Reachability Analysis Using Net Unfoldings*

Claus Schröter and Javier Esparza

Institut für Informatik, Technische Universität München, email: {schroeter,esparza}@informatik.tu-muenchen.de

Abstract. We study four solutions to the reachability problem for 1-safe Petri nets, all of them based on the unfolding technique. We define the problem as follows: given a set of places of the net, determine if some reachable marking puts a token in all of them. Three of the solutions to the problem are taken from the literature [McM92,Mei98,He99], while the fourth one is first introduced here. The new solution shows that the problem can be solved in time $O(n^k)$, where $n$ is the size of the prefix of the unfolding containing all reachable states, and $k$ is the number of places which should hold a token. We compare all four solutions on a set of examples, and extract a recommendation on which algorithms should be used and which ones not.

1 Introduction

Reachability of states is one of the key problems in the area of automatic verification. Most safety properties of systems can be reduced to simple reachability properties; a typical example is the mutual exclusion property of mutual exclusion algorithms [Ray86]. When systems are presented as automata communicating through rendez-vous or through bounded buffers, as synchronous products of transition systems, or as 1-safe Petri nets (all of them models with the same expressive power), the reachability problem is known to be PSPACE-complete. In this paper we consider systems modelled by 1-safe Petri nets, and define the reachability problem as follows: given a set of places of the net, decide if some reachable marking puts a token in each of them. The problem remains PSPACE-complete if the set contains only one place.

The unfolding technique, originally introduced by McMillan in his seminal paper [McM92], has been very successfully applied to deadlock detection. The 1-safe Petri net is “unfolded” into an acyclic net (in a way similar to the unfolding of a rooted graph into a tree) until a so called (finite) complete prefix is generated. This is a finite acyclic net having exactly the same reachable markings as the original one. Once the complete prefix has been generated, three different algorithms can be applied: a branch-and-bound algorithm by McMillan [McM92], an algorithm based on linear programming by Melzer and Römer [MR97], and an algorithm based on SAT solvers (with stable model semantics) by Heljanko [He99]. These algorithms have been compared (see [MR97,He99]).

* Supported by the Sonderforschungsbereich SFB-342 A3 SAM
with the result that SAT algorithms have the edge in most cases. The goal of this paper is to perform the same kind of analysis for the reachability problem.

First of all, we show that the reachability problem is NP-complete in the size of the complete prefix. (This is also the complexity of deadlock detection [McM92].) We then present four different algorithms. McMillan sketches an on-the-fly solution in [McM92]. In [Mel98], Melzer extends the linear programming approach of [MR97] for deadlock detection to reachability, and so does Heljanko in [Hel99]. Both algorithms have exponential complexity in the size of the complete prefix. The fourth algorithm was in a sense implicit in [Mel98], and even in former papers, but to the best of our knowledge it has not been explicitly formulated before. In particular, we do not know of any implementation. It reduces the reachability problem to CLIQUE, and has a better complexity than the former two; it solves the reachability problem in time $O(n^k)$, where $n$ is the size of the complete prefix, and $k$ is the number of places that should be simultaneously marked. Since $n$ is usually much larger than $k$, this is a significant improvement.

In the last part of the paper we present a comparison of the four algorithms based on experiments conducted on a number of examples. The results show that, even though it has a better theoretical complexity, the reduction to CLIQUE cannot compete with the other algorithms. In fact, the two best algorithms are the on-the-fly algorithm and the algorithm based on SAT.

The paper is structured as follows: In section 2 we give an introduction to Petri nets and unfoldings following [ERV96,MR97,ER99]. Thereby we restrict ourselves to 1-safe Petri nets. Section 3 briefly reviews the main ideas of the methods suggested by McMillan, Melzer and Heljanko and introduces our new graph theoretic method. In section 4 we compare the four algorithms and discuss some results. In section 5 we finish with some conclusions.

2 Basic Notations

1-Safe Petri Nets A triple $(P, T, F)$ is a net if $P$ and $T$ are disjoint sets and $F$ is a subset of $(P \times T) \cup (T \times P)$. The elements of $P$ are called places and the elements of $T$ transitions. Places and transitions are generally called nodes. We identify $F$ with its characteristic function on the set $(P \times T) \cup (T \times P)$. The preset $\bullet x$ of a node $x$ is the set $\{y \in P \cup T \mid F(y, x) = 1\}$. The postset $x^\bullet$ of a node $x$ is the set $\{y \in P \cup T \mid F(x, y) = 1\}$.

A marking $M$ of a net $(P, T, F)$ is a mapping $M : P \to \{0, 1\}$. We identify a marking $M$ with the set $P^M \subseteq P$ such that $\forall p \in P : p \in P^M \iff M(p) = 1$ holds. A partial marking $M_{par}$ of a net is a mapping $M_{par} : (P^1_{par} \cup P^0_{par}) \to \{0, 1\}$, where $P^1_{par}, P^0_{par} \subseteq P$ and $\forall p \in P^1_{par} : M_{par}(p) = 1$ and $\forall p \in P^0_{par} : M_{par}(p) = 0$ and $P^1_{par} \cap P^0_{par} = \emptyset$. We identify a partial marking $M_{par}$ with the tuple $(P^1_{par}, P^0_{par})$.

A four-tuple $\Sigma = (P, T, F, M_0)$ is a net system if $(P, T, F)$ is a net and $M_0$ is a marking of $(P, T, F)$. $M_0$ is called the initial marking of the net system $\Sigma$. A marking $M$ enables a transition $t$ if $\forall p \in \bullet t : M(p) = 1$ holds. If $t$ is enabled
at $M$, then $t$ can occur, and its occurrence leads to a new marking $M'$ (denoted $M \xrightarrow{t} M'$), defined by $M'(p) = M(p) - F(p,t) + F(t,p)$ for every place $p$. A sequence of transitions $\sigma = t_1 t_2 \ldots t_n$ is an occurrence sequence if there exist markings $M_1, M_2, \ldots, M_n$ such that $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \ldots M_{n-1} \xrightarrow{t_n} M_n$. $M_n$ is the marking reached by the occurrence of $\sigma$, also denoted by $M_0 \xrightarrow{\sigma} M_n$. $M$ is a reachable marking if there exists an occurrence sequence $\sigma$ such that $M_0 \xrightarrow{\sigma} M$.

**Occurrence Nets** Let $(P,T,F)$ be a net and $x, y \in P \cup T$. The nodes $x$ and $y$ are in conflict (denoted $x \# y$) if there exist distinct transitions $t_1, t_2 \in T$ such that $t_1 \cap t_2 \neq \emptyset$ and $(t_1, x), (t_2, y)$ belong to the reflexive and transitive closure of $F$. The node $x \in P \cup T$ is in self-conflict if $x \# x$. An occurrence net is a net $N = (B, E, F')$, such that:

- $\forall b \in B : |b| \leq 1$,
- $F'$ is acyclic, i.e. the transitive closure of $F'$ is a partial order,
- $N$ is finitely preceded, i.e. for every $x \in B \cup E$, the set of elements $y \in B \cup E$ such that $(y, x)$ belongs to the transitive closure of $F'$ is finite, and
- no element $e \in E$ is in self-conflict.

The elements of $B$ and $E$ are called conditions and events, respectively. $\text{Min}(N)$ denotes the initial marking of an occurrence net, in which the minimal conditions carry exactly one token, and the other conditions no token. The (irreflexive) transitive closure of $F'$ is called the causal relation (denoted by $\prec$). The reflexive and transitive closure of $F'$ is denoted by $\preceq$. A node $x$ is causally related to $y$ if there exists a path from $x$ to $y$. The co-relation $\supseteq B \times B$ is defined in the following way: $(b_1, b_2) \in \supseteq \Leftrightarrow (b_1 \not\prec b_2 \land b_2 \not\prec b_1 \land \neg (b_1 \# b_2))$, i.e. two conditions are called concurrent, if they are not causally related and if they are not in conflict. A set $B' \subseteq B$ of an occurrence net is called co-set if its elements are pairwise in co-relation.

**Branching Processes** Branching processes of a net system $\Sigma = (N, M_0)$ are labelled occurrence nets containing information about both concurrency and conflicts. The conditions of these nets are labelled with places of $N$ and their events are labelled with transitions of $N$. A condition is denoted by $(p, e)$, where $p \in P$ is a place and $e \in E$ is its unique input event. The label of a condition $(p, e)$ is $p$. An event is denoted by $(t, X)$, where $t \in T$ is a transition and $X \subseteq B$ is the set of its input conditions. The label of an event $(t, X)$ is $t$. Minimal conditions of the occurrence net are denoted by $(p, \perp)$, where $p$ carries a token initially, i.e. $p \in M_0$. In the following we write the labelling as a projection $h : (B \cup E) \rightarrow (P \cup T)$, such that $h((x, y)) = x$. Since branching processes are completely determined with this notation by their sets of conditions and events, we represent them as a pair $(B, E)$.

The set of finite branching processes of a net system $(N, M_0)$ with $M_0 = \{p_1, \ldots, p_n\}$ is inductively defined as follows:

- $\{(p_1, \perp), \ldots, (p_n, \perp)\}, \emptyset$ is a branching process of $(N, M_0)$.
– If \((B, E)\) is a branching process, \(t\) is a transition, and \(X \subseteq B\) is a co-set labelled by \(t^*\), then \((B \cup \{(p, e) \mid p \in t^*\}, E \cup \{e\})\) is also a branching process of \((N, M_0)\), where \(e = (t, X)\). If \(e \not\in E\), then \(e\) is called a possible extension of \((B, E)\).

The set of all branching processes of \((N, M_0)\) is obtained by declaring that the union of any finite or infinite set of branching processes is also a branching process, where union of branching processes is defined componentwise on conditions and events. Since branching processes are closed under union, there is a unique maximal branching process. We call it the unfolding of \((N, M_0)\).

**Configurations and Cuts** A configuration \(C\) of an occurrence net is a set of events satisfying the following two conditions: (i) \(C\) is causally closed, i.e. \(e \in C \Rightarrow \forall e' \leq e: e' \in C\) and (ii) \(C\) is conflict-free, i.e. \(\forall e, e' \in C: -(e\#e')\).

A maximal co-set \(B'\) with respect to set inclusion is called a cut. Let \(C\) be a finite configuration and \(\text{Cut}(C) = (\text{Min}(N) \cup C^*) \setminus C\). Then \(\text{Cut}(C)\) is a cut. In particular, the set of places \(h(\text{Cut}(C))\) is a reachable marking denoted by \(\text{Mark}(C)\).

For an event \(e\) we define the local configuration \([e]\) by the set of all events \(e'\) such that \(e' \leq e\). Then we call \(e\) a cut-off event of a branching process \(\beta\) if \(\beta\) contains a local configuration \([e'] \prec [e]\) such that the corresponding markings are equal, i.e. \(\text{Mark}([e]) = \text{Mark}([e'])\). \(\prec\) denotes a total order on the configurations of \(\beta\). See [ERV96] for more details on total orders on configurations of prefixes.

A branching process \(\beta\) of a net system \(\Sigma\) is called complete finite prefix if and only if for every reachable marking \(M\) there exists a configuration \(C\) in \(\beta\) without any cut-off event such that (i) \(\text{Mark}(C) = M\) (i.e. \(M\) is represented in \(\beta\)) and (ii) for every transition \(t\) enabled by \(M\) there exists a configuration \(C \cup \{e\}\) such that \(e \not\in C\) and \(e\) is labelled by \(t\).

Figure 1 shows a 1-safe net system and its complete finite prefix, where \(e_3, e_6, e_7, e_8, e_{10}\) and \(e_{12}\) are cut-off events.

### 3 Different methods for reachability checking

As mentioned in the introduction we investigate the reachability problem of 1-safe Petri nets using complete finite prefixes. We will now define our understanding of the reachability problem more precisely.

**Definition 1.** Reachability problem for 1-safe Petri nets using prefixes

The reachability problem is as follows: Given a net system \((N, M_0)\), and a partial marking \(P_{par} = (P_{par}, P_{par}^0)\), is there a marking \(M\) reachable from \(M_0\) (i.e. \(\exists \sigma : M_0 \xrightarrow{\sigma} M\)) such that for every \(p \in (P_{par}^1 \cup P_{par}^0)\): \(M(p) = M_{par}(p)\).
Theorem 1. *NP-completeness of the reachability problem*

The reachability problem for 1-safe Petri nets using prefixes is NP-complete.

The proof is presented in Appendix A. In the following we briefly review methods based on linear programming [Mel98] and logic programs [Hei99], and introduce a new method using a graph theoretic approach. Moreover, we present an on-the-fly verification technique as mentioned by McMillan [McM92].

3.1 Using linear programming: *CheckLin*

Mekler [Mel98] has introduced a method for checking the reachability of a marking based on linear programming. The basic concept of this method is the so-called marking equation that can be used as an algebraic representation of the set of reachable markings of an acyclic net. Given a marking \( M \) reachable from the initial marking \( M_0 \) and a place \( p \), the number of tokens of \( p \) in \( M \) can be calculated as the number of tokens \( p \) carries in \( M_0 \) plus the difference of tokens added by the input places and removed by the output places. This leads to the following equation: \( M(p) = M_0(p) + \sum_{i \in \text{in}p} \#t - \sum_{j \in \text{out}p} \#t \), where \( \#t \) denotes the number of occurrences of \( t \) in \( \sigma \). Usually this equation is written in the form \( M = M_0 + N \cdot \sigma \), where \( \sigma = (\#t_1, \ldots, \#t_n) \) is called the Parsik vector of \( \sigma \) and \( N \) denotes the incidence matrix of \( N \), a \( P \times T \) matrix given by \( N(p,t) = F(t,p) - F(p,t) \). Additionally we formulate a set of restrictions: for each place \( p_i \in P_{\text{par}} \) we add the restriction \( M(p_i) \geq 1 \), and for each \( p_i \in P^{0}_{\text{par}} \) we add the restriction \( M(p_i) \leq 0 \). Usually the restriction for all places of the marking is given in the matrix form \( A \cdot M \geq b \). With the knowledge that every 1-safe Petri net can be unfolded into an acyclic net and that the marking equation yields a sufficient condition for reachability in acyclic nets, the reachability of a marking can be checked using the following result.

**Theorem 2. Test schema on prefixes [Mel98]**

Let \( (N,M_0) \) be a 1-safe net, \( A \cdot M \geq b \) a restriction and \( \beta \) the prefix of \( (N,M_0) \). The restriction holds for a marking reachable from \( M_0 \) if and only if the following system of linear inequalities has a solution for \( M' \) and \( X \):

Variables: \( M', X \) binary

\[
M' = \text{min}(\beta) + N' \cdot X
\]

\[
h(A) \cdot M' \geq b
\]

We describe this method by means of an example. Consider the net in Figure 1 and the partial marking \( P_{\text{par}} = \{p_2, p_4, p_6\} \). To check if \( P_{\text{par}} \) is a reachable marking, we formulate a corresponding restriction, i.e. \( M(p_2) \geq 1 \) and \( M(p_4) \geq 1 \) and \( M(p_6) \geq 1 \). Using the projection \( h \) this restriction can be transferred to markings \( M' \) of the prefix. Knowing that \( h(b_2) = p_2, h(b_4) = h(b_{10}) = p_4 \) and \( h(b_6) = h(b_8) = h(b_{12}) = p_6 \) we get the restriction \( M'(b_2) \geq 1 \) and \( M'(b_6) + M'(b_{10}) \geq 1 \) and \( M'(b_4) + M'(b_6) + M'(b_{12}) \geq 1 \). The system of linear inequalities looks like this:
\[ M'(b_1) = 1 - X(e_4) \]
\[ M'(b_2) = 1 - X(e_2) \]
\[ M'(b_3) = 1 - X(e_1) \]
\[ M'(b_4) = X(e_1) - X(e_2) - X(e_9) \]
\[ M'(b_5) = X(e_2) - X(e_4) - X(e_9) \]
\[ M'(b_6) = X(e_4) - X(e_6) \]
\[ M'(b_7) = X(e_6) - X(e_8) - X(e_9) \]
\[ M'(b_8) = X(e_9) - X(e_{10}) \]
\[ M'(b_{11}) = X(e_{10}) - X(e_{11}) \]
\[ M'(b_{12}) = X(e_{11}) - X(e_{12}) \]

On the left side the marking equation of the prefix is shown, and on the right side first the three inequalities of the restriction and below it the equalities for the elimination of the cut-off events are shown. The cut-off events can be eliminated because all reachable markings are reachable without firing cut-off events. The elimination of cut-off events reduces the number of binary variables and simplifies the inequality system. Therefore the complexity of this method is exponential in the number of non-cut-off events.

It can easily be seen that \( M' = \{ b_7, b_{10}, b_{12} \} \) with \( X = M' \{ 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0 \} \) yields the desired solution.

### 3.2 Using logic programming: Mecmodels

Heijenko [Hei99] presented a method for reachability checking of complete finite prefixes using logic programs with stable model semantics. The main idea of this approach is to translate the problem into a rule based logic program and to check if there exists a stable model. This method reduces the reachability problem to SAT. The algorithm uses the Smodels tool which is an implementation of a constraint based logic programming framework developed to find stable models of a logic program. We show an example to give an idea of the reduction to SAT, but due to space limitations we refer the reader to [Hei99] for more details.

Consider the net in Figure 1 and the partial marking \( P'_{\text{par}} = (\{ p_2, p_4, p_6 \}, \emptyset) \). We show, how we can reduce the reachability problem to SAT for this example. First define a variable for each condition and for every non-cut-off event of the prefix, e.g. \( b_1, \ldots, b_2 \) and \( e_1, \ldots, e_{11} \). The cut-off events \( e_3, e_5, e_7, e_8, e_{10} \) and \( e_{12} \) can be omitted because each reachable marking can be reached without firing cut-off events. \( b_i \) means that the corresponding condition holds a token, otherwise we write \( \neg b_i \). \( e_i \) means that the corresponding event has fired, otherwise we write \( \neg e_i \). For each condition there is a rule stating when it holds a token. For example, \( b_4 \) holds a token if \( e_1 \) has fired and \( e_2 \) has not fired. Then we need rules describing the causal relation. Finally we need one rule for each place in \( P'_{\text{par}} \), i.e. one rule for \( p_2, p_4 \) and \( p_6 \). For example, the rule for \( p_6 \) is \( b_4 \lor b_6 \lor b_{12} \), because \( p_6 \) holds a token whenever one of these conditions holds a token. Altogether, the partial marking \( P'_{\text{par}} \) is reachable iff the rules are satisfiable.

\[
\begin{align*}
&b_1 \leftrightarrow \neg e_4 & b_4 \leftrightarrow e_1 \land \neg e_2 & b_7 \leftrightarrow e_4 \\
&b_2 \leftrightarrow \neg e_2 & b_5 \leftrightarrow e_2 \land \neg e_4 & b_9 \leftrightarrow e_4 \land \neg e_5 \\
&b_3 \leftrightarrow \neg e_1 & b_6 \leftrightarrow e_2 \land \neg e_6 & b_{11} \leftrightarrow e_5 \land \neg e_7 & b_{12} \leftrightarrow e_7 \\
& & b_4 \leftrightarrow e_4 \land \neg e_6 & b_9 \leftrightarrow e_7 \\
& & b_5 \leftrightarrow e_5 \land \neg e_6 & b_{10} \leftrightarrow e_9 & b_{11} \leftrightarrow e_9 \\
& & b_4 \leftrightarrow e_4 \land \neg e_6 & b_6 \leftrightarrow e_5 \land \neg e_7 & b_{12} \leftrightarrow e_7
\end{align*}
\]
3.3 A new graph theoretic algorithm: CheckCo

Basically, our algorithm uses the co-relation, which is defined on the set of conditions of a prefix. Generally, two conditions are in co-relation iff they are not causally related and not in conflict. In the implementation of Römer [Röm00] the co-relation is calculated while generating the prefix and can directly be used as input for our algorithm. Proposition 1 states that co-sets are reachable.

**Proposition 1.** Reachability of co-sets [BF88]

Let $\beta = (B,E)$ be a prefix and $B' \subseteq B$ ($B'_\text{par} = (B'_\text{par}, \emptyset)$) be a marking (partial marking). $B'$ ($B'_\text{par}$) is reachable from $\text{Min}(\beta)$ iff $B'$ ($B'_\text{par}$) is a co-set.

In [Mel98] it is shown that the result of Proposition 1 together with the fact that all reachable markings of a 1-safe net are coded in its prefix can be combined to derive the following theorem.

**Theorem 3.** Reachability of partial markings [Mel98]

Let $(N, M_0)$ be a 1-safe net, $\beta$ its prefix, and $P'_\text{par} = (P'_\text{par}, \emptyset)$ a partial marking. $P'_\text{par}$ is reachable iff there exists a co-set $B' \subseteq B$ such that for every $p \in P'_\text{par}$ there exists a $b \in B'$ with $h(b) = p$.

By means of the previous example we show, how we can use the co-relation to decide the reachability of partial markings. Consider the net system and its prefix depicted in Figure 1.

In this case the co-relation of the prefix is the symmetrical closure of the set

\{$(b_1,b_2), (b_1,b_3), (b_1,b_4), (b_1,b_5), (b_1,b_6), (b_2,b_3), (b_2,b_4), (b_2,b_5), (b_2,b_6), (b_3,b_5), (b_6,b_7), (b_6,b_8), (b_7,b_8), (b_7,b_9), (b_7,b_{10}), (b_{11},b_{12}), (b_8,b_9)$, \}
Suppose we want to know if the partial marking \( \{p_2, p_4, p_6\}, \emptyset \) is reachable. According to Theorem 3 the marking is reachable if there exist conditions mapped onto \( p_2, p_4, p_6 \) by the projection \( h \) that are all pairwise in co-relation. Considering \( \{b_7, b_{10}, b_{12}\} \) it can easily be seen that the above marking is reachable because \( (b_7, b_{10}) \in \alpha \), \( (b_7, b_{12}) \in \alpha \), and \( (b_{10}, b_{12}) \in \alpha \) with \( h(b_7) = p_2 \), \( h(b_{10}) = p_4 \) and \( h(b_{12}) = p_6 \).

The search for a possible solution corresponds to the graph theoretic problem of finding a k-clique in a k-partite graph. We will explain this in more detail below. Let us construct a k-partite graph in the following way:

**Algorithm 1:** Construction of the k-partite graph \( G_k = (V, E) \)

Let \( N = (P, T, F) \) be a net and \( P_{par}' = (\{p_1, p_2, \ldots, p_k\}, \emptyset) \) a partial marking. Let \( \beta = (B, E) \) be a complete finite prefix of \( N \) and \( \alpha \subseteq B \times B \) be the co-relation.

(i) For each \( p_i \in \{p_1, p_2, \ldots, p_k\} \) calculate the set of conditions \( B_i = \{b_{i1}, b_{i2}, \ldots, b_{im}\} \) with \( h(b_{ij}) = p_i \) for all \( 1 \leq j \leq m \).

(ii) Let \( V := \bigcup_{1 \leq i \leq k} B_i \).

(iii) Draw an arc between \( b_{im}, b_{jn} \in V \) with \( i \neq j \) if \( (b_{im}, b_{jn}) \in \alpha \), i.e. draw an arc between two nodes, if they are in co-relation and belong to different partitions. (This means that each \( B_i \) forms one partition since no two elements in \( B_i \) are connected by an arc).

We show the construction of \( G_k \) for the net and the prefix shown in Figure 1 and the partial marking \( P_{par}' = (\{p_2, p_4, p_6\}, \emptyset) \). The sets \( B_i \) of conditions can be deduced directly from the prefix: \( B_1 = \{b_7\} \), \( B_2 = \{b_5, b_{10}\} \) and \( B_3 = \{b_4, b_6, b_{12}\} \). The elements of \( B_1, B_2 \) and \( B_3 \) form the three partitions of the graph. We draw arcs only between nodes which are in co-relation belonging to different partitions: \( (b_7, b_{10}), (b_7, b_5), (b_7, b_{12}), (b_5, b_6), (b_{10}, b_{12}) \). Figure 2 shows the graph \( G_3 \).

It can easily be seen that \( b_7, b_{10} \) and \( b_{12} \) build a 3-clique, and therefore we can conclude that the partial marking \( P_{par}' = (\{p_2, p_4, p_6\}, \emptyset) \) is reachable.

The following theorem states that we can use the k-partite graph \( G_k \) for checking the reachability of a partial marking.

**Theorem 4.** Reachability of partial markings

Let \( (N, M_0) \) be a 1-safe net, and \( P_{par}' = (P_{par}'^1, \emptyset) \) a partial marking. \( P_{par}' \) is reachable iff the k-partite graph \( G_k \) has a k-clique.
A net system extended with complementary places and its prefix

**Concept of complementary places** The method explained above does not work if the partial marking under consideration includes places that should not be marked. For example, in Figure 3 one might want to know whether there is a reachable marking in which the place $p_2$ carries a token and the place $p_4$ carries no token. For this task it is not sufficient to consider only the co-relation.

There exist co-related conditions in the prefix belonging to the places $p_2$ and $p_4$ and therefore we can conclude that the partial marking ($\{p_2, p_4\}, \emptyset$) is reachable. However we cannot decide if the partial marking ($\{p_2\}, \{p_4\}$) is reachable. To cope with this problem we introduce complementary places.

**Definition 2.** *Complementary place*

Let $(N, M_0)$ with $(P, T, F)$ be a net and $p \in P$ a place. A place $p^c \in P$ is called *complement of p* iff

(i) $\forall (p, t) \in F : (t, p^c) \in F \Leftrightarrow (t, p) \notin F$

(ii) $\forall (t, p) \in F : (p^c, t) \in F \Leftrightarrow (p, t) \notin F$

(iii) $\forall (p^c, t) \in F : (t, p) \in F \Leftrightarrow (t, p^c) \notin F$

(iv) $\forall (t, p^c) \in F : (p, t) \in F \Leftrightarrow (p^c, t) \notin F$

(v) $M_0(p^c) = 1 - M_0(p)$

Using this concept, the problem of checking the reachability of the partial marking ($\{p_2\}, \{p_4\}$) can be reduced to constructing a net with complementary places and checking the reachability of the partial marking ($\{p_2, p_4^c\}, \emptyset$) which is possible using only the co-relation. Figure 3 shows a net, its modification with complementary places and the corresponding prefix. Using the prefix of the modified net it can be seen that the partial marking ($\{p_2\}, \{p_4\}$) is reachable because the conditions $b_5$ with $h(b_5) = p_2$ and $b_6$ with $h(b_6) = p_4^c$ are in co-relation.

**Proposition 2.** *Reachability of partial markings*

Let $(N, M_0)$ be a net and $P^1_{par} = (\{p_1, \ldots, p_k\}, \{p_{k+1}, \ldots, p_n\})$ a partial marking. $P^1_{par}$ is reachable iff $P^2_{par} = (\{p_1, \ldots, p_k, p_{k+1}^c, \ldots, p_n^c\}, \emptyset)$ is reachable.

The complement $p^c$ of a place $p$ can be added as follows: the preset of $p^c$ is the postset of $p$ and the postset of $p^c$ is the preset of $p$. But this may lead into trouble in the special case that place $p$ has a side-loop, i.e. a transition that is both in the preset and in the postset. Figure 4 (left side) illustrates such a
situation. The central part of Figure 4 shows the construction of $p_2$ according to the fashion described above. It can be seen that transition $t_2$ can never fire. This is an undesired behaviour and therefore the arcs between transition $t_2$ and place $p_2$ have to be deleted. Figure 4 (right side) shows the corrected system. Generally speaking, we only connect a complementary place with a transition if the transition is not part of a side-loop.

**Outline of our algorithm** In this section we give an outline of our algorithm.

**Algorithm 2:**

**Input:** Net $[N, M_0]$ and a partial marking $P'_{par} = \{p_1, \ldots, p_k, \{p_{k+1}, \ldots, p_n\}\}.$  

**Output:** NO or YES

begin
  Read($N$);
  InsertAllComplements($N$);
  Unfold($N$);
  Read($P'_{par}$);
  Replace $P'_{par}$ by $\{p_1, \ldots, p_k, p_{k+1}', \ldots, p_n', \emptyset\}$;
  forall $p_i \in \{p_1, \ldots, p_k, p_{k+1}', \ldots, p_n'\}$ do
    Calculate($B_i$);
    / \ * \\
  od

  $L := \emptyset$;
  Check($1$);
  output (NO);
end

**proc Check($\text{int count}$)**

begin
  $j := 1$;
  while $j \leq |B_{count}|$ do
    $L := L \cup \{b_{count}\}$;
    if all elements of $L$ are pairwise in co-relation then
      if $|L| = |P'_{par} \cup P_0|$ then
        output (YES);
        exit;
      else
        Check($\text{count} + 1$);
      endif;
    endif;
    $L := L \setminus \{b_{count}\}$;
    $j := j + 1$;
  od
end Check

Our algorithm works as follows: First we read the original Petri net into an internal net structure and insert all complementary places. Then we unfold the modified net into a complete finite prefix. The co-relation is constructed while generating the prefix. We read the partial marking and replace all places $p_i$ with $M_{par}(p_i) = 0$ by their complements $p_i'$. Then for each place $p_i$ in $P_{par}$, we find $B_i$, i.e. the set of conditions which are mapped to $p_i$ by $h$. The marking is reachable if there exists a clique of conditions, one for each place in $P_{par}$, such that these conditions are pairwise in co-relation. This part is implemented in the procedure Check. The set $L$ yields the desired clique if there is one.

Let us consider again the net system of Figure 1. We want to check if the partial marking $P'_{par} = \{p_2, p_4, p_5\}$ is reachable. First we have to insert complementary places, but it can be seen that $p_3$ and $p_6$ are the complements of $p_1$, $p_3$ and $p_5$. So we can check if the partial marking $P'_{par} = \{\{p_2, p_4, p_5\}, \emptyset\}$ is reachable. For that, we have to calculate the sets $B_i (1 \leq i \leq 3)$. We ob-
tain $B_1 = \{ b_7 \}$ with $h(b_7) = p_2$, $B_2 = \{ b_5, b_6 \}$ with $h(b_5) = h(b_6) = p_4$ and $B_3 = \{ b_3, b_4, b_{12} \}$ with $h(b_3) = h(b_4) = h(b_{12}) = p_6$. Now we can invoke the procedure $\text{Check}(1)$. In the first step we set $L = \{ b_7 \}$ and call $\text{Check}(2)$. We have to test if all elements of $L = \{ b_7, b_5 \}$ are in co-relation. Apparently this is not the case and so we try $L = \{ b_7, b_{10} \}$. These elements are co-related and we can call $\text{Check}(3)$. $L = \{ b_7, b_{10}, b_4 \}$ is no solution because $(b_7, b_4) \notin co$, $L = \{ b_7, b_{10}, b_9 \}$ is no solution because $(b_{10}, b_9) \notin co$, but $L = \{ b_7, b_{10}, b_{12} \}$ yields the desired clique (see Figure 2). Then according to Theorem 3 the partial marking $P_{\text{par}}^u = \{ (p_2, p_4, p_6), \emptyset \}$ is reachable and hence (with Proposition 2) the partial marking $P_{\text{par}}^l = \{ (p_2, p_4), \{ p_6 \} \}$ is reachable. Note that at first it might appear unnecessary to insert all complementary places (as opposed to just the complements of those places $p_i$ for which $M_{\text{par}}(p_i) = 0$). But in this way we avoid having to re-calculate the prefix for each new marking. Inserting all complementary places allows us to calculate the prefix only once for each net and then to reuse it for checking further markings.

**Complexity of CheckCo** We briefly analyze the complexity of CheckCo. Let $(P, T, F)$ be a net, $\beta = (B, E)$ its corresponding prefix, $co$ the co-relation and $P_{\text{par}} = (P_{\text{par}}^l, P_{\text{par}}^u)$ a partial marking. There exists a parameterized function $M_{\text{par}}: (P_{\text{par}}^l \cup P_{\text{par}}^u) \mapsto \{ 0, 1 \}$ that maps the places of the partial marking onto 0 (not marked) and 1 (marked). Each place of $P_{\text{par}}^l \cup P_{\text{par}}^u$ corresponds to at most $|B|$ conditions in the prefix which leads to $|B|^{P_{\text{par}}^l \cup P_{\text{par}}^u}$ possible solutions. Then we need at most $|P_{\text{par}}^l \cup P_{\text{par}}^u|^2$ comparisons for checking if the conditions of one possible solution are in pairwise co-relation. This leads to a complexity of $O(|B|^{P_{\text{par}}^l \cup P_{\text{par}}^u} \cdot |P_{\text{par}}^l \cup P_{\text{par}}^u|^2)$.

### 3.4 On-the-fly verification: OnTheFly

In [ERV96] the authors present an efficient algorithm for constructing a complete finite prefix of 1-safe Petri nets. Knowing that all reachable markings of the net are coded in its prefix [McM92] we can verify the reachability of a partial marking during calculation of the prefix in the following way: Let $P_{\text{par}}^u = \{ \{ p_1, \ldots, p_k \}, \{ p_{k+1}, \ldots, p_n \} \}$ be the partial marking to be checked. As shown in the previous section we insert the complements of the places $p_{k+1}, \ldots, p_n$ into the net and check the partial marking $P_{\text{par}}^l = \{ \{ p_1, \ldots, p_k, p_{k+1}, \ldots, p_n \} \}$. This can be done in a way first suggested by McMillan [McM92]. We insert a new transition $t_{\text{new}}$ into the original net in such a way that $t_{\text{new}}^* = \{ p_1, \ldots, p_k, p_{k+1}, \ldots, p_n \}$. Then we start the unfolding algorithm described in [ERV96]. The algorithm stops if an event $e$ with $h(e) = t_{\text{new}}$ can be inserted into the prefix. At this point we can conclude that the marking $P_{\text{par}}$ is reachable, otherwise the prefix will be generated completely. We explain this method by means of an example. Consider the net in Figure 1 and the partial marking $P_{\text{par}}^l = \{ \{ p_2, p_3, p_6 \}, \emptyset \}$. Figure 1 (consider the dotted lines) illustrates the modified net system and its prefix. The algorithm stops at the dotted line, because the next event that can be inserted has the places $p_2$, $p_3$ and $p_6$ in its preset. Therefore the reachability of $P_{\text{par}}$ is proven.
4 Comparison of the algorithms

In this section we compare the four approaches and try to deduce a rule describing the situations in which one algorithm is more suitable than the others. We confirm our statements by practical results. For our tests we used a representative subset of Corbett’s examples [Cor94] on randomly generated “meaningful” markings. These examples are also used in [MR97,He99]. In the following we briefly explain how “meaningful” markings were generated.

Originally, Corbett’s examples are modelled as communicating finite automata and they are translated from these into Petri nets. The translation procedure yields a division of the nets into components where each component can carry at most one token. For this reason, it would be useless to test markings which include two or more places belonging to the same component because such markings are not reachable. To avoid the generation of such markings, our marking generator works as follows: First we determine the number of components and for each component the number of its places. Then we randomly choose $k$ of the components (where $k$ is the size of the generated marking). For each of the chosen components we randomly select one of its places. Note that the size of the markings is bounded by the number of components of the net system, but this is not a big restriction.

We present results for partial markings with 2, 4 and 6 places. The average verification times are all based on at least 15 different markings. All computations were carried out on the same machine, a SUN SPARC20 with 96 MByte RAM. CheckLin uses CPLEX$^M$ (version 6.5.1) as its underlying MIP-solver, and Meso models uses Smo dels as constraint programming framework.

First we compare the three methods Meso models, CheckLin and CheckCo because they all need a prefix as input. The prefix construction takes the same time for Meso models and CheckLin, but takes more time for CheckCo. All these methods use the same optimized unfolding procedure and therefore the difference between the unfolding times of Meso models/CheckLin and CheckCo is only caused by the additional complementary places. The unfolding times $t_{\text{unf}}$ in the table confirm this. The prefixes have to be constructed only once and can be reused for checking further markings. The times $t_{\text{avg}}$ in the table show the average time
needed for verification without the unfolding time. The rows R2, R4, R6 show tests with reachable markings of size 2, 4, and 6. The row N4 shows the results for unreachable markings with 4 places. Apparently, the table shows that the algorithm Mesmodels yields the fastest verification times for all markings independently of the marking size and independently of whether the markings are reachable or not. So, in a second step, we only need to compare Mesmodels with the on-the-fly method OnTheFly. The OnTheFly algorithm stops the unfolding process if a cut is found which represents the marking under consideration, otherwise it calculates the prefix completely. Therefore it needs for reachable markings at most the unfolding time of the complete prefix. In the case that the markings are unreachable, OnTheFly takes at least the complete unfolding time. With this knowledge we guess that on-the-fly verification is more suitable than Mesmodels for checking reachable markings. On the other hand it seems useful to prefer Mesmodels for unreachable markings. Indeed, the results seem to confirm this suspicion. If we look at the results for reachable markings, the OnTheFly algorithm is always faster than Mesmodels for the systems dpd(7), dpd(6) and oex(5). However, for systems like key(3), elevator(3) and rw(12) the opposite holds. In these cases we can compute the smallest integer \( n \) such that \( n \cdot \text{OnTheFly}_{\text{avg}} \geq \text{Mesmodels}_{\text{avg}} \). Then \( n \) denotes a breakpoint from which it is more efficient to use Mesmodels instead of OnTheFly. More precisely, if we test \( n \) or more markings the total time of Mesmodels (also including the unfolding time) is smaller than the total time of OnTheFly. The values for \( n \) are listed in the last column of the table. A look at the times for unreachable markings (rows N4) confirms our assumption that one should prefer Mesmodels as verification technique because for most systems the breakpoint \( n \) is 1 (meaning that Mesmodels is faster than OnTheFly even for only one marking).

Figure 5 summarizes the results. At any rate, if one guesses that the markings to be checked are unreachable, Mesmodels should be preferred. On TheFly is an efficient technique for the verification of reachable markings, but there may exist a breakpoint from which on it is better to use Mesmodels. This breakpoint, if any exists, is very different for the considered systems and depends on the size of the marking. Our tests have shown that the greater the size of the marking the smaller the breakpoint where it is appropriate to switch from OnTheFly to Mesmodels.

5 Conclusions

We have presented a new algorithm for reachability checking based on net unfoldings using a graph theoretic technique. Moreover, we have reviewed an on-the-fly verification technique and two methods for reachability checking using linear programming and logic programming with stable model semantics. By means
of Corbett's examples we have discussed the different algorithms and suggested which algorithms are most suitable for various reachability checking tasks.

Acknowledgements We would like to thank Stefan Römer for valuable comments and suggestions to this work.

References


A Proof of NP-completeness of the reachability problem

First we prove NP-hardness of the reachability problem by reducing from the SAT-problem for boolean formulae in conjunctive normal form to the reachability problem in polynomial time. Let \( \phi \) be a formula in conjunctive normal form with variables \( x_1, \ldots, x_n \) and clauses \( c_1, \ldots, c_m \). We construct a Petri net \( (N_\phi, M_0) \) in the following way: \( N_\phi \) contains

- a place \( p_{x_i} \) for each variable \( x_i \) such that \( \bullet p_{x_i} = \emptyset \) and \( p_{x_i}^* = \{ t_{x_i}, \overline{t_{x_i}} \} \); 
- a place \( p_j \) for each clause \( c_j \) and each literal \( l \) of \( c_j \) such that \( \bullet p_j = \{ t_l \} \) and \( p_j^* = \{ t_{\overline{l}} \} \); 
- a place \( p_{c_j} \) for each clause \( c_j \) such that \( \bullet p_{c_j} = \bigcup_{l \in c_j} \{ t_l \} \) and \( p_{c_j}^* = \emptyset \).
M₀₈ puts one token on each place pₓ, and no token elsewhere. However, one problem arises from this construction. The generated net is not 1-safe because it may happen that two transitions tᵢ and tᵢ₉ fire independently, and both of them put a token on the place pⱼ. This undesired behaviour can be repaired with a new place which ensures that only one of the transitions can fire. Therefore

- add a place p_r for each clause c_j such that *p_r = ∅ and p_r = \bigcup_{i \in c_j} \{t_i\}.

M₀₈ puts one token on each place p_r. Now we have constructed a 1-safe net with the property that the formula \( \phi \) is satisfiable iff the net \((N_\phi, M₀₈)\) has a reachable marking which puts one token on each place pⱼ. Figure 6 shows an example for the net \((N_\phi, M₀₈)\) with \( \phi = (x₁ \lor x₂ \lor \overline{x₃}) \land (\overline{x₄} \lor x₂) \). Now we have to show how we can construct the prefix \( \beta_\phi \) from the net \((N_\phi, M₀₈)\) in polynomial time. But this is an easy task because we only have to do minor changes to transform \((N_\phi, M₀₈)\) into \( \beta_\phi \). Recalling the definition of occurrence nets we see that the net \((N_\phi, M₀₈)\) only violates the property that all conditions must not have more than one event in their preset. It can be seen that only the places pⱼ have more than one transition in their presets. This can be changed if we duplicate these places. More precisely:

- replace each place pⱼ by \( k = |pⱼ| \) places, i.e. a place pⱼᵢ, for each literal \( l \in c_j \) such that *pⱼᵢ = \{tᵢ\} and pⱼᵢ = ∅.

The dotted lines in Figure 6 show the modified net. Surely, this net is the desired prefix and consequently the formula \( \phi \) is satisfiable iff the prefix has a reachable marking such that for each place pⱼ in the original net there is a token on exactly one of its corresponding places pⱼᵢ.

We have successfully proven \( \text{NP} \)-hardness for the reachability problem. The second step consists of proving that the problem is in \( \text{NP} \). This can be done by reduction from the reachability problem to SAT or another \( \text{NP} \)-complete problem. In sections 3.1 and 3.2 we have presented two methods which reduce the reachability problem to the problem of solving a linear inequation system [Mel98] and to SAT [Hel99].

Fig. 6. Net \((N_\phi, M₀₈)\) with \( \phi = (x₁ \lor x₂ \lor \overline{x₃}) \land (\overline{x₄} \lor x₂) \)