

Spécification et vérification des problèmes
concurrents :
Vérification avec les réseaux de Petri et les
réseaux de haut-niveau

J.F. Pradat-Peyre

28 janvier 2002

Table des matières

1. Réductions structurelles	2
2. Invariants colorés	15

1. Réductions structurelles

Soient π une propriété et \mathcal{M} un modèle. Vérifier que \mathcal{M} satisfait π peut se faire de manière efficace en définissant un modèle \mathcal{M}' , équivalent à \mathcal{M} pour la propriété π , de telle sorte que la propriété soit plus simple à vérifier sur \mathcal{M}' que sur \mathcal{M} .

Cette méthodologie est connue dans le cadre des réseaux de Petri sous le nom de réduction. Une réduction est une transformation de réseau qui réduit la taille du réseau tout en préservant un ensemble de propriétés. Trois points caractérisent une réduction :

1. les conditions d'applications : comment reconnaître que la réduction est applicable sur un réseau ;
2. la transformation de réseau : comment est défini le réseau réduit ;
3. les propriétés préservées : le réseau réduit a-t-il le même comportement que le réseau initial pour une propriété donnée ?

Parmi les réductions proposées dans [Ber83] nous nous intéresserons dans cette section aux agglomérations de transitions (pré et post) et à la suppression de place implicite. Ces réductions offrent en effet un très bon compromis entre fréquence d'utilisation, utilité de la réduction et simplicité d'application.

Nous nous attachons ici de plus à définir leurs équivalents dans le cadre des réseaux de haut niveau.

1.1. Principe d'extension aux réseaux colorés

Différents types de réductions ont été définis pour les réseaux colorés [CMS86], [Gen91], [Had91]. Nous présentons ici à celles définies dans [Had91]. En effet, plutôt que de définir de nouvelles réductions, l'auteur propose une méthodologie d'extension permettant de construire une réduction «colorée» à partir d'une réduction ordinaire qu'il applique à quelques réductions classiques. Cette méthodologie d'extension repose sur les trois principes suivant :

1. se rapprocher des conditions d'applications ordinaires, c'est à dire :
 - (a) ne pas ajouter de contraintes structurelles par rapport à la définition ordinaire ;
 - (b) n'imposer que les conditions fonctionnelles nécessaires au maintien de l'équivalence entre le réseau réduit et l'original.
2. se limiter aux extensions permettant d'obtenir un réseau coloré réellement réduit et utilisable :
 - (a) ne pas augmenter le domaine de couleur des transitions ;
 - (b) n'utiliser pour la définition des nouvelles fonctions de couleur que des compositions de fonctions de couleur ou de leur inverse.
3. maintenir le même ensemble de propriétés préservées.

Une façon simple de procéder est alors d'étudier les conditions sous lesquelles une réduction colorée peut être vue comme la synthèse d'une série de réductions ordinaires. Pour cela, on étudie comment reconnaître directement dans un

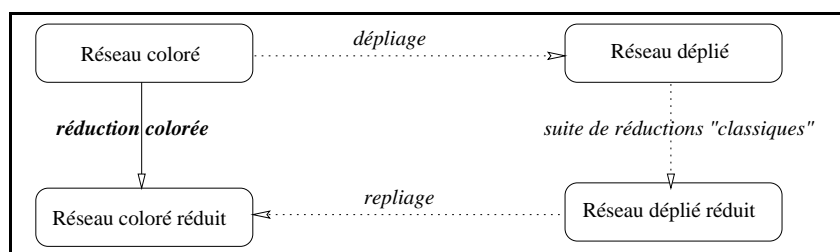


Figure 1. Méthodologie d'extension

réseau coloré les règles d'applications et de transformation liées à l'application d'une série de réductions dans le réseau déplié ainsi que le résume la figure 1.

En plus des conditions portant simplement sur la structure (une place n'a qu'une transition en entrée, une transition n'a qu'une seule place donnée en sortie, ...), il est important de caractériser les fonctions de couleur par rapport à la structure qu'elles induisent dans le réseau déplié. En effet, une transition peut n'avoir qu'une seule place en sortie dans le réseau coloré, alors que dans

le réseau déplié, une instance de cette transition peut avoir un ensemble de places en sortie ainsi que l'illustre la figure 2. Il est donc nécessaire de définir

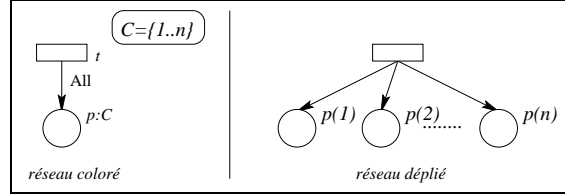


Figure 2. Exemple de structure modifiée dans le réseau déplié

des conditions suffisantes assurant que le dépliage produira un réseau correct du point de vue de la structure attendue. Les définitions suivantes répondent à ce besoin.

Définition 0.1 (Propriétés de fonction) Une fonction f de C dans $Bag(C')$ est dite :

- **unitaire** si et seulement si $\forall c \in C, c' \in C', f(c)(c') = 0$ ou $f(c)(c') = 1$;
- **orthonormale** si et seulement si $C = C'$ et il existe une permutation σ sur C telle que $f(c) = \sigma(c)$;
- **quasi-injective** si et seulement si $\forall c_1, c_2 \in C, \forall c' \in C', f(c_1)(c') \neq 0$ et $f(c_2)(c') \neq 0 \implies c_1 = c_2$.
- **quasi-surjective** si et seulement si $\forall c' \in C', \exists c \in C$ tq $f(c)(c') \neq 0$;
- **quasi-bijective** si et seulement si $C = C'$ et $\forall c \in C, \exists !c' \in C$ tq $f(c)(c') \neq 0$.

Une fonction unitaire ne peut produire que des valuations égales à 0 ou à 1 dans le réseau déplié. Une fonction orthonormale est, à une permutation près, une fonction identité sur un domaine C . Quant aux autres caractérisations de fonction, elles correspondent à la notion usuelle d'application injective, surjective ou bijective, hormis le fait que ces fonctions vont d'un domaine vers un domaine de multi-ensembles. En pratique, la syntaxe des réseaux bien-formés permet de tester facilement des conditions suffisantes précises pour ces propriétés. Par exemple, une fonction quasi-injective sur un arc reliant une place p et une transition t ne peut faire intervenir de projection. Il est alors suffisant de tester que le domaine de t est plus petit (au sens du produit cartésien) que le domaine de p et que cette fonction est un tuple faisant intervenir toutes les variables instanciées par la transition t et n'utilisant pas la fonction *All* sur les classes de couleur communes au domaine de t et de p .

Nous étudions maintenant l'extension de deux types de réductions : la pré et la post agglomération de transition et la suppression de place implicite.

1.2. Pré et post-agglomérations de transitions

L'idée de la pré et de la post-agglomération de transitions est de considérer deux ensembles disjoints de transitions (notés H et F) et de définir des hypothèses permettant d'assurer l'équivalence entre le réseau initial, où les transitions de H et de F seront franchies en séquence, et le réseau réduit, où les couples (h, f) seront franchis atomiquement. Pour la pré-agglomération, des hypothèses structurelles garantissent que l'on peut retarder le franchissement d'une transition h jusqu'au franchissement d'une transition de F sans altérer l'ensemble des propriétés de bases. Ce franchissement peut alors se faire atomiquement, ce qui a pour conséquence de supprimer un état intermédiaire et donc de réduire la taille du graphe des marquages accessibles. De même, pour la post-agglomération, des hypothèses structurelles assurent que l'on peut avancer le franchissement d'une transition f dès que l'on a franchi une transition de H . On peut également considérer que ce franchissement est atomique.

Pré-agglomération de transitions

Pour cette réduction on considère une transition h et un ensemble de transitions F (n'incluant pas h).

Avant de présenter la pré-agglomération colorée, nous rappelons les conditions d'application de cette réduction dans le cas des réseaux ordinaires.

Les conditions structurelles suivantes garantissent qu'il est équivalent de considérer qu'il y a un état intermédiaire entre le franchissement de h et d'une transition de F ou de considérer que le franchissement des séquence $h.f$ est effectué de façon atomique. Le terme équivalent désigne ici équivalent pour le propriété de base (vivacité, caractère borné, ...) [Ber83], [Ber86].

Définition 0.2 (Transitions ordinaires pré-agglomerables) Soit (R, m_0) un réseau de Petri. Un ensemble de transitions F est pré-agglomerable avec une transition h n'appartenant pas à F si et seulement si les conditions suivantes sont vérifiées :

1. Il existe une place p modélisant un état intermédiaire entre le franchissement de h et d'une transition de F :
 - (a) $m_0(p) = 0$;
 - (b) $\bullet p = \{h\}$ et $p^\bullet = F$;
 - (c) $Post(p, h) = 1$
 - (d) $\forall f \in F, Pre(p, f) = 1$.
2. h ne produit des marques que dans p : $h^\bullet = \{p\}$;

3. h n'est en conflit avec aucune autre transition : $\forall q \in \bullet h, q^\bullet = \{h\}$;
4. h a au moins une pré-condition : $\bullet h \neq \emptyset$.

La première hypothèse garantit que toute occurrence de franchissement d'une transition f de F est précédée du franchissement de h . La seconde assure que h n'est utile que pour le franchissement d'une transition de F . La troisième implique que si h est franchissable alors elle le reste jusqu'à ce qu'elle soit franchie (h peut donc être retardée). Enfin la quatrième hypothèse permet d'assurer l'équivalence pour le caractère borné entre le réseau réduit et l'original.

Définition 0.3 (Réseau pré-aggloméré) Une pré-agglomération sur le réseau (R, m_0) produit le réseau réduit $(R_r, m_{0,r})$ défini par :

1. Places et transitions du réseau réduit :
 - $P_r = P \setminus \{p\}, T_r = T \setminus \{h\}$;
 - $\forall q \in P_r, m_{0,r}(q) = m_0(q)$.
2. Pré et Post conditions inchangées : $\forall t \in T_r$,
 - $\forall q \in P_r, Post_r(q, t) = Post(q, t)$
 - $\forall q \in P_r \setminus \bullet h, Pré_r(q, t) = Pré(q, t)$
3. Nouvelles pré-conditions : $\forall f \in F, \forall q \in \bullet h$,
 - $Pré_r(q, f) = Pré(q, h)$

L'extension de cette réduction aux réseaux colorés se fait en ajoutant aux conditions portant sur la structure des conditions portant sur les domaines et les fonctions de couleur (conditions 1.c, 1.d et seconde partie de la condition 3). Nous justifierons par quelques exemples la nécessité de ces nouvelles conditions.

Définition 0.4 (Transitions pré-agglomérables) Soit (CN, m_0) un réseau coloré. Un ensemble de transitions F est pré-agglomérable avec une transition h n'appartenant pas à F si et seulement si les conditions suivantes sont vérifiées :

1. Il existe une place p modélisant un état intermédiaire entre le franchissement de h et d'une transition de F :
 - (a) $m_0(p) = 0$;
 - (b) $\bullet p = \{h\}$ et $p^\bullet = F$;
 - (c) $\mathcal{C}(p) = \mathcal{C}(h)$ et $Post(p, h)$ est une fonction de couleur orthonormale ;
 - (d) $\forall f \in F, Pré(p, f)$ est une fonction de couleur unitaire et quasi-surjective.
2. h n'a que p comme post-condition : $h^\bullet = \{p\}$;
3. h n'est en conflit avec aucune autre transition : $\forall q \in \bullet h, q^\bullet = \{h\}$ et $Pré(q, h)$ est une fonction de couleur quasi-injective ;

4. h a au moins une pré-condition : $\bullet h \neq \emptyset$.

Les deux petits exemples suivants illustrent l'obligation d'ajouter aux conditions de structures des conditions sur les fonctions de couleur. Sur le premier (figure 3), la fonction $Post(p, h) = All - X$ n'est pas orthonormale. Comme on le voit, le réseau déplié (modèle de droite de la figure 3) n'a pas une structure correcte pour la pré-agglomération.

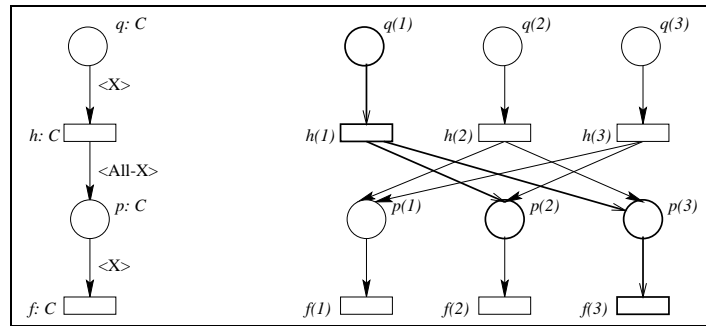


Figure 3. La fonction $Post(p, h)$ doit être orthonormale

Sur le deuxième exemple (figure 4), on voit qu'une fonction non quasi-injective génère un conflit entre des instances de h dans le réseau déplié. De

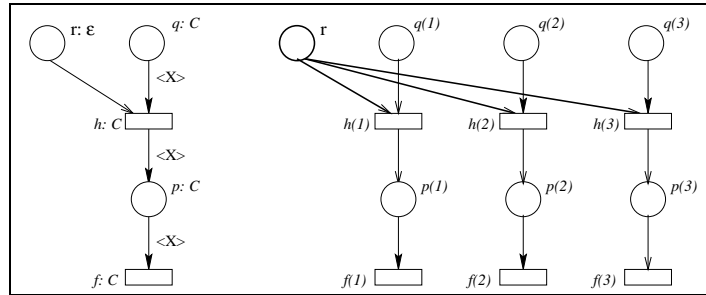


Figure 4. Les fonction $Pré(q, h)$ doivent être quasi-injectives

nouveau, le schéma du réseau déplié (modèle de droite de la figure 4) ne définit pas une structure correcte de pré-agglomération (par exemple, $h(1)$ est en conflit avec $h(2)$ sur la place r).

Sur l'exemple de la figure 5, les conditions d'application de la réduction sont vérifiées dans le réseau coloré (modèle de gauche). On constate alors qu'il est possible d'opérer une série de pré-agglomération ordinaire dans le réseau déplié (modèle de droite) : agglomération de $h(1)$ avec $f(1)$, puis agglomération de

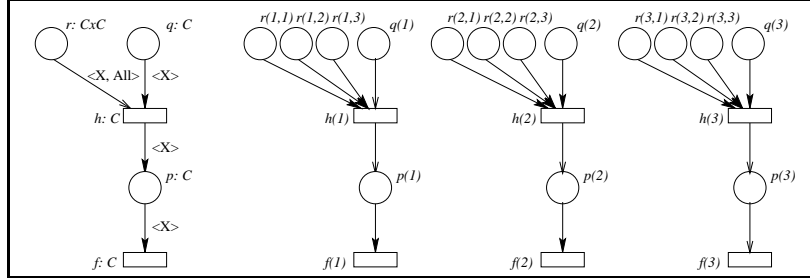


Figure 5. Les conditions de la Pré-agglomération sont respectées

$h(2)$ avec $f(2)$ et enfin agglomération de $h(3)$ avec $f(3)$. On obtient alors le

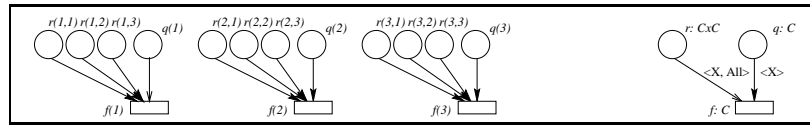


Figure 6. Réseau aggloméré

réseau ordinaire (modèle de gauche) de la figure 6 qui après recoloration donne le modèle de droite de la même figure.

Il est bien entendu possible d'effectuer directement la transformation au niveau du réseau coloré : agglomération de h avec chaque transition de F . La seule différence entre la définition de la transformation colorée et la version ordinaire tient dans le fait qu'il faut composer des fonctions afin de définir les nouvelles pré-conditions des transitions de F (point 3 de la définition suivante). En effet, un jeton consommé par le franchissement d'une transition f dans une place q dans le réseau réduit est un jeton qui était consommé initialement dans la place p après avoir été consommé par h dans q et modifié par la fonction $Post(q, h)$.

Définition 0.5 (Réseau pré-aggloméré) Une pré-agglomération sur le réseau (CN, m_0) produit le réseau réduit (CN_r, m_{0_r}) défini par :

1. Places et transitions du réseau réduit :
 - $P_r = P \setminus \{p\}$, $T_r = T \setminus \{h\}$;
 - $\forall t \in T_r, \forall q \in P_r, C_r(t) = C(t)$, $C_r(q) = C(q)$ et $m_{0_r}(q) = m_0(q)$.
2. Pré et Post conditions inchangées : $\forall t \in T_r$,
 - $\forall q \in P_r, Post_r(q, t) = Post(q, t)$
 - $\forall q \notin \bullet h, Pré_r(q, t) = Pré(p, t)$
3. Nouvelles pré-conditions : $\forall f \in F, \forall q \in \bullet h$,
 - $Pré_r(q, f) = Pré(q, h) \circ (Post(p, h))^{-1} \circ Pré(p, f)$

La figure 7 propose un exemple plus complexe d'application de cette réduction.

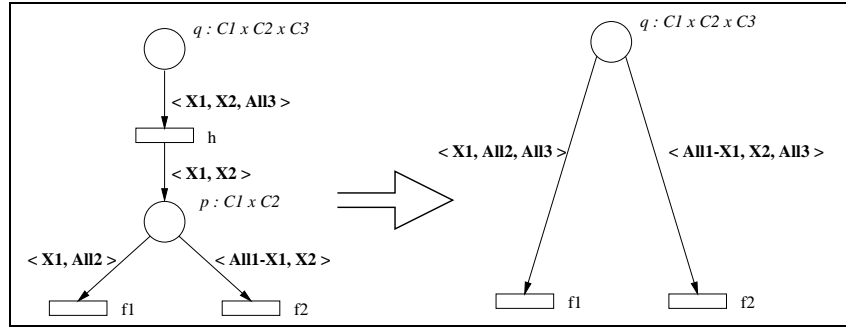


Figure 7. *Un exemple de pré-agglomération*

Post-agglomération de transitions

Les conditions d'application de la post-agglomération assurent que toute transition f de F est immédiatement franchissable après le franchissement d'une transition h de H . De nouveau, les différences entre la version ordinaire et colorée ne portent que sur les valuations des arcs autour de p et sur les domaines de couleur (points 1.c et 1.d).

De nouveau, avant de donner les conditions d'applications de la réduction colorée, nous rappelons les conditions d'applications de la version ordinaire. Ces conditions assurent l'équivalence pour les propriétés de base entre le réseau réduit et l'original [Ber83].

Définition 0.6 (Transitions ordinaires post-agglomerables) Soit (R, m_0) un réseau de Petri. Un ensemble de transitions F est *post-agglomerable* avec un ensemble de transitions H disjoint de F ($H \cap F = \emptyset$) si et seulement si les conditions suivantes sont vérifiées :

1. Il existe une place p modélisant un état intermédiaire entre le franchissement d'une transition de H et d'une transition de F :
 - (a) $m_0(p) = 0$;
 - (b) $\bullet p = H$ et $p \bullet = F$;
 - (c) $\forall h \in H, Post(p, h) = 1$;
 - (d) $\forall f \in F$ et $Pre(p, f) = 1$ ¹.

¹La version originale de Berthelot autorise des valuations hétérogènes. Cependant, ce cas ne se rencontre pas en pratique et complique inutilement l'extension de cette réduction aux réseaux colorés.

2. Les transitions de F n'ont pas d'autre pré-condition que $p : \bullet F = \{p\}$;
3. Il existe une transition f de F ayant une post-condition : $F^\bullet \neq \emptyset$.

Comme pour la pré-agglomération, la première hypothèse garantit un séquençement entre le franchissement des transitions de H et de celles de F . La seconde hypothèse (point clef de la post-agglomération) assure que toute transition f de F est franchissable dès que p est marquée (on peut donc "avancer" le franchissement des transitions f). La dernière assure l'équivalence pour le caractère borné entre le réseau réduit et le réseau original.

Définition 0.7 (Réseau post-aggloméré) Une post-agglomération sur le réseau (R, m_0) produit le réseau réduit $(R_r, m_{0,r})$ défini par :

1. Places et transitions du réseau réduit :
 $P_r = P \setminus \{p\}$ et $T_r = T \cup (H \times F) \setminus (H \cup F)$;
on note hf la transition (h, f) de $H \times F$;
2. Partie du réseau inchangée : $\forall t \in T_r \setminus (H \times F), \forall q \in P_r,$
- $Pré_r(q, t) = Pré(q, t)$ et $Post_r(q, t) = Post(q, t)$;
- $m_{0,r}(q) = m_0(q)$
3. Les nouvelles transitions : $\forall hf \in (H \times F), \forall q \in P_r,$
- $Pré_r(q, hf) = Pré(q, h)$;
- $Post_r(q, hf) = Post(q, h) + Post(q, f)$.

Définition 0.8 (Transitions post-agglomerables) Soit (CN, M_0) un réseau coloré. Un ensemble de transitions F est post-agglomerable avec un ensemble de transitions H disjoint de F ($H \cap F = \emptyset$) si et seulement si les conditions suivantes sont vérifiées :

1. Il existe une place p modélisant un état intermédiaire entre le franchissement d'une transition de H et d'une transition de F :
 - (a) $m_0(p) = 0$;
 - (b) $\bullet p = H$ et $p^\bullet = F$;
 - (c) $\forall h \in H, \mathcal{C}(h) = \mathcal{C}(p) \times C_h$ et $Post(p, h)$ est la composition d'une fonction de couleur orthonormale sur $\mathcal{C}(h)$ et d'une projection de $\mathcal{C}(h)$ dans $\mathcal{C}(p)$;
 - (d) $\forall f \in F, \mathcal{C}(p) = \mathcal{C}(f)$ et $Pré(p, f)$ est une fonction de couleur orthonormale.
2. Les transitions de F n'ont pas d'autre pré-condition que $p : \bullet F = \{p\}$;
3. Il existe une transition f de F ayant une post-condition ² : $F^\bullet \neq \emptyset$.

²Cette condition peut se raffiner en : $\forall c \in \mathcal{C}(p), \exists f \in F, \exists q \in P, tq \text{ } Post(q, f)(c) \neq 0$

Dans le réseau aggloméré, les transitions de H et de F sont fusionnées. Les valuations prennent en compte ces modifications.

Définition 0.9 (Réseau post-aggloméré) Une post-agglomération sur le réseau (CN, M_0) produit le réseau réduit $(CN_r, m_{0,r})$ défini par :

1. Places et transitions du réseau réduit :
 $P_r = P \setminus \{p\}$ et $T_r = T \cup (H \times F) \setminus (H \cup F)$;
on note hf la transition (h, f) de $H \times F$;
2. Partie du réseau inchangée : $\forall t \in T_r \setminus (H \times F), \forall q \in P_r$,
- $C_r(t) = C(t)$ et $C_r(q) = C(q)$;
- $Pré_r(q, t) = Pré(q, t)$ et $Post_r(q, t) = Post(q, t)$;
- $m_{0,r}(q) = m_0(q)$
3. Les nouvelles transitions : $\forall hf \in (H \times F), \forall q \in P_r$,
- $C_r(hf) = C(h)$;
- $Pré_r(q, hf) = Pré(q, h)$;
- $Post_r(q, hf) = Post(q, h) + Post(q, f) \circ Pré^{-1}(p, f) \circ Post(p, h)$.

Dans le cas où l'ensemble F est réduit à un singleton ($F = \{f\}$) il est possible de relâcher les contraintes portant sur les fonctions de couleur valuant les arcs reliant H et la place intermédiaire p (une fonction de couleur unitaire suffit). De plus, le domaine de couleur de h peut être quelconque. La condition 1.c se réécrit alors en :

$$\forall h \in H, Post(p, h) \text{ est une fonction de couleur unitaire;}$$

La définition du réseau réduit est la même que dans le cas précédent. L'exemple suivant (figure 8) illustre cette réduction.

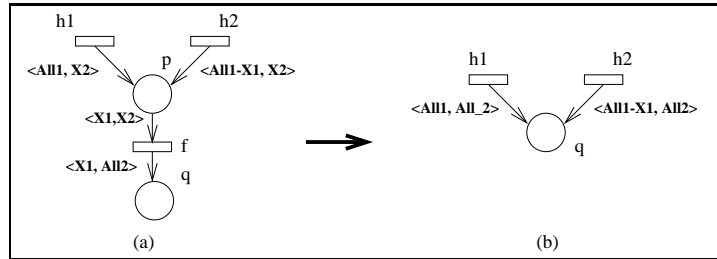


Figure 8. Un exemple de post-agglomération

1.3. *Suppression de place implicite*

La suppression d'une place implicite consiste à supprimer du réseau une place qui n'est jamais à elle seule un obstacle au franchissement de transitions. D'une certaine façon cette place ne sert à rien. Cette place est caractérisée par des conditions portant sur le marquage initial (initialement elle ne sert pas) et des conditions sur l'existence d'un invariant particulier (qui assure que la condition initiale est toujours vérifiée). L'intérêt de cette réduction est de supprimer une place et donc de simplifier les pré ou les post conditions du réseau et de permettre ainsi l'application d'autres réductions (par exemple des agglomérations).

Définition 0.10 (Place implicite) *Soit (CN, m_0) un réseau coloré. Une place p est dite implicite si il existe $P' \subset P$, avec $p \notin P'$ tq :*

1. *Il existe un flot sur le domaine $\mathcal{C}(p)$ $\mathcal{F} \equiv \mathcal{F}_p \cdot p - \sum_{q \in P'} \mathcal{F}_q \cdot q$ avec*
 - \mathcal{F}_p *une application linéaire quasi-bijective sur $Bag(\mathcal{C}(p))$ et*
 - \mathcal{F}_q *des applications linéaires de $Bag(\mathcal{C}(q))$ vers $Bag(\mathcal{C}(p))$.*
2. $\forall t \in T, \forall c_t \in \mathcal{C}(t)$,

$$\mathcal{F}_p(m_0(p)) - \sum_{q \in P'} \mathcal{F}_q(m_0(q)) \geq \mathcal{F}_p(Pré(p, t)(c_t)) - \sum_{q \in P'} \mathcal{F}_q(Pré(q, t)(c_t))$$

La seconde condition assure qu'initialement une place implicite ne peut empêcher le franchissement d'une transition. La première condition garantit que cette hypothèse est reproductible pour tout marquage accessible.

1.4. *Exemples d'application*

Reprenons l'exemple de la base de données répartie présenté chapitre 7. Sur ce modèle, on vérifie aisément que la transition t_4 est "post-agglomérable" avec la transition t_3 . Le schéma structurel est vérifié : la place `Update` est en effet initialement non marquée, sa seule entrée est t_3 et sa seule sortie est t_4 ; la seule pré-condition de t_4 est la place `Update`. De plus, les contraintes fonctionnelles sont elles aussi vérifiées : le domaine de couleur de `Update` est le même que celui de t_3 et de t_4 et la fonction $\langle X1, X2 \rangle$ est bien une fonction orthonormale. Après application de cette réduction on obtient le réseau de la figure 10 où l'on voit que la place `Update` a disparu et que les transitions t_3 et t_4 ont été fusionnées. Sur ce nouveau modèle, il est clair que la place `Start Update` est une place implicite ; en effet elle sera toujours marquée par `A111` et ne sera donc jamais un obstacle au franchissement de la transition t_3, t_4 . Cette place peut donc être supprimée sans modifier les propriétés fondamentales du réseau

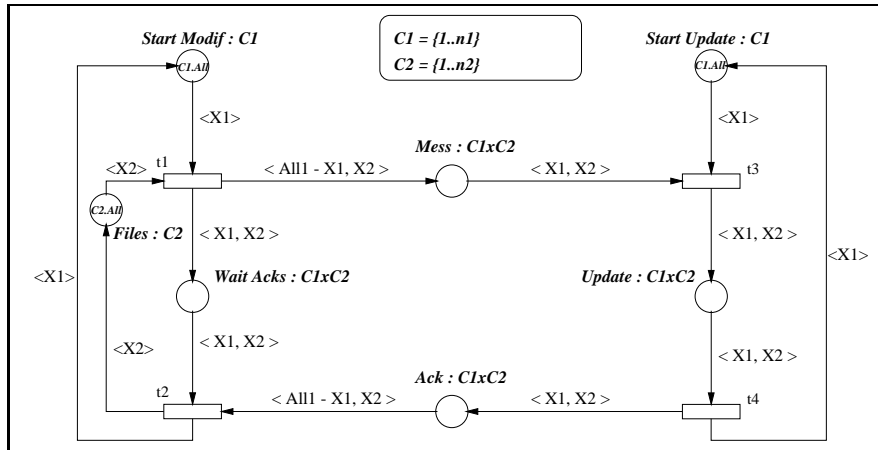


Figure 9. Un modèle de base de données répartie

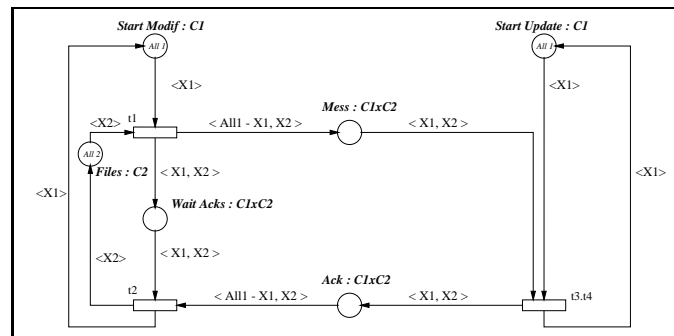


Figure 10. Après agglomération de t_3 et de t_4

initial, et le nouveau modèle est décrit figure 11. On voit ici tout l'intérêt de la suppression d'une place implicite. Si cette réduction ne réduit pas la taille du graphe des marquages accessibles, elle permet l'application de nouvelles réductions. Sur le modèle précédent, après suppression de la place implicite **Start Update**, une post-agglomération peut être effectuée entre t_1 et $t_3.t_4$ ce qui n'était pas le cas avant la suppression de **Start Update**. On obtient alors le modèle de gauche de la figure 12. Sur ce nouveau modèle, la place **Ack** est une place implicite (et non la place **Wait Acks**) comme le prouve le flot de domaine $C_1 \times C_2$ suivant : $\langle X_1, X_2 \rangle.Ack - \langle All_1 - X_1, X_2 \rangle.WaitAcks = 0$. Après suppression de cette place implicite, on obtient le modèle du milieu de la figure 12. Sur ce nouveau modèle, une post-agglomération peut être réalisée entre la transition $t_1.t_3.t_4$ et la transition t_2 , ce qui donne le modèle de droite de cette même figure. Sur ce dernier modèle, les places restantes sont des places

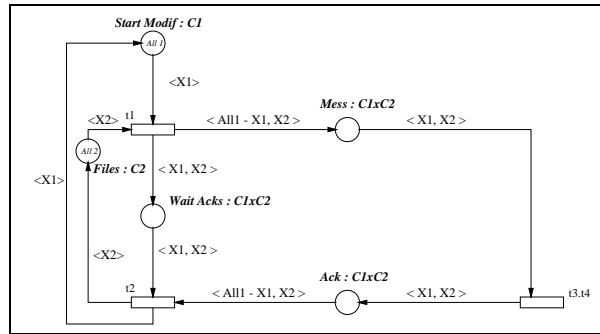


Figure 11. *Après suppression de la place implicite "Start Update"*

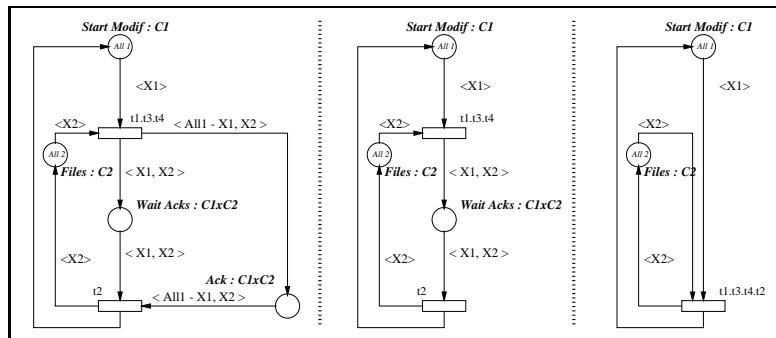


Figure 12. *Après d'autres réductions*

implicites et peuvent être supprimées ; le modèle initial est donc équivalent pour les propriétés fondamentales (vivacité, caractère borné, état d'accueil, ...) au réseau réduit à la transition $t1.t3.t4.t2$ qui vérifie bien évidemment toutes ces propriétés.

Il est évident par contre que certaines propriétés ont été perdues lors de ces phases de réduction. Si l'on veut vérifier des propriétés plus "fines" que la vivacité (par exemple des formules de logique temporelle [PPP00]) il est nécessaire de conserver certaines places et transitions. Si l'on veut par exemple vérifier la propriété qui exprime que le site 1 ne peut simultanément travailler sur le même fichier que le site 2, on ne peut réaliser la dernière agglomération et il faut s'arrêter au modèle du centre de la figure 12.

Si l'on prend le modèle des philosophes proposé dans le chapitre 7 pour illustrer les réseaux ordonnés, plusieurs post-agglomérations sont applicables autour des places *Mange*, *Finir1* et *Finir2* ; on obtient alors le réseau de la figure 13. Ce réseau met en avant le problème d'interblocage possible. Ce

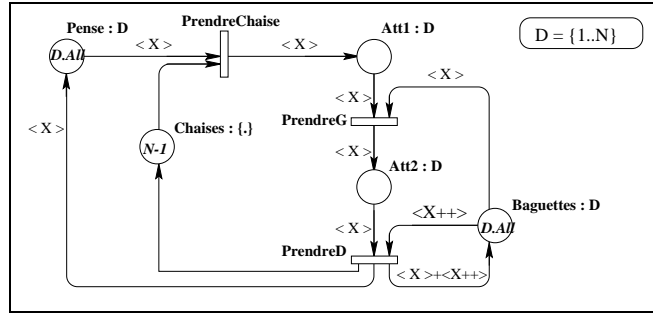


Figure 13. *Le modèle des philosophes réduit*

modèle ne peut plus être réduit. En effet, le problème d'interblocage est lié ici au contrôle du verrou $\cup_{x \in D} \{Att2(x), Baguettes(x)\}$. Dans cette version des philosophes, ce contrôle est réalisé par l'invariant introduit par la place *Chaises* qui garantit que le nombre de philosophes en attente (place *Att2*) est au plus égal au nombre de marques initiales dans la place *Chaises*. La présence ou non d'un interblocage dépend donc du marquage initial de cette place, et ce genre de condition est difficilement caractérisable par des conditions structurelles.

2. Invariants colorés

Nous avons vu au chapitre 3 l'importance des invariants dans l'analyse structurelle des réseaux de Petri. Dans les réseaux de Petri ordinaires, le calcul de ces invariants se fait par résolution du système d'équations déduit de la matrice d'incidence du réseau : les variables de ce système sont les places du réseau, et les équations correspondent à l'incidence des transitions sur les places. La résolution de ce système se fait alors par l'algorithme de Gauss dans le cas général ou par l'algorithme de Farkas dans le cas de recherche d'invariants à coefficients positifs. On obtient alors une famille génératrice c'est à dire permettant de générer par combinaisons linéaires tous les invariants du réseau. Dans le cadre des réseaux colorés, trois types de problèmes se présentent :

- comment définir les invariants d'un réseau coloré et caractériser une famille génératrice ;
- comment calculer de manière efficace une famille génératrice d'invariants d'un réseau coloré lorsque la taille des domaines de couleur est fixée ;
- comment calculer une famille génératrice paramétrée, c'est à dire valide quelle que soit la taille des domaines de couleur.

Les deux premiers problèmes peuvent évidemment être résolus en travaillant explicitement sur le réseau de Petri obtenu par "dépliage" du réseau coloré.

Ce type de méthodologie se heurte cependant à la complexité engendrée par la taille du réseau déplié et à l'interprétation des invariants obtenus dans le réseau coloré. Il est de plus impossible d'essayer d'obtenir par cette approche une famille génératrice et paramétrée d'invariants puisque l'on est obligé au préalable de fixer la taille des domaines de couleur pour effectuer le dépliage.

Les travaux portant sur le calcul d'invariant peuvent être regroupés en deux familles. Une première approche consiste à restreindre la définition des réseaux colorés en limitant par exemple les fonctions de couleur utilisables ou la composition des domaines de couleur des places et des transitions. Ces restrictions induisent alors une structure particulière de la matrice d'incidence et de l'ensemble des invariants de ces sous-classes dont on peut tirer parti pour définir des algorithmes efficaces et paramétrés de calcul de famille génératrice. C'est ainsi que l'on obtient un calcul de famille génératrice pour les réseaux à prédicats/transitions [MV85, Vau87], une famille génératrice paramétrée pour les réseaux réguliers [Had87] et pour les réseaux homogènes [CH88] ou encore une famille génératrice paramétrée d'invariants positifs pour les réseaux réguliers mono-classe [CHP91].

Une seconde approche consiste à généraliser l'algorithme de Gauss au calcul sur les applications linéaires en utilisant en particulier la notion de semi-inverse généralisée [Cou90].

Il est à noter qu'il n'existe à ce jour aucun algorithme permettant de calculer une famille génératrice **et** paramétrée pour un réseau bien-formé ou un réseau coloré quelconque.

2.1. Définitions des invariants colorés

Le choix d'une définition des invariants d'un réseau coloré doit satisfaire au moins les deux exigences suivantes. D'une part, il semble naturel qu'un invariant d'un réseau coloré (ou bien-formé) soit un invariant coloré c'est à dire permettant de prendre en compte la valeur des couleurs du réseau. Par exemple, si un domaine de couleur modélise un ensemble de processus, un invariant doit pouvoir exprimer des propriétés relatives à un processus particulier ou à l'ensemble des processus sauf un sous-ensemble particulier, ou encore, associant un processus à son voisin dans une classe ordonnée. D'autre part, il faut que la définition des invariants d'un réseau coloré soit utilisable mathématiquement, c'est à dire qu'elle permette le développement d'algorithmes efficaces de calcul opérant directement sur le réseau coloré. On peut de plus souhaiter que les invariants soient interprétables en terme de comportement du réseau de haut-niveau

La définition que nous proposons est celle la plus couramment admise comme

satisfaisant ces exigences. Celle-ci définit un invariant comme une somme pondérée de places associée à un domaine de couleur, le domaine de l'invariant qui peut être vu comme étant son domaine d'interprétation. Chaque poids associé aux places est une fonction du domaine de couleur de la place vers le domaine de couleur de l'invariant.

Dans la définition suivante, $Bag_{\mathbb{Q}}(A)$ désigne le \mathbb{Q} espace vectoriel canonique sur l'ensemble non vide A qui inclut $Bag(A)$. On note également W la matrice d'incidence du réseau ($\forall p \in P, t \in T, W(p, t) = Post(p, t) - Pré(p, t)$) afin de ne pas la confondre avec un domaine de couleur.

Définition 0.11 (Invariant coloré) *Un invariant coloré \mathcal{F} est une somme formelle pondérée des places $\sum_{p \in P} \mathcal{F}(p) \cdot p$ définie par :*

1. $\mathcal{C}(\mathcal{F})$ le domaine de couleur de l'invariant ;
2. $\forall p \in P, \mathcal{F}(p)$ une application linéaire de $Bag_{\mathbb{Q}}(\mathcal{C}(p))$ vers $Bag_{\mathbb{Q}}(\mathcal{C}(\mathcal{F}))$ telle que :

$$\forall t \in T, \sum_{p \in P} \mathcal{F}(p) \circ W(p, t) = 0$$

Lorsque l'on impose que les coefficients soient des applications linéaires de $Bag(\mathcal{C}(p))$ vers $Bag(\mathcal{C}(\mathcal{F}))$ on parle de flot positif (ou encore de semi-flot) ; sinon, on parle de flot.

En utilisant cette définition, il vient immédiatement que si \mathcal{F} est un invariant alors pour tout marquage accessible m ,

$$\sum_{p \in P} \mathcal{F}(p)(m(p)) = \sum_{p \in P} \mathcal{F}(p)(m_0(p))$$

Un invariant \mathcal{F} peut donc s'interpréter comme un ensemble d'équations liant le marquage d'un sous-ensemble de places du réseau pour tout état accessible avec le marquage initial :

$$\forall m \in Acc(CN, m_0), \forall c \in \mathcal{C}(\mathcal{F}), \sum_{p \in P} \sum_{c_p \in \mathcal{C}(p)} [\mathcal{F}(p)(c_p)(c)] \cdot m(p(c_p)) = cst$$

On notera ces équations simplement par ³,

$$\forall c \in \mathcal{C}(\mathcal{F}), \sum_{p \in P} \sum_{c_p \in \mathcal{C}(p)} [\mathcal{F}(p)(c_p)(c)] \cdot p(c_p) = cst$$

Le réseau de la figure 14 modélise l'envoi d'un message par un processus à

³ $p(c_p)$ désigne l'instance c_p de la place p .

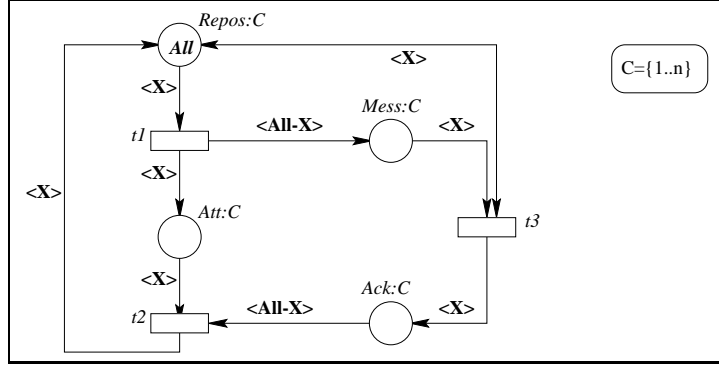


Figure 14. Un réseau coloré

tous les autres (transition $t1$) puis l'attente par celui d'un message d'acquittement de l'ensemble des autres processus (transition $t2$). La prise en compte du message et l'envoi de l'acquittement est modélisé par la transition $t3$.

Pour ce réseau, la somme formelle \mathcal{F} définie par $\mathcal{F} = \langle x \rangle . Mess + \langle x \rangle . Ack - \langle all - x \rangle . Att$ avec $\mathcal{C}(\mathcal{F}) = C$ est un invariant. Il s'interprète comme

$$\forall x \in C, Mess(x) + Ack(x) - \sum_{y \neq x} Att(y) = 0$$

et signifie que si un site x a reçu un message ou a envoyé un acquittement qui n'a pas encore été pris en compte alors nécessairement un autre site $y \neq x$ est en attente d'acquittement. On peut également interpréter cet invariant par «le nombre de messages à destination ou en provenance d'un site x est égal au nombre de sites en attente différent de x » .

Les invariants donnent donc une information précieuse sur les évolutions possibles du réseau sans avoir à construire son graphe des marquages accessibles (ou mieux, son graphe des marquages symboliques). Il semble donc naturel de vouloir connaître tous les flots d'un réseau. Comme cet ensemble est généralement infini, on cherche à représenter l'ensemble de tous les flots par une famille finie ; on parle alors de famille génératrice.

Définition 0.12 (Famille génératrice) *Un ensemble fini $\{\mathcal{F}_i\}$ de flots est une famille génératrice de flots d'un réseau si et seulement si tout flot du réseau s'obtient par combinaison linéaire des flots \mathcal{F}_i , les coefficients de cette combinaison étant des applications linéaires compatibles avec le domaine de couleur des invariants.*

Par exemple, les deux flots suivants de domaine de couleur C forment une famille génératrice du réseau de la figure 14.

1. $\mathcal{F}_1 = \langle x \rangle.Mess + \langle x \rangle.Ack - \langle all - x \rangle.Att$ et $\mathcal{C}(\mathcal{F}_1) = C$
2. $\mathcal{F}_2 = \langle x \rangle.Repos + \langle x \rangle.Att$ et $\mathcal{C}(\mathcal{F}_2) = C$

Ainsi, le flot $\mathcal{F} = \langle x \rangle.Mess + \langle x \rangle.Ack + \langle all - x \rangle.Repos$ de domaine C peut s'écrire comme $\mathcal{F} = \mathcal{F}_1 + \langle All - X \rangle.F_2$.

2.2. Problématique du calcul des flots d'un réseau

Le calcul des flots peut se faire de manière intuitive en adaptant les règles d'élimination de Gauss aux matrices d'applications linéaires. Pour cela, nous partons de la matrice d'incidence du réseau en indiquant à droite de chaque ligne la famille initiale de flots (initialement, chaque place est pondérée par la fonction identité de son domaine de couleur). Pour le réseau précédent (Fig.14), on a la matrice suivante :

$$W = \begin{pmatrix} & t1 & t2 & t3 \\ \langle x \rangle & -\langle x \rangle & \langle x \rangle & 0 \\ \langle x \rangle & \langle x \rangle & -\langle x \rangle & 0 \\ \langle all - x \rangle & \langle all - x \rangle & 0 & -\langle x \rangle \\ 0 & 0 & -\langle all - x \rangle & \langle x \rangle \end{pmatrix} \begin{matrix} \langle x \rangle.Repos \\ \langle x \rangle.Att \\ \langle x \rangle.Mess \\ \langle x \rangle.Ack \end{matrix}$$

La première règle consiste à ajouter à une ligne une autre ligne pré-multipliée par une fonction afin d'annuler certains coefficients de la matrice. Ainsi, en combinant *Att* à *Repos* et *Mess* on obtient la matrice suivante.

$$\begin{pmatrix} & t1 & t2 & t3 \\ 0 & 0 & 0 & 0 \\ \langle x \rangle & \langle x \rangle & -\langle x \rangle & 0 \\ 0 & \langle all - x \rangle & -\langle x \rangle & -\langle x \rangle \\ 0 & 0 & -\langle all - x \rangle & \langle x \rangle \end{pmatrix} \begin{matrix} \langle x \rangle.Repos + \langle x \rangle.Att \\ \langle x \rangle.Att \\ \langle x \rangle.Mess - \langle all - x \rangle.Att \\ \langle x \rangle.Ack \end{matrix}$$

Une seconde règle consiste à supprimer une ligne pour laquelle son coefficient est le seul coefficient non nul d'une colonne. Pour ce faire, afin de ne pas perdre de flot, il est nécessaire que ce coefficient soit une application injective⁴. L'application de cette règle permet de supprimer la ligne *Att*.

$$\begin{pmatrix} & t1 & t2 & t3 \\ 0 & 0 & 0 & 0 \\ 0 & \langle all - x \rangle & -\langle x \rangle & -\langle x \rangle \\ 0 & 0 & -\langle all - x \rangle & \langle x \rangle \end{pmatrix} \begin{matrix} \langle x \rangle.Repos + \langle x \rangle.Att \\ \langle x \rangle.Mess - \langle all - x \rangle.Att \\ \langle x \rangle.Ack \end{matrix}$$

⁴la fonction doit être injective de $Bag_Q(\mathcal{C}(t))$ dans $Bag_Q(\mathcal{C}(p))$. Ainsi, les fonctions sur $Bag_Q(C)$ $X_C, !X_C, All_C - X_C$, sont injectives, alors que la fonction All_C ne l'est pas.

En itérant l'application de ces deux règles on obtient successivement les deux matrices suivantes.

$$\begin{array}{ccc} t1 & t2 & t3 \\ \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -\langle all - x \rangle & \langle x \rangle \end{array} \right) & \begin{array}{l} \langle x \rangle.Repos + \langle x \rangle.Att \\ \langle x \rangle.Mess - \langle all - x \rangle.Att + \langle x \rangle.Ack \\ \langle x \rangle.Ack \end{array} \end{array}$$

$$\begin{array}{ccc} t1 & t2 & t3 \\ \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \end{array} \right) & \begin{array}{l} \langle x \rangle.Repos + \langle x \rangle.Att \\ \langle x \rangle.Mess - \langle all - x \rangle.Att + \langle x \rangle.Ack \end{array} \end{array}$$

Cette dernière matrice est telle que tous ses coefficients sont nuls et donc, chaque ligne correspond à un flot. Le premier décrit une structure de processus : un site x est soit au repos soit en attente d'acquittement. Le second, que nous avons déjà interprété, peut également indiquer qu'en oubliant l'information sur l'identité des sites alors le nombre de messages ou acquittements en circulation est égal à $N - 1$ fois le nombre de sites en attente ($\langle all \rangle.Mess + \langle all \rangle.Ack - (N - 1)\langle all \rangle.Att$).

2.3. Calcul de famille génératrice non paramétrée de flots

La méthode précédente permet d'obtenir des flots d'un réseau mais ne garantit pas que l'on obtient à coup sûr une famille génératrice. En effet, chaque ligne dont tous les coefficients sont nuls correspondra à un flot du réseau, mais rien n'assure que l'on peut toujours obtenir une matrice dont tous les coefficients seront nuls (ce qui garantit l'obtention d'une famille génératrice). En particulier, deux types de problèmes peuvent être rencontrés : on ne peut trouver de combinaison linéaire diminuant les coefficients non nuls ou bien encore, un coefficient non nul est isolé sur une colonne mais ce coefficient n'est pas une fonction injective. Dans ces deux cas, la méthode intuitive précédente ne peut poursuivre le calcul des flots du réseau. Considérons l'exemple suivant (Fig 15). La matrice d'incidence W de ce réseau est la suivante :

$$W = \begin{array}{ccc} & t1 & t2 \\ \left(\begin{array}{cc} -\langle x \rangle & \langle x \rangle \\ -\langle y \rangle & \langle x \rangle \\ \langle x \rangle + \langle y \rangle & -2 * \langle x \rangle \end{array} \right) & \begin{array}{l} \langle x \rangle.ProcType1 \\ \langle x \rangle.ProcType2 \\ \langle x \rangle.Travail \end{array} \end{array}$$

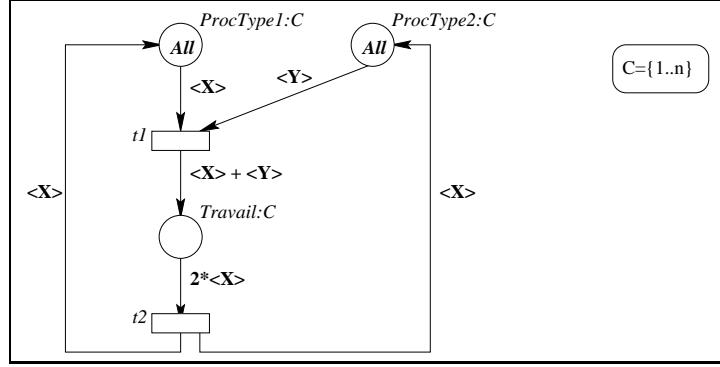


Figure 15. Un réseau coloré

Si l'on commence par annuler la seconde équation (correspondant à $t2$) l'application des règles précédentes conduit à la matrice suivante.

$$\begin{array}{cc} t1 & t2 \\ \begin{pmatrix} -\langle x \rangle + \langle y \rangle & 0 \\ -\langle x \rangle + \langle y \rangle & 0 \end{pmatrix} & \begin{pmatrix} \langle x \rangle.ProcType1 - \langle x \rangle.ProcType2 \\ 2 * \langle x \rangle.ProcType1 + \langle x \rangle.Travail \end{pmatrix} \end{array}$$

La différence de la première ligne avec la seconde donne alors à la matrice :

$$\begin{array}{cc} t1 & t2 \\ \begin{pmatrix} -\langle x \rangle + \langle y \rangle & 0 \\ 0 & 0 \end{pmatrix} & \begin{pmatrix} \langle x \rangle.ProcType1 - \langle x \rangle.ProcType2 \\ \langle x \rangle.ProcType1 + \langle x \rangle.ProcType2 + \langle x \rangle.Travail \end{pmatrix} \end{array}$$

La fonction $-\langle x \rangle + \langle y \rangle$ n'étant pas injective, il n'est pas possible d'aller plus loin.

Si l'on commence par annuler la première équation, on obtient immédiatement la matrice suivante, et de nouveau, on ne peut aller plus loin avec les règles que nous avons présentées précédemment.

$$\begin{array}{cc} t1 & t2 \\ \begin{pmatrix} -\langle x \rangle & \langle x \rangle \\ -\langle y \rangle & \langle x \rangle \\ 0 & 0 \end{pmatrix} & \begin{pmatrix} \langle x \rangle.ProcType1 \\ \langle x \rangle.ProcType2 \\ \langle x \rangle.Travail + \langle x \rangle.ProcType1 + \langle x \rangle.ProcType2 \end{pmatrix} \end{array}$$

Il est en fait possible d'obtenir un algorithme simple de calcul de famille génératrice de flots [Cou90] en modifiant les règles de l'élimination de Gauss par l'ajout d'une nouvelle règle. Celle-ci, basée sur la notion de semi-inverse généralisée, va permettre de diminuer à chaque étape le nombre de coefficients non nuls de la matrice.

La validité de cette transformation se justifie par le fait qu'elle peut être décomposée en trois règles de transformations élémentaires, chacune d'elle préservant la famille génératrice de flots [Cou90].

La première règle consiste à remplacer la colonne t par deux colonnes qui lui sont équivalentes. La première de ces colonnes est obtenue en composant la colonne t avec la fonction $(h \circ g)$ où h est la semi-inverse généralisée de g , et la seconde est obtenue en composant la colonne t avec $(Id - h \circ g)$. Comme $(h \circ g) + (Id - h \circ g) = Id$, on préserve la famille génératrice. De plus, par définition de h , on a $g \circ h \circ g = g$ et $g - g \circ h \circ g = 0$ ce qui justifie la valeur des coefficients encadrés.

$$\left(\begin{array}{cc} t.(h \circ g) & t.(Id - h \circ g) \\ f_1 \circ h \circ g & f_1 \circ (Id - h \circ g) \\ \vdots & \vdots \\ f_k \circ h \circ g & f_k \circ (Id - h \circ g) \\ \boxed{g} & \boxed{0} \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{array} \right) W_1 \begin{array}{c} \mathcal{F}_1 \\ \vdots \\ \mathcal{F}_k \\ \mathcal{F} \\ \mathcal{F}'_1 \\ \vdots \\ \mathcal{F}'_q \end{array}$$

La seconde transformation consiste à soustraire à chaque ligne $\mathcal{F}_i, i = 1..k$ la ligne \mathcal{F} pré-multipliée par $f_i \circ h$. De cette sorte, on élimine de la colonne $t.(h \circ g)$ tous les coefficients non nuls excepté celui de la ligne \mathcal{F} . Cette règle est une règle standard de transformation de l'algorithme de Gauss et préserve donc également la famille génératrice. Nous notons dans cette nouvelle matrice W_2 la matrice résultant de cette transformation sur W_1 .

$$\left(\begin{array}{cc} t.(h \circ g) & t.(Id - h \circ g) \\ 0 & f_1 \circ (Id - h \circ g) \\ \vdots & \vdots \\ 0 & f_k \circ (Id - h \circ g) \\ g & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{array} \right) W_2 \begin{array}{c} \mathcal{F}_1 - \mathcal{F} \circ f_1 \circ h \\ \vdots \\ \mathcal{F}_k - \mathcal{F} \circ f_k \circ h \\ \mathcal{F} \\ \mathcal{F}'_1 \\ \vdots \\ \mathcal{F}'_q \end{array}$$

La dernière transformation consiste à utiliser le fait que si h est la semi-inverse généralisée de g alors $(Id - g \circ h)$ génère toutes les solutions de l'équation $X \circ g = 0$. On peut donc remplacer la ligne \mathcal{F} par la ligne $(Id - g \circ h) \circ \mathcal{F}$ sans perdre de flots. On obtient alors la matrice suivante dans laquelle on a de plus

supprimé la colonne $t.(h \circ g)$ dont tous les coefficients sont devenus nuls.

$$\begin{pmatrix} t.(Id - h \circ g) \\ f_1 \circ (Id - h \circ g) \\ \vdots \\ f_k \circ (Id - h \circ g) \\ \boxed{0} \\ 0 \\ \vdots \\ 0 \end{pmatrix} W_3 \begin{pmatrix} \mathcal{F}_1 - \mathcal{F} \circ f_1 \circ h \\ \vdots \\ \mathcal{F}_k - \mathcal{F} \circ f_k \circ h \\ \mathcal{F} \circ (Id - g \circ h) \\ \mathcal{F}'_1 \\ \vdots \\ \mathcal{F}'_q \end{pmatrix}$$

Reprenons l'exemple de la figure 15. La matrice d'incidence initiale est la suivante ⁵ :

$$W = \begin{pmatrix} & t1 & & t2 \\ \langle x \rangle & -\langle x \rangle & \langle x \rangle & \\ \langle y \rangle & -\langle y \rangle & \langle x \rangle & \\ \langle x \rangle + \langle y \rangle & & -2 * \langle x \rangle & \end{pmatrix} \begin{pmatrix} \langle x \rangle.ProcType1 \\ \langle x \rangle.ProcType2 \\ \langle x \rangle.Travail \end{pmatrix}$$

Une semi-inverse de l'application linéaire $g : \langle x, y \rangle \mapsto \langle x \rangle + \langle y \rangle$ est l'application linéaire $h : \langle x \rangle \mapsto 1/2\langle x, x \rangle$. On a $h \circ g : \langle x, y \rangle \mapsto 1/2(\langle x, x \rangle + \langle y, y \rangle)$ et $g \circ h : \langle x \rangle \mapsto \langle x \rangle$. Si l'on note $f_1 : \langle x, y \rangle \mapsto -\langle x \rangle$ et $f_2 : \langle x, y \rangle \mapsto -\langle y \rangle$, on a par exemple

- $f_1 \circ h \circ g : \langle x, y \rangle \mapsto -1/2(\langle x \rangle + \langle y \rangle)$,
- $f_2 \circ h \circ g : \langle x, y \rangle \mapsto -1/2(\langle x \rangle + \langle y \rangle)$ ou encore,
- $\langle x \rangle - (-2)\langle x \rangle \circ f_1 \circ h = 0_{Bag(C) \rightarrow Bag(C)}$ c'est à dire l'application nulle de $Bag(C)$ dans $Bag(C)$ que l'on note 0.

La transformation conduit donc à la matrice W' (dans laquelle on peut supprimer la troisième ligne qui ne donne aucune solution).

$$W' = \begin{pmatrix} & t1 & & t2 \\ 1/2(\langle y \rangle - \langle x \rangle) & 0 & \langle x \rangle.ProcType1 + 1/2 * \langle x \rangle.Travail \\ 1/2(\langle x \rangle - \langle y \rangle) & 0 & \langle x \rangle.ProcType2 + 1/2 * \langle x \rangle.Travail \\ 0 & 0 & 0_{Bag(C) \rightarrow Bag(C)}.Travail \end{pmatrix}$$

Notons g l'application $\langle x, y \rangle \mapsto 1/2(\langle x \rangle - \langle y \rangle)$. Une semi-inverse de g est l'application $h : \langle x \rangle \mapsto 2\langle x, d \rangle$ où d est une valeur quelconque de C . En effet,

- $g \circ h : \langle x \rangle \mapsto (\langle x \rangle - \langle d \rangle)$,
- $h \circ g : \langle x, y \rangle \mapsto (\langle x, d \rangle - \langle y, d \rangle)$ et
- $g \circ h \circ g : \langle x, y \rangle \mapsto 1/2(\langle x \rangle - \langle d \rangle - \langle y \rangle + \langle d \rangle)$

⁵il faut prendre garde que le domaine de $t1$ est $C \times C$ et que donc $\langle x \rangle$ sur la colonne $t1$ est en fait la fonction $\langle x, y \rangle \mapsto \langle x \rangle$. Par contre, le domaine de $t2$ étant C , $\langle x \rangle$ sur la colonne $t2$ est bien la fonction $\langle x \rangle \mapsto \langle x \rangle$.

et donc $g \circ h \circ g = g$. La transformation conduit alors la matrice W'' :

$$W'' = \begin{pmatrix} t1 & t2 \\ 0 & 0 \end{pmatrix} \begin{matrix} \langle x \rangle.ProcType1 + (\langle x \rangle - \langle d \rangle).ProcType2 + (\langle x \rangle - \langle d \rangle).Travail \\ \langle d \rangle.ProcType2 + 1/2 * \langle d \rangle.Travail \end{matrix}$$

La matrice étant nulle, on a obtenu une famille génératrice. Cette famille peut se réécrire plus élégamment en :

$$\begin{aligned} & - \langle x \rangle.ProcType1 + \langle x \rangle.ProcType2 + \langle x \rangle.Travail \\ & - \langle d \rangle.ProcType1 - \langle d \rangle.ProcType2 \end{aligned}$$

L'algorithme précédent permet donc d'obtenir à coup sûr une famille génératrice d'un réseau coloré quelconque. Néanmoins, son utilisation pratique pose deux problèmes :

1. le calcul de la semi-inverse d'une application peut conduire à déplier partiellement le réseau. Ceci conduit donc à fixer la taille des domaines de couleur. Il n'est donc pas possible de déterminer a priori si l'on obtient une famille génératrice paramétrée ou non ;
2. si le réseau n'utilise qu'un sous-ensemble de fonctions de couleur (par exemple, uniquement l'identité et la fonction successeur), rien ne garantit que les coefficients intervenant dans la définition des flots soient eux-mêmes dans ce sous-ensemble. Ceci peut rendre difficile l'interprétation des flots trouvés et nécessiter d'opérer manuellement des transformations sur la famille génératrice trouvée.

Lorsque l'on utilise des sous-classes de réseaux colorés (réseaux réguliers, réseaux ordonnés, ..) on aura donc intérêt à utiliser un algorithme adapté à cette sous-classe lorsqu'il existe.

2.4. Famille génératrice paramétrée pour les réseaux réguliers

Nous nous intéressons dans cette section au calcul de flots dans les réseaux réguliers (sous-classes des réseaux bien-formés définis en section 4.5 du chapitre précédent). Pour l'exemple suivant (Fig 16) le flot $\mathcal{F} = \langle All \rangle p + \langle X \rangle q$ constitue une famille génératrice de flots et peut s'obtenir simplement à l'aide de l'algorithme général ou par un calcul intuitif.

Il existe cependant une autre écriture de cette famille génératrice à l'aide des deux flots suivants (on pourra vérifier que le flot \mathcal{F} peut effectivement être obtenu par combinaison linéaire de $\mathcal{F}1$ et $\mathcal{F}2$) :

- $\mathcal{F}1 = \langle X - Y \rangle q$ qui s'interprète, en tenant compte du marquage initial, $\forall x, y \in C1, q(x) - q(y) = 0$
- $\mathcal{F}2 = n * \langle All \rangle p + \langle All \rangle q$ avec $n = |C1|$ qui s'interprète $n * \sum_{x \in C1} p(x) + \sum_{x \in C} q(x) = n^2$

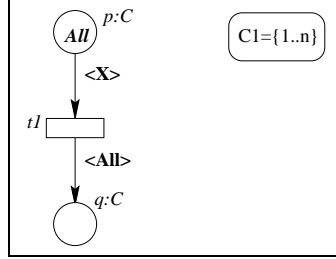


Figure 16. Un réseau régulier

Le premier de ces flots est de forme «différentielle», c'est à dire qu'il s'écrit $\langle X - Y \rangle (\sum_{p \in P} \lambda_p \cdot p)$ et permet de caractériser l'évolution du nombre de marques d'une couleur par rapport à une autre pour une combinaison linéaire de places. Le second de ces flots est de forme «homogène». Il s'écrit sous la forme $\langle All \rangle (\sum_{p \in P} \lambda_p \cdot p)$ et permet de compter l'évolution du nombre de marques dans un sous-ensemble de places (en oubliant l'information sur les couleurs).

Cette écriture possède plusieurs avantages. Tout d'abord l'interprétation de ces flots donne rapidement une information sur l'évolution possible des couleurs dans les places du réseau. Ainsi, $\mathcal{F}1$ traduit que la différence entre le nombre de marques de deux couleurs distinctes dans la place q est toujours constante (ici nulle) et le flot $\mathcal{F}2$ indique qu'il peut y avoir jusqu'à n^2 marques dans la place q . Ensuite, et c'est sans doute ici le point le plus important, on peut démontrer [Had87] que toute famille génératrice de flots d'un réseau régulier peut s'obtenir par application successive d'opérations de différenciation ou d'homogénéisation de couleurs. Ces deux opérations de dérivation vont produire à partir d'un réseau régulier à k classes de couleurs deux réseaux réguliers à $k - 1$ classes de couleurs. Si les réseaux obtenus ne sont plus colorés, on applique l'algorithme de Gauss étendu aux anneaux de polynômes à coefficients dans \mathbb{Z} pour calculer les flots, sinon on réitère les opérations de dérivation sur les réseaux obtenus.

Soient R un réseau régulier et C_1, \dots, C_k les classes de ce réseau. La première forme de dérivation de R par rapport à une classe C_i produit le réseau $D_i(R)$, la seconde forme de dérivation produit le réseau $S_i(R)$. Ces deux réseaux (réguliers) sont composés des mêmes places et transitions que R mais les valuations des arcs ont été modifiées de la façon suivante. Soient p une place et t une transition. On distingue deux cas :

1. le domaine de p contient C_i . Dans ce cas, la valuation de l'arc reliant p et t peut s'écrire $v = \langle f_1, \dots, f_q, \alpha_i \cdot X_i + \beta_i \cdot All_i, f_{q'}, \dots, f_{q''} \rangle$. Cette valuation est remplacée dans $D_i(R)$ par la valuation $D_i(v) = \alpha_i * \langle f_1, \dots, f_q, f_{q'}, \dots, f_{q''} \rangle$ tandis que v est remplacée dans le réseau $S_i(R)$ par la valuation $S_i(v) =$

$(\alpha_i + n_i \cdot \beta_i) * \langle f_1, \dots, f_q, f_{q'}, \dots, f_{q''} \rangle$ avec $n_i = |C_i|$.

- le domaine de p ne contient C_i . Dans ce cas, la valuation reliant p et t n'est pas modifiée.

Par itérations on finit par obtenir 2^k réseaux non colorés avec des valuations pouvant faire apparaître des polynômes à k variables (les tailles des classes n_1, \dots, n_k). On applique alors l'algorithme de Gauss sur chacun de ces réseaux

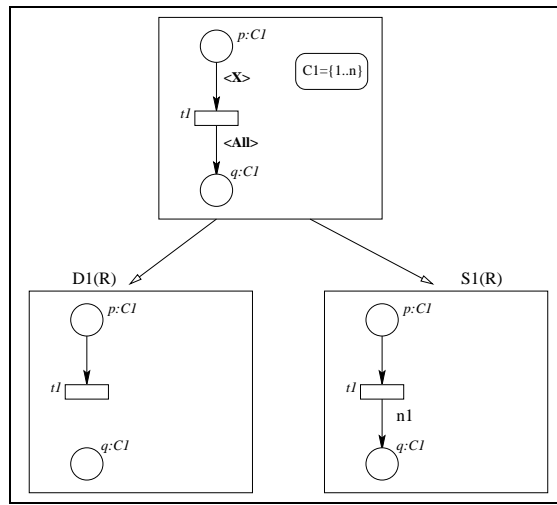


Figure 17. Dérivations du réseau régulier précédent

(ce qui peut conduire à définir des contraintes sur les n_i afin de distinguer les polynômes nuls) puis on «re-colore» les flots obtenus en tenant compte des opérations de dérivation effectuées. Ainsi un flot provenant d'un réseau dérivé $D_i(R)$ sera coloré par la fonction $\langle X_i - Y_i \rangle$ et un flot provenant d'un réseau dérivé $S_i(R)$ sera coloré avec la fonction All_i .

Le seul flot de $D1(R)$ est q qui re-coloré donne le flot $\langle X_1 - Y_1 \rangle q$. De même, le seul flot de $S1(R)$ est $n * p + q$ qui, après re-coloration, donne le flot $\langle All_1 \rangle \cdot (n * p + q)$.

2.5. Cas des flots à coefficients positifs

Il n'existe à ce jour qu'un seul algorithme connu de calcul de famille génératrice de flots positifs dans les réseaux colorés [CHP91]. La difficulté principale par rapport au calcul de flots vu précédemment provient du fait que $(\mathbb{Q}^+, +)$ n'est qu'un semi-groupe.

Cet algorithme, que nous exposons ici brièvement, permet de calculer une famille génératrice d'un réseau régulier mono-classe (une seule classe de couleur) ou d'un réseau prédicats-transitions mono-classe [MV85].

Considérons le cas des réseaux réguliers mono-classe et supposons de plus que chaque place est colorée par cette classe (si ce n'est pas le cas, il suffit de colorer les places non colorées par ce domaine et de remplacer toute valuation entière λ autour de cette place par la valuation $\lambda.All$).

Par définition, tout coefficient de la matrice d'incidence d'un tel réseau est de la forme $W(p, t) = \langle \alpha_{p,t}.X + \beta_{p,t}.All \rangle$. Notons alors A et B les matrices $P \times T$ définies par $A(p, t) = \alpha_{p,t}$ et $B(p, t) = \beta_{p,t}$.

Soit alors le système d'équations :

$$\begin{cases} (A + n.B). \sum_{i=1}^n X(i) = 0 \\ A.X(1) = A.X(2) = \dots = A.X(n) \end{cases} \quad [1]$$

où X un vecteur de $(\mathbb{Q}^+)^P$ avec $n = |C|$ représente l'inconnu du système et $X(i)$ la i -ème composante de X ($X(i)$ est un vecteur à coefficients dans \mathbb{Q}^+ indexé par P ; $X(i) \in (\mathbb{Q}^+)^P$).

Chaque solution X de ce système définit le flot à coefficients positifs de domaine de couleur C , $\mathcal{F} = \sum_{p \in P} F_p.p$ avec $F_p(c) = X(i)(p).c$ ($X(i)(p)$ désigne la p -ème composante de $X(i)$).

Le système d'équations (1) n'est qu'une réécriture du système d'équations associé au calcul des flots positifs d'un réseau régulier. Il vient donc immédiatement qu'une famille génératrice de ce système engendre une famille génératrice de flots positifs. La résolution paramétrée de ce système se fait en deux phases :

1. On résout d'abord le système $A.X(1) = A.X(2) = \dots = A.X(n)$. Pour cela, on calcule par itérations successives un ensemble fini de vecteurs de \mathbb{Q}^T (appelés direction) engendrant une famille génératrice de solutions de cette équation. A chaque direction b correspond un ensemble de vecteurs $Sol(A, b)$ défini par $Sol(A, b) = \{y \in (\mathbb{Q}^+)^P \mid A.y = b\}$. Par construction, chaque élément de l'ensemble $Gen(A, b) = \{X \in ((\mathbb{Q}^+)^P)^n \mid X(i) \in Sol(A, b)\}$ est une solution de l'équation étudiée.
2. Chaque famille de solution engendrée par une direction est mise sous la forme d'une somme formelle de vecteurs puis projetée sur la première équation $((A + n.B). \sum_{i=1}^n X(i) = 0)$. Le système engendré est alors résolu par une extension de l'algorithme de Farkas aux polynômes multi-variables. Chaque solution fournit alors un ensemble de flots positifs du réseau coloré.

Le lecteur intéressé par cet algorithme pourra se référer à [CHP91] et à [Pey93] pour une généralisation à plusieurs paramètres.

Bibliographie

- [Ber83] G. Berthelot. *Transformation et analyse de réseaux de Petri, applications aux protocoles*. Thèse d'état, Université Pierre et Marie Curie, Paris, 1983.
- [Ber86] G. Berthelot. Transformations and decompositions of nets. In *Advances in Petri Nets*, number 254 in LNCS, pages 359–376. Springer-Verlag, 1986.
- [BRA83] G.W. BRAMS. *Réseaux de Petri : Theorie et pratique*. Masson, 1983.
- [CH88] J.M. Couvreur and S. Haddad. Towards a general and powerful computation of flows for parameterized coloured nets. In *9th European Workshop on Application and Theory of Petri Nets*, volume II, Venice (Italy), June 1988.
- [CHP91] J.M. Couvreur, S. Haddad, and J.F. Peyre. Computation of generative families of semi-flows in two types of colored net. In *proc of the 12th International Conference on Application and Theory of Petri-Nets*, Aarhus, Denmark, June 1991.
- [CMS86] J.M. Colom, J. Martinez, and M. Silva. Packages for validating discrete production systems modeled with Petri nets. In *IMACS-IFAC Symposium*, Lille, France, 1986.
- [Cou90] J.M. Couvreur. The general computation of flows for coloured nets. In *proc of the 11th International Conference on Application and Theory of Petri-Nets*, Paris, France, June 1990.
- [Gen91] H.J. Genrich. Equivalence transformations of prt-nets. In Jensen and Rozenberg, editors, *High-level Petri Nets, Theory and Application*, pages 426–453. Springer-Verlag, 1991.
- [Had87] S. Haddad. *Une catégorie régulière de réseau de Petri de haut niveau : définition, propriétés et réductions. Application à la validation de systèmes distribués*. PhD thesis, Université Pierre et Marie Curie, Paris, 1987.
- [Had91] S. Haddad. A reduction theory for colored nets. In Jensen and Rozenberg, editors, *High-level Petri Nets, Theory and Application*, LNCS, pages 399–425. Springer-Verlag, 1991.
- [Mur90] T. Murata. Petri nets : properties, analysis and applications. In *proceedings of the IEEE Vol 6*, number 1, pages 39–50, January 1990.
- [MV85] G. Memmi and J. Vautherin. Computation of flows for unary-predicates/transition nets. *Lecture Notes in Computer Science : Advances in Petri Nets 1984*, 188 :455–467, 1985.
- [Nas76] M. Nashed. Generalized inverses and applications. M. Z. Nashed, ed., *Generalized Inverses and Applications*, Academic Press, New York, 1976.

- [Pey93] J.F. Peyre. *Résolution paramétrée de systèmes linéaires ; Application au calcul d'invariants positifs dans les réseaux colorés, à la validation formelle et à la génération de code*. PhD thesis, Université Pierre et Marie Curie, Paris, 1993.
- [PPP00] D. Poitrenaud and J.F. Pradat-Peyre. Pre and post-agglomerations for *LTL* model checking. In M. Nielsen and D Simpson, editors, *High-level Petri Nets, Theory and Application*, number 1825 in LNCS, pages 387–408. Springer-Verlag, 2000.
- [Vau87] J. Vautherin. Calculation of semi-flows for pr/T-systems. In *Int. Workshop on Petri Nets and Performance Models, Madison, Wisconsin*, pages 174–183, Washington, 1987. IEEE Computer Society Press. NewsletterInfo : 29.