Peter Habermehl[1], Roland Meyer[1], and Harro Wimmel[2]

[1] LIAFA, Paris Diderot University & CNRS
{peter.habermehl,roland.meyer}@liafa.jussieu.fr
[2] Department of Computing Science, University of Rostock
harro.wimmel@uni-rostock.de

Abstract. We show that the downward-closure of a Petri net langua
is effectively computable. This is mainly done by using the notions
fined for showing decidability of the reachability problem of Petri ne
In particular, we rely on Lambert's construction of marked graph tra
sition sequences — special instances of coverability graphs that allow
to extract constructively the simple regular expression corresponding
the downward-closure. We also consider the remaining language ty
for Petri nets common in the literature. For all of them, we provide al
rithms that compute the simple regular expressions of their downwa
closure. As application, we outline an algorithm to automatically anal
the stability of a system against attacks from a malicious environme

1 Introduction

Petri nets or the very similar vector addition systems are a popular fun
model for concurrent systems. Deep results have been obtained in
theory, among them and perhaps most important decidability of the rea
problem [6,10,8], whose precise complexity is still open.

Petri nets have also been studied in formal language theory, and se
tions of Petri net languages have been introduced. The standard notion
we simply refer as *Petri net language* accepts sequences of transition
a run from an initial to a final marking. Other notions are the *prefix*
considering all markings to be final, the *covering language* where seque
ing to markings that dominate a given final marking are accepted, and
languages where all sequences leading to a deadlock are computed.

We study the *downward-closure* of all these languages wrt. the sub
dering [4]. It is well known that given a language L over some finite
its downward-closure is regular; it can always be written as the comp
an upward-closed set, which in turn is characterised by a finite set o
elements. Even more, downward-closed languages correspond to *simp*
expressions [1]. However, such an expression is not always effectively co
This depends on L. For example, the reachability set of lossy channel s
downward-closed but not effectively computable [11], even though me

* The first authors were supported by the French ANR projects Averiss an

S. Abramsky et al. (Eds.): ICALP 2010, Part II, LNCS 6199, pp. 466–477, 2010.

is *effectively computable*. This is done by a careful inspection of the *decidability of the reachability problem* due to Lambert [8]. From his perfect marked graph transition sequences (MGTS) we directly extract ple regular expression corresponding to the downward-closure of the Key to this is an iteration argument that employs Lambert's pumpin for Petri nets and the particular structure of MGTS in a non-trivial w

We also establish computability of the downward-closure for the r language types. For terminal languages we rely on the previous result for covering and prefix languages we directly construct the expressions coverability tree of the Petri net.

To be able to compute the downward-closure of a language is impo several reasons. For example, it is precisely what an environment obse a language in an asynchronous interaction. A component which period serves the actions (or alternatively states) of another process will se the downward-closure of the language of actions the partner issues. application of the downward-closure of a language is the use as a regula proximation of the system behaviour, allowing for safe inclusion checks a Petri net language and all types of languages for which inclusion of language (or even only simple regular expressions) is decidable.

We apply our results to automatically analyse the stability of a syster attacks. Consider a malicious environment that tries to force the syster undesirable state. Then the downward-closure of the environment's provides information about the intrusions the system can tolerate.

The paper is organised as follows. In Section 2, we provide prelimina tions concerning Petri nets, languages, and downward-closed sets. In S we state our main result. The downward-closure of Petri net langua fectively computable. In Section 4, we investigate the other language Section 5 we illustrate an application of our result before concluding in S

2 Petri Nets and Their Languages

Petri nets generalise finite automata by distributed states and explicite nisation of transitions. A *Petri net* is a triple (P, T, F) with finite and sets of *places* P and *transitions* T. The *flow function* $F : (P \times T) \cup (T \times$ determines the mutual influence of places and transitions.

States of Petri nets, typically called *markings*, are functions $M \in$ assign a natural number to each place. We say that a place p *has* k *tok* M if $M(p) = k$. A marking M *enables* a transition t, denoted by M places carry at least the number of tokens required by F, i.e., $M(p) \geq I$ all $p \in P$. A transition t that is enabled in M may be *fired* and yields M' with $M'(p) = M(p) - F(p,t) + F(t,p)$ for all $p \in P$. The firing r extended inductively to transition sequences $\sigma \in T^*$.

$\preceq_\omega \subseteq \mathbb{N}_\omega^P \times \mathbb{N}_\omega^P$ defines the precision of ω-markings. We have M
$M(p) = M'(p)$ or $M'(p) = \omega$.

To adapt the firing rule to ω-markings, we define $\omega - n := \omega =:$
any $n \in \mathbb{N}$. The relation defined above can now be applied to ω-mark
firing a transition will never increase or decrease the number of tokens f
p with $M(p) = \omega$. An ω-marking M' is *reachable* from an ω-markin
net N if there is a firing sequence leading from M to M'. We denote
ω-markings reachable from M by $\mathcal{R}(M)$.

Definition 1. *The* reachability problem **RP** *is the set*

$$\mathbf{RP} := \{(N, M, M') \mid N = (P, T, F), M, M' \in \mathbb{N}_\omega^P, \text{ and } M' \in \mathcal{R}($$

The reachability problem **RP** is known to be decidable. This was fir
by Mayr [9,10] with an alternative proof by Kosaraju [6]. In the '90s,
[8] presented another proof, which can also be found in [13].

To deal with reachability, reachability graphs and coverability gra
introduced in [5]. Consider $N = (P, T, F)$ with an ω-marking $M_0 \in$
reachability graph R of (N, M_0) is the edge-labelled graph $R = (\mathcal{R}(M$
where a t-labelled edge $e = (M_1, t, M_2)$ is in E whenever $M_1[t\rangle M_2$.

A *coverability graph* $C = (V, E, T)$ of (N, M_0) is defined inductively.
is in V. Then, if $M_1 \in V$ and $M_1[t\rangle M_2$, check for every M on a path fro
M_1 if $M \leq M_2$. If the latter holds, change $M_2(s)$ to ω whenever $M_2(s)$
Add, if not yet contained, the modified M_2 to V and (M_1, t, M_2) to
procedure is repeated, until no more nodes and edges can be added.

Reachability graphs are usually infinite, whereas coverability grap
ways finite. But due to the inexact ω-markings, coverability graphs do
for deciding reachability. However, the concept is still valuable in dea
reachability, as it allows for a partial solution to the problem. A ma
is not reachable if there is no M' with $M' \geq M$ in the coverability g
a complete solution of the reachability problem, coverability graphs n
extended as discussed in Section 3.

Our main contributions are procedures to compute representation
net languages. Different language types have been proposed in the litera
we shall briefly recall in the following definition [12].

Definition 2. *Consider a Petri net $N = (P, T, F)$ with initial and fi
ings $M_0, M_f \in \mathbb{N}^P$, Σ a finite alphabet, and $h \in (\Sigma \cup \{\epsilon\})^T$ a labelli
extended homomorphically to T^*. The* language *of N accepts firing seq
the final marking:*

$$\mathcal{L}_h(N, M_0, M_f) := \{h(\sigma) \mid M_0[\sigma\rangle M_f \text{ for some } \sigma \in T^*\}.$$

We write $\mathcal{L}(N, M_0, M_f)$ if h is the identity. The prefix language *o
accepts all transition sequences:*

$$\mathcal{P}_h(N, M_0) := \{h(\sigma) \mid M_0[\sigma\rangle M \text{ for some } \sigma \in T^* \text{ and } M \in \mathbb{N}^P$$

$\mathcal{T}_h(N, M_0) := \{h(\sigma) \mid M_0[\sigma\rangle M$ *with* $\sigma \in T^*, M \in \mathbb{N}^*$, *and* M *is a de*

The covering language *requires domination of the final marking:*

$$\mathcal{C}_h(N, M_0, M_f) := \{h(\sigma) \mid M_0[\sigma\rangle M \geq M_f \text{ for some } \sigma \in T^* \text{ and } M$$

Note that the prefix language $\mathcal{P}_h(N, M_0)$ is the covering language of the
that assigns zero to all places, $\mathcal{P}_h(N, M_0) = \mathcal{C}_h(N, M_0, \mathbf{0})$.

We are interested in the downward-closure of the above languages
subword ordering $\preceq \subseteq \Sigma^* \times \Sigma^*$. The relation $a_1 \dots a_m \preceq b_1 \dots b_n$ requ
$a_1 \dots a_m$ to be embedded in $b_1 \dots b_n$, i.e., there are indices $i_1, \dots, i_m \in \{$
with $i_1 < \dots < i_m$ so that $a_j = b_{i_j}$ for all $j \in \{1, \dots, m\}$. Given a
L, its downward-closure is $L\downarrow := \{w \mid w \preceq v$ for some $v \in L\}$. A do
closed language is a language L such that $L\downarrow = L$. Every downwa
language is regular since it is the complement of an upward-closed s
can be represented by a finite number of minimal elements with resp
This follows from the fact that the subword relation is a well-quasi-or
words [4]. More precisely, every downward-closed set can be written as
regular expression over Σ (see [1]): We call an atomic expression an
expression e of the form $(a + \epsilon)$ where $a \in \Sigma$, or of the form $(a_1 + \cdot$
where $a_1, \dots, a_m \in \Sigma$. A product p is either the empty word ϵ or a finite
$e_1 e_2 \dots e_n$ of atomic expressions. A simple regular expression is then e
a finite union $p_1 + \dots + p_k$ of products.

3 Downward-Closure of Petri Net Languages

Fix a Petri net $N = (P, T, F)$ with initial and final markings $M_0, M_f \in$
labelling $h \in (\Sigma \cup \{\epsilon\})^T$. We establish the following main result.

Theorem 1. $\mathcal{L}_h(N, M_0, M_f)\downarrow$ *is computable as (♣) below.*

Recall that any downward-closed language is representable by a simpl
expression [1]. We show that in case of Petri net languages these ex
can be computed effectively. In fact, they turn out to be rather natu
correspond to the transition sets in the precovering graphs of the ne
this, we shall need some insight into the decidability proof for reach
Petri nets. We follow here essentially the presentation given in [14] fo
the infinity problem of intermediate states in Petri nets.

3.1 A Look at the Decidability of RP

We present here some main ideas behind the proof of decidability of **R**
ing to Lambert [8,13]. The proof is based on marked graph transition s
(MGTS), which are sequences of special instances of coverability grap
ternating with transitions t_j of the form $G = C_1.t_1.C_2 \dots t_{n-1}.C_n$. The

and the *output* $m_{i,out}$. The *initial marking* M_i of C_i is less concrete th
and output, $m_{i,in} \preceq_\omega M_i$ and $m_{i,out} \preceq_\omega M_i$. The transitions t_1, ..., t
MGTS connect the output $m_{i,out}$ of one precovering graph to the inpu
of the next, see Figure 1.

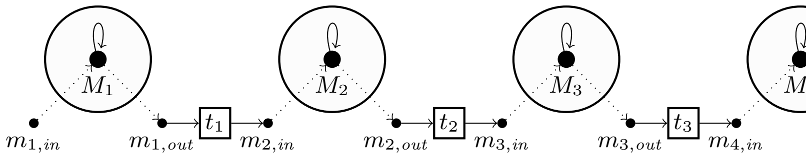

Fig. 1. A marked graph transition sequence $C_1.t_1.C_2 \ldots t_3.C_4$. Dots represent
and circles represent strongly connected precovering graphs with in general
one node. The initial marking is depicted in the center. Solid lines inside th
are transition sequences that must be firable in the Petri net. Dotted lines
entry to and exit from precovering graphs, which do not change the actual
the Petri net. Both $m_{i,in} \preceq_\omega M_i$ and $m_{i,out} \preceq_\omega M_i$ hold for every i.

A *solution* of an MGTS is by definition a transition sequence leading
the MGTS. In Figure 1 it begins with marking $m_{1,in}$, leads in cycles thr
first precovering graph until marking $m_{1,out}$ is reached, then t_1 can fire
$m_{2,in}$, from which the second coverability graph is entered and so on,
MGTS ends. Whenever the marking of some node has a finite value
place, this value *must be reached exactly* by the transition sequence. If
is ω, there are no such conditions. The *set of solutions* of an MGTS G i
by $\mathcal{L}(G)$ [8, page 90].

An instance $RP = (N, M_0, M_f)$ of the reachability problem can be fo
as the problem of finding a solution for the special MGTS G_{RP} depicte
ure 2. The node $\boldsymbol{\omega}$ (with all entries ω) is the only node of the coverability g
we allow for arbitrary ω-markings and firings of transitions between m_1
and $m_{1,out} = M_f$, but the sequence must begin exactly at the (concre

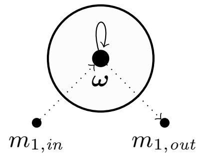

Fig. 2. MGTS representation of an instance $(N, m_{1,in}, m_{1,out})$ of the reachab
lem. The MGTS consists of one precovering graph with a single node $\boldsymbol{\omega}$ which
the ω-marking where all places have an unbounded number of tokens and f
every transition can fire. A solution for this MGTS is a transition sequence f
to $m_{1,out}$.

$$\mathcal{L}(N, M_0, M_f) = \mathcal{L}(G_{RP}).$$

Hence, to decide **RP** it is sufficient to solve arbitrary MGTS. Lambert d
each MGTS a characteristic equation that is fulfilled by all its solutions
words, the equation is a necessary condition for solutions of the MG
precisely, the author derives a system of linear equations $Ax = b$ whe
b range over integers. It encodes the firing behaviour of the precoverin
and intermediary transitions and can become quite large. There is one
for every marking entry $m_{i,in}$ and $m_{i,out}$ (including zero and ω entries)
one variable for each edge in every precovering graph. Since markings
become negative, solutions sought must be semi-positive. This (possibl
set of semi-positive solutions can always be computed [7].

If the characteristic equation was sufficient for the existence of sol
an MGTS, **RP** would have been solved immediately. While not valid in
Lambert provides precise conditions for when this implication holds.
speaking, a solution to the characteristic equation yields a solution to th
if the variables for the edges and the variables for all ω-entries of the mar
unbounded in the solution space. An MGTS with such a sufficient char
equation is called *perfect* and denoted by $\mathbb{G}$. Unboundedness of the vari
be checked effectively [7].

Since not all MGTS are perfect, Lambert presents a decomposition p
[8]. It computes from one MGTS G a new set of MGTS that are to
degree perfect and have the same solutions as G. This means each tran
quence leading through the original MGTS and solving it will also lead
at least one of the derived MGTS and solve it, and vice versa. The
perfectness is discrete and cannot be increased indefinitely. Therefor
composition procedure terminates and returns a finite set Γ_G of perfec
With the assumption that $m_{1,in}$ and $m_{n,out}$ are ω-free, the correspon
composition theorem is simplified to the following form.

Theorem 2 (Decomposition [8,13]). *An MGTS G can be decomp a finite set Γ_G of perfect MGTS with the same solutions,* $\mathcal{L}(G) = \bigcup_{\mathbb{G}\in}$

When we apply the decomposition procedure to the MGTS G_{RP} for the
$RP = (N, m_{1,in}, m_{1,out})$ of the reachability problem (Figure 2), we ob
$\Gamma_{G_{RP}}$ of perfect MGTS. For each of these perfect MGTS $\mathbb{G}$ we can decide
it has solutions, i.e., whether $\mathcal{L}(\mathbb{G}) \neq \emptyset$. If at least one has a solution,
a positive answer to the reachability problem, otherwise a negative ans
means, the following algorithm decides **RP**:

input $RP = (N, m_{1,in}, m_{1,out})$

`create` G_{RP} according to Figure 2

`decompose` G_{RP} into $\Gamma_{G_{RP}}$ with $\mathbb{G}$ perfect for all $\mathbb{G} \in \Gamma_{G_{RP}}$

if $\exists \mathbb{G} \in \Gamma_{G_{RP}}$ `with` $\mathcal{L}(\mathbb{G}) \neq \emptyset$ **answer** `yes` **else answer** `no`.

M_i (cf. Figure 1). We search for covering sequences u_i that indefinitely
the token count on ω-places of M_i. More precisely, u_i is a transition
from M_i to M_i with the following properties.

- The sequence u_i is enabled under marking $m_{i,in}$.
- If $M_i(s) = \omega > m_{i,in}(s)$, then u_i will add tokens to place s.
- If $M_i(s) = m_{i,in}(s) \in \mathbb{N}$, then u_i will not change the token count o

In case $M_i(s) = \omega = m_{i,in}(s)$, no requirements are imposed. Seque
accompanied by a second transition sequence v_i with similar propertie
that the reverse of v_i must be able to fire backwards from $m_{i,out}$. This
the token count on ω-places and lets v_i reach the output node $m_{i,out}$
Having such pairs of covering sequences $((u_i, v_i))_{1 \le i \le n}$ available for a
ering graphs, the following theorem yields a solution to the perfect M

Theorem 3 (Lambert's Iteration Lemma [8,13]). *Consider so MGTS $\mathbb{G}$ with at least one solution and let $((u_i, v_i))_{1 \le i \le n}$ be covering satisfying the above requirements. We can compute $k_0 \in \mathbb{N}$ and tran quences β_i, w_i from M_i to M_i such that* for every $k \ge k_0$ *the sequence*

$$(u_1)^k \beta_1 (w_1)^k (v_1)^k t_1 (u_2)^k \beta_2 (w_2)^k (v_2)^k t_2 \ldots t_{n-1} (u_n)^k \beta_n (w_n)^k (v_n$$

is a solution of $\mathbb{G}$.

Lambert proved that such covering sequences u_i, v_i always exist and tha
one can be computed [8]. Firing u_i repeatedly, at least k_0 times, pum
marking to the level necessary to execute $\beta_i (w_i)^k$. Afterwards v_i pump
to reach $m_{i,out}$. Transition t_i then proceeds to the next precovering gr

3.2 Computing the Downward-Closure

According to the decomposition theorem, we can represent the Petri net
$\mathcal{L}(N, M_0, M_f)$ by the decomposition of the corresponding MGTS G_{RP}.
restrict our attention to perfect MGTS $\mathbb{G}$ that have a solution, i.e., $\mathcal{L}$
They form the subset $\Gamma^{\checkmark}_{G_{RP}}$ of $\Gamma_{G_{RP}}$. As the labelled language just
homomorphism, we derive

$$\mathcal{L}_h(N, M_0, M_f) = h(\mathcal{L}(N, M_0, M_f)) = h(\bigcup_{\mathbb{G} \in \Gamma^{\checkmark}_{G_{RP}}} \mathcal{L}(\mathbb{G})) = \bigcup_{\mathbb{G} \in \Gamma^{\checkmark}_{G_{RP}}} h($$

Since downward-closure $-\downarrow$ and the application of h commute, a
downward-closure distributes over $\cup$, we obtain

$$\mathcal{L}_h(N, M_0, M_f)\downarrow = \bigcup_{\mathbb{G} \in \Gamma^{\checkmark}_{G_{RP}}} h(\mathcal{L}(\mathbb{G})\downarrow).$$

the language of every perfect MGTS $\mathbb{G} \in \Gamma^{\checkmark}_{G_{RP}}$. Then we apply the h
phism to these expressions, $h(\phi_{\mathbb{G}})$, and end up in a finite disjunction

$$\mathcal{L}_h(N, M_0, M_f)\downarrow = \mathcal{L}(\sum_{\mathbb{G} \in \Gamma^{\checkmark}_{G_{RP}}} h(\phi_{\mathbb{G}})).$$

We spend the remainder of the section on the representation of $\mathcal{L}(\mathbb{G})\downarrow$
ingly, the simple regular expression turns out to be just the sequence of t
sets in the precovering graph,

$$\phi_{\mathbb{G}} := T_1^*.(t_1 + \epsilon).T_2^* \ldots (t_{n-1} + \epsilon).T_n^*,$$

where $\mathbb{G} = C_1.t_1.C_2 \ldots t_{n-1}.C_n$ and C_i contains the transitions T_i.

Proposition 1. $\mathcal{L}(\mathbb{G})\downarrow = \mathcal{L}(\phi_{\mathbb{G}})$.

The inclusion from left to right is trivial. The proof of the reverse
relies on the following key observation about Lambert's iteration lem
sequences u_i can always be chosen in such a way that they contain all tr
of the precovering graph C_i. By iteration we obtain all sequences u_i^k.
contains all transitions in T_i, we derive

$$T_i^* \subseteq (\bigcup_{k \in \mathbb{N}} u_i^k)\downarrow = \bigcup_{k \in \mathbb{N}} u_i^k\downarrow \ .$$

Hence, all that remains to be shown is that u_i can be constructed so as t
all edges of C_i and consequently all transitions in T_i. Lets start with a
sequence u_i' that satisfies the requirements stated above and that can
with Lambert's procedure [8]. Since C_i is strongly connected, there i
path z_i from M_i to M_i that contains all edges of C_i. The corresponding t
sequence may have a negative effect on the ω-places, say at most $m \in$
are removed. Concrete token counts are, by construction of precoverin
reproduced exactly. Since u_i' is a covering sequence, we can repeat it m
By the second requirement, this adds at least $m+1$ tokens to every ω
we now append z_i, we may decrease the token count by m but still gu
positive effect of $m + 1 - m = 1$ on the ω-places. This means

$$u_i := {u_i'}^{m+1}.z_i$$

is a covering sequence that we may use instead of u_i' and that contains
sitions. This concludes the proof of Proposition 1.

4 Downward-Closure of Other Language Types

We consider the downward-closure of terminal and covering languages
minal languages that accept via deadlocks we provide a reduction to the
computability result. For covering languages, we avoid solving reachal
give a direct construction of the downward-closure from the coverabili

partially specified markings where the token count on some places is a $M_P \in \mathbb{N}^P$ with $P \subseteq P'$. Each such partial marking corresponds to a ca no transition can fire. Hence, the terminal language is a finite union of pa guages that accept by a partial marking, $\mathcal{T}_h(N, M_0) = \bigcup_{M_P \in \mathcal{P}} \mathcal{L}_h(N,$ We now formalise the notion of a partial language and then prove comp of its downward-closure. With the previous argumentation, this yields sentation for the downward-closure of the terminal language.

A partial marking $M_P \in \mathbb{N}^P$ with $P \subseteq P'$ denotes a potentially infin markings M that coincide with M_P in the places in P, $M|_P = M_P$. T *language* is therefore defined to be $\mathcal{L}_h(N, M_0, M_P) := \bigcup_{M|_P = M_P} \mathcal{L}_h(N$ We apply a construction due to Hack [3] to compute this union.

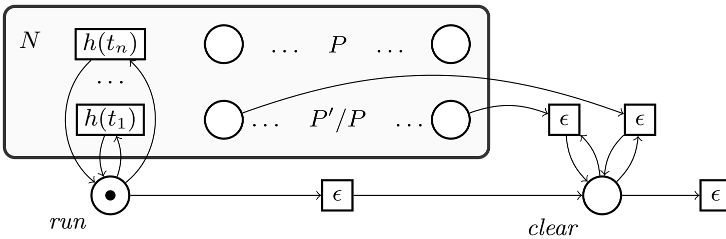

Fig. 3. Hack's construction to reduce partial Petri net languages to ordinary

We extend the given net $N = (P', T, F)$ to $N_e = (P' \cup P_e, T \cup T$ illustrated in Figure 3. The idea is to guess a final state by removing token and then empty the places outside $P \subseteq P'$. As a result, the r from M_0 to a marking M with $M|_P = M_P$ are precisely the runs in N_e to the marking M_f, up to the token removal phase in the end. Marki M_0 with an additional token on the *run* place. Marking M_f coincides and has no tokens on the remaining places. Projecting away the new tr t with $h_e(t) = \epsilon$ characterises the partial language by an ordinary lang

Lemma 1. $\mathcal{L}_h(N, M_0, M_P) = \mathcal{L}_{h \cup h_e}(N_e, M_0^r, M_f)$.

Combined with Theorem 1, a simple regular expression ϕ_{M_p} is comput satisfies $\mathcal{L}_h(N, M_0, M_P)\downarrow = \mathcal{L}(\phi_{M_p})$. As a consequence, the downwar of the terminal language is the (finite) disjunction of these expressions

Theorem 4. $\mathcal{T}_h(N, M_0)\downarrow = \mathcal{L}(\Sigma_{M_P \in \mathcal{P}}\ \phi_{M_p})$.

Note that the trick we employ for the partially specified final mark works for partial input markings. Hence, we can compute the languag terminal language also for nets with partially specified input markings

closure coincides with the downward-closure of the covering-languag
the desired regular expression is computable. The idea is to add edg
coverability tree that represent the domination of markings by their s
and thus, by monotonicity of Petri nets, indicate cyclic behaviour.
states reflect domination of the final marking. In the remainder, fix $N =$
with initial and final markings M_0 and M_f and labelling $h \in (\Sigma \cup \{\epsilon\}$

The coverability tree $CT = (V, E, \lambda)$ is similar to the coverability g
cussed in Section 2 but keeps the tree structure of the computation.
the vertices are labelled by extended markings, $\lambda(v) \in (\mathbb{N} \cup \{\omega\})^P$, and
$e \in E \subseteq V \times V$ by transitions, $\lambda(e) \in T$. A path is truncated as soon as
an already visited marking.

We extend CT to a finite automaton $FA = (V, v_0, V_f, E \cup E', \lambda \cup \lambda')$
backedges. The root of CT is the initial state v_0. States that cover M_f
$V_f := \{v \in V \mid \lambda(v) = M \geq M_f\}$. If the marking of v dominates the m
an E-predecessor v', $\lambda(v) = M \geq M' = \lambda(v')$, we add a backedge $e' =$
E' and label it by $\lambda'(e') = \epsilon$. The downward-closed language of this a
is the downward-closed covering language without labelling.

Lemma 2. $\mathcal{L}(FA)\downarrow = \mathcal{C}(N, M_0, M_f)\downarrow$.

To compute $\mathcal{L}(FA)\downarrow$ we represent the automaton as tree of its stro
nected components $SCC(FA)$. The root is the component C_0 that co
We need two additional functions to compute the regular expression
components $C, C' \in SCC(FA)$, let $\gamma_{C,C'} = (t + \varepsilon)$ if there is a t-labe
from C to C', and let $\gamma_{C,C'} = \emptyset$ otherwise. Let $\tau_C = \varepsilon$ if C contai
state and $\tau_C = \emptyset$ otherwise. Concatenation with $\gamma_{C,C'} = \emptyset$ or $\tau_C = \emptyset$ s
further regular expressions if there is no edge or final state, respectivel
denote the transitions occurring in component C as edge labels. We re
define regular expressions ϕ_C for the downward-closed languages of co

$$\phi_C := T_C^*.(\tau_C + \sum_{C' \in SCC(FA)} \gamma_{C,C'}.\phi_{C'}).$$

Due to the tree structure, all regular expressions are well-defined. The
lemma is easy to prove.

Lemma 3. $\mathcal{L}(FA)\downarrow = \mathcal{L}(\phi_{C_0})$.

As the application of h commutes with the downward-closure, a combi
Lemma 2 and 3 yields the desired representation.

Theorem 5. $\mathcal{C}_h(N, M_0, M_f)\downarrow = \mathcal{L}(h(\phi_{C_0}))$.

Note that $h(\phi_{C_0})$ can be transformed into a simple regular expressio
tributivity of concatenation over $+$ and removing possible occurrences

potentially malicious environment. This means, $N_s = (P_s, T_s, F_s)$ is em
in a larger net $N = (P, T, F)$ where the environment changes the token
restricts the firing behaviour in the subnet N_s. Figure 3 illustrates the
The environment is Hack's gadget that may stop the Petri net and em
places. The results obtained in this paper allow us to approximate th
system N_s can tolerate without reaching undesirable states.

Consider an initial marking M_0^s of N_s and a bad marking M_b^s tha
be avoided. For the full system N we either use $M_0^s, M_b^s \in \mathbb{N}^{P_s}$ as
specified markings or assume full initial and final markings, $M_0, M_b \in$
$M_0|_{P_s} = M_0^s$ and $M_b|_{P_s} = M_b^s$. The stability of N_s is estimated as follo

Proposition 2. *An upward-closed language is computable that unde*
mates the environmental behaviour N_s tolerates without reaching M_b^s

We consider the case of full markings M_0 and M_b of N. For partia
fied markings, Hack's construction in Section 4.1 reduces the problem
one. Let the full system N be labelled by h. Relabelling all transitio
to ϵ yields a new homomorphism h' where only environmental transi
visible. By definition, the downward-closure always contains the langua
$\mathcal{L}_{h'}(N, M_0, M_b)\downarrow \supseteq \mathcal{L}_{h'}(N, M_0, M_b)$. This is, however, equivalent to

$$\overline{\mathcal{L}_{h'}(N, M_0, M_b)\downarrow} \subseteq \overline{\mathcal{L}_{h'}(N, M_0, M_b)}.$$

By Theorem 1, the simple regular expression for $\mathcal{L}_{h'}(N, M_0, M_b)\downarrow$
putable. As regular languages are closed under complementation, the e
for $\overline{\mathcal{L}_{h'}(N, M_0, M_b)\downarrow}$ is computable as well. The language is upward-c
underapproximates the attacks the system can tolerate.

Likewise, if we consider instead of M_b a desirable good marking
language $\mathcal{L}_{h'}(N, M_0, M_g)\downarrow$ overapproximates the environmental influ
quired to reach it. The complement of the language provides behav
definitely leads away from the good marking. Note that for covering i
reachability similar arguments apply that rely on Theorem 5.

6 Conclusion

We have shown that the downward-closures of all types of Petri net l
are effectively computable. As an application of the results, we outlin
gorithm to estimate the stability of a system towards attacks from a
environment. In the future, we plan to study further applications. Esp
concurrent system analysis, our results should yield fully automated a
for the verification of asynchronous compositions of Petri nets with oth

[1] Formally, $N = (P, T, F)$ is *embedded* in $N' = (P', T', F')$ if $P \subseteq P'$, T
$F'|_{(S\times T)\cup(T\times S)} = F$. If homomorphism h labels N and h' labels N' then

system's language. However, cyclic proof rules are challenging.

References

1. Abdulla, P.A., Collomb-Annichini, A., Bouajjani, A., Jonsson, B.: Usin reachability analysis for verification of lossy channel systems. Form. Met Des. 25(1), 39–65 (2004)
2. Courcelle, B.: On constructing obstruction sets of words. Bulletin of the E 178–186 (1991)
3. Hack, M.: Decidability questions for Petri nets. Technical report, Cambr USA (1976)
4. Higman, G.: Ordering by divisibility in abstract algebras. Proc. Lond Soc. 2(3), 326–336 (1952)
5. Karp, R.M., Miller, R.E.: Parallel program schemata. J. Comput. Syst. 147–195 (1969)
6. Kosaraju, S.R.: Decidability of reachability in vector addition systems (p version). In: STOC, pp. 267–281. ACM, New York (1982)
7. Lambert, J.L.: Finding a partial solution to a linear system of equations i integers. Comput. Math. Applic. 15(3), 209–212 (1988)
8. Lambert, J.L.: A structure to decide reachability in Petri nets. Theo Sci. 99(1), 79–104 (1992)
9. Mayr, E.W.: An algorithm for the general Petri net reachability pro STOC, pp. 238–246. ACM, New York (1981)
10. Mayr, E.W.: An algorithm for the general Petri net reachability proble J. Comp. 13(3), 441–460 (1984)
11. Mayr, R.: Undecidable problems in unreliable computations. Theo Sci. 297(1-3), 337–354 (2003)
12. Peterson, J.L.: Petri nets. ACM Computing Surveys 9(3), 223–252 (197
13. Priese, L., Wimmel, H.: Petri-Netze. Springer, Heidelberg (2003)
14. Wimmel, H.: Infinity of intermediate states is decidable for Petri nets tadella, J., Reisig, W. (eds.) ICATPN 2004. LNCS, vol. 3099, pp. 426–434. Heidelberg (2004)