Siberian Division of the Russian Academy of Sciences
A. P. Ershov Institute of Informatics Systems

V. E. Kozura

UNFOLDINGS OF COLOURED PETRI NETS

Preprint
80

Novosibirsk 2000

In this paper the unfolding technique is applied to coloured Petri nets (CPN) [8,9]. The technique is formally described, two algorithms and three finitization criteria are considered. It is also shown how to use the unfolding technique taking into consideration symmetry or equivalence specifications presented in [9]. We require CPN to be finite, n-safe and containing only finite sets of colours.

Российская академия наук
Сибирское отделение
Институт систем информатики
им. А. П. Ершова

В.Е. Козюра

РАЗВЕРТКИ РАСКРАШЕННЫХ СЕТЕЙ ПЕТРИ

Препринт
80

Новосибирск 2000

В данной работе метод развертки применен к раскрашенным сетям Петри (РСП) [8,9]. Метод формально описан, приведены два алгоритма и три критерия финитизации. Также показано как применять метод развертки, используя спецификации симметрии или эквивалентности, описанные в [9]. На РСП накладываются ограничения конечности, n-безопасности и конечности множеств, представляющих цвета.

1. INTRODUCTION

The state space exploring in Petri net (PN) analysis is one of the most important approaches. Unfortunately, it faces the state explosion problem. Among the approaches which are used to avoid this problem are the stubborn set method, symbolic binary decision diagrams (BDD), methods based on partial orders, methods using symmetry and equivalence properties of the state space, etc. [13].

McMillan in [11] has proposed an unfolding technique for PN analysis. In his works, instead of the reachability graph, a finite prefix of maximal branching process, large enough to describe a system, has been considered.

The size of unfolding is exponential in the general case and there are few works which improve in some way the unfolding definitions and the algorithms of unfolding construction [6,10].

Initially McMillan has proposed his method for the reachability and deadlock analysis (which has also been improved in the later work [12]). J.Esparsa has proposed a model-checking approach to unfolding of 1-safe systems analysis [5]. In [1] the model-checking technique has been applied to timed PN. In [3,7,15] LTL-based model-checking has been developed.

Unfolding of CPN has been considered in the general case in [14] for using it in the dependence analysis needed by the Stubborn Set method. In this paper the unfolding method, as it was developed in later works for ordinary PN, is applied to CPN (as they are described in [8,9]). Three types of unfoldings and two algorithms for unfolding generation are considered.

In [9] symmetry and equivalence specifications for CPN are introduced. In this paper it is shown how to use the unfolding technique taking into consideration symmetry or equivalence specifications.

The paper is organized as follows: chapter 2 gives the main definitions of the CPN's theory and the subclass we are interested in, chapters 3 and 4 introduce the unfolding theory, chapter 5 gives two algorithms of the unfolding generation, chapter 6 gives the net examples, chapter 7 describes the deadlock checking technique that uses net unfoldings, chapter 8 describes how to work with net unfoldings in the presence of symmetry or equivalence specifications.

2. INTRODUCTION TO COLOURED PETRI NETS

In this section we briefly give the basic definitions related to CPN and describe the subclass of colours we will use in the paper. More detailed description of coloured Petri nets can be found in [8,9].

Definition 2.1. A *multi-set* is a function m: $S \rightarrow N$, where S is a usual set and N is the set of natural numbers.

In the natural way we can define operations such as m_1+m_2, $n \cdot m$, m_1-m_2, and relations $m_1 \leq m_2$, $m_1 < m_2$. Also $|m|$ can be defined as $|m| = \sum_{s \in S} m(s)$.

Let Var(expression) define the set of variables of expression and Type(expression) define the type of expression.

Definition 2.2. A *coloured Petri net CPN* is the net

$\boldsymbol{N}$ = (S,P,T,A,N,C,G,E,I),

where S,P,T,A are the sets of colours, places, transitions, and arcs such that $P \cap T = P \cap A = T \cap A = \varnothing$, N is a mapping N: $A \rightarrow P \times T \cup T \times P$, C is a colour function C: $P \rightarrow S$, G is a guard function such that for all $t \in T$ Type(G(t))=bool and Type(Var(G(t))) $\subseteq$ S, E is the function defined on arcs with Type(E(a)) = $C(p)_{MS}$, where p is the place from N(a) and Type(Var(E(a))) $\subseteq$ S and I is the initial function defined on places, such that for all $p \in P$ Type(I(p)) = $C(p)_{MS}$.

A(t), Var(t), A(x,y), E(x,y) can be defined in the natural way.

Definition 2.3. A *binding* b is a function from Var(t) such that $b(v) \in Type(v)$ and $G(t)<b>$. The set of bindings for t will be denoted by B(t)

Definition 2.4. A *token element* is a pair (p,c) where $p \in P$ and $c \in C(p)$. The set of all token elements is denoted by TE.

Definition 2.5. A *binding element* is a pair (t,b) where $t \in T$ and $b \in B(t)$. The set of all binding elements is denoted by BE.

Definition 2.6. A *marking* M is a multi-set over TE.

Definition 2.7. A *step* Y is a multi-set over BE.

Definition 2.8. A step Y is *enabled* in the marking M if for all $p \in P$ $\sum_{(t,b) \in Y} E(p,t)<b> \leq M(p)$ and a new marking M_1 is given by

$$M_1(p) = M(p) - \sum_{(t,b) \in Y} E(p,t)<b> + \sum_{(t,b) \in Y} E(t,p)<b>.$$

Now we can define a subclass of colored Petri nets which is large enough to describe many interesting systems and still allows us to build a finite prefix of its branching process. In the description we follow the CPN ML notation given in [8]. The main idea is to consider only finite color domains $s \in S$.
The set of basic color domains is obtained from the four basic types of Standard ML (SML):

color A = int with m..n // m < n
color B = bool
color C = unit
color D1 = string with "x".."y" and m..n // x < y and m < n
color D2 = string with s1|s2|...sn // the explicit enumeration.

Also the explicit specifications of finite colors are possible, such as:

color E = with X1|X2|…|Xn
color F = index expr with m..n,

and the sets obtained by the renaming procedure

color G = bool with (yes,no)
color H = unit with e.

From already defined color sets we can declare new color sets using constructor operators, such as:

color I = product A1×A2×A3×…×An
color J = record i:A1, j:A2, … k:An
color K = list A with m..n
color L = color A

All functions defined in [8] and having the above described classes as their domains are allowed in our subclass. The same can be told about the variables, constants, operators and net expressions. Below we give some examples:

Fun F1(n:A) = if n>2 then 1 else 2
Fun F2(x:E) = case x of p ⇒ 2'e | q ⇒ e

Definition 2.9. The CPN satisfying all the above-mentioned requirements is called *S-finite*.

Definition 2.10. The marking M of a CPN is *n-safe* if $|M(p)| \leq n$ for all $p \in P$. A CPN is called *n-safe* if all of its reachable markings are n-safe. 1-safe net is also called *safe*.

Definition 2.11. A *preset* of an element $x \in P \cup T$ denoted by ${}^\bullet x$ is the set ${}^\bullet x = \{y \in P \cup T \mid \exists a\text{: } N(a) = (y,x)\}$. A *postset* of x denoted by $x^\bullet$ is the set

$x^{\bullet} = \{y \in P \cup T \mid \exists a: N(a) = (x,y)\}$.

The CPN considered in this paper are the CPN satisfying three additional properties:

1. The number of places and transitions is finite.
2. The CPN is n-safe.
3. The CPN is S-finite.

If the opposite is not mentioned, the term CPN has the meaning of a CPN, satisfying these three properties.

3. BRANCHING PROCESS OF COLOURED PETRI NETS.

Let ***N*** be a Petri net. We will use the term *nodes* for both places and transitions.

Definition 3.1. The nodes x_1 and x_2 are *in conflict*, denoted by $x_1 \# x_2$, if there exist transitions t_1 and t_2 such that ${}^{\bullet}t_1 \cap {}^{\bullet}t_2 \neq \varnothing$ and (t_1,x_1) and (t_2,x_2) belong to the transitive closure of N (which we denote by $\mathbf{R_t}$). The node x is in *self-conflict* if $x \# x$. We will write $x_1 \leq x_2$ if $(x_1,x_2) \in \mathbf{R_t}$ and $x_1 < x_2$ if $x_1 \leq x_2$ and $x_1 \neq x_2$.

Definition 3.2. We say that x *co* y, or $x \parallel y$, or x *concurrent* y if neither $x < y$ nor $x > y$ nor $x \# y$.

Definition 3.3. An *Occurrence Petri Net (OPN)* is a usual Petri net ***N*** = (P,T,N), where

(1) P,T are the sets of places and transitions,

(2) $N \subseteq P \times T \cup T \times P$ gives us the incidence function,

satisfying the following properties:

(a) For all $p \in P$ $|{}^{\bullet}p| \leq 1$,

(b) N is acyclic, i.e., the (irreflexive) transitive closure of N is a partial order.

(c) N is finitely preceded, i.e. for all $x \in P \cup T$ the set $\{ y \in P \cup T \mid y \leq x\}$ is finite which gives us the existence of Min(***N***), the set of minimal elements of ***N*** with respect to $\mathbf{R_t}$ (which is considered to contain only the elements from P).

(d) no transition is in self conflict.

Every place $p \in P$ may have some tokens. The initial marking of an OPN M_0 of ***N*** is defined by $M_0(p) = 1$ if $p \in$ Min(***N***) and empty otherwise. If for transition

$t \in T$ we have $M(p)>0$ for all $p \in {}^{\bullet}t$, then t may occur and the obtained marking M_1 is given by $M_1 = M - M({}^{\bullet}t) + M(t^{\bullet})$.

Proposition 3.1. OPN is a 1-safe net.

Proof. The initial marking is 1-safe by definition. Using the restriction $|{}^{\bullet}p| \leq 1$ from the OPN definition, we have that, from the 1-safe marking by the occurrence of any $t \in T$, we can obtain only 1-safe marking. Otherwise we have a contradiction either with the property (b) in the case of $p \in Min(\boldsymbol{N})$ or with the above mentioned property (a) from the OPN definition. ■

Definition 3.4. Let $\boldsymbol{N_1} = (P_1,T_1,N_1)$ and $\boldsymbol{N_2} = (P_2,T_2,N_2)$ be two Petri nets. A *homomorphism* h from $\boldsymbol{N_2}$ to $\boldsymbol{N_1}$ is a mapping $h: P_2 \cup T_2 \rightarrow P_1 \cup T_1$ such that

(a) $h(P_2) \subseteq P_1$ and $h(T_2) \subseteq T_1$.

(b) for all $t \in T_2$ $h|_{{}^{\bullet}t} = {}^{\bullet}t \rightarrow {}^{\bullet}h(t)$.

for all $t \in T_2$ $h|_{t^{\bullet}} = t^{\bullet} \rightarrow h(t)^{\bullet}$.

Now we give the main definition of the chapter. This is the first novelty of the paper, a formal definition of a branching process for coloured Petri nets. After the following definition, the existence result is proven.

Definition 3.5 : A *branching process* of a CPN $\boldsymbol{N_1} = (S_1, P_1, T_1, A_1, N_1, C_1, G_1, E_1, I_1)$ is a tuple $(\boldsymbol{N_2}, h, \varphi, \eta)$, where $\boldsymbol{N_2} = (P_2, T_2, N_2)$ is an OPN, h is a homomorphism from $\boldsymbol{N_2}$ to $\boldsymbol{N_1}$, φ and η are the functions from P_2 and T_2, respectively, such that

(a) $\varphi(p) \in C(h(p))$.

(b) $\eta(t) \in B(h(t))$.

Other requirements are listed bellow:

(c) $Min(\boldsymbol{N_2}) == M_0$.

Here and further the double equality operator means two equal multi-sets of token elements. This also can be written in the following way: for all $p_1 \in P_1$ $\Sigma_{(p \in A)}\varphi(p) = M_0(p_1)$, where $A = \{ p \in Min(\boldsymbol{N_2}) \mid h(p) = p_1\}$.

(d) $G(h(t))<\eta(t)>$ for all $t \in T_2$.

(e) $\forall t' \in T_2 \mid (\exists a \in A_1: N_1(a)=(p,t)$ and $h(t')=t) \Rightarrow$
$E(a)<\eta(t')>=\Sigma_{(p' \in I)}\varphi(p')$, where $I=\{p' \in {}^{\bullet}t' \mid h(p') = p\}$.
$\forall t' \in T_2 \mid (\exists a \in A_1: N_1(a)=(t,p)$ and $h(t')=t) \Rightarrow$
$E(a)< \eta(t)>=\Sigma_{(p' \in I)}\varphi(p')$, where $I=\{p' \in (t,b)^{\bullet} \mid h(p') = p\}$.

(f) If $(h(t_1)=h(t_2))$ and $(\eta(t_1)=\eta(t_2))$ and $({}^{\bullet}t_1 = {}^{\bullet}t_2)$ then $t_1=t_2$.

Important Note: Using the first two properties, we can associate a token element (p,c) of $\boldsymbol{N_1}$ with every place in $\boldsymbol{N_2}$ and the binding element (t,b) of $\boldsymbol{N_1}$ with every transition in N_2**.** So we can further consider the net N_2 as containing the places which we identify with token elements of N_1, and transitions which we identify with binding elements of N_1. So we sometimes use them instead, like h((t,b))=t means h(t')=t and η(t')=b or $p \in {}^{\bullet}(t,b)$ means $p \in {}^{\bullet}t'$ and h(t')=t and η(t')=b. Analogously, we can consider $(p,c) \in P_2$ as $p' \in P_2$ and h(p')=p and φ(p)=c. Also, h(p,c)=p and h(t,b)=t.

It can be shown that any finite CPN has a maximal branching process (MBP) up to isomorphism (proposition 3.2). We can declare existence of the maximal branching process when considering the algorithm of its generation. As such an algorithm we choose the algorithm of unfolding generation proposed by McMillan [11] and applied to coloured Petri nets.

<u>Maximal Branching Process generation algorithm</u>

```
var: P2,T2,N2;
// Places and transitions are natural numbers, N2 is the set of pairs (m,n).
H_Table = {Ph_table[], Th_table[]}
 // This is a table for storing a homomorphism and functions φ and η
// Ph: n→(p,c),  Th: m→(t,b).
T_Fired;
// The list of waiting binding elements.
m, n : integer;
// The place and transition under construction.
// Using H_Table for simplification of the algorithm, we sometimes write
//(p,c) and (t,b) instead of the corresponding n and m.

begin
H_Table:=empty;
N2= ( P2,T2,N2): = ∅; n:=1;m:=1;
for all p∈P1 such that |I(p)|>0 do
   for all c∈I(p) do
     begin
       add(n , P2);
       n:=n+1;
       GenTr({n-1});
     end;
While (T_Fired ≠ ∅) do
 begin
   m0: = head(T_Fired) =  (t,b);
```

```
    delete(m0,T_Fired);
      for all a∈A1 such that N1(a) = (t,p) do
        for all c∈E(a)<b> do
          begin
            Ph_table[n]:=(p,c);
            add((m0,n), N2);
            add(n,P2);
            n:=n+1;
            GenTr({n-1})
          end;
    end;
return N2= ( P2,T2,N2);
end.

procedure GenTr(N);
begin
if  (¬∃t∈T1 | N⊆•t) then return
if Predecessors(N) has forward conflict then return
 for all (t,b)∈TE such that h(N)=•t do
     if (t,b) is enabled in M==N then
     // i.e M = Ph_table[N]
        begin
          add( (N,m), N2);
          insert m = (t,b) in T_Fired in order of |LocalConfig(m)|
          Th_table[m]:=(t,b);
          add(m,T2);
          m:=m+1;
        end;
 for all n∈P2 \ N do
   GenTr(N∪{n});
end.
```

Proposition 3.2. The algorithm gives us the maximal branching process $MBP(N_1)$ of N_1.

Proof:

(1) $\boldsymbol{N_2}= (P_2,T_2,N_2)$ is an Occurence Petri Net (OPN).

(a) $|{}^\bullet p|\leq 1$. We can come to a situation of having $N_2(m,n)$ only when calling $add((m,n),N_2)$. It is called together with $add(n,P_2)$ and the increasing of n by one.

(b) The obtained net is acyclic. While the value of n grows monotonically, the cycle is impossible.
(c) The net is finitely preceded. Since the initial CPN is finite and S-finite, I(p) is also finite.
(d) No transition is in self-conflict. This is checked directly in GenTr(N).

(2) A homomorphism h is given by H_Table.
(a) $h(P_2)\subseteq P_1$, $h(T_2)\subseteq T_1$. This can be seen directly from the H_Table.
(b) for all $t\in T_2$ $h\mid_{\bullet t} = {}^{\bullet}t \rightarrow {}^{\bullet}h(t)$. This means ${}^{\bullet}(t,b) \rightarrow {}^{\bullet}t$, which follows from the condition $h(N)={}^{\bullet}t$ in GenTr(N).
for all $t\in T_2$ $h\mid_{t\bullet} = t^{\bullet} \rightarrow h(t)^{\bullet}$. This follows from the condition $N_1(a) = (t,p)$ followed by the procedure add((m_0,n), N_2), where $m_0=(t,b)$ and Ph_table[n]=(p,c) in the main part of the algorithm.

(3) The algorithm gives us the Branching Process of $\boldsymbol{N_2}= (P_2,T_2,N_2)$.
(a,b) The functions $\varphi(p) \in C(h(p))$ and $\eta(t) \in B(h(t))$ are given by the H_Table[] .
(c) $Min(\boldsymbol{N_2}) == M_0$.
$M_0(p)=I(p)=\sum_{(c\in I(p))} c$.
By the algorithm construction:
$M == Min(\boldsymbol{N_2}) = \sum_{(p\in P1\mid |I(p)|>0)}\sum_{(c\in I(p))} (p,c)$.
If $(I(p) \neq \varnothing)$ $M(p) = \sum_{(c\in I(p))} c = M_0(p)$.
(d) The fact G(h(t,b))<b> follows from the way we choose (t,b) to be added to T_Fired. (t,b) is enabled $\Rightarrow$ G(t)<b> $\Rightarrow$ G(h(t,b))<b>.
(e) $\forall(t,b)\in T_2 \mid (\exists a: N_1(a)=(p,t)) \Rightarrow E(a)\langle b\rangle=\sum_{(p'\in I)}\varphi(p')$,
where $I=\{p'\in{}^{\bullet}(t,b) \mid h(p') = p\}$.
(t,b) is included in T_Fired iff (t,b) is enabled in M==N where $h(N)={}^{\bullet}t$.
$\Rightarrow E(p,t)\langle b\rangle \leq M(p)$. In our case $E(a)\langle b\rangle = E(p,t)\langle b\rangle = M(p)$, since the argument N in GenTr(N) is increased monotonically by one.
$M(p) = \sum_{((p,c)\in N)} c = \sum_{(p'\in I)}\varphi(p')$
$\forall(t,b)\in T_2 \mid (\exists a: N_1(a)=(t,p)) \Rightarrow E(a)\langle b\rangle=\sum_{(p'\in I)}\varphi(p')$,
where $I=\{p'\in(t,b)^{\bullet} \mid h(p') = p\}$.
When constructing the output places of (t,b), we do the following:

```
        for all a∈A1 such that N1(a) = (t,p) do
          for all c∈E(a)<b> do
            begin
              Ph_table[n]:=(p,c);
              add((m0,n), N2);
              add(n,P2);
```

…

end;

Here m_0=(t,b), so E(a)<b>= $\sum_{(c\in E(a)<b>)}c = \sum_{(p'\in I)}\varphi(p')$.

(f) If $(h(t_1)=h(t_2))$ and $(\eta(t_1)=\eta(t_2))$ and $({}^\bullet t_1 = {}^\bullet t_2)$, then $t_1=t_2$.
The fact follows from the impossibility of $N_1 = N_2$, such that $GenTr(N_1)$ and $GenTr(N_2)$ both are called.
The algorithm GenTr(N) starts with {n} and increases this set by passing through the subsets of {1..n-1} and adding them to {n}.

(4) The obtained Branching Process is maximal.
It is sufficient to prove that we cannot add one more transition (t,b) to $\boldsymbol{N_2}$. After adding additional places or arcs, we obtain direct contradictions to definition 3.5 (c) or (e).
If the transition (t,b) was added, then consider the set $N={}^\bullet(t,b)$.
Let n be the maximal element in N. Then, when adding n to P_2,we should call GenTr({n}) which should find the set N and generate the transition (t,b). ■

This branching process can be infinite even for the finite nets if they are not acyclic. We are interested to find a finite prefix of a branching process large enough to represent all the reachable markings of the initial CPN. This finite prefix will be called an unfolding of the initial CPN. In the next section we give the definitions of a configuration, cutoff points and the definition of unfolding of CPN.

4. UNFOLDINGS OF CPN

Definition 4.1. A *configuration* C of an OPN $\boldsymbol{N}$ = (P,T,N) is a set of transitions satisfying the following two conditions:
(1) $t \in C \Rightarrow$ for all $t_0 \le t$: $t_0\in C$
(2) for all $t_1,t_2\in C : \neg(t_1\# t_2)$.

Definition 4.2. A set $X_0 \subseteq X$ of nodes is called a *co-set*, if for all $t_1, t_2\in X_0$: (t_1 co t_2).

Definition 4.3. A set $X_0\subseteq X$ of nodes is called a *cut*, if it is a maximal co-set with respect to the set inclusion.
Finite configurations and cuts are closely related. Let C be a finite configuration of an occurrence net, then $Cut(C) = (Min(\boldsymbol{N})\cup C^\bullet) \setminus {}^\bullet C$ is a cut.

Definition 4.4. Let $\boldsymbol{N_1} = (S_1,P_1,T_1,A_1,N_1,C_1,G_1, E_1,I_1)$ be a CPN and MBP($\boldsymbol{N_1}$) =

($\boldsymbol{N_2}$, h, φ, η), where $\boldsymbol{N_2}$ = (P_2,T_2,N_2) , be its maximal branching process. Let C be a configuration of $\boldsymbol{N_2}$. We define a marking *Mark(C)* == Cut(C) which is a marking of $\boldsymbol{N_1}$. Operator "==" has the same meaning as in definition 3.5 Mark(C)(p) = $\Sigma_{(p' \in Cut(C) \mid h(p') = p)} M_2(p')$.

Definition 4.5. Let ***N*** be an OPN. For all t∈T the configuration
[t] = {t'∈T | t' ≤ t } is called a *local configuration*. (The fact that [t] is a configuration can be easily checked).

Let us consider the maximal branching process for a given CPN. It can be noticed that MBP(***N***) satisfies the completeness property, i.e., for every reachable marking M of ***N*** there exists a configuration C of MBP(***N***) (i.e., C is the configuration of OPN) such that Mark(C) = M. Otherwise we could add a necessary path and generate a larger branching process. This would be a contradiction with the maximality of MBP(N).

Now we are ready to define three types of cutoffs used in the definition of unfolding. The first two definitions can be found in [5,11]. The last is the definition given in [10].

Definition 4.6. A transition t∈T of an OPN is a *GT_0-cutoff*, if there exists t_0∈T such that Mark([t]) = Mark([t_0]) and [t_0] ⊂ [t].

Definition 4.7. A transition t∈T of an OPN is a *GT-cutoff*, if there exists t_0∈T such that Mark([t]) = Mark([t_0]) and ||[t_0]|| < ||[t]||.

Definition 4.8. A transition t∈T of an OPN is a *EQ-cutoff*, if there exists t_0∈T such that

(1) Mark([t]) = Mark([t_0])
(2) ||[t_0]|| = ||[t]||
(3) ¬(t || t_0)
(4) there are no EQ-cutoffs among t' such that t'|| t_0 and ||[t']|| ≤||[t_0]||.

Definition 4.9. For a coloured Petri net ***N***, an *unfolding* is obtained from the maximal branching process by removing all the transitions t', such that there exists a cutoff t and t < t', and all the places p∈t'$^\bullet$. If Cutoff = GT_0(GT)-cutoffs, then the resulted unfolding is called *GT_0(GT)-unfolding*. GT_0(GT)-unfolding is also called the *McMillan unfolding*. If Cutoff = GT-cutoffs ∪ EQ-cutoff, then the resulted unfolding is called *EQ-unfolding*.

It has been shown that the McMillan unfoldings are inefficient in some cases. The resulting finite prefix grows exponentially, when the minimal finite prefix has only a linear growth.

The following proposition can be formulated for these three types of unfoldings.

Proposition 4.2. EQ-unfolding ≤ GT-unfolding ≤ GT_0- unfolding.

Proof: From the cutoff definitions, we have GT_0-cutoffs ⊂ GT-cutoff. By the definition of the McMillan unfolding, we have GT-unfolding ≤ GT_0- unfolding. In the definition of EQ-unfolding, Cutoff = GT-cutoffs ∪ EQ-cutoff and the rules for the unfolding construction are stronger. So we have EQ-unfolding ≤ GT-unfolding. ■

The following theorem presents the main result of this chapter.

Theorem 1. Let $\boldsymbol{N_1}$ be a CPN. Then for its unfoldings we have:
(1) EQ-unfolding, GT-unfolding and GT_0-infolding are finite.
(2) EQ-unfolding, GT-unfolding and GT_0-infolding are safe, i.e.,
if C and C' are configurations, then $C \subseteq C' \Rightarrow Mark(C') \in [Mark(C)\rangle$.
(3) EQ-unfolding, GT-unfolding and GT_0-infolding are complete, i.e.,
$M \in [M_0\rangle \Rightarrow$ there exists a configuration C such that Mark(C) = M.

Proof:

(1) Using proposition 4.2, we only need to prove the finiteness of GT_0-infolding. This will be done in three steps.

(a) Let d(t) denote the depth of the longest chain $t_1<t_2<...<t$ in GT_0-unfolding. For all t∈T, d(t) ≤ M+1, where M is the number of reachable markings in $\boldsymbol{N}$. M is finite because of the properties we require of the CPN used.

(b) For all $t' \in GT_0$-unfolding, $t'^{\bullet}$ and ${}^{\bullet}t'$ are finite. Let t' = (t,b). From the definition (e) of a branching process, we have $\forall(t,b)\in T_2 \mid (\exists a\colon N(a)=(p,t)) \Rightarrow E(a)\langle b\rangle=\sum_{(p'\in Ip)}\varphi(p')$, where $I_p=\{p'\in{}^{\bullet}(t,b) \mid h(p') = p\}$. ${}^{\bullet}t' = \{ I_p \mid \forall p\in{}^{\bullet}t'\}$. Notice that $|E(a)| = | I_p |$. The multi-sets we consider in the paper are m such that $|m|$<const. It follows that $|E(a)|$ < const. The finiteness of $|{}^{\bullet}t\,|$ follows from the finiteness of $\boldsymbol{N_1}$ and finally: $|{}^{\bullet}t'| = \sum_{(p\in\bullet t)}| I_p |$ < const. Using definition 3.5 (e) part 2, we can prove analogously that $|t'^{\bullet}|$<const.

(c) For all natural K there exists only finite number of transitions t∈T such that d(t) ≤ K. We prove this by induction on K. The base K = 0 is true. Let $T_K = \{ t \mid d(t) \leq K \}$ be a finite set. Let us prove the finiteness of T_{K+1}. $T_K^{\bullet}$ is finite by (b) and the induction hypothesis. ${}^{\bullet}T_{K+1} \subseteq T_K{}^{\bullet} \cup \mathrm{Min}(\boldsymbol{N})$. ${}^{\bullet}T_{K+1}$ is finite. Using the property (f) in the definition of a branching process, we have the finiteness of T_{K+1}.

(2) The fact that the unfolding of ***N*** is safe follows immediately from the safety of the branching process of ***N*** which can be proven by induction on $|E|=|C' \setminus C|$. Let us denote by $C \oplus E$ the fact that $C \cup E$ is a configuration and $C \cap E = \varnothing$.

(a) the base: $|E| = 1$, i.e., $E=\{(t,b)\}$ and $C' = C \oplus \{(t,b)\}$. Cut(C) is a marking of OPN. While C is causally closed, i.e., $\forall(t'b') < (t,b)$ such that $(t',b') \in C$, we have ${}^{\bullet}(t,b) \subseteq \mathrm{Cut}(C)$. In the OPN we have: $\mathrm{Cut}(C') = \mathrm{Cut}(C) - {}^{\bullet}(t,b) + (t,b)^{\bullet}$. Applying definition 3.5(e) and the homomorphism definition, we obtain:

$$\sum_{(p\in\bullet t)}\mathrm{Mark}(C')(p)=\sum_{(p\in\bullet t)}\mathrm{Mark}(C)(p)-\sum_{(p\in\bullet t)}E(a)\langle b\rangle+\sum_{(p\in t\bullet)}E(a)\langle b\rangle.$$

Besides, by definition 3.5 (d), $G(h(t,b))\langle b\rangle$. This means that $\mathrm{Mark}(C') \in [\mathrm{Mark}(C)\rangle$, because it is obtained by the occurrence of (t,b).

(b) If $E = \{(t_1,b_1)...(t_n,b_n)\}$, then choose $(t_i,b_i) \in E$ such that there is no $(t_j,b_j) \in E$ such that $(t_j,b_j) > (t_i,b_i)$. Consider the configuration $C_1=C \oplus (E \setminus \{(t_j,b_j)\})$ (It's easy to see that it's a configuration). $\mathrm{Mark}(C_1) \in [\mathrm{Mark}(C)\rangle$. Using the base step considerations, we have $\mathrm{Mark}(C') \in [\mathrm{Mark}(C_1)\rangle$ and finally $\mathrm{Mark}(C') \in [\mathrm{Mark}(C)\rangle$.

(3) Accordingly to proposition 4.2, it's sufficient to prove that EQ-unfolding of ***N*** is complete. Having the completeness of MBP(***N***), we will prove the following result. Let C' be a configuration of MBP(***N***). Then there exists a configuration C of EQ-unfolding of ***N***, such that Mark(C) = Mark(C'). If C' contains no cutoffs, then C' is the necessary configuration.
Else let $C_1...C_n$ be all configurations of MBP(***N***) of minimal size such that $\mathrm{Mark}(C_1) = \mathrm{Mark}(C_2) = ... = \mathrm{Mark}(C_n)$. This set is finite. Let for all i $C_i \notin$ EQ-unfolding. Let us seek for a contradiction. The previous means that C_i has at least one cutoff point. Let C_j contain the cutoff t_1 of maximal depth. There exists $t_2 \in$ EQ-unfolding, such that $\mathrm{Mark}([t_1]) = \mathrm{Mark}([t_2])$. We have two possibilities:

(a) t_2 is a GT-cutoff. This means that $|[t_2]| < |[t_1]|$. If $C_1 \subset C_2$, then there exists E such that $C_1 \oplus E = C_2$. In our case $C_j = [t_1] \oplus E$. Let $C^* = [t_2] \oplus E$. We have $\mathrm{Mark}(C^*) = \mathrm{Mark}(C_j)$ and $C^* < C_j$. So we have a contradiction, because C_j has a minimal size.

(b) t_2 is an EQ-cutoff. This means that $|[t_2]|=|[t_1]|$. Choose a configuration C_k among $C_1...C_n$ containing t_2. $C_k=[t_2]\oplus E$, where E is such that $C_j=[t_1]\oplus E$ and we have $|C_k| = |C_j|$. Since $\neg(t_1 \| t_2)$, we have $t_1 \notin C_k$ and $C_k \neq C_j$. C_k contains no cutoffs of depth $n>|[t_1]|$, because t_1 is of maximal depth among $C_1...C_n$. Also C_k

contains no cutoffs preceding t_2, because $t_2 \in$EQ-unfolding. This means that all cutoffs in C_k must be concurrent with t_2. Since t_2 is an image of EQ-cutoff t_1, then by definition 4.8(4) every cutoff $t_3 \in C_k$ is a GT-cutoff and its image t_4 is such that $\|[t_3]\| < \|[t_4]\|$ and we can apply the reasoning of the first case. ■

5. ALGORITHMS FOR FINITE PREFIX GENERATION.

In this section we give two algorithms: McMillan's algorithm and EQ-unfolding algorithm (the name doesn't mean that we cannot construct a McMillan unfolding by the second algorithm using the appropriate cutoff criteria). All unfoldings here will be constructed by the breadth-first traversal algorithms. The algorithm for generation of GT(GT_0)-unfolding is taken from [11] and the algorithm for construction of EQ-unfolding is taken from [10] and the latter is rather efficient in the speed of unfolding generation. In the case of an ordinary PN it gives the overall complexity $O(N_P N_T)$, where N_P and N_T are the numbers of places and transitions in EQ-unfolding. In our case a close estimation holds if we don't take into consideration the calculation complexity of arc and guard functions. In this case we obtain $O(N_P N_T B)$, where $B=\max\{|B(t)| \mid t \in T_{CPN}\}$. In the general case the first algorithm has an exponential complexity. Although the EQ-unfolding cannot guarantee the minimal sizes of N_P and N_T as it was made for 1-safe systems in [6], the size of EQ-unfolding is still much smaller in some cases than that of GT(GT_0)-unfoldings (which may grow exponentially). In the second algorithm we should store additionally two matrices N_P^2 and N_T^2.

McMillan's algorithm of GT(GT_0)-unfolding generation

```
var: P2,T2,N2;
H_Table = {Ph_table[], Th_table[]}
Hash_table[];
// HashTable for storage of local configurations.
T_Fired;
m, n : integer;
begin
H_Table:=empty; Hash_table:=empty;
N2= ( P2,T2,N2): = ∅; n:=1;m:=1;
for all p∈P1 such that |I(p)|>0 do
   for all c∈I(p) do
      begin
```

```
        add(n , P2);
        n:=n+1;
        GenTr({n-1});
      end;
While (T_Fired ≠ ∅) do
  begin
    m0: = head(T_Fired) =  (t,b);
    delete(m0,T_Fired);
      for all a∈A1 such that N1(a) = (t,p) do
        for all c∈E(a)<b> do
          begin
            Ph_table[n]:=(p,c);
            add((m0,n), N2);
            add(n,P2);
            n:=n+1;
            if not cutoff(m0)
             GenTr({n-1})
          end;
   end;
return N2= ( P2,T2,N2);
end.

procedure GenTr(N);
begin
if  (¬∃t∈T1 | N⊆•t) then return
if Predecessors(N) has forward conflict then return
 for all (t,b)∈TE such that h(N)=•t do
     if (t,b) is enabled in M==N then
      // i.e M = Ph_table[N]
        begin
            add( (N,m), N2);
            insert m = (t,b) in T_Fired in order of |LocalConfig(m)|
            Th_table[m]:=(t,b);
            add(m,T2);
            m:=m+1;
          end;
 for all n∈P2 \ N do
   GenTr(N∪{n});
end.
```

```
function cutoff(m): bool;
  begin
    M:==Cut(LocalConfig(m));
    for all t∈Hash_table[M] do
    // for GT0-unfolding : if t∈ LocalConfig(m)
       if (size(LocalConfig(t)) < size(LocalConfig(m)) ) return
    endfor
     add(m , Hash_Table[M]);
     return false;
  end;
```

If we have (t,b) enabled in few ways at the same step, we take all the possibilities into consideration, although the choice of a the unique set doesn't spoil the completeness of MBP. The obtained net $\boldsymbol{N_2}$ is evidently the prefix of MBP($\boldsymbol{N_2}$), and accordingly to the definition of a cutoff function it gives us GT(GT_0)-unfolding of $\boldsymbol{N_2}$. Due to the finiteness of GT_0-unfolding, the algorithm is correct.

EQ-unfolding will be constructed by a breadth-first traversal, tier by tier. A tier contains transitions of the same depth. We need two tiers to be stored: Current_tier and New_tier.

We need to store an array of transitions (TFired) and two matrices of the ordered relations Relation_T and Relation_P which are constructed on-the-fly. These matrices contain information about precedence, conflict and concurrency relations in the part of the unfolding which is already generated. On-the-fly construction of these matrices is made by inheriting the relations from the transitions (places) that serve as direct predecessors. For example, we will write the inheritance rules for transitions

- Precedence $t_j \Rightarrow t_i$, i.e $t_j \leq t_i$
  (1) $t_j \in {}^{\bullet}({}^{\bullet}t_i)$
  (2) $t_j \Rightarrow t_k$ and $t_k \in {}^{\bullet}({}^{\bullet}t_i)$
- Conflict $t_j \# t_i$
  (1) ${}^{\bullet}t_j \cap {}^{\bullet}t_i \neq \emptyset$
  (2) $t_j \# t_k$ and $t_k \in {}^{\bullet}({}^{\bullet}t_i)$
  (3) $t_k \Rightarrow t_j$ and ${}^{\bullet}t_k \cap {}^{\bullet}t_i \neq \emptyset$

EQ-unfolding algorithm

```
begin
  Reached = empty; TFired = empty; Current_T_tier = empty;
  Current_P_tier = {M0'};  // h(M0') = M0.
    do begin
```

```
      Reached = Reached ∪ Current_tier;
      generate_new_tier;
      Current_T_tier = New_T_tier;
      Current_P_tier = New_P_tier;
      is_unfolding_correct(Current_tier);
   while (Current_P_tier ≠ empty)
   return Reached;
end.

procedure generate_new_tier;
 begin
  New_T_tier = empty; New_P_tier = empty;
  for all (p,c)∈Current_P_tier \ {(p,c) | cutoff∈•(p,c)} do
   for all (t,b)∈TE | t∈p• do
     if enabled((t,b)) then
      // function enabled() uses the matrix N_P^2 for choosing the possible
      // sets of places containing (p,c) and having no forward conflicts.
      TFired = TFired ∪ (t,b);
   New_T_tier = {t_j∈TFired | ∀t_k∈TFired  |[ t_j]| ≤ |[ t_k]| }
    // TFired keeped hashed by the length of [t].
   TFired = TFired - New_T_tier;
   Update_Relations_T(New_T_tier);
   Check_cutoff(New_T_tier);
   for all (t,b)∈New_T_tier do
     begin
    New_P_tier = New_P_tier ∪{(p_i,c_j) | (∪p_i = t•) & (∑c_j =E(t,p)<b>)}
      Update_Relations_P( (t,b) );
     end;
 end;
```

The cutoff checking is made using the definitions and the relation matrices (see [10]. Here also the more detailed description of the algorithm can be found).

It takes $O(N_T^2)$ steps to calculate cutoffs = GT-cutoffs∪ EQ-cutoffs. Instead of $O(RO_p)$ for ordinary PN, we call the function enabled((t,b)) for every place of CPN $O(RO_pB)$ times (O_p is the maximal fan-out set, B is the maximal set of bindings, R is some constant, see[10]). The overall complexity of the algorithm for coloured Petri nets is $O(N_PN_TB)$, where $B=\max\{|B(t)| \mid t\in T_{CPN}\}$.

Initially we put $t_j \parallel t_i$ and $p_j \parallel p_i$.

Finally let us notice that, due to the symmetry of the conflict and concurrency relations and asymmetry of the precedence relation, the matrices can be kept triangle.

6. NET EXAMPLES

As an example let us consider the CPN representing the problem of dining philosophers (Fig. 1). For this net we have

GT_0-unfolding = GT-unfolding = EQ-unfolding.

Unfoldings of this net are represented on Fig.2. As it can be seen from the table bellow, the size of unfoldings is linear in the number of philosophers while the number of reachable markings is exponential.

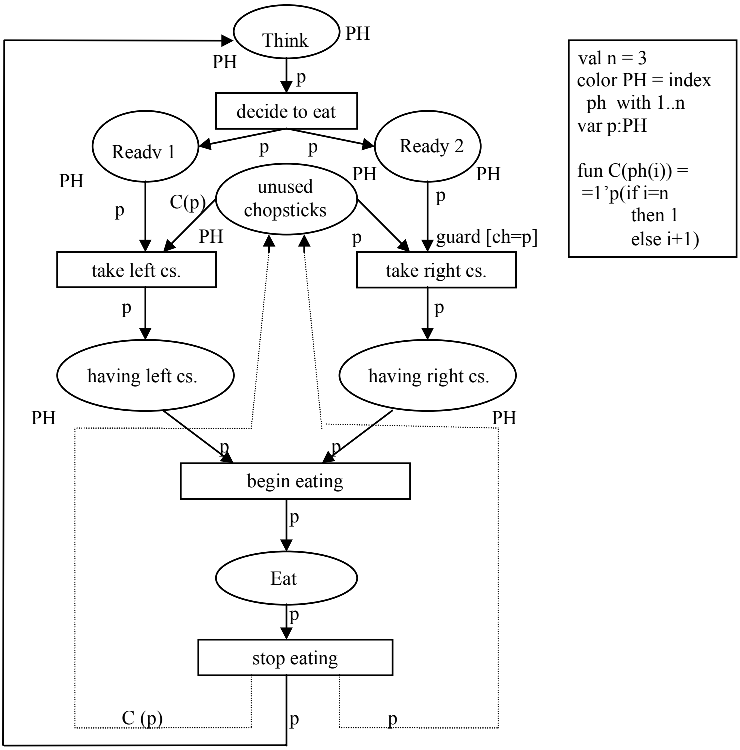


Fig.1. The Dining Philosophers Example

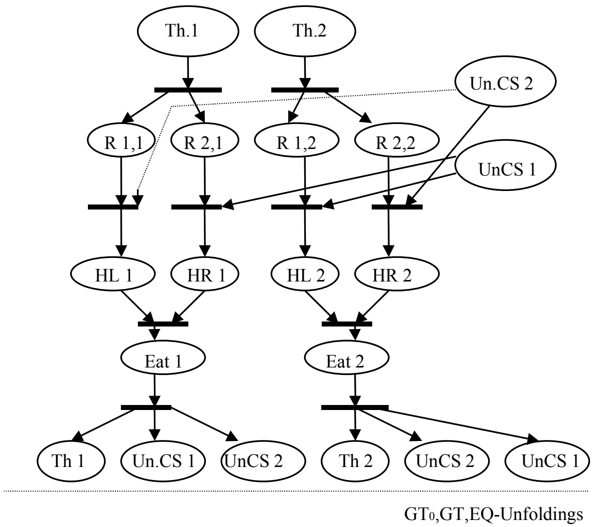


Fig. 2. Unfolding of the Dining Philosophers Example

N	the unfolding sizes (the numbers of transitions) GT_0,GT,EQ-unfoldings	Reachable Markings
2	10	22
3	15	100
4	20	466
5	25	2164

We measure the unfolding size by the number of transitions, because when storing the information about each place in every reachable marking, we have the analogy with storing the fan-out places for every transition. (Anyway, the number of fan-out places is restricted by some constant and doesn't spoil the linear growth of the unfolding size).

As can be seen from the table, the sizes of all unfoldings are equal. In the next example we have the exponential growth of GT_0 and GT-unfoldings $O(2^n)$, when the EQ-unfolding has only the linear growth O(n). The net is shown on Fig.3.

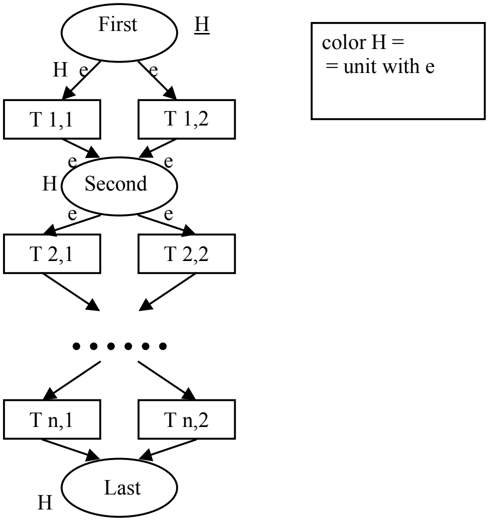


Fig. 3. An example of exponential growth of the McMillan unfolding

The last example is taken from [9] and represents the producer-consumer system (Fig.4). We consider the case when nb=1. The number of reachable markings is $N= (1+c+2*c*d+2*c*d^2)^p (1+p+2*p*d+2*p*d^2)^c(1+p*c*d^2)$.

The unfolding with np=nc=nd=1 is represented in Appendix. The unfolding consists of four parts. When a producer initially produces data, the part labelled PA is working (see Appendix). Part PB may work after a producer laid the first data to the buffer, but a consumer still cannot begin his part. Finally, PC is the part when a consumer definitely begins his work and a producer fulfills the buffer again. A consumer has the unique part CA. We have |PA|=|PB|=|CA|=5 and |PC|=4. The whole size is 19.
When adding either one more producer or one more consumer, we come to the situation of doubling of |PA|, |PB-1| and |CA| and adding the square of the number of parts |PC+1|. Adding one more data acts as adding the square number of possibilities. Finally the size of the unfolding is $UnfSize = |PA|*np*nc*nd^2 + +|CA|*np*nc*nd^2 + |PB-1|*np*nc*nd^2 + |PC+1|*(np*nc*nd^2)^2$.

```
val np=5; val nc=5; val nb=2; val nd=5;
color Prod = index  with 1..np
color Cons = index with 1..nc
color Data = index  with 1..nd
color Prod×Cons = product Prod*Cons
color HalfPack = product Prod*Cons*Data
color FullPack = product Prod*Cons*Data*Data
color ListFullPack = list FullPack
var p:Prod;  var c:Cons;  var d1,d2:Data
var List:ListPack
```

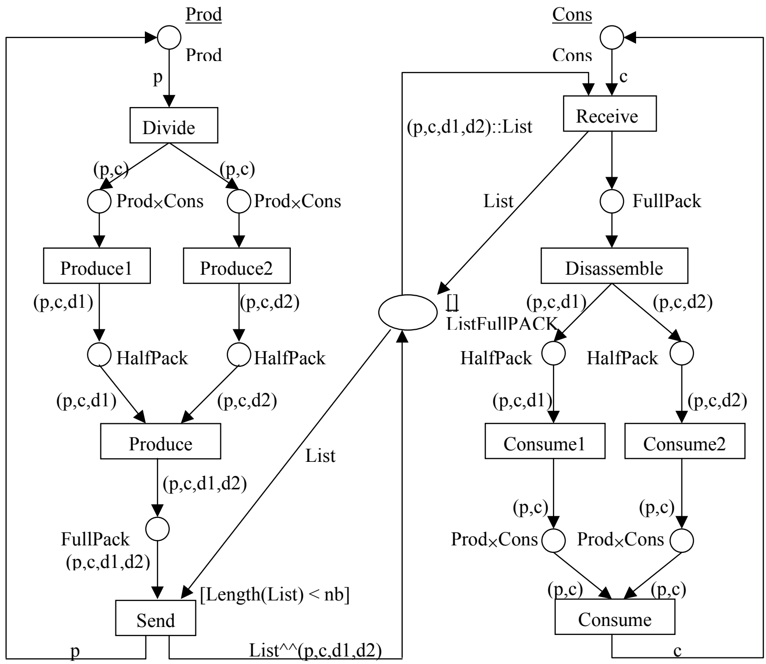


Fig. 4. Producer/Consumer system

The table below demonstrates the growth of the occurrence graph and the respective growth of the unfolding's size. In Chapter 8 we give the same numbers for the occurrence graph and unfolding with a consistent equivalence.

p	c	d	O-graph	Unfolding's Size
1	1	1	72	19
2	2	2	$9.03 * 10^{6}$	$1.4 * 10^{3}$
3	3	3	$1.58 * 10^{13}$	$3.3 * 10^{4}$
5	5	5	$4.5 * 10^{27}$	$1.95 * 10^{6}$
10	10	5	$1.32 * 10^{59}$	$3.1 * 10^{7}$
10	10	10	$7.8 * 10^{70}$	$5.0 * 10^{8}$
20	20	20	$1.73 * 10^{174}$	$1.28 * 10^{11}$
50	50	20	$2.11 * 10^{469}$	$5.0 * 10^{12}$

The table for the producer/consumer system (Fig.4)

7. DEADLOCK CHECKING USING NET UNFOLDING

In this part we describe a deadlock detection technique based on unfoldings of Petri nets. It's easy to see from Theorem 1 that we have a deadlock in a coloured Petri net if and only if we have the corresponding deadlock in its occurrence net which doesn't contain any cutoff point. The same can be told about the reachability property considering the occurrence net as an acyclic and 1-safe net system, where all places of Min(***N***) are initially marked. Since in this case the occurrence net is an acyclic and 1-safe net system, we obtain the results proposed in [12] for an ordinary n-safe net to be true also for CPN.

McMillan in [11] has also proposed the technique of deadlock checking. In this paper this technique will not be considered. The comparative study of these two methods can be found in [12].

In an ordinary PN, if the marking M is reachable from the initial marking M_0 by firing of the sequence σ of transitions, then we can write the following equation: $M(p) = M_0(p) + \sum_{(t\in \bullet p)}\nu(\sigma,t)F(t,p) - \sum_{(t\in p\bullet)}\nu(\sigma,t)F(p,t)$, where the number of occurrences of a transition t in σ is denoted by $\nu(\sigma,t)$. This can be written in the matrix form: $M = M_0 + \mathbf{N}\boldsymbol{\sigma}$, where $\boldsymbol{\sigma}=(\nu(\sigma,t_1)\ ...\nu(\sigma,t_m))$ is called the Parikh vector of σ, and **N** denotes the incidence matrix P×T given by N(p,t) = F(p,t) - F(t,p). The following system is called a marking equation.

Variables: X: vector of integer

$M = M_0 + \mathbf{N}X$

$X \geq 0$

Proposition 7.1. ***(*** [12] ***):*** Let ***N*** be an acyclic net system and let M be a marking. M is reachable from the initial marking M_0 if and only if the marking equation has a nonnegative solution.

Proposition 7.2. ***(*** [12] ***):*** Let ***N*** be a 1-safe and acyclic net system. A vector M is a solution of the following system of inequalities if and only if M corresponds to a dead reachable marking of ***N***:

> Variables M,X: integer;
> $M = M0 + N \cdot X$
> $\sum (p \in {}^{\bullet}t)\, M(p) \leq |{}^{\bullet}t| - 1$ for all $t \in T$
> $X \geq 0$

Theorem 2. Let $\boldsymbol{N_1}$ = $(S_1,P_1,T_1,A_1,N_1,C_1,G_1, E_1,I_1)$ be a CPN and Unf($\boldsymbol{N_1}$) = ($\boldsymbol{N_2}$, h, φ, η), where $\boldsymbol{N_2}$=(P_2,T_2,N_2), be its GT_0 (GT, EQ) - unfolding. $\boldsymbol{N_1}$ is deadlock-free if and only if the following system of inequalities has no solution:

> Variables M,X : vector of integer;
> $M = Min(\boldsymbol{N_2}) + N_2 \cdot X$
> $\sum (p \in {}^{\bullet}t)\, M(p) \leq |{}^{\bullet}t| - 1$ for all $t \in T_2$
> $X(t) = 0$ for all $t \in$ Cutoffs
> $X \geq 0$

Proof: If M is a deadlock then, accordingly to Theorem 1(3), there exists a configuration C such that Mark(C) = M and C contains no cutoffs. From Theorem 1(2) we have $C \subseteq C' \Rightarrow Mark(C') \in [Mark(C)\rangle$. Therefore there is no C' such that $C \subseteq C'$ and Cut(C) is a deadlock in the occurrence net. Cut(C) is a reachable marking in OPN. So, we have that existence of a deadlock in $\boldsymbol{N_1}$ implies existence of a deadlock in $\boldsymbol{N_2}$. If Cut(C) is a deadlock in OPN and C contains no cutoffs, then Mark(C), being a reachable marking, is a deadlock in $\boldsymbol{N_1}$. Otherwise if $Mark(C) \rightarrow^{(t,b)} M_1$ then, using the maximality of the considered branching process, we have that $\exists (t,b) \in T_2 \mid {}^{\bullet}(t,b) \subseteq Cut(C)$ (there are no cutoffs in C) and $C \subset C \cup \{(t,b)\}$ and we come to a contradiction. The cutoff transitions are not the solutions of the inequalities.

Therefore, we can identify dead markings of N_1 with the solutions of the above system of inequalities. ■

The technique works for all three types of unfoldings because we make a deadlock decision using the marking and the next transitions.

8. UNFOLDINGS WITH SYMMETRY AND EQUIVALENCE

In this part the technique of equivalence and symmetry specifications for coloured Petri nets (CPN) will be applied to the unfolding nets of CPN. It will be shown how to generate the maximal branching process and its finite prefixes for a given CPN under the equivalence or symmetry specifications. All symmetry and equivalence specifications are taken from [9].

Definition 8.1. Let ***N*** be a CPN and **M** and BE be the sets of all markings and binding elements of ***N***. The pair $(\approx_M, \approx_{BE})$ is called an *equivalence specification* if $\approx_M$ is an equivalence on **M** and $\approx_{BE}$ is an equivalence on BE. $M_\approx$ and $BE_\approx$ are the *equivalence classes*. We say $(b,M) \approx (b^*,M^*)$ iff $b \approx_{BE} b^*$ and $M \approx_M M^*$. Let us have $X \subseteq \mathbf{M}$ and $Y \subseteq M_\approx$, then we can define:

$[X] = \{M \in \mathbf{M} \mid \exists x \in X : M \approx_M x\}$ - the set of all markings equivalent to the markings from X.

$[Y] = \{M \in \mathbf{M} \mid \exists y \in Y : M \in y\}$ - the set of all markings from the classes from Y.

Definition 8.2. The equivalence specification is called *consistent* if for all $M_1, M_2 \in [[M_0\rangle]$ we have $M_1 \approx_M M_2 \Rightarrow [Next(M_1)] = [Next(M_2)]$, where $Next(M_1) = \{(b,M) \in BE \times \mathbf{M} \mid M_1[b\rangle M\}$.

Definition 8.3. Let a CP-net and a consistent equivalence specification $(\approx_M, \approx_{BE})$ be given. The *occurrence graph with equivalence classes*, also called the *OE-graph*, is the directed graph OEG = (V, A, N) where:

(1) $V = \{C \in M_\approx \mid C \cap [M_0\rangle \neq \varnothing\}$.

(2) $A = \{(C_1,B,C_2) \in V \times BE_\approx \times V \mid \exists (M_1,b,M_2) \in C_1 \times B \times C_2 : M_1[b\rangle M_2\}$.

(3) $\forall a=(C_1,B,C_2) \in A: N(a) = (C_1,C_2)$.

Proposition 8.1.([9] **)** For a consistent equivalence specification, the OE-graph satisfies the following properties:

(1) Each finite occurrence sequence $M_1[b_1\rangle M_2[b_2\rangle M_3 \ldots M_n[b_n\rangle M_{n+1}$, where $M_1 \in [M_0\rangle$ and $b_i \in BE$ for $i \in 1..n$, has a *matching direct path*

$[M_1]\ ([M_1],[b_1],[M_2])\ [M_2]\ ([M_2],[b_2],[M_3])\ [M_3] \ldots$
$\ldots [M_n]\ ([M_n],[b_n],[M_{n+1}])\ [M_{n+1}]$.

(2) Each finite direct path
$C_1\ (C_1,B_1,C_2)\ C_2\ (C_2,B_2,C_3)\ C_3 \ldots C_n\ (C_n,B_n,C_{n+1})\ C_{n+1}$ has, for each marking $M1 \in C1$, a *matching occurrence sequence*

$M_1[b_1\rangle M_2[b_2\rangle M_3...M_n[b_n\rangle M_{n+1}$, where $M_i \in C_i$ for all $i \in 2..n+1$ and $b_i \in B_i$ for all $i \in 1..n$.

In the next definitions, the set of all markings is denoted by $\mathbf{M}$.

Definition 8.4. A *symmetry specification* for a CP-net is a set of functions $\Phi \subseteq [\mathbf{M} \cup BE \to \mathbf{M} \cup BE]$ such that:

(1) $(\Phi, \bullet)$ is an algebraic group.

(2) $\forall \phi \in \Phi$: $(\phi|\mathbf{M}) \in [\mathbf{M} \to \mathbf{M}]$ & $(\phi|BE) \in [BE \to BE]$.

Each element of Φ is called a *symmetry*.

Definition 8.5. A symmetry specification Φ is *consistent* iff the following properties are satisfied for all symmetries $\phi \in \Phi$, all markings $M_1, M_2 \in [M_0\rangle$ and all binding elements $b \in BE$:

(1) $\phi(M_0) = M_0$.

(2) $M_1[b\rangle M_2 \Leftrightarrow \phi(M_1)\,[\,\phi(b)\,\rangle\,\phi(M_2)$.

Proposition 8.2. ([9] **)**
(1) The relation $\approx_M \subseteq \mathbf{M} \times \mathbf{M}$ defined by $M \approx_M M^* \Leftrightarrow \exists \phi \in \Phi : M = \phi(M^*)$ is an equivalence relation on the set of all markings $\mathbf{M}$.
(2) The relation $\approx_{BE} \subseteq BE \times BE$ defined by $b \approx_{BE} b^* \Leftrightarrow \exists \phi \in \Phi : b = \phi(b^*)$ is an equivalence relation on the set of all binding elements BE.

Proposition 8.3. ([9] **)** Each consistent symmetry specification Φ determines a consistent equivalence specification ($\approx_M$, $\approx_{BE}$).

Now the cutoff criteria will be defined for a CPN with a symmetry specification Φ or equivalence specification $\approx$. We call the finite prefix of the maximal branching process of CPN obtained by using new cutoff criteria an *unfolding with symmetry* Unf^{Φ} or *unfolding with equivalence* $Unf^{\approx}$. Since accordingly to propositions 8.2 and 8.3 we can consider the symmetry specification as the case of equivalence specifications, we give the cutoff definitions only for equivalence specifications.

Note: Taking into consideration the consistency of the regarded equivalence, we can conclude that it is sufficient to consider the classes [M] in our definitions of cutoffs. The classes of binding elements will be obtained in a natural way.

Definition 8.6. Let ***N*** be a coloured Petri net and MBP(***N***) be its maximal branching process. Then

(1) a transition $t \in T$ of an OPN is a $GT^{\approx}_0$ *- cuttoff* if there exists $t_0 \in T$ such that $Mark([t]) \approx Mark([t_0])$ and $[\,t_0] \subset [t]$.

(2) a transition $t \in T$ of an OPN is a *$GT^{\approx}$- cutoff* if there exists $t_0 \in T$ such that $Mark([t]) \approx Mark([t_0])$ and $\|[t_0]\| < \|[t]\|$.
(3) a transition $t \in T$ of an OPN is a *$EQ^{\approx}$- cutoff* if there exists $t_0 \in T$ such that
(a) $Mark([t]) \approx Mark([t_0])$
(b) $\|[t_0]\| = \|[t]\|$
(c) $\neg(t \parallel t_0)$
(d) there are no EQ-cutoffs among t' such that $t' \parallel t_0$ and $\|[t']\| \le \|[t_0]\|$.

The notion $Unf^{\approx}$ is used for any type of unfoldings.

Proposition 8.4. $EQ^{\approx}$-unfolding $\le$ $GT^{\approx}$-unfolding $\le$ $GT_0^{\approx}$- unfolding.

Proof: We can apply the ideas of proposition 4.2 changing the symbols "=" into "≈".

Theorem 3. Let ***N*** be a CPN and $\approx = (\approx_M, \approx_{BE})$ be a consistent equivalence on ***N***. Then for an $Unf^{\approx}(N)$ we have:
(1) $[M] \in [[M_0]\rangle \Leftrightarrow \exists C$, a configuration of $Unf^{\approx}(N) \mid Mark(C) \approx_M M$.
(2) $C \subseteq C'$ and C' is a configuration of $Unf^{\approx}(N) \Leftrightarrow [Mark(C')] \in [[Mark(C)]\rangle$

Proof:

(1) $[M] \in [M_0\rangle \Leftrightarrow$ we have a sequence

$$[M_0]([M_0],[b_1],[M_1])[M_1]\ ([M_1],[b_2],[M_2])[M_2]\ \ldots\ [M_{n-1}]([M_{n-1}],[b_n],[M_n])[M_n],$$

where $[M_n] = [M]$**.** From proposition 8.1(2), $\exists\ M_i', b_i'$, $i=1..n$, such that $M_0[b_1'\rangle M_1'[b_2'\rangle M_2' \ldots M_{n-1}'[b_n'\rangle M_{n+1}'$, and $\forall i=1..n\ M_i' \approx_M M_i$ and $b_i' \approx_{BE} b_i$.
Let us consider the configuration $C' = \{b_1' \ldots b_n'\} \mid Mark(C') = M_n' \approx_M M_n$.
Due to proposition 2.5 we only need to consider $EQ^{\approx}$-unfolding.
We have 3 possibilities:

(a) C' contains no cutoffs (in particular, $C = \emptyset$). $C' \in Conf(Unf^{\approx}(\boldsymbol{N}))$ and $Mark(C') \approx_M M$.

(b,c) C' contains a (GT- or EQ-) cutoff.

Let us choose the set of minimal configurations $\{C_j \mid j=1..k\}$, such that $\forall j\ Mark(C_j) \approx Mark(C')$. Consider the situation when none of them belongs to the $EQ^{\approx}$-unfolding. We can apply here the considerations of Theorem 1(3) after changing the symbols "=" into "≈" and applying the transitivity of "≈" relation.

Thus we obtain the configuration C'' such that Mark(C'') ≈ M and C'' is a configuration of EQ$^{\approx}$-unfolding(***N***).

(2) From the safety of MBP, C $\subseteq$ C' $\Rightarrow$ Mark(C')$\in$[Mark(C)$\rangle$, i.e., $M_1[b_1\rangle M_2 ... M_n[b_n\rangle M_{n+1}$, where M_1 = Mark(C) and M_{n+1} = Mark(C').

On applying the proposition 8.1(1), we get [Mark(C')]$\in$[[Mark(C)]$\rangle$. ■

Figures 5,6 show us the dining philosophers CPN and its unfolding with the symmetry specification.

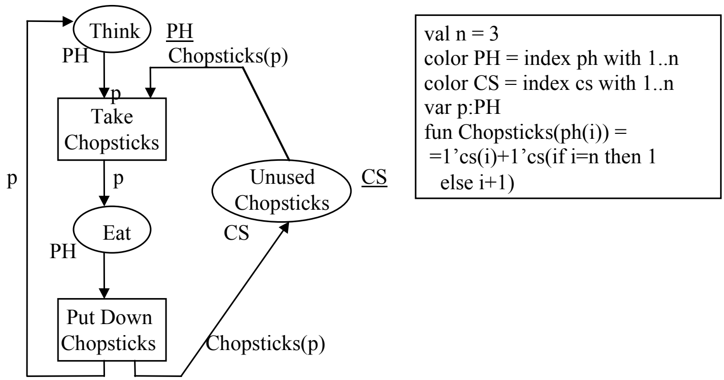


Fig. 5 The dining Philosophers

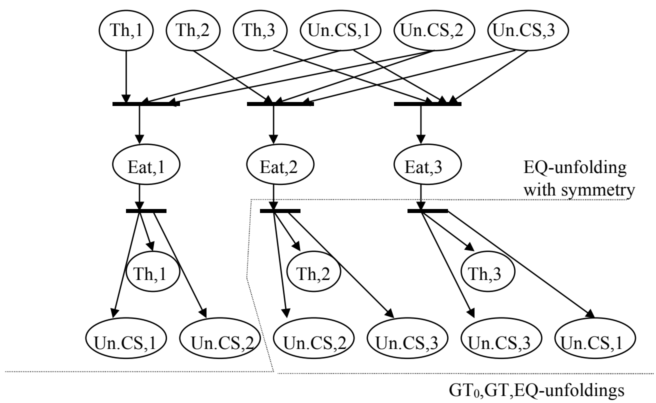


Fig. 6 The unfolding with symmetry

As is seen from the picture, using the symmetry we obtain a much smaller occurrence net when constructing the $EQ^{\approx}$-Unfolding.

To avoid an impression gained from this example that given an equivalence specification it is inefficient to use unfoldings (while the size of the state space of the respective OE-graph (O-graph with equivalence) is just two states), we consider the following example. The next table compares these two possibilities by considering the producer/consumer example (see chapter 6 or [9]). As an equivalence specification, the abstraction from the data d_1 and d_2 is considered. For a graph this means that we can put nd=1. In the case of unfolding we obtain the additional (d-1) transitions in the part PA. The whole size of EQ-unfolding with this equivalence is $UnfSize^{\approx}$ = |PA|*np*nc + |CA|*np*nc + |PB-1|*np*nc + |PC+1|*$(np*nc)^2$+(d-1).

The table below represents the results.

p	C	d	O-graph	Consistent OE-graph	Unfolding's Size	EQ-unfolding with equivalence
1	1	1	72	72	19	19
2	2	2	$9.03 * 10^{6}$	$7.32 * 10^{4}$	$1.4 * 10^{3}$	137
3	3	3	$1.58 * 10^{13}$	$1.68 * 10^{8}$	$3.3 * 10^{4}$	533
5	5	5	$4.5 * 10^{27}$	$3.67 * 10^{15}$	$1.95 * 10^{6}$	$3.5 * 10^{3}$
10	10	5	$1.32 * 10^{59}$	$1.43 * 10^{36}$	$3.1 * 10^{7}$	$5.14 * 10^{4}$
10	10	10	$7.8 * 10^{70}$	$1.43 * 10^{36}$	$5.0 * 10^{8}$	$5.14 * 10^{4}$
20	20	20	$1.73 * 10^{174}$	$5.97 * 10^{82}$	$1.28 * 10^{11}$	$8.0 * 10^{5}$
50	50	20	$2.11 * 10^{469}$	$2.32 * 10^{243}$	$5.0 * 10^{12}$	$3.1 * 10^{7}$

Important note: Let us notice that we should generate EQ-unfolding when using the symmetry specification. In the case of equivalence specifications in general we can use all types of unfoldings.

The next proposition gives us the possibility to find a deadlock of ***N*** considering its unfolding with an equivalence.

Proposition ([9] **).** Let M be a marking of a CPN, then (M is dead) $\Leftrightarrow$ ([M] is terminal i.e. $\neg\, \exists$ M' | [M']$\in$[[M] $\rangle$).

It follows from the proposition that we can apply the technique from chapter 7 to unfoldings with equivalence.

Unfortunately we can't say (M is reachable) $\Leftrightarrow$ ([M] is reachable). Using an unfolding with equivalence, we may declare only the reachability of markings represented in $Unf^{\approx}$(N).

CONCLUSION

In this paper the unfolding technique proposed by McMillan in [11] and developed in later works is applied to coloured Petri nets as they are described in [8, 9]. The technique is formally transferred, two algorithms and three finitization criteria are considered. We require a CPN to be finite, n-safe and to contain only finite sets of colours.

The unfolding is a finite prefix of the maximal branching process. To trancate the occurrence net, we consider three cutoff criteria in the paper. To construct the finite prefix, two algorithms of unfolding generation are formally transferred from the ordinary PN's area.

One of the novelties of the paper is application of the unfolding technique to CP-nets with symmetry and equivalence specifications as they are represented in [9].

The size of unfolding is often much smaller than the size of the reachability graph of a PN. Using the criteria, such as EQ-cutoff criteron, and symmetry or equivalence specifications in the unfolding generation, we can (as it is shown in the last chapter) additionally reduce the size of unfolding.

In the future it is planed to construct finite unfoldings of Timed CPN as they are described in [8, 9], using the technique of unfolding with equivalence, and also to make all the necessary experiments with unfoldings of coloured Petri nets.

Acknowledgments. I would like to thank Dr. Valery Nepomniaschy for drawing my attention to this problem and Dr. Elena Bozhenkova for valuable remarks.

REFERENCES

[1] **Bieber B., Fleischhack H.** Model Checking of Time Petri Nets Based on Partial Order Semantics // Proc. CONCUR'99. — Berlin a.o.: Springer-Verlag, 1999. — P. 210—225. — (Lect. Notes Comput. Sci.; Vol. 1664).

[2] **Cheng A., Christensen S., Mortensen K. H.** Model Checking Coloured Petri Nets Exploiting Strongly Connected Components // DAIMI PB – 519, March 1997.

[3] **Couvreur J.-M., Grivet S., Poitrenaud D.** Designing an LTL Model-Checker Based on Unfolding Graphs // Lect. Notes Comput. Sci. — 2000. — Vol. 1825. — P. 123—145.

[4] **Engelfriet J.** Branching Processes of Petri Nets // Acta Informatica. — 1991. — Vol. 28. — P. 575—591.

[5] **Esparsa J.** Model-Checking Using Net Unfoldings // Lect. Notes Comput. Sci. — 1993. — Vol. 668. — P. 613—628.

[6] **Esparsa J., Römer S., Volger W.** An Improvement of McMillan's Unfolding Algorithm // Proc. TACAS'96. — Berlin a.o.: Springer-Verlag, 1997. — P. 87—106. — (Lect. Notes Comput. Sci.; Vol. 1055).

[7] **Esparza J., Heljanko K.** A New Unfolding Approach to LTL Model-Checking // Lect. Notes Comput. Sci. — 2000. — Vol. 1853. — P. 475—486.

[8] **Jensen K.** Coloured Petri Nets. Vol. 1. — Berlin a.o.: Springer, 1995.

[9] **Jensen K.** Coloured Petri Nets. Vol. 2. — Berlin a.o.: Springer, 1995.

[10] **Kondratyev A., Kishinevsky M., Taubin A., Ten S.** A Structural Approach for the Analysis of Petri Nets by Reduced Unfoldings // 17th Intern. Conf. on Application and Theory of Petri Nets, Osaka, June 1996. — Berlin a.o.: Springer-Verlag, 1996 — P. 346—365. — (Lect. Notes Comput. Sci.; Vol.1091.).

[11] **McMillan K.L.** Using Unfolding to Avoid the State Explosion Problem in the Verification of Asynchronous Circuits // Lect. Notes Comput. Sci. — 1992. — Vol.663. — P. 164—174.

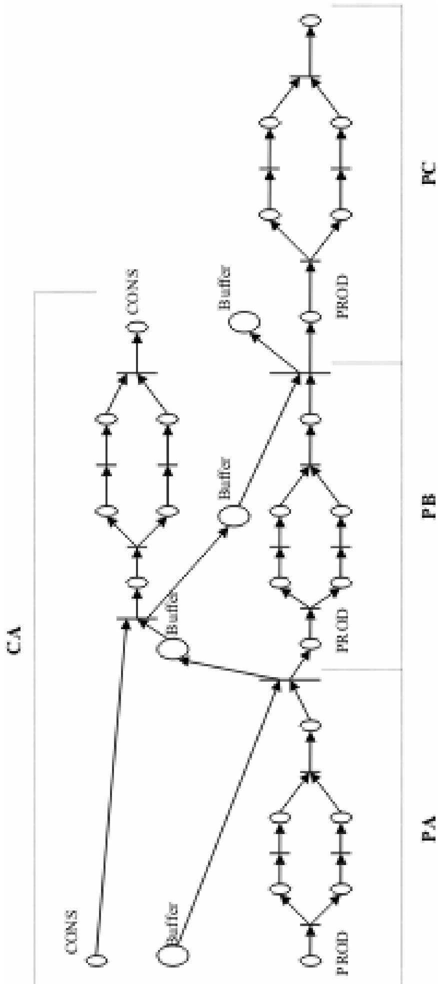
[12] **Melzer S., Römer S.** Deadlock Checking Using Net Unfoldings // Proc. of the Conf. on Computer Aided Verification (CAV'97), Haifa, 1997. —Berlin a.o.: Springer-Verlag, 1997. — P. 352—363. — (Lect. Notes Comput. Sci.; Vol.1254).

[13] **Valmari A.** The State Explosion Problem // Lect. Notes Comput. Sci. — 1998. — Vol.1491. — P. 429—528.

[14] **Valmari A.** Stubborn Sets of Coloured Petri Nets // Proc. of the 12th Intern. Conf. on Application and Theory of Petri Nets. — Gjern, 1991. — P. 102—121.

[15] **Wallner F.** Model-Checking LTL Using Net Unfoldings // Proc. of the Conf. on Computer Aided Verification (CAV'95), Vancouver, 1995. —Berlin a.o.: Springer-Verlag, 1998. — P. 207—218. — (Lect. Notes Comput. Sci.; Vol. 1427).

Unfolding of Producer-Consumer net for np=nc=mb=nd=1



В.Е. Козюра

РАЗВЕРТКИ РАСКРАШЕННЫХ СЕТЕЙ ПЕТРИ

Препринт
80

Рукопись поступила в редакцию 23.10.00
Рецензент Е. Н. Боженкова
Редактор А. А. Шелухина

Подписано в печать 8.12.00
Формат бумаги 60 × 84 1/16 Объем 1.9 уч.-изд.л., 2.1 п.л.
Тираж 50 экз.

НФ ООО ИПО “Эмари” РИЦ, 630090, г. Новосибирск, пр. Акад. Лаврентьева, 6