Backward reachability of Colored Petri Nets for systems diagnosis

Mohamed Bouali, Pavol Barger, Walter Schon

Backward Reachability of Colored Petri Nets for Systems Diagnosis

Mohamed BOUALI*, Pavol BARGER*, Walter SCHON*

Heudiasyc Laboratory, CNRS UMR 6599
Université de Technologie de Compiègne, France

Abstract

Embedded systems development creates a need of new design, verification and validation technics. Formal methods appear as a very interesting approach for embedded systems analysis, especially for dependability studies. The chosen formalism for this work is based on Colored Petri Net (CPN)for two main reasons : the expressivity and the formal nature. Also, they model easily the static and the dynamic natures of the studied systems. The main challenge of this work is to use existing models, which describe the system structure and/or behavior, to extract the dependability information in a most general case and failure diagnosis information in a particular case. The proposed approach is a CPN structural backward reachability analysis. It can be split in two parts. The first one is to perform the proposed analysis : inverse CPN. It is obtained thanks to structural transformations applied on the original CPN. The second part is the analysis implementation. This part needs some complementary concepts of which the most important is the marking enhancement. The proposed approach is studied under two complementary aspects : algorithmic and theoretic aspects. The first one proposes transformations for the CPN inversion and the analysis implementation. The second aspect (the theoretical one) aims to offer a formal proof for the approach by applying two methods which are linear algebra and linear logic.

Keywords: Colored Petri Nets (CPN), Backward Reachability, Structural Analysis, Dependability, Embedded Systems, diagnosis.

1. Introduction

In the last few years, the part of embedded systems in electronic and computer-based devices is ever growing. These omnipresent systems are used in most tasks and activities like communications, transportation or industrial processes. If we look at mobil phones, for example, we can see easily that their complexity increased highly in short period. What was, in the beginning, a heavy and cumbersome device becomes a light ergonomic tablet which includes a multimedia station, office applications, a networks terminal (WiFi, Bluetooth, 3G), a game console, etc. This amount of functionalities is necessary because of the harsh competition between manufacturers. This phenomenon produces two strong constraints on the embedded systems design process : 1) to market delays must be as short as possible ; and 2) the proposed product has to contain more functionalities than its rival. The product has also to respect strong financial and dependability conditions.

To take into account the increasing requirements on embedded systems, Methods and designing tools evolve in the same way as the system evolution. These tools have to deal with hardware increasing complexity, large size embedded software and reduced design delays. In addition, the designed system has to fulfill strict dependability requirements. Even for large consumption products, the dependability requirements are very rigorous. The example of a mobil phone is very illustrative : an entertainment application must not interfere with a voice call ; the later must respect a low electromagnetic emission level with a minimal energy consumption ; the battery, which must have a long autonomy period, must not overheat ; the whole system has to respect pollution norms. All these requirements are a very synthetic overview.

In this economic context, one of the research issues is the development of technics to take into account dependability requirements in embedded system design as the classical methods in this field (failure trees, reliability block diagrams, ...) reach quickly their limits when confronted to new requirements. Discrete event systems based modelling appears as an efficient way to reach this objective in the way that models describe system contents, components hierarchy, process events scheduling and they are easily analysable.

To deal with the present research issues, some works (see [1], [2], [3]) exploited the discrete event systems, and especially Petri nets, to perform some diagnosis and dependability analyses over different embedded systems types. The interest is not only to analyse the system. It is also to do the analysis in a formal framework thanks to mathematic tools like Linear Logic. We undertake, in this work, to extend and to generalize this approach. We choose Colored Petri Nets (CPN) based modelling. CPN are extension of Ordinary Petri Nets which integrates a tokens differentiation semantics. This choice is motivated by the large expressivity of CPN models, their mathematic basics allowing formal analysis, their compliance with embedded systems modelling (like parallel processing, communication and events). The CPN model can be easily analyzed by *forward*

*Corresponding author
Email addresses: mohamed.bouali@hds.utc.fr (Mohamed BOUALI), pavol.barger@hds.utc.fr (Pavol BARGER), walter.schon@hds.utc.fr (Walter SCHON)

analysis, that means, by determining the causes we find consequences. The use of CPN to model the embedded systems gives some interesting issues. First of all, the CPN large expressivity, and especially arc expressions, make the model analysis more complicated than the case of Ordinary Petri Nets. Also, the failure diagnosis is difficult because, in diagnosis, consequences are known (a failure state is indicated using an alarm, a red light, a profil deviation, ...) and we aim to know causes. As in embedded systems, failures are very rare events, the forward analysis can't be applied by simulation.

This work studies a dual vision to the forward analysis called *backward analysis*. It is more adapted to diagnosis and aims to offer a design aided tool for dependable embedded systems. The basic idea is to transform (invert) the model to drive the analysis following consequences (usually which are failures). The inversion is performed thanks to structural transformations. The analysis is then driven on inversed models using complementary mechanisms such as the marking enhancement generalized to CPN. The algorithmic aspects are strengthened by two theoretic studies: the first one for the local aspect (inversion algorithms), the second one for global aspect (backward reachability analysis).

This paper is outlined as follows: after this introduction, the section 2 summarizes the related works in the Petri Nets Inversion Field. The section 3 presents a formal definition of a Colored Petri Net and then details the algorithms to perform the CPN inversion. The two following sections deal with theoretical aspects. The section 4 treats proofs using linear algebra formalism. The section 5 treats proofs using Linear logic formalism. The presented approach is illustrated in the case study of the section 6. Finally, the paper ends with a conclusion and some propositions to future works.

2. Related works

This paper deals with the backward reachability analysis based on an inverse CPN. The main objective is to perform a diagnostic analysis using a Petri Nets based modelling without the reachability graph with an exclusive use of the structure of the model itself. The PN model (and its extensions) inversion is based on different technics according to the objective and the extension. For ordinary Petri Nets, works directed by H. Demmou and R. Valette ([3], [4], [5]) propose the use of an inverse Petri Net model. These ones are obtained by inverting arc directions in the original Petri Nets. The benefit of this method are its theoretical and implemental simplicity. It is used to deriving feared scenarios (which might lead the system to critical situation). But this method suffers from limitations when applying it to Colored Petri Nets. Portinale, in [6], proposes the *B-W Analysis*. It is a backward reachability through *Behavioral Petri Net* (BPN) models which introduce a (reduced) semantic of tokens differentiation. Nevertheless, the proposed extension is very particular and endowed with a limited expressivity. Concerning the CPN class, Cho et al. [7] proposes a method inspired from [8]. It is based on a modified version of the CPN fundamental equation. It allows to express a backward transition firing. It allows also to consider the transition firing even if some tokens are missing. The problem of this method is the association of each such transition firing with a predicate which can speed down analysis and even make it impossible because of undecidability introduced for each transition. In a very close context to CPN, Muller and Schnieder [1] propose an approach using *High Level Petri Nets* (HLPN). It generalizes, to HLPN, the dualization formula of [9] initially applied for Ordinary PN models. It transforms the model by changing places by transitions, transitions by places and inverting arc directions. Physical interpretation of modelled phenomena becomes very hard and the practical application of this method shows that the main benefit of the dualization is the limitation of some calculating intervals but the analysis itself is still done on the original HLPN models.

3. Inverse Colored Petri Net

The Backward Reachability in CPNs is a dual concept to the Forward Reachability. That is, if a marking M_f is reachable from M_0, we say that M_0 is backward reachable from M_f. By backward reachability we mean that M_0 is the cause or the source of M_f. Our contribution concerns firstly the generalization of the backward reachability from the PN class to the CPN class and secondly the development of adapted transformation rules. The generalization implies a definition of an inverse CPN which is obtained by applying the proposed transformations on the original CPN.

3.1. Definition of Petri Net and Colored Petri Net

A Petri Net [10], called also an Ordinary Petri Net or a 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 [11] 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 [12]) 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. $T \times P$).

3.2. CPN transformation for the definition of an inverse CPN

Let $R = (\Sigma, P, T, A, N, C, G, E, I)$ be a CPN and let $R' = (\Sigma', P', T', A', N', C', G', E', I')$ be the inverse CPN of R. To define R', we perform some transformations on R. These transformations are directly dependent on the CPN structure. Consequently, we have to define a transformation for each structure case studied. Nevertheless, the most common transformations in representative cases are presented in this section.

The following transformations (trivial, basic, parallel, parametric) are structure changes applied to CPN and result in inverse CPN. Their usefulness is to allow performing the backward reachability analysis. Each transformation is a generic algorithm applied on transitions type having same characteristics. These characteristics are

- ⋆ the nature of arc expressions (constant, variable, function),
- ⋆ the number of functions using the same variable (none, one, two or more),
- ⋆ arc orientation (input, output).

The combination of these three items leads to 81 possible cases. This number decrease quickly because many associations are obsolete like input functions, output variables [1] and functions without input variables. A second criterium reduces the possible number of associations : the number of constants and number of input arcs marked by the same variable have not the effect on the transformation type to apply but their presence/absence influences. At the end, among 81 initial possible cases, only 5 are relevant : input constant and output constant (trivial transformation), input variable and exactly one output function (basic transformation), input variable and no output function or more than two output functions (parametric transformations) and finally, input variable and more than one output function using it (parallel transformation). Combinations of different transformations types are possible. An example will be given to illustrate this case called mixed transformation.

[1] A variable associated to output arc is considered as an identity function

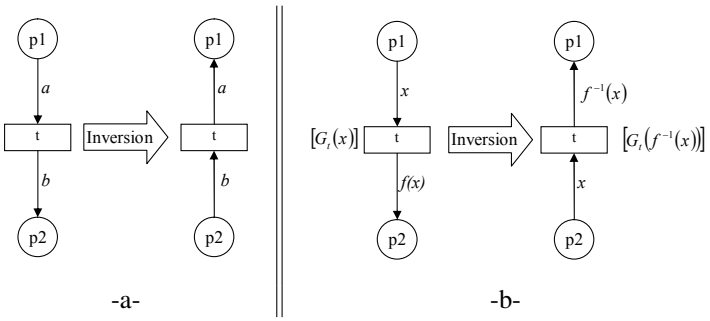


Figure 1: Trivial and basic transformations for CPN inversion

3.2.1. Trivial and basic transformations

Figure 1-a- shows a trivial case of a CPN inversion. Arcs (input and output) are marked with constants a, b. In this case, it is enough to generalize the definition of the inverse PN to CPN obtaining :

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

Note that, in the original CPN, $M_0 = \langle a \rangle.p1$ gives $M_1 = \langle b \rangle.p2$ by firing t. In the inverse CPN $M_1 = \langle b \rangle.p2$ gives $M_0 = \langle a \rangle.p1$ by firing t. Thus, we performed the backward reachability using the inverse CPN.

Figure 1-b- shows the case where the input arc is marked with a variable x, the output arc is marked by a function $f(x)$ and a guard $G(x)$ can be associated to the transition t. Generalized application of the rule shown before (inverting arcs directions) can lead to the construction of an incorrect net. In this example, the input arc would be marked with a function while the output arc would be marked with the variable of this function. That leads to the impossibility for the function evaluation. This is why we suggest marking the input arc by the variable, the output arc by the function f^{-1} and update the guard to express new constraint associated to the transition. This assumes the necessity to know whether the function f is reversible. If yes, it is necessary to define its inverse noted f^{-1}. This transformation gives :

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

$Pre^{-1}(p1,t)$ denotes the transformation of the arc $Pre(p1,t)$ which is marked with a new variable defined on f^{-1} domain. $Post^{-1}(p2,t)$ denotes the transformation of the arc $Post(p2,t)$ which is marked by f^{-1} (the opposite reverse of $Post(p2,t)$ marking function).

3.2.2. Parametric transformations

Some CPN structures can't be transformed to get deterministic markings in the backward reachability. The reason is that the inversion process could be assimilated to mathematic operations whose solutions may be intervals. The CPN inversion, in these cases, is *parametric*. That means that some additional

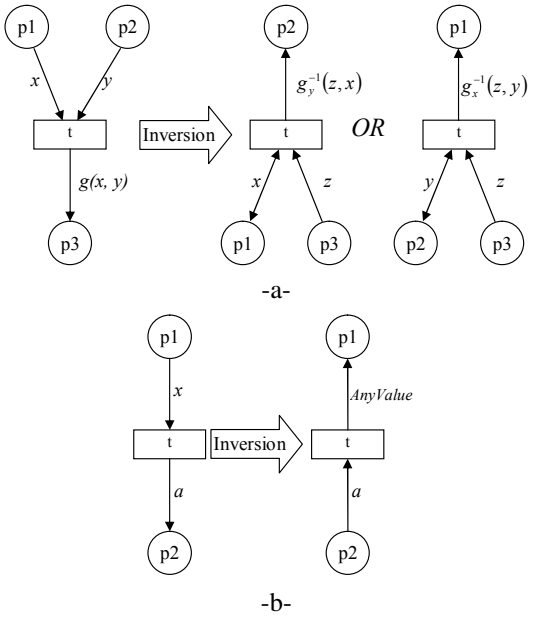


Figure 2: Parametric transformations for CPN inversion

information, like color sets, are needed. Two cases follow : the first treats a solution of multivariable equations, the second treats with input variables which are associated with no output functions.

Figure 2-a- shows a case where output arc is marked with a multivariable function. In this case, the CPN is not directly reversible. The values of variables $\{x, y\}$ can not be deduced by knowing only the value of $g(x, y)$. The *partial (parametric)* solution proposed consists in finding the admissible values of one variable knowing other variable values and the function result. As illustrated in Figure 2-a-, the firing t (in inverse CPN) requires tokens either in $\{p2,p3\}$ or in $\{p1,p3\}$. Functions $g_x^{-1}(y)$ and $g_y^{-1}(x)$ are *partial inverses* of $g(x, y)$.

Figure 2-b- shows a case of a variable which is not associated with a function. We note that input arc expression is a variable and output arc (or arcs) expression is constant. This CPN inversion is similar to the case of Figure 1-a-, except the arc $Post'(p1, t)$ which is marked by a *parametric* expression noted $AnyValue$. It means that this arc expression can take all values out of t in the set of the place $p1$ color.

3.2.3. Parallel transformations

The term *'parallel'* means the existence of a shared variable. Figure 3 shows the case where the same variable is used by more than one function (two functions in this case). To inverse this CPN, we have to calculate the inverse of only one function using the shared variable (which supposes existence of, at least, one reversible function).

Let f be a reversible function. In the original CPN (shown on the left in Figure 3), the transition t firing produces two tokens (in $p2$ and $p3$). Each token value results from a function applied to the shared variable : function f to the token in $p2$ and function h to the token in $p3$. The inverse CPN must produce a token in $p1$ by firing t whose preconditions are $p2$ and $p3$. But it is not enough to have tokens in $p2$ and $p3$ to fire t because token values (in $p2$ and $p3$) must have coherent values towards applied functions f and h. For this reason, we define a *guard* associated to transition t. It checks that the value of $f^{-1}(x)$ applied to h gives as the result the value of y (token initially in $p3$). If the guard value is *True*, t will be fired, then the initial marking of $p1$ will be found. If the guard value is *False*, t will not be fired.

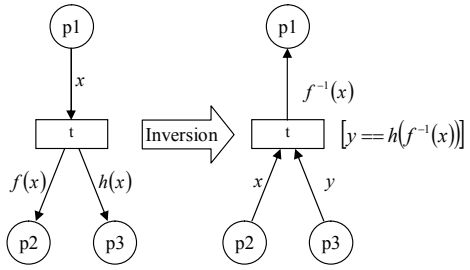


Figure 3: Parallel transformation for CPN inversion

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

- $Pre'(q1, t) = Pre^{-1}(p, t)$,
- $Pre'(qi, t) = Id_i, i = 2 \ldots n$,
- $Post'(p, t) = Post^{-1}(q1, t)$,
- $G'_t = \bigwedge_n^{i=2}[Pre'(qi, t) == f_i(f_1^{-1}(Pre'(q1, t)))]$.

3.2.4. Mixed/Complexe transformations

Transformations previously presented are *"simple"*, easily recognizable and transformable. However, in real models, transitions can gather some characteristics which don't allow them to be categorized in a unique transformation type (eg. parametric and parallel transition at the same time). The transformation of such transition is a mix of elementary transformations. To perform it, all transformation types to apply must be identified and applied in a pertinent order. The procedure often begins by parallel transformation, followed by the parametric, the basic and the trivial. This sequence is a general indication. But for each case, the transformation have to use the maximum available information and answers the expressed need. In real use, the dynamic available information influences the structural transformation choice to apply (practical examples can be found in the case study).

Figure 4 shows a mixed case where some input arcs are marked with variables $\{x, y\}$ and other arcs by constants $\{a, b\}$. Output arcs are marked with constants $\{c\}$ and reversible functions $\{f, g\}$. This CPN inversion is a mix (generalization) of a trivial and a basic transformations. For arcs marked with a constant, it is sufficient to generalize the rule applied on PNs (changing arcs direction). For the remaining arcs, we have to

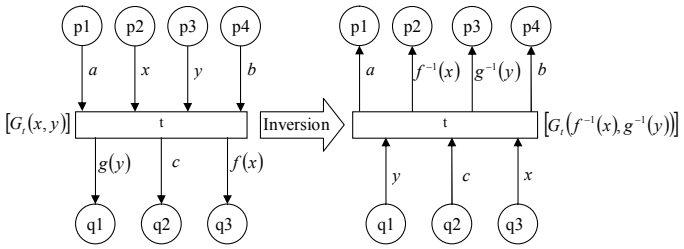


Figure 4: Mixed transformations for CPN inversion

apply the rule illustrated in Figure 1-b-. So, we have to associate each variable to its function in order to respect origine places of arcs marked by variables and sink places which receive resulting tokens. The following constraint has to be verified in this case: each variable is used by one and only one function. If not true, see the *Parametric transformation*.

3.3. Special transformations

Among the huge possible cases of generalized CPN structures, there exists some of them which transformation is difficult. The problem is directly related to the model itself. Whe following presents some of the most frequent cases and proposes solutions (tradeoffs) to perform the backward reachability analysis.

3.3.1. Loop transformations

The models aimed with this study show a recurrent presence loops. Characterization of these cycles can be very difficult in CPN because of the token values evolution. In the case of Ordinary Petri Nets, [13] reduces the number of feared paths returned by the backward reachability by detecting cycles and analyzing them only once. This method can not be generalized to CPN since two successive executions of the same structural cycle can produce tokens with very different values. An partial solution can be found in some cases by applying, for example, transitions agglomeration [12].

The minimal cycle is a loop between a place and a transition representing a very simple timer or sampler. To avoid a repetition of iterations, we propose the use of sequences, and especially, arithmetic and geometric sequences allowing to determine in only one operation the final value after a given number of iterations or, by duality, to calculate the number of necessary iterations to reach a value without having to simulate the loop. So, the main idea here is to consider that the concerned loops have a progression which can be compared to a sequence and according to the form of arc expression, token values generated by the loop execution can be considered as terms of sequences and can be deduced easily.

3.3.2. Constraint programming transformations

The model inversion often supposes the existence of, at least, one inversible function. This one gives source values for known results. In some cases, the requested function is inversible by pieces, inversible in a subset of the colors set, partially inversible or pseudo-inversible [14]. The problem is when the inversible function does not exist. One of possible solutions is the use of constraint programming [15]. This solution express the relation between inputs and outputs in a constraint form instead of a direct (function) form. The use of constraints implies the introduction of the *decidability* notion. In other words, if the constraints are resolved, the solution exists. In contrast, if constraints are not resolved, no response can be made about the existence on solution.

3.4. Supplementary mechanisms

The transformation rules presented above allow the inversion of the CPN structure. That is, they allow to obtain a structure on which the backward reachability can be performed. Sometimes, this analysis becomes impossible because the lack of information consisting on a partial knowledge of the system state. So, in certain situations, the final marking (or the intermediate marking) does not allow a complete analysis. This is why, in order to analyse the inverse dynamic behavior, the following complementary principles have to be introduced.

3.4.1. Potentially Valid Transition

In a marked CPN, a transition is potentially valid if it has, at least, 1.) one precondition place marked by a token whose value is compatible with the transition firing (i.e. token compatible with arc expression and guard) ; and 2.) one precondition place which is not marked by a token whose value is compatible with the transition firing. Identifying this kind of transitions allows the *marking enhancement*.

3.4.2. Marking Enhancement

The marking enhancement is a mechanism allowing to complete the information about a model by adding assumptions about the modelled system state (additional tokens). It is done when constructing successor states. Sometimes, the immediate successor of a given state (which is equivalent to fire a transition) can only be constructed with help of added tokens with appropriate values in certain places. The marking enhancement interpretation consists in assuming that the system is in some given state allowing it to operate. This mechanism has to be performed if only a part of the token distribution is available.

3.5. Partiel Order of transition firings

The partial order in CPN is a relation between transitions (not necessary all transitions of the model) noted by "$\prec$" [2]. This relation defines a precedence in transitions firing. For two transitions $t1$ and $t2$, the formula $t1 \prec t2$ means that the firing of $t1$ occurs before the firing of $t2$ (all properties of this relation are detailed in [2]). The notion of partial order is very important in the backward reachability analysis in order to decide which transition has to be fired first at each step of the analysis evolution. The partial order in the backward reachability is inverted comparing to forward analysis. For example, if in the original CPN, the relation $t1 \prec t2$ is true so, in the inverse CPN, the formula $t2 \prec t1$ is also true.

4. Linear algebra

The structural analysis with backward reachability uses the inverse CPN which is obtained by performing structural transformations on the original CPN. Question asked then are : 1) are transformation algorithms previously presented correct? and 2) is the backward analysis performed thanks to inverse CPN correct?

The answer to these questions needs the study of two theoretical aspects of CPN inversion. The first one, linear algebra, concerns each transition as standalone. The definitions are inspired from [12], [16], [17] and [18]. The methodology is inspired from [12] concerning CPN theory and precisely *Well Formed Petri Nets (WPN)*. The second theoretical aspect (linear logic) is detailed in section 5.

4.1. Definitions and preliminaries

4.1.1. Bilinear form

A *form* is an application of a vector space in his number set 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:

$$\begin{array}{rl} f: & E \times F \to K \\ & (x, y) \to f(x, y) \end{array}$$

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.

4.1.2. Similitudes

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

$$\begin{array}{rl} \phi_f: & E \times E \to K \\ & (x, y) \to \phi(fx, fy) \end{array}$$

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

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

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

4.1.3. Equivalence between linear and bilinear forms

A color function can be defined either from $Bag(C(t))$[2] 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 in the two forms. The formula that gives the relation between the two forms is expressed as follows:

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

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.

4.1.4. Multi sets properties

- The Identity function of $Bag(C)$ (noted Id) is defined as $Id(c) = c$. This function can also be defined as $Id(c', c) =$ (If $(c = c')1$ 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 equivalent definition is: $f(c', c) =$ (If $(\sigma(c) = c')1$ else 0). We can also write this condition as a similitude form [19]: $\forall c \in C, \exists c' \in C', f(c, c') = 1$ 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 equivalent definition is: $Proj(c', \langle c, d\rangle) =$ (If $(c = c')1$ 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 $\forall c' \in C', \forall c \in C : f(c', c) = 0 \vee f(c', c) = 1$

4.1.5. 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 :

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

4.1.6. 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 $Post_r(p, t) = Post(p, t) \circ f$
- $\forall p \in P_r, M_r(p) = M(p)$

[2] 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 $\sum_{u \in U} a_u.u$. Using Bag notion allow treatment of color sets case which are characterized by the repetition of their values.

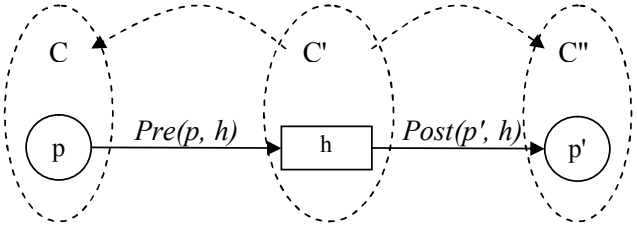


Figure 5: Color correspondance in CPN

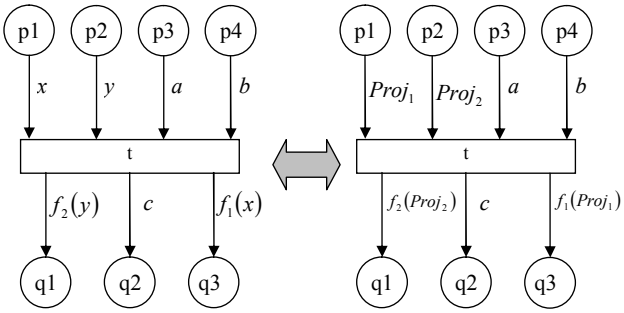


Figure 6: Equivalence between generalized PN and Well-Formed PN

4.2. Color correspondance in CPN

Let us consider the case illustrated in Figure 5. The CPN is constituted of a set of two places $\{p, p'\}$ and a set of only 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 $Pre(p',h) : C' \rightarrow C''$. 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:

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

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

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

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 also $Post(p',h)$ with $Pre^{-1}(p,h)$. Note that $Pre(p,h) \circ Pre^{-1}(p,h)$ equals to identity (Id), which explains 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 is useful for handling easily the symbolic operations applied to incidence matrices of the CPN.

4.2.1. Dual CPN Notation

In this study, we exploit two different notations of CPN : *generalized PN* and *Well-Formed PN*. The first one is very useful to model real systems. It is implemented in softwares like the CPN Tools[3] ([20]). It marks input arcs (of a given transition) by variables and output arcs by functions. This notation is equivalent to the alternative one ([12]) 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 $C = C_1 \times \cdots \times C_n$ where each C_i corresponds to original colors domain of an arc expression. 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. Figure 6 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.

4.3. CPN transformations proofs

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).

4.3.1. Arcs inversion

This proof is illustrated in the trivial case where 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 associate the input arc with the identity function Id and the output arc with f^{-1}.

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

[3] http://wiki.daimi.au.dk/cpntools

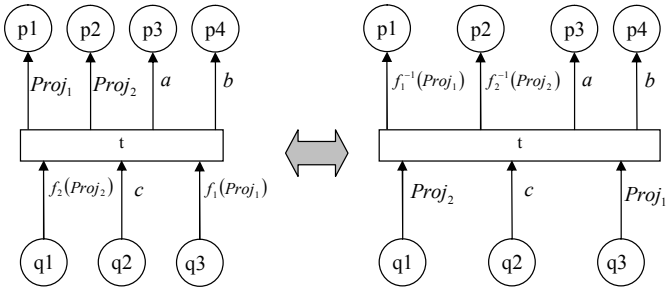


Figure 7: Mixed transformations viewed in generalized PN and Well-Formed PN

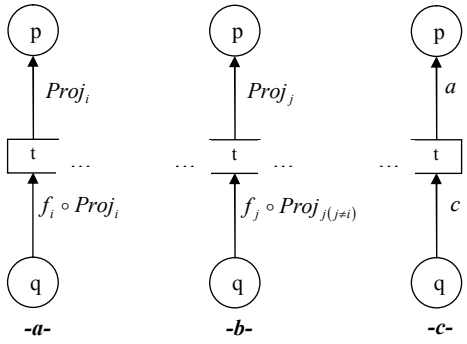


Figure 8: Mixed transformations proof for CPN inversion

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 forward relation between the places 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 cartesian product of places color sets through the term $f \diamond Id$. As $f \diamond Id = Id \circ f^{-1}$, the algorithm provides the inverse CPN.

4.3.2. Mixed transformations

Figure 6 shows a mixed case noted with the two different notations : generalized PN and Well-Formed PN. The proof of this case merges those of trivial and basic transformations.

Proof:

Figure 7 illustrates the two steps allowing to inverse the CPN. The first one is the 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 $C = C_1 \times C_2 \times \cdots \times C_n$. 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 $\tilde{C_i} = C_1 \times \ldots C_{i-1} \times C' \times C_{i+1} \times \cdots \times C_n$. We define $\tilde{f_i} : \tilde{C_i} \rightarrow C'$ with $\tilde{f_i}(x) = (proj_1(x), \ldots, Proj_{i-1}(x), f_i(Proj_i(x)), Proj_{i+1}(x), \ldots, Proj_n(x))$. 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.

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

Finally, we define the function $\tilde{f_i}^{-1}$ such as
$\tilde{f_i}^{-1}(x) = (proj_1(x), \ldots, Proj_{i-1}(x), f_i^{-1}(Proj_i(x)), Proj_{i+1}(x) \ldots, Proj_n(x))$

The transformation is done thanks to orthonormalization sequence using $\tilde{f_i}^{-1}$. For each function $\tilde{f_i}^{-1}$, three different cases are identified as illustrated in Figure 8.

Part a. Figure 8.a shows a 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

$$\begin{aligned} f_i \circ Proj_i \circ \tilde{f_i}^{-1} &= f_i \circ f_i^{-1} \circ Proj_1 \\ &= Proj_i \end{aligned}$$

In output arc side, we have $Proj_1 \circ \tilde{f_i}^{-1} = f_i \circ Proj_i$

We conclude that this composition produces a part of transition which can be evaluated (functions indexed by i).

Part b. Figure 8.b shows a 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.

Part c. Figure 8.c shows 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.

4.3.3. Parallel transformations

As in previous inversions, the proof of the parallel transformation is composed of two steps. The first one is the inversion of arc directions. So, it remains to prove the transformation on the interpreted form of 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 Figure 9 can be done by an orthonormalization of the transition. The choice of the orthonormal function is arbitrary (f or g in Figure 9). This gives two possible inverse CPNs which are equivalent. The scheme Figure 9 left is obtained by a composition with g^{-1} and Figure 9 right is obtained by a composition with f^{-1}.

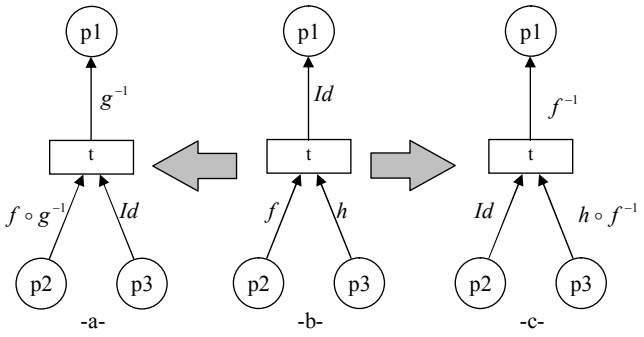


Figure 9: Proof of parallel transformations for CPN inversion

The two inverse CPNs are equivalent so that the functions f^{-1} and g^{-1} are applicable to a subset of the transition color domain $c \in Bag(C(t))$ where $f^{-1}(c) = g^{-1}(c)$. For this color subset, $f \circ g^{-1} = Id$ and $g \circ f^{-1} = Id$. By substitutions in arc expressions, illustrated in Figure 9 (left and right), identical results are obtained. The inverse CPN obtained is endowed with a restrictive condition about the color domains. This condition is called *guard* and it is expressed with the notation related to the transition.

5. Linear logic

The second theoretical aspect studied in this work aims to prove the correctness of results obtained by the backward reachability analysis using the whole (or a part) of the inverse CPN. The proofs are based on the Linear Logic (LL) formalism. Previous studies, like [3] and [21], used LL to prove reachability methods in ordinary PN but not in higher models. The challenge of this paper is to adapt and to use the LL formalism for CPN proofs.

5.1. Introduction to Linear Logic

Linear Logic was introduced by Girard [22]. Its expressive power is demonstrated by some very natural encodings of computational models such as Petri Nets, counter machines, Turing machines and others [23]. The Linear Logic differs from a classic logic by introducing the notion of a ressource. A classic logic deals with static propositions where each proposition is either true or false. Because of the static nature of propositions in a classic logic, there may be *duplicated* $[P \Rightarrow (P \wedge P)]$ and/or *discarded* $[(P \wedge Q) \Rightarrow P]$. Both of these propositions are valid in classical logic for any P and Q. In Linear Logic, these propositions are not valid because no information is available about how the second P is produced (in the first proposition) or where Q is consumed (in the second proposition). The rules of Linear Logic imply that linear propositions stand for dynamic properties or finite resources.

For example, let us consider the propositions E, C and T conceived as resources: 1)*E: one Euro*, 2)*C: cup of Coffee*, 3)*T: cup of Tea*. Consider the following axiomatization of a vending machine: 1)$E \Rightarrow C$, 2)$E \Rightarrow T$. Then in classical logic, the proposition $E \Rightarrow (C \wedge T)$ can be deduced. Which may be read as *"With one Euro, I may buy both a cup of coffee and a cup of tea"*. Although this deduction is valid in classical logic, it is a nonsense in the intended interpretation of propositions as resources: two cups of hot drinks cannot be bought with one Euro from the described vending machine.

In this work, we only use a fragment of the Linear Logic related to Petri Nets which is the MILL fragment (*Multiplicative Intuitionistic Linear Logic*). This fragment contains a multiplicative connector *TIMES* ($\otimes$) and a linear implication connector ($\multimap$). The *TIMES* connector traduces the accumulation of resources. The proposition $A \otimes A$ means that two resources A are available. The Linear implication ($\multimap$) expresses the causality between production and consumption of resources. The proposition $A \multimap B$ means that when the proposition A is consumed, the proposition B is produced. The meta-connector "," (comma) is cumulative [22].

5.2. Sequent calculus

The sequent calculus notation, due to Gentzen [24], is composed of two sequences of formulas separated by a turnstile symbol ($\vdash$). The formula $\Gamma, \Gamma' \vdash \Delta, \Delta'$ means that the conjonction of Γ and Γ' allows to deduce the disjonction Δ or Δ'. A sequent calculus proof rule consists of a set of hypothesis sequents, written above a horizontal line, and a single conclusion sequent, written below the line, as follow:

$$\frac{Hypothesis_1 \qquad Hypothesis_2}{Conclusion}\, Rule_{attribute}$$

The goal is to construct a proof tree. Starting from the sequent, and applying step by step some adapted rule, the proof consists on eliminating the connectors. These rules are shown in Figure 10 where : A is an atom ; F, G and H are formulas ; Γ and Δ are blocs of formulas separated by commas. The attribute indicates whether the rule is applied at left (L), at right (R) or to the whole sequent (empty attribute).

$$\frac{}{A \vdash A}\, id \qquad \frac{\Gamma \vdash F \qquad \Delta, F \vdash H}{\Gamma, \Delta \vdash H}\, Cut$$

$$\frac{\Gamma, F, G, \Delta \vdash H}{\Gamma, G, F, \Delta \vdash H}\, Exchange$$

$$\frac{\Gamma \vdash F \qquad \Delta, G \vdash H}{\Gamma, \Delta, F \multimap G \vdash H}\, \multimap_L \qquad \frac{\Gamma, \Delta, F \vdash G}{\Gamma, \Delta \vdash F \multimap G}\, \multimap_R$$

$$\frac{\Gamma, F, G \vdash H}{\Gamma, F \otimes G \vdash H}\, \otimes_L \qquad \frac{\Gamma \vdash F \qquad \Delta \vdash G}{\Gamma, \Delta \vdash F \otimes G}\, \otimes_R$$

Figure 10: Sequent calculus rules of the MILL fragment

The interest of Linear Logic is that it provides, for example, a natural and simple coding of Petri Net reachability [23]. Based on the sequent calculus, Linear Logic helps to get a necessary and sufficient condition of reachability from one marking to another, thanks to the equivalence between the reachability in a Petri Net and the provability of the corresponding sequent [25].

Moreover Linear Logic gives the partial firing order of the different transitions in order to reach the final marking M_f from an initial marking M_0 [26].

5.3. Linear logic and Petri Nets

To translate a Petri Net to Linear Logic, a logical formula is associated for each marking and each transition. The left hand of the initial sequent must hold the list of all the transitions that can be fired to obtain a marking M_f from an initial marking M_0. The building of the proof generates a proof tree beginning by a sequent and finishing by the identity axiom. For a given Petri Net, the translation is performed as follows:

- An atomic proposition P is associated with each place p of the Petri Net,
- A single sequent using the multiplicative conjunction TIMES ($\otimes$), is associated with each marking, pre-condition and post-condition of transition,
- For each transition t of the net, an implicative formula is defined as follows:

$$t : \bigotimes_{i \in Pre(pi,t)} P_i \multimap \bigotimes_{o \in Post(po,t)} P_o$$

To show the reachability between two markings M_0 and M_f, the proof of the sequent $M_0, t_1, \ldots, t_n \vdash M_f$ is performed as follows: first, the initial marking M_0 is replaced by separate atoms by applying the $\otimes_L$ rule as many times as necessary. Then, by applying $\multimap_L$, the causality relation of atoms (from M_0 to M_f) can be extracted. Each time the $\multimap_L$ rule is applied, the $\otimes_L$ rule is applied if necessary to separate atoms connected with $\otimes$. The proof continues essentially at the right side of the tree because after each application of the $\multimap_L$ rule, the left member is proven by using, if necessary, the $\otimes_R$ rule. The sequent proof ends when all implicative formulae (expressing transitions) are eliminated.

5.4. Linear Logic and Colored Petri Nets

The application of Linear Logic to CPN reachability analysis requiers a translation between the two models (from CPN to Linear Logic). This translation has to respect the CPN characteristics, particularly token types and arc expressions which do not exist in ordinary PN. To express the token differentiation and their value evolution in CPNs, the predicative Linear Logic is very limited ; this is why, in this work, the first order Linear Logic was preferred. This section presents the chosen First Order Multiplicative Intuitionistic Linear Logic (noted MILL1) and then presents the translation algorithm.

5.4.1. First Order Multiplicative Intuitionistic Linear Logic

MILL1 language is defined as follows:

Alphabet:. The alphabet consists of disjoint sets: a set of variable symbols (e.g. x, y), a set of function symbols (e.g. f, g, h), a set of relation symbols (e.g. R, S, T), the binary connectives ($\otimes$, $\multimap$) and quantifier $\forall$. Each language $\mathcal{L}$ is equipped with a map from $\mathcal{L}$ to the set of natural integers $ar : \mathcal{L} \to \mathbb{N}$. This map ar stands for symbol *arity*.

Terms. Given a language $\mathcal{L}$, the first-order terms over $\mathcal{L}$ are defined by the syntactic category below:

$$\begin{array}{rcl} \tau & ::= & x \\ & | & f(\tau_1, \ldots, \tau_n) \end{array}$$

where x ranges over variables and f belongs to the function symbols of $\mathcal{L}$ such that $ar(f) = n$.

Formulas. First-order formulas of MILL are defined by the inductive syntactic category below:

$$\begin{array}{rcl} \varphi, \phi & ::= & R(\tau_1, \ldots, \tau_n) \\ & | & \varphi \otimes \phi \\ & | & \varphi \multimap \phi \\ & | & \forall x \cdot \varphi \end{array}$$

where R belongs to relation symbols of $\mathcal{L}$ such that $ar(R) = n$ and where the variable x occurrences in the formula φ are bound in formula $\forall x \cdot \varphi$ by the universal quantifier. Variables that are not bound by a quantifier are called free.

Sequents. If Γ is a multiset of formulas separated by "," and φ is a formula then $\Gamma \vdash \varphi$ is a sequent. By taking Γ as a multiset, we will implicitly assume that the sequent comma "," is associative and commutative. Γ is called the *antecedent* of the sequent and φ the *succedent*.

Proofs in MILL1 are given in terms of proof trees that are inference rule composition over judgments. Judgments are pairs of a set of formulas Γ and a formula φ that are written $\Gamma \vdash \varphi$. This means that the formula φ is a logical consequence of the conjunction of those of Γ. Inference rules (n-ary) are relations between $n + 1$ judgments that are noted as follows:

$$\frac{\Gamma_1 \vdash \varphi_1 \qquad \cdots \qquad \Gamma_n \vdash \varphi_n}{\Gamma \vdash \varphi}$$

which means that it is sufficient to establish the below-rule judgment $\Gamma \vdash \varphi$ if the above-rule ones hold; in other words, to establish the below-rule judgment it is necessary to prove the above-rule judgments $\Gamma_i \vdash \varphi_i$ $(1 \leqslant i \leqslant n)$. Inference rules of MILL are given in Figure 11.

$$\frac{}{\varphi \vdash \varphi}\, id \qquad \frac{\Gamma, \varphi \vdash \phi}{\Gamma \vdash \varphi \multimap \phi} \multimap_r$$

$$\frac{\Gamma \vdash \varphi \qquad \phi, \Delta \vdash \psi}{\Gamma, \varphi \multimap \phi, \Delta \vdash \psi} \multimap_l$$

$$\frac{\Gamma \vdash \varphi \qquad \Delta \vdash \phi}{\Gamma, \Delta \vdash \varphi \otimes \phi} \otimes_r \qquad \frac{\Gamma, \varphi, \phi \vdash \psi}{\Gamma, \varphi \otimes \phi \vdash \psi} \otimes_l$$

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \forall x \cdot \varphi} \forall_r \quad (*) \qquad \frac{\Gamma, \varphi \vdash \psi}{\Gamma, \forall x \cdot \varphi \vdash \psi} \forall_l$$

The constraint $(*)$ requires that the variable x isn't free in formulas of Γ.

Figure 11: First-Order Multiplicative Intuitionistic Linear Logic

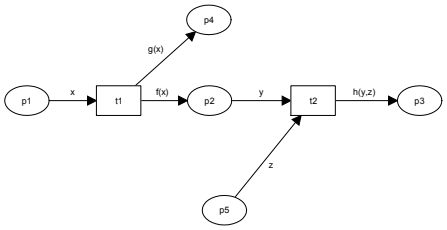


Figure 12: Colored Petri Net Example

5.4.2. Translation of a Colored Petri Net to MILL1

A logical formula is associated to each marking and each transition. The left hand of the initial sequent must contain the list of all the transitions that can be fired to obtain the marking M_f from an initial marking M_0. The building of the proof generates a proof tree beginning by a sequent and finishing by the identity axiom. For a given CPN, the translation is performed as follows:

- A unary atomic predicate R is associated with each place p of the CPN,
- A single sequent using the multiplicative conjunction TIMES ($\otimes$), is associated with each marking, each pre-condition and each post-condition of transition,
- To each transition t of the net, an implicative formula is defined as follows:

$$t : \forall x_1 \ldots \forall x_i \left(\bigotimes_{i \in Pre(pi,t)} R_i(x_i) \multimap \bigotimes_{o \in Post(po,t)} R_o(f_o(X_o)) \right)$$

where: x_i are variables marking the input arcs of t ; f_o are functions associated to the output arcs of t ; X_o are vectors composed of different associations of x_i.

The following example translates the CPN illustrated in Figure 12 to its equivalent in MILL1.

- places p_i are translated by unary atomic predicates $x \mapsto R_i(x)$ where $(1 \leqslant i \leqslant 5)$;
- the transition t_1 is translated by the formula: $\varphi_1 \mathrel{\widehat{=}} \forall x \cdot \Big(R_1(x) \multimap R_2\big(f(x)\big) \otimes R_4\big(g(x)\big)\Big)$
- transition t_2 is translated by the formula: $\varphi_2 \mathrel{\widehat{=}} \forall y \cdot \forall z \cdot \Big(R_2(y) \otimes R_5(z) \multimap R_3\big(h(y,z)\big)\Big)$
- the initial state is translated by the formula: $\phi_0 \mathrel{\widehat{=}} R_1(i) \otimes R_5(j)$
- the final state is translated by the formula: $\phi \mathrel{\widehat{=}} R_3(a) \otimes R_4(b)$

The remaining of the example treats a case of reachability between two markings M_0 and M_f where: $M_0 = \langle i \rangle.p1 + \langle j \rangle.p5$ and $M_f = \langle a \rangle.p3 + \langle b \rangle.p4$. The reachability between M_0 and M_f is obtained by the means of the judgment $\phi_0, \varphi_1, \varphi_2 \vdash \phi$ following proof in MILL1 (see proof tree in Figure 13): first, the initial state formula ϕ_0 is treated by $\otimes_l$, then the transition σ_1 is treated by $\forall_l$ and $\multimap_l$. The result of the first transition is treated by $\otimes_l$. The second transition σ_2 is treated twice by $\forall_l$ and $\multimap_l$. The results of the second transition is treated by $\otimes_r$. In a parallel direction, we treat the final state formula ϕ by $\otimes_r$ as well. We finally apply id rules and unify the remaining equations by the substitution ς:

$$\begin{array}{rcl} \varsigma(a) & = & h\big(f(i), j\big) \\ \varsigma(b) & = & g(i) \end{array}$$

6. Case study

This section shows a practical use of the backward reachability (through an inverse CPN) presented in previous paragraphs. The application example is inspired from the railway transport. It consists in the tram braking system. The aim is to show, through a case study, how to exploit efficiently the backward reachability analysis. The organization of this section begins with the specification of the studied system (in this paper, we focus only on the braking system). It is followed by the detailed modelling using CPN. The model includes complement information about the functions calculating braking values and the impact of each braking type on the vehicle speed. Then, finally, the CPN inversion and the backward reachability analysis are presented.

6.1. Specification

The driver has a control console (Figure 14.a) endowed with traction/braking handle which gives effort orders (acceleration, neutral, braking).

The acceleration or braking orders are transmitted to the different Traction Braking Electronics (TBE) which manage the different train bogies (set of two axles). The common braking mode, named *service braking*, is obtained by pulling the handle beyond the neutral position but without pulling it thoroughly. The extreme position of the handle causes an *emergency braking* (see below). In the case of the service braking, the TBE realize the effort (variable according to the handle position) with an electrodynamic brake. This braking mode uses also, when necessary, the disc brake. To avoid frequent falls in the train, the braking effort is modulated by the passengers load (measured by a weighing device). The device named anti-slip (anti wheels locking, equivalent to ABS system in automotive world) is also active in this mode.

The *emergency braking* is obtained by pulling the handle thoroughly. It is maintained till the train stops. In this mode, maximum deceleration performance is needed. So, TBE order the maximum effort to the electrodynamic brake and the disc brake, the modulation with the passengers weight and the anti-slipping stay active. The sanding (injection of sand under the wheels) is ordered for the whole train. In addition to this, electromagnetic skates (long electromagnet) are applied on the rail.

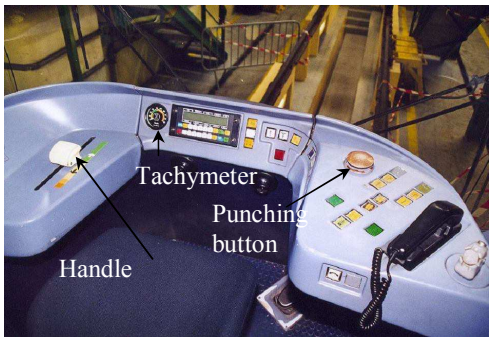


Figure 14: Control console and skate views of a tramway

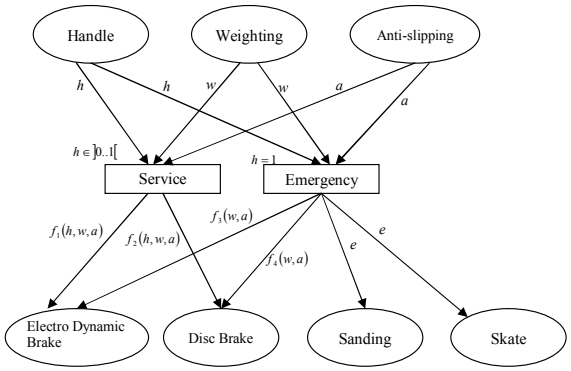


Figure 15: CPN model of the tramway braking system

6.2. *Colored Petri Net Modelling*

The CPN, shown in Figure 15 models the different braking types cited in the specification: service braking and emergency braking. Each braking type is represented by a transition which links causes (pre conditions) and requested devices (post conditions). The specification does not provide numerical values or functions about orders to apply. Their choice has only limited effects on the analysis. So these choices are made by authors to demonstrate the progress and the contribution of the backward reachability analysis.

The transition *Service* models the service breaking which is engaged by the position of the handle (represented by the place *handle*) beyond the neutral position but not in the maximal one. This condition is represented by the guard associated to this transition. It allows the transition firing if the variable $m \in]0..1[$ like that 0 is the neutral position and 1 is the maximal one. The effort order value is proportional to the handle position. It is modulated also by the weighting and the anti-slipping (represented by the places having same names). The values of the variables w and a (for weighting and anti-slipping respectively) belong to the interval $[0..1]$. The output braking order is sent to the electrodynamic brake. It is calculated by the function $f_1(h, w, a) = 5h + 2w + a$. When the cumulated value of w and a is higher then a threshold ($1, 5$ in this case), an order is sent to the disc brake calculated by the function $f_2(h, w, a) = 7h + 2w + a$.

The transition *Emergency* models the emergency braking caused by pulling the handle to its maximal position. This

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{f(x)=y}{R_2(f(x))\vdash R_2(y)}\,id \quad \dfrac{j=z}{R_5(j)\vdash R_5(z)}\,id}{R_5(j),R_2(f(x))\vdash R_2(y)\otimes R_5(z)}\otimes_r \quad \dfrac{h(y,z)=a}{R_3(h(y,z))\vdash R_3(a)}\,id \quad \dfrac{g(x)=b}{R_4(g(x))\vdash R_4(b)}\,id}{R_5(j),R_2(f(x)),R_4(g(x)),R_2(y)\otimes R_5(z)\multimap R_3(h(y,z))\vdash\phi}\otimes_r}{R_5(j),R_2(f(x)),R_4(g(x)),\varphi_2\vdash\phi}\multimap_l}{R_5(j),R_2(f(x))\otimes R_4(g(x)),\varphi_2\vdash\phi}\otimes_l}{R_1(i),R_5(j),R_1(x)\multimap R_2(f(x))\otimes R_4(g(x)),\varphi_2\vdash\phi}\forall_l\times 2}{R_1(i),R_5(j),\varphi_1,\varphi_2\vdash\phi}\multimap_l}{\phi_0,\varphi_1,\varphi_2\vdash\phi}\otimes_l$$

$$\dfrac{i=x}{R_1(i)\vdash R_1(x)}\,id$$

Figure 13: example of proof tree calculating reachability conditions in CPN

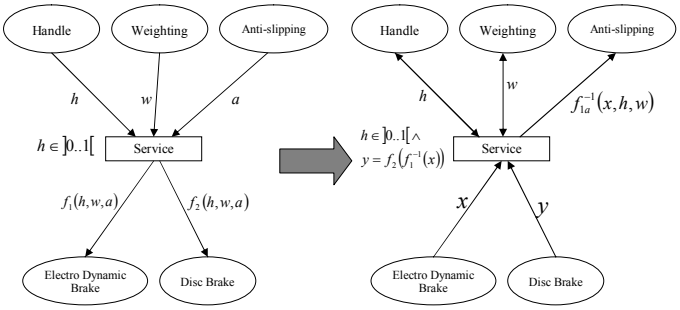


Figure 16: Transformation of the transition *Service*

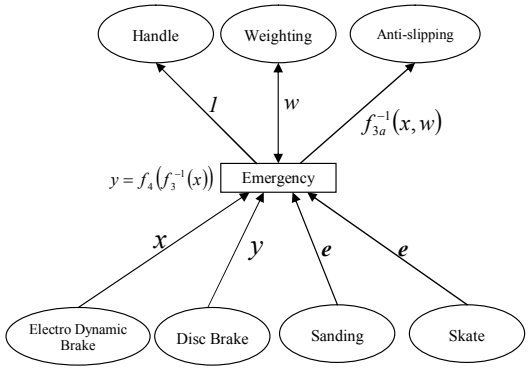


Figure 17: Transformation of the transition *Emergency*

condition is represented by the guard $m = 1$ associated to the transition. This braking causes an action over all braking material devices. It sends orders, modulated by the weighting and the anti-slipping, to the electrodynamic brake and the disc brake calculated by the functions $f_3(p, a) = 5 + 2p + a$ and $f_4(p, a) = 7 + 2p + a$. It engages also the sanding and the skates by a signal represented by an anonymous token e.

6.3. Colored Petri Net Model transformation

The transition *Service* is, at the same time, parallel and parametric. It is parallel because same input variables are used is two output functions. The parametric property comes from the output variables which are multi-variable functions. Several transformations are possible, each one depends on the input variable to calculate, but all of them have to respect the two properties (parallel and parametric). One of possible transformations is illustrated in Figure 16. The values of h and w are assumed in their domaines. So the calculation is done over the variable a using the partial inverse function $f_{1a}^{-1}(x) = x - 5m - 2p$. When calculating, the values that do not belong to the color set of the $Anti - Slipping$ place (so variable a) are deleted. An additional guard is associated to the transition according to the application of the parallel transformation. This guard is written $y = f_2(f_1^{-1}(x))$ and

$$\begin{aligned} f_1^{-1}(x) &= (h, w, a) \\ &= (\tfrac{x-2w-a}{5}, \tfrac{x-5h-a}{2}, x - 5h - 2w) \end{aligned}$$

$$\begin{aligned} f_2(f_1^{-1}(x)) = \quad & If\ (((3x - 15h - 4w - a)/2) < 1.5)\ Then \\ & \quad 0 \\ & Else \\ & \quad (17x - 50h - 24w - 12a)/5 \end{aligned}$$

The transition *Emergency*, for the same reasons as the transition *Service*, is parallel and parametric. In addition, it is mixed because it has constant arc expressions that produce anonymous token e. One of the possible transformations of this transition is illustrated in Figure 17. In the original CPN, the transition firing is conditioned by the value $h = 1$. So, in the transformed model, the expression of the arc $(Emergency, Handle)$ is associated to the constant value 1. The $Anti - slipping$ values are calculated using the function $f_{3a}^{-1}(x, w) = x - 2w - 5$. The added guard express adequation between values of x and y like that $f_4(f_3^{-1}(x)) = 2x - 2w - a - 3$.

6.4. Braking system Analysis

6.4.1. The Service Braking

The first analysis concerns the service braking. The process consists in studying a braking which requires, at the same time, the electrodynamic brake and the disc brake with a known value on the first one (5 in this example). The aim is to find possible command values applied to the disc brake and the control values corresponding (weighting w and anti-slipping a). We consider in this case that the handle position is known and constant: $h = 0.5$

The backward reachability analysis uses the inverse CPN and takes into account, in the partial state, the value 5 of the electrodynamic brake and the value 0.5 of the handle. The (backward) firing of the transition *Service* is done thanks to the marking enhancement as the disc brake order, which is necessary on the transition firing, is unknown. It is added as a complementary assumption meaning that the disc brake device was activated. Calculations use the guard associated to the transition and the constraint $2w + a = 2.5$ is obtained and easily verified in each component of the triplet $f_1^{-1}(x)$, which is itself a component of the guard. The results are illustrated in Figure 18. The backward analysis algorithm, which searches the input values, is written as follows:

for $p \in [0..1]$ **do**
  $a \leftarrow 2.5 - 2p$
  if $0 \leq a \leq 1$ **then**
    Return $f_2(f_1^{-1}(x))$ /*$x = 5, m = 0.5$*/
  end if
end for

The conclusion of this case analysis is that the final marking$M_f = \langle 5 \rangle.ElectroDynamicBrake$ is reachable, by firing the transition $Service$, by the initial marking $M_0 = \langle 0.5 \rangle.Handle + \langle \hat{w} \rangle.Weighting + \langle \hat{a} \rangle.AntiSlipping$ such that $\hat{a} = 2.5 - 2\hat{w}$ and $0 \leq \hat{a} \leq 1$. The final marking really reachable is the enhanced marking $M_{fe} = \langle 5 \rangle.ElectroDynamicBreake + \langle \hat{y} \rangle.DiscBrake$. The values of $\hat{y}$, $\hat{w}$ and $\hat{a}$ are the possible combination of y, w and a (respectively) corresponding to the line segment shown in Figure 18.

To find same results in forward reachability analysis, the search algorithm has to choose among all possible y val-

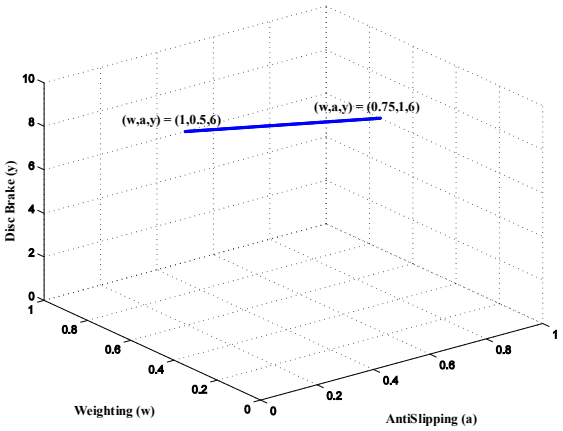


Figure 18: Service braking analysis

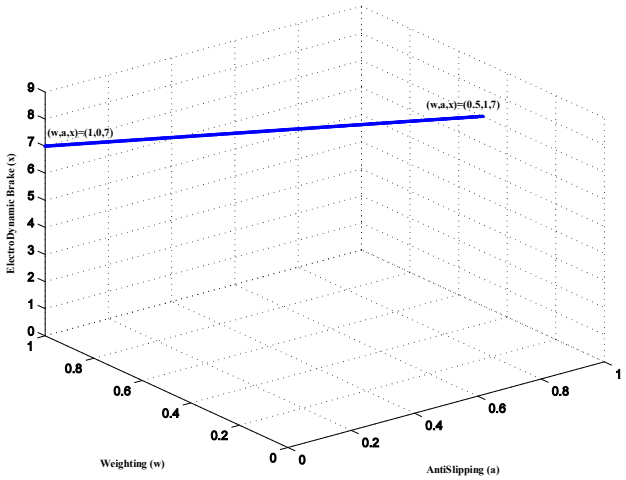


Figure 19: Emergency Braking in case of *Disc Brake* case study

ues those which are really possible using the constraint $f_1(h, w, a) = 5 \wedge h = 0.5$. But no constraint joins the variables w and a. The forward reachability analysis algorithm can be written as follow:

for $a \in [0..1]$ **do**
  for $p \in [0..1]$ **do**
    if $f_1(m, p, a) = 5$ **then**
      Return $f_2(m, p, a)$
    end if
  end for
end for

The difference between the two algorithms (backward reachability and forward reachability) is their algorithmic complexity. Knowing that the the complexity of f_2 and $f_2(f^{-1})$ are strictly identical, the difference between the two is that the second presented algorithm (forward reachability) includes two nested loops whereas the first includes only one loop. This is due to the inverse CPN which exploits the parallelism relation (parallel transformation) thus introducing an a priori knowledge reducing the calculation complexity.

6.5. The Disc Brake

The second analysis concerns the diagnosis of a particular state of the disc brake. The known state is the command value of the disc brake. In this case study, it is chosen as 9. The purpose is to find initial states (M_0) that could produce the final partial state ($M_f = \langle 9 \rangle.DiscBrake$) knowing only this last state. The analysis is done by backward reachability using the inverse CPN marked with marking corresponding to M_f.

6.5.1. The transition Emergency

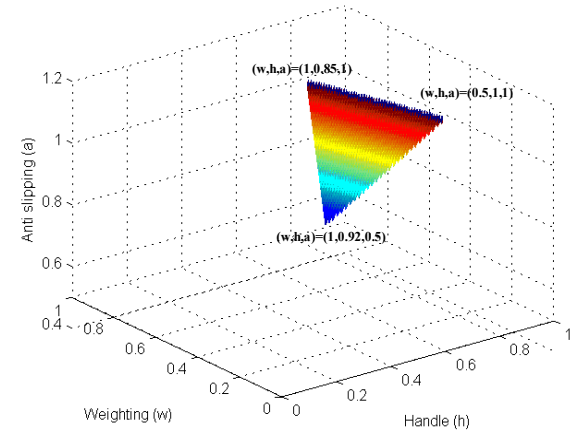
The inverse firing of the transition *Emergency* requires a marking enhancement for $Sanding$, $Skate$ and $ElectroDynamiBrake$. In the first two places, the tokens to add are anonymous (no calculus to do). However, token values in $ElectroDynamiBrake$ (the variable x) have to satisfy the guard. To simplify the process, we rewrite the guard to get x values knowing y. This guard can be written as $x = f_3(f_4^{-1}(y))$ which gives $x = 2y - 2w - a - 9$ under the constraint $2w + a = y - 7$.

Applying an analogue approach as the one used in the previous case study, the possible order values of the electrodynamic brake and the control values corresponding (weighting and anti-slipping) are calculated. The results are illustrated in Figure 19. The conclusion is that the marking $M_f = \langle 9 \rangle.DiscBrake$ is backward reachable from $M_0 = \langle 1 \rangle.Handle + \langle \hat{w} \rangle.Weighting + \langle \hat{a} \rangle.AntiSlipping$. This reachability is conditioned by the marking enhancement. The final marking really reachable is the enhanced marking $M_{fe} = \langle \hat{x} \rangle.FreinElectroDynamique + \langle 9 \rangle.DiscBrake + \langle e \rangle.Sanding + \langle e \rangle.Skate$. The values of $\hat{x}$, $\hat{w}$ and $\hat{a}$ are possible combinations of variables x, w and a (respectively) corresponding to the line segment of Figure 19.

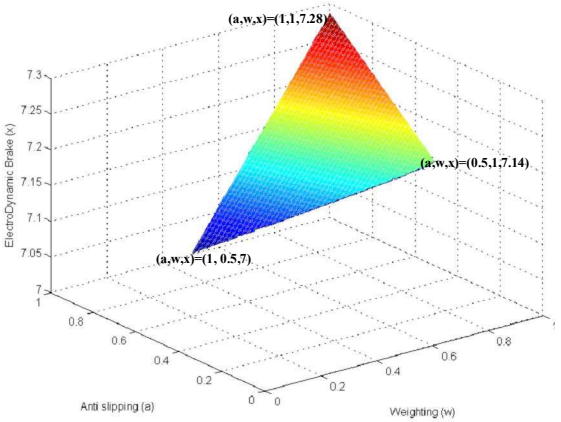
6.5.2. The transition Service

As for the transition *Emergency*, the backward firing of the transition *Service* requires a marking enhancement. The place to enhance is $ElectroDynamicBrake$. Token values in this place (variable x) have to satisfy the guard. Since the value of y is known, the guard is rewritten to calculate easily x values. The new guard is written $x = f_1(f_2^{-1}(y))$. In this case, the condition $y \neq 0$ in the function f_2 is valid. So in the original CPN, the condition $p + a > 1.5$ is valid and the token value in $DiscBrake$ (9 in this example) is obtained by applying the formula $7m + 2p + 1$. The new guard is given by $(0 < m < 1) \wedge (3y - 21m - 4p - a > 3) \wedge (x = (19y - 98h - 24w - 12a)/7)$ under the constraint $a = y - 7h - 2w$.

By using the inverse CPN modified and completed, the possible order values applied to the electrodynamic brake and the control values corresponding (weighting and anti-slipping) are calculated. The results can't be shown in a unique graphic (too many variables). This is why they are drown in the two schemas of Figure 20. The first one (Figure 20.a) shows possible input combinations of variables h, w and a. The second one (Figure 20.b) shows two input variables (h and w) and corresponding x values. The conclusion is that the



-a-



-b-

Figure 20: Service Braking in case of *Disc Brake* case study

marking $M_f = \langle 9 \rangle.DiscBrake$ is backward reachable by $M_0 = \langle \hat{h} \rangle.Handle + \langle \hat{w} \rangle.Weighting + \langle \hat{a} \rangle.AntiSlipping$. This reachability is conditioned by the marking enhancement. The final marking really reachable is the enhanced marking $M_{fe} = \langle \hat{x} \rangle.ElectroDynamicBrake + \langle 9 \rangle.DiscBrake$. The values of $\hat{x}$, $\hat{h}$, $\hat{w}$ and $\hat{a}$ are possible combinations of variables x, h, w and a (respectively) corresponding to the illustrated values in Figure 20.

6.5.3. Summary of the Disc Brake *case study*

The backward reachability analysis of the final marking considered in this case study ($M_f = \langle 9 \rangle.DiscBrake$) shows that it is reachable by two different paths (Figure 21). Each path represents a specific scenario and needs complementary information about the system final state. This is why the final marking was enhanced to assume coherent system states.

7. Conclusion

The diagnostic and dependability issues in embedded systems are very large and not easy to deal with. This work focuses on a particular aspect which is model based verification

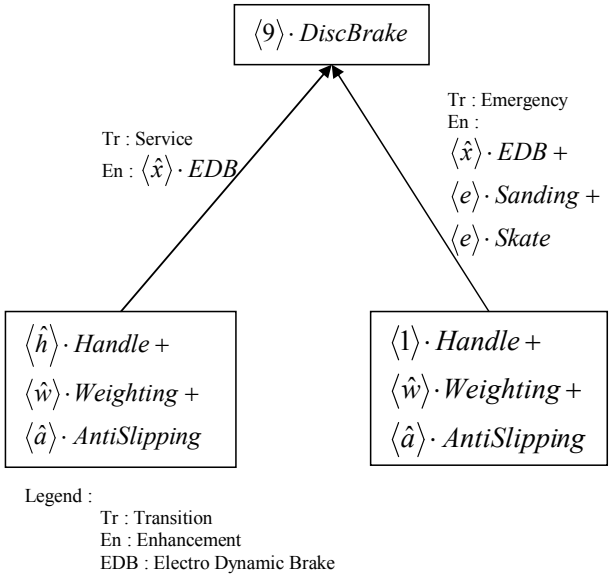


Figure 21: Backward reachability tree of *Disc Brake* case study

and analysis. The main objective is to propose a formal method for: 1) looking for sources of a known failure (diagnosis) ; 2) performing analysis to verify/validate a system relatively to a given specification and, in case of need, to propose a correction of the system to satisfy the specification.

Such a study needs an appropriate model that can represent the structure and the dynamic behavior of the system and that has also a formal basis. The chosen model is a Colored Petri Net (CPN) and in this work we propose a structural analysis by backward reachability using inverse models. The CPN inversion is a research issue because of the model structure. Some solutions can be found in literature ([2], [7], [3], [1]). We proposed a generalization of the method of [3] based on inverse CPN. The basic idea is to define some elementary transformations, each one is appropriate to a specific transition structure type. These transformations can be combined to express a very large amount of possible cases. The definition of the inverse CPN is the first part of the contribution. The second part aims the backward analysis. To achieve this, complementary mechanisms were defined to drive the analysis and to deal with a lack of information.

The inverse CPN and the complementary mechanisms are first presented in an intuitive way. To consolidate the formal aspect of the proposed approach, two theoretical studies were presented. The first formalism (linear algebra) proves the pertinence of presented transformation algorithms in a local way by focusing in a single transition. The second formalism (linear logic) aims to prove the correctness of the performed backward analysis using the inverse CPN. This is done by adapting the LL fragment that has to be used in a CPN context and which is MILL1 (*First Order Multiplicative Intuitionistic Linear Logic*). When the translation MILL1/CPN is made, the reachability is proven thanks to sequent calculus.

To test the proposed method, a practical case inspired from

railway domain was studied. It treats a specification of a tramway braking system. The system was modelled using CPN and then the inverse CPN was generated. The backward analysis answers efficiently a sequence of diagnostic and V&V questions and the approach proves its usefulness.

References

[1] J. Müller, E. Schnieder, Duality in high level petri-nets: a basis to do diagnoses, in: WSC '07: Proceedings of the 39th conference on Winter simulation, IEEE Press, Piscataway, NJ, USA, 2007, pp. 629–636.

[2] L. Portinale, Petri net models for diagnostic knowledge representation and reasoning, Ph.D. thesis, University of Torino (Italy) (1993).

[3] S. Khalfaoui, Mthode de recherche des scnarios redouts pour l'valuation de la sret de fonctionnement des systmes mcatroniques du monde automobile, Ph.D. thesis, Institut National polytechnique de Toulouse (France) (Septembre 2003).

[4] M. Medjoudj, Contribution l'analyse des systmes pilots par calculateurs : Extraction de scnarios redouts et vrification de contraintes temporelles, Ph.D. thesis, Universit Paul Sabatier de Toulouse (France) (March 2006).

[5] N. Sadou, Aide la conception des systmes embarqus sr de fonctionnement, Ph.D. thesis, Universit Paul Sabatier de Toulouse (France) (November 2007).

[6] C. Anglano, L. Portinale, B-W analysis: a backward reachability analysis for diagnostic problem solving suitable to parallel implementation., in: Valette, R. (Ed.), Lecture Notes in Computer Science; Application and Theory of Petri Nets 1994, Proceedings 15th International Conference, Zaragoza, Spain, Vol. 815, Springer-Verlag, 1994, pp. 39–58.

[7] S. M. Cho, H. S. Hing, S. D. Cha, Safety analysis using colored petri nets., in: Proc. Asia-Pacific Software Engineering Conference (APSEC-96), 4-7 December 1996, Seoul, Korea, 1996, pp. 176–183.

[8] N. G. Leveson, J. L. Stolzy, Safety analysis using petri nets, IEEE Trans. Softw. Eng. 13 (3) (1987) 386–397.

[9] H. J. Kowalsky, Lineare Algegra, de Gruyter Lehrbuch, 1979, 9th Edition.

[10] C. A. Petri, Communication with automata, Ph.D. thesis, Darmstadt Institut fr Instrumentelle Mathematik, Bonn (Germany) (1962).

[11] K. Jensen, G. Rozenberg, High-level Petri Nets : Theory and Application, Springer-Verlag, Germany, 1991.

[12] S. Haddad, A reduction theory for coloured nets, Lecture notes in Computer science n 424 Springer-Verlag (1989) 209–235.

[13] N. Sadou, H. Demmou, Reliability analysis of discrete event dynamic systems with petri nets, Reliability Engineering & System Safety 94 (11) (2009) 1848–1861.

[14] S. la direction M. Diaz, Les rseaux de Petri : modles fondamentaux, Herms Science Publications, Paris (France), 2001.

[15] T. Fruewirth, S. Abdennadher, Essentials of Constraint Programming, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.

[16] R. Goblot, Algbre Linaire, Masson, Paris (France), 1995.

[17] S. la direction J. P. Richard, Algbre et analyse pour l'automatique, Herms Science Publications, Paris (France), 2000.

[18] F. Cottet-Emard, Algbre linaire et bilinaire : : cours et exercices corrigs, de Boeck, Bruxelles (Belgium), 2005.

[19] S. Evangelista, S. Haddad, J.-F. Pradat-Peyre, Syntactical colored petri nets reductions, Automated Technology for Verification and Analysis (ATVA 05) Taipei, Taiwan, October 4-7 (2005) 202–216.

[20] A. Ratzer, L. Wells, H. Lassen, M. Laursen, J. Frank, M. Stissing, M. Westergaard, S. Christensen, K. Jensen, Cpn tools for editing, simulating, and analysing coloured petri nets, in: Applications and Theory of Petri Nets 2003: 24th International Conference, ICATPN 2003, Springer Verlag, 2003, pp. 450–462.

[21] R. Valette, L. Knzle, Rseaux de petri pour la dtection et le diagnostic, G.R.Automatique, Journes Nationales Sret, surveillance, supervision.

[22] J. Girard, Linear logic, Theorical Computer Sciences 50 (1987) 1–102.

[23] P. Lincoln, Linear logic, SIGACT News 23 (2) (1992) 29–37.

[24] G.Gentzen, The Collected Works of Gerhard Gentzen, North-Holland Publishing Company, Amsterdam, 1969.

[25] F.Girault, Formalisation en logique linaire du fonctionnement des rseaux de petri, Ph.D. thesis, Universit Paul Sabatier, Toulouse (1997).

[26] H.Demmou, S.Khalfaoui, E.Guilhem, R.Valette, Critical scenarios derivation methodology for mechatronic systems., Reliability Engineering and System Safety 84 (2004) 33–44.