International Journal of Foundations of Computer Science
© World Scientific Publishing Company

ON THE EFFICIENT COMPUTATION OF THE MINIMAL COVERABILITY SET OF PETRI NETS*

GILLES GEERAERTS

JEAN-FRANÇOIS RASKIN

LAURENT VAN BEGIN

Université Libre de Bruxelles, Département d'Informatique, CPI–212
Boulevard du Triomphe, B-1050 Bruxelles Belgium
{gigeerae, jraskin, lvbegin}@ulb.ac.be

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

The *minimal coverability set* (MCS) of a Petri net is a finite representation of the downward-closure of its reachable markings. The minimal coverability set allows to decide several important problems like coverability, semi-liveness, place boundedness, etc. The classical algorithm to compute the MCS constructs the Karp&Miller (KM) tree [10]. Unfortunately the KM tree is often huge, even for small nets. An improvement of this KM algorithm is the Minimal Coverability Tree (MCT) algorithm [3], which has been introduced nearly 20 years ago, and implemented since then in several tools such as Pep [9]. Unfortunately, we show in this paper that the MCT is flawed: it might compute an under-approximation of the reachable markings. We propose a new solution for the efficient computation of the MCS of Petri nets. Our algorithm is based on new ideas, and the experimental results show that it behaves much better in practice than the KM algorithm.

Keywords: Petri nets, Coverability set

1. Introduction

Petri nets [12] are a very popular formalism for the modeling and verification of parametric and concurrent systems [8]. Although the underlying transition graph of a Petri net is potentially infinite, a large number of interesting verification problems are decidable on this model. Among these decidable problems are the *coverability* problem (to which many safety verification problem can be reduced); the *boundedness* problem (is the number of reachable markings finite ?); the *place boundedness*

*Work supported by the projects: (*i*) Quasimodo: 'Quantitative System Properties in Model-Driven-Design of Embedded', `http://www.quasimodo.aau.dk/`, (*ii*) Gasics: 'Games for Analysis and Synthesis of Interactive Computational Systems', `http://www.ulb.ac.be/di/gasics/`, and (*iii*) Moves: 'Fundamental Issues in Modelling, Verification and Evolution of Software', `http://moves.vub.ac.be`, a PAI program funded by the Federal Belgian Government.

problem (is the maximal reachable number of tokens bounded for some place p ?); the *semi-liveness* problem (is there a reachable marking in which some transition t is enabled ?).

In order to decide these problems, a classical technique consists in building a so-called *coverability set* of the net, a notion introduced by Finkel [2]. This set is a *finite representation* of some over-approximation of the reachable markings. Coverability sets are thus very useful tools for the analysis of Petri net, and an efficient algorithm to compute them is highly desirable.

The history of algorithms for computing a coverability set of a Petri net spans a long period of time. The seminal paper is that of Karp and Miller [10] where they study, among others, decidability problems on *Vector Addition Systems* (VAS), a formalism which, in their setting, is identical to Petri net. The decidability results are established thanks to a procedure, that, given a Petri net (or VAS), builds a finite labeled tree (the so-called Karp & Miller tree) that summarizes the potentially infinite reachability set of the Petri net.

In particular, this algorithm relies on an *acceleration technique*, to compute the limit of repeating any number of times some sequence of transitions that strictly increases the number of tokens in a subset of the places. The acceleration technique is sound because of two well-known properties of Petri nets. First, Petri nets are *monotonic*. This property relies on the ordering $\preccurlyeq$ over markings of PN, where $\mathbf{m} \preccurlyeq \mathbf{m}'$ means, roughly speaking, that the markings $\mathbf{m}'$ assigns more tokens to each place than the markings $\mathbf{m}$ does. Then the monotonicity property states that, if a sequence of transitions can be fired from a marking $\mathbf{m}_1$ and produces a marking $\mathbf{m}_2$, then, the same sequence of transitions can be fired from any marking $\mathbf{m}_3 \succcurlyeq \mathbf{m}_1$, and this will produce a marking $\mathbf{m}_4 \succcurlyeq \mathbf{m}_2$. Second, transitions of Petri nets have *constant effect*: they add and subtract in each place the same number of tokens no matter from which marking they are fired. These two properties imply that a sequence of transitions that strictly increases the number of tokens in a subset P' of the places (while not modifying the number of tokens in the other places) can be fired an arbitrary number of times, and the places of P' are therefore unbounded.

Although Karp and Miller do not state it explicitly, the information contained in this tree can be used to obtain a coverability set of the Petri net. Unfortunately, the Karp & Miller algorithm is generally useless in practice because the Karp & Miller tree, albeit finite, cannot be constructed in reasonable time for Petri nets modeling realistic systems. As a consequence, a more efficient algorithm is needed. In [3], such an algorithm, called the Minimal Coverability Tree (MCT), is proposed. The MCT builds on the idea of the Karp & Miller tree but tries to take advantage more aggressively of the monotonicity of Petri nets. Indeed, several[a] coverability sets can be associated to the a single Petri net, and all of them represent the same over-approximation of the reachable markings. However, there exists a unique *minimal* coverability set which is the most compact of them all. Given any coverability set,

[a]Potentially, infinitely many for unbounded Petri nets.

it is trivial to extract the minimal coverability set from it (hence, the minimal coverability set can be extracted from the Karp & Miller tree), because the MCS contains the maximal elements (wrt to $\preccurlyeq$) of any coverability set. However, it is a difficult task to *directly* compute the minimal coverability set, by avoiding building *redundant* parts of the Karp & Miller tree, and keeping the set of constructed markings *minimal* at all times during the execution of the algorithm.

This is the purpose of the MCT algorithm. Its main idea is to construct a tree where all markings that label nodes are incomparable wrt $\preccurlyeq$. To achieve this goal, reduction rules are applied at each step of the algorithm: each time a new marking is computed, it is compared to the other markings. If the new marking is smaller than an existing marking, the construction is not pursued from this new marking. If the new marking is larger than an existing marking, the subtree starting from that smaller marking is removed. The informal justification for this is as follows: the markings that are reachable from removed markings will be covered by markings reachable from the marking that was used for the removal, by the monotonicity property of Petri nets. While this idea is appealing and leads to small trees in practice, we show in this paper that, unfortunately, it is not correct: the MCT algorithm is not complete and can compute a strict under-approximation of the minimal coverability set. The flaw is intricate and we do not see an easy way to get rid of it.

So, instead of trying to fix the MCT algorithm, we consider the problem from scratch and propose a new efficient method to compute the minimal coverability set. It is based on novel ideas: first, we do not build a tree but handle sets of pairs of markings. Second, in order to exploit the monotonicity property, we define an adequate order on pairs of markings that allows us to maintain sets of maximal pairs only. As a consequence, our new solution manipulates sets that are minimal too (as the MCT intends to do), but for a different ordering (an ordering on pairs, instead of an ordering on markings).

We give in this paper a detailed proof of correctness for this new method, and explain how to turn it into an efficient algorithm for the computation of the minimal coverability set of practically relevant Petri nets. We have implemented our algorithm in a prototype and compared its performance with the Karp & Miller algorithm. Our algorithm is often much faster than the Karp & Miller algorithm.

The rest of the paper is organized as follows. In Section 2, we recall necessary preliminaries. In Section 3, we recall the Karp & Miller and the MCT algorithms. In Section 4, we expose the bug in the MCT algorithm using an example and explain the essence of the flaw. In Section 5, we define the covering sequence, a sequence of sets of pairs of ω-markings that allows to compute the minimal coverability set. In Section 6, we discuss a prototype implementation of this new method.

2. Preliminaries

Petri nets Let us first recall the model of Petri nets, and fix several notations. Let $\mathbb{N}$ denote the set of natural numbers (including 0). Then:

Definition 1. *A* Petri net [12] *(PN for short) is a tuple $\mathcal{N} = \langle P, T\rangle$, where $P = \{p_1, p_2, \ldots, p_{|P|}\}$ is a finite set of places and $T = \{t_1, t_2, \ldots, t_{|T|}\}$ is a finite set of transitions. Each transition is a tuple $\langle I, O\rangle$, where $I : P \mapsto \mathbb{N}$ and $O : P \mapsto \mathbb{N}$ are respectively the input and output functions of the transition.*

To define the semantics of PN, we first introduce the notion of ω-marking. An *ω-marking* $\mathbf{m}$ is a function $\mathbf{m} : P \mapsto (\mathbb{N} \cup \{\omega\})$ that associates a number of *tokens* to each place (where ω means 'any natural number'). An ω-marking $\mathbf{m}$ is denoted either as $\big\langle \mathbf{m}(p_1), \mathbf{m}(p_2), \ldots, \mathbf{m}(p_{|P|})\big\rangle$ (vector), or as $\{\mathbf{m}(p_{i_1})p_{i_1}, \mathbf{m}(p_{i_2})p_{i_2}, \ldots, \mathbf{m}(p_{i_k})p_{i_k}\}$ (multiset), where $p_{i_1}, p_{i_2}, \ldots, p_{i_k}$ are exactly the places that contain at least one token (we omit $\mathbf{m}(p)$ when it is equal to 1). For example, $\langle 0, 1, 0, \omega, 2\rangle$ and $\{p_2, \omega p_4, 2p_5\}$ denote the same ω-marking. An ω-marking $\mathbf{m}$ is a *marking* iff $\forall p \in P : \mathbf{m}(p) \neq \omega$.

Let $\mathcal{N} = \langle P, T\rangle$ be a PN, $\mathbf{m}$ be an ω-marking of $\mathcal{N}$ and $t = \langle I, O\rangle \in T$ be a transition. Then, t is *enabled* in $\mathbf{m}$ iff $\mathbf{m}(p) \geq I(p)$ for all $p \in P$ (we assume that $\omega \geq \omega$ and $\omega > c$ for any $c \in \mathbb{N}$). In that case, t can *fire* and transforms $\mathbf{m}$ into a new ω-marking $\mathbf{m}'$ s.t. for any $p \in P$: $\mathbf{m}'(p) = \mathbf{m}(p) - I(p) + O(p)$ (assuming that $\omega - c = \omega = \omega + c$ for any $c \in \mathbb{N}$). We denote this by $\mathbf{m} \xrightarrow{t} \mathbf{m}'$, and extend the notation to sequences of transitions $\sigma = t_1 t_2 \cdots t_k \in T^*$, i.e., $\mathbf{m} \xrightarrow{\sigma} \mathbf{m}'$ iff either $\sigma = \varepsilon$ (the empty sequence) and $\mathbf{m} = \mathbf{m}'$, or there are $\mathbf{m}_1, \ldots, \mathbf{m}_{k-1}$ s.t. $\mathbf{m} \xrightarrow{t_1} \mathbf{m}_1 \xrightarrow{t_2} \cdots \xrightarrow{t_k} \mathbf{m}'$. Given an ω-marking $\mathbf{m}$ of some PN $\mathcal{N} = \langle P, T\rangle$, we let $\mathsf{Post}(\mathbf{m}) = \{\mathbf{m}' \mid \exists t \in T : \mathbf{m} \xrightarrow{t} \mathbf{m}'\}$ and $\mathsf{Post}^*(\mathbf{m}) = \{\mathbf{m}' \mid \exists \sigma \in T^* : \mathbf{m} \xrightarrow{\sigma} \mathbf{m}'\}$. Given a sequence of transitions $\sigma = t_1 t_2 \cdots t_k$ with $t_i = \langle I_i, O_i\rangle$ for any $1 \leq i \leq k$, we let, for any place p, $\sigma(p) = \sum_{i=1}^{k}(I_i(p) - O_i(p))$, i.e., the effect of σ on p.

In the following, we use the partial order $\preccurlyeq$ for ω-markings.

Definition 2. *Let P be a set of places of some PN. Then, $\preccurlyeq \subseteq (\mathbb{N} \cup \{\omega\})^{|P|} \times (\mathbb{N} \cup \{\omega\})^{|P|}$ is s.t. for any $\mathbf{m}_1, \mathbf{m}_2$, $\mathbf{m}_1 \preccurlyeq \mathbf{m}_2$ iff for any $p \in P$: $\mathbf{m}_1(p) \leq \mathbf{m}_2(p)$.*

We write $\mathbf{m} \prec \mathbf{m}'$ when $\mathbf{m} \preccurlyeq \mathbf{m}'$ but $\mathbf{m} \neq \mathbf{m}'$. Finally, it is well-known that PN are *strictly monotonic*. That is, if $\mathbf{m}_1$, $\mathbf{m}_2$ and $\mathbf{m}_3$ are three markings and t is a transition of some PN $\mathcal{N}$ s.t. $\mathbf{m}_1 \xrightarrow{t} \mathbf{m}_2$ and $\mathbf{m}_1 \prec \mathbf{m}_3$, then, t is enabled in $\mathbf{m}_3$ and the marking $\mathbf{m}_4$ with $\mathbf{m}_3 \xrightarrow{t} \mathbf{m}_4$ is s.t. $\mathbf{m}_2 \prec \mathbf{m}_4$.

Example 3. *A simple example of Petri net is given in Fig. 1. This PN has three places p_1, p_2 and p_3 and two transitions: t_1 and t_2. The initial marking, $\langle 1, 0, 0\rangle$ is shown on the picture.*

Covering and coverability sets Given a set M of ω-markings, we define the set of maximal elements of M as $\mathsf{Max}^{\preccurlyeq}(M) = \{\mathbf{m} \in M \mid \nexists \mathbf{m}' \in M : \mathbf{m} \prec \mathbf{m}'\}$. Given

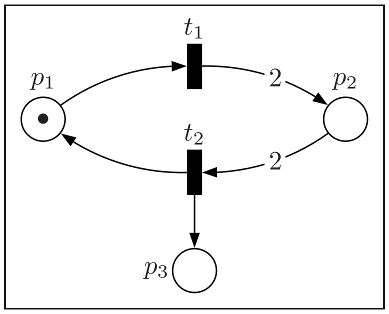


Fig. 1. An example of Petri net

an ω-marking $\mathbf{m}$ (ranging over set of places P), its *downward-closure* is the set *of markings* $\downarrow^{\preccurlyeq}(\mathbf{m}) = \{\mathbf{m'} \in \mathbb{N}^{|P|} \mid \mathbf{m'} \preccurlyeq \mathbf{m}\}$. Given a set M of ω-markings, we let $\downarrow^{\preccurlyeq}(M) = \cup_{\mathbf{m}\in M}\downarrow^{\preccurlyeq}(\mathbf{m})$. A set $D \subseteq \mathbb{N}^{|P|}$ is *downward-closed* iff $\downarrow^{\preccurlyeq}(D) = D$. Then:

Definition 4. *Let $\mathcal{N} = \langle P, T\rangle$ be a* PN *and let $\mathbf{m}_0$ be the initial ω-marking of $\mathcal{N}$. The* covering set *of $\mathcal{N}$, denoted as* $\mathsf{Cover}(\mathcal{N}, \mathbf{m}_0)$ *is the set* $\downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$.

Given a PN $\mathcal{N}$ with initial marking $\mathbf{m}_0$, a *coverability set* for $\mathcal{N}$ and $\mathbf{m}_0$ is a finite sub-set $S \subseteq (\mathbb{N} \cup \{\omega\})^{|P|}$ such that $\downarrow^{\preccurlyeq}(S) = \mathsf{Cover}(\mathcal{N}, \mathbf{m}_0)$. Such a set always exists because any downward-closed set of markings can be represented by a finite set of ω-markings:

Lemma 5 ([14]) *For any subset $D \subseteq \mathbb{N}^k$ such that $\downarrow^{\preccurlyeq}(D) = D$ there exists a finite subset $S \subset (\mathbb{N} \cup \{\omega\})^k$ such that $\downarrow^{\preccurlyeq}(S) = D$.*

It is also well-known [3] that there exists one minimal (in terms of $\subseteq$) coverability set (called the *minimal coverability set*).

Example 6. *The (infinite) reachability set of the* PN *in Fig. 1 is* $\{\langle 1,0,i\rangle \mid i \in \mathbb{N}\} \cup \{\langle 0,2,i\rangle \mid i \in \mathbb{N}\}$. *The downward closure of this set forms the covering set. The sets* $S_1 = \{\langle 0,0,0\rangle, \langle 1,0,\omega\rangle, \langle 0,1,5\rangle, \langle 0,2,\omega\rangle\}$ *and* $S_2 = \{\langle 1,0,\omega\rangle, \langle 0,2,\omega\rangle\}$ *both form a coverability set of the* PN. S_2 *is the minimal coverability set.*

Labeled trees Finally, let us introduce the notion of *labeled tree*:

Definition 7. *Given a set of places P, a labeled tree is a tuple $\mathcal{T} = \langle N, B, root, \Lambda\rangle$, s.t. $\langle N, B, root\rangle$ forms a tree (N is the set of nodes, $B \subseteq N \times N$ is the set of edges and $root \in N$ is the root node) and $\Lambda : N \mapsto (\mathbb{N} \cup \{\omega\})^{|P|}$ is a labeling function of the nodes by ω-markings.*

Given two nodes n and n' in N, we write respectively $B(n, n')$, $B^*(n, n')$ $B^+(n, n')$ instead of $(n, n') \in B$, $(n, n') \in B^*$, $(n, n') \in B^+$.

3. The Karp&Miller and the MCT algorithms

The Karp and Miller algorithm The Karp&Miller algorithm [10] is a well-known solution to compute a coverability set of a PN. It consists in building a labeled tree whose root is labeled by $\mathbf{m}_0$. The tree is obtained by unfolding the transition relation of the PN, and by applying *accelerations*, which exploit the strict monotonicity property of PN. That is, let us assume that $\mathbf{m}_1$ and $\mathbf{m}_2$ are two ω-markings s.t. $\mathbf{m}_1 \prec \mathbf{m}_2$ and there exists a sequence of transitions σ with $\mathbf{m}_1 \xrightarrow{\sigma} \mathbf{m}_2$. By (strict) monotonicity, σ is firable from $\mathbf{m}_2$ and produces an ω-marking $\mathbf{m}_3$ s.t. $\mathbf{m}_2 \prec \mathbf{m}_3$. As a consequence, all the places p s.t. $\mathbf{m}_1(p) < \mathbf{m}_2(p)$ are unbounded. Hence, the ω-marking $\mathbf{m}_\omega$ defined as

$$\mathbf{m}_\omega(p) = \begin{cases} \omega & \text{if } \mathbf{m}_1(p) < \mathbf{m}_2(p) \\ \mathbf{m}_1(p) & \text{otherwise} \end{cases}$$

has the property that$\downarrow^{\preccurlyeq}(\mathbf{m}_\omega) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_1))$. This can be generalized to the case where we consider an ω-marking $\mathbf{m}$ and a set S of ω-markings s.t. for any $\mathbf{m}' \in S$: $\mathbf{m} \in \mathsf{Post}^*(\mathbf{m}')$. These are the main ideas behind the following acceleration function:

$$\forall p \in P : \mathsf{Accel}(S, \mathbf{m})(p) = \begin{cases} \omega & \text{if } \exists \mathbf{m}' \in S : \mathbf{m}' \prec \mathbf{m} \text{ and } \mathbf{m}'(p) < \mathbf{m}(p) \\ \mathbf{m}(p) & \text{Otherwise} \end{cases}$$

The Karp&Miller procedure (see Algorithm 1) relies on this function: when developing the successors of a node n, it calls the acceleration function on every $\mathbf{m} \in \mathsf{Post}(\Lambda(n))$, by letting S be the set of all the markings that are met along the branch ending in n. This procedure terminates and computes a coverability set:

Theorem 8 ([10]) *For any PN $\mathcal{N} = \langle P, T\rangle$ with initial ω-marking $\mathbf{m}_0$, the KM procedure produces a finite labeled tree $\mathcal{T} = \langle N, B, root, \Lambda\rangle$, s.t. $\downarrow^{\preccurlyeq}(\{\Lambda(n) \mid n \in N\}) = \mathsf{Cover}(\mathcal{N}, \mathbf{m}_0)$.*

Properties of the Karp&Miller tree Let $n \neq root$ be a node of some Karp&Miller tree. Hence, $\Lambda(n)$ has been obtained by calling Accel with parameters S and $\mathbf{m}$. In this case, we say that n has been obtained *by the acceleration of* $\mathbf{m}$ (with S). For any node $n \neq root$ of any Karp&Miller tree, we assume that $\mathsf{M}(n)$ is the marking $\mathbf{m}$ s.t. $\Lambda(n)$ has been obtained by the acceleration of $\mathbf{m}$. Remark that $\forall n \neq root$, $\mathsf{M}(n) \in \mathsf{Post}(\Lambda(n'))$ where n' is the father of n, and it might be the case that $\Lambda(n) = \mathsf{M}(n)$.

Let $\mathcal{N} = \langle P, T\rangle$ be a PN with initial marking $\mathbf{m}_0$ and let $\mathcal{T} = \langle N, B, root, \Lambda\rangle$ be its Karp&Miller tree. Then, $\varsigma : N \mapsto T^*$ is a function that associates a sequence of transitions to every node n, as follows. (i) If $n = root$, then $\varsigma(n)$ returns the empty sequence. (ii) If there is no $n' \in N$ s.t. $B^+(n', n)$ and $\Lambda(n') \prec \Lambda(n)$ (hence, n is such that $\Lambda(n) = \mathsf{M}(n)$), then $\varsigma(n)$ returns the empty sequence. (iii) Otherwise, $\Lambda(n)$ has been obtained by the acceleration of $\mathsf{M}(n)$. Let $P_a = \{p \in P \mid \Lambda(n)(p) = \omega \text{ and } \mathsf{M}(n)(p) \neq \omega\}$ and let $P_\omega = \{p \in P \mid \Lambda(n)(p) = \mathsf{M}(n)(p) = \omega\}$. In that case,

```
Data: A PN $\mathcal{N} = \langle P, T \rangle$ and an initial $\omega$-marking $\mathbf{m}_0$.
Result: The minimal coverability set of $\mathcal{N}$ for $\mathbf{m}_0$.
$\mathsf{KM}(\mathcal{N}, \mathbf{m}_0)$ begin
    $\mathcal{T} \leftarrow \langle N, B, n_0, \Lambda \rangle$ where $N = \{n_0\}$, $B = \emptyset$ and $\Lambda(n_0) = \mathbf{m}_0$ ;
    $to\_treat \leftarrow \{n_0\}$ ;
    while $to\_treat \neq \emptyset$ do
        Choose and remove a node $n$ from $to\_treat$;
        if $\nexists \overline{n} : B^+(\overline{n}, n) \wedge \Lambda(\overline{n}) = \Lambda(n)$ then
            foreach $\mathbf{m} \in \mathsf{Post}(\Lambda(n))$ do
                $S \leftarrow \{\Lambda(n') \mid B^*(n', n)\}$ ;
                Let $n'$ be a new node s.t. $\Lambda(n') = \mathsf{Accel}(S, \mathbf{m})$ ;
                $N \leftarrow N \cup \{n'\}$ ;
                $B \leftarrow B \cup \{(n, n')\}$ ;
                $to\_treat \leftarrow to\_treat \cup \{n'\}$ ;
    return($\{\Lambda(n) \mid n \in N \wedge \nexists n' \in N : \Lambda(n') \succ \Lambda(n)\}$) ;
end
```

Algorithm 1: The KM algorithm.

$\varsigma(n)$ returns one of the finite non-empty sequences s.t. for any $p \in P_a$: $\varsigma(n)(p) > 0$; for any $p \in P \setminus (P_a \cup P_\omega)$: $\varsigma(n)(p) = 0$; and $\varsigma(n)$ is firable from $\mathsf{M}(n)$.

The existence of $\varsigma(n)$ in the third case is guaranteed by the following lemma, that can be extracted from the main proof of the Algorithm 1, in [10]:

Lemma 9 ([10]) *Let $\mathcal{N} = \langle P, T \rangle$ be a PN with initial ω-marking $\mathbf{m}_0$ and let $\mathcal{T} = \langle N, B, root, \Lambda \rangle$ be its Karp&Miller tree. Let $n \neq root$ be a node of $\mathcal{T}$. Let $P_a = \{p \in P \mid \Lambda(n)(p) = \omega$ and $\mathsf{M}(n)(p) \neq \omega\}$ and $P_\omega = \{p \in P \mid \Lambda(n)(p) = \mathsf{M}(n)(p) = \omega\}$. Then, there exists a sequence of transitions $\sigma \in T^*$ s.t.: (i) for any $p \in P_a$: $\sigma(p) > 0$. (ii) for any $p \in P \setminus (P_a \cup P_\omega)$: $\sigma(p) = 0$. (iii) σ is firable from $\mathsf{M}(n)$.*

Two other properties of the K&M tree are given by Lemma 10 hereunder, which states that, for every node n' of the K&M tree that is different from the root, $\downarrow^{\preccurlyeq}(\Lambda(n'))$ is included in $\downarrow^{\preccurlyeq}(\mathsf{Post}^*(n))$, where n is the father of n'. This is consistent with the intuition that the children of a node n can be obtained by unrolling a (possibly infinite) sequence of transitions firable from n.

Lemma 10 ([10]) *Let $\mathcal{N} = \langle P, T \rangle$ be a PN with initial ω-marking $\mathbf{m}_0$ and let $\mathcal{T} = \langle N, B, root, \Lambda \rangle$ be its Karp&Miller tree. Then, for every n,n' in N s.t. $B(n, n')$: $\downarrow^{\preccurlyeq}(\Lambda(n')) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(n))$.*

The MCT algorithm The *minimal coverability tree algorithm* (MCT for short) has been introduced by Finkel in [3], as an optimization of the Karp&Miller algorithm. It is recalled in Algorithm 2, and relies on two auxiliary functions: given a labeled tree $\mathcal{T}$ and a node n of $\mathcal{T}$, $\mathsf{removeSubtree}(n, \mathcal{T})$ removes the subtree rooted by n from $\mathcal{T}$. The function $\mathsf{removeSubtreeExceptRoot}(n, \mathcal{T})$ is similar to $\mathsf{removeSubtree}(n, \mathcal{T})$ except that the root node n is not removed. The main idea consists in exploiting the monotonicity property of PN in order (i) to avoid developing part of the nodes of the Karp&Miller tree, and (ii) to remove some subtrees during the construction. With respect to the Karp&Miller algorithm, four main differences can be noted. Let n be a node picked from *to_treat*. First, when there already exists another node $\overline{n}$ with $\Lambda(n) = \Lambda(\overline{n})$ in the tree[b], n is not developed (line (a)). Second, when n is accelerated (line (b)), the result of the acceleration is assigned to the label of its highest ancestor $\overline{n}$ s.t. $\Lambda(\overline{n}) \prec \Lambda(n)$, and the whole subtree of $\overline{n}$ is removed from the tree. Third, the algorithm avoids adding a node n' to the tree if there is another node $\overline{n}$ s.t. $\Lambda(\overline{n}) \succcurlyeq \Lambda(n')$ (line (c)). Finally, the adjunction of n' to the tree (when it happens) triggers the deletion of all the subtrees rooted in some node n'' s.t. $\Lambda(n'') \prec \Lambda(n')$ (line (d)).

Remark that this algorithm is *non-deterministic*, in the sense that no ordering is imposed on the nodes in *to_treat*. Hence, any strategy that fixes the exploration order (which can possibly improve the efficiency of the algorithm) can be chosen.

4. Counter-example to the MCT algorithm

In this section, we introduce a PN on which the MCT algorithm might compute a *strict under-approximation* of the covering set (see Fig. 2). Fig 2(a) is the PN on which we run the MCT algorithm, and Fig. 2(b) through 2(f) are the key points of the execution.

Let us briefly comment on this execution. First remark that place p_5 of the PN in Fig. 2(a) is unbounded, because marking $\{p_3\}$ is reachable from the initial marking $\mathbf{m}_0 = \{p_1\}$ by firing $t_1 t_2$, and the sequence $t_3 t_4$ can be fired an arbitrary number of times from $\{p_3\}$, and strictly increases the markings of p_5. Then, one possible execution of MCT is as follows (markings in the frontier are underlined):

Fig. 2(b) The three successors of $\mathbf{m}_0$ are computed. Then, the branch rooted in $\{p_2\}$ is unfolded, by firing t_2, t_3 and t_4. At that point, two comparable markings $\{p_3\}$ and $\{p_3, p_5\}$ are met with and an acceleration occurs (line (b) of Algorithm 2) . The result is $\{p_3, \omega p_5\}$, which is put into *to_treat*.

Fig. 2(c) The subtree rooted in $\{p_6\}$ is unfolded. After the firing of $t_6 t_4$, one obtains $\{p_3, 3p_5\}$, which is smaller than $\{p_3, \omega p_5\}$. Hence, $\{p_3, 3p_5\}$ is not put into *to_treat* and the branch is stopped (line (c)).

Fig. 2(d) The subtree rooted in $\{p_7\}$ is developed. The unique successor $\{p_2, p_5\}$ of $\{p_7\}$ is larger than $\{p_2\}$. Hence, the subtree rooted in $\{p_2\}$ (including $\{p_3, \omega p_5\}$, still in the frontier) is removed (line (d)).

Fig. 2(e) and 2(f) The tree (actually a single branch) rooted in $\{p_2, p_5\}$ (only node in the frontier) is further developed through the firing of t_2 and t_3.

[b]Thus, $\overline{n}$ is not necessarily an ancestor of n.

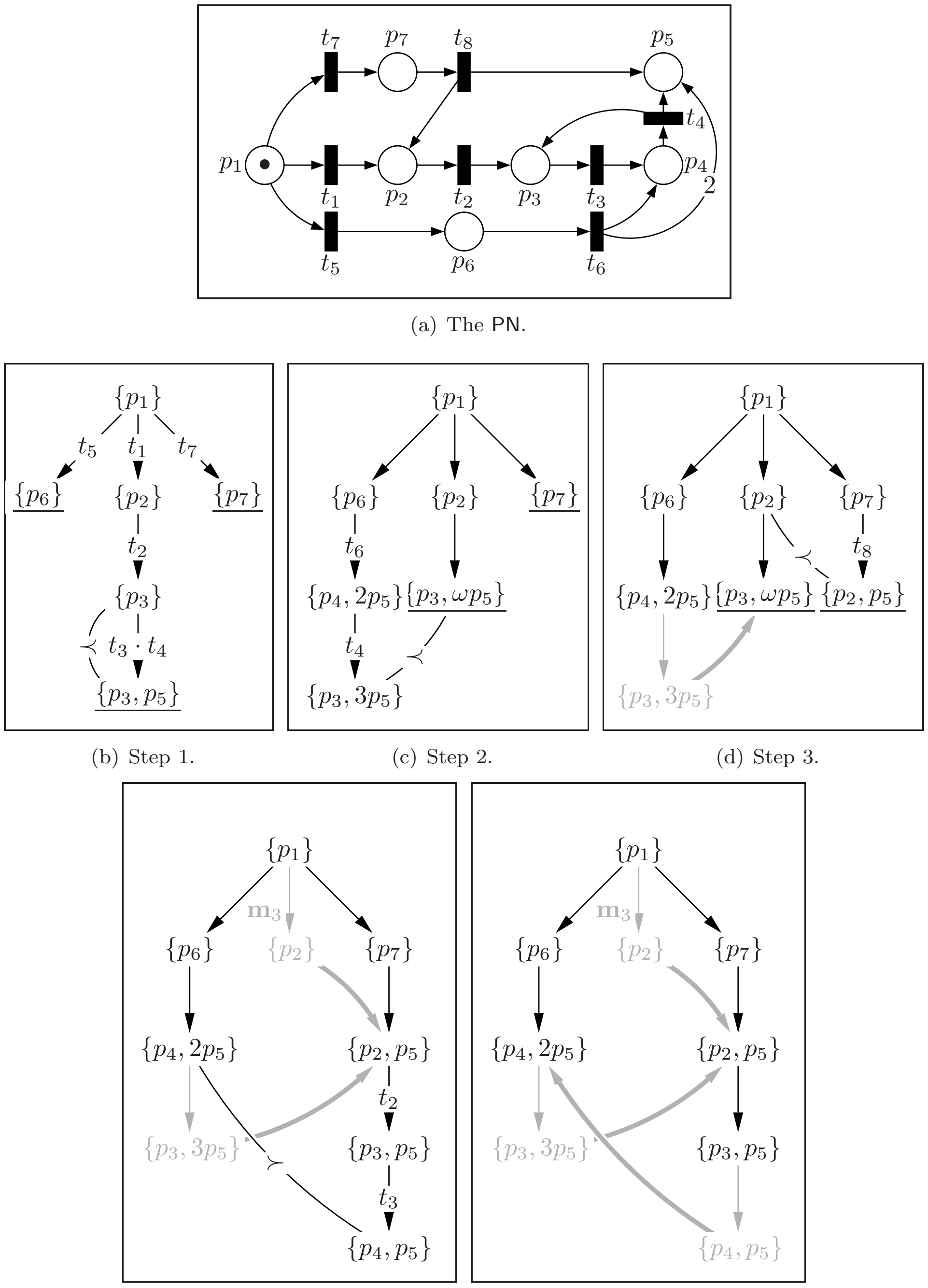
```
Data: A PN $\mathcal{N} = \langle P, T \rangle$ and an initial marking $\mathbf{m}_0$
Result: The minimal coverability set of $\mathcal{N}$.
MCT $(\mathcal{N}, \mathbf{m}_0)$ begin
    $\mathcal{T} \leftarrow \langle N, B, n_0, \Lambda \rangle$ where $N = \{n_0\}$, $B = \emptyset$ and $\Lambda(n_0) = \mathbf{m}_0$ ;
    $to\_treat \leftarrow \{n_0\}$ ;
    while $to\_treat \neq \emptyset$ do
        Choose and remove a node $n$ from $to\_treat$;
(a)     if $\nexists \overline{n} \in N$ s.t. $\Lambda(\overline{n}) = \Lambda(n)$ then
            foreach $\mathbf{m} \in \mathsf{Post}(\Lambda(n))$ do
(b)             if $\exists \overline{n} : B^*(\overline{n}, n)$ and $\Lambda(\overline{n}) \prec \mathbf{m}$ then
                    Let $\overline{n}$ be the highest node s.t. $B^*(\overline{n}, n) \wedge \Lambda(\overline{n}) \prec \mathbf{m}$ ;
                    Let $\mathcal{A}$ be $\{n' \in N \mid B^*(n', n)\}$ ;
                    $\Lambda(\overline{n}) \leftarrow \mathsf{Accel}(\mathcal{A}, \mathbf{m})$ ;
                    $to\_treat \leftarrow \big(to\_treat \setminus \{n' \mid B^*(\overline{n}, n')\}\big) \cup \{\overline{n}\}$ ;
                    removeSubtreeExceptRoot$(\overline{n}, \mathcal{T})$ ;
                    break ;
(c)             else if $\nexists \overline{n} \in N$ s.t. $\mathbf{m} \prec \Lambda(\overline{n})$ then
                    Let $n'$ be a new node s.t. $\Lambda(n') = \mathbf{m}$ ;
                    $N \leftarrow N \cup \{n'\}$ ; $B \leftarrow B \cup (n, n')$; $to\_treat \leftarrow to\_treat \cup \{n'\}$;
(d)         while $\exists n_1, n_2 \in N : \Lambda(n_1) \prec \Lambda(n_2)$ do
                $to\_treat \leftarrow to\_treat \setminus \{n \mid B^*(n_1, n)\}$ ;
                removeSubtree$(n_1, \mathcal{T})$ ;
    return($\{\Lambda(n) \mid n \in N\}$) ;
end
```

Algorithm 2: The MCT algorithm.

The resulting node $\{p_4, p_5\}$ is strictly smaller than $\{p_4, 2p_5\}$. Hence, that branch is stopped too (line (c)), and the frontier becomes empty. The final result of the algorithm is shown in Fig. 2(f). It is not difficult to see that the set of labels of this tree does not form a coverability set, because it contains no marking $\mathbf{m}$ s.t. $\mathbf{m}(p_5) = \omega$.

Comment on the counter-example This counter-example allows us to identify a flaw in the logic of the MCT algorithm. The algorithm stops the development of a node n or removes the subtree rooted in n because it has found another node n' s.t. $\Lambda(n')$ is larger than $\Lambda(n)$. In that case, we say that n' is *a proof for* n. The intuition behind this notion is that, by monotonicity, all the successors of n should be covered by some successor of n'. Thus, when n' is a proof for n, the algorithm makes implicitly the hypothesis that either all the successors of n' will be fully developed, or that they will be covered by some other nodes of the tree.

In our counter-example, that reasoning fails because *cycles* appear in 'proofs'.


t_7 p_7 t_8 p_5 t_4 p_1 p_4 2 t_1 p_2 t_2 p_3 t_3 t_5 p_6 t_6
$\{p_1\}$ $\{p_6\}$ $\{p_2\}$ $\{p_7\}$ $\{p_3\}$ $t_3 \cdot t_4$ $\{p_3, p_5\}$
$\{p_4, 2p_5\}$ $\{p_3, \omega p_5\}$ $\{p_3, 3p_5\}$ $\{p_2, p_5\}$ $\{p_4, p_5\}$ $\mathbf{m}_3$ $\prec$

(a) The PN.

(b) Step 1. (c) Step 2. (d) Step 3.

(e) Step 4. (f) The result.

Fig. 2. A counter-example to the MCT algorithm. Underlined markings are in the frontier. A gray arrow from n to n' means that n' is a 'proof' for n.

In Fig. 2, we have drawn a thick gray arrow from n to n' when n' is a proof for n. In Fig. 2(d), the node labeled by $\{p_3, \omega p_5\}$, which is a proof for $\{p_3, 3p_5\}$ is deleted, because of $\{p_2, p_5\}$. Hence, $\{p_2, p_5\}$, becomes the proof of $\{p_3, 3p_5\}$ (see Fig. 2(e)). The cycle clearly appears in Fig 2(f): all the successors of $\{p_4, 2p_5\}$ will be eventually covered under the assumption that all the successors of $\{p_2, p_5\}$ are covered. However, this happens only if all the successors of $\{p_4, 2p_5\}$ are eventually covered.

Implementation of the MCT in the Pep tool Apparently, the flaw in the MCT algorithm has independently been discovered by the team of Prof. Peter Starke. They have implemented in INA (a component of the toolkit Pep [9]) a variation of the MCT which is supposed to correct the aforementioned bug. To the best of our knowledge, this implementation (and the discovery of the bug) has been documented only in a master's thesis in German [11]. Unfortunately, their version of the MCT contains a flaw too, because it offers no guarantee of termination [13, 5], although the Master's thesis [11] contains a proof of termination. We have presented, in a technical report [4] a counter-example to termination. Thus, from our point of view, fixing the bug of the MCT algorithm seems to be a difficult task.

5. The covering sequence

Instead of trying to fix the bug in the MCT algorithm (or the flawed version implemented in INA), we propose a different solution based on novel ideas. To introduce our new solution, let us get back to the basics. Remember that we want to compute an effective representation of $\downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$. It is easy to show that this set is the limit of the following infinite sequence of sets: $X_0 = \downarrow^{\preccurlyeq}(\{\mathbf{m}_0\})$, and for $i \geq 1$, $X_i = \downarrow^{\preccurlyeq}(X_{i-1} \cup \mathsf{Post}(X_{i-1}))$. Note that by strict monotonicity of Petri nets, we can try to compute a symbolic representation of $\downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$ by considering the following sequence that handles maximal elements only: $Y_0 = \mathsf{Max}^{\preccurlyeq}(\{\mathbf{m}_0\})$, and $Y_i = \mathsf{Max}^{\preccurlyeq}(Y_{i-1} \cup \mathsf{Post}(Y_{i-1}))$ for any $i \geq 1$. Unfortunately, the naive procedure that consists in iteratively computing each Y_i up to stabilization is not an effective way to compute the minimal coverability set as there is no guarantee that this sequence eventually stabilizes. This is not surprising since the underlying transition system of a PN is, in general, infinite, and may contain infinitely increasing sequences of reachable markings. This is also the reason why an acceleration function has been introduced in the KM algorithm, in order to cope with such infinite sequences.

Remark, however, that such a complex function (which considers *all* the ancestors of a given node) is not needed in general, since *pairs* of markings are sufficient to compute an acceleration. This motivates our new solution which constructs a sequence of sets of pairs of markings on which we systematically apply a variant of the Post operator and a variant of the acceleration function. By defining an adequate order $\sqsubseteq$ on pairs of markings, we show that we can concentrate on maximal elements for $\sqsubseteq$. Let us define these notions more formally.

Preliminaries We introduce several operators that work directly on pairs of ω-markings. Given a set R of pairs of ω-markings, we let $\mathsf{Flatten}(R) = \{\mathbf{m} \mid \exists \mathbf{m}' : (\mathbf{m}', \mathbf{m}) \in R\}$. We use the $\overline{\mathsf{Post}}$ function to define the notion of *successor of a pair* of ω-markings $(\mathbf{m}_1, \mathbf{m}_2)$: $\overline{\mathsf{Post}}((\mathbf{m}_1, \mathbf{m}_2)) = \{(\mathbf{m}_1, \mathbf{m}'), (\mathbf{m}_2, \mathbf{m}') \mid \mathbf{m}' \in \mathsf{Post}(\mathbf{m}_2)\}$.

Our new solution relies on a weaker acceleration function than that of Karp&Miller (because its first argument is restricted to a single marking instead of a set of markings). Given two ω-markings $\mathbf{m}_1$ and $\mathbf{m}_2$ s.t. $\mathbf{m}_1 \preccurlyeq \mathbf{m}_2$, we let $\mathsf{AccelPair}(\mathbf{m}_1, \mathbf{m}_2) = \mathbf{m}_\omega$ s.t. for any place p, $\mathbf{m}_\omega(p) = \mathbf{m}_1(p)$ if $\mathbf{m}_1(p) = \mathbf{m}_2(p)$; $\mathbf{m}_\omega(p) = \omega$ otherwise. Similarly to $\overline{\mathsf{Post}}$, we use $\overline{\mathsf{Accel}}$ to define the notion of *acceleration of a pair* of ω-markings $(\mathbf{m}_1, \mathbf{m}_2)$: $\overline{\mathsf{Accel}}((\mathbf{m}_1, \mathbf{m}_2)) = \{(\mathbf{m}_2, \mathsf{AccelPair}(\mathbf{m}_1, \mathbf{m}_2))\}$ if $\mathbf{m}_1 \prec \mathbf{m}_2$; and $\overline{\mathsf{Accel}}((\mathbf{m}_1, \mathbf{m}_2))$ is undefined otherwise.

We extend the $\overline{\mathsf{Post}}$ and $\overline{\mathsf{Accel}}$ functions to sets R of pairs in the following way: $\overline{\mathsf{Post}}(R) = \cup_{(\mathbf{m}_1,\mathbf{m}_2)\in R} \overline{\mathsf{Post}}((\mathbf{m}_1, \mathbf{m}_2))$ and $\overline{\mathsf{Accel}}(R) = \cup_{(\mathbf{m}_1,\mathbf{m}_2)\in R, \mathbf{m}_1 \prec \mathbf{m}_2} \{\overline{\mathsf{Accel}}(\mathbf{m}_1, \mathbf{m}_2)\}$.

Now that we have defined function to handle the semantics of the PN in terms of pairs, we define the ordering $\sqsubseteq$ that will allow us to reduce the size of the sets our algorithm manipulate. Let $\mathbf{m}_1$ and $\mathbf{m}_2$ be two ω-markings. Then, $\mathbf{m}_1 \ominus \mathbf{m}_2$ is a function $P \mapsto \mathbb{Z} \cup \{-\omega, \omega\}$ s.t. for any place p: $(\mathbf{m}_1 \ominus \mathbf{m}_2)(p)$ is equal to ω if $\mathbf{m}_1(p) = \omega$; $-\omega$ if $\mathbf{m}_2(p) = \omega$ and $\mathbf{m}_1(p) \neq \omega$; $\mathbf{m}_1(p) - \mathbf{m}_2(p)$ otherwise. Then, given two pairs of ω-markings $(\mathbf{m}_1, \mathbf{m}_2)$ and $(\mathbf{m}'_1, \mathbf{m}'_2)$, we have $(\mathbf{m}_1, \mathbf{m}_2) \sqsubseteq (\mathbf{m}'_1, \mathbf{m}'_2)$ iff $\mathbf{m}_1 \preccurlyeq \mathbf{m}'_1$, $\mathbf{m}_2 \preccurlyeq \mathbf{m}'_2$ and for any place p: $(\mathbf{m}_2 \ominus \mathbf{m}_1)(p) \leq (\mathbf{m}'_2 \ominus \mathbf{m}'_1)(p)$.

Example 11. *Let us assume a PN with two places. Then:* $(\langle 0,1\rangle, \langle 0,2\rangle) \sqsubseteq (\langle 1,1\rangle, \langle 2,5\rangle)$ *and* $(\langle 0,1\rangle, \langle 0,2\rangle) \sqsubseteq (\langle 1,\omega\rangle, \langle 2,\omega\rangle)$. *However,* $(\langle 0,1\rangle, \langle 0,2\rangle) \not\sqsubseteq (\langle 1,7\rangle, \langle 2,7\rangle)$ *although* $\langle 0,1\rangle \preccurlyeq \langle 1,7\rangle$ *and* $\langle 0,2\rangle \preccurlyeq \langle 2,7\rangle$. *Indeed,* $\langle 0,2\rangle \ominus \langle 0,1\rangle = \langle 0,1\rangle$, $\langle 2,7\rangle \ominus \langle 1,7\rangle = \langle 1,0\rangle$ *and* $\langle 0,1\rangle \not\preccurlyeq \langle 1,0\rangle$.

For any $(\mathbf{m}_1, \mathbf{m}_2)$, we let $\downarrow^{\sqsubseteq}((\mathbf{m}_1, \mathbf{m}_2)) = \{(\mathbf{m}'_1, \mathbf{m}'_2) \mid (\mathbf{m}'_1, \mathbf{m}'_2) \sqsubseteq (\mathbf{m}_1, \mathbf{m}_2)\}$. We extend this to sets of pairs R as follows: $\downarrow^{\sqsubseteq}(R) = \cup_{(\mathbf{m}_1,\mathbf{m}_2)\in R} \downarrow^{\sqsubseteq}((\mathbf{m}_1, \mathbf{m}_2))$. Given a set R of pairs of markings, we let $\mathsf{Max}^{\sqsubseteq}(R) = \{(\mathbf{m}_1, \mathbf{m}_2) \in R \mid \nexists (\mathbf{m}'_1, \mathbf{m}'_2) \in R : (\mathbf{m}_1, \mathbf{m}_2) \neq (\mathbf{m}'_1, \mathbf{m}'_2) \wedge (\mathbf{m}_1, \mathbf{m}_2) \sqsubseteq (\mathbf{m}'_1, \mathbf{m}'_2)\}$

Let us provide some intuition on the ordering $\sqsubseteq$. The intended use of this ordering is to maintain sets of pairs that are *minimal* wrt to $\sqsubseteq$ but still contain enough information to ensure the correctness and completeness of the computations. Let $(\mathbf{m}_1, \mathbf{m}_2)$ and $(\mathbf{m}'_1, \mathbf{m}'_2)$ be two pairs of markings s.t. $(\mathbf{m}_1, \mathbf{m}_2) \sqsubseteq (\mathbf{m}'_1, \mathbf{m}'_2)$. Then it is easy to observe that:

(1) For any $(\mathbf{m}_3, \mathbf{m}_4) \in \overline{\mathsf{Post}}((\mathbf{m}_1, \mathbf{m}_2))$, there is $(\mathbf{m}'_3, \mathbf{m}'_4) \in \overline{\mathsf{Post}}((\mathbf{m}'_1, \mathbf{m}'_2))$ s.t. $(\mathbf{m}_3, \mathbf{m}_4) \sqsubseteq (\mathbf{m}'_3, \mathbf{m}'_4)$. This is due to the monotonicity property of PN, and the fact that $\mathbf{m}_2 \preccurlyeq \mathbf{m}'_2$

(2) Similarly, for any $(\mathbf{m}_3, \mathbf{m}_4) \in \overline{\mathsf{Accel}}((\mathbf{m}_1, \mathbf{m}_2))$, there is $(\mathbf{m}'_3, \mathbf{m}'_4) \in$

$\overline{\mathsf{Accel}}((\mathbf{m}'_1, \mathbf{m}'_2))$ s.t. $(\mathbf{m}_3, \mathbf{m}_4) \sqsubseteq (\mathbf{m}'_3, \mathbf{m}'_4)$. Remark that in this latter case, the definition of the $\ominus$ operator is crucial, since $\mathbf{m}_3 \preccurlyeq \mathbf{m}'_3$ and $\mathbf{m}_4 \preccurlyeq \mathbf{m}'_4$ alone do not ensure this property.

Properties of the acceleration function Before we can introduce our new algorithm, we discuss two properties of our acceleration function, defined above. The first lemma states that this acceleration is *sound*, in the sense that it will never produce markings that are not covered by some successors of the markings that have produced the acceleration:

Lemma 12. *Let $\mathcal{N}$ be a PN and let $\mathbf{m}_1$ and $\mathbf{m}_2$ be two ω-markings of $\mathcal{N}$ that respect $\mathbf{m}_1 \preccurlyeq \mathbf{m}_2$ and $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_1))$. Then, $\downarrow^{\preccurlyeq}(\mathsf{AccelPair}(\mathbf{m}_1, \mathbf{m}_2)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_2))$.*

Proof. Let P' be the set of places $\{p \mid \mathbf{m}_1(p) < \mathbf{m}_2(p)\}$. Remark that, since $\mathbf{m}_1 \preccurlyeq \mathbf{m}_2$, $\mathbf{m}_1(p) = \mathbf{m}_2(p)$ for any $p \notin P'$. In order to establish the lemma, we first show that, since $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_1))$ (by hypothesis) and since $\mathbf{m}_1 \preccurlyeq \mathbf{m}_2$, there must exist a sequence of transitions σ that is firable from $\mathbf{m}_1$ and allows to increase the number of tokens in the places of P'. That is, we show that there exists σ and a marking $\overline{\mathbf{m}}$ s.t. (i) $\mathbf{m}_1 \xrightarrow{\sigma} \overline{\mathbf{m}}$, (ii) $\mathbf{m}_1 \preccurlyeq \overline{\mathbf{m}}$ and (iii) for any place $p \in P'$, $\overline{\mathbf{m}}(p) > \mathbf{m}_1(p)$. Indeed, let $\mathbf{m}'$ be defined as follows:

$$\forall p \in P : \mathbf{m}'(p) = \begin{cases} 0 & \text{if } p \notin P' \text{ and } \mathbf{m}_1(p) = \omega \\ \mathbf{m}_1(p) & \text{if } p \notin P' \text{ and } \mathbf{m}_1(p) \neq \omega \\ \mathbf{m}_1(p) + 1 & \text{if } p \in P' \end{cases}$$

By definition, $\mathbf{m}' \in \downarrow^{\preccurlyeq}(\mathbf{m}_2)$ but $\mathbf{m}' \notin \downarrow^{\preccurlyeq}(\mathbf{m}_1)$ (remark in particular that $p \in P'$ implies that $\mathbf{m}_1(p) \neq \omega$ and $\mathbf{m}_2(p) \geq \mathbf{m}_1(p) + 1$). Since $\mathbf{m}' \in \downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_1))$, there exists a marking $\overline{\mathbf{m}}$ and a sequence of transitions σ s.t. $\mathbf{m}_1 \xrightarrow{\sigma} \overline{\mathbf{m}}$ and $\mathbf{m}' \preccurlyeq \overline{\mathbf{m}}$. Hence, $\mathbf{m}'(p) \leq \overline{\mathbf{m}}(p)$ for every place p. Let us show that $\overline{\mathbf{m}} \succcurlyeq \mathbf{m}_1$ and that, for any $p \in P'$: $\overline{\mathbf{m}}'(p) > \mathbf{m}_1(p)$. We consider three cases. (i) when $\mathbf{m}_1(p) = \omega$, we have necessarily $\overline{\mathbf{m}}(p) = \omega$. Hence, $\mathbf{m}_1(p) \leq \overline{\mathbf{m}}(p)$ for every place p s.t. $\mathbf{m}_1(p) = \omega$. (ii) when $\mathbf{m}_1(p) \neq \omega$ and $p \notin P'$, we have $\mathbf{m}'(p) = \mathbf{m}_1(p)$, by definition of $\mathbf{m}'$. Hence $\mathbf{m}_1(p) \leq \overline{\mathbf{m}}(p)$ for every such p. (iii) when $p \in P'$ (hence $\mathbf{m}_1(p) \neq \omega$), we have $\mathbf{m}_1(p) < \mathbf{m}'(p)$, by definition of $\mathbf{m}'$ again. Hence $\mathbf{m}_1(p) < \overline{\mathbf{m}}(p)$ for every such p. We conclude that $\mathbf{m}_1 \preccurlyeq \overline{\mathbf{m}}$ and that $\mathbf{m}_1(p) < \overline{\mathbf{m}}(p)$ for every $p \in P'$.

The next step of the proof consists in discussing the effect of iterating σ an arbitrarily number of times from $\mathbf{m}_1$. The results of this discussion will be used in the last part of the proof. Let $\overline{\mathbf{m}}_i$ $(i \geq 1)$ be the marking s.t. $\mathbf{m}_1 \xrightarrow{\sigma^i} \overline{\mathbf{m}}_i$, i.e. the marking obtained after having fired i times σ from $\mathbf{m}_1$. Since PN transitions have constant effect, we have:

$$\forall i \geq 1 : \forall p : \overline{\mathbf{m}}_i(p) = \mathbf{m}_1(p) + i \cdot (\overline{\mathbf{m}}(p) - \mathbf{m}_1(p)) \qquad (1)$$

Remark that, for any $i \geq 1 : \overline{\mathbf{m}}_i \in \mathsf{Post}^*(\mathbf{m}_1)$, and that, by monotonicity, $\forall i \geq 1 : \overline{\mathbf{m}}_i \preccurlyeq \overline{\mathbf{m}}_{i+1}$.

In the case where $p \in P'$, the value $\overline{\mathbf{m}}(p) - \mathbf{m}_1(p)$ is > 0. Hence, by (1), we have:

$$\forall p \in P' : \forall n \in \mathbb{N} : \exists k : \overline{\mathbf{m}}_k(p) > n \tag{2}$$

On the other hand, by definition of the acceleration function, and since $\overline{\mathbf{m}}_i \succcurlyeq \mathbf{m}_1$ for any $i \geq 1$:

$$\forall p \notin P' : \forall i \geq 1 : \overline{\mathbf{m}}_i(p) \geq \mathbf{m}_1(p) = \mathbf{m}_2(p) = \mathsf{AccelPair}\,(\mathbf{m}_1, \mathbf{m}_2)(p) \tag{3}$$

We conclude the proof by considering $\mathbf{m}$ in $\downarrow^{\preccurlyeq}(\mathsf{AccelPair}\,(\mathbf{m}_1, \mathbf{m}_2))$ and showing that $\mathbf{m} \in \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$. Clearly, $\mathbf{m} \preccurlyeq \mathsf{AccelPair}\,(\mathbf{m}_1, \mathbf{m}_2)$ and for any place p: $\mathbf{m}(p) \neq \omega$. Hence, by (3), for any $p \notin P'$, for any $i \geq 1$, $\mathbf{m}(p) \leq \overline{\mathbf{m}}_i(p)$. Moreover, by (2), there exists, for any $p \in P'$, a value $k(p)$ s.t. $\overline{\mathbf{m}}_{k(p)}(p) > \mathbf{m}(p)$. Since the sequence $\overline{\mathbf{m}}_1, \overline{\mathbf{m}}_2, \ldots$ is $\preccurlyeq$-increasing, the marking $\overline{\mathbf{m}}_k$, with $k = \max\{k(p) \mid p \in P'\}$ is s.t. for any $p \in P' : \overline{\mathbf{m}}_k(p) \geq \mathbf{m}(p)$. We conclude that there exists $k \geq 1$ with $\overline{\mathbf{m}}_k \succcurlyeq \mathbf{m}$. Since $\overline{\mathbf{m}}_k \in \mathsf{Post}^*(\mathbf{m}_1)$, and since $\mathbf{m}_2 \succcurlyeq \mathbf{m}_1$, there exists, by monotonicity, a marking $\mathbf{m}'$ s.t. $\mathbf{m}' \in \mathsf{Post}^*(\mathbf{m}_2)$ and $\mathbf{m}' \succcurlyeq \overline{\mathbf{m}}_k \succcurlyeq \mathbf{m}$. Since this is true for any $\mathbf{m} \in \downarrow^{\preccurlyeq}(\mathsf{AccelPair}\,(\mathbf{m}_1, \mathbf{m}_2))$, we conclude that: $\downarrow^{\preccurlyeq}(\mathsf{AccelPair}\,(\mathbf{m}_1, \mathbf{m}_2)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_2))$. □

The second property draws a link between AccelPair and the Karp&Miller acceleration. It shows that, although AccelPair is weaker, it can produce the same results than the Karp&Miller acceleration, when properly applied. Given a labeled tree $\mathcal{T} = \langle N, B, root, \Lambda \rangle$, we let for any $n \in N$, $\mathsf{Anc}\,(\mathcal{T}, n) = \{n' \in N \mid B^*(n', n)\}$ (that is, $\mathsf{Anc}\,(\mathcal{T}, n)$ is the set of 'ancestors' of n in $\mathcal{T}$, n included). Then the lemma is as follows:

Lemma 13. *Let $\mathcal{N} = \langle P, T \rangle$ be a Petri net with initial marking $\mathbf{m}_0$ and let $\mathcal{T} = \langle N, B, root, \Lambda \rangle$ be its Karp&Miller tree. Let $n \neq root$ be a node of $\mathcal{T}$. Let $\mathbf{m}'$ be s.t. $\mathsf{M}(n) \xrightarrow{\varsigma(n)} \mathbf{m}'$. Then, $\Lambda(n) \preccurlyeq \mathsf{AccelPair}\,(\mathsf{M}(n), \mathbf{m}')$.*

Proof. Let $P_a = \{p \in P \mid \Lambda(n)(p) = \omega \wedge \mathsf{M}(n)(p) \neq \omega\}$. By construction, for any place p: $\Lambda(n)(p) = \omega$ if $p \in P_a$; $\Lambda(n)(p) = \mathsf{M}(n)$ otherwise. Moreover, by definition of AccelPair, for any place p: $\mathsf{AccelPair}\,(\mathsf{M}(n), \mathbf{m}')(p) = \omega$ if $\varsigma(n)(p) > 0$; and $\mathsf{AccelPair}\,(\mathsf{M}(n), \mathbf{m}')(p) = \mathsf{M}(n)(p)$ otherwise. By definition of $\varsigma(n)$, $p \in P_a$ implies $\varsigma(n)(p) > 0$. Hence the Lemma. □

Definition of the sequence We are now ready to introduce the *covering sequence* that allows to compute efficiently the MCS of a PN. The definition of the sequence is rather generic: it assumes the existence of an *oracle* which is defined as a procedure that produces pairs of markings, respecting simple properties (see Definition 14). In the sequel, we will see that this notion of oracle can be used to formalize different kinds of optimizations which potentially allow the sequence to converge faster to a coverability set, thereby leading to an efficient algorithm to compute this set. Since we provide, in the present section, a proof of correctness of the sequence for *any oracle* (according to Definition 14), all the optimizations that one could formalize thanks to the notion of oracle will remain correct. In particular, we will show, in the next section, that the oracle can be implemented by a recursive call to the procedure that computes the covering sequence, with a modified initial marking.

The formal definition of an oracle is a follows:

Definition 14. *Given a Petri net* $\mathcal{N}$ *and an initial* ω*-marking* $\mathbf{m}_0$*, an* oracle *is a function* $\mathsf{Oracle} : \mathbb{N} \mapsto (\mathbb{N} \cup \{\omega\})^{|P|} \times (\mathbb{N} \cup \{\omega\})^{|P|}$ *that returns, for any* $i \geq 0$*, a set of pairs of* ω*-markings that satisfies the two following conditions:*

$$\downarrow^{\preccurlyeq}(\mathsf{Post}\,(\mathsf{Flatten}\,(\mathsf{Oracle}\,(i)))) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(\mathsf{Oracle}\,(i))) \tag{4}$$

$$\downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(\mathsf{Oracle}\,(i))) \subseteq \mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0)\,. \tag{5}$$

Thus, an oracle is a function that, for any $i \geq 0$ returns a set $\mathsf{Oracle}\,(i)$ of pairs of ω-markings. Moreover, each set $\mathsf{Flatten}\,(\mathsf{Oracle}\,(i))$ is an *under-approximation* of the covering set, by (5). It is thus easy to understand that such an information (no matter how it is computed) could be used to speed up a procedure that computes the coverability set of a PN, because it is a 'part of the solution'. Finally, each set $\mathsf{Flatten}\,(\mathsf{Oracle}\,(i))$ is also closed under Post, by (4). The usefulness of this requirement will become clear in the sequel. Remark, in particular, that the function Oracle s.t. $\mathsf{Oracle}\,(i) = \emptyset$ for any i, $\mathcal{N}$ and $\mathbf{m}_0$ is a trivial oracle.

We can now define the notion of *covering sequence*:

Definition 15. *Let* $\mathcal{N} = \langle P, T\rangle$ *be a* PN*,* $\mathbf{m}_0$ *be an initial marking, and* Oracle *be an oracle. Then, the covering sequence of* $\mathcal{N}$*, noted* $\mathsf{CovSeq}\,(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle})$ *is the infinite sequence* $(V_i, F_i, O_i)_{i\geq 0}$*, defined as follows:*

- $V_0 = \emptyset$*,* $O_0 = \emptyset$ *and* $F_0 = \{(\mathbf{m}_0, \mathbf{m}_0)\}$*;*
- *For any* $i \geq 1$*:* $O_i = \mathsf{Max}^{\sqsubseteq}\,(O_{i-1} \cup \mathsf{Oracle}\,(i))$*;*
- *For any* $i \geq 1$*:* $V_i = \mathsf{Max}^{\sqsubseteq}\,(V_{i-1} \cup F_{i-1}) \setminus \downarrow^{\sqsubseteq}(O_i)$*;*
- *For any* $i \geq 1$*:* $F_i = \mathsf{Max}^{\sqsubseteq}\,\big(\overline{\mathsf{Post}}\,(F_{i-1}) \cup \overline{\mathsf{Accel}}\,(F_{i-1})\big) \setminus \downarrow^{\sqsubseteq}(V_i \cup O_i)$*.*

A covering sequence is thus a sequence of triples (V_i, F_i, O_i), for any $i \geq 0$, where all the V_i's, F_i's and O_i's are sets of pairs of markings. At step i, O_i is a set of pairs that represents all the information provided by the *oracle* during steps $0 \ldots i$. The set V_i represents the *visited* pairs, i.e. the pairs that are present in some V_j or some F_j, for $0 \leq j \leq i-1$. However, in order to avoid redundancy, we do not store in V_i the pairs that are smaller (wrt to $\sqsubseteq$) than a pair previously provided by the oracle (and is thus in O_i). Finally, F_i is the *frontier*, i.e., the set of pairs which (i) are obtained by applying either the $\overline{\mathsf{Post}}$ or the $\overline{\mathsf{Accel}}$ operator to some pair in F_{i-1} and (ii) have not been visited or produced by the oracle before. Remark that, in all these sets, only maximal (wrt to $\sqsubseteq$) pairs are kept. This allows to keep the size of these sets smaller.

Example 16. *As an example, we apply the covering sequence to the* PN $\mathcal{N}$ *with its initial marking* $\mathbf{m}_0$ *in Fig. 1. In this example, we assume the oracle to be trivial, i.e.,* Oracle $(i) = \emptyset$ *for any* $i \geq 0$*. Thus,* $O_i = \emptyset$ *for any* $i \geq 0$*. We provide hereunder the values of the sets* V_i *and* F_i *for the first few values of* i*. The pairs that are covered by some other pair, and thus not kept by the* $\mathsf{Max}^{\sqsubseteq}$ *function have been struck through. This allows to see how the ordering* $\sqsubseteq$ *is effective in reducing the size of the computed sets, even on this toy example.*

i	V_i	F_i
0	$\emptyset$	$(\langle 1,0,0\rangle, \langle 1,0,0\rangle)$
1	$(\langle 1,0,0\rangle, \langle 1,0,0\rangle)$	$(\langle 1,0,0\rangle, \langle 0,2,0\rangle)$
2	$(\langle 1,0,0\rangle, \langle 1,0,0\rangle)$ $(\langle 1,0,0\rangle, \langle 0,2,0\rangle)$	$(\langle 1,0,0\rangle, \langle 1,0,1\rangle)$ $(\langle 0,2,0\rangle, \langle 1,0,1\rangle)$
3	~~$(\langle 1,0,0\rangle, \langle 1,0,0\rangle)$~~ $(\langle 1,0,0\rangle, \langle 0,2,0\rangle)$ $(\langle 1,0,0\rangle, \langle 1,0,1\rangle)$ $(\langle 0,2,0\rangle, \langle 1,0,1\rangle)$	$(\langle 1,0,0\rangle, \langle 1,0,\omega\rangle)$ ~~$(\langle 1,0,0\rangle, \langle 0,2,1\rangle)$~~ $(\langle 1,0,1\rangle, \langle 0,2,1\rangle)$ $(\langle 0,2,0\rangle, \langle 0,2,1\rangle)$
4	~~$(\langle 1,0,0\rangle, \langle 0,2,0\rangle)$~~ ~~$(\langle 1,0,0\rangle, \langle 1,0,1\rangle)$~~ $(\langle 0,2,0\rangle, \langle 1,0,1\rangle)$ $(\langle 1,0,0\rangle, \langle 1,0,\omega\rangle)$ $(\langle 1,0,1\rangle, \langle 0,2,1\rangle)$ $(\langle 0,2,0\rangle, \langle 0,2,1\rangle)$	$(\langle 1,0,\omega\rangle, \langle 1,0,\omega\rangle)$ $(\langle 0,2,1\rangle, \langle 0,2,\omega\rangle)$ ~~$(\langle 1,0,0\rangle, \langle 0,2,\omega\rangle)$~~ $(\langle 1,0,\omega\rangle, \langle 0,2,\omega\rangle)$ ~~$(\langle 1,0,1\rangle, \langle 1,0,2\rangle)$~~ $(\langle 0,2,1\rangle, \langle 1,0,2\rangle)$ ~~$(\langle 0,2,0\rangle, \langle 1,0,2\rangle)$~~

The next steps of the sequence have been omitted and are left as an exercise to the reader. Remark that the set Flatten (F_4) *already contains the two* ω*-markings of the coverability set.*

The rest of this sections consists in proving that for any PN $\mathcal{N}$, any initial marking $\mathbf{m}_0$, and any oracle Oracle, s.t. CovSeq $(\mathcal{N}, \mathbf{m}_0, \text{Oracle}) = (V_i, F_i, O_i)_{i\geq 0}$, there exists k s.t. Flatten $(V_k \cup O_k)$ is a coverability set of $\mathcal{N}$ (from which the minimal coverability set can easily be extracted). The style of presentation adopted in this work might seem rather technical to the reader. Yet, it is our belief that, considering the history of the problem of finding an efficient algorithm to compute the MCS of a PN (as discussed in Section 4), the uttermost prudence is required when presenting a new solution. We have thus striven to provide a most detailed proof of our new solution. In order to alleviate the burden of reading these technical proofs, we first provide some intuition on why the covering sequence is a correct solution to compute the minimal coverability set.

Consider a PN $\mathcal{N}$ and an initial marking $\mathbf{m}_0$. We proceed step by step, starting from a stripped down version of the covering sequence and providing intuitive explanations of the different constructions at work in the sequence:

(1) First, consider the sequence $(V_i, F_i)_{i\geq 0}$, where: $F_0 = \{(\mathbf{m}_0, \mathbf{m}_0)\}$, $V_0 = \emptyset$ and, for any $i \geq 1$: $V_i = V_{i-1} \cup F_{i-1}$ and $F_i = \overline{\mathsf{Post}}\,(F_{i-1}) \setminus V_i$. Roughly speaking, such a sequence consists in computing, step by step, all the successors of the initial marking, in a *breadth-first fashion.* Clearly, since $\mathsf{Post}^*\,(\mathbf{m}_0)$ is possibly infinite, there is no guarantee that $\downarrow^{\preccurlyeq}(V_k) = \downarrow^{\preccurlyeq}(V_{k-1}) = \mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0)$ for some k.

(2) As a second step, let us consider the sequence $(V_i, F_i)_{i\geq 0}$ defined as the previous one except that now $F_i = (\overline{\mathsf{Post}}\,(F_{i-1}) \cup \overline{\mathsf{Accel}}\,(F_{i-1})) \setminus V_i$ for any $i \geq 1$. The acceleration function has been discussed at the beginning of the current section, and, by Lemma 12 and 13, we know that it is sound and complete. More precisely, by using $\overline{\mathsf{Accel}}$ together with $\overline{\mathsf{Post}}$, we are guaranteed that every ω-marking that appears in some K&M tree of $\mathcal{N}$ will also appear in some set V_i. Since any K&M tree is finite, we are now sure that there exists $k \geq 0$ s.t. $\downarrow^{\preccurlyeq}(V_k) = \downarrow^{\preccurlyeq}(V_{k-1}) = \mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0)$. This simple procedure (which is clearly a simplification of the covering sequence) thus computes a coverability set of the PN.

(3) Then, let us improve on this procedure by exploiting the $\sqsubseteq$ ordering. That is, we let $V_i = \mathsf{Max}^{\sqsubseteq}\,(V_{i-1} \cup F_{i-1})$ for any $i \geq 1$ and $F_i = \mathsf{Max}^{\sqsubseteq}\left(\overline{\mathsf{Post}}\,(F_{i-1}) \cup \overline{\mathsf{Accel}}\,(F_{i-1})\right) \setminus \downarrow^{\sqsubseteq}(V_i)$. Remark that we also remove from the frontier pairs that are covered by some pair of V_i. It is easy to see that this optimization is correct as far as the computation of a coverability set is concerned. Let $(\mathbf{m}_1, \mathbf{m}_2)$ and $(\mathbf{m}'_1, \mathbf{m}'_2)$ be two pairs of markings s.t. $(\mathbf{m}_1, \mathbf{m}_2) \sqsubseteq (\mathbf{m}'_1, \mathbf{m}'_2)$. By definition of $\sqsubseteq$ (see discussion above), we know that, for any $(\mathbf{m}_3, \mathbf{m}_4) \in \overline{\mathsf{Post}}\,((\mathbf{m}_1, \mathbf{m}_2)) \cup \overline{\mathsf{Accel}}\,((\mathbf{m}_1, \mathbf{m}_2))$, there is $(\mathbf{m}'_3, \mathbf{m}'_4) \in \overline{\mathsf{Post}}\,((\mathbf{m}'_1, \mathbf{m}'_2)) \cup \overline{\mathsf{Accel}}\,((\mathbf{m}'_1, \mathbf{m}'_2))$ s.t. $(\mathbf{m}_3, \mathbf{m}_4) \sqsubseteq (\mathbf{m}'_3, \mathbf{m}'_4)$. This implies in particular that $\downarrow^{\preccurlyeq}(\mathbf{m}_3) \subseteq \downarrow^{\preccurlyeq}(\mathbf{m}'_3)$ and $\downarrow^{\preccurlyeq}(\mathbf{m}_4) \subseteq \downarrow^{\preccurlyeq}(\mathbf{m}'_4)$. Thus, we do not lose any information by removing the pair $(\mathbf{m}_1, \mathbf{m}_2)$ from the sets we maintain: all the successors and all the accelerations we could compute from $(\mathbf{m}_1, \mathbf{m}_2)$ will be covered by some successor or acceleration computed from $(\mathbf{m}'_1, \mathbf{m}'_2)$.

(4) Finally, assume that the oracle is not trivial anymore, and returns some non-empty sets of pairs of markings (that respect Definition 14) and consider the full covering sequence as in Definition 15. This full sequence bears three differences with the simplified sequence discussed at the previous point. First, we store the pairs returned by the oracle in the sets O_i. As stated above, we will prove that there exists $k \geq 0$ s.t. $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_k \cup V_k)) = \mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0)$. It makes sense to consider the markings return by the oracle too, since, by (5) in Definition 14, we know that all these markings are covered by the covering set. Second, we remove from V_i the pairs that are covered by some pair in O_i. This optimization simply allows to avoid redundant storage of comparable pairs in the sets V_i and O_i. Third, we remove from F_i the pairs that are covered by some pair from O_i. This optimization is more subtle, since we are supposed to compute the successors of the pairs present in F_i, by means of $\overline{\mathsf{Post}}$ or $\overline{\mathsf{Accel}}$, and careless removal of pairs from F_i might lead to an under-approximation of the coverability set. More precisely, when removing a pair $(\mathbf{m}_1, \mathbf{m}_2)$ from F_i because a larger pair $(\mathbf{m}'_1, \mathbf{m}'_2)$ has been found in O_i, we have to ensure that all the successors of $\mathbf{m}'_1$ and $\mathbf{m}'_2$ will eventually be computed, or covered by some computed marking. By condition (4) of Definition 14, we know that it is the case because the sets

of markings returned by the oracle form a fixed point for the Post operator. Hence, the removals from F_i are safe.

Let us now proceed to a formal proof of correctness of the covering sequence.

Invariants of the covering sequence We start the actual proof of the covering sequence by providing four invariants (Lemma 18 to 21) that will be useful in the sequel. In order to prove these lemmata, we rely on the two following obvious properties, where A and B are any sets of pairs of ω-markings:

$$\downarrow^{\sqsubseteq}(A \cup B) = \downarrow^{\sqsubseteq}(A) \cup \downarrow^{\sqsubseteq}(B) \tag{6}$$

$$\downarrow^{\sqsubseteq}\Big(\mathsf{Max}^{\sqsubseteq}(A)\Big) = \downarrow^{\sqsubseteq}(A) \tag{7}$$

The following lemma will be useful too when establishing the invariants:

Lemma 17. *Let $\mathcal{N}$ be a PN, A and B be two sets of ω-markings of $\mathcal{N}$. Then, $\downarrow^{\preccurlyeq}(A) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(B))$ implies that $\downarrow^{\preccurlyeq}(\mathsf{Post}^*(A)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(B))$.*

Proof. The following is well-known [6] and stems from the monotonicity of Post:

$$\forall X \subseteq \big(\mathbb{N} \cup \{\omega\}\big)^{|P|} : \downarrow^{\preccurlyeq}(\mathsf{Post}^*(X)) = \downarrow^{\preccurlyeq}\Big(\mathsf{Post}^*\big(\downarrow^{\preccurlyeq}(X)\big)\Big) \tag{8}$$

Thus:

$$\begin{array}{rll} & \downarrow^{\preccurlyeq}(A) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(B)) & \\ \Rightarrow & \downarrow^{\preccurlyeq}\big(\mathsf{Post}^*\big(\downarrow^{\preccurlyeq}(A)\big)\big) \subseteq \downarrow^{\preccurlyeq}\big(\mathsf{Post}^*\big(\downarrow^{\preccurlyeq}(\mathsf{Post}^*(B))\big)\big) & \text{Monotonicity of } \mathsf{Post} \text{ and } \downarrow^{\preccurlyeq} \\ \Rightarrow & \downarrow^{\preccurlyeq}(\mathsf{Post}^*(A)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathsf{Post}^*(B))) & \text{By (8)} \\ \Rightarrow & \downarrow^{\preccurlyeq}(\mathsf{Post}^*(A)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(B)) & \end{array}$$

□

We are now ready to state and prove the four invariants.

Lemma 18. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking, Oracle be an oracle, and let $\mathsf{CovSeq}(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}) = (V_i, F_i, O_i)_{i \geq 0}$. Then, for all $i \geq 0$: $\overline{\mathsf{Post}}(V_i) \cup \overline{\mathsf{Accel}}(V_i) \subseteq \downarrow^{\sqsubseteq}(V_i \cup F_i \cup O_i)$.*

Proof. The proof is by induction on i.
Base case $(i = 0)$ $V_0 = \emptyset$ implies that $\overline{\mathsf{Post}}(V_0) = \emptyset$ and $\overline{\mathsf{Accel}}(V_0) = \emptyset$. We conclude that $\overline{\mathsf{Post}}(V_0) \cup \overline{\mathsf{Accel}}(V_0) = \emptyset \subseteq \downarrow^{\sqsubseteq}(V_0 \cup F_0 \cup O_0)$.
Inductive case $(i \geq 1)$ By Definition 15, $V_i \subseteq V_{i-1} \cup F_{i-1}$. Thus, the proof consists in showing separately that (i) $\overline{\mathsf{Post}}(F_{i-1}) \cup \overline{\mathsf{Accel}}(F_{i-1}) \subseteq \downarrow^{\sqsubseteq}(V_i \cup F_i \cup O_i)$ and (ii) $\overline{\mathsf{Post}}(V_{i-1}) \cup \overline{\mathsf{Accel}}(V_{i-1}) \subseteq \downarrow^{\sqsubseteq}(V_i \cup F_i \cup O_i)$. The Lemma is then obtained by $\subseteq$-monotonicity of these two operators.

Let us first show that $\overline{\mathsf{Post}}\,(F_{i-1}) \cup \overline{\mathsf{Accel}}\,(F_{i-1}) \subseteq {\downarrow}^{\sqsubseteq}(V_i \cup F_i \cup O_i)$:

$$\begin{aligned}
& \overline{\mathsf{Post}}\,(F_{i-1}) \cup \overline{\mathsf{Accel}}\,(F_{i-1}) && \\
\subseteq\ & {\downarrow}^{\sqsubseteq}\big(\big(\overline{\mathsf{Post}}\,(F_{i-1}) \cup \overline{\mathsf{Accel}}\,(F_{i-1})\big) \setminus {\downarrow}^{\sqsubseteq}(V_i \cup O_i) \cup {\downarrow}^{\sqsubseteq}(V_i \cup O_i)\big) && \\
=\ & {\downarrow}^{\sqsubseteq}(V_i \cup O_i) \cup {\downarrow}^{\sqsubseteq}\big(\big(\overline{\mathsf{Post}}\,(F_{i-1}) \cup \overline{\mathsf{Accel}}\,(F_{i-1})\big) \setminus {\downarrow}^{\sqsubseteq}(V_i \cup O_i)\big) && \text{By (6)} \\
=\ & {\downarrow}^{\sqsubseteq}(V_i \cup O_i) \cup {\downarrow}^{\sqsubseteq}\Big(\mathsf{Max}^{\sqsubseteq}\,\big(\big(\overline{\mathsf{Post}}\,(F_{i-1}) \cup \overline{\mathsf{Accel}}\,(F_{i-1})\big) \setminus {\downarrow}^{\sqsubseteq}(V_i \cup O_i)\big)\Big) && \text{By (7)} \\
=\ & {\downarrow}^{\sqsubseteq}(V_i \cup O_i) \cup {\downarrow}^{\sqsubseteq}(F_i) && \text{Def. } F_i \\
=\ & {\downarrow}^{\sqsubseteq}(V_i \cup F_i \cup O_i) && \text{By (6)}
\end{aligned}$$

Then, let us show that $\overline{\mathsf{Post}}\,(V_{i-1}) \cup \overline{\mathsf{Accel}}\,(V_{i-1}) \subseteq {\downarrow}^{\sqsubseteq}(V_i \cup F_i \cup O_i)$:

$$\begin{aligned}
& \overline{\mathsf{Post}}\,(V_{i-1}) \cup \overline{\mathsf{Accel}}\,(V_{i-1}) && \\
\subseteq\ & {\downarrow}^{\sqsubseteq}(V_{i-1} \cup F_{i-1}) \cup {\downarrow}^{\sqsubseteq}(O_{i-1}) && \text{By ind. hyp. and (6)} \\
\subseteq\ & {\downarrow}^{\sqsubseteq}(V_{i-1} \cup F_{i-1}) \cup {\downarrow}^{\sqsubseteq}(O_i) && {\downarrow}^{\sqsubseteq}(O_{i-1}) \subseteq {\downarrow}^{\sqsubseteq}(O_i) \text{ (Def. 15)} \\
=\ & {\downarrow}^{\sqsubseteq}\Big(\mathsf{Max}^{\sqsubseteq}\,(V_{i-1} \cup F_{i-1})\Big) \cup {\downarrow}^{\sqsubseteq}(O_i) && \text{By (7)} \\
=\ & {\downarrow}^{\sqsubseteq}\Big(\mathsf{Max}^{\sqsubseteq}\,(V_{i-1} \cup F_{i-1}) \setminus {\downarrow}^{\sqsubseteq}(O_i)\Big) \cup {\downarrow}^{\sqsubseteq}(O_i) && \\
=\ & {\downarrow}^{\sqsubseteq}(V_i) \cup {\downarrow}^{\sqsubseteq}(O_i) && \text{Def. 15} \\
=\ & {\downarrow}^{\sqsubseteq}(V_i \cup O_i) && \text{By (6)} \\
\subseteq\ & {\downarrow}^{\sqsubseteq}(V_i \cup F_i \cup O_i) &&
\end{aligned}$$

□

To each step i of the sequence, we can associate a downward-closed set of computed markings. This set is the downward-closure of the visited markings, and of the markings computed by the oracle, i.e., ${\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(V_i \cup O_i))$. The second lemma tells us that the sequence of computed downward-closed sets is increasing along the sequence (wrt $\subseteq$). Later, we will show that it eventually stabilizes to the covering set of the PN.

Lemma 19. *Let $\mathcal{N}$ be a* PN*, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, and let* $\mathsf{CovSeq}\,(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}) = (V_i, F_i, O_i)_{i \geq 0}$*. Then, for all $i \geq 0$:* ${\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(V_i \cup O_i)) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(V_{i+1} \cup O_{i+1}))$.

Proof. For any $i \geq 1$, we have:

$$\begin{aligned}
& {\downarrow}^{\sqsubseteq}(V_{i-1} \cup O_{i-1}) && \\
\subseteq\ & {\downarrow}^{\sqsubseteq}(V_{i-1} \cup F_{i-1} \cup O_i) && {\downarrow}^{\sqsubseteq}(O_{i-1}) \subseteq {\downarrow}^{\sqsubseteq}(O_i) \text{ and by (6)} \\
=\ & {\downarrow}^{\sqsubseteq}\Big(\mathsf{Max}^{\sqsubseteq}\,(V_{i-1} \cup F_{i-1}) \cup O_i\Big) && \text{By (6) and (7)} \\
=\ & {\downarrow}^{\sqsubseteq}\Big(\Big(\mathsf{Max}^{\sqsubseteq}\,(V_{i-1} \cup F_{i-1})) \setminus {\downarrow}^{\sqsubseteq}(O_i)\Big) \cup O_i\Big) && \text{Def. } \setminus \text{ and } {\downarrow}^{\sqsubseteq} \\
=\ & {\downarrow}^{\sqsubseteq}(V_i \cup O_i) && \text{Def. 15}
\end{aligned}$$

It follows that for any pair $(\mathbf{m}_1, \mathbf{m}_2) \in V_{i-1} \cup O_{i-1}$ there exists a pair $(\mathbf{m}'_1, \mathbf{m}'_2) \in V_i \cup O_i$ such that $(\mathbf{m}_1, \mathbf{m}_2) \sqsubseteq (\mathbf{m}'_1, \mathbf{m}'_2)$. Hence, for all $\mathbf{m} \in \mathsf{Flatten}\,(V_{i-1} \cup O_{i-1})$ there is $\mathbf{m}' \in \mathsf{Flatten}\,(V_i \cup O_i)$ such that $\mathbf{m} \preccurlyeq \mathbf{m}'$. Thus, ${\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(V_i \cup O_i)) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(V_{i+1} \cup O_{i+1}))$ for all $i \geq 0$. □

The third invariant states a property about any pair $(\mathbf{m}_1, \mathbf{m}_2)$ that belongs to some V_i or F_i. It explains the relationship that exists between $\mathbf{m}_2$ and $\mathbf{m}_1$: the downward-closure of $\mathbf{m}_2$ is covered by the successors of $\mathbf{m}_1$.

Lemma 20. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, and let* $\mathsf{CovSeq}\left(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}\right) = (V_i, F_i, O_i)_{i\geq 0}$. *Then, for all $i \geq 0$: for all $(\mathbf{m}_1, \mathbf{m}_2) \in F_i \cup V_i$: $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_1\right))$.*

Proof. The case $i = 0$ is trivial. The other cases are by induction on i.
Base case $(i = 1)$ $F_0 = \{(\mathbf{m}_0, \mathbf{m}_0)\}$ by Definition 15. Hence, $V_1 = \{(\mathbf{m}_0, \mathbf{m}_0)\}$. Clearly, $\downarrow^{\preccurlyeq}(\mathbf{m}_0) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_0\right))$. For any $(\mathbf{m}_1, \mathbf{m}_2) \in F_1$: $(\mathbf{m}_1, \mathbf{m}_2) \in \overline{\mathsf{Post}}\left((\mathbf{m}_0, \mathbf{m}_0)\right)$. Hence, for any $(\mathbf{m}_1, \mathbf{m}_2) \in F_1$: $\mathbf{m}_1 = \mathbf{m}_0$ and $\{\mathbf{m}_2\} \in \mathsf{Post}\left(\mathbf{m}_0\right)$. It follows that $\{\mathbf{m}_2\} \subseteq \mathsf{Post}^*\left(\mathbf{m}_1\right)$, hence $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_1\right))$ by $\subseteq$-monotonicity of $\downarrow^{\preccurlyeq}$.
Inductive case $(i \geq 2)$ By Definition 15, $(\mathbf{m}_1, \mathbf{m}_2) \in V_i$ implies that $(\mathbf{m}_1, \mathbf{m}_2) \in V_{i-1} \cup F_{i-1}$. By induction hypothesis we conclude that, for any $(\mathbf{m}_1, \mathbf{m}_2) \in V_i$: $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_1\right))$. Let us show that the same holds for any $(\mathbf{m}_1, \mathbf{m}_2)$ in F_i. We consider two cases:

(1) If $(\mathbf{m}_1, \mathbf{m}_2) \in \overline{\mathsf{Accel}}\left(F_{i-1}\right)$, then, there exists $\mathbf{m}_3$ s.t. $(\mathbf{m}_3, \mathbf{m}_1) \in F_{i-1}$, $\mathbf{m}_3 \prec \mathbf{m}_1$ and $\mathbf{m}_2 = \mathsf{AccelPair}\left(\mathbf{m}_3, \mathbf{m}_1\right)$. By induction hypothesis, $\downarrow^{\preccurlyeq}(\mathbf{m}_1) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_3\right))$. By Lemma 12, this implies that $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_1\right))$;
(2) If $(\mathbf{m}_1, \mathbf{m}_2) \in \overline{\mathsf{Post}}\left(F_{i-1}\right)$, then there exists, by construction, a pair $(\mathbf{m}_3, \mathbf{m}_4) \in F_{i-1}$ such that $\mathbf{m}_2 \in \mathsf{Post}\left(\mathbf{m}_4\right)$ and either $\mathbf{m}_3 = \mathbf{m}_1$ or $\mathbf{m}_4 = \mathbf{m}_1$. In the first case, by induction hypothesis $\downarrow^{\preccurlyeq}(\mathbf{m}_4) \in \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_1\right))$, hence $\downarrow^{\preccurlyeq}(\mathsf{Post}\left(\mathbf{m}_4\right)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_1\right))$ by Lemma 17 since $\downarrow^{\preccurlyeq}(\mathsf{Post}\left(\mathbf{m}_4\right)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_4\right))$. Finally, $\mathbf{m}_2 \in \mathsf{Post}\left(\mathbf{m}_4\right)$ by construction. It implies $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}\left(\mathbf{m}_4\right))$ by monotonicity of $\downarrow^{\preccurlyeq}$. We conclude that $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_1\right))$. In the second case, we have that $\mathbf{m}_2 \in \mathsf{Post}\left(\mathbf{m}_1\right)$, hence $\{\mathbf{m}_2\} \subseteq \mathsf{Post}^*\left(\mathbf{m}_1\right)$. Thus, $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_1\right))$ by monotonicity of $\downarrow^{\preccurlyeq}$. □

The last invariant states a property of the markings that appear in the pairs belonging to some O_i set. As far as the downward-closure is concerned, these sets of markings are closed to the Post^* operator:

Lemma 21. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, and let* $\mathsf{CovSeq}\left(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}\right) = (V_i, F_i, O_i)_{i\geq 0}$. *Then, for all $i \geq 0$: $\downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathsf{Flatten}\left(O_i\right)\right)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(O_i\right))$.*

Proof. The proof is by induction on i.
Base case $(i = 0)$ Trivial
Inductive case $(i \geq 1)$ Let $\mathbf{m}$ be a marking in $\mathsf{Flatten}\left(O_i\right)$. By Definition 15, either $\mathbf{m} \in \mathsf{Flatten}\left(\mathsf{Oracle}\left(i\right)\right)$ or $\mathbf{m} \in \mathsf{Flatten}\left(O_{i-1}\right)$. In the first case, $\downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}\right)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(\mathsf{Oracle}\left(i\right)\right))$, by Definition 14 point (4)

and $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(\mathsf{Oracle}\,(i))) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_i))$, by Definition 15. In the latter case, $\downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathbf{m})) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_{i-1}))$ by induction hypothesis and $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_{i-1})) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_i))$ by Definition 15. □

Thanks to these invariants, we are now ready to show that the sequence allows to obtain a coverability set of a PN. We proceed as follows. First, we show that the sequence is complete in the sense that there exists k s.t. $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_k))$ is the covering set of the PN. Then, we show the sequence will now compute markings that do not belong to covering set of the PN, and is thus complete.

Completeness of the sequence As a first step towards the proof of correctness of the covering sequence, we prove that every marking computed by the KM algorithm will eventually be covered by some marking computed by the covering sequence. In order to make this proof more readable, we split it into two lemmata. Lemma 22 is rather technical and is ancillary to Lemma 23. The latter shows that for all markings $\mathbf{m}$ computed by the KM algorithm, there exists a finite value k s.t. $\downarrow^{\preccurlyeq}(\mathbf{m}) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_k))$, where k depends on the depth of the node labeled by $\mathbf{m}$ in the KM tree.

Lemma 22. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, and* $\mathsf{CovSeq}\,(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}) = (V_i, F_i, O_i)_{i \geq 0}$. *Let $i \geq 0$ be a natural number, $\mathbf{m}$ be an ω-marking in* $\mathsf{Flatten}\,(V_i)$. *Let $\sigma = t_1 t_2 \cdots t_\ell$ be a non-empty sequence of transitions firable from $\mathbf{m}$, and let $\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_\ell$ be the markings s.t. $\mathbf{m} \xrightarrow{t_1} \mathbf{m}_1 \xrightarrow{t_2} \cdots \xrightarrow{t_\ell} \mathbf{m}_\ell$. Then, there exists $i+1 \leq k \leq i+\ell$ s.t.:*

- *either there is $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_\ell) \in V_k$ with $(\mathbf{m}, \mathbf{m}_\ell) \sqsubseteq (\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_\ell)$;*
- *or $\downarrow^{\preccurlyeq}(\mathbf{m}_\ell) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_k))$.*

Proof. We first consider the case where there are $1 \leq j \leq \ell$ and $i \leq k \leq i+\ell$ s.t. $\downarrow^{\preccurlyeq}(\mathbf{m}_j) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_k))$. In that case, it is straightforward to see that $\downarrow^{\preccurlyeq}(\mathbf{m}_\ell) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_k))$, since $\downarrow^{\preccurlyeq}(\mathbf{m}_\ell) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathbf{m}_j))$, by definition, and since $\downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathbf{m}_j)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_k))$, by Lemma 21. Thus $\downarrow^{\preccurlyeq}(\mathbf{m}_\ell) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_k))$ for some $i \leq k \leq i+\ell$. However, since $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_j)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_{j+1}))$ for any $j \geq 0$ (see Definition 15), we conclude that $\downarrow^{\preccurlyeq}(\mathbf{m}_\ell) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_k))$ for some $i+1 \leq k \leq i+\ell$.

Then, for the rest of the proof, we assume that for every $\mathbf{m}_j$ $(1 \leq j \leq \ell)$ there is no O_n $(i+1 \leq n \leq i+\ell)$ s.t. $\downarrow^{\preccurlyeq}(\mathbf{m}_j) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_n))$. Remark that it implies that for any $i \leq n \leq i+\ell$, there is, in O_n, no pair whose second coordinate is larger than or equal to some $\mathbf{m}_j$ $(1 \leq j \leq \ell)$. Let us show by induction on j that, for any $1 \leq j \leq \ell$, there is, in V_{i+j} a pair $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_j)$ such that $(\mathbf{m}, \mathbf{m}_j) \sqsubseteq (\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_j)$. This means, in particular, that there is, in $V_{i+\ell}$ a pair $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_\ell)$ such that $(\mathbf{m}, \mathbf{m}_\ell) \sqsubseteq (\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_\ell)$.

Base case ($j = 1$) Since $\mathbf{m} \xrightarrow{t_1} \mathbf{m}_1$, and since $\mathbf{m} \in \mathsf{Flatten}\,(V_i)$, the pair $(\mathbf{m}, \mathbf{m}_1)$ is in $\overline{\mathsf{Post}}\,(V_i)$. Hence, by Lemma 18, $(\mathbf{m}, \mathbf{m}_1) \in \downarrow^{\sqsubseteq}(V_i \cup F_i \cup O_i)$. However, $(\mathbf{m}, \mathbf{m}_1) \notin \downarrow^{\sqsubseteq}(O_i)$, by hypothesis. Hence, there is, in $\mathsf{Max}^{\sqsubseteq}\,(V_i \cup F_i)$ a pair $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_1)$ such that

$(\mathbf{m}, \mathbf{m}_1) \sqsubseteq (\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_1)$. Thus $\widehat{\mathbf{m}}_1 \succcurlyeq \mathbf{m}_1$, which implies that $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_1) \notin {\downarrow}^{\sqsubseteq}(O_{i+1})$. Hence, $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_1) \in \mathsf{Max}^{\sqsubseteq}(V_i \cup F_i) \setminus {\downarrow}^{\sqsubseteq}(O_{i+1}) = V_{i+1}$.

Inductive case ($j \geq 2$) By induction hypothesis, there is a pair $(\overline{\mathbf{m}}, \overline{\mathbf{m}}_{j-1})$ in V_{i+j-1} such that $(\mathbf{m}, \mathbf{m}_{j-1}) \sqsubseteq (\overline{\mathbf{m}}, \overline{\mathbf{m}}_{j-1})$. Hence, t_{j-1} is firable from $\overline{\mathbf{m}}_{j-1}$, and, since PN transitions have constant effect, there is a pair $(\overline{\mathbf{m}}, \overline{\mathbf{m}}_j)$ in $\overline{\mathsf{Post}}(V_{i+j-1})$ s.t. $(\overline{\mathbf{m}}, \overline{\mathbf{m}}_j) \sqsupseteq (\mathbf{m}, \mathbf{m}_j)$. By Lemma 18, this implies that $(\overline{\mathbf{m}}, \overline{\mathbf{m}}_j) \in {\downarrow}^{\sqsubseteq}(V_{i+j-1} \cup F_{i+j-1} \cup O_{i+j-1})$. By hypothesis, and since $\overline{\mathbf{m}}_j \succcurlyeq \mathbf{m}_j$, $(\overline{\mathbf{m}}, \overline{\mathbf{m}}_j) \notin {\downarrow}^{\sqsubseteq}(O_{i+j-1})$. Thus, there is in $\mathsf{Max}^{\sqsubseteq}(V_{i+j-1} \cup F_{i+j-1})$ a pair $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_j)$ s.t. $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_j) \sqsupseteq (\overline{\mathbf{m}}, \overline{\mathbf{m}}_j) \sqsupseteq (\mathbf{m}, \mathbf{m}_j)$. Finally, since $\widehat{\mathbf{m}}_j \succcurlyeq \mathbf{m}_j$, the pair $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_j)$ is not in ${\downarrow}^{\sqsubseteq}(O_{i+j})$. Hence, $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}_j) \in \mathsf{Max}^{\sqsubseteq}(V_{i+j-1} \cup F_{i+j-1}) \setminus {\downarrow}^{\sqsubseteq}(O_{i+j}) = V_{i+j}$. □

Lemma 23. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, $\mathcal{T} = \langle N, B, root, \Lambda\rangle$ be the KM tree of $\mathcal{N}$ and* $\mathsf{CovSeq}(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}) = (V_i, F_i, O_i)_{i\geq 0}$. *Then, $\forall n \in N$: $\forall k \geq \sum_{n' \in \mathsf{Anc}(\mathcal{T},n)}(|\varsigma(n')| + 2)$: ${\downarrow}^{\preccurlyeq}(\Lambda(n)) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup O_k))$.*

Proof. The proof is by induction on the length[c] ℓ of the branch ending in n.

Base case ($\ell = 1$) In that case, $n = root$ and $\Lambda(root) = \mathbf{m}_0$. By Definition 15, $V_1 = \{(\mathbf{m}_0, \mathbf{m}_0)\} \setminus {\downarrow}^{\sqsubseteq}(O_1)$, hence $\mathbf{m}_0 \in {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(V_1 \cup O_1))$. Moreover, we have that $1 \leq \sum_{n' \in \mathsf{Anc}(\mathcal{T},root)}(|\varsigma(n')| + 2) = |\varsigma(root)| + 2 = 2$.

Inductive case ($\ell > 1$) Let $n_1, n_2, \ldots, n_{\ell-1}, n_\ell$ be a branch of $\mathcal{T}$ (hence, $n_1 = root$) of length ℓ. By induction hypothesis, there exists $k \leq \sum_{j=1}^{\ell-1}(|\varsigma(n_j)| + 2)$ s.t. ${\downarrow}^{\preccurlyeq}(\Lambda(n_{\ell-1})) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup O_k))$. We consider two cases: either ${\downarrow}^{\preccurlyeq}(\Lambda(n_{\ell-1})) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(O_k))$ or not.

(1) In the case where ${\downarrow}^{\preccurlyeq}(\Lambda(n_{\ell-1})) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(O_k))$, we also have ${\downarrow}^{\preccurlyeq}(\mathsf{Post}^*(\Lambda(n_{\ell-1}))) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(O_k))$, by Lemma 21. However, by Lemma 10, ${\downarrow}^{\preccurlyeq}(\Lambda(n_\ell)) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Post}^*(\Lambda(n_{\ell-1})))$. Hence, ${\downarrow}^{\preccurlyeq}(\Lambda(n_\ell)) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(O_k))$;

(2) In the second case, ${\downarrow}^{\preccurlyeq}(\Lambda(n_{\ell-1})) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(V_k))$. By Lemma 9 and Lemma 13, there are $\mathsf{M}(n_\ell)$, t, $\mathbf{m}'$ and $\varsigma(n_\ell)$ s.t. $\Lambda(n_{\ell-1}) \xrightarrow{t} \mathsf{M}(n_\ell) \xrightarrow{\varsigma(n_\ell)} \mathbf{m}'$ and $\Lambda(n_\ell) \preccurlyeq \mathsf{AccelPair}(\mathsf{M}(n_\ell), \mathbf{m}')$. Remark that by definition of the acceleration, for any marking $\mathbf{m}_1$ in ${\downarrow}^{\preccurlyeq}(\mathsf{Accel}(\mathsf{M}(n_\ell), \mathbf{m}'))$ there exists a marking $\mathbf{m}_2 \succcurlyeq \mathbf{m}_1$ which can be obtained by firing κ times $\varsigma(n_\ell)$ from $\mathsf{M}(n_\ell)$, for some $\kappa \geq 1$. Hence, it is clear that:

$$
{\downarrow}^{\preccurlyeq}(\mathsf{AccelPair}(\mathsf{M}(n_\ell), \mathbf{m}')) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Post}^*(\mathsf{M}(n_\ell))) \tag{9}
$$

Then, since ${\downarrow}^{\preccurlyeq}(\Lambda(n_{\ell-1})) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(V_k))$, and since V_k is a finite set, there is, $\overline{\mathbf{m}} \in V_k$ s.t. $\overline{\mathbf{m}} \succcurlyeq \Lambda(n_{\ell-1})$. Hence, t is firable from $\overline{\mathbf{m}}$ and we let $\overline{\mathbf{m}}'$ be the marking s.t. $\overline{\mathbf{m}} \xrightarrow{t} \overline{\mathbf{m}}'$. Remark that $\overline{\mathbf{m}}' \succcurlyeq \mathsf{M}(n_\ell)$. We can now apply Lemma 22 with $\mathbf{m} = \overline{\mathbf{m}}$ and $\sigma = t$, and conclude that either there is a pair $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}') \in V_{k+1}$ s.t. $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}') \sqsupseteq (\overline{\mathbf{m}}, \overline{\mathbf{m}}')$ or ${\downarrow}^{\preccurlyeq}(\overline{\mathbf{m}}') \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(O_{k+1}))$. In either cases, we conclude that ${\downarrow}^{\preccurlyeq}(\mathsf{M}(n_\ell)) \subseteq {\downarrow}^{\preccurlyeq}(\overline{\mathbf{m}}') \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}(V_{k+1} \cup O_{k+1}))$.

[c] The length of a tree's branch is defined as the number of nodes it contains.

In other words, either $\downarrow^{\preccurlyeq}(\mathsf{M}(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_{k+1}))$ or $\downarrow^{\preccurlyeq}(\mathsf{M}(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_{k+1}))$. We conclude this case by considering three cases:

(a) If $\varsigma\,(n_\ell)$ is the empty sequence, we have $\Lambda\,(n_\ell) = \mathsf{M}(n_\ell)$, hence $\downarrow^{\preccurlyeq}(\Lambda\,(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_{k+1} \cup O_{k+1}))$. Clearly, $k+1 \leq k + |\varsigma\,(n_\ell)| + 2 = k+2$;

(b) If $\varsigma\,(n_\ell)$ is not empty and $\downarrow^{\preccurlyeq}(\mathsf{M}(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_{k+1}))$, then $\downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathsf{M}(n_\ell))) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_{k+1}))$ too, by Lemma 21. However, $\downarrow^{\preccurlyeq}(\Lambda\,(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{AccelPair}\,(\mathsf{M}(n_\ell)\,, \mathbf{m}'))$, by Lemma 13 and $\downarrow^{\preccurlyeq}(\mathsf{AccelPair}\,(\mathsf{M}(n_\ell)\,, \mathbf{m}')) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathsf{M}(n_\ell)))$ by (9). We conclude that $\downarrow^{\preccurlyeq}(\Lambda\,(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_{k+1}))$. Moreover, $k+1 \leq k + |\varsigma\,(n_\ell)| + 2$;

(c) If $\varsigma\,(n_\ell)$ is not empty and $\downarrow^{\preccurlyeq}(\mathsf{M}(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_{k+1}))$, we consider the marking $\overline{\mathbf{m}} \in \mathsf{Flatten}\,(V_{k+1})$ s.t. $\overline{\mathbf{m}} \succcurlyeq \mathsf{M}(n_\ell)$ and let $\overline{\mathbf{m}}'$ be s.t. $\overline{\mathbf{m}} \xrightarrow{\varsigma(n_\ell)} \overline{\mathbf{m}}'$. We apply Lemma 22 again, with $\mathbf{m} = \overline{\mathbf{m}}$ and $\sigma = \varsigma\,(n_\ell)$. We conclude that there exists $r \leq k + 1 + |\varsigma\,(n_\ell)|$ s.t. either $\downarrow^{\preccurlyeq}(\overline{\mathbf{m}}') \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_r))$ or there is a pair $(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}')$ in V_r such that $(\overline{\mathbf{m}}, \overline{\mathbf{m}}') \sqsubseteq (\widehat{\mathbf{m}}, \widehat{\mathbf{m}}')$. In the first case, we conclude that $\downarrow^{\preccurlyeq}(\Lambda\,(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_r))$ using the same reasoning as in point 2b. In the former case, we have $(\mathsf{M}(n_\ell)\,, \mathbf{m}') \sqsubseteq (\widehat{\mathbf{m}}, \widehat{\mathbf{m}}')$. Hence, $\Lambda\,(n_\ell) = \mathsf{AccelPair}\,(\mathsf{M}(n_\ell)\,, \mathbf{m}') \preccurlyeq \mathsf{AccelPair}\,(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}')$. Moreover, $\downarrow^{\preccurlyeq}(\mathsf{AccelPair}\,(\widehat{\mathbf{m}}, \widehat{\mathbf{m}}')) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_{r+1} \cup O_{r+1}))$ by Lemma 18 and Definition 15 (with $r + 1 \leq k + |\varsigma\,(n_\ell)| + 2$).

In all the cases, we conclude that $\downarrow^{\preccurlyeq}(\Lambda\,(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_r \cup O_r))$ for some $r \leq k + |\varsigma\,(n_\ell)| + 2 \leq \sum_{j=1}^{\ell}(|\varsigma\,(n_j)| + 2)$. Finally, by Lemma 19, we conclude that, for any $k \geq \sum_{j=1}^{\ell}(|\varsigma\,(n_j)| + 2)$: $\downarrow^{\preccurlyeq}(\Lambda\,(n_\ell)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_k \cup O_k))$. □

As a consequence, the covering sequence is complete:

Corollary 24. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, and* $\mathsf{CovSeq}\,(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}) = (V_i, F_i, O_i)_{i \geq 0}$. *There exists $k \geq 0$ such that for all $\ell \geq k$ we have* $\mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(V_\ell \cup O_\ell))$.

Soundness of the sequence In order to show that the covering sequence is correct, it remains to show that any marking $\mathbf{m}$ produced by the sequence is s.t. $\downarrow^{\preccurlyeq}(\mathbf{m}) \subseteq \mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0)$. For that purpose, we need the following ancillary lemma:

Lemma 25. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, and let* $\mathsf{CovSeq}\,(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}) = (V_i, F_i, O_i)_{i \geq 0}$. *Then, $\forall i \geq 1$: $\forall \mathbf{m} \in$* $\mathsf{Flatten}\,(V_i \cup F_i \cup O_i)$*:* $\downarrow^{\preccurlyeq}(\mathbf{m}) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathbf{m}_0))$.

Proof. The proof is by induction on i.
Base case ($i = 0$) Trivial.
Inductive case ($i \geq 1$) By Definition 15, $(\mathbf{m}_1, \mathbf{m}_2) \in V_i$ implies that $(\mathbf{m}_1, \mathbf{m}_2) \in V_{i-1} \cup F_{i-1}$. By induction hypothesis, we conclude that $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \in \downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathbf{m}_0))$. Furthermore, $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(\mathsf{Oracle}\,(i))) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathbf{m}_0))$, by Definition 14, point (5). Hence, for all $\mathbf{m} \in \downarrow^{\preccurlyeq}(\mathsf{Flatten}\,(O_i))$: $\downarrow^{\preccurlyeq}(\mathbf{m}) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\,(\mathbf{m}_0))$.

It remains to show that for any $(\mathbf{m}_1, \mathbf{m}_2) \in F_i$: $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \in \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$. We consider two cases:

(1) If $(\mathbf{m}_1, \mathbf{m}_2) \in \overline{\mathsf{Post}}(F_{i-1})$, then by construction it implies that there exists a pair $(\mathbf{m}_3, \mathbf{m}_4) \in F_{i-1}$ such that either $\mathbf{m}_3 = \mathbf{m}_1$ or $\mathbf{m}_4 = \mathbf{m}_1$ and $\mathbf{m}_2 \in \mathsf{Post}(\mathbf{m}_4)$. By induction hypothesis, since $\mathbf{m}_4 \in \downarrow^{\preccurlyeq}(\mathsf{Flatten}(F_{i-1}))$, we know that $\downarrow^{\preccurlyeq}(\mathbf{m}_4) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$. By Lemma 17, it follows that $\downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_4)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$. Since $\mathbf{m}_2 \in \mathsf{Post}(\mathbf{m}_4) \subseteq \mathsf{Post}^*(\mathbf{m}_4)$, we conclude that $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_4))$, by monotonicity of $\downarrow^{\preccurlyeq}$. Hence, $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$.

(2) If $(\mathbf{m}_1, \mathbf{m}_2) \in \overline{\mathsf{Accel}}(F_{i-1})$, then there exists, by construction, a pair $(\mathbf{m}_3, \mathbf{m}_1) \in F_{i-1}$ such that $\mathbf{m}_3 \prec \mathbf{m}_1$ and $\mathbf{m}_2 = \mathsf{AccelPair}(\mathbf{m}_3, \mathbf{m}_1)$. By Lemma 20, $\downarrow^{\preccurlyeq}(\mathbf{m}_1) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_3))$. Hence, by Lemma 12, we conclude that $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_1))$. Furthermore, $\downarrow^{\preccurlyeq}(\mathbf{m}_1) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$, by induction hypothesis. By Lemma 17, it follows that $\downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_1)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$. We conclude that $\downarrow^{\preccurlyeq}(\mathbf{m}_2) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*(\mathbf{m}_0))$. □

As a consequence, we directly obtain our soundness result:

Corollary 26. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, and* $\mathsf{CovSeq}(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}) = (V_i, F_i, O_i)_{i \geq 0}$. *Then,* $\forall i \geq 1$, $\downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_i \cup O_i)) \subseteq \mathsf{Cover}(\mathcal{N}, \mathbf{m}_0)$.

Corollary 24 and 26 allow us to obtain the next Theorem.

Theorem 27. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be its initial marking,* Oracle *be an oracle, and* $\mathsf{CovSeq}(\mathcal{N}, \mathbf{m}_0, \mathsf{Oracle}) = (V_i, F_i, O_i)_{i \geq 0}$. *Then, there exists $k \geq 0$ such that*

(1) for all $1 \leq i < k : \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_i \cup O_i)) \subset \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_{i+1} \cup O_{i+1}))$;
(2) for all $i \geq k : \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_i \cup O_i)) = \mathsf{Cover}(\mathcal{N}, \mathbf{m}_0)$.

Proof. By Corollary 24 and 26, we conclude that there exists at least one $k \in \mathbb{N}$ such that $\downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup O_k)) = \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_{k+1} \cup O_{k+1}))$. Let us consider the smallest $k \in \mathbb{N}$ that satisfies that condition and let us prove that $\downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup O_k)) = \mathsf{Cover}(\mathcal{N}, \mathbf{m}_0)$. Note that by Lemma 19 we have for all $0 \leq i < k : \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_i \cup O_i)) \subset \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_{i+1} \cup O_{i+1}))$.

First, we prove that $\downarrow^{\preccurlyeq}(\mathsf{Flatten}(F_k)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup O_k))$. By construction, $\downarrow^{\sqsubseteq}(F_k) \subseteq \downarrow^{\sqsubseteq}(V_{k+1} \cup O_{k+1})$. Hence, $\downarrow^{\preccurlyeq}(\mathsf{Flatten}(F_k)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_{k+1} \cup O_{k+1}))$, by definition of $\sqsubseteq$. However, $\downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_{k+1} \cup O_{k+1})) = \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup O_k))$, by definition of k.

Let us finish the proof by showing that $\downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_i \cup O_i))$ is indeed the covering set $\mathsf{Cover}(\mathcal{N}, \mathbf{m}_0)$. By Lemma 18, $\overline{\mathsf{Post}}(V_k) \cup \overline{\mathsf{Accel}}(V_k) \subseteq \downarrow^{\sqsubseteq}(V_k \cup F_k \cup O_k)$, which implies that $\downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\big(\overline{\mathsf{Post}}(V_k) \cup \overline{\mathsf{Accel}}(V_k)\big)\big) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup F_k \cup O_k))$. In particular, $\downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\big(\overline{\mathsf{Post}}(V_k)\big)\big) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup F_k \cup O_k))$. Since $\downarrow^{\preccurlyeq}(\mathsf{Flatten}(F_k)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}(V_k \cup O_k))$, we have $\downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\left(\overline{\mathsf{Post}}\left(V_k\right)\right)\big) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_k \cup O_k\right))$. By Definition of $\overline{\mathsf{Post}}$, this means that $\downarrow^{\preccurlyeq}(\mathsf{Post}\left(\mathsf{Flatten}\left(V_k\right)\right)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_k \cup O_k\right))$. Furthermore, by (4) and definition of O_k, $\downarrow^{\preccurlyeq}(\mathsf{Post}\left(\mathsf{Flatten}\left(O_k\right)\right)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(O_k\right))$. We conclude that $\downarrow^{\preccurlyeq}(\mathsf{Post}\left(\mathsf{Flatten}\left(V_k \cup O_k\right)\right)) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_k \cup O_k\right))$. Then, by Lemma 19, and since $\downarrow^{\preccurlyeq}(\mathbf{m}_0) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_1 \cup O_1\right))$, we have $\downarrow^{\preccurlyeq}(\mathbf{m}_0) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_k \cup O_k\right))$. Hence, $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_k \cup O_k\right))$ is a Post fixpoint that contains $\downarrow^{\preccurlyeq}(\mathbf{m}_0)$. That is, $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_k \cup O_k\right)) \supseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_0\right))$. Since, by Corollary 26, $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(O_k \cup V_k\right)) \subseteq \mathsf{Cover}\left(\mathcal{N}, \mathbf{m}_0\right)$, we have: $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_k \cup O_k\right)) = \mathsf{Cover}\left(\mathcal{N}, \mathbf{m}_0\right)$. Finally, by Lemma 19 and Corollary 26, we conclude that $\forall i > k$: $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_i \cup O_i\right)) = \downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(V_k \cup O_k\right)) = \mathsf{Cover}\left(\mathcal{N}, \mathbf{m}_0\right)$. $\square$

6. An efficient oracle

To implement the method in practice, we have to instantiate the oracle. First, recall that the *empty oracle*, i.e. $\mathsf{Oracle}\left(i\right) = \emptyset$ for all $i \geq 1$, is a correct one. Thus, the oracle can be regarded as an optional optimization of our algorithm. Yet, this optimization can be very powerful, as we are about to show.

In order to obtain an efficient algorithm, we implement the oracle as a *recursive call* on selected ω-markings resulting from an acceleration. Let us explain the intuition behind this idea. First, observe that, for any PN $\mathcal{N}$ with initial marking $\mathbf{m}_0$, and for any $\mathbf{m}_1$ s.t. $\downarrow^{\preccurlyeq}(\mathbf{m}_1) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_0\right))$, $\mathsf{Cover}\left(\mathcal{N}, \mathbf{m}_1\right) \subseteq \mathsf{Cover}\left(\mathcal{N}, \mathbf{m}_0\right)$. Thus, one can, in some sense, divide the work of computing a coverability set of $\mathcal{N}$ from $\mathbf{m}_0$ by first computing a coverability from some well-chosen ω-markings $\mathbf{m}_1, \ldots, \mathbf{m}_\ell$ that are reachable from $\mathbf{m}_0$, and then completing this partial solution. As a consequence, it is easy to see that a function $f(i)$ that would return, for any $i \geq 0$ a set of pairs S s.t. $\downarrow^{\preccurlyeq}(\mathsf{Flatten}\left(S\right)) = \mathsf{Cover}\left(\mathcal{N}, \mathbf{m}_i\right)$ for some $\mathbf{m}_i$ s.t. $\downarrow^{\preccurlyeq}(\mathbf{m}_i) \subseteq \downarrow^{\preccurlyeq}(\mathsf{Post}^*\left(\mathbf{m}_0\right))$ respects the definition of an oracle. It thus makes perfectly sense to perform a recursive call to implement the oracle. The question that remains to be solved is thus: how to select the markings on which recursive calls will be performed so that the algorithm is efficient ?

The strategy we have adopted to select these markings is rooted in the following observation. When using the empty oracle our method performs a *breadth first* search (see the discussion on the definition of the covering sequence, in the previous section). In particular, if several accelerations can be applied from an ω-marking $\mathbf{m}$, each of them putting ω's in different places, then all the possible orders for their application will be investigated. For instance, consider a marking $\mathbf{m}$ from which two infinitely increasing sequences of markings are reachable. In the first sequence, the marking of p_1 strictly increases, whereas the marking of p_2 strictly increases in the second sequence. Hence, two accelerations (one that produces an ω in p_1 and one that produces an ω in p_2) can be applied. With the empty oracle, our algorithm will first build a marking $\mathbf{m}_1$ resulting from the acceleration on p_1 (i.e., with an ω in p_1, but no ω in p_2), and a marking $\mathbf{m}_2$ resulting from the acceleration on p_2. Then, at the next step, a marking $\mathbf{m}_3$ with ω's in both p_1 and p_2 will be computed,

```
Data: A PN $\mathcal{N} = \langle P, T\rangle$, an initial $\omega$-marking $\mathbf{m}_0$
Result: A set of pairs of markings.
CovProc $(\mathcal{N}, \mathbf{m}_0)$ begin
    $i := 0$ ; $\overline{O}_0 := \emptyset$ ; $\overline{V}_0 := \emptyset$ ; $\overline{F}_0 := \{(\mathbf{m}_0, \mathbf{m}_0)\}$ ;
    repeat
        $i := i + 1$ ;
        $R_i := \cup_{\mathbf{m}\in S}\mathsf{CovProc}\,(\mathcal{N}, \mathbf{m})$ where $S \subseteq \mathsf{Flatten}\,\big(\overline{\mathsf{Accel}}\,(F_{i-1})\big)$;
        $\overline{O}_i := \mathsf{Max}^{\sqsubseteq}\,\big(\overline{O}_{i-1} \cup R_i\big)$ ;
        $\overline{V}_i := \mathsf{Max}^{\sqsubseteq}\,\big(\overline{V}_{i-1} \cup \overline{F}_{i-1}\big) \setminus {\downarrow^{\sqsubseteq}}\big(\overline{O}_i\big)$ ;
        $\overline{F}_i := \mathsf{Max}^{\sqsubseteq}\,\big(\overline{\mathsf{Post}}\,\big(\overline{F}_{i-1}\big) \cup \overline{\mathsf{Accel}}\,\big(\overline{F}_{i-1}\big)\big) \setminus {\downarrow^{\sqsubseteq}}\big(\overline{O}_i \cup \overline{V}_i\big)$ ;
    until ${\downarrow^{\preccurlyeq}}\big(\mathsf{Flatten}\,\big(\overline{O}_i \cup \overline{V}_i\big)\big) \subseteq {\downarrow^{\preccurlyeq}}\big(\mathsf{Flatten}\,\big(\overline{O}_{i-1} \cup \overline{V}_{i-1}\big)\big)$ ;
    return($\overline{O}_i \cup \overline{V}_i$) ;
end
```

Algorithm 3: The CovProc algorithm.

that summarizes both accelerations, and is strictly larger than $\mathbf{m}_1$ and $\mathbf{m}_2$. Thus, the building of *both* $\mathbf{m}_1$ and $\mathbf{m}_2$ can be regarded as unnecessary, and only one of them should be investigated.

As a consequence, our strategy consists in giving the priority to the markings $\mathbf{m}$ that have the largest amount of places p s.t. $\mathbf{m}(p) = \omega$. These markings can be identified thanks to the $\overline{\mathsf{Accel}}$ function: when it returns a non-empty set of pairs we are guaranteed that the markings that appear in these pairs contain *more ω's* than the initial marking. These markings are thus good candidates to perform the recursive call. Remark that, when applying this strategy, the initial breadth first search is mixed with a depth first search that allows to develop first the ω-markings resulting from an acceleration.

The CovProc procedure The CovProc procedure implements these ideas and is shown in Algorithm 3. It closely follows the definition of the covering sequence. At each step i, the oracle is implemented as a finite number of recursive calls to CovProc, where the initial ω-markings are the results of the accelerations occurring at this step. In the presentation of the procedure, the recursive call is not applied on all the accelerated ω-markings but in a non-deterministically chosen subset S. Indeed, in practice, if we have two accelerated ω-markings $\mathbf{m}_1$ and $\mathbf{m}_2$ with ${\downarrow^{\preccurlyeq}}(\mathbf{m}_2) \subseteq {\downarrow^{\preccurlyeq}}(\mathsf{Post}^*\,(\mathbf{m}_1))$, then it is not necessary to apply CovProc on $\mathbf{m}_2$ to explore the ω-markings that are reachable from $\mathbf{m}_2$. Keeping non-determinism in the CovProc procedure allows to provide a more general proof: we show hereunder that the procedure is correct for any possible resolution of the non-determinism. We later explain how we have chosen to resolve the non-determinism in the experiments we report.

This strategy allows to mix the breadth-first exploration of the covering sequence and the depth-first exploration due to the recursive calls which favor ω-markings with more ω. It turns out to be very efficient in practice (see hereunder). Since, for any pair $(\mathbf{m}_1, \mathbf{m}_2)$, $\mathsf{AccelPair}\,(\mathbf{m}_1, \mathbf{m}_2)$ contains strictly more ω's than $\mathbf{m}_1$ and $\mathbf{m}_2$, and since the number of places of the PN is bounded, the depth of recursion is bounded too, which ensures termination.

Let us show that this solution is correct and terminates. For any ω-marking $\mathbf{m}$, we let $\mathsf{Nb}\omega\,(\mathbf{m}) = |\{p \mid \mathbf{m}(p) = \omega\}|$. Then:

Lemma 28. *Let $\mathcal{N}$ be a PN, $\mathbf{m}_0$ be an ω-marking, and let $\overline{F}_i$ be the sets computed by $\mathsf{CovProc}\,(\mathcal{N}, \mathbf{m}_0)$. Then, $\forall i \geq 0$: for any $\mathbf{m} \in \mathsf{Flatten}\left(\overline{F}_i\right)$: $\mathsf{Nb}\omega\,(\mathbf{m}) \geq \mathsf{Nb}\omega\,(\mathbf{m}_0)$.*

Proof. The proof is by induction on i.
Base case $(i = 0)$ Trivial.
Inductive case $(i \geq 1)$ First remark that for any $\mathbf{m}$, and any $\mathbf{m}' \in \mathsf{Post}\,(\mathbf{m})$, $\mathsf{Nb}\omega\,(\mathbf{m}') = \mathsf{Nb}\omega\,(\mathbf{m})$, because PN transitions have constant effect. Moreover, for any pair of markings $(\mathbf{m}_1, \mathbf{m}_2)$ s.t. $\mathbf{m}_1 \prec \mathbf{m}_2$: $\mathsf{Nb}\omega\,(\mathsf{AccelPair}\,(\mathbf{m}_1, \mathbf{m}_2)) > \mathsf{Nb}\omega\,(\mathbf{m}_2)$. Thus, by definition of $\overline{F}_i$, for any $\mathbf{m} \in \mathsf{Flatten}\left(\overline{F}_i\right)$, there exists $\mathbf{m}' \in \mathsf{Flatten}\left(\overline{F}_{i-1}\right)$ s.t. $\mathsf{Nb}\omega\,(\mathbf{m}) \geq \mathsf{Nb}\omega\,(\mathbf{m}')$. However, by induction hypothesis, $\mathsf{Nb}\omega\,(\mathbf{m}') \geq \mathsf{Nb}\omega\,(\mathbf{m}_0)$ for any $\mathbf{m}' \in \mathsf{Flatten}\left(\overline{F}_{i-1}\right)$. Hence the lemma. $\square$

Then, the proof of total correctness of $\mathsf{CovProc}$ is as follows:

Theorem 29. *For any PN $\mathcal{N}$ and any ω-marking $\mathbf{m}_0$: $\mathsf{CovProc}\,(\mathcal{N}, \mathbf{m}_0)$ terminates and ${\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(\mathsf{CovProc}\,(\mathcal{N}, \mathbf{m}_0))) = \mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0)$.*

Proof. The proof works by induction on $\mathsf{Nb}\omega\,(\mathbf{m}_0)$.
Base case $(\mathsf{Nb}\omega\,(\mathbf{m}_0) = |P|)$ In that case, $\mathsf{CovProc}\,(\mathcal{N}, \mathbf{m}_0)$ finishes after two iterations and returns $\overline{O}_2 \cup \overline{V}_2 = \{(\mathbf{m}_0, \mathbf{m}_0)\}$. Remark that no recursive call is performed because $R_1 = \overline{\mathsf{Accel}}\left(\overline{F}_0\right) = \emptyset$ and $R_2 = \overline{\mathsf{Accel}}\,(\{(\mathbf{m}_0, \mathbf{m}_0)\}) = \emptyset$. Moreover, ${\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(\mathsf{CovProc}\,(\mathcal{N}, \mathbf{m}_0))) = {\downarrow}^{\preccurlyeq}(\mathbf{m}_0) = \mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0)$.
Inductive case $(\mathsf{Nb}\omega\,(\mathbf{m}_0) = k < |P|)$ We consider two cases. First, assume that the algorithm terminates after ℓ iterations, i.e., assume that ${\downarrow}^{\preccurlyeq}\left(\mathsf{Flatten}\left(\overline{O}_\ell \cup \overline{V}_\ell\right)\right) = {\downarrow}^{\preccurlyeq}\left(\mathsf{Flatten}\left(\overline{O}_{\ell-1} \cup \overline{V}_{\ell-1}\right)\right)$, but for any $1 \leq j \leq \ell - 1$: ${\downarrow}^{\preccurlyeq}\left(\mathsf{Flatten}\left(\overline{O}_j \cup \overline{V}_j\right)\right) \neq {\downarrow}^{\preccurlyeq}\left(\mathsf{Flatten}\left(\overline{O}_{j-1} \cup \overline{V}_{j-1}\right)\right)$. By Lemma 28, for any $1 \leq j \leq \ell$, for any $(\mathbf{m}_1, \mathbf{m}_2) \in \overline{F}_j$: $\mathsf{Nb}\omega\,(\mathbf{m}_2) \geq k$. Hence, for any $1 \leq j \leq \ell$, for any $\mathbf{m} \in \mathsf{Flatten}\left(\overline{\mathsf{Accel}}\left(\overline{F}_j\right)\right)$: $\mathsf{Nb}\omega\,(\mathbf{m}) \geq k + 1$. Thus, by induction hypothesis, for any $1 \leq j \leq \ell$, for any $\mathbf{m} \in \mathsf{Flatten}\left(\overline{\mathsf{Accel}}\left(\overline{F}_j\right)\right)$, $\mathsf{CovProc}\,(\mathcal{N}, \mathbf{m})$ terminates and returns a set of pairs such that ${\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(\mathsf{CovProc}\,(\mathcal{N}, \mathbf{m}))) = \mathsf{Cover}\,(\mathcal{N}, \mathbf{m})$. As a consequence, and since $\mathsf{Flatten}\left(\overline{\mathsf{Accel}}\left(\overline{F}_j\right)\right)$ is a finite set for any $1 \leq j \leq \ell$, we conclude that R_j is computed in a finite amount of time and that ${\downarrow}^{\preccurlyeq}(\mathsf{Post}\,(\mathsf{Flatten}\,(R_j))) \subseteq {\downarrow}^{\preccurlyeq}(\mathsf{Flatten}\,(R_j)) \subseteq \mathsf{Cover}\,(\mathcal{N}, \mathbf{m}_0)$, for any $1 \leq j \leq \ell$.

Let Ω denote the function s.t., for any $1 \leq j \leq \ell$: $\Omega(j) = R_j$, and, for any $j > \ell$, $\Omega(j) = \emptyset$. Thus, Ω is an oracle. Let us assume that $\mathsf{CovSeq}\,(\mathcal{N}, \mathbf{m}_0, \Omega) = (V_i, F_i, O_i)_{i \geq 1}$. Clearly, for any $0 \leq j \leq \ell$, $\overline{V}_j = V_j$ and $\overline{O}_j = O_j$. Thus, by Theorem 27, there exists k s.t. $\downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\left(\overline{V}_{k-1} \cup \overline{O}_{k-1}\right)\big) = \downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\left(\overline{V}_{k} \cup \overline{O}_{k}\right)\big) = \mathsf{Cover}\left(\mathcal{N}, \mathbf{m}_0\right)$, and s.t. for every $1 \leq j \leq k-1$: $\downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\left(\overline{V}_{j-1} \cup \overline{O}_{j-1}\right)\big) \subset \downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\left(\overline{V}_{j} \cup \overline{O}_{j}\right)\big)$. Hence, $k = \ell$, and we conclude that $\mathsf{CovProc}\left(\mathcal{N}, \mathbf{m}_0\right)$ terminates and returns $\downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\left(\overline{V}_{\ell} \cup \overline{O}_{\ell}\right)\big) = \mathsf{Cover}\left(\mathcal{N}, \mathbf{m}_0\right)$.

In the latter case, we assume that the algorithm does not terminate and derive a contradiction. This can happen either because the test of the **repeat** loop is never fulfilled, or because some step j of the loop takes an infinite time to complete. The latter is not possible. Indeed, R_j is computed in a finite amount of time and the functions $\mathsf{Max}^{\sqsubseteq}, \mathsf{Flatten}, \overline{\mathsf{Post}}, \overline{\mathsf{Accel}}$, as well as the guard of the loop are computable. Thus, if the algorithm does not terminate, it computes an infinite sequence of sets $(\overline{V}_i, \overline{F}_i, \overline{O}_i)_{i\geq 0}$. Symmetrically to the first part of this proof, we build an oracle Ω s.t. $\Omega(j) = R_j$ for any $j \geq 1$. Let us assume that $\mathsf{CovSeq}\left(\mathcal{N}, \mathbf{m}_0, \Omega\right) = (V_i, F_i, O_i)_{i\geq 0}$. Clearly, for any $j \geq 0$, we have $V_j = \overline{V}_j$ and $O_j = \overline{O}_j$. Hence, by Theorem 27, we conclude that there exists $k \geq 1$ s.t. $\downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\left(\overline{V}_{k} \cup \overline{O}_{k}\right)\big) = \downarrow^{\preccurlyeq}\big(\mathsf{Flatten}\left(\overline{V}_{k-1} \cup \overline{O}_{k-1}\right)\big)$, which compels the algorithm to terminate at step k. Contradiction. □

Empirical evaluation We have implemented a prototype that computes the coverability set of a PN, thanks to the covering sequence method and the Karp&Miller algorithm. We have selected bounded and unbounded PN [d]. The prototype has been written in the PYTHON programming language in a very straightforward way. As a consequence, running times are given for the sake of comparison only. Nevertheless, the prototype performs very well on our examples (Table 1).

Let us briefly discuss the selected examples. *RTP* is a basic communication protocol. *Lamport*, *Peterson* and *Dekker* are models of these three mutual exclusion protocols, with two threads. *ReadWrite* is a another mutual exclusion protocol, where threads compete to read from and write to a shared resource. *basicME* is a basic mutual exclusion protocol, with an unbounded number of threads. *CSM*, *FMS*, *Kanban* and *Multipoll* describe concurrent production systems. *PNCSA* [3] is a security protocol. Finally *Mesh2×2* describes a system in which a set of processors share the execution of various processes.

In Table 1, we have compared two implementations of the covering sequence to the KM algorithm. The former (column **Cov. Seq. w/o oracle**) is the covering sequence where $\mathsf{Oracle}\,(i) = \emptyset$ for any $i \geq 0$ (that is, the oracle is disabled). In that case the sets of pairs built by our algorithm are small (see column Max P.) wrt the size of the KM tree (column Nodes), although the number of pairs created by the algorithm (column Tot. P.) is not dramatically small wrt the KM tree. This shows the efficiency of our approach based on *pairs* of ω-markings (and on the $\sqsubseteq$ order).

The latter implementation is the CovProc procedure (Algorithm 3) where the non-determinism in the choice of the set S is resolved as follows. We consider the accelerated ω-markings one by one. A recursive call is applied on an accelerated

[d]See `http://www.ulb.ac.be/di/ssd/ggeeraer/eec` for a complete description.

ω-marking iff it is not yet covered by the Flatten of the pairs computed by previous recursive calls. In the case of *bounded* PN, CovProc performs as the covering sequence with trivial oracle, which is not surprising since no acceleration occur. In the case of *unbounded* PN, the oracle-based optimization is successful: the sets of pairs built by CovProc are much smaller than the respective KM trees, and negligible with respect to the sets built with the trivial oracle. The CovProc procedure always terminates within 20 minutes and outperforms the covering sequence with trivial oracle. Finally, the execution times of CovProc are several order of magnitudes smaller than those of the KM procedure, showing the interest of our new algorithm.

7. Conclusion

In the present paper, we have discussed methods to compute the minimal coverability set of Petri nets. We have seen that the MCT algorithm [3], that had been introduced as an improvement of the classical Karp&Miller algorithm [10] is flawed, and that previous attempts to fix it [11] introduce new errors. The reason for these problems seems to be that the requirement of keeping, *along the whole computation* trees whose nodes are all incomparable, is too strong. Instead, we have introduced a different minimality criterion: we consider pairs of markings, and we are able to compute the minimal coverability set of a PN by maintaining sets of *incomparable pairs* only.

The empirical results presented at the end of Section 6 show the potential of our new approach: the sizes of the structures manipulated by our algorithm are significantly smaller than the sizes of the trees manipulated by the Karp&Miller algorithm. Yet, a truly efficient implementation of the ideas introduced in this paper is still to be done. Such an implementation should rely on a symbolic datastructure, such as the Covering Sharing Trees [1]. This structure has already been exploited in tools that solve the coverability problem of Petri nets [14], to store and manipulate, in an efficient fashion, sets of markings. It can however easily be adapted to cope with pairs of markings.

Acknowledgements

The authors are grateful to Prof. Peter Starke and Prof. Alain Finkel for their friendly cooperation, and to the anonymous reviewers for their comments and suggestions.

References

[1] Delzanno, G., Raskin, J.-F. and Van Begin, L. CSTs (Covering Sharing Trees): compact Data Structures for Parameterized Verification. International Journal on Software Tools for Technology Transfer, 5(2-3):268-297, 2004, Springer

[2] Finkel, A.: Reduction and covering of infinite reachability trees. Information and Computation, **89**(2), 1990, Academic Press, Inc.

[3] Finkel, A.: The minimal coverability graph for Petri nets. Papers from the 12th International Conference on Applications and Theory of Petri Nets: Advances in Petri Nets 1993, Lecture Notes In Computer Science, Vol. 674, 1991, Springer.
[4] Finkel, A., Geeraerts, G., Raskin, J.F., Van Begin, L.: A counter-example to the minimal coverability tree algorithm. Technical Report 535, ULB (2005)
[5] Geeraerts, G.: Coverability and Expressiveness Properties of Well-structured Transition Systems. PhD thesis, Université Libre de Bruxelles, Belgium (2007)
[6] Geeraerts, G., Raskin, J.F., Van Begin, L.: Well-structured languages. Acta Informatica, **44**(3-4), 2007, Springer.
[7] Geeraerts, G., Raskin, J.F., Van Begin, L.: On the efficient computation of the coverability set for Petri nets. Proceedings of ATVA07, 5th International Symposium on Automated Technology for Verification and Analysis. Lecture Notes In Computer Science, Vol. 4762, 2007, Springer.
[8] German, S., Sistla, A.: Reasoning about Systems with Many Processes. Journal of the ACM (JACM), **39**(3), 1992, ACM Press.
[9] Grahlmann, B.: The pep tool. Proceedings of CAV97, 9th International Conference on Computer Aided Verification, Lecture Notes In Computer Science, Vol. 1254, 1997, Springer.
[10] Karp, R.M., Miller, R.E.: Parallel Program Schemata. Journal of Computer ans System Sciences, **3**, 1969, Elsevier.
[11] Luttge, K.: Zustandsgraphen von Petri-Netzen. Master's thesis, Humboldt-Universität (1995)
[12] Reisig, W.: Petri Nets. An introduction. Springer (1986)
[13] Starke, P.: Personnal communication
[14] Van Begin, L.: Efficient Verification of Counting Abstractions for Parametric systems. PhD thesis, Université Libre de Bruxelles, Belgium (2003)

Table 1. Empirical evaluation of the covering sequence. Experiments on an Intel Xeon 3GHz. Times in seconds ($\times$ = no result within 20 minutes). P = number of places; T = number of transitions; MCS = size of the minimal coverability set ; Tp = Bounded or Unbounded PN; Max P. = $\max\{|V_i \cup O_i \cup F_i|, i \geq 1\}$; Tot. P. = tot. number of pairs created along the whole execution.

Example					KM		**Cov. Seq. w/o Oracle**			CovProc		
Name	P	T	MCS	Tp	Nodes	Time	Max P.	Tot. P.	Time	Max P.	Tot. P.	Time
RTP	9	12	9	B	16	0.18	47	47	0.10	47	47	0.13
lamport	11	9	14	B	83	0.18	115	115	0.17	115	115	0.17
peterson	14	12	20	B	609	2.19	170	170	0.21	170	170	0.25
dekker	16	14	40	B	7,936	258.95	765	765	1.13	765	765	1.03
readwrite	13	9	41	B	11,139	529.91	1,103	1,103	1.43	1,103	1,103	1.75
manuf.	13	6	1	U	32	0.19	9	101	0.18	2	47	0.14
kanban	16	16	1	U	9,839	1221.96	593	9,855	95.05	4	110	0.19
basicME	5	4	3	U	5	0.10	5	5	0.12	5	5	0.12
CSM	14	13	16	U	$>2.40\times10^6$	$\times$	371	3,324	14.38	178	248	0.34
FMS	22	20	24	U	$>6.26\times10^5$	$\times$	>4,460	$\times$	$\times$	477	866	2.10
PNCSA	31	36	80	U	$>1.02\times10^6$	$\times$	>5,896	$\times$	$\times$	2,617	13,408	113.79
multipoll	18	21	220	U	$>1.16\times10^6$	$\times$	>7,396	$\times$	$\times$	14,034	14,113	365.90
mesh2$\times$2	32	32	256	U	$>8.03\times10^5$	$\times$	>6,369	$\times$	$\times$	10,483	12,735	330.95