

---

# Preuves de logique linéaire et process de réseaux de Petri

**Jean Fanchon — Nicolas Rivière — Brigitte Pradin-Chézalviel — Robert Valette**

LAAS-CNRS

7, avenue du Colonel Roche

F-31077 Toulouse Cedex 4

{fanchon,nriviere,chezalvi,robert}@laas.fr

---

*RÉSUMÉ. L'équivalence entre accessibilité dans les réseaux de Petri et prouvabilité de certains séquents de logique linéaire a été montrée de différentes manières. Notre travail présenté ici raffine ce résultat, d'une part parce qu'il prend en compte les spécificités de la traduction présentée dans [PRA 99] et d'autre part parce qu'il étudie les relations entre l'ordre d'application des règles dans la preuve des séquents et les process finis associés au même problème d'accessibilité. Nous prouvons qu'à partir d'un arbre de preuve canonique, nous pouvons construire un process fini équivalent et que réciproquement, pour chaque process fini, un arbre de preuve canonique peut être construit. Une contribution de cet article est de montrer que l'efficacité du calcul des séquents en logique linéaire dans le fragment multiplicatif mis en œuvre ici peut être utilisée afin de déterminer les process finis d'un réseau de Petri quelconque.*

*ABSTRACT. Equivalence between reachability in Petri nets and provability of certain sequent of Linear logic has been proven in various ways. Our contribution refines this result in two ways. It considers the specific translation introduced in [PRA 99] and it studies the relationships between the order of the rule application in the sequent proof and the finite processes associated with the same reachability problem. We prove that from a canonical proof tree, an "equivalent" finite process can be derived and reciprocally that for each finite process, an "equivalent" canonical proof tree can be constructed. One of the contributions of this paper is that the effectiveness of linear logic sequent calculus in the multiplicative fragment used here can be used for determining the finite processes of any Petri net.*

*MOTS-CLÉS : Process, Réseau de Petri, Logique linéaire, Ordres partiels, Arbres de preuves*

*KEYWORDS: Process, Petri net, Linear logic, Proof trees, Partial orders*

---

## 1. Introduction

Différentes traductions d'un réseau de Petri en logique linéaire [GIR 87] ont été proposées. Dans certaines d'entre elles ([MAR 89, GUN 89]), les transitions sont considérées comme des axiomes à ajouter à la logique linéaire. D'importantes propriétés de la logique linéaire telles que l'élimination des coupures ("cut rule") sont ainsi perdues. Toutefois, c'est dans ce contexte que les premières relations entre des preuves de logique linéaire et des ordres partiels (dans le cas particulier des ordres "série/parallèle") ont été explorées [GUN 89].

Dans [BRO 89], un réseau de Petri est traduit en une unique formule de logique linéaire. De cette manière, la structure du réseau de Petri est combinée avec son marquage initial [GIR 97] et les déductions possibles en logique linéaire ne correspondent pas nécessairement à ce qui est possible. Dans [ENG 90], une approche sémantique est utilisée et les réseaux de Petri sont considérés comme des modèles de la logique linéaire. Cette approche ne s'est pas avérée très fructueuse car les propriétés des réseaux de Petri classiques ne pouvaient pas être traduites en logique linéaire.

Ce papier est basé sur une autre approche [GIR 97, PRA 99] dans laquelle les marquages sont représentés par des monômes en  $\otimes$  (les atomes logiques sont des jetons) et les franchissements de transitions par des formules implicatives ( $Pre(t) \multimap Post(t)$  avec  $Pre(t)$  et  $Post(t)$  des monômes en  $\otimes$ ). Un problème d'accessibilité entre un marquage initial  $M_0$  et un marquage final  $M_n$  à partir d'une liste de transitions  $l$  s'exprime par un séquent à prouver.

Dans cette dernière approche, aucun axiome n'est ajouté à la logique linéaire et l'élimination de la règle de coupure est préservée. L'équivalence entre accessibilité et prouvabilité reste vraie [GIR 97] et les arbres de preuves peuvent être construits de manière canonique en appliquant de façon itérative la règle d'élimination de l'implication linéaire à gauche ("règle  $\multimap_L$ ") [PRA 99, AND 92] (une application de la règle pour chaque élément de la liste  $l$ ). Cette règle est assimilée à un franchissement de transition. L'implication linéaire exprime la relation de causalité entre les jetons consommés et produits. Chaque jeton produit puis consommé pendant la preuve correspond à une relation de précedence entre deux applications de la règle  $\multimap_L$  dans la preuve (celle qui a produit le jeton et celle qui l'a consommé). Dans [PRA 99], ces relations de précedence sont interprétées comme des précédences entre les franchissements des transitions correspondantes afin de calculer des durées de scénario. Dans [MAN 02] nous avons montré que l'ensemble des relations de précedence déduit de l'arbre de preuve canonique définit un ordre partiel parmi l'ensemble des tirs de transitions et que l'on pouvait en déduire un graphe de contraintes temporelles quand des durées étaient associées aux places ou aux transitions.

D'un autre côté, les dépliages et les process de réseaux de Petri constituent une des principales sémantiques *réellement* concurrentes des réseaux de Petri ([NIE 81, GOL 83, BES 87, MES 96, HOO 96, ESP 96]). Les process et les dépliages d'un réseau de Petri pour un marquage initial donné, sont des graphes acycliques, finis ou infinis composés de places et de transitions. En fait, dans de tels réseaux dénommés

*réseaux d'occurrence*, les places symbolisent les jetons et les transitions symbolisent les franchissements de transitions. L'unique transition d'entrée d'une place correspond au tir qui a produit le jeton correspondant. Un process ne contient pas de conflit, chaque place a une seule transition de sortie représentant le tir qui a consommé le jeton correspondant.

Dans cet article nous prouvons que, à partir d'un arbre de preuve canonique en logique linéaire on peut construire un process fini d'un réseau de Petri. Réciproquement, à partir de chaque process fini entre un marquage initial et un marquage final, un arbre de preuve canonique d'un séquent composé des marquages correspondant et de la liste des transitions franchies peut être construit. Ainsi la décidabilité et l'efficacité du calcul des séquents en logique linéaire dans le fragment multiplicatif utilisé ici ([GIR 87]) peuvent être utilisés pour déterminer les process finis de n'importe quel réseau de Petri (borné ou non par exemple).

L'article possède la structure suivante. La section 2 détaille comment les problèmes d'accessibilité dans les réseaux de Petri sont traduits sous la forme de preuves de séquent de logique linéaire et comment les arbres de preuves canoniques sont construits. Dans la section 3 nous rappelons comment les process sont obtenus à partir d'un réseau de Petri et de son marquage initial, et nous présentons deux opérations sur les process qui seront utilisées dans les sections 4 et 5. La section 4 montre comment l'ensemble des relations de précédence obtenues à partir d'un arbre de preuve donne un process fini entre les marquages initial et final du séquent. Réciproquement, la section 5 démontre comment obtenir, à partir de tout process fini, un arbre de preuve prouvant le séquent composé du marquage initial, du marquage final et d'une liste de franchissements de transitions correspondant aux transitions du process. Nous terminons par une discussion des résultats et en donnant quelques perspectives de recherche.

## 2. Accessibilité et logique linéaire

Dans cette section, nous donnons les notations utilisées avant de présenter la traduction d'un problème d'accessibilité sous forme d'une preuve de séquent en logique linéaire. Enfin, nous présentons l'arbre de preuve canonique.

### 2.1. Notations

Un réseau de Petri  $N$  ([REI 98]) est un triplet  $(S, T, W)$  où  $S$  est un ensemble fini de places,  $T$  est un ensemble fini de transitions, tels que  $S \cap T = \emptyset$ , et  $W : \{(S \times T) \cup (T \times S)\} \rightarrow \mathbb{N}$  est la matrice d'incidence.  $X = S \cup T$  est l'ensemble des éléments de  $N$ . L'ensemble des éléments d'entrée d'un noeud  $x \in X$  (noté  $\bullet x$ ) est l'ensemble  $\{y \in X, W(y, x) \neq 0\}$  et l'ensemble des éléments de sortie de  $x$  (noté  $x^\bullet$ ) est l'ensemble  $\{y \in X, W(x, y) \neq 0\}$ . L'ensemble  $\bullet x \cup x^\bullet$  s'écrit  $\bullet x^\bullet$ . Un marquage  $M$  de  $N$  est un multi-ensemble  $M : S \rightarrow \mathbb{N}$ .  $Pre(t)$  et  $Post(t)$  sont des ensembles de places tels que  $\forall s, t : Pre(t)(s) = W(s, t)$  et  $Post(t)(s) = W(t, s)$ . Soit deux multi-ensembles de places  $M_1$  et  $M_2$ , nous notons  $M_1 \sqsubseteq M_2$  la propriété

$\forall s \in S, M_1(s) \leq M_2(s)$ . Nous notons  $M[\sigma > M']$  l'accessibilité de  $M'$  à partir de  $M$  en exécutant la séquence  $\sigma \in T^*$  (les éléments de  $\sigma$  étant ordonnés).

## 2.2. Traduction en logique linéaire

Le fragment multiplicatif de la logique linéaire intuitioniste (MILL) est suffisant pour cette approche. Il ne contient que le connecteur multiplicatif " $\otimes$ " (conjonction des hypothèses) et l'implication linéaire " $\multimap$ ". Il n'y a pas de négation et le meta-connecteur " $,$ " est commutatif. La spécificité de la logique linéaire (par rapport à la logique classique) est que les propositions logiques sont consommées pendant une déduction. Prouver un séquent revient à vérifier que les hypothèses requises sont disponibles lorsqu'elles sont utilisées lors d'une étape d'une preuve.

Les atomes symbolisent des jetons et leurs noms sont les places où elles se trouvent. Les marquages sont représentés par des monômes en  $\otimes$ . Les franchissements de transitions sont représentés par des formules de la forme suivante :

$$t : Pre(t) \multimap Post(t) \quad [1]$$

où  $Pre(t)$  et  $Post(t)$  sont des monômes en  $\otimes$ , comme les marquages.

Une preuve d'accessibilité s'exprime par le séquent suivant :

$$M_0, l_0 \vdash M_n \quad [2]$$

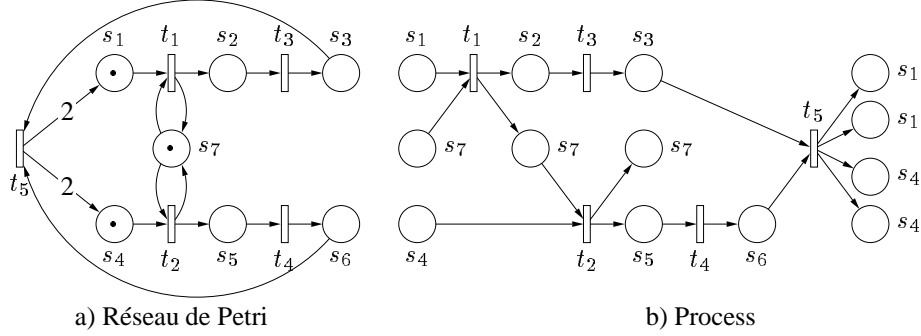
où  $M_0$  est le marquage initial,  $M_n$  le marquage final et  $l_0$  est une liste non ordonnée de formules représentant des tirs de transitions sous la forme 1. Ces formules représentent des tirs car si une transition  $t$  est franchie  $n$  fois pour atteindre un marquage alors la formule correspondante doit être présente  $n$  fois dans  $l_0$ . Cette liste est un bloc de formules reliées par le meta-connecteur " $,$ ".

Le fait que la preuve d'un séquent 2 prouve l'accessibilité de  $M_n$  depuis  $M_0$  est une conséquence de l'équivalence entre prouvabilité et accessibilité [GIR 97]. Le travail présenté dans cet article peut être vu comme un raffinement de ce résultat parce qu'il est prouvé qu'en plus de l'accessibilité, nous pouvons construire les process existant entre les deux marquages.

## 2.3. Exemple

Soit le fragment de réseau de Petri de la figure 1. Les franchissements de transitions sont notés comme suit :

$$\begin{array}{ll} t_1 : s_1 \otimes s_7 \multimap s_2 \otimes s_7 & t_3 : s_2 \multimap s_3 \\ t_2 : s_4 \otimes s_7 \multimap s_5 \otimes s_7 & t_4 : s_5 \multimap s_6 \\ t_5 : s_3 \otimes s_6 \multimap s_1 \otimes s_1 \otimes s_4 \otimes s_4 & \end{array} \quad [3]$$



**Figure 1.** Exemple de process de réseau de Petri

L'accessibilité d'un marquage avec deux jetons dans chacune des places  $s_1$  et  $s_4$  et un jeton dans la place  $s_7$  à partir d'un marquage avec un jeton dans  $s_1$ ,  $s_4$  et  $s_7$  en tirant une seule fois chacune des transitions  $t_1, t_2, t_3, t_4$  et  $t_5$  s'écrit :

$$\underbrace{s_1 \otimes s_4 \otimes s_7}_{M_0}, \underbrace{t_1, t_2, t_3, t_4, t_5}_{l_0} \vdash \underbrace{s_1 \otimes s_1 \otimes s_4 \otimes s_4 \otimes s_7}_{M_n} \quad [4]$$

#### 2.4. Preuve d'un séquent

Un arbre de preuve est une preuve syntaxique et donc une suite d'applications de règles. Chaque application prouve qu'un connecteur a correctement été introduit. L'arbre est lu de bas en haut ; le bas étant le séquent à prouver. Les feuilles qui terminent les différentes branches de l'arbre sont des identités. Dans le fragment MILL, comme la règle de coupure peut être éliminée, il est toujours possible de réécrire un arbre sous la forme d'un autre arbre de preuve qui n'utilise que les règles d'élimination (éliminer c'est montrer que le connecteur a été correctement introduit) des connecteurs  $\otimes$  et  $\multimap$  avec, bien sûr la règle Identité. De plus, comme la partie droite de nos séquents ne comporte aucune implication linéaire (car c'est un marquage), nous n'utilisons que la règle d'introduction à gauche de cette implication.

Soit  $A$  un atome,  $F, G$  et  $H$  des formules,  $\Gamma$  et  $\Delta$  des blocs de formules (connectés par ","). Les règles nécessaires aux preuves sont :

$$\frac{}{A \vdash A} \text{id} \quad [5]$$

$$\frac{\Gamma \vdash F \quad \Delta, G \vdash H}{\Gamma, \Delta, F \multimap G \vdash H} \multimap_L \quad [6]$$

$$\frac{\Gamma \vdash F \quad \Delta \vdash G}{\Gamma, \Delta \vdash F \otimes G} \otimes_R \quad [7]$$

$$\frac{\Gamma, F, G \vdash \Delta}{\Gamma, F \otimes G \vdash \Delta} \otimes_L \quad [8]$$

Soit  $\mathcal{M} = S_1, S_2, \dots, S_n$  une liste d'atomes (jetons portant des noms de places). Nous définissons  $\eta(\mathcal{M})$  comme le multi-ensemble d'atomes  $M = \eta(\mathcal{M})$  avec  $M(s) = |\{i \in [1, n] : S_i = s\}|$ .

### 2.5. Arbre de preuve canonique

Un séquent peut être prouvé par plusieurs arbres de preuves. Les séquents d'accessibilité ayant une forme spécifique, nous avons utilisé une méthode canonique pour construire les arbres de preuves.

#### 2.5.1. Étape initiale

##### Principe

Démarrant avec un séquent de la forme 2, l'introduction à gauche du  $\otimes$  (règle 8) est appliquée de manière itérative jusqu'à ce que le marquage  $M_0$  soit transformé en une liste d'atomes séparés par des virgules. D'un point de vue logique, cela signifie que les atomes peuvent être utilisés indépendamment dans les règles. Soit  $\mathcal{M}_0$  cette liste qui n'est plus une formule mais un bloc d'atomes. Nous avons  $M_0 = \eta(\mathcal{M}_0)$  et le séquent obtenu est  $\mathcal{M}_0, l_0 \vdash M_n$ .

##### Exemple

Prenons le cas du séquent 4 :

$$\mathcal{M}_0 = s_1, s_4, s_7, \quad l_0 = t_1, t_2, t_3, t_4, t_5 \quad \text{et} \quad M_n = s_1 \otimes s_1 \otimes s_4 \otimes s_4 \otimes s_7 .$$

Nous avons :

$$\frac{\frac{s_1, s_4, s_7, t_1, t_2, t_3, t_4, t_5 \vdash s_1 \otimes s_1 \otimes s_4 \otimes s_4 \otimes s_7}{s_1, s_4 \otimes s_7, t_1, t_2, t_3, t_4, t_5 \vdash s_1 \otimes s_1 \otimes s_4 \otimes s_4 \otimes s_7} \otimes_L}{s_1 \otimes s_4 \otimes s_7, t_1, t_2, t_3, t_4, t_5 \vdash s_1 \otimes s_1 \otimes s_4 \otimes s_4 \otimes s_7} \otimes_L \quad [9]$$

#### 2.5.2. Étape itérative

##### Principe

Cette étape doit être exécutée une et une seule fois par franchissement de transition de la liste  $l$ . Elle est exécutée pour les séquents de la même forme que 2 après l'étape initiale qui est :

$$\mathcal{M}_i, l_i \vdash M_n \quad [10]$$

où  $\mathcal{M}_i$  est une liste d'atomes séparés par des virgules,  $l_i$  est une liste de tirs de transitions et  $M_n$  le marquage final. Elle est décomposée en trois sous-étapes (ou pas itératifs).

**Premier pas** Le premier pas est l'application de l'introduction à gauche de  $\multimap$  (règle 6) pour le tir de transition choisi  $t$ . Le séquent courant de la preuve est ainsi

réécrit sous la forme de deux séquents. Nous avons  $F = Pre(t)$  et  $G = Post(t)$  dans la règle 6. Pour l'arbre canonique, nous nous restreignons au cas où  $\Gamma$  est une sous-liste de  $\mathcal{M}_i$ . Étant donné qu'un séquent qui est mal équilibré du point de vue des atomes ne peut pas être prouvé, une preuve ne peut être continuée que si  $\Gamma$  contient exactement les atomes de  $Pre(t)$  avec la même multiplicité : ceci est une condition nécessaire et suffisante. En conséquence,  $Pre(t) \sqsubseteq \mathcal{M}_i$  et  $\eta(\Gamma) \sqsubseteq Pre(t)$ . Le bloc  $\Delta$  est composé du reste de  $\mathcal{M}_i$  concaténé avec le reste de  $l_i$  après avoir éliminé la formule correspondant à  $t$ .

Considérons le franchissement de la transition  $t$  et posons  $\mathcal{M}_i = \Gamma, \mathcal{M}'_i$  et  $l_i = Pre(t) \multimap Post(t), l_{i+1}$ , nous obtenons la construction suivante :

$$\frac{\Gamma \vdash Pre(t) \quad \mathcal{M}'_i, Post(t), l_{i+1} \vdash M_n}{\Gamma, \mathcal{M}'_i, Pre(t) \multimap Post(t), l_{i+1} \vdash M_n} \multimap_L \quad [11]$$

Notons que  $\mathcal{M}_i, \mathcal{M}'_i, \Gamma$  et  $l_{i+1}$  sont des blocs et  $Pre(t), Post(t)$  et  $M_n$  sont des formules.

**Second pas** Ce second pas consiste à appliquer la règle 8 ( $\otimes_L$ ) itérativement sur la formule  $G = Post(t)$  afin de la transformer en une liste d'atomes. Le séquent obtenu est alors de la forme de 10.  $\mathcal{M}_{i+1}$  est obtenu à partir de  $\mathcal{M}_i$  en y enlevant les atomes de  $Pre(t)$  et en y ajoutant les atomes de  $Post(t)$ .  $l_{i+1}$  est obtenu à partir de  $l_i$  en enlevant la formule correspondant à  $t$ . La partie droite du séquent ( $M_n$ ) reste inchangée.

**Troisième pas** Le troisième pas consiste à appliquer la règle 7 ( $\otimes_R$ ) itérativement sur la formule  $F = Pre(t)$  afin d'obtenir autant de séquents identités qu'il y a d'atomes dans  $F$ . Nous appliquons alors la règle 5 identité pour produire les feuilles de l'arbre de preuve.

### Exemple

Prenons la preuve du séquent 4. Si nous considérons le franchissement de la transition  $t_1$ , nous avons le fragment d'arbre de preuve suivant :

$$\frac{\frac{\frac{}{s_1 \vdash s_1} \text{id} \quad \frac{}{s_7 \vdash s_7} \text{id}}{s_1, s_7 \vdash s_1 \otimes s_7} \otimes_R \quad \frac{s_2, s_4, s_7, t_2, t_3, t_4, t_5 \vdash M_n}{s_4, s_2 \otimes s_7, t_2, t_3, t_4, t_5 \vdash M_n} \otimes_L}{s_1, s_4, s_7, \underbrace{s_1 \otimes s_7 \multimap s_2 \otimes s_7}_{t_1}, t_2, t_3, t_4, t_5 \vdash M_n} \multimap_L} \quad [12]$$

### 2.5.3. Étape finale

#### Principe

L'étape finale qui débute lorsque la liste  $l_i$  est vide, consiste à appliquer itérativement la règle  $\otimes_R$  dans le but de séparer le séquent en deux ensembles d'identités afin de pouvoir appliquer la règle identité. Cette étape n'est possible que si la dernière liste de jetons  $\mathcal{M}_i$  est composée des mêmes jetons que ceux qui composent le marquage final  $M_n$ . Si ce n'est pas le cas, la preuve échoue ; une preuve qui échoue ne signifie pas forcément que le séquent n'est pas prouvable.

#### Exemple

Prenons de nouveau la preuve du séquent 4, on obtient le fragment de preuve suivant :

$$\frac{\frac{\frac{}{s_1 \vdash s_1} \text{id} \quad \frac{\frac{}{s_1 \vdash s_1} \text{id} \quad \frac{\dots \quad \dots}{s_4, s_4, s_7 \vdash s_4 \otimes s_4 \otimes s_7} \otimes_R}{s_1, s_4, s_4, s_7 \vdash s_1 \otimes s_4 \otimes s_4 \otimes s_7} \otimes_R}{s_1, s_1, s_4, s_4, s_7 \vdash s_1 \otimes s_1 \otimes s_4 \otimes s_4 \otimes s_7} \otimes_R}{[13]}$$

## 3. Process de réseaux de Petri

### 3.1. Process et réseaux causaux

Un réseau causal [REI 98] ou réseau d'occurrences déterministe [NIE 81, BES 87] est un réseau de Petri (étendu, car éventuellement de taille infinie) acyclique *sauf* (1-borné) où chaque place possède au plus une transition en entrée et en sortie. Un process de réseau de Petri  $N$  est un réseau causal  $O$  associé à un morphisme de graphes biparties de  $O$  vers  $N$ , *i.e.* une application des places de  $O$  vers les places de  $N$ , des transitions de  $O$  vers les transitions de  $N$ , et satisfaisant quelques propriétés additionnelles. Dans notre travail, nous ne considérerons que les process *finis*. Formellement, les réseaux causaux et les process sont définis comme suit :

Un **réseau causal** est un réseau de Petri  $O = (B, E, F)$  tel que :

- 1)  $F : B \times E \cup E \times B \longrightarrow \{0, 1\}$  : les transitions ne peuvent produire (resp. consommer) que des jetons uniques.
- 2)  $\forall e \in E, \bullet e \neq \emptyset \neq e^\bullet$ , *i.e.* toute transition possède au moins une place d'entrée et une place de sortie.
- 3)  $\forall b \in B, |\bullet b| \leq 1$  et  $\forall b \in B, |b^\bullet| \leq 1$  : les places sont des entrées et des sorties d'au plus une seule transition.
- 4)  $F^+$  est acyclique où  $F^+ \subseteq B \cup E \times B \cup E$  est la fermeture transitive de  $F$ .

Un **process** de réseau de Petri  $N = (S, T, W)$  est une paire  $(O, \lambda)$  où  $O = (B, E, F)$  est un réseau causal et  $\lambda$  est une application  $\lambda : B \cup E \longrightarrow S \cup T$  telle que :

1)  $\lambda(B) \subseteq S, \lambda(E) \subseteq T$  :  $\lambda$  associe les transitions (resp. les places) de  $O$  aux transitions (resp. aux places) de  $N$ .

2)  $\forall e \in E : \lambda(\bullet e) = \bullet \lambda(e), \lambda(e\bullet) = \lambda(e)\bullet$  :  $\lambda$  préserve l'ensemble des places d'entrée et l'ensemble des places de sortie des transitions.

3)  $\forall e \in E, \forall s \in S : W(s, \lambda(e)) = |\lambda^{-1}(s) \cap \bullet e| \wedge W(\lambda(e), s) = |\lambda^{-1}(s) \cap e\bullet|$  :  $\lambda$  préserve les arités d'entrée et de sortie des transitions.

4) Soit  $M_{in}$  un marquage de  $N$ , alors  $(O, \lambda)$  est un process du réseau marqué  $(N, M_{in})$  seulement si  $\forall s \in S, M_{in}(s) = |\{b \in B : \bullet b = \emptyset \wedge \lambda(b) = s\}|$ . Cela signifie que les places initiales de  $O$  correspondent de manière bijective aux jetons du marquage initial  $M_{in}$ .

Par la suite,  $(O, \lambda)$  est un process de réseau marqué  $(N, M_{in})$  avec  $N = (S, T, W)$  et  $O = (B, E, F)$ . Une transition  $e \in E$  est appelé un **événement** et peut être vu comme (une occurrence de) un tir de la transition  $\lambda(e) \in T$ . Une place  $b \in B$  modélise un jeton dans la place  $\lambda(b) \in S$ .

#### Remarque 1

Un process  $((B, E, F), \lambda)$  est défini "à isomorphisme près" : les ensembles  $B$  et  $E$  des places et des transitions peuvent être substitués par n'importe quels ensembles  $B'$  et  $E'$  d'intersection vide au moyen de bijections  $B \rightarrow B'$  et  $E \rightarrow E'$ . Ces bijections induisent de façon unique une relation  $F'$  image de  $F$  et un morphisme  $\lambda'$  compatible avec le morphisme  $\lambda$ . Le process  $((B', E', F'), \lambda')$  obtenu est le même que celui de départ "à isomorphisme près". En particulier dans une représentation canonique d'un process,  $B$  pourrait être défini par  $e\bullet = \cup_{s \in t\bullet} \{(e, s)\} \times [W(t, s)]$ , pour tout  $e \in E$  avec  $\lambda(e) = t$ . Pour tout entier  $k$  on note  $[k]$  l'ensemble  $[k] = \{1, 2, \dots, k\}$ .

#### B-coupes et marquages accessibles

Soit  $F^*$  la fermeture transitive-réflexive de  $F$ . Comme  $F^+$  est acyclique,  $F^*$  (resp.  $F^+$ ) est un ordre partiel (resp. un ordre partiel strict) que nous notons  $\leq_F$  (resp.  $<_F$ ). Une B-coupe est un ensemble de places  $C \subseteq B$  tel que deux places de  $C$  sont incomparables pour  $\leq_F$  (propriété Cut1 ci-dessous) et est un sous-ensemble maximal de  $B$  pour cette propriété (propriété Cut2). On note par  $Cut(O)$  l'ensemble des B-coupes de  $O$  et ainsi si  $C \subseteq B$  :

$$C \in Cut(O) \iff \begin{cases} \mathbf{Cut1} : b, b' \in C \implies \neg(b <_F b' \vee b' <_F b) \\ \mathbf{Cut2} : b \notin C \implies \exists b' \in C : (b <_F b' \vee b' <_F b) \end{cases}$$

Un ensemble de places qui ne satisfait que la condition 1 est appelé une **anti-chaîne**.

Une place sans transition d'entrée (resp. transition de sortie) est appelé minimale (resp. maximale). Les ensembles des places minimales et maximales sont notés  $Min(O)$  et  $Max(O)$  :  $Min(O) = \{b, \bullet b = \emptyset\}$ ,  $Max(O) = \{b, b\bullet = \emptyset\}$ .  $Min(O)$

et  $Max(O)$  sont des exemples de B-coupes :  $Min(O) \in Cut(O)$  et  $Max(O) \in Cut(O)$ .

Le passé d'une B-coupe  $C$  est le réseau d'occurrences  $\downarrow C = (B_C, E_C, F_C)$  avec  $B_C = \{b \in B \mid \exists b' \in C, b \leq_F b'\}$ ,  $E_C = \{e \in E \mid \exists b \in C, e \leq_F b\}$  et  $F_C = F /_{B_C \cup E_C}$ . Avec l'application  $\lambda_C = \lambda /_{B_C \cup E_C}$ , la paire  $(\downarrow C, \lambda_C)$  est un process de  $N$ .

### Ordre partiel étiqueté associé à un process

A chaque process  $(O, \lambda)$  nous associons l'ordre partiel étiqueté par les transitions  $P(O, \lambda) = (E, \leq_F /_{E \times E}, \lambda /_E)$  induit par les restrictions de  $\leq_F$  et de  $\lambda$  à l'ensemble des événements. Nous notons aussi cet ordre partiel par  $P(O)$  lorsqu'il n'y a aucune ambiguïté. L'ensemble des extensions linéaires (ou linéarisations) de  $P(O, \lambda)$  est noté  $Lin(P(O, \lambda))$ , formellement  $Lin(P(O, \lambda))$  est l'ensemble des mots de  $T$  défini par :

$$Lin(P(O, \lambda)) = \{\lambda(e_1) \dots \lambda(e_n) \in T^*, E = \{e_1, \dots, e_n\} \wedge (e_i <_F e_j \implies i < j)\}$$

### Marquages associés aux B-coupes

Soit  $C \in Cut(O)$ , nous associons à  $C$  un marquage de  $N$  (un multi-ensemble de places de  $N$ ) noté  $\mu(C)$ , avec  $\mu(C) : S \rightarrow \mathbb{N}$ , défini par :

$$\mu(C)(s) = |\lambda^{-1}(s) \cap C|$$

Cela signifie que les places de  $\lambda^{-1}(s) \cap C$  correspondent de manière bijective aux jetons de  $\mu(C)$  dans la place  $s$ .

La condition 4 de la définition d'un process, qui dit que  $M_{in}(s) = |\{b \in Min(O) : \lambda(b) = s\}|$ , peut être reformulé par  $\mu(Min(O)) = M_{in}$ . Notons que le multi-ensemble  $\mu(D)$  peut être défini pour tout sous-ensemble  $D \subset B$  même si  $D$  n'est pas une coupe, en particulier si  $D$  est une antichaîne de  $O$ .

Une propriété importante des B-coupes est que les marquages associés sont accessibles dans le réseau de Petri : si  $(O, \lambda)$  est un process de  $(N, M_{in})$  et que  $C \in Cut(O)$  alors  $\mu(C)$  est accessible dans  $(N, M_{in})$  et pour tout  $\sigma \in Lin(P(\downarrow C))$  nous avons  $M_{in}[\sigma > \mu(C)]$ .

$$\forall C \in Cut(O), \forall \sigma \in Lin(P(\downarrow C)) : M_{in}[\sigma > \mu(C)]$$

Tout marquage accessible d'un réseau de Petri et toute séquence de tir y menant peuvent également être extraits d'un process. Nous utilisons la propriété générale associant des séquences de tirs à des process [BES 87] :

$$\begin{aligned} \forall \sigma \in T^* : M[\sigma > M'] &\iff \\ \exists (O, \lambda) : \mu(Min(O)) = M, \mu(Max(O)) = M', \sigma &\in Lin(P(O, \lambda)) \end{aligned}$$

qui signifie que les séquences de tirs correspondent à des linéarisations de process. D'autres propriétés importantes des process les lient aux séquences de pas et aux franchissements d'ordres partiels (voir [BES 87]).

### 3.2. Ajouter une occurrence de transition à un process

En associant un process à un arbre de preuve dans la section 4, chaque pas dans la construction d'un arbre de preuve sera modélisé en ajoutant au process courant une transition comme suit.

#### Définition

Soit  $t \in T$  et  $D \subseteq \text{Max}(O)$  un sous-ensemble de places maximales tel que  $\mu(D) = \text{Pre}(t)$ , ou de manière équivalente  $\lambda(D) = \bullet t \wedge (\forall s \in \bullet t, |\lambda^{-1}(s) \cap D| = W(s, t))$ . Dans ce cas, nous pouvons "ajouter une occurrence de  $t$  à  $(O, \lambda)$  après  $D$ ", ce qui donne un nouveau process de  $N$  noté  $(O', \lambda') = (O, \lambda).(D, t)$  où  $O' = (B', E', F')$  et  $\lambda'$  sont définis par :

- 1)  $E' = E \uplus \{e\}$ , où  $e$  est un nouvel événement ( $\uplus$  représente l'union disjointe).
- 2)  $B' = B \uplus B_e$  avec une bijection  $r : B_e \rightarrow \cup_{s \in \bullet t} \{s\} \times [W(t, s)]$
- 3)  $F' = F \cup \{(b, e), b \in D\} \cup \{(e, b), b \in B_e\}$
- 4)  $\lambda'/B \cup E = \lambda$ ,  $\lambda'(e) = t$ ,  $\lambda'(b) = s$  si  $b \in B_e$  et  $r(b) = (s, i)$ .

L'introduction de  $B_e$  et de  $r$  est un artifice qui assure que  $B \cap B_e = \emptyset$  et qui permet de définir  $\lambda'$  sur  $B_e$  dans le point 4 ci-dessus. Le lemme suivant est une conséquence directe des définitions :

**Lemme 1** *Le couple  $(O', \lambda') = (O, \lambda).(D, t)$  défini ci-dessus est un process de  $N$  qui satisfait les points suivants :*

- 1)  $B_e$  est une antichaine telle que  $\mu(B_e) = \text{Post}(t)$ .
- 2)  $\text{Max}(O') = (\text{Max}(O) - D) \uplus B_e$ .
- 3)  $\mu(\text{Max}(O')) = \mu(\text{Max}(O)) - \text{Pre}(t) + \text{Post}(t)$ .

Notons que  $\text{Min}(O') = \text{Min}(O)$ .

### 3.3. Construction itérative d'un process

Un process de réseau de Petri peut être construit en ajoutant itérativement les transitions comme défini précédemment. Ceci sera utilisé dans la section 5 pour construire un arbre de preuve à partir d'un process.

Soit  $(O, \lambda)$  un process de  $N$  où  $O = (B, E, F)$ , soit  $n = |E|$  et soit  $e_1..e_n$  une énumération de  $E$  telle que  $e_i <_F e_j \implies i < j$ . Clairement  $\sigma = \lambda(e_1).. \lambda(e_n)$  est une linéarisation de  $P(O, \lambda)$ ,  $\sigma \in \text{Lin}(P(O, \lambda))$ . Soit  $O_i = (B_i, E_i, F_i)$  ( $i = 1..n$ ) la séquence de réseaux causaux définié par  $E_i = \{e_1, \dots, e_{i-1}, e_i\}$ ,  $B_i = \bullet E_i^*$ ,  $F_i = F/B_i \cup E_i$ , et soit  $\lambda_i = \lambda/B_i \cup E_i$ , alors le lemme suivant est une conséquence directe des définitions :

**Lemme 2** *Pout tout  $i = 1..n$ ,  $(O_i, \lambda_i)$  est un process de  $N$  qui peut être défini par ajout de la transition  $\lambda(e_i)$  à  $(O_{i-1}, \lambda_{i-1})$  après  $\bullet e_i$  :*  
 $(O_i, \lambda_i) = (O_{i-1}, \lambda_{i-1}).(\bullet e_i, \lambda(e_i))$ . *Qui plus est  $(O, \lambda) = (O_n, \lambda_n)$ .*

#### 4. Des arbres de preuves canoniques aux process

Dans cette section, nous partons d'un arbre de preuve canonique et nous voulons construire un process suivant la méthode itérative de preuve.

Dans un arbre de preuve canonique  $\tau$  prouvant un séquent  $M_0, l_0 \vdash M_n$ , les étapes initiales et finales sont uniques et entièrement définies par  $M_0$  et  $M_n$  (resp. marquages initial et final). L'arbre de preuve est alors caractérisé par l'ordre dans lequel les franchissements de transitions de la liste  $l_0$  sont éliminés de manière itérative. On note  $\sigma(\tau)$  la séquence de transitions correspondante à la séquence d'éliminations dans l'arbre de preuve canonique  $\tau$ .

Nous associons un process à chaque arbre de preuve canonique  $\tau$  d'un séquent décrivant un problème d'accessibilité tel que séquent et le process ont les mêmes marquages initial et final, et tel que  $\sigma(\tau)$  est une linéarisation de l'ordre partiel étiqueté associé.

La construction se déroule comme suit : soit  $\tau$  l'arbre de preuve du séquent  $M_0, l_0 \vdash M_n$ , à chaque pas  $i$  ( $1 \leq i \leq n$ ) de la construction de l'arbre de preuve, caractérisé par un séquent courant  $\mathcal{M}_i, l_i \vdash M_n$  et un fragment courant de l'arbre  $\tau_i$ , nous définissons un process  $(O_i, \lambda_i)$  tel que  $\mu(\text{Min}(O_i)) = M_0$ ,  $\mu(\text{Max}(O_i)) = \eta(\mathcal{M}_i)$  et tel que la séquence de transitions  $\sigma(\tau_i)$  est une linéarisation de l'ordre partiel  $P(O_i, \lambda_i)$ .

##### Process initial :

Démarrons avec le séquent  $\mathcal{M}_0, l_0 \vdash M_n$ . Le process  $(O_0, \lambda_0)$  est défini par :  $O_0 = (B_0, \emptyset, \emptyset)$  avec  $B_0$  et  $\lambda_0$  tels que  $\mu(B_0) = M_0$ .

##### Pas itératif $i + 1$ :

Nous avons construit un process  $(O_i, \lambda_i)$  tel que  $\mu(\text{Min}(O_i)) = M_0$ ,  $\mu(\text{Max}(O_i)) = \eta(\mathcal{M}_i)$  et  $\sigma(\tau_i)$  est une linéarisation de l'ordre partiel étiqueté  $P(O_i, \lambda_i)$ . Soit  $t_{i+1}$  la transition éliminée au pas  $i + 1$  de la preuve. Cela signifie que  $\text{Pre}(t_{i+1}) \sqsubseteq \eta(\mathcal{M}_i)$ . Puisque  $\mu(\text{Max}(O_i)) = \eta(\mathcal{M}_i)$ , nous pouvons trouver (au moins, voir remarque 3 ci-après) une antichaîne  $C_{i+1} \subseteq \text{Max}(O_i)$ , telle que  $\mu(C_{i+1}) = \text{Pre}(t_{i+1})$ . Les hypothèses de la construction de la section 3.2 sont satisfaites et nous pouvons donc ajouter une occurrence de  $t_{i+1}$  à  $(O_i, \lambda_i)$  après  $C_{i+1}$ . Le process obtenu est le process construit au pas courant :  $(O_{i+1}, \lambda_{i+1}) = (O_i, \lambda_i).(C_{i+1}, t_{i+1})$ .

Suite au lemme 1 nous obtenons  $\mu(\text{Max}(O_{i+1})) = \mu(\text{Max}(O_i)) - \text{Pre}(t_{i+1}) + \text{Post}(t_{i+1})$ . Soit  $\mathcal{M}_{i+1}, l_{i+1} \vdash M_n$  le séquent après avoir éliminé  $t_{i+1}$ , suite à la règle 11, nous savons que  $\eta(\mathcal{M}_{i+1}) = \eta(\mathcal{M}_i) - \text{Pre}(t_{i+1}) + \text{Post}(t_{i+1})$ . Nous avons prouvé que  $\mu(\text{Max}(O_{i+1})) = \eta(\mathcal{M}_{i+1})$ . De plus  $l_{i+1}$  est déduit de  $l_i$  en enlevant un

franchissement de  $t_{i+1}$ . Soit  $\tau_{i+1}$  le nouveau fragment de l'arbre de preuve canonique, comme  $\sigma(\tau_i)$  est une linéarisation de l'ordre partiel étiqueté  $P(O_i, \lambda_i)$ ,  $\sigma(\tau_{i+1})$ , par construction  $\sigma(\tau_{i+1})$  est une linéarisation de l'ordre partiel étiqueté  $P(O_i, \lambda_{i+1})$ .

### Process final :

Après le pas  $n$ , nous avons construit un process  $(O_n, \lambda_n)$  tel que :  
 $\mu(\text{Max}(O_n)) = M_n$  et  $\sigma(\tau) \in \text{Lin}(P(O_n, \lambda_n))$ .

Nous avons donc prouvé le théorème suivant :

**Théorème 1** Soit  $M_0, l_0 \vdash M_n$  un séquent exprimant l'accessibilité d'un marquage  $M_n$  depuis un marquage  $M_0$  dans un réseau de Petri  $N$ . Soit  $\tau$  une preuve canonique de ce séquent et soit  $\sigma(\tau)$  la séquence d'étiquettes de transitions caractérisant la séquence d'élimination de tir dans l'arbre de preuve. Il est alors possible de construire au moins un process  $(O, \lambda)$  de  $N$  tel que :  
 $\mu(\text{Min}(O)) = M_0$ ,  $\mu(\text{Max}(O)) = M_n$  et  $\sigma(\tau) \in \text{Lin}(P(O, \lambda))$ .

### Remarque 2

Quand il y a plus d'un jeton dans une place, le process peut être prolongé de différentes façons qui ne sont pas toujours équivalentes. C'est pourquoi à un arbre de preuve on peut parfois associer plus d'un process.

### Exemple

Pour le réseau de Petri de la figure 1, l'arbre de preuve canonique du séquent 4 peut être prouvé en éliminant successivement (après le pas initial) les tirs de  $t_1, t_2, t_3, t_4$  et  $t_5$ . La preuve se termine par le pas final. Pour cette arbre de preuve  $\tau$ ,  $\sigma(\tau) = t_1; t_3; t_2; t_4; t_5$ , le process correspondant est représenté sur la figure 1.b.

Ce process est unique car c'est seulement pour le marquage final que certaines places contiennent plus d'un jeton. Pour chaque pas de la preuve, il n'y a aucune ambiguïté dans la construction du process : il n'y a qu'une seule façon d'ajouter une occurrence de tir de la transition considérée.

Pour le marquage initial les deux transitions  $t_1$  et  $t_2$  sont en conflit. En sélectionnant dans la preuve l'élimination du tir de  $t_1$  avant celle de  $t_2$ , ou le contraire, nous aboutissons à deux arbres de preuve canoniques caractérisés par deux séquences différentes  $\sigma$ . C'est la même chose pour les process. En construisant le process de manière itérative guidée par  $\sigma$ , nous construisons en effet les process correspondant à la même résolution de conflit.

## 5. Des process aux arbres de preuves canoniques

Dans la section précédente (4), nous avons montré qu'à partir de n'importe quel arbre de preuve canonique nous pouvions construire un process. Dans cette section,

nous démontrons l'inverse : à partir de n'importe quel process fini, nous pouvons construire un arbre de preuve canonique.

Soit  $(O, \lambda)$  un process de  $N$  où  $O = (B, E, F)$ , soient  $M_0 = \mu(\text{Min}(O))$  et  $M_n = \mu(\text{Max}(O))$ . Soient  $n = |E|$  et  $e_1..e_n$  une énumération de  $E$  telle que  $e_i <_F e_j \implies i < j$ , et soit  $(O_i, \lambda_i)$  avec  $i = 1, n$  la séquence associée des process définie dans la section 3.3. Soit  $\sigma = \lambda(e_1).. \lambda(e_n)$  ( $\sigma \in \text{Lin}(P(O, \lambda))$ ). Soit  $l_0$  la liste non ordonnée des éléments de  $\sigma$ . Nous construisons itérativement un arbre de preuve  $\tau$  du séquent  $M_0, l_0 \vdash M_n$  tel que  $\sigma(\tau) = \sigma$ .

### Fragment initial de l'arbre de preuve :

Il est formé par l'étape initiale (section 2.5.1) et il est complètement déterminé par  $\text{Min}(O)$ .

### Pas itératif $i + 1$ :

Nous avons construit les  $i$  premiers pas itératifs de l'arbre de preuve. Soit  $\tau_i$  ce fragment. Le séquent en haut de l'arbre  $\tau_i$  est  $\mathcal{M}_i, l_i \vdash M_n$ . Les hypothèses d'induction sont  $\eta(\mathcal{M}_i) = \mu(\text{Max}(O_i))$ ,  $\sigma(\tau_i) = \lambda(e_1).. \lambda(e_i)$  et  $l_i$  est la liste non ordonnée des éléments de  $\lambda(e_{i+1}).. \lambda(e_n)$ .

Suite au lemme 2, nous obtenons  $(O_{i+1}, \lambda_{i+1}) = (O_i, \lambda_i).(\bullet_{e_{i+1}}, \lambda(e_{i+1}))$ . Soit  $t_{i+1} = \lambda(e_{i+1})$ , par construction nous obtenons  $\bullet_{e_{i+1}} \subseteq \text{Max}(O_i)$  et  $\mu(\bullet_{e_{i+1}}) = \text{Pre}(t_{i+1})$ . Par conséquent  $\text{Pre}(t_{i+1}) \subseteq \mu(\text{Max}(O_i))$  et  $\text{Pre}(t_{i+1}) \subseteq \eta(\mathcal{M}_i)$ . Nous pouvons ajouter au fragment de l'arbre de preuve  $\tau_i$  la règle 11 pour la transition  $t_{i+1}$ . Le nouveau séquent  $\mathcal{M}_{i+1}, l_{i+1} \vdash M_n$  au sommet de l'arbre est tel que les hypothèses d'induction sont vérifiées pour  $i + 1$  (les calculs de  $\eta(\mathcal{M}_{i+1})$  et  $\mu(\text{Max}(O_{i+1}))$  sont identiques à ceux de la section 4).

### Arbre final :

Quand la séquence  $e_1 \dots e_n$  a été complètement explorée, nous procédons à l'étape finale (section 2.5.3) pour obtenir un arbre de preuve canonique  $\tau$  tel que  $\sigma(\tau) = \sigma$ . Pour chaque linéarisation de  $P(O, \lambda)$ , un arbre de preuve canonique différent peut être obtenu. Pour une linéarisation donnée, l'arbre de preuve canonique est unique car les pas itératifs sont complètement définis par les noms des transitions.

**Théorème 2** *Pour chaque process fini  $(O, \lambda)$  tel que  $\mu(\text{Min}(O)) = M_0$ ,  $\mu(\text{Max}(O)) = M_n$  et pour chaque  $\sigma \in \text{Lin}(P(O, \lambda))$ , il existe exactement un arbre de preuve canonique  $\tau$  du séquent  $M_0, l_0 \vdash M_n$ .*

### Exemple

Pour le réseau de Petri de la figure 1, un process fini démarrant avec un marquage initial composé d'un jeton dans les places  $s_1$ ,  $s_4$  et  $s_7$  et terminant avec un marquage final composé de deux jetons dans chacune des places  $s_1$  et  $s_4$  et de un jeton dans la place  $s_7$  est représenté figure 1.b. C'est un process dont la transition  $t_1$  est franchie

avant  $t_2$ . Différents arbres de preuve canonique peuvent lui être associés : par exemple l'arbre  $\tau'$  avec  $\sigma(\tau') = t_1; t_3; t_2; t_4; t_5$  ou l'arbre  $\tau''$  avec  $\sigma(\tau'') = t_1; t_2; t_3; t_4; t_5$ . Ils sont tous équivalents quant au process et à l'ordre partiel généré.

## 6. Conclusion

Dans cet article, il a été prouvé que les ensembles de relations de précédence obtenus par les preuves canoniques de séquents sont des process de réseau de Petri et que réciproquement, à partir d'un process fini, un arbre de preuve canonique peut être construit. Ceci justifie bien la correspondance entre des applications de la règle " $\rightarrow_L$ " dans les arbres de preuve et les tirs de transition d'un réseau de Petri. C'est aussi une preuve que les preuves canoniques sont complètes, i.e. qu'elles génèrent tous les ordres partiels possibles parmi les tirs de transitions. De plus, si le marquage final est accessible depuis un marquage initial donné, il est possible de prouver le séquent par une preuve canonique.

Les résultats présentés dans cet article donnent une autre approche pour construire les process de réseau de Petri qui est similaire à celle basée sur les dépliages de réseau de Petri. Quelles sont donc les différences ? La plus importante est que l'approche basée sur la logique linéaire produit des process *finis* entre deux marquages prédéfinis. Comme nous ne travaillons qu'avec des process finis, cette approche est très utile pour prouver des propriétés de vivacité plutôt que de sûreté. Par exemple, elle n'est pas adaptée pour prouver qu'un état est inaccessible. Par contre, il est possible de prouver qu'un système peut être contrôlé afin d'avoir un certain comportement prédéfini. Le fait que seulement les process *finis* soient concernés est la raison pour laquelle l'approche est exactement la même, que le réseau de Petri soit ou non, ou même qu'il soit non borné. L'accessibilité n'est étudiée que dans le cas d'une liste finie prédéfinie de franchissements de transitions. Avoir des transitions *sources* (transitions sans place d'entrée) dans la liste n'est pas un problème car le nombre de fois qu'elles seront franchies est défini par le séquent d'accessibilité. On doit simplement utiliser l'élément neutre associé au connecteur  $\otimes$ .

Tous ces points sont des conséquences directes de la décidabilité du fragment multiplicatif de la logique linéaire parce que nous avons choisi une traduction des réseaux de Petri dans ce fragment sans avoir ajouté d'axiomes (l'élimination des coupures est préservée).

Notre travail futur concerne le développement d'approches compositionnelles et modulaires pour construire des process finis de réseaux de Petri car l'approche logique linéaire semble bien indiquée pour cela. En effet, l'utilisation de la règle de coupure (mais aussi dans plusieurs cas la règle " $\otimes_R$ ") que nous avons éliminée jusque-là est en fait une manière de séparer la preuve d'un séquent en une preuve de deux séquents de plus petite taille. En gardant une trace des franchissements qui ont produit et consommé chacun des jetons, il sera alors possible de fabriquer un process corres-

pendant à une composition séquentielle (règle de coupure) ou parallèle (règle  $\otimes_R$ ) de deux sous-process.

## 7. Bibliographie

- [AND 92] ANDREOLI J., « Logic programming with focusing proofs in linear logic », *Journal of logic and computation*, vol. 2, n° 3, 1992, p. 297-347.
- [BES 87] BEST E., DEVILLERS R., « Sequential and concurrent behaviour in Petri nets theory », *Theoretical Computer Science*, vol. 50, 1987, p. 87-136.
- [BRO 89] BROWN C., « Relating Petri Nets to Formulae of Linear Logic », LFCs series, Report ECS-LFCS-89-87, 1989, Laboratory for Foundations of Computer Science, University of Edinburgh, UK.
- [ENG 90] ENGBERG U., WINSKEL G., « Petri nets as models of linear logic », *Proceedings of the 15th Colloquium on Trees in Algebra and Programming*, vol. 431 de *Lecture Notes in Computer Science*, Copenhagen, Denmark, 1990, Berlin, Germany : Springer-Verlag, 1990, p. 147-161.
- [ESP 96] ESPARZA J., ROMER S., W. VOGLER, « An improvement of McMillan's unfolding algorithm », *TACAS'96*, vol. 1055 de *Lecture Notes in Computer Science*, 1996, p. 87-106.
- [GIR 87] GIRARD J. Y., « Linear logic », *Theoretical Computer Science*, vol. 50, 1987, p. 1-102.
- [GIR 97] GIRAULT F., « Formalisation en logique linéaire du fonctionnement des réseaux de Petri », PhD thesis, Université Paul Sabatier, Toulouse, France, Décembre 1997.
- [GOL 83] GOLTZ U., REISIG W., « The non-sequential behaviour of Petri nets », *Information and Computation*, vol. 57, 1983, p. 125-147.
- [GUN 89] GUNTER C., GEHLOT V., « Nets as tensor theories », *10<sup>th</sup> International Conference on Application and Theory of Petri nets*, Lecture Notes in Computer Science, Bonn, Germany, 1989, Berlin, Germany : Springer-Verlag, 1989, p. 174-191.
- [HOO 96] HOODGERS P., KLEIJN H., THIAGARAJAN P., « Event Structure Semantics for general Petri Nets », *Theoretical Computer Science*, vol. 153, 1996, p. 129-170.
- [MAN 02] MANCEL C., LOPEZ P., RIVIÈRE N., VALETTE R., « Relationships between Petri nets and constraint graphs : application to manufacturing », *15th IFAC world congress*, Barcelona, Spain, 21-26 July 2002, page n. 634.
- [MAR 89] MARTÍ-OLIET N., MESEGUER J., « From Petri Nets to Linear Logic », rapport n° 89-4R2, December 1989, SRI International, Computer Science Laboratory, Menlo Park, CA, USA, In : Pitt, D.H. et al. : *Lecture Notes in Computer Science*, Vol. 389.
- [MES 96] MESEGUER J., MONTANARI U., SASSONE V., « Process versus unfolding semantics for Place/Transition Petri nets », *Theoretical Computer Science*, vol. 153, 1996, p. 171-210.
- [NIE 81] NIELSEN M., PLOTKIN G., WINSKEL G., « Petri Nets, Event Structures and Domains », *Theoretical Computer Science*, vol. 13, 1981, p. 85-108.
- [PRA 99] PRADIN-CHÉZALVIEL B., VALETTE R., KÜNZLE L., « Scenario duration characterization of t-timed Petri nets using Linear logic », *IEEE PNPM'99*, Zaragoza, Spain, September 6-10 1999, p. 208-217.
- [REI 98] REISIG W., ROZENBERG G., *Lectures on Petri Nets I : Basic Models*, vol. 1491 de *Lecture Notes in Computer Science*, Springer-Verlag, 1998.