Internal Report 2011-09 August 2011

Universiteit Leiden
Opleiding Informatica

Coverability Tree Constructions

and

Inhibitor Places

Sander van der Vlugt

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

BACHELOR THESIS
August 30, 2011
Supervisor: Dr. Jetty Kleijn
Second reader: Dr. Hendrik Jan Hoogeboom

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Coverability Tree Constructions and Inhibitor Places

Sander van der Vlugt, sandervdvlugt@gmail.com

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University, The Netherlands
Supervisor: Dr. Jetty Kleijn

Abstract. The Coverability Tree is a useful tool for the behavioural analysis of Petri nets. A powerful extension of Petri nets are inhibitor arcs, but these introduce non-monotonicity. Because of this, the classical construction for a Coverability Tree needs to be modified. In this paper it is investigated which constructions are available for Petri nets with inhibitor arcs. For the most obvious modification, it was known that it may not always terminate for Petri nets with 3 or more inhibitor places. Here it is proved that it also does not always terminate in case of 2 inhibitor places. Also a comprehensive overview is presented of what is known concerning the reachability problem and the coverability problem for both Petri nets with and without inhibitor arcs, under the sequential and the (a priori) step semantics. To do this two constructions from [10] are reconsidered. First the construction to reduce step to sequential semantics for Petri nets with a single inhibitor arc is discussed. And next a general construction to reduce Petri nets with weighted (inhibitor) arcs to Petri nets with only unweighted (inhibitor) arcs.

1 Introduction

Petri nets offer a graphical notation for the description of distributed systems. Unlike other modelling languages, such as UML([5]) activity diagrams, Petri nets have an exact mathematical definition, and has a range of mathematical tools for analysis. One of these tools, the Coverability Tree([9]), will be investigated. In particular for two subclasses of Petri nets, PT-nets and PTI-nets. Where PT-nets stands for Place/Transition-net, and a PTI-net is a PT-net with inhibitor arcs.

It is well-known that for all PT-nets a finite coverability tree can be constructed([4]). However, in general for PTI-nets, the classical approach will not work ([3]). Modified constructions for (subclasses of) PTI-nets are available([10, 3]). For one of these constructions, the Modified Coverability Tree from [10], it is known that it does not terminate for all PTI-nets with three inhibitor places. However, it does terminate for all PTI-nets with one inhibitor place. The main goal of this paper was to find if this algorithm terminates for PTI-nets with

exactly two inhibitor places. In Section 6 it is proven that there exists such a PTI-net for which the modified CTC does not terminate.

After a preliminary section, the relation between sequential and step semantics in PTI-nets with a single inhibitor arc will be presented, using a construction to show that reachability can be reduced to fs-reachability. In the next section another construction will prove that simplicity can be considered as a normal form of PTI-nets. In Section 5 the Coverability Tree Construction for Petri nets will be investigated, in particular how it will be affected under the different semantics, i.e. step or sequential. After this section, the influence of inhibitor arcs will be analysed, including an overview of earlier decidability results, depending on certain properties, such as the amount of inhibitor arcs/places and the semantics used.

2 Definitions

2.1 Multisets

A multiset[1] (over a set X) is a function $\mu : X \to \mathbb{N}$, and an *extended multiset* (over X) is a function $\mu : X \to \mathbb{N} \cup \{\omega\}$. Here $\mathbb{N}$ is the set of natural numbers, and ω being the first infinite ordinal. We assume that $\omega + \omega = \omega, \omega - \omega = \omega, n < \omega, n - \omega = 0, 0 \cdot \omega = 0$ and $\omega + n = \omega - n = k \cdot \omega = \omega$, where n is any natural number and k any positive natural number. A multiset may always be considered as an extended multiset. In this paper, X will always be a finite set. We denote $x \in \mu$ if $\mu(x) > 0$. For two extended multisets μ and μ' over X, we denote $\mu \leq \mu'$ (or μ' *covers* μ) if $\mu(x) \leq \mu'(x)$ for all $x \in X$. There is also the multiset $\mathbf{0}$ and the extended multiset $\mathbf{\Omega}$. These are defined as follows: $\mathbf{0}(x) = 0$ and $\mathbf{\Omega}(x) = \omega$ for all x.

2.2 PT-nets

A Place/Transition-net, in short *PT-net* is a tuple $\mathcal{N} = (P, T, W, M_0)$ such that P and T are disjoint finite sets of *places* and *transitions*, respectively, and $W : (T \times P) \cup (P \times T) \to \mathbb{N}$ is the *weight* function of $\mathcal{N}$. M_0 is a multiset over P and denotes the *initial marking* of the net. In diagrams, places are drawn as circles and transitions as rectangles. If $W(x, y) \geq 1$ for some $(x, y) \in (T \times P) \cup (P \times T)$, then (x, y) is an *arc* leading from x to y. As usual, arcs are annotated with their weight if this is 2 or more. A double headed arrow with weight k between p and t indicates that $W(p, t) = W(t, p) = k$. We assume that, for every $t \in T$, there is a place p such that $W(p, t) \geq 1$ or $W(t, p) \geq 1$ (i.e., transitions are never isolated).

A *marking* of $\mathcal{N}$ is a multiset of places. Following standard terminology, given a marking M of $\mathcal{N}$ and a place $p \in P$, we say that $M(p)$ is the number of *tokens* in p. In diagrams, a token is drawn as a small black dot. Multiple tokens in a place can be represented by a number. For a transition t, ${}^\bullet t$ denotes the

[1] Also known as a bag [16].

multiset of places given by ${}^\bullet t(p) \stackrel{def}{=} W(p,t)$ for all places p. And likewise $t^\bullet$ is $t^\bullet(p) \stackrel{def}{=} W(t,p)$ for all places p.

Transitions represent actions which may occur at a given marking and then lead to a new marking. First, we discuss the *sequential semantics*. Formally, t is *enabled* at M, denoted by $M[t\rangle$, if ${}^\bullet t \leq M$. If t is enabled at M, then it can be *fired* leading to the marking $M' = M - {}^\bullet t + t^\bullet$. This can also be written as $M[t\rangle M'$. A firing sequence from a marking M to marking M' in $\mathcal{N}$ is a possibly empty sequence of transitions $\sigma = t_1 \dots t_n$ such that $M = M_0[t_1\rangle M_1 \dots M_{n-1}[t_n\rangle M_n = M'$.

Besides sequentially, the semantics of nets can also be defined in terms of concurrently occuring transitions: *step semantics*. A step of a PT-net $\mathcal{N}= (P,T,W,M_0)$ is a non-empty multiset of transitions, $U : T \to \mathbb{N}$. A step U is enabled, at a marking M if ${}^\bullet U \leq M$. Thus in order for U to be enabled at M, for each place p, the number of tokens in p under M should at least be equal to the accumulated number of tokens needed as input to each of the transitions in U, respecting their multiplicities in U. If U is enabled, it can be executed leading to the marking $M' = M - {}^\bullet U + U^\bullet$, denoted $M[U\rangle M'$.

2.3 PTI-nets

A PTI-net is a PT-net together with a (possibly empty) set of weighted inhibitor arcs leading from places to transitions. A PTI-net $\mathcal{N}$ is specified as a tuple (P,T,W,I,M_0) such that (P,T,W) is a net (the underlying net of $\mathcal{N}$) and I — the inhibitor mapping — is an extended multiset over $P \times T$. M_0 is again the initial marking. If $I(p,t) = k \in \mathbb{N}$, then p is an inhibitor place of t meaning intuitively that t can only be executed if p does not contain more than k tokens; in particular, if $k = 0$ then p must be empty. $I(p,t) = \omega$ means that t is not inhibited by the presence of tokens in p. If I always returns 0 or ω, then we are dealing with a PTI-net with *unweighted* inhibitor arcs, which can only be used to test whether a place is empty or not. In diagrams, inhibitor arcs have small circles as arrowheads. As with the standard PT-net arcs, inhibitor arcs are annotated by their weights. In this case, the weight 0 is not shown, and if $I(p,t) = \omega$, then there is no inhibitor arc at all drawn between p and t. The set of inhibitor places of transition t is given by ${}^\circ t$. Under the sequential semantics for PT-nets, a transition t is enabled at M, again denoted by $M[t\rangle$, if ${}^\bullet t \leq M \leq {}^\circ t$. Thus, if t is enabled at M, then it can be fired leading to the marking $M' = M - {}^\bullet t + t^\bullet$. This can again be written as $M[t\rangle M'$. Also for PTI-nets there are semantics in terms of concurrently occuring transitions. A step U is *a priori enabled*[2] at a marking M of $\mathcal{N}$ if ${}^\bullet U \leq M \leq {}^\circ U$ and again, if U is enabled, it can be executed leading to the marking $M' = M - {}^\bullet U + U^\bullet$, denoted $M[U\rangle M'$.

[2] Besides a priori, there is also a posteriori step-semantics(i.e. used in [3]). Reachability for this coincides with fs-reachability. Lastly there is also an intermediate variation, provided in [19], which is also more restricted than a priori. For a more detailed discussion on semantics in inhibitor nets, see [8]

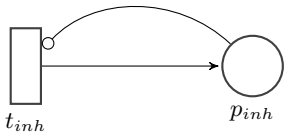


Fig. 1. A PTI-net $\mathcal{N}$

2.4 Reachability and coverability

In the following definitions, keep in mind that a PT-net is a PTI-net with zero inhibitor places, thus these definitions also cover PT-nets.

- A marking M is said to be *reachable* in a PTI-net $\mathcal{N}$ with initial marking M_0, if and only if, there is a step sequence σ, such that $M_0[\sigma\rangle M$. Hence the reachability problem is the problem of deciding, given a marking M in a PTI-net $\mathcal{N}$, is there a step sequence σ, such that $M_0[\sigma\rangle M$ from initial marking M_0.
- A marking M is said to be *fs-reachable* in a PTI-net $\mathcal{N}$ with initial marking M_0, if and only if, there is a firing sequence σ, such that $M_0[\sigma\rangle M$. Hence the fs-reachability problem is the problem of deciding, given a marking M in a PTI-net $\mathcal{N}$, is there a firing sequence σ, such that $M_0[\sigma\rangle M$ from initial marking M_0.
- A marking M is said to be *coverable* in a PTI-net $\mathcal{N}$ with initial marking M_0, if and only if, there is a step sequence σ, such that $M_0[\sigma\rangle M'$, and $M' \geq M$. Hence the coverability problem is the problem of deciding, given a marking M in a PTI-net $\mathcal{N}$, is there a step sequence σ, such that $M_0[\sigma\rangle M'$ from initial marking M_0 such that $M' \geq M$.
- A marking M is said to be *fs-coverable* in a PTI-net $\mathcal{N}$ with initial marking M_0, if and only if, there is a firing sequence σ, such that $M_0[\sigma\rangle M'$, and $M' \geq M$. Hence the coverability problem is the problem of deciding, given a marking M in a PTI-net $\mathcal{N}$, is there a firing sequence σ, such that $M_0[\sigma\rangle M'$ from initial marking M_0 such that $M' \geq M$.

3 Step and sequential semantics with a single inhibitor arc

In this section we take a closer look at one of the constructions in [10]. We will show that the (step-)reachability problem can be reduced to the fs-reachability problem in the case of a unique unweighted inhibitor arc. This means that for each PTI-net $\mathcal{N}$ under the (a priori) step semantics there is a corresponding PTI-net $\mathcal{N}'$ under the sequential semantics, both with a unique unweighted inhibitor arc and a relation ρ between the reachable markings $R_{step}(\mathcal{N})$ and $R_{fs}(\mathcal{N}')$. This means that for all multisets $M : P \to \mathbb{N}$:
$M \in R_{step}(\mathcal{N}) \Leftrightarrow \rho(M) \in R_{fs}(\mathcal{N}')$.

We will first identify three cases. For the third of these cases a construction will be provided. An example of such a construction can be seen in Figure 2 and Figure 3.

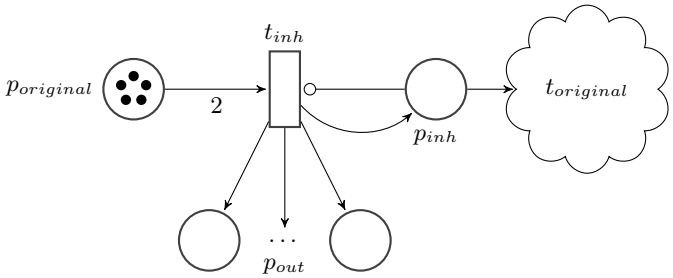


Fig. 2. Example net $\mathcal{N}$

3.1 Situation

Consider PTI-net $\mathcal{N}$ with a single unweighted inhibitor arc, from transition t_{inh} to place p_{inh}. Transition t_{inh} has, like all other transitions, at least one input place or one output place (transitions are never isolated). The following cases concerning p_{inh} and t_{inh} are possible:

- $p_{inh} \in {}^\bullet t_{inh}$
  Thus transition t_{inh} will never be enabled, because either t_{inh} will have no input, or it will be inhibited by p_{inh}. As transition t_{inh} will never fire, it behaves as a PT-net, from this it follows that $\mathcal{N} = \mathcal{N}'$ and $R_{step}(\mathcal{N}) = R_{fs}(\mathcal{N}')$ and thus ρ is the identity.
- $p_{inh} \notin {}^\bullet t_{inh}$ and $p_{inh} \notin t_{inh}{}^\bullet$
  Assume $M[U\rangle$ for some marking M and step U. Then there are two cases, either $M(p_{inh}) > 0$ or $M(p_{inh}) = 0$. In the first case, transition t_{inh} is inhibited, so $U(t_{inh}) = 0$. When p_{inh} is empty however, it may happen that p_{inh} gets filled by the execution of one or more transitions (different from t_{inh}) in U. In that case, U can be sequentialised by executing all firings of t_{inh} first. Therefor again, $\mathcal{N} = \mathcal{N}'$ and $R_{step}(\mathcal{N}) = R_{fs}(\mathcal{N}')$. And again ρ is the identity.
- $p_{inh} \notin {}^\bullet t_{inh}$ and $p_{inh} \in t_{inh}{}^\bullet$
  Only in this case reachability and fs-reachability are not always the same. See, e.g. Figure 1. In this PTI-Net, under the sequential semantics, the only reachable markings are $M(p_{inh}) = 0$ and $M(p_{inh}) = 1$. But, when using a priori step-semantics, $M(p_{inh})$ can be any integer ≥ 1. However, using the construction from the next subsection, it can be seen that reachability can still be reduced to fs-reachability.

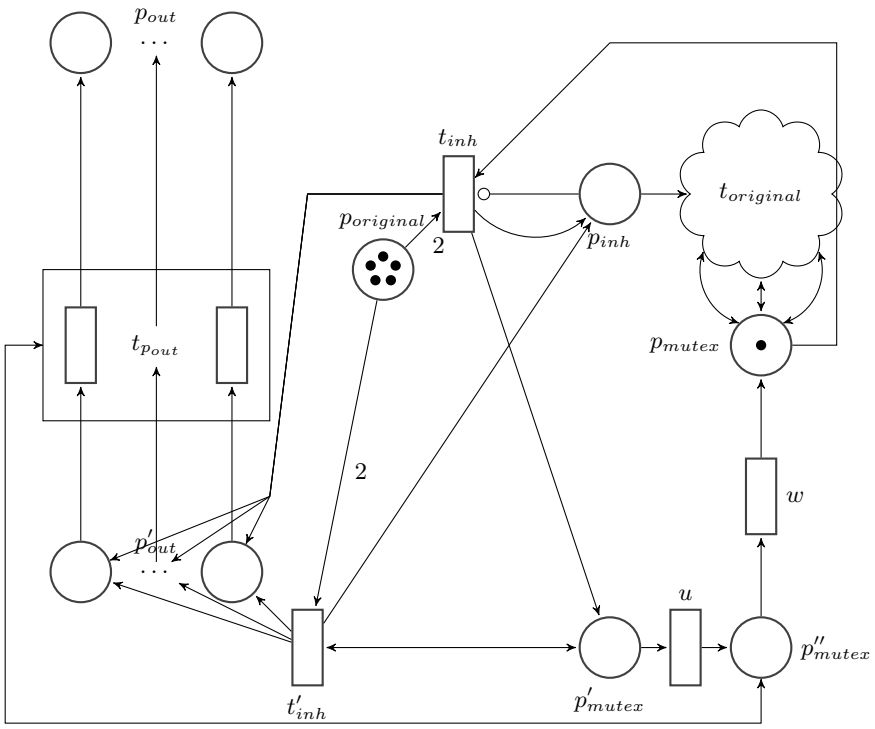


Fig. 3. Constructed net $\mathcal{N}'$

3.2 Construction

For the construction, we start with a PTI-net $\mathcal{N} = (P, T, W, I, M_0)$.Again p_{inh} is the only inhibitor place and (p_{inh}, t_{inh}) the only inhibitor arc. We assume $p_{inh} \notin {}^\bullet t_{inh}$ and $p_{inh} \in t_{inh}{}^\bullet$. Creating $\mathcal{N}' = (P', T', W', I', M_0')$ is as follows:

- P' is defined by retaining all original places and adding three new places: p_{mutex}, p'_{mutex} and p''_{mutex}. Also for every output place p_{out} of t_{inh} (except p_{inh}) a copy p'_{out} is created.
- Also T' is defined by retaining all original places and adding three new transitions: u, w and t'_{inh}. Also for every output place p_{out} of t_{inh} (except p_{inh}) an additional transition $t_{p_{out}}$ is created. From now on when referring to p_{out}, p'_{out} and $t_{p_{out}}$ we refer to the set of these places and transitions.
- The inhibitor mapping I' remains the same as I. No additional inhibitor arcs are added.
- In W', all original arcs from W are remain, except for those from t_{inh} to p_{out} (note that this does not include p_{inh}). These are redirected to p'_{out}.
  These places have an outgoing arc to their respective transition in $t_{p_{out}}$, which in turn have outgoing arc to their respective place in p_{out}.
  The transition t'_{inh} is in many ways a copy of t_{inh}, it is connected to the same input places with the same weight. The output is the same too (thus p'_{out} and p_{inh}), but as mentioned before, it is not inhibited by p_{inh}.
  The *mutex* places are connected as follows: p_{mutex} has an outgoing arc to t_{inh} and double arrows to all other transitions that were in T. It also has an incoming arc from the new transition w. The place p'_{mutex} has an incoming arc from t_{inh}, a double arrow to t'_{inh}, and an outgoing arc to u. Lastly p''_{mutex}, this one has an incoming arc from u, double arrows to all transitions in $t_{p_{out}}$ and an outgoing arc to w.
- The initial marking M_0' is the same as M_0 for all places in P. All other (new) places are empty, except for p_{mutex}, which has a single token.

Given a $M[U\rangle M'$ in $\mathcal{N}$, with $t_{inh} \in U$(that is, $U(t_{inh}) > 0$), the (simultaneous) firing of t_{inh} can always be simulated in $\mathcal{N}'$ by first firing t_{inh} once, this empties p_{mutex}, thus *disabling all original transitions* including t_{inh}. Now p'_{mutex} is marked and hence u is enabled. Moreover, if $U(t_{inh}) > 1$, then also t'_{inh} has become enabled. After t'_{inh} has fired $U(t_{inh}) - 1$ times, u fires, emptying p'_{mutex} and marking p''_{mutex}. Then the various transitions $t_{p_{out}}$ can fire, so that the set of places p_{out} can serve as input for the original transitions again. Lastly, w is fired, filling p'_{mutex} again and giving control back to the original transitions, so that all transitions $t \in U$ except t_{inh} can fire.

Now a marking similar to that of the original net, using a priori step sequence semantics, is reached. This new marking is defined by the relation ρ. For every original marking M, there is a new marking $\rho(M)$, where $M'(P') = M(P)$ for all places $P \cap P'$, $M'(p_{mutex}) = 1$ and all other places are zero. Thus $M_0' = \rho(M_0)$.

So in short $M[U\rangle M' \Rightarrow \rho(M)[t_{inh}{t'}_{inh}{}^x u {t_{p_{out}}}^y w t^z\rangle \rho(M')$. With x being $U(t_{inh}) - 1$, y the number of tokens in the set of places $t_{p_{out}}$ and z the number of transitions $t \in U$ except t_{inh}.

Now for every firing sequence σ, such that $\rho(M_i)[\sigma\rangle\rho(M_{i+1})$ in $\mathcal{N}'$, there is a simulating step sequence in $\mathcal{N}$. If σ is a single transition (not t_inh) then this can be simulated by firing the corresponding transition in $\mathcal{N}$. If t_{inh} is fired, p_{mutex} is emptied, and thus the current marking is not defined by ρ. By firing the following sequence: ${t_{inh}t'_{inh}}^x u {t_{p_{out}}}^y w t^z$, $\mathcal{N}$ can fire ${t_{inh}}^{x+1}$ to simulate this. Note that in $\mathcal{N}'$ it is not required to empty the set of place p'_{out} (by firing $t_{p_{out}}$ y times). However as p'_{out} are new places, no marking in ρ will be reached as long as they are not empty. Thus $\rho(M_i)[{t_{inh}t'_{inh}}^x u w t^z \sigma_0 t_{inh} {t'_{inh}}^x u {t_{p_{out}}}^y w t^z\rangle\rho(M_{i+1})$, where at the first part $t_{p_{out}}$ is not fired at all, followed by a firing sequence σ_0 (which can include t_{inh} again, but p'_{out} is not emptied) finished by another firing of t_{inh} where p'_{out} is finally emptied.

4 Unweighted PTI-net

A PTI-net is said to be *unweighted* if all inhibitor arcs and ordinary arcs are unweighted. The following construction shows that for each PTI-net, an unweighted PTI-net with equivalent reachability and boundedness problems can be constructed. Because of this result, unweightedness can be considered as a normal form of PTI-nets when considering reachability and boundedness. A similar construction was proposed in [10]. The differences between these two constructions will be discussed in more detail at Section 4.2.

4.1 Construction

Let $\mathcal{N}= (P, T, W, I, M_0)$ be a PTI-net. We can assume that for every place p, there is at most one transition t connected to it by an inhibitor arc [3]. The construction consists of two steps. The first removes the weigths from the inhibitor arcs, and the second removes the weights from all other arcs.

4.1.1 Unweighted inhibitor arcs The first step removes the weight of the inhibitor arcs, resulting in the intermediate net $\mathcal{N}' = (P', T', W', I', M'_0)$. For each inhibitor place q, we let inh_q be the weight of the arc attached to it.

- The places P' are copied from P, and for each inhibitor place q, two additional places are added: q_1 and q_2.
- Also for T' all transitions are copied from T, and for each inhibitor place q, two additional transitions, w_q and u_q are added.
- In W', all arcs between places q and transitions t remain the same, except if there is an inhibitor arc between q and t. If this is the case, all (non-inhibiting) arcs are removed between q and t, and these are redirected to q_2 instead. They are also connected in the opposite direction, but still with the

[3] We can always make enough copies of a place retaining the standard connectivity and distribute the inhibitor arcs among them. As a result the number of inhibitor arcs remain the same, but the number of inhibitor places increases.

same weight to q_1.
The two new transitions for each inhibitor place q are connected with unweigthed arcs in the following way: w_q has incoming arcs from q and q_1, and an outgoing arc to q_2. The other transition u_q has an incoming arc from q_2, and outgoing arcs to q_1 and q.

- To get I', all inhibitor arcs from I become unweighted. Thus if $I(p,t) \neq \omega$ then $I'(p,t) = 0$, else $I'(p,t) = \omega$.
- The initial marking M_0' is equal to M_0 for all places that were in P. Every new place q_1 gets inh_q tokens, and every place q_2 remains empty.

Due to how the assistent places q_1 and q_2 are connected, the sum of tokens in q_1 and q_2 will always be inh_q. This means that a maximum of inh_q tokens can be removed from q, in order to allow t to fire. In the case that a place q is both in ${}^\bullet t$ and t°, then the regular (possibly weighted) arc from q to t is redirected to be an arc from q_2 to t. If $W(q,t) > I(p,t)$, then t will still never be enabled, as q_2 will never contain more than $I(p,t)$ tokens. Otherwise, if $W(q,t) < I(p,t)$, then t can still fire if q does not overflow. An example of this can be seen in Figure 4.

In any net $\mathcal{N}$, the firing of a step U can be simulated in $\mathcal{N}'$. As only transition t is changed we will look at a step in the form of $M[t\rangle M'$. This can be simulated by a firing sequence $\rho(M)[{w_q}^x t {u_q}^y\rangle\rho(M')$. Where $x = M(q))$, $y = x - W(q_2,t)$ and ρ is the relation between the reachable markings in $\mathcal{N}$ and $\mathcal{N}'$. This relation is defined as follows:

$\rho(M(p)) = M(p)$ for all $p \in P$
$\rho(M(q_1)) = inh_q$ for all inhibitor places q
$\rho(M(q_2)) = 0$

This means that first w_q is fired until q is empty and if t was enabled in $\mathcal{N}$, q_2 will hold sufficient tokens to fire t. After firing t, transition u_q is fired until q_2 is empty again, and the new marking will be in $\rho(M)$ again.

For the other way around, any step sequence $\rho(M)[U\rangle\rho(M')$ in $\mathcal{N}'$ has a corresponding step sequence $M[\tilde{U}\rangle M'$ in $\mathcal{N}$. A step sequence $\tilde{U}$ is equal to U with all occurences of transitions w_q and u_q removed.

4.1.2 Unweighted arcs The second step removes the weights from the ordinary arcs, resulting in the unweighted PTI-net $\mathcal{N}''= (P'', T'', W'', I'', M_0'')$. For this construction it is required that all inhibitor arcs are unweighted and all inhibitor places inhibit at most one transition. These restrictions can be achieved by using the previous construction.

The second step is essentially the construction from [16], originally proposed in [6]. We define $max_p = max(\{W'(p,t)|t \in T'\} \cup \{W'(t,p)|t \in T'\})$. Now $\mathcal{N}'$ is transformed into $\mathcal{N}''$ by transforming each place p into a ring of max_p places named $p_1, p_2, \ldots p_{max_p}$ (with transitions connecting neighbouring places). Each weighted arc, with weight w between transition t and place p can be represented by w unweighted arcs connecting t with individual places $p_1 \ldots p_w$ from the ring of places representing p. Initially, all tokens are in p_1. Also each inhibitor arc,

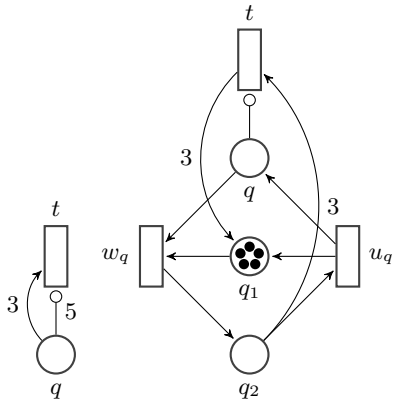


Fig. 4. From $\mathcal{N}$ to $\mathcal{N}'$ with $I(q,t) = 5$ and $W(q,t) = 3$ (fragment)

between place p and transition t, is represented by an inhibitor arc from every place in the ring of places representing p to transition t. An example of a resulting net $\mathcal{N}''$ is shown in Figure 5. We already showed that $\mathcal{N}$ is equivalent with $\mathcal{N}'$. So now we will show that $\mathcal{N}'$ and $\mathcal{N}''$ are equivalent. Now in any $\mathcal{N}'$, the firing of a transition t with $M[t\rangle M'$ can be simulated in $\mathcal{N}''$ as follows: If one or more places q inhibit t, then first all tokens are moved to place q. Then w_q is fired untill this place q is empty. After that, for every input place q_2 from $\mathcal{N}'$, the tokens are distributed over the ring of places representing q_2 in $\mathcal{N}''$, enabling t. After t fires, all tokens are moved to q again.

In any net $\mathcal{N}'$, the firing of a step U can be simulated in $\mathcal{N}''$. As only transition t is changed compared to the construction from [16] we will look at a step in the form of $M[t\rangle M'$. This can be simulated by a firing sequence $\rho(M)[{t_q}^r w_q{}^x {t_{q_2}}^z t {t_{q_2}}^v {t_{q_1}}^w {u_q}^y\rangle\rho(M')$. Where ${t_q}^r$ is the moving of all tokens to q, $x = M(q))$, ${t_{q_2}}^z$ moves the tokens around in the circle representing q_2, ${t_{q_2}}^v$ moves them back to q_2, ${t_{q_1}}^w$ does the same for q_1 and lastly ${u_q}^y$ empties q_2 again. Here ρ is the relation between the reachable markings in $\mathcal{N}'$ and $\mathcal{N}''$. This relation is defined as follows:
$\rho(M(p)) = M(p)$ for all $p \in P'$
$\rho(M(q_1)) = inh_q$ for all inhibitor places q
$\rho(M(q_2)) = 0$
All other new places contain zero tokens too.

Here too we look at the other way around, any step sequence $\rho(M)[U\rangle\rho(M')$ in $\mathcal{N}''$ has a corresponding step sequence $M[\tilde{U}\rangle M'$ in $\mathcal{N}'$. A step sequence $\tilde{U}$ is equal to U with the firing of all transitions that are added in the rings of places excluded.

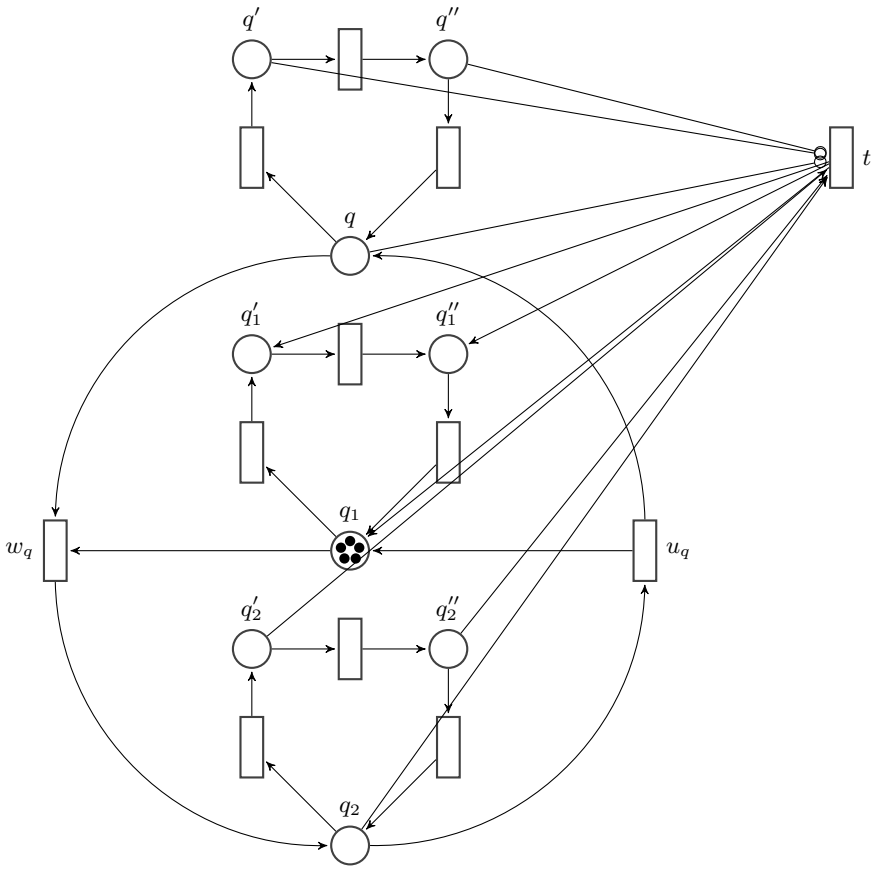


Fig. 5. From $\mathcal{N}'$ to $\mathcal{N}''$

4.2 Discussion

In the second step of the construction, the result of the first step was taken as a prerequisite. This was needed, because each inhibitor place is represented by multiple new places. If the weights on the inhibitor arcs remained the same, these could be 'fooled' by spreading the tokens, and thus staying below the thesholds of the inhibitor arcs. It may be interesting to find a construction that removes the weights from normal arcs, but keeps the inhibitor arcs untouched. Another weakness of the constructions is the increase of inhibitor places. In the first construction this is done by demanding that every place has at most one inhibitor arc. And in the second step this is done because every inhibitor place is being represented by a ring of (inhibitor) places.

When comparing the results as in example in Figure 6 and Figure 5 is the size in number of places and transitions. The new construction creates less additional transitions. Another difference is the intermediate net. Opposed to the original construction, the new construction its intermediate net remains equivalent. This means that the construction can not only be used to construct a unweighted PTI-net, but also to construct a PTI-net with only unweighted inhibitor arcs. Thus making analysis simpler, yet keeping the advantage of readability by using weighted normal arcs.

5 Coverability tree constructions

5.1 Coverability tree

A coverability tree is a tree representation of all possible firing sequences and markings in a PT-net. A node represents a set of markings, while an arc represents the firing of a transition. An arc labeled t from node v to node w, also denoted as $v \xrightarrow{t} w$, means that t can be fired in v, which results in marking w. It was originally introduced in [9], under the name of reachability tree, to be used with Vector Addition Systems, and also proved to be useful for PT-nets[4]. From this tree certain useful properties can be derived [15]:

- A PT-net is *bounded* if and only if ω does not appear in node.
- A PT-net is *k-bounded* if and only if numbers less than or equal to k appear in any node. (*Safe* if $k = 1$)
- A transition in a PT-net is dead if and only if it does not appear in any arc of the tree.
- If marking M (a multiset) is reachable from the initial marking, then there exists a node with label M' that covers M.
- If the PT-net is bounded, the coverability tree coincides with the reachability tree as it contains all possible markings.

5.2 CTC for PT-nets

The algorithm to construct a coverability tree for PT-nets under sequential semantics is shown in Algorithm 1, and is based on the *Karp-Miller construction*

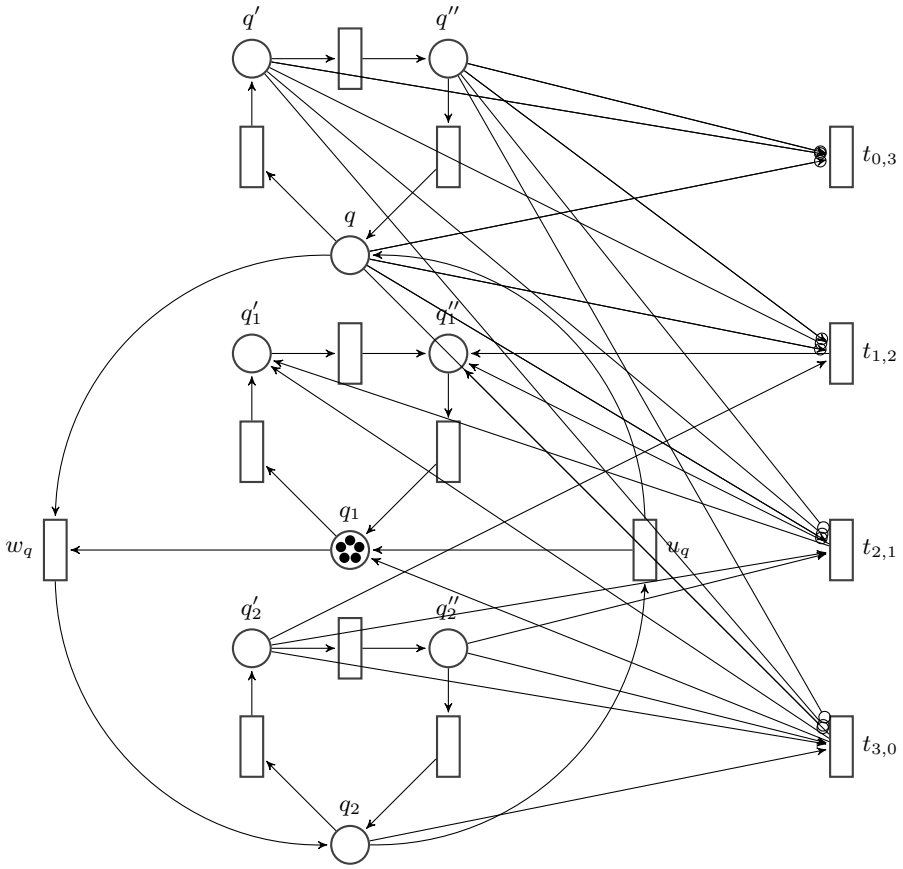


Fig. 6. The resulting net from Figure 4 after the construction of [10].

from [9]. The construction starts with a single node v_0, corresponding to the initial marking. Then, for each transition that is enabled, an arc and a new node representing the resulting marking is added. When this is repeated for each node, it may lead to an infinite tree, therefor some boundaries are needed. First, there can be nodes with markings that correspond with markings appearing earlier in the tree. The successors of this node have already been considered, so it does not need to be done again. In the algorithm this is done in the line: **if** $\mu(v) \notin \mu(V \backslash unprocessed)$.
For nets with unbounded places, another 'trick' is needed. If the marking represented by the new node M covers an ancestor node M', then each $M(p)$ is replaced by ω if $M'(p) < M(p)$.

Algorithm 1: Coverability Tree Construction. Generates a coverability tree of a PT net $\mathcal{N} = (P, T, W, M_0)$

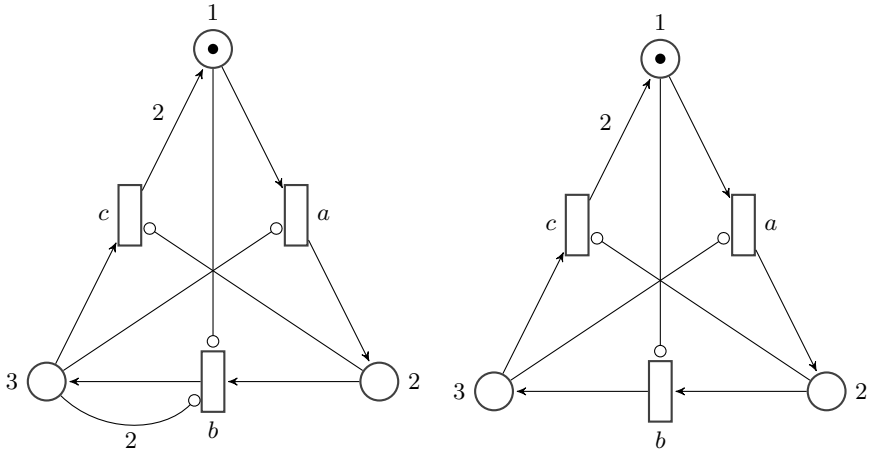
$CT = (V, A, \mu, v_0)$ *where* $V = \{v_0\}, A = \emptyset$ *and* $\mu(v_0) = M_0$
$unprocessed = \{v_0\}$
while $unprocessed \neq \emptyset$ **do**
  let $v \in unprocessed$
  if $\mu(v) \notin \mu(V \backslash unprocessed)$ **then**
    foreach $\mu(v)[t\rangle M$ **do**
      $V = V \uplus \{w\}$ and $A = A \cup \{v \xrightarrow{t} w\}$
      and $unprocessed = unprocessed \cup \{w\}$
      if *there is* u *such that* $u \rightsquigarrow_A v$ *and* $\mu(u) < M$ **then**
        $\mu(w)(p) = (\ \textbf{if}\ \mu(u)(p) < M(p)\ \textbf{then}\ \omega\ \textbf{else}\ M(p))$
      else
        $\mu(w) = M$
  $unprocessed = unprocessed \backslash \{v\}$

5.3 CTC for PTI-nets

When applying this algorithm to PTI-nets, the main problem one encounters is the *non-monotonicity*. Due to this, the algorithm can produce ω's for bounded places, which invalidates most of the useful properties of the coverability tree, mentioned in Section 5.1. In fact, the tree may contain nodes that are not reachable in the actual net. An example is Figure 7(a), which is clearly bounded, but for which the CTC would produce ω's. To strengthen the condition to generate ω-components, it is suggested in [10] to modify the algorithm to Algorithm 2. The new condition makes sure that no inhibitor arc features were used along a path where the number of tokens has grown. As in [10] we will refer to this algorithm from now on as the *modified* CTC. The results of this modification will be discussed in Section 7.

Algorithm 2: *Modified* CTC. Generates a CT of a PTI net $\mathcal{N}=(P,T,W,I,M_0)$

$CT=(V,A,\mu,v_0)$ *where* $V=\{v_0\}, A=\emptyset$ *and* $\mu(v_0)=M_0$
$unprocessed=\{v_0\}$
while $unprocessed\neq\emptyset$ **do**
  let $v\in unprocessed$
  if $\mu(v)\notin\mu(V\backslash unprocessed)$ **then**
    foreach $\mu(v)[t\rangle M$ **do**
      $V=V\uplus\{w\}$ and $A=A\cup\{v\xrightarrow{t}w\}$
      and $unprocessed=unprocessed\cup\{w\}$
      if *there is* u *such that* $u\rightsquigarrow{}^{\sigma}_{A}v$ *and* $\mu(u)<M$ *and such that* $\mu(u)(p)<M(p)$ *implies that* $^\circ t'(p)=\omega$, *for all transitions* t' *in* $\sigma t*$ **then**
        $\mu(w)(p)=(\ \textbf{if}\ \mu(u)(p)<M(p)\ \textbf{then}\ \omega\ \textbf{else}\ M(p))$
      else
        $\mu(w)=M$
  $unprocessed=unprocessed\backslash\{v\}$



(a) PTI-net for which the CTC incorrectly produces ω-components.

(b) PTI-net for which the Modified CTC does not terminate.

Fig. 7. PTI-nets with three inhibitor places.

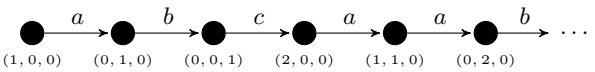


Fig. 8. First six nodes of the (infinite) CT of Figure 7(b) produced by Algorithm 2.

5.4 Step CTC

In [10] another modification of the CTC was introduced: the Step Coverability Tree Construction(SCTC). This algorithm is shown in Algorithm 3. It generates a CT for PT-nets under step semantics. It is similar to Algorithm 1, however as the set of enabled steps can be infinite, it uses *select*(.), to choose a representative subset. Moreover extended markings are compared with the $\sqsubset$ relation, instead of $<$. Both *select* and $\sqsubset$ need to be specified, depending on the subclass of PT(I)-nets that is investigated. In [11] this algorithm is investigated further.

Algorithm 3: Algorithm generating a SCT of a PTI net

$SCT = (V, A, \mu, v_0)$ *where* $V = \{v_0\}, A = \emptyset$ *and* $\mu(v_0) = M_0$
$unprocessed = \{v_0\}$
while $unprocessed \neq \emptyset$ **do**
  let $v \in unprocessed$
  if $\mu(v) \notin \mu(V \backslash unprocessed)$ **then**
    foreach $\mu(v)[U\rangle M$ *with* $U \in select(\mu(v))$ **do**
      $V = V \uplus \{w\}$ and $A = A \cup \{v \xrightarrow{U} w\}$
      and $unprocessed = unprocessed \cup \{w\}$
      if *there is* u *such that* $u \rightsquigarrow {}^{\sigma}_{A} v$ *and* $\mu(u) \sqsubset M$ **then**
        $\mu(w)(p) = (\ \textbf{if}\ \mu(u)(p) < M(p)\ \textbf{then}\ \omega\ \textbf{else}\ M(p))$
      else
        $\mu(w) = M$
  $unprocessed = unprocessed \backslash \{v\}$

6 Modified CTC and two inhibitor places

In [10] it was found that Algorithm 2 would not always terminate for PTI-nets with three inhibitor places. An example of such a net is in Figure 7(b). The initial goal for this article was to find out if Algorithm 2 would always terminate for PTI-nets with two inhibitor places. In the next section the process of constructing a counterexample will be shown. After that the actual counterexample will be presented.

6.1 Constructing a counterexample

Beforehand, it is useful to identify the properties of the net that helped us construct the counterexample. To be a good counterexample, a net needs to comply to at least the following properties:

- The net needs at least one unbounded inhibitor place.
  Motivation: Algorithm 2 only differs from Algorithm 1 for inhibitor places. This means that for bounded inhibitor places, the algorithm will terminate under the same conditions as it would for Algorithm 1.

- The two inhibitor places need to grow simultaneously.
  This follows from the proof in [10] for PTI-nets with one inhibitor place. In short, if they would not grow simultaneously, one of the inhibitor places would generate an ω-component. The remainder of the construction would regard the net as a PTI-net with a single inhibitor place, and terminate.

6.2 A counterexample

Our counterexample will be Figure 9. We will first prove that this net is indeed unbounded. Standard techniques will not work for this, because in general boundedness is undecidable for PTI-nets with two inhibitor places. After this, we will show that the Modified CTC will not terminate for this net.

Proposition 1. *There is a PTI-net with 2 inhibitor places for which the Modified CTC will never terminate.*

Proof. Consider the PTI-net in Figure 9. This net can execute exactly one infinite sequence of transitions $\sigma = \sigma_0\sigma_1\sigma_2\ldots$, where $\sigma_i = a^{2^i}cb^{2^i}d$ for every $i \geq 0$. For any such a σ_i, the firing of transitions a, c and d does not change the total amount of tokens. The firing of transition b does change the total amount of tokens, in fact it doubles it by taking tokens from place 2, and placing twice that amount in place 1. In σ_{i+1} all tokens from place 1 are moved to place 2 again, therefor both place 1 and place 2 are unbounded, and thus the net is unbounded.

Now we will show that the Modified CTC will not terminate. In Figure 10, a part of the execution of the PTI-net from Figure 9 is shown. It starts with a marking of x tokens in place 1, and one token in place 3 ($x \in \mathbb{N}$ and $x > 0$). There are three nodes covering ancestor nodes. The first node, $(x, 0, 1, 0)$, is being covered by the node $(2x, 0, 1, 0)$ and the node $(2x - k, k, 1, 0)$ with $0 < k \leq x$. Between these nodes, both transition c and d fire. As both transition c and d are inhibited at the second node, the line marked with an $*$ in Algorithm 2 is false, and there will be no ω produced for this node. The second node, $(x - i, i, 1, 0)$ is being covered by the node,$(2x - k, k, 1, 0)$. Between these nodes, both transition c and d fire. As both transition c and d are inhibited at the second node, the line marked with an $*$ in Algorithm 2 is false, and there will be no ω produced for this node. The third node, $(0, x, 1, 0)$, is being covered by the node $(2x - k, k, 1, 0)$ with $x \leq k < 2x$ and the node $(0, 2x, 1, 0)$. Between these nodes, both transition c and d fire. As both transition c and d are inhibited at the fifth node, the line marked with an $*$ in Algorithm 2 is false, and there will be no ω produced for this node.

The ninth node is in fact a repetition of the third node, as $2x \in \mathbb{N}$ and $2x > 0$, $2x$ can be replaced by x again. Thus the possible coverings will continue like this, with transitions c and d being fired between any two $<$-comparable markings. In our PTI-net in Figure 9, x is set equal to 1, as $1 \in \mathbb{N}$ and $1 > 0$, the Modified CTC will never generate any ω components in this net, and thus never terminates. □

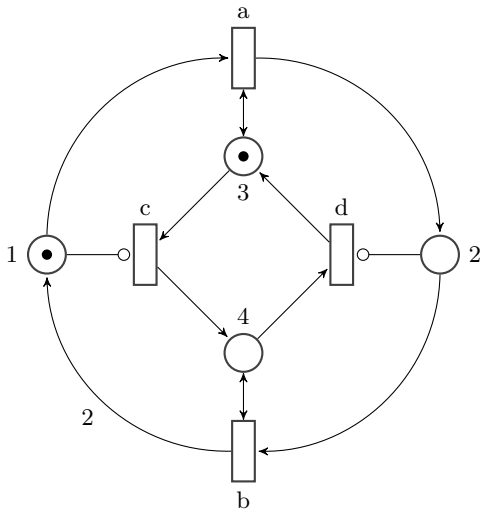


Fig. 9. A PTI-net with 2 inhibitor places for which the Modified CTC does not terminate.

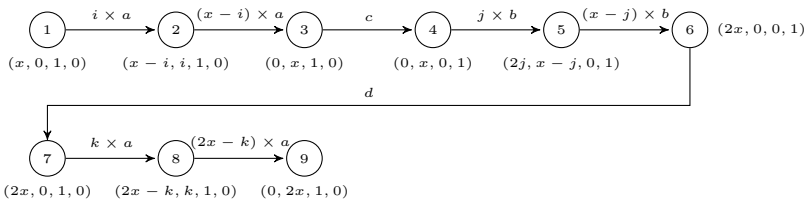


Fig. 10. Execution of the PTI-net in Figure 9. With $0 < i < x$ and $0 < j < x$ and $0 < k < 2x$.

7 Overview of decidability results

Since the 70's, a lot of research has been done concerning PT(I)-nets and their analytical tools. In this section a comprehensive overview will be given on the decidability of both the reachability problem and the coverability problem. The overview will first be divided by semantics (firing sequence or step) and then we will explore the trade-off between decidability and expressiveness. For example the ability to test for emptiness makes it possible to model more complex systems, but certain properties will become undecidable.

The decidability of both reachability and coverability have their own uses. They can both be used to check whether certain (unwanted) situations can arise. With reachability this can be a precise marking (configuration of the system), while coverability investigates more general behaviour, such as boundedness of a place.

Recall that a CT is a tree labeled with extended markings. All labels are coverings of reachable markings, and ω is only introduced when a place is unbounded. The decidability of coverability will be split in two issues: first whether there is an algorithm that generates a *finite* CT (and thus all labels cover reachable markings). Then whether the algorithm produces a *complete* CT, where all reachable markings are covered. When an algorithm is guaranteed to produce a finite CT, we know that the algorithm will always terminate, and that the resulting CT tells us if the *net* is bounded. If the resulting CT is a complete CT, then there is an algorithm that produces not only a finite CT, but also provides full information about unboundedness for *each place* of a PT(I)-net.

As we have seen in Section 4, a construction is available to remove the weight from inhibitor arcs. Therefor we assume that all inhibitor arcs are unweighted. All results will be presented in Table 1.

7.1 Sequential semantics

The most elementary semantics of PT-nets were based on sequences of firing single transitions, thus generating a firing sequence [16].

An overview of the results for sequential semantics can be found in Table 1(a), and will now be discussed.

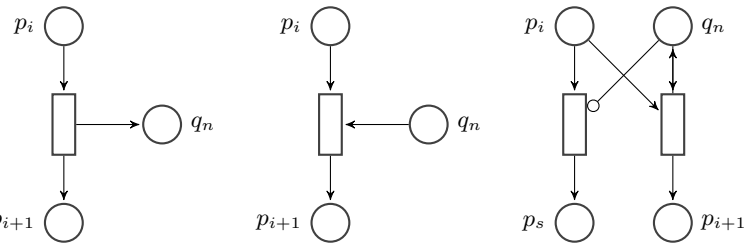
7.1.1 Coverability and reachability for PT-nets One of the first notable results in 1976 was the proof that the Karp-Miller construction [9] intended for Vector Addition Systems was also useful for Petri nets. In [7] it was proven that this Coverability Tree Construction indeed resulted in a finite CT and that any marking reachable in a PT-net, would also be covered by a node in the resulting tree, and thus a complete CT. We discussed this in more detail in Section 5.

The decidability of the reachability problem turned out to be harder to establish. It remained one of the most important open questions, until 1981, when it was proven to be decidable in [12].

7.1.2 Reachability in PTI-nets With the introduction of inhibitor arcs in [2], leading to PTI-nets, new issues arised. It was originally proved in [1] that these nets are equivalent to Turing machines, but this could not be traced by the author of this thesis. However, the equivalence is easy to show in terms of register machines, introduced in [18]. A register machine exists of a finite number of registers and a programming unit. This programming unit executes a program consisting of a sequence of instructions. The number of registers is finite, and each can store any natural number. The programming unit can inspect the registers, but it can only discern whether a register is empty or not. Using results from [14], it was found in [18] that Turing machines can be simulated by a register machine with only three basic instructions and two registers. The three instructions are $P(n)$, "Increase register n by 1"; $D(n)$, "Decrease register n by 1"; and $J(n)[s]$, "Jump to instruction s if register n is 0". In [16] it is shown how these instructions can be converted into fragments of a PTI-net, see Figure 11. Places p_i, p_{i+1} and p_s correspond to instructions i, $i+1$ and s. The place q_n is register n (with $n = 1, 2$). Thus a full net consists of a sequence of instructions p_i (identification). Initially the registers q_1 and q_2 are empty and only p_1 is marked with a single token. Only the $J(n)[s]$ type instructions use an inhibitor arc. As there are only two registers needed, two inhibitor places are sufficient to reach Turing equivalence, as every Turing machine can be modeled in this way. As a consequence of this, reachability is undecidable for PTI-nets with two or more inhibitor places.

In [17] it has been proven, using semilinear sets, that fs-reachability is decidable for PTI-nets with only one inhibitor arc. Using the construction from [10, Theorem 1], it can be seen that every PTI-net with a single inhibitor *place* can be transformed into a PTI-net with a single inhibitor *arc* with equivalent fs-reachability. Thus fs-reachability for PTI-nets with a single inhibitor place is decidable.

Fig. 11. Instructions of a register machine.



(a) Instruction $P(n)$ of a register machine.

(b) Instruction $D(n)$ of a register machine.

(c) Instruction $J(n)[s]$ of a register machine.

7.1.3 Coverability in PTI-nets In [10], the Modified CTC (here Algorithm 2) is presented to construct a CT for PTI-nets. It was proven that for PTI-nets with one inhibitor place this construction always constructs a finite CT, although it does not always produce a complete CT. Thus the boundedness of PTI-nets with one inhibitor place is decidable, even though this algorithm is not always able to find all unbounded places.

For PTI-nets with three inhibitor places this algorithm does not terminate as shown in [10]. A counterexample can be seen in Figure 7(b). In Section 6 of this paper it is shown that there are also PTI-nets with two inhibitor places for which the MCTC does not terminate. An example of such a net is Figure 9.

That the MCTC does not always result in a complete CT in the case of one inhibitor place, does not proof that there is no algorithm at all that creates a complete CT for PTI-nets with a single inhibitor place. Therefor this question remains open in Table 1(a).

7.2 Step semantics

When considering step-semantics a lot of the analytical tools that are available for PTI-nets become invalid. Recall Figure 1, even for this simple PTI-net the modified CTC creates a CT which is infinite in breadth. To obtain decidability results new tools are needed, or a reduction from the results of the sequential semantics needs to be found.

Note that properties that are decidable under step-semantics are also decidable under sequential semantics. Because any PT(I)-net under sequential semantics can be simulated by a PT(I)-net using step-semantics by adding an additional runplace. This runplace contains a single token and is connected to every transition with a double arrow. This ensures that every transition can only fire once, just like it would under sequential semantics, resulting in corresponding reachability and coverability problems. As a direct result, if reachability is undecidable under sequential semantics, it is also undecidable under step-semantics.

An overview of the results for step-semantics can be found in Table 1(b).

7.2.1 Coverability and reachability in PT-nets In [10], the concept of a step coverability tree construction is introduced which has been further specialised in [11]. This SCTC generates a step coverability tree, which is finite and in which every step sequence (and thus also every reachable marking) can be retraced. If not exactly, then at least through a covering step sequence, thus we have a complete SCT.

When no inhibitor arcs are involved, any marking reachable under the step-semantics, is also reachable under the sequential semantics. This is because any enabled step, can be sequentialised to a firing sequence, and vice versa any firing sequence can be simulated by a step sequence. As a result, the reachability problems are equal and therefor the reachability problem for PT-nets is decidable.

7.2.2 Reachability in PTI-nets Recall from Section 7.1.3 that for PTI-nets with one inhibitor place the fs-reachability problem can be reduced to the decidable fs-reachability problem with one inhibitor arc. Unfortunately, the construction used to prove this reduction does not work under the step semantics. Therefor it is still unclear whether the reachability problem is decidable in the case of a single inhibitor place. We do know that, as seen in Section 3, the reachability problem under the step semantics for PTI-nets with a single inhibitor arc can be reduced to the fs-reachability problem for PTI-nets with a single inhibitor arc, and therefor is decidable.

For PTI-nets with two or more inhibitor places, it is certain that anything that is fs-reachable is also step-reachable. This is because every firing sequence can be simulated by steps of size one, as this step size can be enforced by adding a runplace, as mentioned in Section 7.2. It follows from Section 7.1.2 that a PTI-net with two inhibitor places can simulate a Turing machine, making fs-reachability undecidable. Hence, reachability is undecidable under the step-semantics for PTI-nets with two or more inhibitor places.

7.2.3 Coverability in PTI-nets As with reachability, we do not know if a single inhibitor place can be reduced to a single inhibitor arc. We do know however, as seen in Section 3, that any PTI-net with one inhibitor arc under the step semantics can be transformed into a PTI-net with one inhibitor arc under the sequential semantics with equivalent reachability and boundedness problems. For these nets the Modified CTC is applicable, and the resulting tree will be a finite CT, which can be used to detect net boundedness. Like in Section 7.1.3, it is not known whether there exists an algorithm that can construct a complete CT. Therefor both finite SCT and complete SCT are unknowns in Table 1(b), with the note that finite (S)CT is decidable for a single inhibitor arc.

As under the sequential semantics, it is not possible to create a finite SCT for PTI-nets with two or more inhibitor places, for the same reasons reachability is undecidable under step semantics too.

8 Conclusion

The initial goal of this paper was to answer an open problem from [10]: to find an example of a PTI-net with two inhibitor places for which the modified coverability tree construction would never terminate. To gain a better understanding of PTI-nets, we started with two constructions from [10].

The first construction from step-reachability to fs-reachability in case of a single inhibitor arc was merely clarified. The second construction removed the weights from PTI-nets for both inhibitor arcs as regular arcs. We found an alternative construction, which had the advantage of a clear separation of concerns. The construction was split in two parts. One that removed the weights from the inhibitor arcs, and one that removed the weights from the remaining arcs. The second part relied on properties which could be achieved by the first part, but

Table 1. Decidability of Reachability and Coverability for PTI-nets

(a) Sequential semantics

#Inhibitor places	0	1	>= 2
Reachability	Yes[12]	Yes[17]	No[3]
Finite CT[A]	Yes[7]	Yes[10]	No[2]
Complete CT[A]	Yes[7]	???[10]	No[10]

[1] Yes: MCTC can produce it.
???: MCTC cannot produce it, but other algorithms might.
No: It cannot be done.
[2] See Section 6.
[3] See Section 7.1.2.

(b) Step semantics

#Inhibitor places	0	1	>= 2
Reachability	Yes[2]	???[4]	No[3]
Finite SCT[1]	Yes[11]	???[4]	No[5]
Complete SCT[1]	Yes[11]	???[10]	No[10]

[1] Yes: MCTC or SCTC can produce it.
???: MCTC and SCTC cannot produce it, but other algorithms might.
No: It cannot be done.
[2] See Section 7.2.1 and [12].
[3] See Section 7.2.2 and [13].
[4] Decidable with a single arc. See Section 7.2.1, Section 7.2.3 and [10].
[5] See Section 6.

the intermediate net held the same reachability and boundedness properties of the original weighted net.

After this we investigated several coverability tree construction algorithms for both PT-nets (sequential and step semantics) and PTI-nets (sequential semantics).

Finally we zoomed in on the modified CTC. This straightforward approach to generate a coverability tree for PTI-nets gave some information on the boundedness of a PTI-net with a single inhibitor place ([10]). In Section 6 an example was given demonstrating that this construction may not terminate for PTI-nets with no more than two inhibitor places, as was the original aim of this paper.

An interesting subclass of PTI-nets that we have not discussed before are *Primitive* PTI-nets. These nets, introduced in [3], have the following constraint: it is possible to associate a threshold value with each inhibiting place, in such a way that, if the number of tokens in the place exceeds this threshold at some stage of the computation, then that place will never be succesfully tested for the absence of tokens, because it cannot be emptied any more. In [3] this threshold is called the emptiness limit. In [3] several properties of these nets are proven under the sequential semantics. One of these properties is that a finite and complete coverability tree can be computed. Also for every primitive PTI-net it is possible to construct a PT-net with equivalent reachability problems. In [10] this subclass is investigated further, in particular under the a priori step semantics. It is proven that the step coverability tree construction can, if the correct *select* and $\sqsubset$ relations are instantiated, produce a finite and complete SCT for Primitive PTI-nets. As PT-nets can be considered as Primitive PTI-nets with an emptiness limit of -1, the SCTC can also produce a finite and complete SCT for PT-nets with the same instantiation. So far, this is also the largest subclass of PTI-nets to which the STCT has been applied. Unfortunately, primitivity itself is not

decidable[3], but can often be guaranteed by construction. In particular PTI-nets with bounded and complemented inhibitor places are primitive.

As can be seen in Table 1 there are still some open questions, most involving the gap between a single inhibitor arc and a single inhibitor place. Under the sequential semantics reachability and net-boundedness are both decidable for these two cases. However, this is not enough to solve the coverability problem. To find the answer to this problem, the boundedness problem must be decidable for every place in the net. When considering step-semantics, the gap only seems to become bigger. Under these semantics, none of the properties of PTI-nets with a single inhibitor place are known so far.

References

1. Agerwala, T.: A Complete Model for Representing the Coordination of Asynchronous Processes. Tech. rep., Johns Hopkins University (1974)
2. Agerwala, T., Flynn, M.: Comments on Capabilities, Limitations and "Correctness" of Petri Nets. In: ISCA '73: Proceedings of the 1st annual symposium on Computer architecture. pp. 81–86. ACM, New York, NY, USA (1973)
3. Busi, N.: Analysis Issues in Petri Nets with Inhibitor Arcs. Theoretical Computer Science 275(1-2), 127 – 177 (2002)
4. Finkel, A.: A Generalization of the Procedure of Karp and Miller to Well Structured Transition systems. In: 14th International Colloquium on Automata, languages and programming. pp. 499–508. Springer-Verlag, London, UK (1987)
5. Fowler, M., Scott, K.: UML distilled (2nd ed.): a brief guide to the standard object modeling language. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (2000)
6. Hack, M.: Decision Problems for Petri Nets and Vector Addition Systems. Cambridge, Mass.: MIT, Project MAC, TR-59 (1975)
7. Hack, M.: Decidability Questions for Petri Nets. Tech. rep., Massachusetts Institute of Technology, Cambridge, MA, USA (1976)
8. Janicki, R., Koutny, M.: Semantics of Inhibitor Nets. Information and Computation 123(1), 1 – 16 (1995)
9. Karp, R.M., Miller, R.E.: Parallel Program Schemata. Journal of Computer and System Sciences 3(2), 147 – 195 (1969)
10. Kleijn, J., Koutny, M.: Steps and Coverability in Inhibitor Nets. In: Lodaya, K., Mukund, M., Ramanujam, R. (eds.) Perspectives in Concurrency Theory. pp. 264–295. Universities Press, Hyderabad (2008)
11. Kleijn, J., Koutny, M.: Applying Step Coverability Trees to Communicating Component-Based Systems. In: Arbab, F., Sirjani, M. (eds.) Fundamentals of Software Engineering, Lecture Notes in Computer Science, vol. 5961, pp. 178–193. Springer Berlin / Heidelberg (2010)
12. Mayr, E.W.: An Algorithm for the General Petri Net Reachability Problem. In: STOC '81: Proceedings of the thirteenth annual ACM symposium on Theory of computing. pp. 238–246. ACM, New York, NY, USA (1981)
13. Minsky, M.L.: Computation: Finite and Infinite Machines. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1967)
14. Minsky, M.L.: Recursive unsolvability of Post's problem. Lexington, Mass. : Massachusetts Institute of Technology, Lincoln Laboratory (1960)

15. Murata, T.: Petri nets: Properties, Analysis and Applications. In: Proceedings of the IEEE, Apr 1989, Volume: 77 Issue:4. pp. 541 – 580 (1989)
16. Peterson, J.L.: Petri Net Theory and the Modeling of Systems. Prentice Hall PTR, Upper Saddle River, NJ, USA (1981)
17. Reinhardt, K.: Reachability in Petri Nets with Inhibitor Arcs. Tech. rep., Wilhelm-Schickhard Institut für Informatik, Universität Tübingen (1996)
18. Shepherdson, J.C., Sturgis, H.E.: Computability of recursive functions. J. ACM 10(2), 217–255 (1963)
19. Vogler, W.: Partial Order Semantics and Read Arcs. Theor. Comput. Sci. 286(1), 33–63 (2002)