Linear Time Analysis of Properties of Conflict-Free and General Petri Nets

Paola Alimonti[a], Esteban Feuerstein[b], Luigi Laura[a], Umberto Nanni[a]

[a]*Dipartimento di Informatica e Sistemistica "Antonio Ruberti", "Sapienza" University of Rome, via Ariosto 25, I-00185 Roma, Italy.*
[b]*Departamento de Computacion, Universidad de Buenos Aires, Pabellon I, Ciudad Universitaria, 1428 Buenos Aires, Argentina*

Abstract

We introduce the notion of *T-path* within Petri nets, and propose to adopt the model of *directed hypergraphs* in order to determine properties of nets; in particular, we study the relationships between T-paths and *firable sequences* of transitions. Let us consider a Petri net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ and the set of places with a positive marking in M_0, i.e., $P_0 = \{p \mid M_0(p) > 0\}$. If we regard the net as a *directed graph*, the existence of a simple path from any place in P_0 to a transition t is, of course, a necessary condition for the potential firability of t. This is sufficient only if the net is a *State Machine*, where $|{}^\bullet t| = |t^\bullet| = 1$ for all $t \in T$. In this paper we show that the existence of a T-path from any subset of P_0 to a transition t is a more restrictive condition and is, again, a necessary condition for the potential firability of t. But, in this case: (a) if $\mathcal{P}$ is a Conflict Free Petri net, this is also a *sufficient* condition, (b) if $\mathcal{P}$ is a general Petri net, t is potentially firable by *increasing the number of tokens in* P_0.

For *Conflict-Free* nets (*CFPN*) we consider the following problems: (a) determining the set of firable transitions, (b) determining the set of coverable places, (c) determining the set of live transitions, (d) deciding the boundedness of the net. For all these problems we provide algorithms requiring linear space and time, i.e., $O(|P|+|T|+|A|)$, for a net $\mathcal{P} = \langle P, T, A, M_0 \rangle$. Previous results for this class of networks are given by Howell, Rosier and Yen [20], providing algorithms for solving problems in *Conflict-Free* nets in $O(|P| \times |T|)$ time and space.

Given a Petri net and a marking M, the well-known *coverability* problem consists in finding a reachable marking M' such that $M' \geq M$; this problem is known to be EXPSPACE-hard [33]. For general Petri nets we provide a partial answer to this problem. M is *coverable by augmentation* if it is coverable from an *augmented marking* M'_0 of the initial marking M_0: $M'_0 \geq M_0$ and, for all $p \in P$, $M'_0(p) = 0$ if $M_0(p) = 0$. We solve this problem in linear time.

The algorithms to compute T-paths are *incremental*: it is possible to modify the network (adding new places, transitions, arcs, tokens), and *update* the set of potentially firable transitions and coverable places without recomputing them from scratch. This feature is meaningful when used during the interactive design of a system.

Keywords: Petri nets, conflict-free Petri nets, Marked Graphs, coverability, liveness, boundedness, directed hypergraphs, incremental algorithms

Email addresses: `palimonti@dis.uniroma1.it` (Paola Alimonti), `efeuerst@dc.uba.ar` (Esteban Feuerstein), `laura@dis.uniroma1.it` (Luigi Laura), `nanni@dis.uniroma1.it` (Umberto Nanni)
URL: `www.dc.uba.ar/people/profesores/feuerstein` (Esteban Feuerstein), `www.dis.uniroma1.it/~laura` (Luigi Laura), `www.dis.uniroma1.it/~nanni` (Umberto Nanni)

1. Introduction

Petri nets [31] are used to model basic properties of concurrent systems and to analyze their behaviour. Petri nets have been used to capture the basic properties of systems in an increasing number of areas, including communication protocols, flexible manufacturing, workflow management, chemical reaction networks, and biological systems. In Section 2 we provide basic definitions and properties about Petri nets.

The analysis of Petri nets has proven to be a challenging task for the research community. In general Petri nets, though less expressive than Turing machines (if we do not allow enhancements, like inhibitor arcs, or time constraints), some problems turn out to be undecidable, such as the containment (or equivalence) of the reachability set of two distinct nets (see, e.g., [30]). Other problems have been proven to be decidable, such as *reachability* [28] (deciding whether a given distribution of tokens can be reached), *liveness* [18, 28] (the property that all transitions in a net can fire infinitely many times), *boundedness* [24] (deciding whether all places have a bounded number of tokens in every reachable marking). Nevertheless, the computational complexity is often far from being practical: for many of these problems finding a solution in a general net requires at least exponential space [27]. For example the problems of deciding whether a single transition is potentially firable or whether a single place is coverable by some token requires exponential space to be solved, since the *coverability* problem (given a marking M, is it possible to reach a supermarking $M' \geq M$?) that requires exponential space [33], in general nets can be reduced to either of them. Due to the wide interest for Petri nets, the study of this model has considered both specific subclasses of nets, and weaker problem formulations providing at least partial answers to basic questions.

Several classes of Petri nets have been studied which, although lacking the expressive power of the general model, still can be used in order to describe interesting concurrent systems, and whose properties can be decided more efficiently than for arbitrary nets (see, e.g., [23, 14] for surveys of results).

A situation of *conflict* in a Petri net arises when the firing of a transition may disable another transition. In a *persistent* Petri net, there is no conflict in any reachable marking, but deciding persistence of a Petri net is PSPACE hard [9]. In a *Conflict-Free* Petri net [13, 26] (*CF* net) conflicts are avoided by a structural constraint, therefore *CF* nets are persistent for any initial marking. *CF* nets lack the nondeterminism of the general Petri nets, but can still be used to model certain kinds of distributed systems. For example, *CF* nets are equivalent to the control of decision-free flow-chart schemata first studied in [24]. A massive research activity and many monographs have been devoted to the application of Petri nets in *Flexible Manufactoring Systems (FMS)*. The *Marked Graphs* [11], i.e., Petri nets where $|^\bullet p| = |p^\bullet| = 1$ for all p, are a subset of *CF*-nets, and have been widely adopted to model *FMS* (see, e.g., [39]).

Deciding reachability in a *CF* net is still NP-complete [23]. Howell et al. show that both liveness [19] and boundedness [20, 19] for this class of nets can be decided in polynomial time. In particular Howell, Rosier, and Yen [20] provide algorithms for solving boundedness in conflict-free *Vector Replacement Systems*, a model equivalent to Petri nets, which require $O(n^{1.5})$ time for a VRS with total size n, and $O(|P| \times |T|)$ time and space for a net $\mathcal{P} = \langle P, T, A, M_0 \rangle$.

In this paper we propose to adopt strategies and algorithms devised for the model of *directed hypergraphs* [2, 16] as an effective computational framework, useful to determine structural properties of a Petri net. We introduce the purely structural concept of *T-path* within a Petri net, and relate this on the one hand to the notion of *hyperpath* in directed hypergraphs, and on the other hand to a firable sequence of transitions in Petri nets. Based on efficient algorithmic strategies that have been devised for directed hypergraphs [5, 16, 4, 3, 37], checking the (non)-existence of a T-path provides an answer in linear time to reachability questions both for the class of *Conflict-Free* Petri netsand, in some extent, for general Petri nets as well.

Namely, we consider the following problems on Conflict Free Petri nets:

a) determining the set of firable transitions: a transition t is *potentially firable* if there exists a marking $M \in R(M_0)$ such that t is enabled in M;

b) determining the set of coverable places: a place p is *coverable* if there exists a marking $M \in R(M_0)$ such that $M(p) > 0$;

c) determining the set of live transitions: a transition is *live* if it is potentially firable in every reachable markings;

d) deciding the boundedness of the net: a net is *bounded* if there exists a constant k such that, for each reachable marking M and each place $p \in P$, $M(p) \leq k$.

For all these problems we provide algorithms requiring linear space and time, i.e., $O(|P| + |T| + |A|)$, for a *CF* net $\mathcal{P} = \langle P, T, A, M_0 \rangle$. As remarked before, previous algorithms for these problems were proposed in [20, 19], and require $O(|P| \times |T|)$ space and time.

Let us consider a net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ and the set of places with a positive marking in M_0, i.e., $P_0 = \{p | M_0(p) > 0\}$. Let us call *augmented marking* (or *augmentation*) M_0^+ of M_0 any marking such that $M_0^+ \geq M_0$ and, for all $p \in P$, $M_0^+(p) = 0$ if $M_0(p) = 0$.

As claimed above, the existence of a T-path from any subset of P_0 to a transition t is a necessary and sufficent condition for the potential firability of t in a *CF* net. Focusing on *general* Petri nets, we have the following results. The existence of a T-path from any subset of P_0 to any transition t (or to any place p):

a) is a *necessary* condition for the potential firability of transition t (or for the coverability of place p);

b) is a *necessary and sufficient* condition for the potential firability by augmentation of transition t (or for the coverability by augmentation of place p).

In other words, if there is a T-path from P_0 to any node x (transition or place) in the net, then node x can be reached by a firable sequence by *increasing the number of tokens* in the initial marking.

Furthermore, as a consequence of result (b) above, we provide an answer to the problem of *coverability by augmentation*: given a Petri net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ and a target marking M, determine whether a supermarking $M' \geq M$ is reachable in $\mathcal{P}$ from some augmentation M_0^+ of the initial marking M_0. Note that the coverability problem is EXPSPACE hard. We can get an answer to coverability by augmentation in linear time.

The algorithms proposed for finding T-paths, and then to compute the set of coverable places and firable transitions are *incremental*: it is possible to incrementally modify the network (adding new places, transitions, arcs, tokens), and *update* the set of coverable places and potentially firable transitions without recomputing the solution from scratch. This feature is meaningful when used during the interactive design of a system. The total time and space requirements to handle a sequence ω of incremental operations of these kinds are bounded by $O(|\omega| + |P| + |T| + |A|)$.

The rest of this paper is organized as follows: after the presentation of the basic terminology and the considered problems in Section 2, in Section 3 we detail the properties of *CF* nets. Our approach based on T-paths is introduced in Section 4, and applied to coverability of places and firability of transitions in Section 5. Then, the problems of liveness and boundedness are considered in Sections 6 and 7, respectively.

2. Basic definitions

In the following we give the basic definitions and notations about Petri nets [30, 34].

An *unmarked Petri net* is a 3-tuple $\hat{\mathcal{P}} = \langle P, T, A \rangle$, where P is a finite set of *places*, T is a finite set of *transitions*, $A \subseteq (P \times T) \cup (T \times P)$ is a finite set of *arcs*. We consider the *ordinary* Petri nets, i.e., all arcs have unitary weight, and no multiple arcs are allowed.

If t is a transition in T, the two sets ${}^\bullet t = \{p \mid (p,t) \in A\}$ and $t^\bullet = \{p \mid (t,p) \in A\}$ are respectively the *input set* and the *output set* of t. Notation and terminology are extended to places: ${}^\bullet p = \{t \mid (t,p) \in A\}$ and $p^\bullet = \{t \mid (p,t) \in A\}$. We use the same notation also for sets as well: if X is either a set of places or a set of transitions, then ${}^\bullet X = \bigcup_{x \in X} {}^\bullet x$ is the union of its input sets, and $X^\bullet = \bigcup_{x \in X} x^\bullet$ is the union of its output sets.

The status of a Petri net is modeled by using *tokens*; a given distribution of tokens on places is called *marking* of the net, i.e., a function $M : P \to \mathcal{N}$, where $\mathcal{N}$ is the set of natural numbers, such that, for each place p, the quantity $M(p)$ is the number of tokens in p. Given a marking M, when no ambiguity arises we may refer to M as the set $\{p \mid M(p) > 0\}$, i.e., as the set of places with a positive marking. As an example, we will write $p \in M$ to denote that, for the given place $p \in P$, $M(p) \neq 0$.

A *(marked) Petri net* is a 4-tuple $\mathcal{P} = \langle P, T, A, M_0 \rangle$, where $M_0 : P \to \mathcal{N}$ is the *initial marking*. An example is shown in Figure 1.

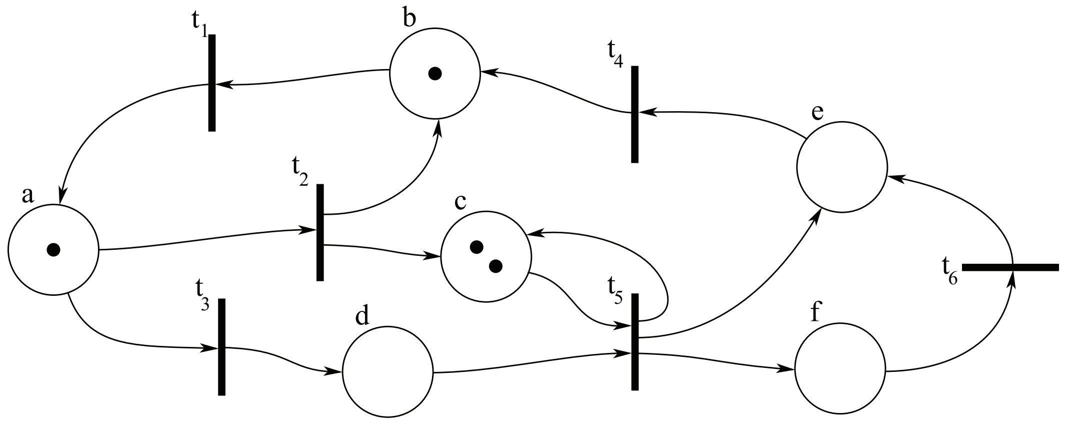

Figure 1: A Petri net $\mathcal{P} = \langle P, T, A, M_0 \rangle$. Places ($P$) are represented by circles, transitions (T) by bold segments, and the initial marking $M_0 : P \to \mathcal{N}$ as a distribution of tokens (represented as dots); in particular, $M_0(c) = 2$, $M_0(a) = M_0(b) = 1$, and $M_0(d) = M_0(e) = M_0(f) = 0$, i.e., $P_0 = \{a, b, c\}$.

The dynamic behavior of a net is due to transitions, which allow the net to change its state in the following way: a transition t is said to be *enabled* when each place in its input set ${}^\bullet t$ contains at least one token. If a transition is enabled, it can *fire*: this implies the removal of one token from each place in ${}^\bullet t$ and the introduction of one additional token in each place in the output set $t^\bullet$.

For any marking M and any $t \in T$, we write $M \xrightarrow{t}$ to denote that t is enabled on marking M. Furthermore $M \xrightarrow{t} M'$ means that the firing of t in M yields the marking M'. Following a common notation in this area, when no ambiguity arises, we may refer to the marking $M \xrightarrow{t}$

as the marking resulting from the firing of transition t, which must enabled in M. The notation is extended to a sequence of transitions $\sigma = \langle t_1, t_2, \ldots, t_n \rangle \in T^*$, called *firing sequence*: $M \xrightarrow{\sigma}$ is a shorthand for $M \xrightarrow{t_1} M_1 \xrightarrow{t_2} M_2 \ldots M_{n-1} \xrightarrow{t_n}$. As an example, in the net in Figure 1, the enabled transitions are: t_1, t_2. A firing sequence is: $\langle t_3, t_5, t_6, t_1, t_4, t_4, t_1 \rangle$.

The *set of reachable markings* or *reachability set* of a a Petri net $\mathcal{P} = \langle P, T, A, M_0 \rangle$, is the set $R(M_0) = \{M \mid \text{there exists a sequence } \sigma \in T^* \text{ such that } M_0 \xrightarrow{\sigma} M\}$.

A transition t is said to be *potentially firable in a marking* M if there exists a marking $M' \in R(M)$ such that $M' \xrightarrow{t}$. A transition t is said to be *potentially firable* if it is potentially firable in M_0, otherwise transition t is said to be *dead*. Analogously, a sequence σ is *potentially firable* if there exists a marking $M \in R(M_0)$ such that $M \xrightarrow{\sigma}$. As an example, in the net in Figure 1, all transitions are potentially firable.

A transition $t \in T$ is said to be *live* if it is potentially firable in every $M \in R(M_0)$[1]. A net $\mathcal{P}$ is said to be *live* if every transition $t \in T$ is live in $\mathcal{P}$.

A place p is *coverable* if there exists a marking $M \in R(M_0)$ such that $M(p) \neq 0$. We use the following notation to compare markings in a Petri net. A marking M *covers* a marking M', written $M \geq M'$, if for every place $p \in P$, $M(p) \geq M'(p)$. Furthermore the covering is *proper*, written $M > M'$, if $M \geq M'$, and there exists a place $p \in P$ such that $M(p) > M'(p)$.

A place $p \in P$ is said to be *bounded*, if there exists a constant k such that, for any $M \in R(M_0)$, $M(p) \leq k$, otherwise it is *unbounded*. A Petri net is said to be *bounded* if every place $p \in P$ is bounded.

If we are given a Petri net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ and ignore the initial marking, we get a *bipartite directed graph* that we refer to as the *unmarked net* $\hat{\mathcal{P}} = \langle P, T, A \rangle$.

2.1. Connectivity and Strong Connectivity in Graphs

Here we provide few definitions about directed and undirected graphs. We recall that a *graph* is a pair $G = (V, E)$ of sets such that $E \subseteq \binom{V}{2}$. Elements of V are the *vertices*, or *nodes*, of the graph, whilst elements of E are its *edges*. If there is an edge $e = (x_1, x_2)$, we say that x_1 and x_2 are connected.

A *path* is a non-empty graph $P = (V, E)$ such that $V = \{x_0, x_1, \ldots, x_n\}$, $E = \{(x_0, x_1), (x_1, x_2), \ldots, (x_{n-1}, x_n)\}$; a *simple path* is a path where all the x_i are distinct.

A graph is said to be *connected* if any two of its vertices are linked by a path. If a graph is not connected, then its maximal connected subgraphs are said to be its *connected components*.

A *directed graph*, or *digraph*, is a pair $D = (V, A)$ of sets such that $A \subseteq V^2$. As before, the elements of V are the *vertices*, or *nodes*, of the graph, whilst elements of A are its *arcs*, or *directed edges*, or *oriented edges*. If there is an arc $a = x_1 \to x_2$, x_1 can reach x_2, but not necessarily vice-versa.

A *directed path* is a non-empty graph $P_D = (V, A)$ such that $V = \{x_0, x_1, \ldots, x_n\}$, $A = \{x_0 \to x_1, x_1 \to x_2, \ldots, x_{n-1} \to x_n\}$, and all the x_i are distinct.

A directed graph is said to be *strongly connected*, if there is a directed path from any of its vertices to any other vertex. If a graph is not strongly connected, then its maximal strongly connected subgraph are said to be its *strongly connected components*.

In Figure 2, it is shown an example of a directed graph (left) and its undirected version. It is possible to observe that the directed version is not strongly connected, but it has three strongly

[1]A consequence of this definition is that a live transition is firable infinitely many times from any marking $M \in R(M_0)$. Beside this notion, also named *l4-liveness*, other definitions of liveness have been considered in the literature (see, e.g., [17]).

connected components (SCCs), respectively the sets $\{A, B, C, D\}$, $\{E, F, G\}$, and $\{H, I, J, K\}$. Furthermore, its undirected version is not connected, but it has only two connected components (CCs), respectively the sets $\{A, B, C, D\}$ and $\{E, F, G, H, I, J, K\}$. We will use the number of SCCs of a directed graph, together with the number of CCs of its undirected version, to characterize the boundedness of a *CF* network (Section 7).

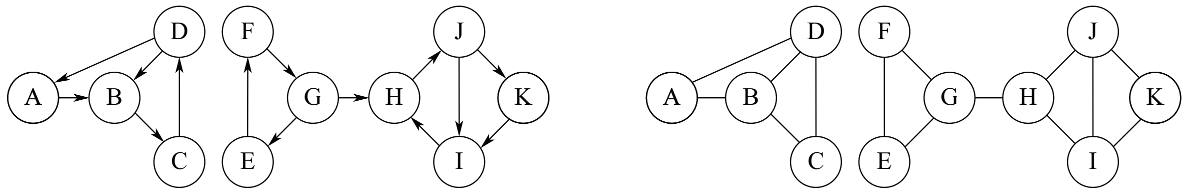

Figure 2: A directed graph (left) and its undirected version (right)

3. Conflict-free Petri nets

Many basic problems on Petri nets have been proven to be intractable, such as *reachability* or *boundedness*, or even undecidable. For this reason, much attention has been devoted to defining subclasses of Petri nets that, although lacking the expressive power of the general model, capture interesting classes of concurrent systems, and allow efficient algorithms to be devised for their analysis (see, e.g., [13, 23, 26, 20, 15]).

The behavior of a Petri net is computationally hard due to the intrinsic nondeterminism associated to the (enabled) transitions: when a transition fires, this might disable some other transition. A situation of *conflict* arises when an enabled transition may be disabled by the firing of another one (see Figure 3).

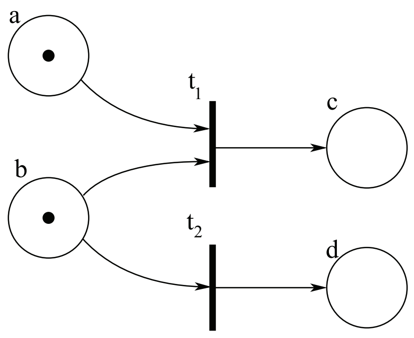

Figure 3: A situation of conflict: t_1 and t_2 are both enabled, but the firing of either of them would disable the other one.

A net is *persistent* if there is no conflict in any reachable marking. Therefore this is a *behavioural* property, i.e., it depends on the initial marking. As for many problems of this kind, deciding if a Petri net is persistent is PSPACE hard [9].

A more restricted class of nets are the *Conflict-Free Petri nets* [24, 13, 26]. These class of nets, that are persistent for each possible initial marking, are defined by means of a *structural* property that can be checked in linear time. A simple example is shown in Figure 4.

Definition 3.1. *A Petri net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ is* conflict-free *(CF) if each place $p \in P$ satisfies one of the following:*

a) $|p^\bullet| \leq 1$*, i.e., there is at most one arc leaving* p *(in this case* p *is said* unbranched*);*

b) $|p^\bullet| > 1$ *and, for each transition* $t \in p^\bullet$*,* $p \in t^\bullet$ *(in this case* p *is said* branched*).*

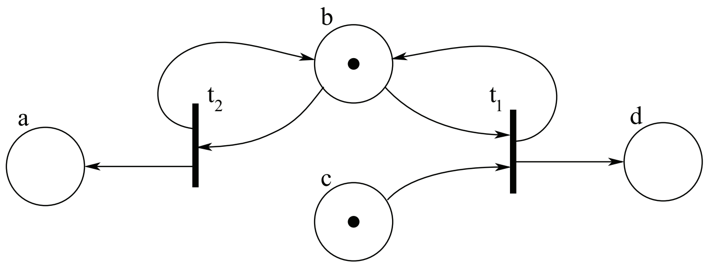

Figure 4: A Conflict Free net. Place b is branched; the remaining places are unbranched

Some consequences of this definition are formally stated in the following. The first lemma recalls a basic property of persistent nets; a comprehensive discussion of this property, related to Keller's theorem [25], can be found, e.g., in Best [7].

Lemma 3.1. *In a CF net, if a transition* t *is enabled in a given marking* M*, then for any marking* M' *such that* $M \xrightarrow{\sigma} M'$*, either* $t \in \sigma$*, or* t *is still enabled in* M'*.*

Proof. Let us suppose, by contradiction, that $t \notin \sigma$ and t is not enabled in M'. Hence, some place $p \in {}^\bullet t$ has lost a token due to the firing of a transition $\bar{t}$ in σ, with $p \in {}^\bullet\bar{t}$ and $\bar{t} \neq t$. From definition 3.1, this is not possible since either *a)* p is unbranched, then p cannot be in the input set of both t and $\bar{t}$, or *b)* p is branched, then $p \in {}^\bullet\bar{t} \cap \bar{t}^\bullet$, and the token cannot leave place p due to transition $\bar{t}$. □

The next lemma states a basic property of *CF* nets that is exploited in our algorithms; it is a consequence of a property proven by Howell, Rosier and Yen [20], Lemma 3.1, that, for our purposes and with our notations, can be restated as follows: *in any CF net* $\mathcal{P}$ *there exists a firable sequence that contains exactly once all the potentially firable transitions in* $\mathcal{P}$.

Lemma 3.2. *In a CF net* $\mathcal{P}$*, a transition* t *is potentially firable if and only if, for each* $p \in {}^\bullet t$*,* p *is coverable in* $\mathcal{P}$*.*

Proof. The "only if" case is trivial: in order to fire transition t, then every place in its input set must be coverable in $\mathcal{P}$.

To prove the nontrivial side of the lemma (i.e., the "if" case), let us suppose that, for a given $t \in T$, any place $p \in {}^\bullet t$ is coverable, and let σ_F be a firable sequence that contains exactly once all the potentially firable transitions in $\mathcal{P}$. Starting with the initial marking and firing sequence σ_F supplies at least one token to each coverable place, and then to each $p \in {}^\bullet t$. For each place p in this set:

- if p is unbranched, t is the only transition in $p^\bullet$, then the token will remain there if t does not fire;
- if p is branched, the token will remain there forever.

Therefore, after firing the longest prefix of σ_F not containing transition t, the input set ${}^\bullet t$ is covered and t is enabled, hence t is potentially firable. □

The following lemma, exploited by our algorithms, states that, in order to determine the set of firable transitions and coverable places, it is possible to consider the set of marked places, disregarding the number of tokens in each place.

Lemma 3.3. *In a CF net, if a transition t is potentially firable in a marking M (if a place p is coverable in M), then t is potentially firable (p is coverable) in any marking M' such that for each place p, if $M(p) > 0$ then $M'(p) > 0$.*

Proof. By contradiction, let us consider a sequence of transitions which is firable in M and containing at least one transition which is not potentially firable in M' (by any sequence). Namely, let $\bar{t}$ be the first transition that is not potentially firable in M', and σ be the sequence enabling such transition from M (note that σ might be not firable in M'). We can summarize the current hypotheses by contradiction:
(1) transition $\bar{t}$ is potentially firable in M, and enabled in the marking $M \xrightarrow{\sigma}$;
(2) transition $\bar{t}$ is not potentially firable in M';
(3) all transitions in σ are potentially firable in M'.
Due to (1), each place $p \in {}^\bullet\bar{t}$ is coverable from M, and either it is marked initially, i.e. $M(p) > 0$, or it is in the output set of some transition t_p in σ. In both cases, p is coverable in M', as well, because either $M'(p) > 0$ (by hypothesis), or it is marked by any sequence that fires t_p, which is potentially firable in M', due to hypothesis (3).

Since each $p \in {}^\bullet\bar{t}$ is coverable in M', then, by lemma 3.2, $\bar{t}$ is potentially firable in M', contradicting (2). □

4. *T*-paths in Petri nets and related problems

In this section we introduce the notion of *T-path* in Petri nets. This concept captures some structural properties of the net: finding T-paths allows various problems to be answered very efficiently for *CF*, and in some cases also for general nets.

Different notions of "paths" have been considered to deal with properties of Petri nets. Yen [38] proposes *Petri net paths* as a witness to prove the satisfiability of a formula which is related to a firable sequence: the reduction employed by Yen is useful to prove in a uniform framework an exponential space upper bound for a number of problems on Petri nets; this result (i.e. Yen [38]) follows the work by Rackoff [33] (about covering and boundedness), and by Howell, Rosier and Yen [21] (relating reachability and *fair nontermination*).

We now introduce the notion of T-paths, and show what information can be derived both for *CF* nets and for general Petri nets. The algorithms to find T-paths are provided in Section 5.

4.1. T-paths in Petri nets

Definition 4.1. *Let $\hat{\mathcal{P}} = \langle P, T, A\rangle$ be an unmarked Petri net, and $P_M \subseteq P$ be a nonempty set of places. A* T-path *from P_M to a place $p \in P$ is a (possibly empty) set $\tau(P_M, p) \subseteq T$ of transitions such that one of the following conditions holds:*

- extended reflexivity*: $p \in P_M$; in this case $\tau(P_M, p)$ is empty;*
- extended transitivity*: there exists a transition t such that $p \in t^\bullet$, and there exists a T-path (see below) $\tau(P_M, t)$ from P_M to t. In this case:*

$$\tau(P_M, p) = \tau(P_M, t).$$

Analogously, a T-path *from a set of places $P_M \subseteq P$ to a transition $t \in T$ is a nonempty set $\tau(P_M, t) \subseteq T$ of transitions such that one of the following conditions holds:*

- extended reflexivity*: $^\bullet t \subseteq P_M$ (including the case $|^\bullet t| = 0$); in this case $\tau(P_M, t) = \{t\}$;*
- extended transitivity*: for each $p \in {}^\bullet t$ there exists a T-path $\tau(P_M, p)$ from P_M to p; in this case:*
$$\tau(P_M, t) = \bigcup_{p \in {}^\bullet t} \tau(P_M, p) \ \cup \ \{t\}.$$

Note that if a transition t has an empty input set, i.e., $^\bullet t = \emptyset$, then for any place $p \in t^\bullet$ by extended transitivity there exists a nonempty T-path from any set of places $P_M \subseteq P$ to p. We also remark that, given a set of places P_M and a transition t such that a T-path $\tau(P_M, t)$ exists, then for any place $p \in t^\bullet$ there exists a T-path $\tau(P_M, p) = \tau(P_M, t)$. An example of T-path is shown in Figure 5.

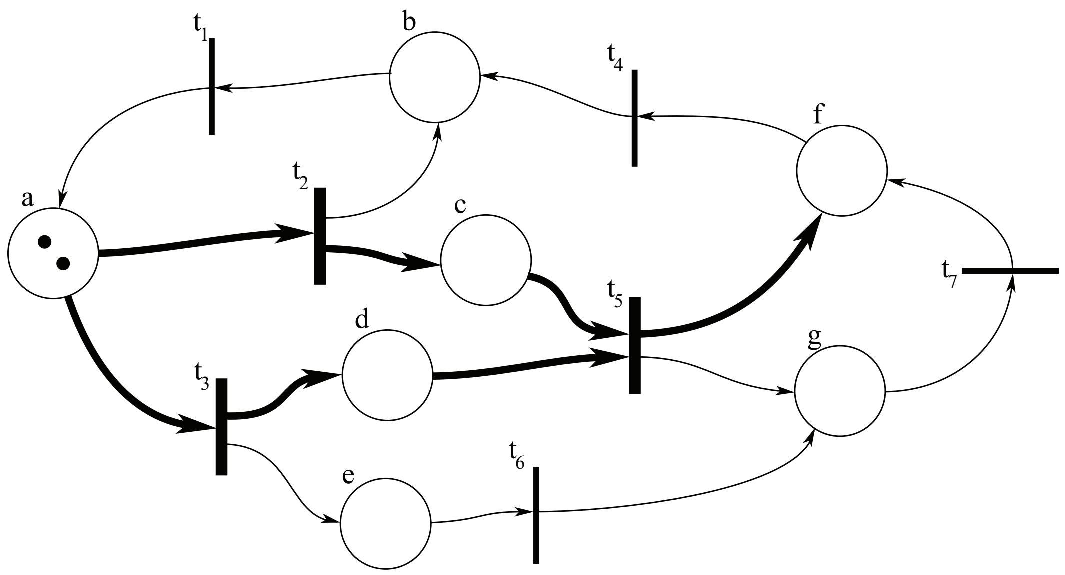

Figure 5: Bold transitions define a T-path $\tau(a, f) = \tau(a, c) \cup \tau(a, d) \cup \{t_5\} = \{t_2, t_3, t_5\}$. Relevant arcs involved in the definition (i.e., those that propagate extended transitivity) are highlighed in bold.

Our interest for T-paths is due to the relationships between T-paths and reachable markings. Let us consider a Petri net $\mathcal{P}$, with initial marking M_0, and let $P_0 = \{p \mid M_0(p) > 0\}$ be the set of initially marked places. In the following we prove that:
- if $\mathcal{P}$ is a Conflict Free net the the existence of a T-path from P_0 to a transition t (i.e., its T-reachability) is a necessary and sufficient condition for the potential firability of t;
- if $\mathcal{P}$ is a general Petri net, the T-reachability of t is (a) a necessary condition for its potential firability, and (b) a necessary and sufficient condition for its potential firability by *augmenting* the number of tokens in the initial marking (a concept that will be formally stated in section 4.3).

Analogous properties hold for coverability of places.

T-path is defined as a *set* of transitions subject to certain conditions, whilst in Petri nets we have an interpretation for *sequences* of transitions. Hence we have that a T-path underspecifies the corresponding firing sequence. On the other side, one of the traditional tools in order to compute and investigate effectively the impact of a firing sequence on the marking of a Petri net is based on a "relaxed" description of a firing sequence as a set (actually, a multiset). For a sequence $\sigma \in T^*$ of transitions, its *Parikh vector* [7] (also named *firing count vector* by other authors [29, 10]) $\Psi(\sigma)$ is a vector of natural numbers, such that $\Psi(\sigma)[t]$ is the number of occurrences of t in σ. Based on this definition, any permutation of a given firing sequence has the same Parikh vector. In the case of persistent nets, the Parikh-equivalence is studied in Best [7]: if σ, θ are two sequences which are firable in a reachable marking M, if $\Psi(\theta) = \Psi(\sigma)$, then the two final markings $M \xrightarrow{\theta}$ and $M \xrightarrow{\sigma}$ are identical.

We show how to build up a firable sequence of transitions corresponding to a T-path. If we are given a net with initial marking M_0, a T-path τ has a recursive definition which defines a partial order "$\prec_\tau$" among the transitions in the set τ: if $\tau(P_0, t) = \bigcup_{p \in {}^\bullet t} \tau(P_0, p) \ \cup \ \{t\}$ then $\bar{t} \prec_\tau t$, for all $\bar{t} \in \bigcup_{p \in {}^\bullet t} \tau(P_0, p)$. Note that this ordering is coherent with the recursive definition of T-path from the target node back to the initial marking. If we complete this a partial order "$\prec_\tau$" with any total order complying it, we get a sequence of transitions $\sigma_\tau = \langle t_1, t_2, \ldots, t_k \rangle$: in turn, this sequence defines a total order "$<_\sigma$" such that, if $t_i \prec_\tau t_j$, then $t_i <_\sigma t_j$. A prof of the firability of such a sequence is provided in Theorem 4.3.

4.2. T-path and Syphons

A *siphon* [29] is a set S of places that cannot gain tokens, since any input transition of S is also an output transition of S: ${}^\bullet S \subseteq S^\bullet$. A *trap* is a set of places that remain marked once they have gained at least one token, i.e. a trap is a set Z such that $Z^\bullet \subseteq {}^\bullet Z$ (some examples are shown in Figure 6). Siphons have been widely used as key tools for determining properties of a net, in particular, to analyze deadlocks [22] or for deadlock prevention strategies [32]. As an example, for any dead net, i.e., when no transition is enabled, the set of unmarked places is a siphon. A sufficient condition for deadlock freeness [10] is the so called *Commoner condition*: every siphon contains a trap marked by the initial marking M_0.

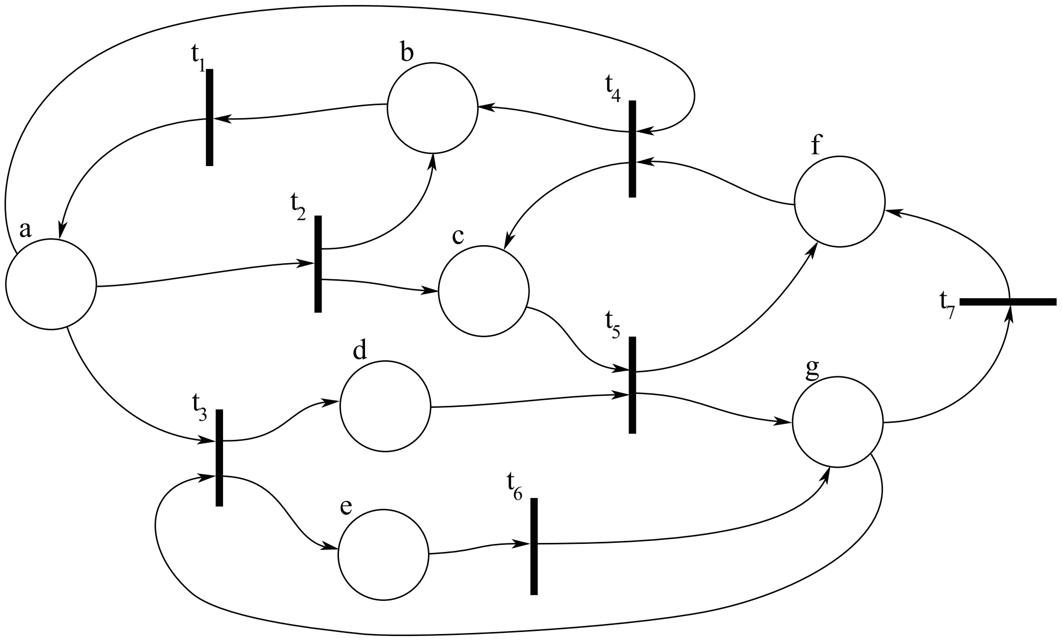

Figure 6: A set of places S is a siphon if ${}^\bullet S \subseteq S^\bullet$; examples are $\{a, b\}$, $\{d, e, g\}$, and P. A set of places Z is a trap if $Z^\bullet \subseteq {}^\bullet Z$; examples are $\{c, f\}$, $\{c, d, e, f, g\}$ and P.

The connection between T-path reachability and siphons is stated in the following lemma.

Lemma 4.1. *Let us consider an unmarked Petri net $\hat{\mathcal{P}} = \langle P, T, A\rangle$ and any arbitrary marking M. The set of places S which are* not *T-reachable from M is a* siphon.

Proof. By hypothesis, all places in $P - S$ are T-reachable from M, and all places in S are not. We need to prove that each transition with an output place in S has at least one input place in S.

Let us consider a generic transition $t \in {}^\bullet p$, with $p \in S$. If, by contradiction, all places in ${}^\bullet t$ would be T-reachable, then there would exist a T-path from M to to t, and hence to all places in $t^\bullet$, including p: in this case p would be T-reachable. So, at least one place $p_u \in {}^\bullet t$ is not T-reachable, hence $p_u \in S$, and $t \in S^\bullet$.

Since this same argument applies to every $t \in {}^\bullet S$, we have that ${}^\bullet S \subseteq S^\bullet$, i.e., S is a siphon. $\square$

The following property may be considered as an alternative definition of siphon.

Lemma 4.2. *Given an unmarked Petri net $\hat{\mathcal{P}} = \langle P, T, A\rangle$, a set of places $S \subseteq P$ is a* siphon *if and only if no place in S is T-reachable from $P - S$.*

Proof.
($\Longrightarrow$) By hypothesis ${}^\bullet S \subseteq S^\bullet$, that is, each transition with an output place in S has at least one input place in S. By contradiction, let us assume that there exists a place $p \in S$ which is T-reachable from $P - S$. As a consequence, by definition 4.1, there exists a transition $t \in {}^\bullet p$ and a T-path $\tau(P - S, t)$ such that all places in ${}^\bullet t$ are T-reachable from $P - S$. In this case ${}^\bullet t \cap S = \emptyset$, and we have a contradiction.

($\Longleftarrow$) Note that, by definition 4.1, each place in $P - S$ is T-reachable from $P - S$ (by extended reflexivity).

If, by hypothesis, no place in S is T-reachable from $P - S$, then S is exactly the set of places which are not T-reachable from $P - S$. Hence, we have a special case of Lemma 4.1 and conclude that S is a siphon. $\square$

4.3. T-path reachability and Coverability by Augmentation

Let us consider a general (i.e., not necessarily Conflict-Free) Petri net $\mathcal{P} = \langle P, T, A, M_0\rangle$. For any given marking M, let us consider the set of *augmented markings* $(M)^+$ obtained from M by possibly adding tokens in any place p where $M(p) > 0$:

$$(M)^+ = \{M' \mid \text{ for any } p \in P\text{: } M'(p) \geq M(p), \text{ and } M'(p) = 0 \text{ if and only if } M(p) = 0\}.$$

Analogously, we can consider the set of nets $(\mathcal{P})^+$ having the same structure as $\mathcal{P}$, but an augmented initial marking:

$$(\mathcal{P})^+ = \{\langle P, T, A, M_0'\rangle \mid M_0' \in (M_0)^+\}.$$

Theorem 4.3. *Let $\mathcal{P} = \langle P, T, A, M_0\rangle$ be a general Petri net, and $P_0 = \{p \mid M_0(p) > 0\}$.*

If there exists a T-path $\tau(P_0, t)$ from P_0 to the transition $t \in T$, then transition t is potentially firable in some net $\mathcal{P}' \in (\mathcal{P})^+$.

If there exists a T-path $\tau(P_0, p)$ from P_0 to the place $p \in P$, then place p is coverable in some net $\mathcal{P}' \in (\mathcal{P})^+$.

Proof. We prove the theorem by induction on the structure of the T-path, following the cases of definition 4.1.

Basis.

- *Extended reflexivity for places*: if we consider any place $p \in P_0$, there exists an empty T-path $\tau(P_0, p)$ from P_0 to p. Note that in this case p is (trivially) coverable from M_0 by an empty sequence.
- *Extended reflexivity for transitions*: if we consider any transition t such that ${}^\bullet t \subseteq P_0$, there exists a T-path $\tau(P_0, t) = \{t\}$. In this case t is firable in M_0: $M_0 \xrightarrow{t}$.

Inductive step.

- *Extended transitivity for places*: let us suppose that, for a place $p \in P$, there exists a transition $t \in p^\bullet$ and a T-path from P_0 to t: $\tau(P_0, p) = \tau(P_0, t)$. If, by inductive hypothesis, transition t is potentially firable in some net $(\mathcal{P})^+$, then place p is coverable in $(\mathcal{P})^+$.
- *Extended transitivity for transitions*: let us suppose that, for a transition $t \in T$, there exists a T-path from P_0 to t:
$$\tau(P_0, t) = \bigcup_{p \in {}^\bullet t} \tau(P_0, p) \ \cup \ \{t\}.$$
Let us suppose, by inductive hypothesis, that each place $p_i \in {}^\bullet t$ is coverable in some net $\mathcal{P}_i \in (\mathcal{P})^+$. We have to prove that, in this case, t is potentially firable in some net $\mathcal{P}' \in (\mathcal{P})^+$.
Let us consider the input set of transition t, i.e., ${}^\bullet t = \{p_1, p_2, \ldots, p_k\}$. Each net $\mathcal{P}_i$, where p_i is coverable by a firing sequence σ_i, has a suitable initial marking $M_{0,i} \in (M_0)^+$; for $i = 1, 2, \ldots, k$ we have that $M_{0,i} \xrightarrow{\sigma_i} M'_i$, with $M'_i(p_i) > 0$.
Let us consider:
  - the initial marking: $M'_0 = M_{0,1} + M_{0,2} + \ldots + M_{0,k}$
  - the firing sequence: $\sigma' = \sigma_1 \cdot \sigma_2 \cdot \ldots \cdot \sigma_k$

  In order to prove the theorem, we claim that sequence σ' is firable in the net $\mathcal{P}' = \langle P, T, A, M'_0 \rangle$, and it enables transition t.
Let us "colour" the tokens of marking M'_0 by the colors $c_1, c_2, \ldots, c_k$ coherently with the number of tokens in the markings $M_{0,1}, M_{0,2}, \ldots, M_{0,k}$. Let us consider the firing of σ' split in k "stages", where stage i, consists in firing the sequence σ_i while propagating only the tokens with color c_i. Note that, in stage i, the marking of the net $\mathcal{P}'$ restricted to tokens of color c_i is identical to the marking of the net $\mathcal{P}_i$ while firing sequence σ_i.
After firing σ', each place $p_i \in {}^\bullet t$ is marked (at least) with a token with color c_i. Hence transition t is enabled.

□

On the other hand, in a general Petri net $\mathcal{P} = \langle P, T, A, M_0 \rangle$, the existence of a T-path from P_0 to a transition t is a *necessary* condition for the potential firability of transition t in $\mathcal{P}$.

Theorem 4.4. *Let $\mathcal{P} = \langle P, T, A, M_0 \rangle$ be a general Petri net, and $P_0 = \{p \mid M_0(p) > 0\}$. If a transition $t \in T$ is potentially firable, then in $\mathcal{P}$ there exists a T-path $\tau(P_0, t)$. Analogously, if a place $p \in P$ is coverable, then in $\mathcal{P}$ there exists a T-path $\tau(P_0, p)$.*

Proof. Let us consider a transition $t \in T$ and a firable sequence $\sigma_n = \langle t_1, t_2, \ldots, t_n \rangle$ which enables t, i.e., such that $M_0 \xrightarrow{\sigma_n} M_n \xrightarrow{t}$. We will prove that the arcs used in σ_n build up a T-path $\tau(P_0, t)$.

We proceed by induction on the length of sequence σ_n. Let us consider its prefix sequences $\sigma_i = \langle t_1, t_2, \ldots, t_i \rangle$, for $i = 1, 2, \ldots, n$, and let σ_0 be the empty sequence; furthermore, let us denote markings so that for any $i = 0, 1, 2, \ldots, n$: $M_0 \xrightarrow{\sigma_i} M_i$.

Basis. $|\sigma| = 0$. By hypothesis the empty sequence σ_0 enables t and then ${}^\bullet t \subseteq P_0$. In this case, by Definition 4.1, the set $\tau(P_0, t) = \{t\}$ is a T-path from P_0 to t in $\mathcal{P}$.

Inductive step. Let us suppose that for a given positive integer $k < n$, and for any nonnegative $j \le k$, if the prefix sequence $\sigma_j = \langle t_1, t_2, \ldots, t_j\rangle$ is firable in $\mathcal{P}$, then there exists a T-path $\tau(P_0, t_j)$ from P_0 to t_j. We will show that the hypothesis also holds for the firable sequence $\sigma_{k+1} = \sigma_k \cdot t_{k+1}$ with $M_0 \xrightarrow{\sigma_k} M_k \xrightarrow{t_{k+1}} M_{k+1}$.

By inductive hypothesis, t_{k+1} is firable in M_k and furthermore, for each $p_i \in M_k$, there exists in $\mathcal{P}$ a T-path $\tau(P_0, p_i)$ from P_0 to p_i. Hence, by Definition 4.1, a T-path from P_0 to t_{k+1} can be built by considering the set of transitions:

$$\tau(P_0, t_{k+1}) = \bigcup_{p_i \in {}^\bullet t_{k+1}} \tau(P_0, p_i) \,\cup \{t_{k+1}\}.$$

In case of a coverable place $p \in P$, either $p \in P_0$ (and the hypothesis is trivially true) or there exists a transition t which is potentially firable and such that $p \in t^\bullet$. Hence the problem is reduced to the above case. □

For general nets, from Theorems 4.3 and 4.4 we know that if no T-path from P_0 to a given transition t exists, then we can conclude that the structure of the net makes impossible to enable transition t starting from the given initial marking. On the contrary, if such a T-path $\tau(P_0, t)$ exists, we know that it is sufficient to *increase* the initial marking M_0 (by possibly adding tokens in places p such that $M_0(p) > 0$) to make t potentially firable; analogous considerations apply to coverability of places. Furthermore, we can deal with a relaxed version of the *coverability* problem.

Definition 4.2. *Let $\mathcal{P} = \langle P, T, A, M_0\rangle$ be a general Petri net, and M be any marking of $\mathcal{P}$. Marking M is* coverable by augmentation *if there exist two markings $M_0^+ \in (M_0)^+$ and $M' \in R(M_0^+)$ such that, for all $p \in P$, $M'(p) \ge M(p)$.*

In other words M is coverable by augmentation if and only if, starting from some augmented net $\mathcal{P}^+ \in (\mathcal{P})^+$, is it possible to reach a marking M' which *covers* M.

Theorem 4.5. *In a general net $\mathcal{P} = \langle P, T, A, M_0\rangle$ a marking M is coverable by augmentation if and only if, for each p such that $M(p) > 0$, there exists a T-path $\tau(P_0, p)$ in $\mathcal{P}$.*

Proof.

$(\Longrightarrow)$ Coverability by augmentation is a special case of coverability. As a consequence of the hypothesis, since marking M is coverable then, for each place p with $M(p) > 0$, p is coverable and, by Theorem 4.4, there exists a T-path from P_0 to p.

$(\Longleftarrow)$ Let us denote as P_M the set of places with a positive marking in M. By hypothesis for each $p_i \in P_M$, there exists a T-path in $\mathcal{P}$. Hence, by Theorem 4.3 we have that each p_i is coverable in some net $\mathcal{P}'_i \in (\mathcal{P}_i)^+ = \langle P, T, A, M_{0,i}\rangle$. Hence, by definition 4.2, each p_i is coverable by augmentation in by some sequence σ_i, which is firable in $M_{0,i} \in (M_0)^+$. Hence, for each $p_i \in P_M$, $M_{0,i} \xrightarrow{\sigma_i} M_{F,i}$.

We can apply similar arguments as in the proof of Theorem 4.3. If we consider an initial marking $M'_0 = M_{0,1} + M_{0,2} + \ldots + M_{0,k}$ and a firing sequence: $\sigma' = \sigma_1 \cdot \sigma_2 \cdot \ldots \cdot \sigma_k$ we have that $M'_0 \xrightarrow{\sigma'} M_F = M_{F,1} + M_{F,2} + \ldots + M_{F,k}$, with $M_{F,i}(p_i) > 0$. If, for $i = 1, 2, \ldots, k$ we color tokens from each $M_{0,i}$ with colour c_i, these tokens can be used while firing the subsequence σ_i.

Since $M_0' \in (M_0)^+$, σ' is firable in M_0', and the final marking M_F covers all places in P_M, then M is coverable by augmentation in case that, for each $p \in P$, $M(p) \leq 1$.

If, for some p_j, $M(p_j) > 1$, it is sufficient: *i.* to multiply the contribution of place p_j to the initial marking by a factor $M(p_j)$ and *ii.* to reply the firing of σ_j $M(p_j)$ times. In other words, we consider:

$M_0' = \sum_{p_i \in P_M} M_{0,1} * M(p_i)$;

$\sigma' = \sigma_1^{M(p_i)} \cdot \sigma_2^{M(p_2)} \cdot \ldots \cdot \sigma_k^{M(p_k)}$

In this situation, with similar arguments to the unitary marking above, we can verify that $M_0' \xrightarrow{\sigma'} M_F = M_{F,1} * M(p_1) + M_{F,2} * M(p_2) + \ldots + M_{F,k} * M(p_k)$, where each $M_{F,i}$ covers place p_i with at least one token. Hence, for all places in $p_i \in P_M$, we have that $M_F(p_i) \geq M(p_i)$, i.e., marking M is covered. □

In general Petri nets the problem of coverability by augmentation can be solved in linear time in terms of T-paths, as shown in the next section. The coverability problem, instead, requires exponential space [27].

4.4. T-paths in CF Petri nets

In the following, we fist prove that deciding the potential firability of a transition t (the coverability of a place p) in *CF* nets can be reduced to verifying the existence of a T-path $\tau(P_0, t)$ ($\tau(P_0, p)$) from the set P_0 of places in the initial marking to transition t (to place p).

Lemma 4.6. *Let $\mathcal{P} = \langle P, T, A, M_0 \rangle$ be a CF net, and $P_0 = \{p \mid M_0(p) > 0\}$.*

a) If there exists a T-path $\tau(P_0, t)$ from P_0 to the transition $t \in T$, then transition t is potentially firable in $\mathcal{P}$.

b) If there exists a T-path $\tau(P_0, p)$ from M_0 to the place $p \in P$, then place p is coverable in $\mathcal{P}$.

Proof. By induction on the cardinality of the T-path.

Basis.

a) $|\tau(P_0, t)| = 1$. This means that the considered T-path consists only of the single transition t with $^\bullet t \subseteq P_0$. Thus t is potentially firable in $\mathcal{P}$.

b) $|\tau(P_0, p)| = 0$. This means that $p \in P_0$; hence $M_0(p) > 0$, and p is trivially coverable in $\mathcal{P}$ (by an empty sequence of transitions).

Inductive step. Let us suppose that the lemma is true for any T-path containing less than n transitions.

a) Let $\tau(P_0, t)$ be a T-path containing n transitions. By Definition 4.1, $\tau(P_0, t) = \bigcup_{p_i \in {}^\bullet t} \tau(P_0, p_i) \cup \{t\}$ and then for each $p_i \in {}^\bullet t$ there exists a (possibly empty) T-path $\tau(P_0, p_i) \subset \tau(P_0, t)$. Since the cardinality of such T-paths is smaller than n, then by inductive hypothesis for any $p_i \in {}^\bullet t$, p_i is coverable in $\mathcal{P}$; by Lemma 3.2 this implies that transition t is potentially firable.

b) Analogously, if $\tau(P_0, p)$ is a T-path containing n transitions then, by Definition 4.1, there exists a transition t such that $p \in t^\bullet$ and $\tau(P_0, t) = \bigcup_{p_i \in {}^\bullet t} \tau(P_0, p_i) \cup \{t\}$. Furthermore, for any i, $|\tau(P_0, p_i)| < n$, and so p_i is coverable by inductive hypothesis. Therefore, by Lemma 3.2, transition t is potentially firable.

□

5. Algorithms for T-paths in Petri nets

Let $\mathcal{P} = \langle P, T, A, M_0 \rangle$ be a general Petri net. In this section we provide linear time algorithms to determine the portion of $\mathcal{P}$ reachable by T-paths from $P_0 = \{p \mid M_0(p) > 0\}$, namely the subnet $\mathcal{P}_R = \langle P_R, T_R, A_R, M_0 \rangle$, where $T_R \subseteq T$ (respectively, $P_R \subseteq P$) are the transitions (respectively, the places) reachable by a T-path from P_0 and $A_R \subseteq A$ is the set of arcs induced by the sets T_R and P_R.

In a general Petri net, finding such sets provides a solution to the problem of coverability by augmentation, as stated in Theorem 4.5. For a *CF* net, the sets T_R and P_R actually are the set of potentially firable transitions and the set of coverable places, respectively.

The concepts proposed in this section are related to *directed hypergraphs* (see, e.g., [6, 8, 2]). A directed hypergraph is a pair (V, H), where V is the set of *nodes*, and $H \in \mathcal{S}^+(V) \times V$ is the set of *hyperarcs*, where $\mathcal{S}^+(V)$ denotes the family of nonempty subsets of V. Directed hypergraphs have been extensively used as a suitable mathematical representation model in different areas of computer science, such as problem solving [8], functional dependencies in relational databases [2], linear programming [16], and logic programming [4].

Efficient algorithms have been devised to efficiently update the structure of a directed hypergraph while updates are performed [4, 5]. In this section, beside the basic algorithms to find the reachable portion of a *CF* net, we propose an *incremental* solution as well: we are allowed to modify the net (by using the set of operations stated below), and recompute the new sets P_R, T_R, and A_R without recomputing them from scratch, but *updating* the previous solution. The incremental version of these algorithms is intended to be an effective tool for software systems using Petri nets. In particular, the following operations are supported by the incremental version of the algorithms:

a) inserting a disconnected place in P;

b) inserting a new transition t defined together with its input set ${}^\bullet t$;

c) inserting an arc (t, p) in A;

d) extending the initial marking M_0, including a place p in the initially marked places P_0;

e) asking whether there exists a T-path from P_0 to any transition or place.

5.1. Data structures and algorithms

In our data structures, we represent a Petri net as a bipartite graph with adjacency lists with size $O(|P| + |T| + |A|)$: for any node $x \in P \cup T$, the nodes in the sets ${}^\bullet x$ and $x^\bullet$ are stored as linked lists. Moreover, for any node $x \in P \cup T$, a counter $C(x)$, which is defined in the following way, is maintained:

- for any transition $t \in T$: $C(t) = \Big| \, \{p \mid p \in {}^\bullet t \text{ and } p \text{ is not coverable } \} \, \Big|$;
- for any place $p \in P$:
$$C(p) = \begin{cases} 1 & \text{if } p \text{ is not coverable} \\ 0 & \text{if } p \text{ is coverable} \end{cases}$$

Let us consider first the static computation performed by the Algorithm *Reachability*, shown in Figure 7: this computes the subnet $\mathcal{P}_R = \langle P_R, T_R, A_R, M_0 \rangle$ and sets the correct value for the counters $C(y)$ for every $y \in P \cup T$. After the initialization (lines 2–7), the places in the initial marking P_0 are considered by calling the procedure *ExtendMarking* (see Figure 8). Then, all the transitions with empty input set are taken into account: for every transition t having

```
     Algorithm Reachability;
     Input: $\mathcal{P} = \langle P, T, A, M_0 \rangle$;
     Output: $\mathcal{P}_R = \langle P_R, T_R, A_R, M_0 \rangle$;
1.   begin
2.       $P_0 \leftarrow \{p \mid p \in P$ and $M_0(p) > 0\}$;
3.       $P_R \leftarrow \emptyset$;
4.       $T_R \leftarrow \emptyset$;
5.       $A_R \leftarrow \emptyset$;
6.       for each $p \in P$ do $C(p) \leftarrow 1$;
7.       for each $t \in T$ do $C(t) \leftarrow |{}^\bullet t|$;
8.       for each $p \in P_0$ do ExtendMarking($p$);
9.       for each $t \in T$ do
10.          if $|{}^\bullet t| = 0$
11.             then for each $p \in t^\bullet$ do if $C(p) > 0$ then Reach($p$);
12.  end.
```

Figure 7: Algorithm *Reachability* to compute the reachable portion of a net.

```
     Procedure ExtendMarking($p$:place);
1.   begin
2.       if $C(p) > 0$ then
3.           begin
4.               $C(p) \leftarrow 0$;
5.               insert $p$ in $P_R$;
6.               for each $t \in p^\bullet$ do Reach($t$);
7.           end
8.   end.
```

Figure 8: Procedure *ExtendMarking* extending the initial marking to a new place p.

$|{}^\bullet t| = 0$, we have that $C(t) = 0$, meaning that the transition is potentially firable, regardless of the initial marking.

Procedure *Reach*, shown in Figure 9, is in charge of visiting the reachable portion of the net and updating the data structures according to their definition: for any node $x \in P \cup T$, the value of counter $C(x)$ is set to zero if and only if node x is reachable (i.e., if there is a T-path from P_0 to x). Also note that an arc (x, y) is in A_R if and only if the transition that must be one of the endpoints of the arc is potentially firable.

The following incremental procedures allow the user to perform modifications of the net, and update the reachable portion of the net accordingly:

ExtendMarking (in Figure 8) is the same procedure called by Algorithm Reachability, and handles incremental updates to the initial marking, i.e., insert a place p given in input in the set of initially marked places P_0;

Insert_Place, shown in Figure 10, carries out the insertion of an isolated place p in P;

Insert_Transition (see Figure 11) is in charge to perform the insertion of a new transition t in T, together with its input set ${}^\bullet t$;

```
Procedure Reach($y$:node);
begin
    $C(y) \leftarrow C(y) - 1$;
    if $C(y) = 0$
       then begin
                if $y$ is a place
                   then insert $y$ in $P_R$
                   else begin
                               insert $y$ in $T_R$;
                               for each $p$ in ${}^\bullet y$ do insert $(p, y)$ in $A_R$;
                               for each $p$ in $y^\bullet$ do insert $(y, p)$ in $A_R$;
                        end;
                for each $z$ in $y^\bullet$ do
                   if $C(z) > 0$ then Reach($z$);
             end
end.
```

Figure 9: Procedure *Reach*.

```
Procedure Insert_Place;
begin
    $p \leftarrow$ Make_New(place);
    insert $p$ into $P$;
    ${}^\bullet p \leftarrow \emptyset$;
    $p^\bullet \leftarrow \emptyset$;
    $C(p) \leftarrow 1$;
    return $p$;
end.
```

Figure 10: Procedure *Insert_Place* inserting a new isolated place p.

Insert_Arc, shown in Figure 12, handles the insertion of a new arc (x, y) in A.

Note that procedure *Insert_Arc* updates the data structures while inserting in $\mathcal{P}$ an arc either from a transition x to a place y, or from a place x to a transition y. We remark that, using the procedures shown in this Section, only the insertion of an arc from a transition to a place (and not vice-versa) can be performed efficiently.

This "asymmetric" behaviour is more evident by considering the consequence to the reachability set $R(M)$ due to the arc insertion. Inserting an arc from a transition t to a place p can only extend the set of reachable markings: when t fires, an additional token is inserted in the new connected place. On the contrary, inserting an arc from a place to a transition can only reduce the set of reachable states, since this new arc plays the role of an additional constraint for the firability of the connected transition. Therefore the insertion of an arc from a place to a transition is not "incremental" but, by using a terminology adopted in dealing with dynamic graphs, has to be considered "decremental". This means that the linear time bound for any sequence of updates to the net might not hold if one carries out insertion of arcs from places to transitions, too. Nevertheless, procedure *Insert_Arc* handles also this case in order to provide

```
    Procedure $\textit{Insert\_Transition}$(InSet: set of places);      {$\textit{InSet}$ is the input set $^\bullet t$}
1.  begin
2.      $t \leftarrow \textit{Make\_New}$(transition);
3.      insert $t$ into $T$;
4.      $^\bullet t \leftarrow \emptyset$;
5.      $t^\bullet \leftarrow \emptyset$;
6.      for each $p \in$ InSet do
7.          begin
8.              insert $p$ into $^\bullet t$;
9.              insert $t$ into $p^\bullet$;
10.             insert $(p,t)$ into $A$;
11.         end;
12.     $C(t) \leftarrow \sum_{p \in \mathit{InSet}} C(p)$;
13.     return $t$;
14. end.
```

Figure 11: Procedure *Insert_Transition* building up a new transition t with input set *InSet*.

a more comprehensive set of primitives. Namely, procedure *Insert_Arc* requires a complete recomputation from scratch of the reachable subnet $\mathcal{P}_R$ (by calling Algorithm *Reachability*) in the only case that all the following conditions hold:

1. the arc (x, y) has to be inserted from a place $x \in P$ to a transition $y \in T$;
2. $x \notin P_R$, i.e., $C(x) > 0$;
3. $y \in T_R$, i.e., $C(y) = 0$.

These conditions are verified in line 13, line 5 (through the "else" branch), and line 14, respectively. In this case the insertion of an arc (x, y) requires to remove transition y from the set T_R, and in our approach this requires a complete recomputation from scratch of the reachable portion of the net, which is executed by a call to algorithm *Reachability* (line 15).

Another nontrivial case holds when an arc (x, y) from a transition $x \in T_R$ to a place $y \notin P_R$ has to be inserted: a call to procedure *Reach* is required to properly update the data structures (line 11).

5.2. On the complexity of finding and maintaining T-paths

Now we can state our results, proving that finding and/or maintaining information about all T-paths from the initially marked places $P_0 = \{p \mid M_0(p) > 0\}$, while performing incremental updates to the net, can be done in linear time.

Theorem 5.1. *Let $\mathcal{P} = \langle P, T, A, M_0 \rangle$ be a general Petri net. Finding all the places $p \in P$ (transitions $t \in T$) such that there exists a T-path $\tau(P_0, p)$ ($\tau(P_0, t)$) requires $O(|P| + |T| + |A|)$ time.*

Proof. For each node $y \in P \cup T$, testing the reachability by a T-path from P_0 can be simply checked as ownership to P_R or T_R, or also by checking $C(x) = 0$.

By inspection of the code of algorithm *Reachability* and procedure *ExtendMarking* it is possible to verify that these require a time proportional to $|P| + |T|$, plus at most $|A|$ calls to procedure *Reach*.

```
     **Procedure** *Insert_Arc*($x, y$:node);
 1.  **begin**
 2.     insert $y$ into $x^\bullet$;
 3.     insert $x$ into $^\bullet y$;
 4.     insert $(x, y)$ into $A$;
 5.     **if** $C(x) = 0$
 6.        **then if** $C(y) = 0$
 7.              **then** insert $(x, y)$ into $A_R$;
 8.              **else if** $x$ is a transition
 9.                    **then begin**
10.                             insert $(x, y)$ into $A_R$;
11.                             *Reach*$(y)$;
12.                         **end**
13.        **else if** $x$ is a place
14.              **then if** $C(y) = 0$
15.                    **then** *Reachability*;     {recompute data structures from scratch}
16.                    **else** $C(y) \leftarrow C(y) + 1$;
17.  **end**.
```

Figure 12: Procedure *Insert_Arc* to perform arc insertion.

Any call to *Reach*(y) may require either constant time if $C(y) > 0$ after the update, or additional work if $C(y) = 0$. In the latter case, all the arcs leaving node y are scanned, performing a recursive call to *Reach*(z) for each $z \in y^\bullet$ (line 13); furthermore, if y is a transition, all arcs leaving or entering transition y are inserted in A_R.

In conclusion, in any nontrivial call to *Reach*(y), a constant time is spent both for node y and for each arc either leaving or entering node y: this can happen at most once for each node $y \in P \cup T$. In fact, when a counter $C(x)$ gets the value 0, no call to *Reach*(x) can be performed any more: in the three procedures this test is performed before any call to procedure *Reach*.□

Theorem 5.2. *Let $\mathcal{P} = \langle P, T, A, M_0 \rangle$ be a general Petri net. There exist data structures and algorithms to perform the following operations:*

a) inserting a disconnected place in P;
b) inserting a new transition t defined together with its input set $^\bullet t$;
c) inserting an arc (t, p) in A;
d) extending the initial marking M_0, including a place p in the initially marked places P_0;
e) asking whether there exists a T-path from P_0 to any transition or place.

The total time required to perform an arbitrary sequence ω of operations of the above kinds, starting from an empty net, is $O(|\omega| + |P| + |T| + |A|)$, where the cardinalities of the sets refer to the final net. In particular an operation of kind e), that is, asking whether there exists a T-path from P_0 to a given place or transition, requires constant worst-case time.

Proof. Let us consider the total time spent by each kind of operation along the whole sequence of operations.

Each operation of the kinds *a)* and *e)* requires constant time, and the total work done for such operations in a sequence ω is obviously $O(|\omega|)$.

Operations of kind *b)* require globally a total time bounded by $|\{{}^\bullet t \mid t \in T\}|$, and hence $O(|A|)$ on the whole sequence.

Let us consider operations *c)* and *d)*. We have that both spend constant time plus zero or more calls to procedure *Reach*[2]. The argument used in the proof of Theorem 5.1 to bound the total work done by procedure *Reach* still holds, also in case of subsequent calls due to incremental updates of the net. This lead to a bound of $O(|P| + |T| + |A|)$ for the total time spent by procedure *Reach* due to a sequence of update operations.

Therefore the total time spent for a sequence $|\omega|$ of calls to the procedures is cumulatively bounded by the quantity $O(|\omega| + |P| + |T| + |A|)$, where the cardinalities refer to the final net. $\square$

6. Liveness

In this section we examine the liveness problem for *CF* nets, and propose linear time algorithms to determine the set of live transitions of a given net in this class. We recall that the liveness problem is to determine the set of live transitions: a transition is live if it is potentially firable in every reachable markings. In particular, we will show that for this class of nets the liveness problem can be reduced to the verification of *structural* properties of the net. We introduce the notion of *autonomous set*, a set of potentially firable transitions for which the input set is contained in the output set. An autonomous set is "self-feeding" in the sense that, as its transitions can fire at least once, they are able to re-fill their own input places, and hence fire infinitely many times. Autonomous sets are a kind of dual of the notions of traps [29].

As already mentioned, a *trap* is a set of places which remain marked once they have gained at least one token: a trap is a subset S of places s.t. $S^\bullet \subseteq {}^\bullet S$ (i.e., any output transition of S is also an input transition of S). In a dual fashion, at least one transition in an autonomous set will be enabled, once some transition in the set has been enabled. Traps have been largely used to analyze classes of Petri nets (see, e.g., [10, 22]). Usually results are stated by considering "traps marked by M_0". In *CF* nets analogous properties (namely, the liveness) may be stated by checking the potential firability of any transition in an autonomous set; in turn, this property is related to the notion of T-path reachability from M_0.

A simple variation of the topological sort algorithm (see, e.g., [12]), eliminating transitions that are not part of autonomous sets, can be used to determine the live subnet $\mathcal{P}_L$ of a given net $\mathcal{P} = \langle P, T, A, M_0\rangle$. This is defined by the set of live transitions, together with the union of all the input and output places of these: $\mathcal{P}_L = \langle P_L, T_L, A_L\rangle$, where $T_L = \{t | t \in T$, and t is live$\}$, and $P_L = T_L^\bullet \cup {}^\bullet T_L$.

Definition 6.1. *Given a Petri net $\mathcal{P} = \langle P, T, A, M_0\rangle$ a set of transitions $C \subseteq T$ is an* autonomous set *if:*

a) for every $t \in C$, t is potentially firable in M_0, and

b) ${}^\bullet C \subseteq C^\bullet$

Examples of autonomous set are shown in Figure 13.

[2]As remarked above, here we do not take into account the insertion of arcs from places to transitions.

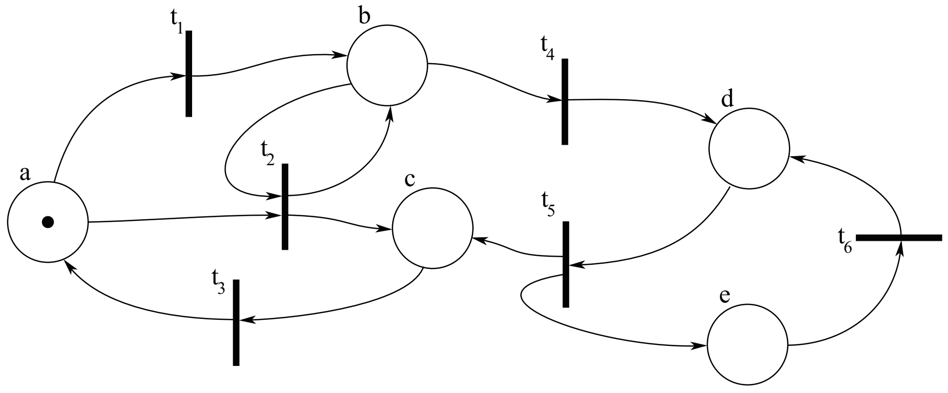

Figure 13: A set of places $C \subseteq P$ is an autonomous set if every transition is potentially firable, and ${}^\bullet C \subseteq C^\bullet$; examples are $\{t_1, t_2, t_3\}$, $\{t_5, t_6\}$, and P.

The reminder of this section is organized as follows. We first show a relationship between autonomous sets and sets of live transitions in *CF* nets and then we provide an algorithm that determines the set of all live transitions.

In the following we use a characterization provided by Howell and Rosier in [19], Lemma 3.2: *in a CF net a transition t is live if and only if there exists a firable sequence that uses t infinitely many times.*

Lemma 6.1. *Given a CF net $\mathcal{P} = \langle P, T, A, M_0 \rangle$, a transition $t \in T$ is live if and only if there exists an autonomous set of transitions $C \subseteq T$ such that $t \in C$.*

Proof.
$(\Longrightarrow)$ Let σ_t be a firable sequence containing infinitely many times the live transition $t \in T$. Let $C(\sigma_t)$ be the set of transitions occurring infinitely many times in σ_t. For each transition $s \in C(\sigma_t)$, and each $p_i \in {}^\bullet s$, since s fires infinitely many times in σ_t, there must be a transition s_i with $p_i \in s_i^\bullet$, which occurs infinitely many times in σ_t. Therefore, $s_i \in C(\sigma_t)$, and ${}^\bullet s \subseteq \bigcup_i s_i^\bullet \subseteq \bigcup_{t \in C(\sigma_t)} t^\bullet$. Since this is true for all $s \in C(\sigma_t)$, we have that ${}^\bullet C(\sigma_t) \subseteq C(\sigma_t)^\bullet$, i.e., $C(\sigma_t)$ is an autonomous set.

$(\Longleftarrow)$ Let us consider an autonomous set C and any transition $t \in C$. Since all transitions in C are potentially firable, there exists a sequence σ_F firable in M_0 which uses exactly once every transition in C. Let us consider the marking $M_{C,0}$ such that $M_0 \xrightarrow{\sigma_F} M_{C,0}$, and the sequence σ_C built from σ_F by deleting all the transitions in the set $F - C$, i.e., σ_C consists of a permutation of the transitions in C.

Finally, let us consider the infinite sequence $\sigma = \sigma_F \sigma_C \sigma_C \sigma_C \ldots$: we claim that this sequence is firable in $\mathcal{P}$; since this contains t infinitely many times, this claim will prove the lemma. Let us consider a generic transition $t_i \in C$ assuming that its index refer to the i-th position in σ_C. We prove the claim for $i = 1, 2, \ldots$, and focusing on the first occurrence of σ_C in σ, after firing σ_F which is known to be firable in $\mathcal{P}$.

Basis. For $i = 1$, we have that all transitions in $C - \{t_1\}$ have been fired exactly once in the suffix of σ_F following the unique occurrence of t_1. Therefore, while firing that suffix which does not contains t_1 and contains all other transitions in C, each place in ${}^\bullet t_1 \subseteq {}^\bullet C$ has received at least one token. Hence, by using the same argument of Lemma 3.2, in a *CF* net no token can be removed from places in ${}^\bullet t_1$ by other transitions different from t_1.

```
    Algorithm Liveness;
    Input: $\mathcal{P}_R = \langle P_R, T_R, A_R, M_0\rangle$;   {Reachable subnet: $\exists T$-path from $M_0$ to all $x \in P_R \cup T_R$}
    Output: $\mathcal{P}_L = \langle P_L, T_L, A_L\rangle$;   {Live subnet: $T_L = \{t | t$ is live$\}$; $P_L = T_L^\bullet \cup {}^\bullet T_L$}
1.  begin
2.      for each $p \in P_R$ do
3.          $count(p) \leftarrow |{}^\bullet p|$;
4.      $S \leftarrow \{p \in P_R \mid count(p) = 0\}$;
5.      $P_L \leftarrow P_R - S$;
6.      $T_L \leftarrow T_R$;
7.      while $S$ is not empty do
8.          begin
9.              choose $p \in S$;
10.             $S \leftarrow S - \{p\}$;
11.             for each $t \in p^\bullet$ do
12.                 begin
13.                     $T_L \leftarrow T_L - \{t\}$;
14.                     for each $p' \in t^\bullet$ do
15.                         begin
16.                             $count(p') \leftarrow count(p') - 1$;
17.                             if $count(p') = 0$
18.                                 then begin
19.                                     $S \leftarrow S \cup \{p'\}$
20.                                     $P_L \leftarrow P_L - \{p'\}$
21.                                 end;
22.                         end;
23.                 end;
24.         end;
25.     $A_L \leftarrow \{(p,t) \mid t \in T_L\} \cup \{(t,p) \mid t \in T_L\}$
26. end.
```

Figure 14: algorithm *Liveness* to compute the live subnet.

Inductive step. Let us suppose that the claim is true for any $j < i$. Between any two consecutive occurrences of a transition t_i in σ we have that all the transitions in $C - \{t_i\}$ are fired exactly once and, again, each place in ${}^\bullet t_i \subseteq {}^\bullet C$ has received at least one token. In a *CF* net this is sufficient to conclude that t_i is enabled.

We conclude by remarking that the same arguments above applies regarding the firability of a generic transition $t_i \in C$, which is enabled by the firing of all transitions in $C - \{t_i\}$ within any two consecutive occurrences of σ_C in σ. □

The algorithm *Liveness* given in Figure 14 determines the set of live transitions T_L of a given *CF* net. The algorithm, whose input is the reachable portion of the net, iteratively finds and deletes from the net all the transitions whose input places will not be re-filled once they have been emptied.

Theorem 6.2. *Algorithm Liveness determines the live subnet $\mathcal{P}_L = \langle P_L, T_L, A_L\rangle$ of a reachable CF net $\mathcal{P}_R = \langle P_R, T_R, A_R, M_0\rangle$ in $O(|P_R| + |T_R| + |A_R|)$ time.*

Proof. *(Correctness)* We will show that, after the execution of algorithm *Liveness* in Figure 14, the set T_L contains exactly the set of live transitions. We will use the characterization of a live transition provided by Lemma 6.1: a transition t is live if and only if there exists an autonomous set C containing t.

We first prove the completeness of the algorithm, i.e., all the live transitions are found. After the initialization, all potentially firable transitions are part of T_L, and the algorithm proceeds by deleting transitions in each step (line 13). Suppose, by contradiction, that at least one transition belonging to some autonomous set C is deleted by T_L, and let $t \in C$ be the first of such transitions removed by the algorithm. This can only happen if there exists a place $p \in t^\bullet$ (lines 9–11) that was inserted in S (at lines 4 or 19) since $count(p) = 0$. But then no transition t' such that $t' \in {}^\bullet p$ is part of T_L, and hence each one of those t' must have been deleted from T_L before. But liveness conditions state that at least one of these t' with $p \in t'^\bullet$ is in the same C as t, and t was the first transition belonging to an autonomous set to be deleted, a contradiction.

On the other hand all the transitions found by the algorithm are live. Consider a transition t such that $t \in T_L$: in this case every place $p \in {}^\bullet t$ has $count(p) \neq 0$, and then, for each of such places, there is a transition $t_p \in T_L$ such that $p \in t_p^\bullet$. This implies that $\cup_{t \in T_L} {}^\bullet t \subseteq \cup_{t \in T_L} t^\bullet$, and then T_L is an autonomous set.

(Time complexity) The initialization requires no more than one visit to the net, and hence time $O(|P_R| + |T_R| + |A_R|)$; the "while" loop (lines 8–24) is executed at most $|P|$ times, and each arc (x, y) with $y \in x^\bullet$ (at lines 11, or 14), is considered at most once during the execution of the algorithm, just after that node x has been deleted from the live subnet (lines 10, or 13). Hence, the overall running time of the algorithm is $O(|P_R| + |T_R| + |A_R|)$. $\square$

7. Boundedness

Karp and Miller [24] have shown that boundedness of Petri Nets is a decidable property, but a solution for this problem may require exponential time and space for general nets [27]. For *CF* nets, though, Howell et al. have shown that boundedness of a net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ can be decided in $O(|P| \times |T|)$ time and space [20]; in this section we give algorithms and data structures for deciding boundedness for the class of *CF* nets in $O(|P| + |T| + |A|)$ time and space. We determine necessary and sufficient conditions on the set of live transitions T_L (determined by algorithm *Liveness* provided above) for the net to be bounded. Following Karp and Miller's characterization, we have that a Petri net is unbounded if and only if it can execute a positive loop, that is, if there exists a set $C \subseteq T$ and a potentially firable sequence σ (from a marking M) which uses exactly once every transition in C such that $M \xrightarrow{\sigma} M'$ and $M' > M$.

In the following lemma we will consider the notion of *displacement* $\delta_\sigma(p)$ (i.e., the total variation of the number of tokens) caused on place p by a sequence σ. This can be defined by considering the transitions occurring in σ and such that p is either in the input set, or in the output set. More precisely: $\delta_\sigma(p) = \sum_{p \in t_i^\bullet} \Psi(\sigma)[t_i] - \sum_{p \in {}^\bullet t_i} \Psi(\sigma)[t_i]$, where $\Psi(\sigma)$ is the Parikh vector (see Section 4.1) of sequence σ, providing the number of occurrences of each transition. If sequence σ is firable in a marking M, with $M \xrightarrow{\sigma} M'$, then $\delta_\sigma(p) = M'(p) - M(p)$.

Lemma 7.1. *A CF net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ is bounded if and only if for every autonomous set $C \subseteq T$, $|{}^\bullet C| = |C^\bullet|$.*

Proof.
($\Longrightarrow$) Let us consider any autonomous set C in $\mathcal{P}$. Then there exists a reachable marking M and a potentially firable sequence σ such that $M \xrightarrow{\sigma} M'$, with $M' \geq M$, where for all $t \in C$, t occurs exactly once in σ. Let $\delta_\sigma(p)$ be the displacement caused on place p by sequence σ. Since the net is bounded, for all $p \in P$, $\delta_\sigma(p) = 0$, that is the quantity of tokens consumed by the firing of the transitions in C is exactly the number of tokens produced, i.e., $\sum_{p\in P} |p^\bullet \cap C| = \sum_{p\in P} |{}^\bullet p \cap C|$. Since we have that $\sum_{p\in P} |p^\bullet \cap C| = \sum_{t\in C} |{}^\bullet t|$, and $\sum_{p\in P} |{}^\bullet p \cap C| = \sum_{t\in C} |t^\bullet|$, i.e., the number of output arcs for the places is equal to the number of input arcs for the transitions. On the other hand, the number of output arcs for the transitions is equal to the number of input arcs for the places, and therefore $|{}^\bullet C| = |C^\bullet|$.
($\Longleftarrow$) By contradiction, let us consider an unbounded *CF* net $\mathcal{P}$. There exists a potentially firable sequence σ such that $M \xrightarrow{\sigma} M'$, where each transition in σ is used exactly once, and $M' = M + \delta_\sigma > M$, with $\delta_\sigma > 0$. Therefore, by similar arguments as those in the previous step of this proof, there exists an autonomous set $C_\sigma = \{t \in T \mid t \text{ is used in } \sigma\}$ such that $\sum_{t\in C_\sigma} |{}^\bullet t| < \sum_{t\in C_\sigma} |t^\bullet|$, i.e., $|{}^\bullet C| < |C^\bullet|$. $\square$

A possible algorithm to determine the boundedness of a Petri net in the considered classes, would consist in verifying if the above equality holds for every autonomous set C; however, this procedure could be extremely expensive. Instead, verifying the equality only for the entire set of live transitions T_L can be done in linear time. If it is not satisfied, then the net is unbounded. Conversely, note that an unbounded net must contain some autonomous set that produces more tokens than it consumes, so if the equality holds, then there must be some autonomous set containing the former that "hides" the unboundedness by means of transitions that consume more tokens than they produce.

Based on this idea, we will show that the notion of autonomous set allows us to reduce the boundedness problem for *CF* nets to the verification of structural properties of the underlying graph. More precisely, we will show that the boundedness property of a marked *CF* net $\mathcal{P} = \langle P, T, A, M_0\rangle$ can be checked by examining the structure of the live subnet $\hat{\mathcal{P}}_L = \langle P_L, T_L, A_L\rangle$.

We will construct an unmarked net, derived from the original one, where the presence of autonomous sets that produce more tokens than they consume (leading to unboundedness) is mapped into the presence of arcs belonging to no cycles (in a graph-theoretical sense). That leads to the existence of connected components of the net (considering the arcs as undirected) that are not strongly connected.

Lemma 7.2. *Let us consider a CF net* $\mathcal{P} = \langle P, T, A, M_0\rangle$ *and its live subnet* $\mathcal{P}_L = \langle P_L, T_L, A_L\rangle$. *If* $\sum_{t\in T_L} |{}^\bullet t| = \sum_{t\in T_L} |t^\bullet|$ *then, for every place* $p \in P$, $|p^\bullet \cap T_L| = |{}^\bullet p \cap T_L|$.

Proof. For any place $p \in P_L$, $|{}^\bullet p| > 0$. Furthermore, since P_L is live and conflict-free, $|{}^\bullet p| \geq |p^\bullet|$ for each $p \in P_L$. On the other side, $\sum_{t\in T_L} |{}^\bullet t| = \sum_{t\in T_L} |t^\bullet|$ implies $\sum_{p\in P_L} |p^\bullet \cap T_L| = \sum_{p\in P_L} |{}^\bullet p \cap T_L|$. The two relationships imply $|p^\bullet \cap T_L| = |{}^\bullet p \cap T_L|$ for all $p \in P_L$. $\square$

In order to decide efficiently the boundedness of *CF* nets, we need to introduce a transformation of the network by splitting the branched places.

Definition 7.1. *Given a CF net* $\mathcal{P} = \langle P, T, A, M_0\rangle$, *let* $\hat{\mathcal{P}}_L = \langle P_L, T_L, A_L\rangle$ *be the portion of the live subnet* $\mathcal{P}_L$ *such that* $\sum_{t\in T_L} |{}^\bullet t| = \sum_{t\in T_L} |t^\bullet|$. $\hat{\mathcal{P}}'_L = \langle P'_L, T'_L, A'_L\rangle$ *is the unmarked net obtained from* $\hat{\mathcal{P}}_L$ *by splitting branched places in the following way: given a branched place p,* *for every transition t such that $p \in {}^\bullet t$ we introduce a place p_t with $p_t^\bullet = {}^\bullet p_t = \{t\}$; p_t is said to be* new *and the original place p is removed from the net together with all the incident arcs. Unbranched places in $\hat{\mathcal{P}}$ are left unchanged and are called* old.

Note that this decomposition preserves the notion of autonomous set, since, for any set $C \subseteq T$, if $\bigcup_{t \in C} {}^\bullet t \subseteq \bigcup_{t \in C} t^\bullet$ in $\hat{\mathcal{P}}_L$, then $\bigcup_{t \in C} {}^\bullet t \subseteq \bigcup_{t \in C} t^\bullet$ in $\hat{\mathcal{P}}'_L$.

In the remaining of this section, and when no confusion arises, the adjectives marked and unmarked will be omitted.

Lemma 7.3. *Let $\mathcal{P} = \langle P, T, A, M_0 \rangle$ be a CF net. If in $\hat{\mathcal{P}}'_L$ there is an arc that is not part of a cycle, then $\mathcal{P}$ is unbounded.*

Proof. Consider an arc from a place p to a transition t that is not part of a cycle. Then p cannot be a new place, as by construction all arcs leaving new places are part of cycles. Then p is old, and, as p is part of $\mathcal{P}_L$, there is at least one arc entering p. The arcs entering p are not part of cycles and hence, if there is an arc that is not part of a cycle, there must be an arc from a transition t to a place p that is not part of a cycle. Consider now such an arc and suppose that the arcs entering t are part of cycles (if there is no such arc, then all arcs are part of cycles). If ${}^\bullet t = \emptyset$, then clearly the net is unbounded, and if ${}^\bullet t \neq \emptyset$, then all the places in ${}^\bullet t$ can be filled without p been emptied, so the net is unbounded. $\square$

Lemma 7.4. *If a CF net $\mathcal{P}$ is unbounded and $\sum_{t \in T_L} |{}^\bullet t| = \sum_{t \in T_L} |t^\bullet|$, then in $\hat{\mathcal{P}}'_L = \langle P'_L, T'_L, A'_L \rangle$ there exists an arc that is not part of a cycle.*

Proof. By Lemma 7.2, we have that for every place $p \in P$, $|p^\bullet \cap T_L| = |{}^\bullet p \cap T_L|$. But then, by construction of the net $\hat{\mathcal{P}}'_L$, for every place p, we have that $|p^\bullet| = |{}^\bullet p| = 1$.

On the other hand, since the net is unbounded, there exists an autonomous set C such that $\sum_{t \in C} |{}^\bullet t| < \sum_{t \in C} |t^\bullet|$ and hence $\sum_{p \in P} |p^\bullet \cap C| < \sum_{p \in P} |{}^\bullet p \cap C|$. It follows that there must be a place $p \in P'_L$ such that $|p^\bullet \cap C| < |{}^\bullet p \cap C|$.

This place p cannot be branched. In this case, for each transition $t \in {}^\bullet p$ such that $t \in C$, also transition $\texttt{corr}(t)$ must be in C with $\texttt{corr}(t) \in p^\bullet$. But then t and $\texttt{corr}(t)$ together contribute equally to the cardinality of ${}^\bullet p$ and $p^\bullet$.

Therefore p can only be unbranched, hence $|p^\bullet \cap C| < |{}^\bullet p \cap C|$ implies that $|p^\bullet \cap C| = 0$ and $|{}^\bullet p \cap C| = 1$, and p is an old place in P_L. Let us consider the unique arc in A'_L entering p. If this arc were part of a cycle, then this cycle should re-enter C either via a transition or via a place. Both alternatives are impossible, because C is an autonomous set: then every place in the input set of any transition in C must be also in the output set of some transition in C, and every place in P'_L has only one input in T'_L. $\square$

Testing whether there is an arc that does not belong to a cycle can be easily done by computing both the *strongly connected components* of the graph and the *connected components* of its undirected version, as stated in the following lemma.

Lemma 7.5. *Given a directed graph $D = (V, A)$, and its undirected version $G = (V, E)$, if the number of strongly connected components of D is different from the number of connected components of G, then there is at least one arc that it is not part of a cycle.*

```
function $Boundedness$;
Input:  $\mathcal{P}_L = \langle P_L, T_L, A_L \rangle$;
begin
    $C_{IN} \leftarrow \sum_{t \in T_L} |{}^{\bullet}t|$;
    $C_{OUT} \leftarrow \sum_{t \in T_L} |t^{\bullet}|$;
    if $C_{IN} \neq C_{OUT}$
       then $Boundedness \leftarrow$ FALSE
       else begin
                $CC \leftarrow$ Connected-Components($T_L \cup P_L, A_L$)
                $SCC \leftarrow$ Strongly-Connected-Components($T_L \cup P_L, A_L$)
                if $CC \neq SCC$
                   then $Boundedness \leftarrow$ FALSE
                   else $Boundedness \leftarrow$ TRUE
            end;
end.
```

Figure 15: Function *Boundedness* that determines the boundedness of a *CF* net.

Proof. By contradiction, let us suppose that all arcs are part of a cycle. Let us consider two vertices x and y in the same connected component, i.e., there exists a path in the undirected graph G. For each edge (i, j) of this path, there exist both an arc $i \rightarrow j$ in A and a path from j to i (or vice-versa) that build up a cycle. Since this is true for all edges in the path, if x and y are in the same connected component of G then x and y are in the same strongly connected component of D. This contradicts the fact that $|SCC(D)| > |CC(G)|$. □

The main result of this section is provided in the following theorem, while an algorithm to decide the boundedness of any *CF* net is shown in Figure 15.

Theorem 7.6. *A CF net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ is bounded if and only if the live subnet $\mathcal{P}_L$ of $\mathcal{P}$ fulfills the following properties:*

a) $\sum_{t \in T_L} |{}^{\bullet}t| = \sum_{t \in T_L} |t^{\bullet}|$ *and*

b) every arc of $\hat{\mathcal{P}}'_L$ is part of a cycle.

The computation required to verify these conditions can be performed in $O(|P| + |T| + |A|)$ time.

Proof. The characterization for bounded *CF* nets is a straightforward consequence of Lemmas 7.3, 7.4, and 7.5.

In order to bound the complexity, we notice that the first step requires the computation of the live subnet; then algorithm *Boundedness*, shown in Figure 15, requires linear time. Namely, the algorithm first verifies whether condition a) is satisfied (line 4): in the negative case the net is unbounded. This test can be carried out in linear time by exploring the live subnet $\hat{\mathcal{P}}_L$. Otherwise, if the number of output arcs from all transitions is equal to the number of input arcs to transitions, the presence of arcs that are part of no cycles is checked, thus complying condition b). As seen in Lemma 7.5, this test can be carried out by computing the strongly connected components and the connected components (of the undirected graph) in the live subnet $\hat{\mathcal{P}}_L$. This computation can be done using well known linear time algorithms (see, e.g., [1, 12]). □

We already mentioned that all the algorithms presented in this paper are incremental; it is important to mention that, when we switch to a dynamic scenario, whilst the running time of all the algorithm presented in the previous sections are linear, to evaluate the *boundedness* we need to use dynamic versions of graph algorithms to compute both connected components and strongly connected components, and the running time must be updated accordingly; more precisely, we can use the classical algorithm of Tarjan [36] for the connected components, with an amortized cost of $O(\alpha(m, n))$, where α is a a very slowly increasing function, a functional inverse of the Ackermann's function. The best approach is to use the algorithm, proposed by Roditty and Zwick [35], for the dynamic maintaintance of the strongly connected components, whose complexity is $O(m\alpha(m, n))$. For a faster implementation, when an arc $i \rightarrow j$ is inserted, one may simply visit the graph in order to check whether i is reachable from j, thus paying only $O(m)$.

Summing up, in a dynamic scenario, when we add an arc to the underlying graph, i.e. a transition or an arc in the Petri Net, the time needed for computing the boundedness is $O(\alpha(m, n) + m) = O(m)$ whilst, as already mentioned, all the algorithm presented in the previous sections requires constant time per insertion.

8. Conclusions

In this paper we propose an extension of techniques developed within the context of directed hypergraphs to deal with Petri nets, and propose algorithms suitable for a practical and straightforward implementation; the incremental versions provided in this paper are tailored to be embedded in applications supporting an interactive analysis and design of nets.

From a theoretical point of view, we propose an approach for analyzing structural properties of Petri nets based on the notion of T-path reachability. For the class of *CF* Petri nets this approach leads to linear time algorithms for determining the coverable places, the potentially firable transitions, the live subnet, and for deciding the boundedness of the net. These results improve the time and space bounds of the previous known solutions by Howell et al. [20, 19] from $O(|P| \times |T|)$ to $O(|P| + |T| + |A|)$.

In a general Petri net, the well known *coverability* problem requires exponential space [33]. By analyzing T-path reachability we provide a partial answer to a weaker formulation of this problem, that we name *coverability by augmentation*: this is especially meaningful in situations where one is more interested in the set of places with tokens, rather than to the actual number of tokens; we determine an answer to this problem on general nets in linear time.

T-path reachability is a notion that might be further exploited for both structural and behavioural problems. As an example, given a net $\mathcal{P} = \langle P, T, A, M_0 \rangle$ and a target marking M, the following problems can be answered efficiently by analyzing T-path reachability: determining the subset of places $P'_0 \subseteq P_0 = \{p \in P | M_0(p) > 0\}$ that must be necessarily marked in order to let M be coverable by augmentation (or a minimal set $P'_0 \subseteq P_0$ with such property).

As shown in Section 4, T-path unreachability is related to *siphons*, widely used to analyze deadlocks in Petri nets. The connections between T-reachability and deadlocks deserve further study.

References

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.

[2] G. Ausiello, A. D'Atri, and D. Saccà. Graph algorithms for functional dependency manipulation. *Journal of the ACM*, 30:752–766, 1983.

[3] G. Ausiello, P. G. Franciosa, and D. Frigioni. Partially dynamic maintenance of minimum weight hyperpaths. *Journal of Discrete Algorithms*, 3(1):27–46, 2005.

[4] G. Ausiello and G. F. Italiano. Online algorithms for polynomially solvable satisfiability problems. *Journal of Logic Programming*, 10:69–90, 1991.

[5] G. Ausiello, G. F. Italiano, and U. Nanni. Dynamic Maintenance of Directed Hypergraphs. *Theoretical Computer Science*, 72(2-3):97–117, 1990.

[6] C. Berge. *Graphs and Hypergraphs.* North Holland, Amsterdam, 1973.

[7] E. Best. A Note on Persistent Petri Nets. In *Concurrency, Graphs and Models: Essays Dedicated to Ugo Montanari on the Occasion of His 65th Birthday*, volume 5065 of *Lecture Notes in Computer Science*, pages 427–438. Springer-Verlag, 2008.

[8] H. Boley. Directed recursive labelnode hypergraphs: A new representation language. *Artificial Intelligence*, 9:49–85, 1977.

[9] A. Cheng, J. Esparza, and J. Palsberg. Complexity Results for 1-safe Nets. *Theoretical Computer Science*, 147(1-2):117–136, 1995.

[10] F. Chu and X. Xie. Deadlock Analysis of Petri Nets Using Siphons and Mathematical Programming. *IEEE Transactions on Robotics and Automation*, 13(6):793–804, 1997.

[11] F. Commoner, A. W. Holt, S. Even, and A. Pnueli. Marked Directed Graphs. *Journal of Computer and System Sciences*, 5(5):511–523, 1971.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* MIT Press, second edition, 2001.

[13] S. Crespi-Reghizzi and D. Mandrioli. A Decidability Theorem for a Class of Vector Additions Systems. *Information Processing Letters*, 3(3):78–80, 1975.

[14] J. Esparza and M. Nielsen. Decidability Issues for Petri Nets - a survey. *Bulletin EATCS*, 52:245–262, 1994.

[15] J. Esparza and M. Silva. A polynomial-time algorithm to decide liveness of bounded free-choice nets. *Theoretical Computer Science*, 102(1):185–205, 1992.

[16] G. Gallo, G. Longo, S. Nguyen, and S. Pallottino. Directed Hypergraphs and Applications. Technical Report 3/90, Dip. di Informatica, Univ. of Pisa, Italy, January 1990.

[17] C. Ghezzi, D. Mandrioli, S. Morasca, and M. Pezzè. A Unified High-Level Petri Net Formalism for Time-Critical Systems. *IEEE Transactions on Software Engineering*, 17(2):160–172, 1991.

[18] M. H. T. Hack. The recursive equivalence of the reachability problem and the liveness problem for Petri nets and vector addition systems. In *15th Annual Symposium on Switching and Automata Theory*, pages 156–164, 1974.

[19] R. Howell and L. Rosier. Problems concerning fairness and temporal logic for conflict-free Petri nets. *Theoretical Computer Science*, 64(3):305–329, 1989.

[20] R. Howell, L. Rosier, and H. Yen. An $O(n^{1.5})$ Algorithm to Decide Boundedness for Conflict-Free Vector Replacement System. *Information Processing Letters*, 25(1):27–33, 1987.

[21] R. Howell, L. Rosier, and H. Yen. A taxonomy of fairness and temporal logic problems for Petri nets. *Theoretical Computer Science*, 82(2):341–372, 1991.

[22] M. D. Jeng and X. Xie. Analysis of modularly composed nets by siphons. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 29(4):399–406, 1999.

[23] N. D. Jones, L. H. Landweber, and Y. E. Lien. Complexity of some Problem in Petri Nets. *Theoretical Computer Science*, 4(3):277–299, 1977.

[24] R. M. Karp and R. Miller. Parallel Program Schemata. *Journal of Computer and System Sciences*, 3(2):147–195, 1969.

[25] R. Keller. A fundamental theorem of asynchronous parallel computation. In Tse-yun Feng, editor, *Parallel Processing*, volume 24 of *Lecture Notes in Computer Science*, pages 102–112. Springer Verlag, 1975.

[26] L. Landweber and E. Robertson. Properties of Conflict-Free and Persistent Petri Nets. *Journal of the ACM*, 25(3):352–364, 1978.

[27] R. Lipton. The reachability problem requires exponential space. Technical Report 62, Department of Computer Science, Yale University, 1976.

[28] E. Mayr. An algorithm for the general Petri net reachability problem. *SIAM Journal on Computing*, 13(3):441–459, 1984.

[29] T. Murata. Petri Nets: Properties, Analysis and Applications. In *Proceedings of the IEEE*, pages 541–580, 1989.

[30] J. L. Peterson. Petri Nets. *Computing Surveys*, 9(3):223–252, 1977.

[31] C. A. Petri. Communication with automata. Technical Report Supplement 1 to Tech. Report RADC-TR-65-377, 1966, Rome Air Development Center (U.S. Air Force), 1962. Original in German: *Kommunikation mit Automaten*, Ph.D. Thesis, Univ. of Bonn, 1962.

[32] L. Piroddi, R. Cordone, and I. Fumagalli. Combined siphon and marking generation for deadlock prevention in Petri nets. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 39(3):650–661, 2009.

[33] C. Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6(2):223–231, 1978.

[34] W. Reisig. *Petri Nets, an introduction.* ETACS Monographs Theorethical Computer Science. Springer-Verlag, 1985.

[35] L. Roditty and U. Zwick. A fully dynamic reachability algorithm for directed graphs with an almost linear update time. In *STOC '04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing, Chicago, IL, June 13-15, 2004*, pages 184–191. ACM, 2004.

[36] R. E. Tarjan. Efficiency of a Good But Not Linear Set Union Algorithm. *Journal of the ACM*, 22(2):215–225, 1975.

[37] M. Thakur and R. Tripathi. Linear connectivity problems in directed hypergraphs. *Theoretical Computer Science*, 410(27-29):2592–2618, 2009.

[38] H. Yen. A unified approach for deciding the existence of certain Petri nets paths. *Information and Computation*, 96(1):119–137, 1992.

[39] M. C. Zhou and F. DiCesare. Parallel and sequential mutual exclusions for Petri net modeling for manufacturing systems with shared resources. *IEEE Transactions on Robotics and Automation*, 7(4):515–527, 1991.