

Colored Petri Net inversion for Backward Reachability Analysis

Mohamed BOUALI, Pavol BARGER, Walter SCHON

*Heudiasyc Laboratory, CNRS UMR 6599
Université de Technologie de Compiègne, France
email: {mohamed.bouali,pavol.barger,walter.schon}@hds.utc.fr*

Abstract: Colored Petri Nets (CPN) are a powerful, recognized and intuitive modelling tool. They allow a precise representation of distributed, embedded and/or real time systems. These models can be then used among others for the dependability assessment. This paper develops a new method of CPN analysis called the *Backward reachability*. It provides information about different ways of reaching a particular CPN marking that represents, for example, a failure state or a transient fault. This analysis is performed on an *inverse CPN* which is obtained by transforming original CPN structure while preserving the model properties. The work develops mathematical tools to prove the pertinence of transformations allowing the definition of inverse CPN. The main advantage of this method is that it allows to determine the sequence leading from the initial to the final marking for any possible final marking vector.

Keywords: Colored Petri Net, Backward Reachability, formal methods, dependability, safety.

1. INTRODUCTION

System dependability is an important research issue especially if addressed to critical domain. Systems are verified through tools that check compliance of their properties with design specifications. Formal methods provide an interesting way to study and develop verification tools thanks to their precise characterization of the modelled system. It is in this context that the presented work applies a formal method approach to systems modelling and analysis.

Petri Nets noted PN (Petri (1962)), and particularly, Colored Petri Nets noted CPN (Jensen and Rozenberg (1991)), are a powerful and recognized modelling tool. They are endowed with a big expressiveness and allow to represent the two aspects of a system : static thanks to the PN structure and dynamic thanks to the token distribution evolution. The PN analysis can be done in several manners: exhaustive reachable state space enumeration, simulation, structural analysis, etc. These methods allow to study request/action effects on the model behavior. The exhaustive reachable state space enumeration gives correct results but suffers from the combinatorial explosion. The Monte Carlo simulation is a robust method but although the results are statistically correct, they are bound with an estimable error which can be decreased with an augmentation of simulation repetitions. To avoid drawbacks of these methods, we propose to use the structural analysis using directly the model itself.

Usually, the performed analysis is the forward one. That means, by knowing an initial state, possible final states are calculated. This is particularly adapted to the studies of performances and the quality of service (QoS) of systems. But, it is not adapted for studies with rare final state. In such analysis, all possible initial configurations must be

studied to find those leading to the considered failure state. This is why, in the case of ordinary Petri Nets, Khalfaoui (2003) interested in the reachability between two markings studied under two dual manners: the forward reachability and the backward reachability. In the backward reachability, the idea is to build, from the present state, predecessor states until the reach of a state derived from the initial state. Khalfaoui (2003) defined inverse Petri Net. Thus, this method is defined and applicable for ordinary PNs but not for CPNs.

Cho et al. (1996) propose a method, based on Leveson and Stolzy (1987), for safety analysis using backward reachability of CPN. It separates token values from transition firing conditions. Possible tokens values are obtained by applying a variation of the CPN fundamental evolution equation allowing the backward evolution of the marking. Firing conditions are obtained by performing logical unification of the possible values with arc expressions. This process is not always possible that is why the method introduced the *don't care condition*. It allows to split places of the CPN into two distinct sets. The first set contains places whose initial marking is known. The second contains places whose initial marking is not known and allows a certain liberty about unknown token values. This mechanism is efficient but it truncates some conditions of the backward firing transition and so hides some firing information.

This work proposes an approach based on the backward reachability analysis applied to CPNs. To do this, the method based on inverse PNs is generalized and *inverse CPNs* are defined. The construction of inverse CPN proceeds by transformations done on the structure of the original CPN. It allows to show up all conditions of the backward firing transition. In this paper, the generalization process is presented and transformations are formally proved.

The paper is organized as follows: Section 2 gives the theoretical preliminaries required for transformations proof. Section 3 shows the color mapping in CPNs and their dual notation. Section 4 summarizes the construction of the inverse PN. The main theoretical contribution of this paper is presented in section 5 with the development of elementary inversion rules for CPN. Section 6 gives a comprehensive application example. The paper ends by the conclusion and outlines future perspectives.

2. DEFINITIONS AND PRELIMINARIES

2.1 Petri Net and Colored Petri Net

A Petri Net (Petri (1962)), called also Ordinary Petri Net or Place/Transition Net, is a directed bipartite graph defined by the 4-tuple $(P, T, Pre, Post)$, where: P is a finite set of places, T is a finite set of transitions ($P \cap T = \emptyset$), Pre is the backward incidence application, $Post$ is the forward incidence application.

The notation of Colored Petri Net noted CPN (Jensen and Rozenberg (1991)) introduces the notion of token types, namely tokens are differentiated by colors, which may be arbitrary data values. Each place has an associated type determining the kind of data that the place may contain. A non-hierarchical CPN is defined by the 9-tuple $(\Sigma, P, T, A, N, C, G, E, I)$ where :

- Σ is a finite set on non-empty types,
- P is a finite set of *places*,
- T is a finite set of *transitions*,
- A is a finite set of *arcs* such that: $P \cap T = P \cap A = T \cap A = \emptyset$,
- N is a node function. It is defined from A into $P \times T \cup T \times P$,
- C is a *color* function. It is defined from $P \cup T$ into Σ ,
- G is a *guard* function. It is defined from T into expressions such that:
 $\forall t \in T : [Type(G(t)) = B \wedge Type(Var(G(t))) \subseteq \Sigma]$,
- E is an *arc expression* function. It is defined from A into expressions such that:
 $\forall a \in A : [Type(E(a)) = C(p(a))_{MS} \wedge Type(Var(E(a))) \subseteq \Sigma]$
 Where $p(a)$ is the place of $N(a)$,
- I is an *initialization* function. It is defined from P into closed expressions such that:
 $\forall p \in P : [Type(I(p)) = C(p)_{MS}]$

For practical reasons, we write E in a split form Pre and $Post$ (as used in Haddad (1989)) such that Pre (resp. $Post$) is the backward (resp. forward) incidence application. It is defined as E where $p(a)$ is the place of a part of $N(a)$ defined from A to $P \times T$ (resp. $T \times P$).

2.2 Bilinear form

A *form* is an application of a vector space in his number corp K (the number corp is a number set defining external vectors multiplication such as integers, reals or complexes). A *Bilinear form* is an application defined on a couple of vectors x and y , from the Cartesian product of $E \times F$ that have the same number corp. For a given application f , we write:

$$f : E \times F \rightarrow K \\ (x, y) \rightarrow f(x, y)$$

A form is said linear for its first variable if for each y_0 , the application f which, to x , associates $f(x, y_0)$ is linear. In the same way, the form is linear for its second variable if for each x_0 , the application f which, to y , associates $f(x_0, y)$ is linear. If the two previous proprieties are satisfied, the form is bilinear.

2.3 Similitudes

Let consider E be a space provided with the bilinear form $\phi : E \times E \rightarrow K$. Let $f : E \rightarrow E$ be a linear application. We define the symmetric bilinear form ϕ_f by

$$\phi_f : E \times E \rightarrow K \\ (x, y) \rightarrow \phi(fx, fy)$$

We say that f is a *similitude* of ϕ multiplier μ (noted also $\mu(f)$) if the following propriety is verified:

$$\forall x \in E, \forall y \in E, \phi(fx, fy) = \phi_f(x, y) = \mu\phi(x, y)$$

A result produced by this definition is that orthogonal applications are similitudes (whose multiplier is 1).

2.4 Equivalence between linear and bilinear forms

A color function can be defined either from $Bag(C(t))$ ¹ to $Bag(C(p))$ or from $C(t) \times C(p)$ to \mathbb{N} . The two forms are useful to define CPN transformations (developed later in this work). It is why the same symbol is used to the two forms. The formula that gives relation between the two forms is expressed as follows:

$$f(c) = \sum_{c' \in C(p)} f(c', c) \cdot c'$$

Where $f(c)$ denotes the mapping of c to an item of $Bag(C(p))$ by f as a linear application and where $f(c', c)$ denotes the mapping of (c', c) to an integer value. We note that no confusion can appear since the first definition implies one argument while the second definition implies two arguments.

2.5 Multi sets properties

- The Identity function of $Bag(C)$ (noted Id) is defined by $Id(c) = c$. This function can also be defined by $Id(c', c) = (\text{If } (c = c')1 \text{ else } 0)$
- A function f from $Bag(C)$ to $Bag(C')$ is *orthonormal* if and only if there exists a substitution σ of C such that $f(c) = \sigma(c)$. The second definition is: $f(c', c) = (\text{If } (\sigma(c) = c')1 \text{ else } 0)$. We can also write this condition as a similitude form Evangelista et al. (2005): $\forall c \in C, \exists c' \in C', f(c, c') = 1$ and $\forall c' \in C', \exists c \in C, f(c, c') = 1$
- The projection from $Bag(C \times C')$ to $Bag(C)$ (noted $Proj$) is defined by $Proj(\langle c, d \rangle) = c$. The second definition is: $Proj(c', \langle c, d \rangle) = (\text{If } (c = c')1 \text{ else } 0)$.
- A function f from $Bag(C)$ to $Bag(C')$ is quasi-injective if and only if $\forall c' \in C', \forall c_1 \in C, \forall c_2 \in C : f(c', c_1) \neq 0 \wedge f(c', c_2) \neq 0 \Rightarrow c_1 = c_2$
- A function f from $Bag(C)$ to $Bag(C')$ is unitary if and only if $\forall c' \in C', \forall c \in C : f(c', c) = 0 \vee f(c', c) = 1$

¹ We note $Bag(U)$ the multi set defined over U (U is a finite set). A *Bag* (or *multi set*) is a non ordered set where the repetition is permitted. An element of $Bag(U)$ is noted $\sum_{u \in U} a_u \cdot u$. Using *Bag* notion allow treatment of color sets case which are characterized by the repetition of their values.

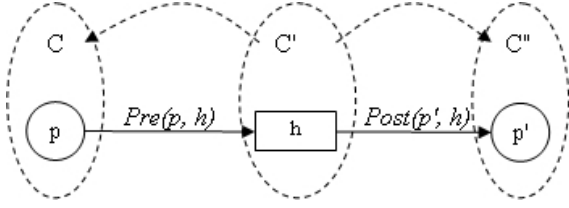


Fig. 1. Color correspondance in CPN

2.6 Composition

Let f be a function from $Bag(C)$ to $Bag(C')$ and g a function from $Bag(C')$ to $Bag(C'')$. The composition of f and g is a function $g \circ f$ from $Bag(C)$ to $Bag(C'')$ defined by :

$$g \circ f(c) = g(f(c)) = \sum_{c'' \in C''} \left(\sum_{c' \in C'} g(c'', c') \cdot f(c', c) \right) \cdot c''$$

2.7 Orthonormalization of a transition

Let R be a CPN where $R = (\Sigma, P, T, A, N, C, G, E, I)$, t be a transition of R and f an orthonormal function of $C(t)$. The CPN $R_r = (\Sigma_r, P_r, T_r, A_r, N_r, C_r, G_r, E_r, I_r)$ obtained by f - orthonormalization of t is defined by:

- $P_r = P, T_r = T$
- $\forall t \in T_r, \forall p \in P_r, C_r(t) = C(t)$ AND $C_r(p) = C(p)$
- $\forall t' \in T_r - \{t\}, \forall p \in P_r, Post_r(p, t') = Post(p, t')$ AND $Pre_r(p, t') = Pre(p, t')$
- $\forall p \in P, Pre_r(p, t) = Pre(p, t) \circ f$ AND $Post_r(p, t) = Post(p, t) \circ f$
- $\forall p \in P_r, M_r(p) = M(p)$

3. COLOR CORRESPONDANCE IN CPN

Let us consider the case illustrated in Fig.1. The CPN is constituted by a set of two places $\{p, p'\}$ and a set of one transition $\{h\}$ such that the precondition of h is p and the post condition of h is p' . The orthonormal function $Pre(p, h)$ is defined from $Bag(C(h))$ to $Bag(C(p))$. The function $Post(p', h)$ is defined from $Bag(C(h))$ to $Bag(C(p'))$. So, we have: $Pre(p, h) : C' \rightarrow C$ and $Post(p', h) : C' \rightarrow C''$. The goal is to express relation between C and C'' .

By its definition, the function $Pre(p, h)$ is orthonormal, i.e. there exists a substitution σ of C such that $f(c) = \sigma(c)$. Using this substitution, we define the inverse substitution σ^{-1} . This definition is possible thanks to a part of the orthonormality condition which is $\forall c' \in C', \exists c \in C, f(c, c') = 1$. This inverse substitution is associated to a new color function noted Pre^{-1} defined from $Bag(C(p))$ to $Bag(C(h))$. Expressions of $Pre^{-1}(p, h)$ and $Post(p, h)$ can be expressed as follows:

$$\begin{cases} Pre^{-1}(p, h)(c) = \sum_{c' \in C(h)} Pre^{-1}(p, h)(c', c) \cdot c' & \text{for } c \in C(p) \dots (1) \\ Post(p', h)(c') = \sum_{c'' \in C(p')} Post(p', h)(c'', c') \cdot c'' & \text{for } c' \in C(h) \dots (2) \end{cases}$$

Replacing c' expressed in (1) by (2) gives :

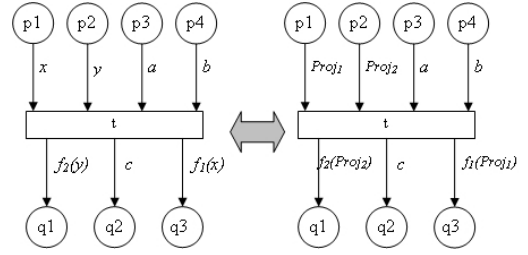


Fig. 2. From practical notation to formal notation

$$\begin{aligned} & \sum_{c' \in C(h)} Pre^{-1}(p, h)(c', c) \cdot \sum_{c'' \in C(p')} Post(p', h)(c'', c') \cdot c'' \\ &= \sum_{c'' \in C(p')} c'' \cdot Post(p', h)(c'', c') \cdot \sum_{c' \in C(h)} Pre^{-1}(p, h)(c', c) \\ &= \sum_{c'' \in C(p')} c'' \cdot \sum_{c' \in C(h)} Post(p', h)(c'', c') \cdot Pre^{-1}(p, h)(c', c) \\ &= Post(p', h) \circ Pre^{-1}(p, h)(c'', c) \end{aligned}$$

Note that the relation between C and C'' is expressed as $Post(p', h) \circ Pre^{-1}(p, h)$ which is defined from C to C'' . This result can be obtained by orthonormalization of the transition h with the orthonormal function $Pre^{-1}(p, h)$. This operation composes $Pre(p, h)$ with $Pre^{-1}(p, h)$ and composes also $Post(p', h)$ with $Pre^{-1}(p, h)$. Note that $Pre(p, h) \circ Pre^{-1}(p, h)$ equals to identity (Id), which denotes the result.

Since the two CPNs are equivalent, we define a relation (noted \diamond) such that $f \diamond g = g \circ f^{-1}$. This operator allows to express that f is a precondition and g is a postcondition and it is useful to handle easily symbolic operations applied to incidence matrices of the CPN.

3.1 Dual CPN Notation

In this study, we exploit two different notations of CPN. The first one is very useful to model real systems. It is implemented in industrial softwares like the CPN-tools² (Ratzer et al. (2003)). It marks input arcs (of some transition) by variables and output arcs by functions. This notation is equivalent to the second one (Haddad (1989)) which is useful to perform formal proofs. In the second notation, a transition takes its color values in a set defined by a cartesian product $C = C_1 \times \dots \times C_n$ where each C_i corresponds to colors original domain of arc expressions. So $Pre_i = Proj_i(C)$ noted $Proj_i$ (or *constant*). $Proj_i$ projects transition color values to its i^{th} precondition place. Variables of output function take their values in C . Fig.2 illustrates a concrete example. $C = C(t) = C_1 \times C_2 \times C_3 \times C_4$. The function f_1 takes its values in C_1 such that $C_1 = Proj_1(C)$ (that can be noted $Proj_1$). In the same manner, f_2 takes its values in C_2 which is a projection of C on its second item.

4. ORDINARY PN INVERSION

Consider an ordinary PN $R = (P, T, Pre, Post)$. Inverse PN R^{-1} is defined by $R^{-1} = (P', T', Pre', Post')$ where:

- $P' = P, T' = T,$

² <http://wiki.daimi.au.dk/cpntools>

- Pre' and $Post'$ are defined as $\forall p \in P, \forall t \in T$

$$\begin{cases} Pre'(p, t) = Post(p, t) \\ Post'(p, t) = Pre(p, t) \end{cases}$$

In an informal way, a place/transition PN inversion is done by direction inversion of arcs composing the PN which is equivalent to replace Pre by $Post$ and vice versa.

To study formally this aspect, it is necessary to work on incidence matrices (Pre and $Post$). To express forward relation between places (one step), we multiply $Pre_{|P| \times |T|}$ ($|P|$ lines and $|T|$ columns) by $Post^t_{|T| \times |P|}$ ($Post$ transposed having $|T|$ lines and $|P|$ columns). To express backward relation between places, we transpose the matrix $Pre.Post^T$. In an other side, we have the relation $(Pre.Post^T)^T = (Post^T)^T.Pre^T = Post.Pre^T$. Note that to express the inverse relation in the PN the same formula is used with a permutation between Pre and $Post$. This proves the coherence of the inverse PN construction.

5. COLORED PETRI NET INVERSION

The inversion applied to ordinary PN can be generalized to CPN in two steps: 1) inversion of arcs direction and 2) CPN functions transformation. The application of only the first step, i.e. generalize the inversion method as announced for ordinary PN, may lead to the construction of a CPN whose precondition expressions are neither orthonormal, nor unitary, nor quasi-injective. This is why it is necessary to check and to transform (second step).

5.1 Trivial transformation

In trivial case, input arc is marked by identity function (Id) and the output arc by a function f . In this case, it is necessary to know whether f is orthonormal. If yes, it is necessary to define its inverse f^{-1} . Applying the two steps mentioned above leads to mark input arc by the identity function Id and the output arc by f^{-1} . This gives:

$$\begin{cases} Pre'(p2, t) = Pre^{-1}(p1, t) \\ Post'(p1, t) = Post^{-1}(p2, t) \end{cases}$$

The proof is a generalization of the demonstration applied in Ordinary PN. The forward relation between places (one step) is given by $Pre.Post^T$ and the backward relation (one step) is given by the transpose of this last matrix which is $Post.Pre^T$. Let's check these relations in the Trivial transformation case.

We have :

$$Pre = \begin{pmatrix} Id \\ 0 \end{pmatrix}, Post = \begin{pmatrix} 0 \\ f \end{pmatrix}, Pre.Post^T = \begin{pmatrix} 0 & Id \diamond f \\ 0 & 0 \end{pmatrix}$$

So, the relation between $P1$ and $P2$ is verified by the expression $Id \diamond f = f \circ Id^{-1} = f$. In addition, we have backward relation $Post.Pre^T = \begin{pmatrix} 0 & 0 \\ f \diamond Id & 0 \end{pmatrix}$ which verifies a relation from $P2$ to $P1$ through the term $f \diamond Id$. As $f \diamond Id = Id \circ f^{-1}$, the algorithm provides the inverse CPN.

5.2 Mixed transformations

Fig.2 shows a mixed case where some input arcs are marked with variables $\{x, y\}$ and other arcs by constants

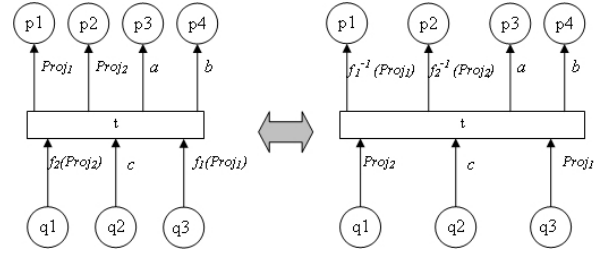


Fig. 3. Mixed transformations for CPN inversion

$\{a, b\}$. Output arcs are marked with constants $\{c\}$ and reversible functions $\{f_1, f_2\}$. This CPN inversion is a mix (generalization) of basic transformations. For arcs marked with a constant, it is sufficient to generalize the rule applied on PNs (changing the direction of arcs). For the remaining arcs, we have to apply the rule used in the trivial case. So, we have to associate each variable to its function in order to respect origin places of arcs marked by variables and sink places which receive resulting tokens. The following constraint has to be verified: each variable is used by one and only one function. The algorithm describing the mixed transformation can be written as follows :

- $Pre'(qj, t) = Post(qj, t)$, $j = 1 \dots m$ if $Post(qj, t)$ is marked with a constant,
- $Post'(pi, t) = Pre(pi, t)$, $i = 1 \dots n$ if $Pre(pi, t)$ is marked with a constant,
- $Pre'(qj, t) = Pre^{-1}(pi, t)$, $i = 1 \dots n$, $j = 1 \dots m$ if $Pre(pi, t)$ is marked with a variable and this variable is associated to the function $Post(qj, t)$,
- $Post'(pi, t) = Post^{-1}(qj, t)$, $i = 1 \dots n$, $j = 1 \dots m$ if $Post(qj, t)$ is marked with a function and this function is associated to the variable $Pre(pi, t)$.

Proof: Fig.3 illustrates the two steps allowing to inverse the CPN. The first one is equivalent to what was applied in trivial case, means, inversion of arcs direction. It remains to prove the transformation to the interpreted form of the CPN.

Let's note by C colors domain of the transition such as $C = C_1 \times C_2 \times \dots \times C_n$. We define $f_i : C_i \rightarrow C'$, and a projection $Proj_i : C \rightarrow C_i$. The composition of the two functions is defined by $f_i(Proj_i) : C \rightarrow C'$.

Let's note by \tilde{C}_i The colors domain defined by $\tilde{C}_i = C_1 \times \dots \times C_{i-1} \times C' \times C_{i+1} \times \dots \times C_n$. We define $\tilde{f}_i : \tilde{C}_i \rightarrow C'$ with $\tilde{f}_i(x) = (proj_1(x), \dots, Proj_{i-1}(x), f_i(Proj_i(x)), Proj_{i+1}(x), \dots, Proj_n(x))$. We define also a projection $Proj_i : \tilde{C}_i \rightarrow C$. The composition of the two functions is defined by $Proj_i(\tilde{f}_i) : C \rightarrow C'$.

Note that $f(Proj_i)$ and $Proj_i(\tilde{f}_i)$ gives the same result.

$$\begin{aligned} & Proj_i(\tilde{f}_i)(x) \\ &= Proj_i(proj_1(x), \dots, Proj_{i-1}(x), f_i(Proj_i(x)), \\ & Proj_{i+1}(x), \dots, Proj_n(x)) \\ &= f(Proj_i)(x) \end{aligned}$$

Finally, we define the function \tilde{f}_i^{-1} such as

$$\tilde{f}_i^{-1}(x) = (proj_1(x), \dots, Proj_{i-1}(x), f_i^{-1}(Proj_i(x)), Proj_{i+1}(x), \dots, Proj_n(x))$$

The transformation is done thanks to orthonormalization sequence using \tilde{f}_i^{-1} . For each function \tilde{f}_i^{-1} , three different parts are identified :

- (1) The first one is the part of the transition composed by an input arc marked with the function $f_i \circ Proj_i$ and an output arc marked by $Proj_i$. The composition with the function \tilde{f}_i^{-1} gives the following results: In input arc side we have

$$f_i \circ Proj_i \circ \tilde{f}_i^{-1} = f_i \circ f_i^{-1} \circ Proj_i = Proj_i$$
 In output arc side, we have $Proj_i \circ \tilde{f}_i^{-1} = f_i \circ Proj_i$
 We conclude that the composition produces a part of transition which can be evaluated (functions indexed by i).
- (2) The second one is the part of the transition composed by an input arc marked with the function $f_j \circ Proj_j$ (such as $j \neq i$) and an output arc marked with the function $Proj_j$. The composition with the function \tilde{f}_i^{-1} gives the following results. At input arc side, we have $f_j \circ Proj_j \circ \tilde{f}_i^{-1} = f_j \circ Proj_j$. At output arc side, we have $Proj_j \circ \tilde{f}_i^{-1} = Proj_j$. So, this composition does not affect other functions that those indexed by i .
- (3) The last one is the remaining part of the transition, means, arcs marked with constants. Knowing that composition does not affect constants, this still true for the composition with \tilde{f}_i^{-1} .

To complete the transition transformation, it is enough to repeat precedent steps for all functions marking input arcs.

5.3 Parallel transformations

The term *parallelism* means here the existence of a shared variable, means the same variable is used by more than one function. To inverse such a transition, we have to calculate the inverse of only one function using the shared variable (that supposes existence of, at least, one reversible function). Let be f a reversible function. In the original CPN, the firing of transition t produces one tokens in each post place of t . Each token value results from a function applied to the shared variable. The associated inverse CPN must produce a token in the pre place of t by firing it. But it is not enough to have tokens in post places to fire t . That means, token in post places must have coherent values toward applied functions. For this reason, we define a *guard* associated to transition t . It checks coherence of the values in post places. If the guard value is *True*, t will be fired, then the initial marking will be found. If the guard value is *False*, t will not be fired.

The algorithm describing the parallelism transformation can be written as follow:
 Let R be a CPN containing a variable x ($Pre(p1, t) = x$) which is shared by n functions f_1, f_2, \dots, f_n ($Post(qi, t) = f_i(x), i = 1 \dots n$) with f_1 reversible. The inverse CPN is defined by :

- $Pre'(q1, t) = Pre^{-1}(p, t)$,
- $Pre'(qi, t) = Id_i, i = 2 \dots n$,
- $Post'(p, t) = Post^{-1}(q1, t)$,
- $Guard(t) = \bigwedge_{i=2}^n [Pre'(qi, t) == f_i(f_1^{-1}(Pre'(q1, t)))]$.

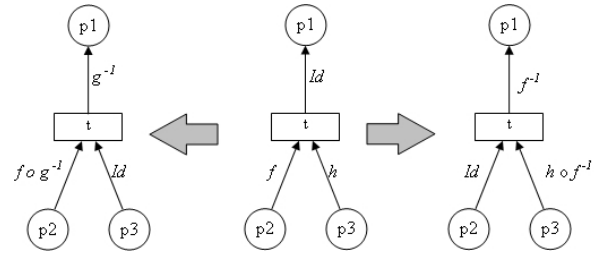


Fig. 4. Proof of Mixed transformations for CPN inversion

Proof: As in previous inversions, the proof on parallel transformation is composed of two steps. The first is the inversion of arcs directions. So, it remains to prove the transformation to interpreted form on the inverse CPN. This proof can be performed in the case of two parallel functions, and then, generalized for any number of parallel functions.

The transformation, as illustrated in Fig.4 can be done by an orthonormalization of the transition. The choice of the orthonormal function is arbitrary (f or g in Fig.4). This gives two possible inverse CPN which are equivalent. The scheme Fig.4 left is obtained by composition with g^{-1} and Fig.4 right is obtained by composition with f^{-1} .

The two inverse CPN are equivalent means that functions f^{-1} and g^{-1} are applicable to a subset of transition color domain $c \in Bag(C(t))$ where $f^{-1}(c) = g^{-1}(c)$. For this colors subset, $f \circ g^{-1} = Id$ and $g \circ f^{-1} = Id$. By substitutions in arc expressions, illustrated in Fig.4 (left and right), identical results are obtained. The inverse CPN obtained is endowed with a restrictive condition about colors domains. This condition is called *guard* and it is expressed with the notation related to the transition.

6. CASE STUDY

This section shows a practical use of the backward reachability (through an inverse CPN). The application example is inspired from "Simple Protocol"³. It describes a communication protocol with an identified acknowledgment.

The basic question studied here is whether the receiver could possibly receive a hypothetical sequence of frames.

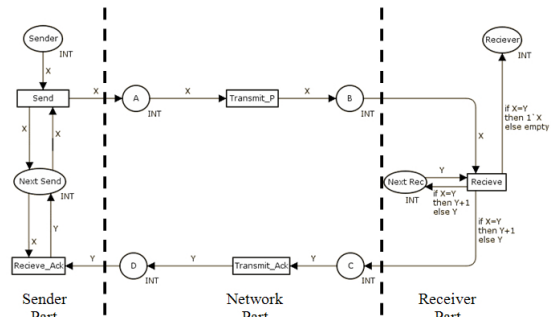


Fig. 5. CPN model of the communication protocol

The normal functioning of this system would not allow doublets (the correct reception of two identical frames). Thus for example the marking with 3 following packets in

³ <http://wiki.daimi.au.dk>

place *Receiver* where the first is identified as 1, two others frames equal to each other and identified as 2 represent and unreachable state. In the following this marking will be noted as $\langle 1 + 2 + 2 \rangle$.*Send*.

To verify the reachability of the previously given marking ($\langle 1 + 2 + 2 \rangle$.*Send*), the backward reachability analysis is performed using the inverse CPN, illustrated in Fig.6. Inversions of *Transmit_P* and *Transmit_Ack* are basic transformations. Those of *Receive_Ack* is a case of parametric transformation (The input variable X is not used in an output function). The goal of this transition is to delete the old value in *Next_Send* and to replace it by the token value of the place D . So, in the inverse reasoning, the goal is to put in D the token value of *Next_Send* and to put in *Next_Send* an arbitrary value. The token already contained in this place is then returned. The inversion of transition *Send* is a parallel one. This can be seen through the input variable X which is used in two output functions (toward A and *Next_Send*). This explains the new variable Z and the guard $[X = Z]$. The inversion of *Receive* is a combination of parallel and parametric (the function are parameterized by X and Y). In the inverse problem, tokens in *Receive* are known which is equivalent to consider that possible initial values of X are known.

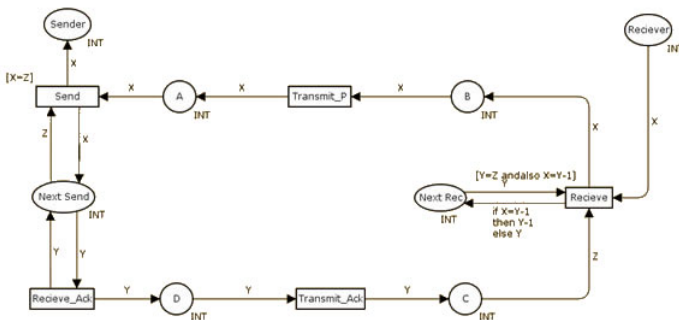


Fig. 6. Inverse CPN of the communication protocol

In the original CPN, the sequence of transition firing is : *Send*, *Transmit_P*, *Receive*, *Transmit_Ack*, *Receive_Ack*. This corresponds to a partial order of transition firings. The inverse CPN analysis (endowed by the marking $\langle 1 + 2 + 2 \rangle$.*Send*) must respect the inversed order starting with transition *Receive* which is the unique potentially valid transition. The *receive* firing is done using marking enhancement. Two tokens have to be added to the model : the first one into place *Next_Rec* and the second into place C . The two tokens are numbered 3 because this value is the only one that fulfills guard constraints. This marking enhancement corresponds to the assumption that the packet numbered 2 was correctly received. The result is a token of value 2 in places *Next_Rec* and B after firing *Transmit_P* in A . The firing of *Send* requires a token valued 2 in *Next_Send* which assumes a correct packet transmission. The marking evolution (without any enhancement) processes the packet numbered 1 thanks to all transitions firings (one time for each one). This leads to a deadlock situation with a packet numbered 2 in place *Receiver*.

Thus the conclusion is reached and expresses that there is no coherent initial marking possible to obtain the $\langle 1 + 2 +$

$2 \rangle$.*Send* marking in place *Receiver*. Upon the detection of such marking, an unexpected fault has to be declared.

7. CONCLUSION

The present paper introduces backward reachability applied to Colored Petri Nets by using inverse CPN. As this is a new approach of direct high level model based analysis, the theoretical developments are introduced. They are completed with an analysis of a practical example.

The main application domain of this work is the reachability analysis of undesired state. This can be a fundamental question for diagnostic, safety and security fields. The cases concern the situations when the initial and final states are both perfectly known. But the final state represents a rare state which is difficult to find with classical forward analysis. Reaching a limit value is an example of such a final state.

In the actual progress phase, only a limited number of inversion patterns are developed. This is why the application at the moment can only be done on small academic examples. The development of rules required for realistic examples is the main perspective. As the inclusion of all possible cases will not be possible, the studied domain will have to be delimited and clearly stated. The method is then intended to be applied in the corresponding studies in our industrial research projects concerned with real-time system safety demonstrations.

REFERENCES

- Cho, S.M., Hing, H.S., and Cha, S.D. (1996). Safety analysis using colored petri nets. In *Proc. Asia-Pacific Software Engineering Conference (APSEC-96)*, 4-7 December 1996, Seoul, Korea, 176-183.
- Evangelista, S., Haddad, S., and Pradat-Peyre, J.F. (2005). Syntactical colored petri nets reductions. *Automated Technology for Verification and Analysis (ATVA 05) Taipei, Taiwan, October 4-7*, 202-216.
- Haddad, S. (1989). A reduction theory for coloured nets. *Lecture notes in Computer science n 424 Springer-Verlag*, 209-235.
- Jensen, K. and Rozenberg, G. (1991). *High-level Petri Nets: Theory and Application*. Springer-Verlag, Germany.
- Khalfaoui, S. (2003). *Mthode de recherche des scenarios redouts pour l'valuation de la sret de fonctionnement des systmes mcatroniques du monde automobile*. Ph.D. thesis, Institut National polytechnique de Toulouse (France).
- Leveson, N.G. and Stolzy, J.L. (1987). Safety analysis using petri nets. *IEEE Trans. Softw. Eng.*, 13(3), 386-397.
- Petri, C.A. (1962). *Communication with automata*. Ph.D. thesis, Darmstadt Institut fr Instrumentelle Mathematik, Bonn (Germany).
- Ratzer, A., Wells, L., Lassen, H., Laursen, M., Frank, J., Stissing, M., Westergaard, M., Christensen, S., and Jensen, K. (2003). Cpn tools for editing, simulating, and analysing coloured petri nets. In *Applications and Theory of Petri Nets 2003: 24th International Conference, ICATPN 2003*, 450-462. Springer Verlag.