

Application of linear logic to Backward Reachability Analysis of Colored Petri Nets

Mohamed BOUALI, Jérôme ROCHETEAU & Pavol BARGER

Heudiasyc Laboratory, CNRS UMR 6599

Université de Technologie de Compiègne, France

Tel : +33 3 44 23 44 23

Fax : +33 3 44 23 44 77

{mohamed.bouali, jerome.rocheteau, pavol.barger}@hds.utc.fr

ABSTRACT: This paper deals with a formal method for the study of the backward reachability analysis applied on Colored Petri Nets (CPN). The proposed method proceeds in two steps : 1) it translates CPN to terms of the Multiplicative Intuitionistic Linear Logic (MILL); 2) it proves sequents by constructing proof trees. The translation from CPN to MILL must respect some properties such as the semantic associated to tokens. That is why, the First-Order MILL (MILL1) is used for translation. The reachability between two markings, the initial marking and the final marking, is expressed by a sequent which can be proven (if the initial marking is backward-reachable from the final one) using first-order terms unification and/or marking enhancement.

1 Introduction

System dependability is an important research issue especially if applied to critical domains. Systems are verified through tools that check the compliance of their properties with the 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 system modelling and analysis.

Petri Nets (PN) (Petri 1962), and especially, Colored Petri Nets (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 evolution. The PN analysis can be done in several manners like exhaustive reachable state space enumeration or Monte Carlo simulation (Metropolis and Ulam 1949). These methods allow to study request/action effects on the model behavior, but they suffer from drawbacks such as combinatorial explosion and results correctness (some results are bound with a potentially known error range). To avoid these drawbacks, we propose to use the structural analysis using directly the model itself. In addition, this work is particularly interested in the diagnosis and the fault driven analysis. That means, by knowing a particu-

lar final state (which represents a failure state), the sources of this state can be found. In such analysis, all possible initial configurations must be studied to find those leading to the considered failure state. This is why, it is interesting to study the reachability between two markings studied under two dual manners (Khalifaoui 2003): the forward reachability and the backward reachability. The first case consists in building state successors one by one, starting with the initial marking M_0 and ending with the final marking M_f . Thus, considering M_0 as the present state, M_f is considered as the future state. In the backward reachability, M_f constitutes the present state and M_0 is regarded as the past state. The general idea is to build, from the present state, the predecessor states until the reach of the past state which is logically the source of the present marking.

To perform the backward reachability analysis over CPN, (M.Bouali, P.Barger, and W.Schon 2009a; M.Bouali, P.Barger, and W.Schon 2009b) developed a method based on inverse Colored Petri Net (inverse CPN) which results from transformations applied on the original CPN. Knowing that, in the case of ordinary Petri Nets, the linear logic offers another more formal method to study reachability between to states, the idea is to generalize this method to CPN and especially for inverse CPN. The equivalence between structural analysis and linear logic (applied to CPN)

can be obtained by performing appropriate *translation* from CPN to linear logic. The linear logic formalism is very interesting because it can constitute a formal proof for the structural backward reachability.

CPNs model the system structure and its dynamic behavior in the same model. The dynamic behavior is modelled thanks to token evolution. After each transition firing, some tokens are consumed and some other are produced. This notion of production/consumption cannot be expressed in classical logic, that is why the MILL was preferred. On the other hand, unlike Ordinary Petri Nets, token in CPN is of a certain type (color) and belongs to a set of this type (color set) and is transformed by arc expressions. So the translation from CPN to MILL must respect these properties. That is why, the First-Order MILL (MILL1) is used for the translation. CPN Places are expressed in MILL1 by unary relation symbols (Propositional variables) which allow to deal with arc expressions and token values. CPN transitions are expressed in MILL1 by implicative formulas allowing introduction of universal quantifiers. The reachability between the two markings M_0 (initial marking) and M_f (final marking) is expressed by a sequent which can be proven (if M_0 is backward-reachable from M_f) using first-order terms unification and/or marking enhancement.

This paper is organized as follows : The section 2 gives definitions of Petri Nets and Colored Petri Nets. It also introduces the structural backward reachability in CPN. The section 3 presents the Linear Logic and its relation with Petri Nets. It is followed, in the section 4, by the details of translation from CPN to MILL1 and the reachability analysis using sequents in linear Logic. The section 5 contains a presentation of the case study. The paper ends by the conclusion and an outline of future perspectives.

2 Structural Backward Reachability for Colored Petri Nets

2.1 Petri Net and Colored Petri Net

A Petri Net (Petri 1962), called also 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 (CPN) (Jensen and Rozenberg 1991) introduces the notion of token types, namely tokens are differentiated by colors, which may contain 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 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:

$$\forall p \in P : [Type(I(p)) = C(p)_{MS}]$$

2.2 Colored Petri Net inversion

Backward reachability in CPNs is a dual concept of the forward reachability. That is, if a marking M_f is reachable from M_0 , M_0 is said backward reachable from M_f . Backward-reachability means that M_0 is a cause or a source of M_f . To perform the backward reachability analysis, an inverse CPN is used. It is obtained by application of a set of structural transformations on the original CPN. These transformations are directly dependent on the CPN structure. Consequently, we have to define a transformation rule for each structure case studied. Nevertheless, the most common transformations in representative cases are presented bellow.

Basic transformation rule : Tab.1.a shows a trivial case of a CPN inversion: input and output arcs are marked with constants a, b . Tab.1.b shows a case where the input arc is marked with variable x , the output arc is marked with function $f(x)$ and a guard $G(x)$ is associated to the transition t .

Mixed transformation rule : Tab.1.c 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 the basic transformation. Each variable is used by one and only one function. In the opposite case, see the *Parametric transformation* below.

Parametric transformation rule : Some CPN structures can't be reversed 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, some additional information, like color sets, are needed. Two cases follow : the first concerns arc expressions which are multivariable equations (Tab.1.d), the second treats input variables which are not associated with output functions (Tab.1.e).

Parallel transformation rule : The term '*parallel*' means existence of a shared variable. Tab.1.f shows the case where the same variable is used by more than one function (two functions in this case).

3 Linear Logic

Linear logic was introduced by Girard (Girard 1987). Its expressive power is demonstrated by some very natural encodings of computational models such as Petri Nets, counter machines, Turing machines, and others (Lincoln 1992). Linear logic differs from classical logic by introducing the notion of a resource. Classical logic deals with static propositions where each proposition is either true or false. Because of the static nature of propositions in classical 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, 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 nonsense in the intended interpretation of propositions as resources: two cups of hot drinks cannot be

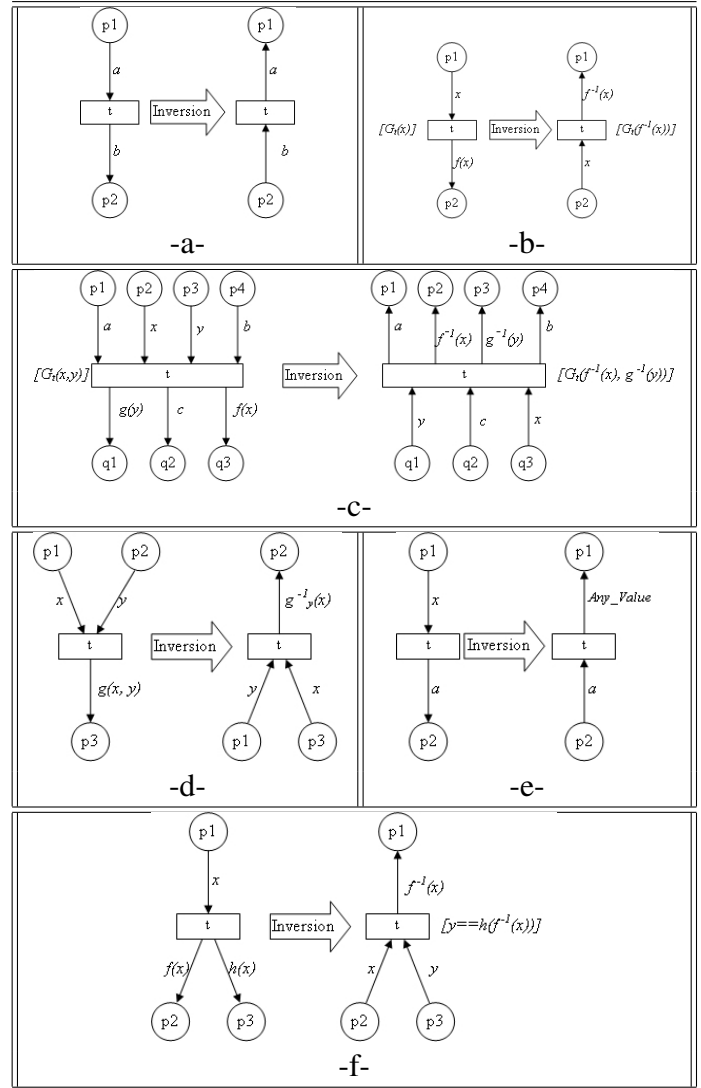


Table 1: CPN transformations for inversion

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 MILL fragment (*Multiplicative Intuitionistic Linear Logic*). This fragment contains the multiplicative connector *TIMES* (\otimes) and 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 (Girard 1987).

The sequent calculus notation, due to Gentzen (G.Gentzen 1969), is composed of two sequences of formulas separated by a turnstile symbol (\vdash). The formula $\Gamma, \Gamma' \vdash \Delta, \Delta'$ means that the conjunction of Γ and Γ' allows to deduce the disjunction 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:

$$\frac{\text{Hypothesis}_1 \quad \text{Hypothesis}_2}{\text{Conclusion}} \text{Rule}_{\text{attribute}}$$

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 Fig. 1 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).

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

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

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

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

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

Figure 1: 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 (Lincoln 1992). 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 (F.Girault 1997). Moreover linear logic gives the partial firing order of the different transitions to reach a final marking M_f from an initial one M_0 (H.Demmou, S.Khalifaoui, E.Guilhem, and R.Valette 2004).

To translate a Petri Net to linear logic, a logical formula is associated to 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,
- To each transition t of the net an implicative formula is defined as follows:

$$t : \bigotimes_{i \in \text{Pre}(p_i, t)} P_i \multimap \bigotimes_{o \in \text{Post}(p_o, t)} P_o$$

To show reachability between two markings M_0 and M_f , the proof of the sequent $M_0, t_1, \dots, t_n \vdash M_f$ is performed as follows: first, the initial marking M_0 is replaced in separate atoms by applying the \otimes_L rule as many times as necessary. Then, by applying \multimap_L , causality relation of atoms (from M_0 to M_f) can be extracted. Each time \multimap_L rule is applied, \otimes_L rule is applied if necessary to separate atoms relied by \otimes . The proof continues essentially at the right side of the tree because after each application of \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.

4 Linear logic and Colored Petri Nets

The application of linear logic to CPN reachability analysis requires translation between the two models (from CPN to linear logic). This translation have to respect characteristics of CPN, particularly tokens types and arc expressions which not exist in ordinary PN. To express tokens differentiation and their values 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

linear logic fragment used to translate CPNs which is First Order Multiplicative Intuitionistic Linear Logic (noted MILL1) and then it presents the translation algorithm.

4.1 First Order Multiplicative Intuitionistic Linear Logic

MILL1 language is defined as follow:

Alphabet: The alphabet consists on 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 $ar : \mathcal{L} \rightarrow \mathbb{N}$. 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:

$$\tau ::= x \quad | \quad f(\tau_1, \dots, \tau_n)$$

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

Formulas: First-order formulas (or formulæ) of MILL are defined by the inductive syntactic category below:

$$\varphi, \phi ::= R(\tau_1, \dots, \tau_n) \quad | \quad \varphi \otimes \phi \quad | \quad \varphi \multimap \phi \quad | \quad \forall x \cdot \varphi$$

where R belongs to the relation symbols of \mathcal{L} such that $ar(R) = n$ and where the variable x occurrences in 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:

$$\frac{\Gamma_1 \vdash \varphi_1 \quad \dots \quad \Gamma_n \vdash \varphi_n}{\Gamma \vdash \varphi}$$

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 $\Gamma_i \vdash \varphi_i$ ($1 \leq i \leq n$). Inference rules of MILL are given in figure 2.

$$\begin{array}{c} \frac{}{\varphi \vdash \varphi} id \quad \frac{\Gamma, \varphi \vdash \phi}{\Gamma \vdash \varphi \multimap \phi} \multimap_r \\ \frac{\Gamma \vdash \varphi \quad \phi, \Delta \vdash \psi}{\Gamma, \varphi \multimap \phi, \Delta \vdash \psi} \multimap_l \\ \frac{\Gamma \vdash \varphi \quad \Delta \vdash \phi}{\Gamma, \Delta \vdash \varphi \otimes \phi} \otimes_r \quad \frac{\Gamma, \varphi, \phi \vdash \psi}{\Gamma, \varphi \otimes \phi \vdash \psi} \otimes_l \\ \frac{\Gamma \vdash \varphi}{\Gamma \vdash \forall x \cdot \varphi} \forall_r \quad (*) \quad \frac{\Gamma, \varphi \vdash \psi}{\Gamma, \forall x \cdot \varphi \vdash \psi} \forall_l \end{array}$$

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

Figure 2: First-Order Multiplicative Intuitionistic Linear Logic

4.2 Translation of a Colored Petri Net to MILL1

For a given CPN, the translation is performed as follows:

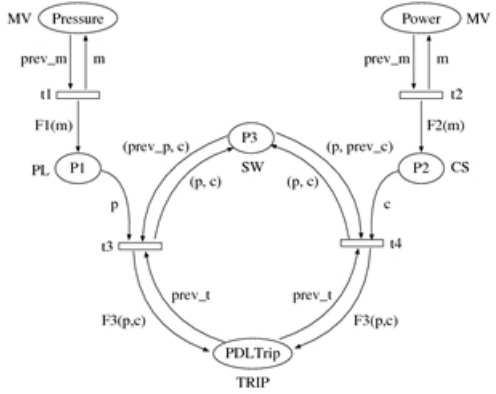
- A unary atomic predicate R is associated with each place p of the CPN,
- A single sequent using the multiplicative conjunction TIMES (\otimes), is associated with each marking, pre-condition and post-condition of transition,
- To each transition t of the net an implicative formula is defined as follows:

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

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 Fig. 3 to its equivalent in MILL1.

- places p_i are translated by unary atomic predicates $x \mapsto R_i(x)$ where ($1 \leq i \leq 5$);
- the transition t_1 is translated by the formula: $\varphi_1 \hat{=} \forall x \cdot \left(R_1(x) \multimap R_2(f(x)) \otimes R_4(g(x)) \right)$
- transition t_2 is translated by the formula: $\varphi_2 \hat{=} \forall y \cdot \forall z \cdot \left(R_2(y) \otimes R_5(z) \multimap R_3(h(y, z)) \right)$



```

color MV = int with 0..5000; (* Pressure and power *)
color PL = low | normal | high; (* Pressure level *)
color CL = enable | disable; (* Conditioning state *)
color SW = product PL * CS; (* Software state *)
color TRIP = open | close; (* PDL trip *)

var m, prev_m : MV;
var p, prev_p : PL;
var c, prev_c : CS;
var e, prev_e : TRIP;

(* Determine pressure level *)
fun F1(m) = if p ≤ 1287 then low
           else if p ≥ 4810 then high
           else normal;
(* Determine conditioning state *)
fun F2(m) = if m < 2739 then disable
           else enable;
(* Determine PDL trip parameter *)
fun F3(p, c) = if (p = low or p = high) and c = enable
              then open
              else close;

```

Figure 5: CPN for PDL trip parameter

- Determine the PDL trip parameter: If the trip conditioning is enabled and ($\Delta P < 1287mV$ or $\Delta P > 4810mV$), open PDL trip parameter. Otherwise, close PDL trip parameter.

Let's consider the state vector $(Pressure, Power, P1, P2, P3, PDLtrip)$. The state $(x, y, ?, ?, ?, z)$ ¹ is a failure state if $x \leq 1287$ or $x \geq 4810$, $y \geq 2739$ and $z = close$

The CPN inversion can be performed by applying a parallel transformation for transitions $t1, t2$ and a parameterized parallel one for $t3, t4$ (see Fig.6)

Transitions $t1$ and $t2$: The firing of these transitions corresponds to an update of *Pressure* and *Power* tokens values. Old values are deleted and replaced by those of the variable m . New token values in $P1$ and $P2$ are calculated using functions $F1$ and $F2$ respectively. The inversion of these transitions exploits values produced by $F1$ and $F2$ to find backward those in *Pressure* and *Power* using the functions $F1inverse(p)$ and $F2inverse(c)$.

Transitions $t3$ and $t4$: These transitions are in the same time parallel and parametric. Parallel because the output arcs expressions are functions of same input variables. Parametric cause of the multi variable function $F3$. The inversion displays guards $[t =$

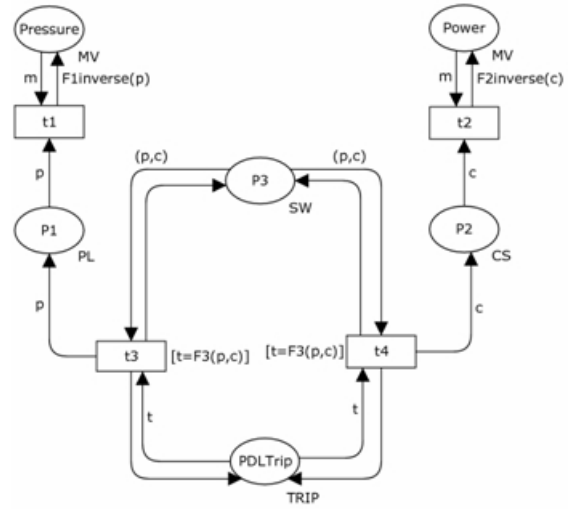


Figure 6: inverse CPN for PDL trip parameter

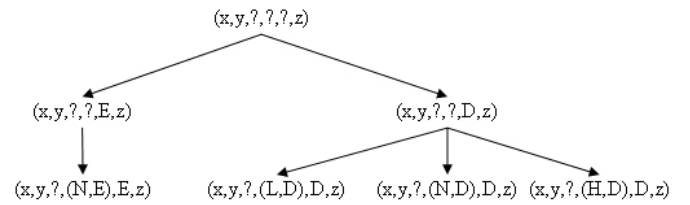


Figure 7: First inverse reachability tree for PDL trip parameter

$F3(p, c)$. The inversion of the function $F3$ is different in $t3$ and $t4$. Around $t3$, the variable c is constant, on the other hand, around $t4$ the variable p is constant.

Backward analysis: In this paragraph, same results are obtained using both structural backward reachability analysis (inverse CPN) and linear logic (sequent proofs). The structure of the inverse CPN is translated in MILL1 as described in the section 4. Then proof trees are calculated.

Let's consider, in the inverse CPN, the same state vector as the original CPN: $(Pressure, Power, P1, P2, P3, PDLtrip)$. The failure marking is $(x, y, ?, ?, ?, z)$ where $x \leq 1287$ or $x \geq 4810$, $y \geq 2739$ and $z = close$.

The marking enhancement consists in definition of possible values of unknown tokens (in $P1, P2, P3$) starting by *conditioning state* represented by the token value in $P2$ which corresponds also to the value of the variable c in the schema. The set of possible colors in this case (in $P2$) is limited, by definition, to two values: *Enable* and *Disable* (noted E and D respectively). Corresponding values for the variable p are calculated toward the parameterized inverse of the function $F3$ which inputs are z and c . Results are shown in the tree Fig.7.

Another manner to enhance the marking is to define possible values of unknown tokens (in $P1, P2, P3$) starting by *Pressure level* represented by the to-

¹the character "?" stands for a missing information

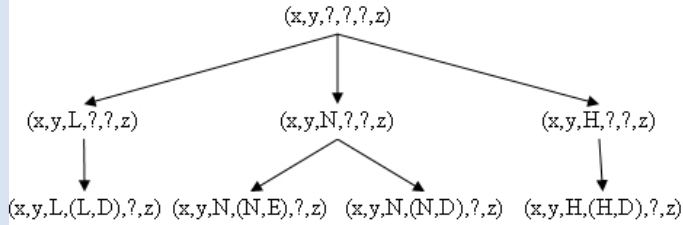


Figure 8: Second inverse reachability tree for PDL trip parameter

ken value in $P1$ which corresponds also to the value of the variable p in the schema. The set of possible colors in this case (in $P1$) is limited, by definition, to three values: *Low*, *Normal* and *High* (noted L , N and H respectively). Corresponding values for the variable c are calculated toward the parameterized inverse of the function $F3$ which inputs are z and p . Results are shown in the tree Fig.8.

6 Conclusion

This paper presents the application of the linear logic (LL) to express reachability, and especially, backward reachability in Colored Petri Nets. To do, the equivalence between LL and Petri Nets is exploited. The LL fragment used is the Multiplicative Intuitionistic Linear Logic (MILL). This equivalence is extended to CPN. The conversion between CPN and MILL must respect properties of CPN, i.e. tokens are from of a certain type and they are transformed using arcs expressions. The LL fragment in which these constraints can be expressed is the MILL1. The reachability in CPN is expressed in LL by sequents which can be proven and proof trees can be constructed.

The main application field for this work is the formal model diagnostics during the conception. The system initial state is usually well determined and so is the final state which in this case represents an undesired state. The proposed method can provide answers to the possibility of undesired event occurrence, the earliest failure time, the system evolution vector, etc.

The perspectives start with the development of an automated tool for a verification of the CPN structure in order to be acceptable for the backward reachability analysis. The theory advances include the generalization of transformation rules to composed functions. The second path to explore is the possible of presence of non deterministic time constraints for the transition firing. The study of stochastic firing vectors is also considered as a potential subject of interest.

REFERENCES

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

F.Girault (1997). *Formalisation en Logique Linéaire du fonctionnement des réseaux de Petri*. Ph. D. thesis, Université Paul Sabatier, Toulouse.

G.Gentzen (1969). *The Collected Works of Gerhard Gentzen*. Amsterdam: North-Holland Publishing Company.

Girard, J. (1987). Linear logic. *Theoretical Computer Sciences* 50, 1–102.

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

Jensen, K. and G. Rozenberg (1991). *High-level Petri Nets : Theory and Application*. Germany: Springer-Verlag.

Khalifaoui, S. (2003, Septembre). *Méthode de recherche des scénarios redouts pour l'évaluation de la sûreté de fonctionnement des systèmes mécatroniques du monde automobile*. Ph. D. thesis, Institut National polytechnique de Toulouse (France).

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

M.Bouali, P.Barger, and W.Schon (2009a, June). Colored petri nets inversion for backward reachability analysis. *Second IFAC Workshop on Dependable Control of Discrete Systems (DCDS'09)*, Bari, Italia.

M.Bouali, P.Barger, and W.Schon (2009b, May). Towards an efficient structural analysis of colored petri nets. *International Conference on Industrial Engineering and Systems Management (IESM'09)*, Montreal, Canada.

Metropolis, N. and S. Ulam (1949, September). The monte carlo method. *Journal of American Statistical Association* 44(247), 335–341.

Petri, C. A. (1962). *Communication with automata*. Ph. D. thesis, Darmstadt Institut für Instrumentelle Mathematik, Bonn (Germany).