1989

Reachability Trees for High Level Petri Nets With Marking Variables

Chuang Lin

Dan C. Marinescu

Report Number:
89-857

https://docs.lib.purdue.edu/cstech/731

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.
REACHABILITY TREES FOR HIGH LEVEL PETRI NETS WITH MARKING VARIABLES

Chuang Lin
Dan C. Marinescu

CSD-TR-857
February 1989
REACHABILITY TREES
FOR HIGH LEVEL PETRI NETS WITH MARKING VARIABLES

Chuang Lin
Institute of Information Science and its Applications
State Information Center
58 Sanlihe Road Beijing, China

Dan Cristian Marinescu
Computer Science Department
Purdue University
West Lafayette, IN 47907
USA

Abstract

Net invariants and reachability tree are used to investigate dynamic properties of Petri Nets. Both concepts have been generalized for different classes of High Level Petri Nets. But the exponential explosion of the reachability set of High Level Petri Nets is a major obstacle encountered in implementing efficiently algorithms to construct the reachability tree. In this paper we define High Level Petri Nets with marking variables and show that for this class of nets it is possible to construct efficiently the reachability tree. The algorithm proposed in this paper is based upon equivalence and covering relations among marking variables. A significant advantage of our method is that the information about individual markings is available from the domains of the marking variables.

Keywords: High-Level Petri Nets, reachability tree, marking variable and equivalence of marking variables.
1. INTRODUCTION

Petri Nets, PNs, are one of the most interesting tools for the specification, modeling and analysis of concurrent systems. Several methods for analyzing PNs depend upon the ability to construct the reachability set and the reachability tree of the net. The reachability tree is used for investigating properties as boundedness, mutual exclusions between place markings, liveliness, etc.

The reachability tree of a PN is constructed by organizing all reachable markings in a tree structure where each node has attached a marking, while each arc has attached a transition which transforms the marking of a node into the marking of its descendent.

High Level Petri Nets, HLPNs are a family of nets with individual tokens and token variables. Prediction/Transition Nets, Coloured Petri Nets and Relation Nets belong to this category of nets. The main advantage in using HLPNs is derived from their power of representation. Rather complex systems can be modeled as HLPNs in a succinct and readable way. While the descriptive power of High Level Petri Nets is unquestionable, the analysis of such nets raises difficult problems. Genrich and Lautenbach, [2] and Jensen [3] have generalized the concept of place invariants, transition invariants and reachability tree to different families of High Level nets. But tools to analyze dynamical properties HLPN of models have been rather slow to emerge.

Recently we have investigated applications of High Level Petri Nets to performance analysis. The Stochastic High Level Petri Nets, SHLPNs introduced in [5], [6] are useful in modeling complex systems encountered in many parallel and distributed applications. The compound marking concept defined in connection with SHLPNs reflects the state aggregation principles from the theory of stochastic processes. State aggregation is very useful, it eventually transforms performance analysis problems which are virtually unsolvable into solvable problems by reducing drastically the size of the state space. To make effective use of the modeling power of SHLPNs algorithms for constructing the reachability tree and the invariants of a High Level Petri Net are necessary.

The construction of reachability trees for HLPNs is still an open problem. The “obvious” approach, unfolding a HLPN into the equivalent PN is impractical, it leads to an explosion of the reachability set. Methods to reduce the size of the reachability set of HLPN models need to be investigated.

One approach to solve this problem is reported by Jensen and his co-workers, [3], [4]. In their method some subtrees of a reachability tree of a HLPN are combined based upon the equivalent marking concept. The equivalence relation which partitions the set of markings into equivalence classes can be quite general and equivalence classes of markings can be determined only after a careful analysis of the model. Consequently, no general algorithms for constructing classes of equivalent markings are known. Moreover, information concerning some markings is lost and cannot be obtained from
the reachability tree of a Coloured Petri Net, since some subtrees do not appear in the tree.

The equivalence of markings is based on the fact that a class of entities flowing through the system exhibit an identical behavior, and they move in a particular subnet of the HLPN model. The only distinction between such entities is the identity of the token carried by each entity. If, in addition, the system consists of identical processing procedures, i.e., we do not distinguish the individual entities, it is possible to lump together a number of markings in order to obtain a smaller reachability tree for a HLPN model of the system.

In this paper we introduce the concept of marking variable to characterize a set of equivalent markings. Information about each individual marking in a set of equivalent markings, can be obtained from the domain of the corresponding marking variable. The representation power of marking variables introduced in this paper is considerably greater than that of individual markings. Equally important is the fact that by introducing marking variables it becomes feasible to design simple procedures to test in a systematic way for equivalence of marking variables. In this case we only need to observe the identity of the token types and the relation between the tokens as shown by transition colours enabled at the marking variables.

The paper is organized as follows. Section 2 introduces formally the concepts and the terminology related to High Level Petri Nets. Section 3 defines the token type and introduces High Level Petri Nets with marking variables. Reachability trees for HLPNs with marking variables are discussed in Section 3 and an algorithm for the construction of the trees is presented in Section 4. Section 5 illustrates the algorithm with an example based upon the known HLPN model of the philosopher system.

2. DEFINITIONS AND NOTATIONS

This section introduces notations and terminology to be used throughout the paper. Multi-sets, multi-set extension of functions, High Level Petri Nets and related concepts as the incidence matrix of a HLPN, the enabling of a marking, reachable marking, reachability set are defined formally.

Denote by \( N \) the set of all non-negative integers and \( Z \) the set of all non-negative integers including zero.

2.1 Definition: A multi-set \( S' \) is a function defined on a non-empty set \( S \), \( S' \in [S \rightarrow N] \).

Intuitively, a multi-set is a set which can contain multiple occurrences of the same element. In this paper only finite multi-sets are considered. Given the multi-set \( S' \) let \( s' \in S' \). Denote by \( |s'| \) the multiplicity of \( s' \).
2.2 Definition: Given two non-empty sets \( S \) and \( R \) and a function \( f \in [S \rightarrow R] \) let \( S' \) and \( R' \) be multi-sets defined independently on \( S \) and \( R \) respectively. A function \( f' \in [S' \rightarrow R'] \) is called a multi-set extension of \( f \) iff

\[
\forall s' \in S' \quad f'(s') = f(s)
\]

and

\[
|s'| = |f'(s')|.
\]

The multi-set extension function \( f' \) is a one-to-one function iff \( f \) is a one-to-one function and if the following two conditions are true

\[
s'_1 \neq s'_2 \quad \Rightarrow \quad f'(s'_1) \neq f'(s'_2)
\]

and

\[
s'_1 = s'_2 \quad \Rightarrow \quad f(s'_1) = f'(s'_2).
\]

Similarly, when \( f \) is a onto function and \( \text{range}(f') = R' \), then \( f' \) is a onto function.

Several definitions of High Level Petri Nets are available in the literature. Here a definition based upon [1] and [4] is given.

2.3 Definition: A High Level Petri Net is a 8-tuple \( H = \langle P, T, A, V, D, X, W, M_0 \rangle \), where

\( P \): is a finite, non-empty set of places,

\( T \): is a finite, non-empty set of transitions,

\( A \): is a finite, non-empty set of atomic colours,

\( V \): is a finite set of variables,

\( D \): is a function defined on \( V \) such that for each variable \( v \in V \), \( D(v) \) is a set of atomic colours called the domain of the variable \( v \).

\( X \): is a colour function defined on \( P \cup T \). \( X(P) \) represents the set of place colours for a given maximal arity of places, \( n \), \( X(P) = \bigcup_{0 \leq k \leq n} A^k \) with \( A^k = \{<>\} \). \( X \) associates to each place a set of possible token colours, i.e., \( \forall p \in P \), \( X(p) \subseteq X(P) \). \( X(T) \) represents the set of transition colours. \( X \) attaches to each transition a set of possible occurrence colours, i.e., \( \forall t \in T \), \( X(t) \) is the set of substitutions of all variables appearing free in the immediate surrounding arc-expressions of \( t \) and \( X(T) = \bigcup_{t \in T} X(t) \).

\( W \): is an arc function defined on \( (P \times T) \cup (T \times P) \). It indexes family of multi-sets over \( \bigcup_{0 \leq k \leq n} (A \cup V)^k \), i.e.,

\[
\forall (x,y) \in (P \times T) \cup (T \times P), \quad W_{x,y} : \bigcup_{0 \leq k \leq n} (A \cup V)^k \rightarrow N.
\]

\( M_0 \): is called the initial marking of \( H \). A marking of \( H \) is a \( P \)-indexed family of multi-sets over \( X(P) \) such that

\[
\forall p \in P \quad X(p) \rightarrow N.
\]
It is known that each High Level Petri Net with a finite set of colours can be transformed into an equivalent Place/Transition Net obtained through unfolding each place \( p \) into the set of places \( \{(p,a) | a \in X(p)\} \), and unfolding each transition \( t \) into the set of transitions \( \{(t,\sigma) | \sigma \in X(t)\} \). At times instead of a transition \( t \) we will talk about the step \( (t,\sigma) \) if \( \sigma \in X(t) \).

2.4 Definition: The incidence matrix of a High Level Petri Net \( H \), is the matrix \( C = (C_{p,t}) \) for all \( p \in P \), and \( t \in T \) with \( C_{p,t} \) defined as \( C_{p,t} = W_{t,p} - W_{p,t} \).

Thus

\[ C_{p,t} = \bigcup_{0 \leq k \leq n} (A \cup X)^k \rightarrow Z. \]

Let \( W_{p,t}(\sigma) \) be the multi-set obtained from \( W_{p,t} \) by substituting the free variables by atomic colours according to \( \sigma \). For instance, if

\( \sigma = (x \leftarrow b, y \leftarrow a) \) and \( W_{p,t} = <a, x> + <x, y> \) then \( W_{p,t}(\sigma) = <a, b> + <b, a> \).

2.5 Definition: The transition \( t \) is enabled at the marking \( M \) iff:

\( \exists \sigma \in X(t) \text{ such that } W_{p,t}(\sigma) \leq M(p) \forall p \in P. \)

We will say that the step \( (t,\sigma) \), rather than transition \( t \) is enabled if a colour function \( \sigma \) exists, i.e. \( \exists \sigma \in X(t) \). The step \( (t,\sigma) \) is not enabled if a colour function \( \sigma \) does not exist, i.e. \( (t,\sigma) = 0 \) if \( \sigma \notin X(t) \).

2.6 Definition: When a step \( (t,\sigma) \) is enabled at \( M_1 \), it can fire and transform marking \( M_1 \) into a directly reachable marking \( M_2 \) defined as

\( M_2(p) = M_1(p) - W_{p,t}(\sigma) + W_{t,p}(\sigma) \forall p \in P. \)

Denote the \( t \)-th column of the incidence matrix \( C \) by \( C^t \) and note that \( C^t \) is associated with transition \( t \). To underline the fact that \( M_2 \) is reached from \( M_1 \) when transition \( t \) fires, the definition of the follower marking presented above can be rewritten as

\( M_2 = M_1 + C^t(\sigma). \)

Let \( q \) be a step sequence, the equivalent of a firing sequence

\( q = [(t_1,\sigma_1), (t_2,\sigma_2), \ldots, (t_k,\sigma_k)]. \)

2.7 Definition: A marking \( M \) is reachable from the initial marking \( M_0 \) iff a step sequence exists such that

\( M = M_0 + C \ast q. \)

Here \( C \ast q \) is defined as
\[ C \otimes q = \sum_{(t, \alpha) \in q} C \otimes (t, \alpha) = \sum_{(t, \alpha) \in q} C'(\alpha). \]

2.8 **Definition:** The reachability set corresponding to the initial marking \( M_0 \) is denoted as \( [M_0] \) and it is defined as the set of all markings which are reachable from some initial marking \( M_0 \).

Throughout the paper it is assumed that token colours do not appear explicitly in the net. This restriction does not affect most of the models of interest.

### 3. HIGH LEVEL PETRI NETS WITH MARKING VARIABLES

In this section we introduce the token type and token variable, concepts which are necessary to define High Level Nets with marking variables.

**Token Type and Token Variables**

3.1 **Definition:** Two atomic tokens \( a \) and \( b \) are of the same type denoted by \( A_i \), i.e., \( a \in A_i \) and \( b \in A_i \) iff

\[ \forall \nu \in V \ a \in D(\nu) \iff b \in D(\nu) \]

Note that in case of Stochastic High Level Petri Nets [5], there is another condition to decide that two tokens are of the same type. The firing rates of all transitions \( t \) the tokens flow through should be the same for both tokens.

According to the decision condition we divide the atomic colour set \( A \) into disjoint sets of atomic tokens types. If \( A_i \) and \( A_j \), both are type sets, then \( A_i \cap A_j = 0 \) and \( \cup A_i = A \).

Let us now consider compound tokens, \( a \) and \( b \) such that \( a, b \in A^k \). Denote by \( a_i \) and \( b_i \), \( 1 \leq i < k \) the \( i \)-th element of \( a \) and \( b \) respectively.

3.2 **Definition:** Two compound tokens are of the same type of compound tokens iff all \( a_i \) and \( b_i \) are atomic tokens of the same type.

In the same way, we can divide entire \( X(P) \) into type sets of tokens.

Let us now consider two properties of tokens of the same type:

- They have the same subnet, in which the colours of the places include them, because they are always represented by the same variables.
- They have the same behavior, since they have the same subnet and the same representations in the arcs (in the SHLPN, as well as the same corresponding firing rates).
For atomic tokens we use a two-ary vector variable to represent an atomic colour. Its first attribute is the token type and the second attribute is the token identity, defined either as a rotation variable, a permutation variable or an identity-function. These symmetry type variables should have the same domain for the same type of tokens, but they have to have the different values at the same time, such that they identify the different tokens.

The compound token variables can be constructed by the atomic token variables. In compound token vector variable, each element is still an atomic token vector variable.

The acceptance of the suitable symmetry type variables is determined by the person who analyzes the system and it must obey the inherent nature of the system. In fact, suitable symmetry variables and their relations are present in the net. It is possible to get the available presentation for the token variables from the incidence matrix of a HLPN.

Definition of High Level Petri Nets With Marking Variables

In this section we present a formal definition of High Level Petri Nets with marking variables and we redefine concepts like enabling of a transition or enabling of a step, reachable marking, and reachability set.

3.3 Definition: Let \( H = <P,T,A,V,D,X,W,M_0> \) be a High Level Petri Net. Let \( \Phi \) be a set of symmetries such that each symmetry function \( \phi \in \Phi \) is related to several atomic colour subsets of \( A \). This means that each \( a \in A \) has attached a symmetry variable. Then \( H' = <P,T,A',V,D',X',W,M_0'> \) is a High Level Petri Net with marking variables, iff

\[
\begin{align*}
A' &= \{ a' | a' = \phi(a) \text{ for all } a \in A \text{ and } \phi \in \Phi \}, \\
D'(v) &= \{ a' | a' = \phi(a) \text{ for all } a \in D(v) \text{ and } \phi \in \Phi \}, \text{ for all } v \in V, \\
X'(P) &= \bigcup_{0 \leq k \leq n} (A')^k \text{ with } (A')^0 = \{<>\}, \\
X'(p) &= \{ a' | a' \leftarrow \phi \rightarrow a, \text{ i.e., } a' = \phi(a) \text{ or } a'_i = \phi(a_i), \text{ where } a_i \text{ and } a'_i \text{ are separately the } i\text{-th element of } a \text{ and } a', \text{ for all } a \in X(p) \text{ and } \phi \in \Phi \}, \text{ for all } p \in P, \\
X'(t) &= \{ \sigma' | \forall (x \leftarrow a) \in \sigma \leftarrow (x \leftarrow \phi(a)) \in \sigma' \text{ for all } \sigma \in X(t) \text{ and } \phi \in \Phi \}, \text{ for all } t \in T. \\
M_0'(p)(a') &= M_0(p)(a) \text{ where } \forall a' \in X'(p), \forall a \in X(p) \text{ and } a' \leftarrow \phi \rightarrow a, \text{ for all } p \in P.
\end{align*}
\]
3.4 **Definition:** The transition \( t \) is enabled at the marking \( M' \) if \( \exists \sigma' \in X'(t), \forall p \in P, W_{p}(\sigma) \leq M'(p). \)

3.5 **Definition:** When a step \( (t, \sigma') \) is enabled at marking \( M'_1 \) it can fire and transform \( M'_1 \) into a directly reachable marking \( M'_2 \) defined by \( M'_2 = M'_1 + C'(\sigma') \).

3.6 **Definition:** A step sequence, \( q' \) is a sequence of steps defined by \( q' = \{(t_1, \sigma'_1), (t_2, \sigma'_2), \ldots, (t_k, \sigma'_k)\} \) such that \( q' : T \rightarrow X'(T) \).

3.7 **Definition:** A marking \( M' \) is reachable from the marking \( M'_0 \), if there exists a step sequence \( q' \) such that \( M' = M'_0 + C \ast q' \).

Where \( C \ast q' = \sum_{(t, \sigma') \in q'} C \ast (t, \sigma') = \sum_{(t, \sigma') \in q'} C'(\sigma') \).

3.8 **Definition:** The reachability set \([M'_0] \) is the set of all markings reachable from the initial marking \( M'_0 \).

3.9 **Definition:** A HLPN with marking variables is bounded at place \( p \in P \) and to the token \( a' \in X'(p) \) iff:

\[ k \in N \ \forall M' \in [M'_0] : M'(p)(a') \leq k \]

A High Level Petri Net with marking variables is bounded iff it is bounded at all places and all tokens.

4. **DEFINITION OF REACHABILITY TREES WITH MARKING VARIABLES**

To construct reachability trees for Place/Transition nets reference [2] introduces the following taxonomy of markings of the net: covering markings, duplicate markings, dead markings. Covering markings which introduce \( \omega \)-symbols limit the trees to a finite size. Duplicate markings cut away their same subtrees. Dead markings have no subtrees.

Let us now present a taxonomy of the marking variables for the High Level Petri Nets introduced in this paper.

- **Dead Marking Variables:** A marking variable is dead iff no step is enabled in it. No new marking variables can be produced.
- **Duplicated Marking Variables**: Marking variables which have previously appeared in the tree are called duplicate marking variables. \( M'_1 \) is a duplicate marking variable of \( M'_2 \) iff:

\[
\forall p \in P \ \forall a' \in X'(p) \ M'_1(p)(a') = M'_2(p)(a')
\]

- **Equivalent Marking Variables**: Represent a generalization of duplicate marking variables. Two marking variables \( M'_x \) and \( M'_y \) are equivalent iff condition (C1) is satisfied:

(C1) \( \forall p \in P, \forall a', b' \in X'(p), \ M'_x(p)(a') = M'_y(p)(b') \) where \( a' \) and \( b' \) are the same type of tokens.

Let \( q \) be the multi-set of steps enabled at the marking variable \( M'_x \) and let \( g \) be the multi-set of steps enabled at the marking variable \( M'_y \). There must be a one-to-one and onto function \( r \) between \( q \) and \( g \) such that for

\[
\forall(t_1, \sigma'_1) \in q \ \text{and} \ \forall(t_2, \sigma'_2) \in g \ \Rightarrow \ r((t_1, \sigma'_1)) = (t_2, \sigma'_2)
\]

In addition for any variable \( y \) substituted, \( \sigma'_1(y) \) and \( \sigma'_2(y) \) are the same type of atomic tokens.

- **Covering Marking Variables**: When a marking variable \( M'_x \) strictly covers the marking variable \( M'_y \) of a predecessor, the step sequence transforming \( M'_x \) into \( M'_y \) can be repeated infinite times. Thus, it is possible to get an arbitrarily large value for each coefficient which has increased from \( M'_x \) to \( M'_y \). In the tree, we indicate this by substituting \( \omega \)-symbol for each such coefficient. Given two marking variables \( M'_x \) and \( M'_y \) of a predecessor, \( M'_x \) is said to cover strictly \( M'_y \) iff condition (C2) is satisfied:

(C2) \( \forall p \in P, \forall a', b' \in X'(p), \ M'_x(p)(a') \geq M'_y(p)(b') \) and \( \exists p \in P, \ a', b' \in X'(p), \ M'_x(p)(a') > M'_y(p)(b') \) where \( a' \) and \( b' \) are the same type of tokens.

We sketch now the algorithm used to construct the reachability tree for a HLPN with marking variables, which is reduced with respect to covering marking variables and equivalent marking variables. Let us consider a \( M'_x \) to be processed in sequence (which is generated by Definition 3.5). The algorithm consists of the following steps

**Step 1. Determine the Node Type**

1.1 The new node \( x \) is equivalent to an already existing node \( y \) and is a leaf node of the tree iff \( M'_x \) and \( M'_y \) satisfy condition (C1).

1.2 If no step is enabled for the marking variable \( M'_x \), then the node \( x \) is a dead node and it is a leaf node.
1.3 If the node \( x \) strictly covers one of its predecessors \( y \), i.e., the condition (C2) is satisfied, then we assign \( M_y(p)(a') = \omega \) for satisfying the condition above \( M_y(p)(a') > M_y(b)(b') \).

1.4 If \( x \) is not equivalent to an existing node, it is not a dead node and it does not cover one of its predecessors then node \( x \) is a normal node.

Step 2. Each node contains a marking variable and related information which indicates whether the marking variable is equivalent to the marking variable of an earlier processed node, covering the marking variable of a predecessor or it is a dead node. When this node is a normal node, the information is empty.

Step 3. An arc-label is a list of occurrence information. Each element is a step \((t, \sigma)\) where \( t \in T \) and \( \sigma' \in X'(t) \). Each step is enabled at the father-marking variable. An occurrence of the first step in the list results in the son-marking variables, and occurrences of other steps result in brother-marking variables, which are not equivalent to the son-marking variables. Other marking variables, which are transformed by steps in this list and are equivalent to the son-marking variables, disappear in the tree.

5. AN EXAMPLE

To illustrate the simplicity and the power of our method, let us consider the philosopher system, consisting of five philosophers who alternately think and eat. There are only five forks on a circular table and there is a fork between two philosophers. Each philosopher needs to use two forks adjacent to him when he eats. Obviously two neighbors cannot eat at the same time. The philosopher system is described by the High Level Petri Net, shown in Figure 1. The model has three places and two transitions. The significance of places and transition is

\- T: the "thinking" place. If \( T \) holds tokens, the corresponding philosophers are thinking or waiting for eating.
\- E: the "eating" place. If \( E \) holds tokens, the corresponding philosophers are eating.
\- F: the "free forks" place. If \( F \) holds tokens, the corresponding forks are free.
\- G: the "getting forks" transition.
\- R: the "releasing forks" transition.

According to the method describe in Section 4, we define two types of tokens, philosopher tokens and fork tokens and we associate to both the same rotation symmetry variable. All markings are indexed on variable \( i \), \( i \in [1,5] \), as shown in Figure 2.

In the initial marking variable transition \( G \) can fire in all colours of philosopher tokens producing five equivalent marking variables. Only one is included in the tree, but information about the others is contained in the domain of this marking variable. In this way, five individual markings can be obtained from the third marking variable in
the tree.

This method can be applied to other examples like the models of a shared memory multiprocessor system and of the communication protocols presented in [5]-[7]. In the shared memory multiprocessor system model, the arbitrary permutation symmetry variable would be applied. Rotation is still the suitable symmetry variable for the communication protocol model.

6. CONCLUSIONS

In this paper we introduce the concept of a marking variable and then describe a simple algorithm for constructing reachability trees for High Level Petri Nets.

It is important to understand that the reachability tree with marking variables is a complete tree, but it is a dense tree. The reachability tree constructed according to the method discussed in [3] and [4] is not a complete tree. Moreover, in order to construct classes of equivalent markings a symmetry function is introduced as an external entity which is not integrated into the description of the net. As a result the algorithm proposed by Jensen is fairly complex and difficult to be implemented in the general case. Our method uses symmetry functions built into the net representation and this approach leads to a systematic construction of the reachability tree.

A software package based upon this method has been implemented and has proved the simplicity of the method described in this paper.

REFERENCES

Figure 1. Modeling of the philosopher system with HLPN.
Figure 2. A reachability tree with marking variables for the philosopher system (omitting representation fork colours in steps).