrightarrow 2^P$).

For $t \in T$, $pre(t)$ and $post(t)$ are the sets of input and output places of t, denoted also by ${}^\bullet t$ and $t^\bullet$, respectively. Similarly, the sets of input and output transitions of a place $p \in P$ are defined by ${}^\bullet p = \{t \in T \mid p \in t^\bullet\}$ and $p^\bullet = \{t \in T \mid p \in {}^\bullet t\}$, respectively.

Two transitions t and t' are in structural conflict, denoted by $t \perp t'$ iff $pre(t) \cap pre(t') \neq \emptyset$. We denote by $CFS(t) = \{t' \in T | t \perp t'\} = ({}^\bullet t)^\bullet$ the set of transitions in structural conflict with t. They are in structural weak-conflict iff $t \perp^* t'$, where $\perp^*$ is the transitive closure of $\perp$. We denote by $CFS^*(t) = \{t' \in T | t \perp^* t'\}$ the set of transitions in structural weak-conflict with t. Notice that $t \in CFS(t)$ and $CFS(t) \subseteq CFS^*(t)$.

A marking of an ordinary Petri net indicates the distribution of tokens over its places. It is defined as a multi-set over places. A marked PN is a pair $\mathcal{N} = (PN, M_0)$, where PN is an ordinary Petri net and $M_0 \in P_{MS}$ is its initial marking. Starting from its initial marking, PN evolves by firing enabled transitions. For the following, we fix a marked PN $\mathcal{N}$, a marking $M \in P_{MS}$ and a transition $t \in T$ of $\mathcal{N}$.

The transition t is enabled at M, denoted by $M[t\rangle$ iff all the required tokens for firing t are present in M, i.e., $M \geq pre(t)$. The transition t is partially enabled in M iff t is not enabled in M and, at least, one of its input places is marked. In case t is enabled at M, its firing leads to the marking $M' = M - pre(t) + post(t)$. The notation $M[t\rangle M'$ means that t is enabled at M and M' is the marking reached from M by t. We denote by $En(M)$ the set of transitions enabled at M, i.e., $En(M) = \{t \in T \mid M \geq pre(t)\}$. The marking M is a deadlock iff $En(M) = \emptyset$.

For any subset $X \subseteq T$ of transitions. We denote by $\Omega(X)$ the sets of all finite and infinite sequences of transitions over X including the empty sequence ϵ. A sequence $\omega \in \Omega(X)$ is elementary if the occurrence of each transition in the sequence does not exceed 1. For any non-empty sequence of transitions $\omega = t_1 t_2 ... t_n \in \Omega(T) - \{\epsilon\}$, the usual notation $M[t_1 t_2 ... t_n\rangle$ means that there exist markings $M_1, ..., M_n$ such that $M_1 = M$ and $M_i[t_i\rangle M_{i+1}$, for $i \in [1, n-1]$ and $M_n[t_n\rangle$. The sequence ω is said to be a firing sequence of M. The notation $M[t_1 t_2 ... t_n\rangle M'$ gives, in addition, the marking reached by the sequence. The marking M' is said to be reachable from M by ω. By convention, we have $M[\epsilon\rangle M$. We denote by $\overrightarrow{M}$ the set of markings reachable from M, i.e., $\overrightarrow{M} = \{M' \in P_{MS} | \exists \omega \in \Omega(T), M[\omega\rangle M'\}$.

A firing sequence ω of M is maximal if either it is infinite or it is finite and ends with a deadlock marking. Two sequences of transitions ω and ω' are equivalent,

denoted by $\omega \equiv \omega'$ iff they are identical or each one can be obtained from the other by a series of permutations of transitions. If $\omega \equiv \omega'$ then $\forall M', M'' \in P_{MS}, (M[\omega\rangle M' \wedge M[\omega'\rangle M'') \Rightarrow M' = M''$. We denote by $[\omega]$ the set of transitions in ω. The firing sequences of $\mathcal{N}$ are the firing sequences of its initial marking.

The different possible evolutions of $\mathcal{N}$ are represented in a marking graph MG defined by the structure $MG = (\overrightarrow{M}_0, [\rangle, M_0)$. Let n be a natural number. The marked PN $\mathcal{N}$ is n-bounded iff for every reachable marking of M_0, the number of tokens in each place does not exceed n. It is safe iff it is 1-bounded. It is bounded iff it is k-bounded for some natural number k.

A firing step τ of $\mathcal{N}$ is a non-empty subset of transitions ($\tau \subseteq T$) fired simultaneously and atomically from a marking of $\mathcal{N}$. From an interleaving semantic point of view, it represents an abstraction of all firing orders of its transitions. For instance, $\tau = \{t_1, t_2, t_3\}$ represents the following six sequences: $t_1t_2t_3$, $t_1t_3t_2$, $t_2t_1t_3$, $t_2t_3t_1$, $t_3t_1t_2$ and $t_3t_2t_1$. The intermediate markings are abstracted to keep only the markings before and after the firing step. Let $M \in P_{MS}$ be a marking and $\tau \subseteq T$ a firing step of $\mathcal{N}$. The firing step τ is enabled in M, denoted by $M[\tau\rangle$ iff $M \geq \sum_{t\in\tau} pre(t)$, which means that there are enough tokens to fire concurrently all the transitions within the step. If τ is enabled in M, its firing leads to the marking $M' = M + \sum_{t\in\tau}(post(t) - pre(t))$. The notation $M[\tau\rangle M'$ means that τ is enabled at M and M' is the marking reached from M by τ. We denote by $EnS(M)$ the set of all enabled steps in M, i.e., $EnS(M) = \{\tau \subseteq T \mid M \geq \sum_{t\in\tau} pre(t)\}$. The firing step τ is maximal in M iff it is maximal for the inclusion in $EnS(M)$, i.e., $M \geq \sum_{t\in\tau} pre(t)$ and $\forall t' \in En(M) - \tau, M \ngeq pre(t') + \sum_{t\in\tau} pre(t)$.

A step graph of $\mathcal{N}$ is a structure $SG = (MM, R, M_0)$, where $MM \subseteq \overrightarrow{M}_0$ is a subset of reachable markings, M_0 is the initial marking and $R \subseteq MM \times 2^T \times MM$ is the relation defined by $(M, \tau, M') \in R \Rightarrow M[\tau\rangle M'$.

For the rest of paper, we fix an ordinary Petri net $\mathcal{N} = (P, T, pre, post, M_0)$.

3 Persistent sets and step graphs

3.1 Strong-persistent sets

Let M be a marking. Informally, a persistent set of M is a subset μ of enabled transitions such that no transition of μ can be disabled, as long as no transition of μ is fired [7,12]. A persistent graph is obtained by recursively firing from each marking a persistent set. Persistent graphs preserve deadlocks of Petri nets [12].

However, this strong definition of persistent sets can be weakened while preserving deadlocks of Petri nets. The idea comes from the stubborn sets [15]. But unlike the stubborn sets, all the transitions inside a persistent set are enabled. To distinguish between the two definitions of persistent sets, the persistent sets of [7,12] are referred as strong-persistent sets. The others are referred as weak-persistent sets.

Definition 1 Let M be a marking and $\mu \subseteq En(M)$ a subset of enabled transitions. Formally, the subset μ is a strong-persistent set of M, if all the following conditions are satisfied:

- $D0 : En(M) \neq \emptyset \Leftrightarrow \mu \neq \emptyset$.
- $D1 : \forall t \in \mu, \forall \omega \in \Omega(T-\mu), M[\omega\rangle \Rightarrow M[\omega t\rangle$.
- $D2 : \forall t \in \mu, \forall \omega \in \Omega(T-\mu), M[\omega t\rangle \Rightarrow M[t\omega\rangle$.

The subset μ is a weak-persistent set of M, if it satisfies all the following conditions:

- $D0$: $En(M) \neq \emptyset \Leftrightarrow \mu \neq \emptyset$.
- $D1w$: $\exists t \in \mu, \forall \omega \in \Omega(T-\mu), M[\omega\rangle \Rightarrow M[\omega t\rangle$.
- $D2$: $\forall t \in \mu, \forall \omega \in \Omega(T-\mu), M[\omega t\rangle \Rightarrow M[t\omega\rangle$.

Intuitively, Condition $D0$ ensures that a strong/weak-persistent set of M is empty only if M is a deadlock. Conditions $D1w$ means that there is at least a transition inside μ that is maintained enabled as long as no transition of μ is fired. Conditions $D1$ states that all transitions of μ are maintained enabled as long as no transition of μ is fired. Condition $D2$ means that if some sequence ω with no transition from μ is firable before any transition t of μ, then it is also firable after t.

The transitions of μ that satisfy $D1w$ are called the key-transitions of μ [15]. Note that the strong-persistent sets contains only key-transitions. A strong/weak-persistent selective search from M_0[4] preserves all deadlock markings of Petri nets [16] (written, abusively, the strong/weak-persistent sets preserve deadlocks).

In the following, we investigate the combination of the strong/weak-persistent sets with the step graphs.

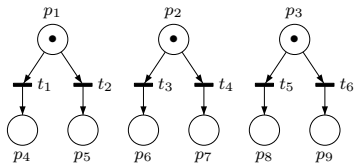

Fig. 1 Model PN1

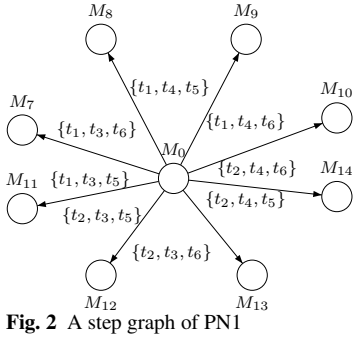

Fig. 2 A step graph of PN1

3.2 Step graphs

The aim of the step graph methods is to represent by a single path a largest possible set of equivalent maximal firing sequences of the model, by choosing appropriately, from each marking, the transitions to be fired together in steps. All transitions of the equivalent sequences are represented in the path but the concurrent ones are grouped together in steps. Step graphs allow to reduce the path depths of the marking graph.

[4] A strong/weak-persistent selective search of the Petri net is a procedure that recursively computes a strong/weak-persistent set and the successor markings reachable by the transitions of this set, for the initial marking and each new computed marking.

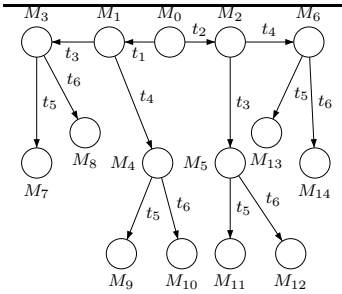

Fig. 3 A persistent set graph of PN1

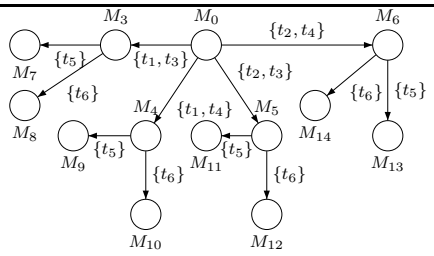

Fig. 4 A persistent step graph of PN1

However, in case there are several sets of transitions that are independent from each other, the number of steps and their lengths may be very large. For example, consider the model PN1 at Fig.1. Its marking graph consists of 27 nodes and 54 arcs. Its initial marking M_0 has 3 strong-persistent sets that are independent from each other: $\{t_1, t_2\}$, $\{t_3, t_4\}$ and $\{t_5, t_6\}$. A strong-persistent graph of PN1 is shown in Fig.3. Note that there are different strong-persistent graphs but they have all the same size ($2^4 - 1 = 15$ nodes and $15 - 1 = 14$ arcs). Using the 3 independent strong-persistent sets of M_0, steps can be built by picking a transition from each strong-persistent set. Its step graph is shown in Fig.2 and consists of $9 = 2^3+1$ nodes, $8 = 2^3$ arcs and 2^3 maximal steps. It is a covering-steps graph, as the set of steps selected from each marking covers all its enabled transitions. But, the number of successors of the initial marking exceeds the number of enabled transitions and is exponential with the number of independent strong-persistent sets. Even if the covering-steps graph is smaller than the persistent graph, the number of maximal steps and their lengths may be very large, which limits the usefulness of the step graph method.

To take advantage of the strengths of each method, the persistent sets and step graphs are combined in [12]. The idea is to compute a strong-persistent set and then determine the transitions within this set to be fired together in steps. As an example, for the strong-persistent set $\{t_1, t_2, t_3, t_4\}$ of the initial marking M_0, we can build 4 steps: $\{t_1, t_3\}$, $\{t_1, t_4\}$, $\{t_2, t_3\}$ and $\{t_2, t_4\}$. The resulting reduced graph is reported in Fig.4 and consists of 13 nodes and 12 arcs. This combination allows to control the number and the length of the steps to be considered from each marking, while yielding a graph that is larger than the step graph but smaller than the persistent graph.

4 Persistent step sets

We first define the notion of persistent step sets. Then, we show that the resulting graphs preserve deadlock markings of Petri nets.

Definition 2 Let M be a marking and $\mu_i \subseteq T$, for some $n > 0$ and $i \in [1, n]$, n subsets of enabled transitions pairwise disjunct, i.e., $\mu_i \cap \mu_j = \emptyset$ for $i \neq j \in [1, n]$. Let $\mu = \bigcup_{i \in [1,n]} \mu_i$ and $SS = (\otimes \mu_1) \otimes ... \otimes \mu_n$.
The set SS is a strong-persistent step set if it satisfies all the following conditions:

- $DS0$: $En(M) \neq \emptyset \Leftrightarrow \mu \neq \emptyset$.
- $DS1$: $\forall i \in [1,n], \forall t \in \mu_i, \forall \omega \in \Omega(T-\mu_i), M[\omega\rangle \Rightarrow M[\omega t\rangle$.
- $DS2$: $\forall i \in [1,n], \forall t \in \mu_i, \forall \omega \in \Omega(T-\mu_i), M[\omega t\rangle \Rightarrow M[t\omega\rangle$.

Similarly, the set SS is a weak-persistent step set if it satisfies all the following conditions:

- $DS0$: $En(M) \neq \emptyset \Leftrightarrow \mu \neq \emptyset$.
- $DSw1$: $\forall i \in [1,n], \exists t \in \mu_i, \forall \omega \in \Omega(T-\mu_i), M[\omega\rangle \Rightarrow M[\omega t\rangle$.
- $DS2$: $\forall i \in [1,n], \forall t \in \mu_i, \forall \omega \in \Omega(T-\mu_i), M[\omega t\rangle \Rightarrow M[t\omega\rangle$.

In other words, SS is a sort of cartesian product of pairwise disjunct strong/weak-persistent sets. Note that if $SS = \otimes\{t_1, ..., t_n\} = \{\{t_1\}, ..., \{t_n\}\}$ is a strong/weak-persistent step set then $\mu = \{t_1, ..., t_n\}$ is a strong/weak-persistent set.

Example 1 Consider the model PN1 at Fig.1.

- The set $\mu = \{t_1, t_2, t_3\}$ is a weak-persistent set but not a strong-persistent set of the initial marking $M_0 = p_1 + p_2 + p_3$, as it satisfies $D0$, $D1w$ and $D2$ but it does not satisfy $D1$. There are two key-transitions in μ: t_1 and t_2.
- The step set $SS = (\otimes\{t_1, t_2\}) \otimes \{t_3\} = \{\{t_1, t_3\}, \{t_2, t_3\}\}$ is not a persistent step set of M_0, as it does not satisfy Condition $DS1$ for $\{t_3\}$. Indeed, $t_4 \in \Omega(T - \{t_3\})$, we have $M_0[t_4 t_1\rangle$ and $\neg M_0[t_4 t_1 t_3\}\rangle$.

Theorem 1 *Strong/weak persistent step sets preserve deadlocks of the Petri net.*

Proof As a strong-persistent set is a weak-persistent set, we provide the proof for weak-persistent sets. It is obvious that all deadlocks reachable by the weak-persistent step selective search from M_0 are also reachable in the Petri net. Let M be a marking reached in a weak-persistent step selective search from M_0[5] and D a deadlock marking reachable from M in the Petri net. Let us show that D is also reached by the weak-persistent step selective search from M_0. The marking M is reachable in the Petri net. Let ω be a firing sequence leading to the marking D from M in the Petri net. The proof is by induction on the length of ω.

a) If $\omega = \epsilon$ then $M = D$.
b) If $\omega = t$ and $\{t\} \in SS$ then D is reached by the persistent step selective search.
c) If $\omega = t$ and $\{t\} \notin SS$ then, by $DSw1$, D is not a deadlock marking as there is, at least, a transition from μ that is firable after t, which is in contradiction with the fact that D is a deadlock.

Suppose that Theorem 1 holds for any marking M' (reachable in the weak-persistent step set selective search) and D is reachable from M' by a sequence ω' such that $|\omega'| < |\omega|$.

- If there is no step of SS (scattered or not) in ω (i.e., ω is free from all transitions of some μ_i), then, by $DSw1$, there is, at least, a missed transition of a step that is firable after ω. It means that D has, at least, a successor, which is in contradiction with the fact that D is a deadlock.
- If there is, at least a step τ of SS, scattered or not, in ω, then, by $DS2$ the transitions of this step can be shifted to the front to constitute a step firable from M, as the sets μ_i for $i \in [1,n]$ are pairwise disjunct. Firing this step from M leads to some marking M' that is reachable by the weak-persistent step selective search. Moreover, D is reachable from M', in the Petri net, by some sequence ω' s.t. $|\omega'| < |\omega|$. Therefore, D is reachable by the weak-persistent step selective search from M_0.

□

[5] A persistent step selective search from M_0 is a procedure that recursively computes a persistent step set and the successor markings reachable by these steps, for the initial marking and each new computed marking.

5 Parametric combination of persistent sets with step graphs

We propose, in the following, two parametric selection algorithms of persistent step sets, based on strong-persistent sets and weak-persistent sets, respectively.

For a given marking M and a subset S of transitions enabled in M ($S \subseteq En(M)$), the idea is to compute a persistent step set that covers, at least, the transitions of S. Unlike, the approach proposed in [12], the set S is not necessarily a strong-persistent set. As we will show, according to the parameter S, the provided persistent step set is either a strong/weak-persistent set, a covering-step set or a set of steps that covers partially the enabled transitions in M. We suppose available two procedures SPS and WPS that compute a strong-persistent set and a weak-persistent set, respectively, tor a given marking M and a transition t enabled in M.

5.1 Computing strong-persistent step sets

A computation procedure of strong-persistent step sets is provided in Algorithm 1. For a given marking M and a set of enabled transitions $S \subseteq En(M)$, Algorithm 1 returns a set of steps firable from M. The parameter S allows to specify the set of enabled transitions that must be, at least, covered by the set of steps.

The computed set of steps is a sort of product of some disjoint strong-persistent sets. The first term of the product is $R = SPS(t, M)$, where t is chosen randomly in S' (a copy of S). Afterwards, the transitions of R are deleted from S' to ensure that the next terms are disjoints from those computed so far. If the resulting set S' is not empty, then the same process is repeated to compute the next term of the product, and so on. Theorem 2 establishes that the returned set of steps is a strong-persistent step set.

Algorithm 1 Strong-persistent step set of a marking M covering the transitions of S

1: Input : A marking M and a subset S of enabled transitions such that $S \neq \emptyset$;
2: Output : A strong-persistent step set SS of M w.r.t. S;
3: $SS := \emptyset$; $S' := S$;
4: **while** ($S' \neq \emptyset$) **do**
5: Choose $t \in S'$;
6: $R := SPS(t, M)$;
7: $S' := S' - R$;
8: $SS := SS \otimes R$;
9: **end while**
10: **return** SS;
11: [For $X \in 2^T$ and $Y \subseteq T$, $X \otimes Y = \{x \cup \{y\} \mid x \in X \wedge y \in Y\}$ and $\emptyset \otimes Y = \{\{y\} \mid y \in Y\}$]

Example 2 Consider the initial marking M_0 of the model PN1 at Fig.1.

- For $S = \{t_1, t_2, t_3\}$, Algorithm 1 computes SS as follows. It starts by setting SS and S' to $\emptyset$ and $\{t_1, t_2, t_3\}$, respectively. If t_1 of S' is the first transition selected in the loop $while$, then $R = SPS(t_1, M_0) = \{t_1, t_2\}$, $S' = S' - R = \{t_3\}$ and $SS = \emptyset \otimes R = \{\{t_1\}, \{t_2\}\}$. For the second iteration, t_3 is selected then

$R = SPS(t_3, M_0) = \{t_3, t_4\}$, $S' = S' - R = \emptyset$ and $SS = SS \otimes \{t_3, t_4\} = \{\{t_1, t_3\}, \{t_1, t_4\}, \{t_2, t_3\}, \{t_2, t_4\}\}$. Algorithm 1 returns SS.

- For M_0 and $S = En(M_0)$, Algorithm 1 returns the set:
  $SS = (\{\{t_1\}, \{t_2\}\} \otimes \{\{t_3, t_4\}\}) \otimes \{\{t_5, t_6\}\}$.
  Indeed, initially, we have $S' = S = En(M_0)$. The loop $while$ will perform successively the following updates of R, S' and SS, for the case where the selected transitions are successively t_1, t_3 and t_5:
  For t_1: $R = \{t_1, t_2\}$, $S' = \{t_3, t_4, t_5, t_6\}$ and $SS = \{\{t_1\}, \{t_2\}\}$.
  For t_3: $R = \{t_3, t_4\}$, $S' = \{t_5, t_6\}$ and $SS = SS \times R$.
  For t_5: $R = \{t_5, t_6\}$, $S' = \emptyset$ and then
  Then, $SS = (\{\{t_1\}, \{t_2\}\} \otimes \{t_3, t_4\}) \otimes \{t_5, t_6\}$. Note that in this case, SS is a covering-step set.
- For M_0 and $S' = S = \{t_1, t_2\}$, Algorithm 1 returns $SS = \{\{t_1\}, \{t_2\}\}$, as the transitions of S are all key-transitions (i.e., $SPS(t_1, M_0) = SPS(t_2, M_0) = S$). If t_1 (or t_2) is selected first then $R = SPS(t_1, M_0) = \{t_1, t_2\}$, $S' = S' - R = \emptyset$ and $SS = \{\{t_1\}, \{t_2\}\}$.

Theorem 2 *Algorithm 1 returns a set of steps that satisfies both $DS1$ and $DS2$ of the strong-persistent step sets.*

Proof Suppose that n ($n > 0$) iterations are needed to complete the loop $while$ of Algorithm 1. During the i^{th} iteration (for $i \in [1, n]$), a transition $t^{(i)}$ is selected from $S'^{(i)}$ and $R^{(i)} = SPS(t^{(i)}, M)$. The set $R^{(i)}$ is a strong-persistent set, which means that all its transitions are keys. Therefore, it holds that:

$$\forall i \in [1, n], \forall t^{(i)} \in R^{(i)}, \forall \omega \in \Omega(T - R^{(i)}), M[\omega\rangle \Rightarrow M[\omega t^{(i)}\rangle \wedge M[t^{(i)}\omega\rangle.$$

Consequently, SS satisfies $DS1$ and $DS2$. □

5.2 Computing weak-persistent step sets

To achieve further reductions, Algorithm 2 computes, in SS, a product of some pairwise disjunct weak-persistent sets, instead of strong-persistent sets. Unlike disjunct strong-persistent sets, the product of disjunct weak-persistent sets may contain some non enabled steps. These steps are deleted from SS to keep only the enabled ones. According to Theorem 3, SS is a weak-persistent step set.

Algorithm 2 Weak-persistent-step set of a marking M covering the transitions of S

```
1: Input : A marking M and a subset S of enabled transitions such that S ≠ ∅;
2: Output : A weak-persistent step set SS of M w.r.t. S;
3: SS := ∅; S' := S; R' := ∅;
4: while (∃t ∈ S' s.t. WPS(t, M) ∩ R' = ∅) do
5:     Choose t ∈ S' s.t. WPS(t, M) ∩ R' = ∅;
6:     R := WPS(t, M);
7:     S' := S' − R;
8:     R' := R' ∪ R;
9:     SS := SS ⊗ R;
10: end while
11: return SS ∩ EnS(M);
12: [For X ∈ 2^T and Y ⊆ T, X ⊗ Y = {x ∪ {y} | x ∈ X ∧ y ∈ Y} and ∅ ⊗ Y = {{y} |y ∈ Y}]
```

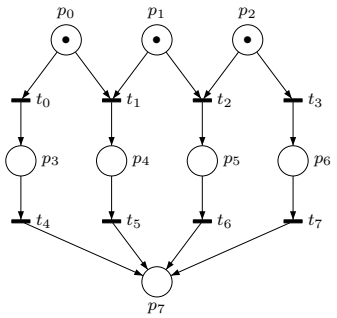

Fig. 5 Model PN2

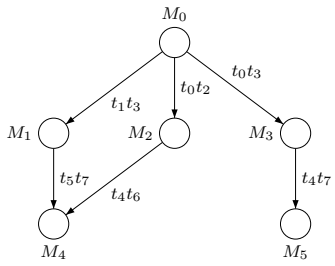

Fig. 6 WCSG: Covering-steps graph of PN2 (using Algorithm 2)

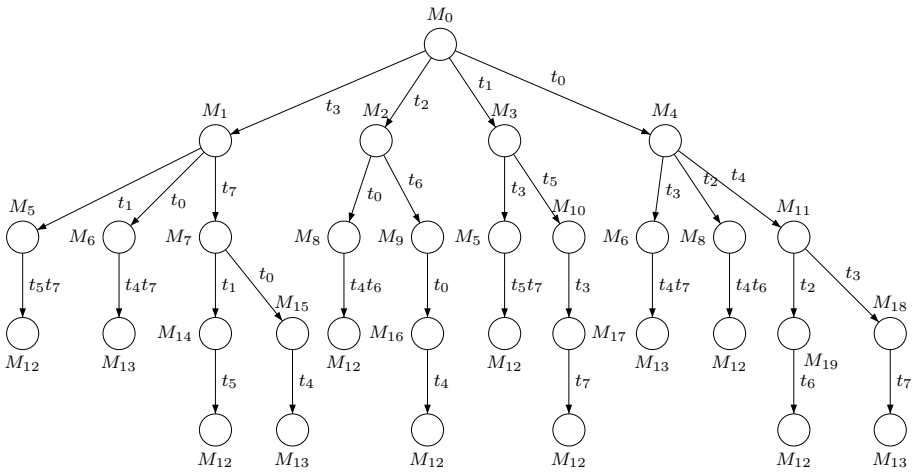

Fig. 7 SCSG: Covering-steps graph of PN2 (using the tool TINA)

Example 3 Consider the model PN2 at Fig.5 and its initial marking M_0.
For $S = \{t_0, t_1, t_2, t_3\}$, Algorithm 2 first sets SS, S' and R' to $\emptyset$, $\{t_0, t_1, t_2, t_3\}$ and $\emptyset$, respectively. Then, if t_0 of S' is the first transition selected in the loop $while$ on S', then $R = WPS(t_0, M_0) = \{t_0, t_1\}$, $S' = S' - R = \{t_2, t_3\}$, $SS = \emptyset \otimes R = \{\{t_0\}, \{t_1\}\}$ and $R' = R' \cup \{t_0, t_1\}$. For the second iteration, t_3 is selected, as: $WPS(t_2, M_0) \cap R' = \{t_1\}$ and $WPS(t_3, M_0) \cap R' = \emptyset$. Therefore, $R = WPS(t_3, M_0) = \{t_2, t_3\}$, $S' = S' - R = \emptyset$ and

$$SS = SS \otimes \{t_2, t_3\} = \{\{t_0, t_2\}, \{t_0, t_3\}, \{t_1, t_2\}, \{t_1, t_3\}\}.$$

Finally, Algorithm 2 returns $SS \cap EnS(M_0)$, i.e., $\{\{t_0, t_2\}, \{t_0, t_3\}, \{t_1, t_3\}\}$. The step $\{t_1, t_2\}$ is eliminated as it is not enabled in M_0.

Theorem 3 *Algorithm 2 returns a set of steps that satisfies conditions $DSw1$ and $DS2$ of the weak-persistent sets.*

Proof SS is a product of some pairwise disjunct persistent sets.
Suppose that $SS = R^{(1)} \otimes R^{(2)} .... \otimes R^{(n)}$ with $(n > 0)$.

Then, (i) $\forall i \in [1,n], \exists t^{(i)} \in R^{(i)}, \forall \omega \in \Omega(T - R^{(i)}), M[\omega\rangle \Rightarrow M[\omega t^{(i)}\rangle$ and
(ii) $\forall i \in [1,n], \forall t^{(i)} \in R^{(i)}, \forall \omega \in \Omega(T - R^{(i)}), M[\omega t^{(i)}\rangle \Rightarrow M[t^{(i)}\omega\rangle$. Condition (i) is satisfied for the key-transitions of each weak-persistent set. The step consisting of key-transitions of weak-persistent set is enabled in M. Eliminating the non enabled steps does not affect the validity of Condition (i). Condition (ii) is satisfied too as the eliminated steps are not enabled in M. Consequently, $SS \cap EnS(M)$ satisfies $DSw1$ and $D2$. □

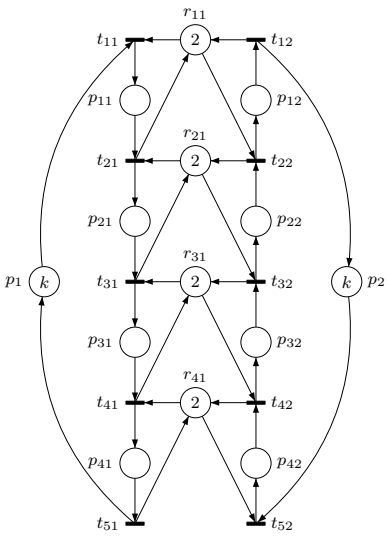

Fig. 8 Model PN3(k): A flexible manufacturing system (with deadlocks) taken from [3]

Example 4 For the model PN2 at Fig.5, the graphs depicted at Fig.6 and Fig.7 are obtained using the step selection Algorithm 2 and the tool TINA, respectively for the case where all the enabled transitions are covered from each marking. The graph at Fig.6 is a covering-steps graph obtained by recursively applying Algorithm 2 to the initial marking and each new computed marking. The step selective algorithms are always invoked for a marking M and $S = En(M)$.

Let us apply, Algorithm 1 to the initial marking M_0 for $S = En(M_0)$. It starts by $SS = \emptyset, S' = S = \{t_0, t_1, t_2, t_3\}$. If t_0 of S' is the first transition selected in the loop $while$ on S', then R is set to $SPS(t_0, M_0) = \{t_0, t_1, t_2, t_3\}$ and S' to $\emptyset$. Then, Algorithm 1 returns $SS = \emptyset \otimes R = \{\{t_0\}, \{t_1\}, \{t_2\}, \{t_3\}\}$.

Note that for M_0 and $S = En(M_0)$, Algorithm 2 returns the covering-step set $SS = \{\{t_0, t_2\}, \{t_0, t_3\}, \{t_1, t_3\}\}$ (see Example 3).

In this section, two parametric algorithms are presented for computing strong/weak-persistent step sets. They allow to specify, for each marking M, the subset S of enabled transitions to be covered by the set of steps. For $S = En(M)$, all the enabled transitions are covered by the provided steps, as for the covering-steps graphs. For $S = SPS(t, M)$, the realised combination corresponds to the one proposed in [12] for building persistent step graphs. In case the parameter S is a singleton then Algorithm 1 (resp. Algorithm 2) provides a set of singletons whose union is a strong

(resp. weak) persistent set. Therefore, these algorithms allows to compute not only persistent step sets but also strong/weak-persistent sets.

Another feature of these algorithms is the possibility to control the number of steps by fixing the dimension of the cartesian product of the strong/weak-persistent sets. Indeed, the loops $while$ in algorithms 1 and 2 can be, in addition, constrained by a given number of iterations or an appropriate number of steps. It is also possible to restrict the Cartesian product of strong/weak-persistent sets to singletons, except for one of them. More precisely, Line 9 of Algorithm 2 (Line 8 of Algorithm 1) is replaced with: if$(SS = \emptyset \vee |R| = 1)$ then $SS := SS \otimes R$. This modification allows to limit the number of steps and, at the same time, the redundancy of transitions between steps. These variants of algorithms 1 and 2 provide persistent step sets.

6 Experimental results

We have implemented and tested algorithms 1 and 2 on several ordinary Petri Nets and models derived from Petri nets at Fig.5, Fig.8 and Fig.9.

The model $\|_n\ PN2$ is a simple parallel composition of n instances of the model $PN2$ at Fig.5. The model $\|_n\ PN3(k)$ is a parallel composition of n instances of $PN3(k)$ at Fig.8, where the place p_2 of the i^{th} instance of $PN3(k)$ is merged with the place p_1 of the $(i+1)^{th}$ instance, for $i \in [1, n-1]$. Finally, the model $\|_n\ PN4(k)$ is a parallel composition of n instances of $PN4(k)$ at Fig.9, where the place p_{31} of the i^{th} instance of $PN4$ is merged with the place p_{11} of the $(i+1)^{th}$ instance, for $i \in [1, n-1]$. The last two models are Petri net representations of two flexible manufacturing systems. The first one is either free from deadlocks or not, depending on parameter k. The second one is free from deadlocks.

For these tests, the strong-persistent is computed using algorithms 3. This algorithm is based on the stubborn sets method [16]. Given a marking M and an enabled transition t, it starts by setting S to the singleton $\{t\}$. For each enabled transition t' within S, it adds to S the transitions in structural conflict with t'. For each non enabled transition t' within S, Algorithm 3 adds to S the input transitions of all the non marked input places of t'. This step ensures that, at the end of the process, for each non enabled transition t' in S, there is in S at least an enabled transition that must be fired before t'. This process is repeated until reaching a fixed point. Finally, the non enabled transitions are eliminated to obtain a strong-persistent set that satisfies conditions $D0$, $D1$ and $D2$ [16].

Table 1 is devoted to the comparison of the covering-steps and persistent set methods. It reports the sizes of the marking graph (MG), the covering-steps graphs (CSG, SCSG), the persistent graphs (PG, SPG) and the persistent step graph (PSG) for the three models described above. The CSG is the covering-steps graph provided by the tool TINA. The SCSG is the covering-steps graph computed using algorithms 1 and 3[6], for the case where all the enabled transitions are covered from each marking (i.e., $S = En(M)$). The PG is the persistent graph provided by the tool TINA. The SPG is the persistent graph computed using algorithms 1 and 3 for the case where S is a

[6] Algorithm 3 is used for $SPS(t, M)$.

Algorithm 3 $SPS(t, M)$: A strong-persistent set of the marking M containing, at least, the transition t

```
1: Input : A marking M and a transition t enabled in M;
2: Output : A strong-persistent set S of M for t;
3: S = {t};
4: repeat
5:     S' := S;
6:     for (t' ∈ S) do
7:         if (t' ∉ En(M)) then
8:             S := S ∪ {p ∈ •t' | p ∈ •t' ∧ M(p) = 0};
9:         else
10:            S := S ∪ CFS(t');
11:        end if
12:    end for
13: until (S = S')
14: S := S ∩ En(M);
15: return S;
```

singleton. The PSG combines the persistent sets with the step graph methods and is provided by the tool TINA. The last column reports the number of deadlock markings. For all tested models, the SCSG and SPG provide better results than the CSG, PG and PSG generated by the tool TINA. This improvement increases with the size of the models. Furthermore, the persistent sets methods outperform the covering-steps methods for the tested models. However, their combination outperforms the both methods. To achieve further reductions, the next section presents an algorithm for computing the weak-persistent sets to be used by Algorithm 2.

7 Towards weaker persistent sets

To our knowledge, in almost all partial order approaches, if an enabled transition is selected then all the enabled transitions in weak-conflict are selected too. In the following, we show how to weaken the selection procedure to handle, in a better way, the weak-conflicts for a subclass of ordinary Petri nets. This subclass excludes all impure ordinary Petri nets [7]. It is known that each impure Petri net can be transformed into a pure Petri net with the same behaviour, by adding appropriate dummy places and transitions [5]. Another restriction is to keep only pure ordinary Petri nets such that if a transition is enabled in some marking then, at most, one instance of each transition contributes to this enabledness. A structural sufficient condition for this property can be stated as: two or more input places $(p_1, p_2, ...)$ of the same transition t cannot be reached from two or more output transitions $(t_1, t_2, ...)$ of the same place p. If this property is not satisfied for some transition t and place p then all tokens that reach p could be transferred to the same input place of t leading to a dead transition or an unbounded place. Petri nets that do not satisfy this property are considered as

[7] An impure Petri net is a Petri net with one or more self-loops (i.e., place in the net, which is both an input and output place of the same transition).

Table 1 Experimental results for Algorithm 1 vs TINA

	MG	CSG TINA	SCSG $Alg.1\&3$ $S = En(M)$	PG TINA	SPG $Alg.3$ $(Alg.1\&3$ $\|S\| = 1)$	PSG TINA	Dead
$\|_4$ $PN2$							
Markings	160000	23138	1569	6905	226	6737	16
Edges	1024000	71801	2848	11048	285	10880	
CPU (s)	2	0.5	0	0	0	0	
$\|_5$ $PN2$							
Markings		265640	8833	48361	466	47681	32
Edges	> 3000000	952997	16576	80488	589	79808	
CPU (s)		3	0	0.3	0	0.3	
$\|_6$ $PN2$							
Markings			50817	333417	946	330689	64
Edges	> 3000000	> 3000000	97408	570792	1197	568064	
CPU (s)			0.4	2.5	0	3.2	
$\|_2$ $PN3(5)$							
Markings		403998	89714	50196	7853	54709	31
Edges	> 3000000	2579317	635090	114242	13580	138382	
CPU (s)		12.2	5.5	0.4	0	0.6	
$\|_3$ $PN3(2)$							
Markings	368905	359824	119605	101486	25775	101482	9
Edges	2623520	2221255	672509	263007	50509	266696	
CPU (s)	7	9.5	9.3	1	0.3	1.2	
$\|_3$ $PN3(3)$							
Markings				1051463	115391	1068339	16
Edges	> 3000000	> 3000000	> 3000000	2676615	220535	2868287	
CPU (s)				13.2	3.2	18	
$\|_1$ $PN4(5)$							
Markings	1251	1063	76	83	51	83	0
Edges	6526	3795	272	156	84	156	
CPU (s)	0	0	0	0	0	0	
$\|_2$ $PN4(5)$							
Markings			5109	3909	837	3909	0
Edges	> 3000000	> 3000000	32836	8568	1512	8568	
CPU (s)			0	0	0	0	
$\|_3$ $PN4(5)$							
Markings				143959	10935	143959	0
Edges	> 3000000	> 3000000	> 3000000	331668	20412	331668	
CPU (s)				1.7	0.4	2.3	

not well-handled[8] in [17]. In this paper, we call the subclass of Petri nets that satisfy this property as well-handled Petri nets.

Given a marking M and an enabled transition t in M, in Algorithm 3, t is first selected and if an enabled transition t' is selected then all the enabled transitions in weak-conflict with t' are selected too. We propose in Algorithm 4 to limit this selection to transitions in conflict t' that are not enabled in M or their firing may disable t' and enable it again later. Theorem 4 shows with this restrictive selection the returned set S satisfies both conditions $Dw1$ and $D2$ of the weak-persistent sets. To preserve all deadlocks of Petri nets, it suffices to ensure that Condition $D0$ is also satisfied. Condition $D0$ is satisfied if Algorithm 4 is called for each non deadlock marking reachable from the initial marking M_0, by the persistent set selective search.

For example, for the model $PN4$ at Fig.9 and its initial marking M_0, no matter which transition is selected first, Algorithm 3 will select all the enabled transitions in M_0. For M_0 and transition t_{11}, Algorithm 4 will select the transitions t_{11} and t_{21}. Indeed, at Line 3, S is set to $CFS(t_{11}) = \{t_{11}, t_{21}\}$. For t_{11}, the loop for will not add any transition to S, as all transitions of $CFS(t_{11})$ are already in S and enabled. For t_{21}, the transition t_{24} is the only transition, outside S, in structural conflict with t_{21}. This transition is enabled in M_0 and connected to t_{21} by a path but

[8] A Petri net is well-handled iff, for any pair of nodes x and y such that one of the nodes is a place and the other a transition and for any pair of elementary paths $C1$ and $C2$ leading from x to y, $nodes(C1) \cap nodes(C2) = \{x, y\} \Rightarrow C1 = C2$, where $nodes(C)$ is the set of nodes in the path C [17].

its firing does not disable t_{21}. Thus, it is not selected. Suppose that t_{21} is disabled later by another occurrence of t_{24} and enabled again after some non-empty firing sequence (as the model is a pure Petri net). The transitions that contribute to its new enabledness constitute an elementary sequence ω (as the model is a well-handled Petri net). Therefore, at most one instance of t_{24} contributes to the new enabledness of t_{21}. The sequence $t_{24}\omega$ can be fired before other instances of t_{24} to prevent the disabledness of t_{21}. This results in an equivalent sequence that maintains t_{21} enabled until its firing. This is the intuitive idea behind Algorithm 3. The resulting persistent graph is shown in Fig.10. Note that firing t_{11} or t_{21} from M_0 will disable neither t_{24} nor t_{31}. These transitions are maintained enabled as long as the fired transitions are enabled directly/indirectly by t_{11} or t_{21}. Their firings are postponed indefinitely in favour of t_{11}, t_{21} and the transitions they will enable directly/indirectly.

Algorithm 4 $WPS(t, M)$: A weak-persistent set of the marking M where t is a key-transition

1: Input : A marking M and a transition t enabled in M;
2: Output : A weak-persistent set S of M for t;
3: $S := CFS(t)$;
4: **repeat**
5: $S' := S$;
6: **for** $(t' \in S)$ **do**
7: **if** $(t' \notin En(M))$ **then**
8: $S := S \cup \{^\bullet p \mid p \in {}^\bullet t' \wedge M(p) = 0\}$;
9: **else**
10: **for** $(u \in CFS(t'))$ **do**
11: **if** $(u \notin En(M) \vee (\pi(u, t') \wedge \exists p \in {}^\bullet t' \cap {}^\bullet u, \sum_{v \in p^\bullet \cap En(M)} pre(v)(p) > M(p)))$ **then**
12: $S := S \cup \{u\}$;
13: **end if**
14: **end for**
15: **end if**
16: **end for**
17: **until** $(S = S')$
18: $S := S \cap En(M)$;
19: **return** S;
20: [$\pi(u, t')$ is true iff there is a path in the Petri net connecting u to t'.]

Theorem 4 *Algorithm 4 returns a set that satisfies both conditions $D1w$ and $D2$ of the weak-persistent sets for the well-handled Petri nets.*

Proof According to Line 3, the set S returned by Algorithm 4 contains $CFS(t) \cap En(M)$. Lines $7-8$ inside the two loops ($repeat$ and for) ensure that S includes all transitions of $En(M)$ that could enable directly or indirectly any transition of $CFS(t) - En(M)$. Therefore, S satisfies conditions $D1w$ and $D2$ of the weak-persistent sets for t. Indeed, the transitions outside S cannot enable any transition in conflict with t as long as no transition from S is fired.
Let us show that S satisfies $D2$, i.e., $\forall t' \in S, \forall \omega \in \Omega(T - S), M[\omega t'\rangle \Rightarrow M[t'\omega\rangle$.
1- If t' is not disabled during the firing of ω then $M[\omega t'\rangle \Rightarrow M[t'\omega\rangle$.
2- Otherwise, t' is disabled by some transition u and enabled again later. Let $\omega_1 u \omega_2 t' \omega_3 \equiv \omega$ be a sequence equivalent to ω such that only the transitions which have a role in the disabledness or in the enabledness of t' are kept in $\omega_1 u \omega_2$. If $u \in CFS(t') - En(M)$ then Algorithm 4 ensures to keep in S all the enabled transitions that could enable any transition in conflict with t'. It means that there is in ω_1

at least a transition from S, which is in contradiction with $\omega_1 \in \Omega(T-S)$. Consequently, all transitions in ω in conflict with t' are enabled in M. Let us now consider two cases:
2.1- If each transition of $(CFS(t') \cap En(M)) - S$ appears at most one time in the sequence $\omega_1 u$ then by Algorithm 4, u cannot disable t'. Indeed, Algorithm 4 ensures that firing one instance of each enabled transition in conflict with t' before t' will not disable t'.
2.2- If there is in the sequence $\omega_1 u$ at least a transition v from $CFS(t') \cap En(M)$ that appears two or more times then the other instances of v in ω_1 are not needed to enable again t' but needed to empty some input places of t'. Indeed, for the subclass of Petri nets considered in this section, at most one instance of each transition is needed to be fired before the enabledness of a transition. In the worst case, at most one instance of each transition in $\omega_1 u$ contributes with those of ω_2 to enable t'. Thus these extra instances can be shifted to be fired after t'. In the resulting sequence $\omega'_1 u \omega_2 t' \omega'_3$, the sequence ω_2 does not disable t' if t' is not disabled by $\omega'_1 u$. This is the case because, each transition of $CFS(t') \cap En(M)$ appears at most one time in the sequence $\omega'_1 u$ and by 2.1, firing all transitions of $CFS(t') \cap En(M)$ will not disable t'. Therefore, S satisfies $D2$.
To sum up S satisfies $D1w$ and $D2$ for the well-handled Petri nets. □

The comparison results of algorithms 3, 4 and the persistent graph of the tool TINA are reported in Table 2. As expected, for models with weak-conflicts, Algorithm 4 outperforms the others. For the model $\|_n\ PN4(5)$, the weak-persistent graph generated by Algorithm 4 has the same very small size, independently of the number of instances n of $PN4(5)$. Thus, it is not exaggerated to say that the weak-persistent sets methods are very promising to fight the state explosion and to address the verification of very large asynchronous concurrent systems, where the interplay between concurrency and weak-conflict is expanded. Table 3 shows the comparison results of the PSG of the tool TINA with the one provided using algorithms 2 and 4. The column $VWPSG$ is a variant of the $WPSG$ for the case where Line 9 of Algorithm 2 is replaced with: if$(SS = \emptyset \vee |R| = 1)$ then $SS := SS \otimes R$. The obtained results show that the possibility to control the size and number of steps may be very useful to achieve further reductions.

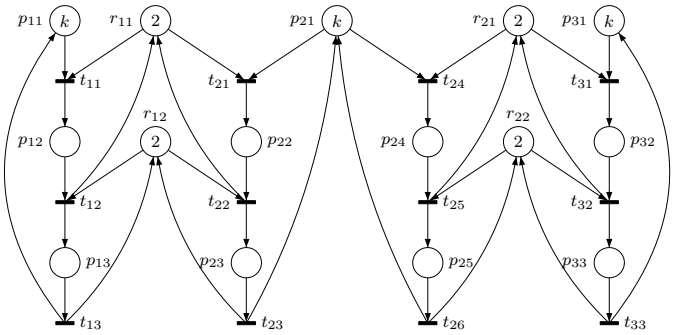

Fig. 9 Model $PN4(k)$: A flexible manufacturing system (free from deadlocks)

n this work, we have proposed a new parametric combination of the persistent sets with step graphs, based on a better understanding of the intricacy of the relations between concurrency and conflict, revealing local persistency and leading to a significant reduction.

Table 2 Experimental results for persistent graphs

	PG TINA	SPG $Alg.3$ ($Alg.1$&3 $\|S\|=1$)	WPG $Alg.4$ ($Alg.2$&4 $\|S\|=1$)	Dead
$\|_4$ $PN2$				
Markings	6905	226	136	16
Edges	11048	285	150	
CPU (s)	0	0	0	
$\|_5$ $PN2$				
Markings	48361	466	280	32
Edges	80488	589	310	
CPU (s)	0.3	0	0	
$\|_6$ $PN2$				
Markings	333417	946	568	64
Edges	570792	1197	630	
CPU (s)	2.5	0	0	
$\|_2$ $PN3(5)$				
Markings	50196	7853	1514	31
Edges	114242	13580	2353	
CPU (s)	0.4	0	0	
$\|_3$ $PN3(2)$				
Markings	101486	25775	23409	9
Edges	263007	50509	42423	
CPU (s)	1	0.3	0.2	
$\|_3$ $PN3(3)$				
Markings	1051463	115391	95356	16
Edges	2676615	220535	165027	
CPU (s)	13.2	3.2	0.7	
$\|_1$ $PN4(5)$				
Markings	83	51	10	0
Edges	156	84	14	
CPU (s)	0	0	0	
$\|_2$ $PN4(5)$				
Markings	3909	837	10	0
Edges	8568	1512	14	
CPU (s)	0	0	0	
$\|_3$ $PN4(5)$				
Markings	143959	10935	10	0
Edges	331668	20412	14	
CPU (s)	1.7	0.4	0	

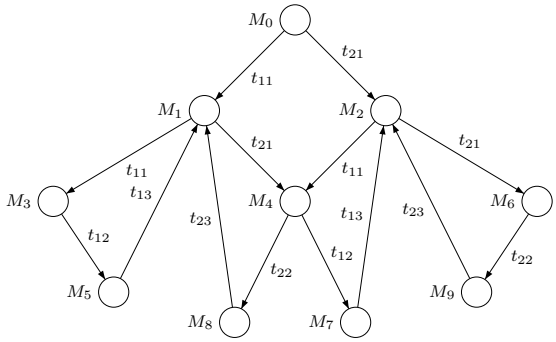

Fig. 10 A weak-persistent graph of $PN4(5)$

8 Conclusion

In this work, we have proposed a new parametric combination of the persistent sets with step graphs, based on a better understanding of the intricacy of the relations between concurrency and conflict, revealing local persistency and leading to a significant reduction. We have also shown how to weaken the sufficient conditions used to build persistent sets. The proposed approach takes into account, in a finer way, the structure of the net, while preserving deadlocks of Petri Nets. Indeed, unlike the method in [18], persistent steps may contain some transitions that are in weak-

Table 3 Experimental results for persistent step graphs

	PSG TINA	WPSG $Alg.\ 2\&4$ $S = SPS(t, M)$	VWPSG $Alg.\ 2\&4$ $S = SPS(t, M)$ with restriction	Dead
$\|_4$ $PN2$				
Markings	6737	121	136	16
Edges	10880	135	150	
CPU (s)	0	0	0	
$\|_5$ $PN2$				
Markings	47681	249	280	32
Edges	79808	279	310	
CPU (s)	0.3	0	0	
$\|_6$ $PN2$				
Markings	330689	505	568	64
Edges	568064	567	630	
CPU (s)	3.2	0	0	
$\|_2$ $PN3(5)$				
Markings	54709	8815	1448	31
Edges	138382	14599	2307	
CPU (s)	0.6	0	0	
$\|_3$ $PN3(2)$				
Markings	101482	26620	23201	9
Edges	266696	51336	44117	
CPU (s)	1.2	0.3	0.2	
$\|_3$ $PN3(3)$				
Markings	1068339	131966	98256	16
Edges	2868287	243487	177068	
CPU (s)	18	7	4.4	
$\|_1$ $PN4(5)$				
Markings	83	44	10	0
Edges	156	64	14	
CPU (s)	0	0	0	
$\|_2$ $PN4(5)$				
Markings	3909	1169	10	0
Edges	8568	1898	14	
CPU (s)	0	0	0	
$\|_3$ $PN4(5)$				
Markings	143959	28631	10	0
Edges	331668	51154	14	
CPU (s)	2.3	0.6	0	

Table 4 Reachable markings of the weak-persistent graph of $PN4(5)$

Marking M_0	$2r_{22} + 2r_{12} + 5p_{21} + 2r_{21} + 2r_{11} + 5p_{31} + 5p_{11}$
$En(M_0)$	$\{t_{11}, t_{21}, \mathbf{t_{24}}, \mathbf{t_{31}}\}$
Marking M_1	$2r_{22} + 2r_{12} + 1p_{12} + 5p_{21} + 2r_{21} + 1r_{11} + 5p_{31} + 4p_{11}$
$En(M_1)$	$\{t_{11}, t_{21}, \mathbf{t_{24}}, t_{12}, \mathbf{t_{31}}\}$
Marking M_2	$1p_{22} + 2r_{22} + 2r_{12} + 4p_{21} + 2r_{21} + 1r_{11} + 5p_{31} + 5p_{11}$
$En(M_2)$	$\{t_{11}, t_{21}, \mathbf{t_{24}}, t_{22}, \mathbf{t_{31}}\}$
Marking M_3	$2r_{22}, 2r_{12}, 2p_{12}, 5p_{21}, 2r_{21}, 5p_{31}, 3p_{11}\}$
$En(M_3)$	$\{\mathbf{t_{24}}, t_{12}, \mathbf{t_{31}}\}$
Marking M_4	$1p_{22} + 2r_{22} + 2r_{12} + 1p_{12} + 4p_{21} + 2r_{21} + 5p_{31} + 4p_{11}$
$En(M_4)$	$\{\mathbf{t_{24}}, t_{12}, t_{22}, \mathbf{t_{31}}\}$
Marking M_5	$2r_{22} + 1r_{12} + 1p_{12} + 1p_{13} + 5p_{21} + 2r_{21} + 1r_{11} + 5p_{31} + 3p_{11}$
$En(M_5)$	$\{t_{11}, t_{21}, \mathbf{t_{24}}, t_{12}, \mathbf{t_{31}}, t_{13}\}$
Marking M_6	$2p_{22} + 2r_{22} + 2r_{12} + 3p_{21} + 2r_{21} + 5p_{31} + 5p_{11}$
$En(M_6)$	$\{\mathbf{t_{24}}, t_{22}, \mathbf{t_{31}}\}$
Marking M_7	$1p_{22} + 2r_{22} + 1r_{12} + 1p_{13} + 4p_{21} + 2r_{21} + 1r_{11} + 5p_{31} + 4p_{11}$
$En(M_7)$	$\{t_{11}, t_{21}, \mathbf{t_{24}}, t_{22}, \mathbf{t_{31}}, t_{13}\}$
Marking M_8	$2r_{22} + 1r_{12} + 1p_{23} + 1p_{12} + 4p_{21} + 2r_{21} + 1r_{11} + 5p_{31} + 4p_{11}$
$En(M_8)$	$\{t_{11}, t_{21}, \mathbf{t_{24}}, t_{12}, \mathbf{t_{31}}, t_{23}\}$
Marking M_9	$1p_{22} + 2r_{22} + 1r_{12} + 1p_{23} + 3p_{21} + 2r_{21} + 1r_{11} + 5p_{31} + 5p_{11}$
$En(M_9)$	$\{t_{11}, t_{21}, \mathbf{t_{24}}, t_{22}, \mathbf{t_{31}}, t_{23}\}$

conflict. Moreover, it allows choosing the transitions to be covered while controlling the length and the number of steps to be selected from each marking. It can thus compute persistent graphs, covering steps graphs and graphs generated from different kinds of their combination.

Finally, the performed tests show the effectiveness of the proposed approach in terms of state space reduction, relatively to the covering-steps and strong-persistent sets methods or their combination implemented in the tool TINA. They also suggest that combining appropriately step graphs with any partial order technique is of very great interest for model-checking.

References

1. Kamel Barkaoui, Jean-Michel Couvreur, and Kais Klai. On the equivalence between liveness and deadlock-freeness in petri nets. In *Applications and Theory of Petri Nets 2005, 26th International Conference, ICATPN 2005, Miami, USA, June 20-25, 2005, Proceedings*, pages 90–107, 2005.
2. Kamel Barkaoui and Jean-François Pradat-Peyre. On liveness and controlled siphons in petri nets. In *Application and Theory of Petri Nets 1996, 17th International Conference, Osaka, Japan, June 24-28, 1996, Proceedings*, pages 57–72, 1996.
3. Yufeng Chen. *Optimal Supervisory Control of Flexible Manufacturing Systems*. Thesis. Le Cnam, 2015.
4. YuFeng Chen, ZhiWu Li, and Kamel Barkaoui. New Petri net structure and its application to optimal supervisory control: Interval inhibitor arcs. *IEEE Transactions on Systems, Man, and Cybernetics*, 44(10):1384–1400, 2014.
5. René David and Hassane Alla. "Du grafcet aux réseaux de Petri". In *Hermés*, 1989.
6. Jörg Desel and Gabriel Juhás. "what is a petri net?". In *Unifying Petri Nets, Advances in Petri Nets*, pages 1–25, 2001.
7. Patrice Godefroid. *Partial-Order Methods for the Verification of Concurrent Systems - An Approach to the State-Explosion Problem*, volume 1032 of *Lecture Notes in Computer Science*. Springer, 1996.
8. Tomm Junttila. On the symmetry reduction method for petri nets and similar formalisms. In *Ph.D. dissertation, Helsinki University of Technology, Espoo, Finland*, 2005.
9. ZhiWu Li and Mi Zhao. On controllability of dependent siphons for deadlock prevention in generalized petri nets. *IEEE Transactions on Systems, Man, and Cybernetics*, 38(2):369–384, 2008.
10. Doron Peled. All from one, one for all: on model checking using representatives. In *Computer Aided Verification, 5th International Conference, CAV '93, Elounda, Greece, June 28 - July 1, 1993, Proceedings*, pages 409–423, 1993.
11. Doron Peled and Thomas Wilke. Stutter-invariant temporal properties are expressible without the next-time operator. *Inf. Process. Lett.*, 63(5):243–246, 1997.
12. Pierre-Olivier Ribet, François Vernadat, and Bernard Berthomieu. On combining the persistent sets method with the covering steps graph method. In *Formal Techniques for Networked and Distributed Systems - FORTE 2002, 22nd IFIP WG 6.1 International Conference Houston, Texas, USA, November 11-14, 2002, Proceedings*, pages 344–359, 2002.
13. Antti Valmari. A stubborn attack on state explosion. *Formal Methods in System Design*, 1(4):297–322, 1992.
14. Antti Valmari. The state explosion problem. In *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the volumes are based on the Advanced Course on Petri Nets, held in Dagstuhl, September 1996*, pages 429–528, 1996.
15. Antti Valmari and Henri Hansen. Can stubborn sets be optimal? *Fundam. Inform.*, 113(3-4):377–397, 2011.
16. Antti Valmari and Henri Hansen. Stubborn set intuition explained. *T. Petri Nets and Other Models of Concurrency*, 12:140–165, 2017.
17. Wil van der Aalst. Finding errors in the design of a workflow process : a petri-net-based approach. 1998.
18. François Vernadat, Pierre Azéma, and François Michel. Covering step graph. In *Application and Theory of Petri Nets 1996, 17th International Conference, Osaka, Japan, June 24-28, 1996, Proceedings*, pages 516–535, 1996.