Des bisimulations de places pour la réduction des réseaux de Petri

Wilfried Quivrin-Pfister

THÈSE

présentée par

W. Quivrin-Pfister

pour obtenir le grade de Docteur
de l'Institut National Polytechnique de Grenoble
(Arrêté ministériel du 30 mars 1992)

Spécialité : informatique

Des bisimulations de places pour la réduction des réseaux de Petri

Soutenue le 28 novembre 1995

J. Sifakis	Président
R. De Simone S. Haddad	Rapporteurs
G. Berthelot Ph. Schnoebelen	Examinateurs

Thèse préparée au sein du **L**aboratoire d'**I**nformatique **F**ondamentale et d'**I**ntelligence **A**rtificielle

In memoriam Pascale Quivrin

Remerciements

« C'est l'honneur des individus que de
travailler à une œuvre qui les dépasse
et dont ils ne verront pas l'achèvement. »
J. JAURÈS

Honneur classique, je tiens à remercier tout d'abord les membres du jury, qui ont accepté de m'écouter anonner les quelques résultats de ces trois années :

- J. Sifakis, qui a aussi accepté de le présider ;
- S. Haddad et R. de Simone qui ont bien voulu lire et corriger la première épreuve de cette thèse. J'espère qu'en fait d'épreuve, ils n'ont pas trop souffert.
- G. Berthelot qui, je l'espère, sera convaincu à la fin de cet exposé.

Particulièrement merci à Philippe Schnoebelen, guide spirituel[1] , qui m'a laissé découvrir avec une grande liberté la recherche, ses joies et ses vicissitudes. Il m'a aussi indiqué ICS, mais c'est une autre histoire...

Ce serait évidemment incomplet si je ne remerciais les membres de l'équipe «Parallélisme», particulièrement C. Autant qui rédigeait sa thèse lorsque j'arrivais, et qui malgré tout à bien voulu guider Candide au pays des réseaux de Petri, d'UNIX, de C++ et d'Internet.

Une fois n'est pas coutume, je tiens à souligner l'importance des «administratifs», médiathèque en tête, pour le soutien qu'ils apportent à tout chercheur. Sincèrement, merci.

Toute ma gratitude va à ceux qui m'ont aidé, poussé et tiré tour-à-tour, me remettant sur les rails lorsque je flânais trop.

Le monde est petit, vu d'une console. Où plutôt, Internet est un monde dans le monde où étrangement le temps est délivré de la distance[2]. J'étais

[1] Mais oui, mais oui...

[2] Quoique certains `traceroute` m'ont fait me demander si l'on n'avait pas codé l'algorithme de recherche du plus *long* chemin entre deux points dans les protocoles de communication...

souvent à une extrémité du câble, pour trouver des indications ou des solutions, ou tout simplement pour satisfaire une terrible curiosité. À tous les cybernautes aussi il faut que je dise merci, je souhaite simplement leur avoir aussi rendu service.

Je ferais une mention spéciale à la liste de discussion HotDocs, qui contribuera, j'en suis sûr, à faire évoluer le statut du doctorant. La réflexion est lancée, les idées arrivent, leur application viendra.

Surtout je suis heureux que Géraldine, Oriane, Anastasia et Maxence forment tout simplement ma famille...

Table des matières

Liste des figures

Introduction

Entre les années trente et aujourd'hui l'informatique fondamentale a fortement évolué, passant des systèmes séquentiels (qui ont donné le formalisme de TURING) aux systèmes parallèles (avec des formalismes tels que ceux de KARP & MILLER). Informellement, nous pouvons identifier les premiers à un individu isolé, les seconds à un ensemble d'éléments interactifs (communiquant, se coordonnant ou coopérant) et hétérogènes (où interviennent ordinateurs, machines, humains, etc).

«Système distribué» ou «système parallèle» ? Il s'agit de deux réalisations du même objet. La première expression s'attache à voir une tâche que l'on a répartie, la seconde observe les éléments qui concourent à la réalisation de cette tâche. Historiquement, on peut dire que les systèmes distribués ont vu le jour lorsqu'on a relié des ordinateurs par réseau pour les faire travailler de conserve (donc avec une certaine idée d'hétérogénéité et de dispersion, les machines étant non nécessairement identiques et dans des lieux distincts). Les systèmes parallèles sont nés de la volonté de mettre dans une seule machine plusieurs processeurs et de les faire travailler ensemble (donc avec une certaine idée d'homogénéité et de centralisation, les processeurs étant a priori semblables et dans un même boîtier). Les problèmes posés par les deux types de systèmes sont différents en pratique, mais semblables en théorie. En fait, nous pouvons définir un tel système (parallèle ou distribué) comme *un ensemble d'agents relativement autonomes qui se coordonnent pour effectuer certaines tâches.*

On notera que cette différence devient cruciale si nous considérons le «parallélisme physique» (lié au problème de la distribution des tâches sur différents processeurs), puisque les temps de communication (entre autres) deviennent une donnée importante. Dans cette thèse, nous ne nous intéressons qu'à leur base commune : le «parallélisme logique» (étudiant la possibilité de décrire un programme par différentes tâches indépendantes).

Une *théorie du parallélisme* va donc s'attacher à définir un cadre mathématique pour décrire ces systèmes et à en étudier les comportements à l'aide d'un modèle formel. Très informellement, les comportements du systèmes sont l'ensemble des exécutions possibles. Les buts de cette théorie seront d'une part de développer des outils de conception et de réalisation, d'autre

part de permettre de vérifier la «consistance» d'une conception et de vérifier que la réalisation correspond aux spécifications, i.e. que les *comportements* du systèmes sont conformes à ceux prévus par le *modèle*. La préférence donnée aux modèles mathématiques s'explique aisément par la possibilité de démonstration.

De ce point de vue, deux classes de modèles se distinguent : les modèles *comportementaux* qui s'intéressent au comportement de la machine et les modèles *structuraux* qui s'attachent au système lui-même.

Diverses théories ont été développées pour les systèmes parallèles, parmi lesquelles les réseaux de Petri.

Du parallélisme

Tout d'abord, précisons les concepts qui sont à la base du parallélisme. Les caractéristiques fondamentales d'un système parallèle sont aujourd'hui bien définies (voir [RT86, Old91b, Hoa94]) :

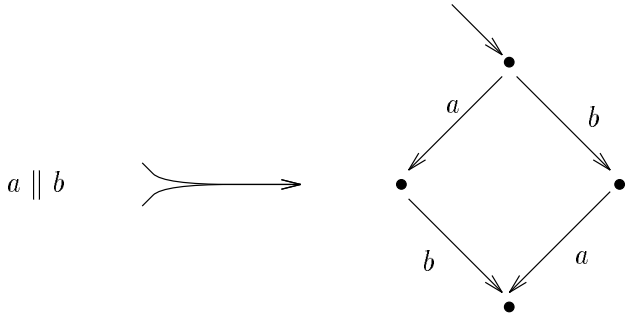
parallélisme : c'est la faculté d'exécuter différentes actions simultanément, en les synchronisant éventuellement. Par exemple, en décrivant les programmes par des prédicats, la parallélisation est la conjonction des programmes et ils se coordonnent sur les actions communes.

non-déterminisme : c'est la capacité du système de prendre en compte ses défaillances. Dans le cas des description par prédicat, il est assimilé à la disjonction.

causalité : c'est le complément du parallélisme. Il permet de définir les liens entre les actions et leur causes ou conséquences.

Existe-t-il un paradigme du parallélisme ? Tous les modèles ne sont pas toujours directement comparables puisqu'ils ne formalisent pas les même notions. Nous pouvons cependant donner une classification informelle selon deux paramètres fondamentaux :

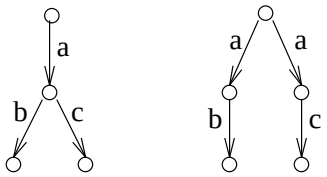
- Un paramètre est la manière de représenter le parallélisme. On parle souvent de «faux» et de «vrai» parallélisme. Grossièrement, il existe deux interprétations le parallélisme.
  - Une première consiste à dire que deux actions ne peuvent avoir lieu que séquentiellement, une relation de causalité ne peut alors être exprimée. C'est une sémantique dite «d'entrelacement». Elle est essentiellement basée sur un axiome «d'instantanéité» des transitions. Elle vient de la théorie des automates où la seule manière de représenter le parallélisme est le *non-déterminisme séquentiel*, puisqu'un état est non-distribué comme sur la figure ci-dessous :



À l'argument donné dans [CMP87] (et montrant qu'on ne peut raffiner dans une sémantique d'entrelacement), nous pouvons ajouter l'émergence de modèles prenant en compte la *durée* des actions (comme les réseaux de Petri temporisés [Zub80]).

– La seconde interprétation consiste à fixer un ordre partiel entre les actions reflétant les relations de causalité ou d'indépendance et rend donc explicites les points de choix dans une exécution. C'est une *sémantique causale*.

- Un autre paramètre est la façon de voir le futur d'un calcul (moments des choix). Nous parlerons alors de «temps linéaire» ou de «temps arborescent». Dans le premier cas, on considère qu'une exécution est complètement déterminée par la suite des actions qui la composent. Dans le second, on retient aussi les moments où des choix ont été faits. Un exemple classique pour les différentier est le suivant :



Dans cet exemple, les deux systèmes peuvent faire ab ou ac. Cependant, dans le premier cas, c'est seulement *après* avoir fait a que l'on choisit de faire b ou c. .

Une classification simpliste des modèles peut donc être envisagée selon la sémantique (d'entrelacement ou causale) et une temporalité (linéaire ou arborescente).

Des modèles du parallélismes

Nous présentons les modèles abstraits les plus étudiés dans la littérature[3] :

[3]Ces modèles ont des origines historiques bien distinctes, mais ils ont évolué et aujourd'hui chacun a des variantes où se retrouvent les idées des autres.

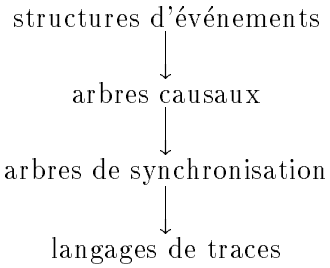
le langage des traces de Hoare [Hoa85] sont un modèle d'entrelacement et de temps linéaire. Le comportement d'un processus est décrit par les mots du langage qu'il génère, par conséquent, il ne rend compte que de l'ordre des actions.

les arbres de synchronisation de Milner [Mil80] sont un modèle d'entrelacement et de temps arborescent. Ce sont des arbres dont les branches sont étiquetées par les actions : les points de choix et l'ordre des actions sont donc explicites.

les «pomsets» de Pratt [Pra86] sont un modèle de sémantique causale et de temps linéaire. Les dépendances causales y sont clairement représentées par un ordre partiel entre les actions (relation de *causalité*). Une exécution est alors vue comme une suite de multi-ensembles d'actions indépendantes, respectant la relation de causalité.

les structures d'événements [Win87] sont un modèle de sémantique causale et de temps arborescent. À une relation de causalité, on ajoute une notion de *conflit* : deux événements sont en conflits si l'occurrence de l'un est exclusive de celle de l'autre.

[DGV93] relie ces différents modèles :



Nielsen, Plotkin et Winskel dans [NPW81], relient les structures d'événements à une classe de réseaux de Petri. De fait, tous ces modèles peuvent être utilisés pour analyser et représenter le comportement des réseaux.

Dans [HKT92], Hoogers, Kleijn et Thiagarajan montrent que les réseaux saufs, les structures d'événements et les réseaux de Petri forment une suite strictement croissante quant à leur pouvoir d'expression.

Les réseaux de Petri étiquetés

Les réseaux de Petri, introduits dans [Pet62], sont un des modèles formels les plus populaires pour les systèmes concurrents, tant d'un point de vue pratique ([Rei92, MAS92, Ber93, PX95]) que d'un point de vue théorique (plus de 5000 références dans la dernière compilation de la «Petri Net Newsletter»).

Plusieurs raisons font que les réseaux de Petri sont un modèle agréable pour le parallélisme :

- Leur simplicité et le fait que le parallélisme y est exprimé de manière naturelle. De ce fait ils sont fréquemment utilisés comme une base sémantique pour l'interprétation de langages parallèles (p.ex. le langage Γ de Banâtre et Le Métayer, CHAM de Berry et Boudol, des fragments du π-calcul de Milner[4]).

- Ainsi que l'ont montré Winskel dans [Win87] ou Meseguer et Montanari dans [Mes90], les réseaux de Petri ont une structure algébrique très simple.

- Il est relativement facile d'enrichir leur structure afin de modéliser des systèmes plus complexes (arcs inhibiteurs, temporisation, etc).

- Enfin, ils sont un modèle très performant en pratique (voir p.ex. [Rei92, PX95]). En particulier, ils permettent le raffinage et la décomposition en sous-systèmes.

Informellement, les réseaux de Petri sont un graphe dont certains nœuds (les *transitions*) représentent les actions élémentaires du système et les autres (les *places*) ses ressources.

Graphiquement. les places sont représentées par des cercles et les transitions par des rectangles portant une *étiquette*. Cette étiquette indique ce qu'un observateur extérieur verra lorsque le système franchira cette transition.

La figure 0.1 schématise le système «rédiger sa thèse».

Les ressources effectivement présentes dans le systèmes sont représentées par des *jetons* distribués dans les places.

Pour franchir une transition, le système doit avoir des ressources suffisantes. P.ex., pour `soutenir`, il faut que les *deux* rapporteurs acceptent[5].

Les réseaux de Petri expriment clairement les différents concepts de base (voir p. , p. 14) en pouvant utiliser une sémantique d'entrelacement ou causale (voir la figure 0.2).

Les méthodes de réduction dans les réseaux de Petri

La réduction est une étape importante entre la production d'un réseau de Petri et sa vérification, puisqu'elle permet en général des gains en complexité lors de cette dernière étape. Si le problème est bien cerné pour les systèmes de transitions (avec des algorithmes efficaces, voir p.ex. [Fer89] et le système

[4]Voir respectivement [BM93], [BB] et [Eng93].

[5]Et typiquement, il ne faut pas épuiser son directeur de thèse avant d'avoir soumis aux rapporteurs...

Figure 0.1 : Rédiger sa thèse.

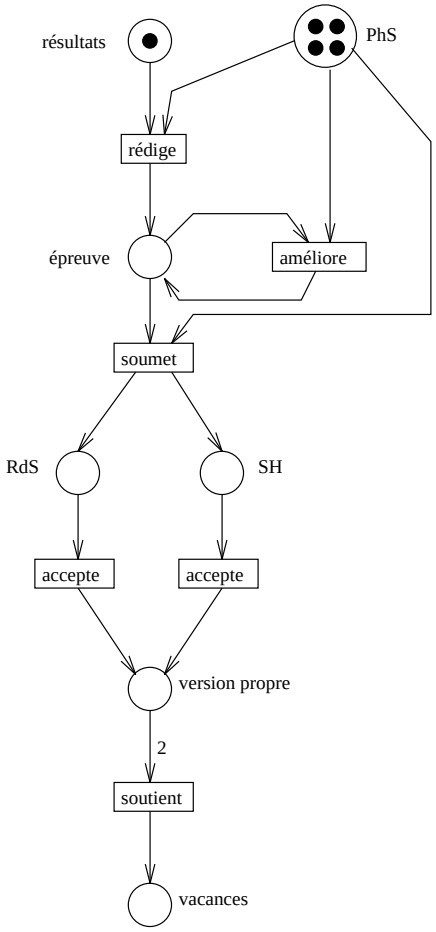
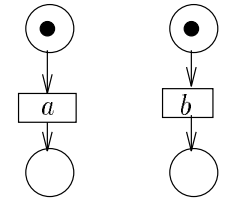
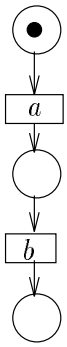


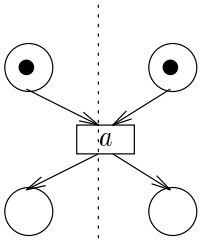
Figure 0.2 : Les concepts de base exprimés dans les réseaux de Petri



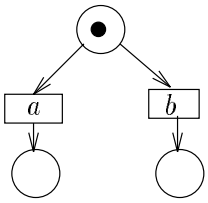
Parallélisme : $a \parallel b$



Séquentialité : $a; b$



Communication : $a \parallel_a a$



Non-déterminisme : $a + b$

ALDÉBARAN), il a fait l'objet de peu d'études pour les réseaux de Petri ([Ber85, Had90, SM83, LFB87]).

Le problème de la réduction des réseaux a d'abord été considéré comme le moyen de réduire l'espace des états sans modifier un certain nombre de propriétés (telles que «être borné», «être vivant», etc) afin de faciliter leur étude.

Ces travaux (e.g. [Ber85]) cherchent aussi à utiliser les réductions comme des *algorithmes* permettant d'évaluer lesdites propriétés.

Nous nous plaçons ici dans une toute autre optique, puisque nous allons chercher à créer une méthode de réduction non pas en voulant conserver un ensemble de propriétés, mais préservant le *comportement* du réseau.

Il existe une autre différence entre les méthodes de réductions citées ci-dessus de la notre : leur «localité». En effet, elles ne considèrent le réseau que localement (i.e. un sous-réseau assez petit). Ceci entraîne un coût algorithmique élevé, voire une automatisation impossible (comme pour [SM83]). Nous y reviendrons plus en détail dans la conclusion.

Réduire et rester «équivalent»

Le choix d'une méthode de réduction dépend de l'ensemble des propriétés à conserver. Nous avons eu pour objectif de préserver le comportement du réseau, donc de fait un grand nombre de propriétés.

Il existe une myriade d'équivalences comportementales dans la littérature (voir par exemple [CH93b]). Nous avons choisi la bisimulation pour diverses raisons :

- une équivalence respecte le temps arborescent ssi elle est plus fine ou égale à la bisimulation ([Gla94]) ;
- elle est liée aux logiques temporelles (particulièrement à HML [HM85]) ;
- il existe de nombreuses variantes (pour les actions invisibles [Wei89], pour le raffinage [Vog92], pour les sémantiques causales) ;
- dans la cas des automates (où la bisimulation a d'abord été définie), la bisimilarité se décide plus facilement que des équivalences du temps linéaires ;
- il existe une littérature abondante et c'est une notion bien cernée maintenant.

Il doit bien être clair dans l'esprit du lecteur que si la bisimulation classique (entre les marquages) permet de dire si deux réseaux sont équivalents, notre méthode, basée sur la *bisimulation de places*, cherche quant à elle à *réduire* des réseaux en ayant la bisimulation (classique) comme *critère de correction*.

Historique de la bisimulation de places

C'est Olderog qui le premier a proposé une notion de bisimulation de places dans [Old89, Old91a] où il utilise le terme de «strong bisimulation». Même si sa présentation mettait en avant d'autres objectifs, Olderog proposait une méthode simple permettant de prouver que deux réseaux avaient les mêmes processus concurrents. Pour cela, Olderog proposait de mettre en relation les places des deux réseaux, d'une façon vérifiant certaines contraintes locales.

Dans [ABS91a], la proposition d'Olderog est corrigée et les propriétés de la notion (revue) de bisimulation de places sont étudiées en détail. En particulier, il y est montré comment la bisimulation de place permet de réduire les réseaux de Petri. C'est dans [AS92] que les applications à la réduction sont mises en avant, et qu'on montre comment la bisimulation de places préserve aussi certains aspects causaux des comportements. Un outil informatique a été réalisé pour implémenter les algorithmes ([Pfi92]).

Dans [APS94], la bisimulation de places est étendue à la τ-bisimulation, permettant de prendre en compte les actions invisibles du systèmes. Enfin, [Pfi94] présente les résultats obtenus dans le cadre des réseaux à arcs inhibiteurs.

Structure et objectifs de la thèse

La thèse est organisée en quatre parties :

1. Après un rappel sur la structure et le comportement des réseaux de Petri, le chapitre 2 établit le cadre général de l'étude. Nous y fixons nos objectifs et montrons quelques résultats très généraux. Particulièrement, nous montrons que la «meilleure» réduction (notée $\sim$) n'est pas calculable. Nous présentons ensuite la bisimulation de places stricte qui est incluse dans $\sim$ et calculable efficacement.

2. Nous étendons alors le concept de la bisimulation de places au cas des actions invisibles. Le chapitre 4 et l'annexe B présentent la τ-bisimulation de places, basée sur la τ-bisimulation ([Mil80]). Nous étudions au chapitre 5 les bisimulations de branchement ([GW89b, Wei89]) qui ne donnent pas de résultats aussi satisfaisants.

   Nous terminons cette partie par les réseaux à arcs inhibiteurs. Pour ces réseaux, les algorithmes ont un coût nettement plus élevé (exponentiel au lieu de polynomial).

3. Nous étudions ensuite brièvement quelques liens entre la bisimulation de places et la théorie des réseaux de Petri. Particulièrement, nous abordons la construction de réseaux et montrons qu'à l'exception du

raffinement, notre méthode se combine bien avec les opérateurs classiques (mise en parallèle, renommage et abstraction). Enfin, nous présentons brièvement le logiciel `PetriS`qui implémente les divers algorithmes.

4. Nous terminons avec quelques annexes, dont une dédiée aux problèmes ouverts.

Notes bibliographiques

L'introduction de [Hoo94] en fait un historique très complet de la genèse des réseaux de Petri.

[RT86] est un article assez ancien mais complet sur les réseaux à *arcs simples* (i.e. dont les arcs n'ont pas de poids.

Les propriétés mathématiques des réseaux de Petri sont largement abordées dans [Reu89].

[PX95], présente les réseaux d'une manière très appliquée. [Rei92], présente d'une manière plus «théorique» leur utilité pour la conception de systèmes.

La réduction des réseaux de Petri a été abordée dans divers papiers ([Ber86, Had90, SM83, LFB87]), mais souvent avec une optique différente : réduire l'espace d'états ou vérifier qu'un réseau possède telle ou telle propriété.

Première partie

Le cas classique

Chapitre 1

Les réseaux de Petri étiquetés

Ce chapitre introduit les différentes notions liées aux réseaux de Petri et à leur sémantique. Nous mettons l'accent sur les concepts utiles pour cette thèse mais n'hésitons pas à déborder du cadre fixé pour présenter quelques points d'intérêt.

Les réseaux de Petri permettent de modéliser les systèmes concurrents par un formalisme séparant actions, états et interrelation entre les deux. De cette manière, la structure et le comportement des systèmes sont décrits par le même formalisme.

Les deux premières sections fixent les notations générales. En particulier celles concernant les *multi-ensembles* qui sont le cadre mathématique choisi pour modéliser le comportement des réseaux.

Les deux sections suivantes présentent la structure et le comportement des réseaux de Petri tels que nous les utiliserons tout au long de ce manuscrit. Nous nous intéresserons particulièrement aux réseaux étiquetés : en effet, notre méthode de réduction est fortement lié au langage **CCS** de MILNER et à son équivalence observationnelle (voir [Mil80]). Nous confrontons parfois nos notations à d'autres trouvées dans la littérature afin de les motiver. Nous rappelons aussi quelques sémantiques pour les réseaux, particulièrement celle des processus dont nous aurons besoin ultérieurement.

Nous terminons par quelques notes bibliographiques.

1.1 Préliminaire mathématique

Nous précisons simplement ici les quelques notations de base que nous utiliserons sur les ensembles et les relations.

Notations 1.1.1. *Ensembles*

Soient X et Y deux ensembles, nous noterons $X \subseteq Y$ si Y est un sous-ensemble de X et $X \subsetneq Y$ ou $X \subset Y$ si X est un sous ensemble propre de Y.

$|X|$ est le cardinal *de X.*

Nous notons $\mathbb{N}$ l'ensemble des entiers naturels et $\mathbb{N}^+ = \mathbb{N} - \{0\}$. Nous noterons $\mathbb{Z}$ celui des relatifs.

Notations 1.1.2. *Relations et fonctions*

Une relation R sur un ensemble[1] *X est un sous-ensemble de $X \times X$. Nous utiliserons les notations suivantes :*

$id_X \stackrel{\text{déf}}{=} \{(x,x) \mid x \in A\}$ *est la relation d'identité ;*

$R^{-1} \stackrel{\text{déf}}{=} \{(y,x) \mid (x,y) \in R\}$ *est l'inverse de R ;*

R^k *, pour $k \in \mathbb{N}$ est définie récursivement par $R^0 \stackrel{\text{déf}}{=} id_X$ et, pour $k > 1$,* $R^k \stackrel{\text{déf}}{=} R^{k-1} \circ R \stackrel{\text{déf}}{=} \{(x,z) \mid \exists y \in X \ : \ (x,y) \in R^{k-1} \text{ et } (y,z) \in R\}$ *;*

$R^+ \stackrel{\text{déf}}{=} R^1 \cup R^2 \cup \ldots$ *la fermeture transitive de R ;*

$R^* \stackrel{\text{déf}}{=} id_A \cup R^+$ *la fermeture transitive et réflexive de R ;*

$\widetilde{R} \stackrel{\text{déf}}{=} (R \cup R^{-1})^*$ *la fermeture réflexive, symétrique et transitive de R. C'est la plus petite relation d'équivalence contenant R.*

Soit E une relation d'équivalence sur X, nous noterons $[x]_E$ la classe de $x \in X$ pour E et, par abus de notation, le représentant canonique de cette classe.

Pour une fonction $f : X \to Y$, nous définissons :

$codom(f) = \{y \in Y \ : \ \exists x \in X,\ y = f(x)\}$ *;*

$dom(f) = \{x \in X \ : \ \exists y \in Y,\ f(x) = y\}$.

Les séquences joueront un rôle particulier. Nous considérerons principalement des séquences finies, qui sont isomorphes à des mots sur un alphabet.

Notations 1.1.3. *Séquences*

Soit A un ensemble, une séquence finie *sur A est une application de $\{1,\ldots,n\}$ dans A. L'application $\varepsilon : \emptyset \to A$ est appelée la* séquence vide. *Nous représenterons la séquence finie $\rho : \{1,\ldots,n\} \to A$ par la suite $a_1\, a_2\, \ldots\, a_n$*

[1] Nous nous restreignons volontairement aux relations d'un ensemble sur lui-même puisqu'elles seules nous intéresserons.

d'éléments de A, où $a_i = \rho(i)$, pour $i \in \{1, \ldots, n\}$. n est la longueur *de ρ, notée $|\rho|$. La séquence vide ε est de longueur 0.*

Une séquence infinie *est une application $\rho : \mathbb{N}^+ \to A$.*

Nous noterons A^* l'ensemble des séquences finies sur A et A^ω celui des séquences infinies. Nous noterons $A^\infty = A^* \cup A^\omega$.

Si $\rho = a_1\, a_2\, \ldots\, a_n$ et $\rho' = b_1\, b_2\, \ldots\, b_m$ sont deux séquences, nous définissons leur concaténation *par $\rho.\rho' = a_1\, a_2\, \ldots\, a_n\, b_1\, b_2\, \ldots\, b_m$.*

Si ρ est une séquence finie, la séquence ρ^k est définie récursivement par :

$$\begin{array}{rcl} \rho^0 & \stackrel{\text{déf}}{=} & \varepsilon \\ \rho^{i+1} & \stackrel{\text{déf}}{=} & \rho^i\, \rho \end{array}$$

Le support *de $\rho = a_1\, a_2\, \ldots\, a_n$ est son codomaine. Il est noté $\mathcal{S}(\rho)$. Il est défini récursivement par :*

$$\begin{array}{rcl} \mathcal{S}(\varepsilon) & \stackrel{\text{déf}}{=} & \emptyset \\ \mathcal{S}(\rho u) & \stackrel{\text{déf}}{=} & \mathcal{S}(\rho) \cup \{u\} \end{array}$$

La restriction *de ρ à un ensemble $B \subseteq A$, notée $\rho|_B$ est définie récursivement par :*

$$\begin{array}{rcl} \varepsilon|_B & \stackrel{\text{déf}}{=} & \varepsilon \\ (a\,\rho)|_B & \stackrel{\text{déf}}{=} & \begin{cases} a(\rho|_B) & si\ a \in B \\ \rho|_B & sinon \end{cases} \end{array}$$

Le nombre d'occurrences *de $a \in A$ dans ρ, noté $occ_\rho(a)$ est défini récursivement par :*

$$\begin{array}{rcl} occ_\varepsilon(a) & \stackrel{\text{déf}}{=} & 0 \\ occ_{(x\,\rho)}(a) & \stackrel{\text{déf}}{=} & \begin{cases} occ_\rho(a) + 1 & si\ x = a \\ occ_\rho(a) & sinon \end{cases} \end{array}$$

Une autre représentation très utilisée des séquences sont les vecteurs (dans la mesure où l'ordre des éléments n'est pas pertinent).

Notations 1.1.4. *Relations d'ordre*

Un ordre partiel *$(X, <)$ est constitué par un ensemble X et une relation irréflexive et transitive $<$. Un* ensemble partiellement ordonné avec étiquetage dans un alphabet *$\mathcal{A}$ est un triplet $(X, <, l)$, où $(X, <)$ est un ordre partiel et $l : X \to \mathcal{A}$ une fonction d'étiquetage.*

Deux ensembles partiellement ordonnés $(X_1, <_1, l_1)$ et $(X_2, <_2, l_2)$ sont dits isomorphes *s'il existe une bijection $f : X_1 \to X_2$ qui respecte la relation d'ordre et l'étiquetage.*

1.2 Les multi-ensembles

C'est le choix du cadre mathématique naturel pour représenter le comportement des réseaux. Nous aurions pu prendre les graphes ([Old91b]) ou les hypergraphes ([AFN92]) ou d'autres encore. Dans un multi-ensembles, un élément peut apparaître plusieurs fois (on utilise parfois la terminologie de "sac").

Définition 1.2.1. *Multi-ensembles*

Étant donné un ensemble X, un multi-ensemble *sur X est une fonction $M : X \to \mathbb{N}$. Pour tout $x \in X$, nous appellerons $M(x)$ la* multiplicité *de x dans M.*

Nous noterons $\mathcal{M}(X)$ l'ensemble des multi-ensembles sur X.

Nous dirons qu'un multi-ensemble est fini *ssi pour tout $x \in X$, $M(x) = 0$ sauf pour un nombre fini d'entre eux.*

Nous noterons le multi-ensemble vide *de la même manière que l'ensemble vide : $\emptyset$.*

La notation mathématique classique des multi-ensembles est la double accolade $\{\{x_1, x_1, x_1, \ldots x_n, x_n\}\}$, chaque x_i étant répété $M(x_i)$ fois. Cependant, partout où il n'y aura pas de risques de confusion, nous nous contenterons d'accolades simples.

Nous pouvons maintenant définir les opérations usuelles sur les multi-ensembles :

Notations 1.2.1. *Opérations sur les multi-ensembles*

Nous donnons ci-dessous le tableau des opérateurs qui seront utilisés par la suite, les M_i étant des multi-ensembles sur un même ensemble X :

$$
\begin{array}{rcl}
M_1 = M_2 & \stackrel{\text{déf}}{\iff} & \forall x \in X \ : \ M_1(x) = M_2(x) \\
M_1 \subseteq M_2 & \stackrel{\text{déf}}{\iff} & \forall x \in X \ : \ M_1(x) \leq M_2(x) \\
M_1 \subsetneq M_2 & \stackrel{\text{déf}}{\iff} & M_1 \subseteq M_2 \wedge \exists x \in X \ : \ M_1(x) < M_2(x) \\
\text{ou} \quad \begin{array}{l} M_1 = M_2 + M_3 \\ M_1 = M_2 \cup M_3 \end{array} & \stackrel{\text{déf}}{\iff} & \forall x \in X \ : \ M_1(x) = M_2(x) + M_3(x) \\
M_1 = M_2 \cap M_3 & \stackrel{\text{déf}}{\iff} & \forall x \in X \ : \ M_1(x) = min\{M_2(x), M_3(x)\} \\
M_1 = M_2 \setminus M_3 & \stackrel{\text{déf}}{\iff} & M_3 \subseteq M_2 \wedge \forall x \in X \ : \ M_1(x) = M_2(x) - M_3(x) \\
M_1 = M_2 - M_3 & \stackrel{\text{déf}}{\iff} & M_1 = M_2 \setminus (M_2 \cap M_3) \\
|M_1| & \stackrel{\text{déf}}{=} & \displaystyle\sum_{x \in X} M_1(x)
\end{array}
$$

Exemple : Soit $X = \{x_1, x_2, x_3\}$ et soient $M_1 = \{x_1, x_1, x_1, x_3\}$ et $M_2 = \{x_1, x_2, x_2, x_3\}$ deux multi-ensembles sur X, alors nous aurons :

- $M_1 + M_2 = \{x_1, x_1, x_1, x_2, x_2, x_3, x_3\}$
- $M_1 \cap M_2 = \{x_1, x_3\}$

- $M_1 - M_2 = \{x_1, x_1\}$
- $M_2(x_2) = 2$
- $M_1 \not\subseteq M_2$ et $M_2 \not\subseteq M_1$
- $M_1 \setminus M_2$ n'est pas défini.

1.2.1 Les multi-ensembles partiellement ordonnés

Un *multi-ensemble partiellement ordonné* (ou *pomset*) est une classe d'isomorphisme d'ensembles partiellement ordonnés avec étiquetage.

Nous dirons qu'un pomset $\rho = (X, <, l)$ est *moins séquentiel* que $\rho' = (X', <', l')$, noté $\rho \preceq \rho'$, s'il existe une bijection $h : X \to X'$ préservant l'étiquetage et telle que $x < y$ implique $h(x) < h(y)$. Nous notons $\rho \equiv \rho'$ l'isomorphisme entre pomsets.

1.3 La structure des réseaux de Petri étiquetés

Nous allons tout d'abord introduire la structure des réseaux de Petri et les propriétés essentielles, puis nous nous intéresserons à l'étiquetage. Nous n'utiliserons que des réseaux étiquetés, par conséquent nous omettrons le terme *étiqueté* partout où cela ne prêtera pas à confusion.

Il existe dans la littérature un certain nombre de «variantes» (étiquetage sur les places, places avec capacité, etc). Nous donnons en fin de chapitre quelques idées sur la manière dont elles s'accorderont avec la méthode de réduction présentée dans les chapitres suivants.

1.3.1 Les réseaux de Petri

Un réseau de Petri est un graphe bi-partite. Les deux classes de sommet sont les *places* (qui représentent souvent les ressources) et les *transitions* (qui figurent les actions du système). Formellement :

Définition 1.3.1. *Réseau de Petri, sous-réseau*
Nous appellerons réseau de Petri *un triplet* $N = (P, T, W)$ *tel que :*

P *est l'ensemble des* places *du réseau ;*

T *est l'ensemble des* transitions *du réseau, t.q.* $P \cap T = \emptyset$ *;*

$W : (P \times T) \cup (T \times P) \to \mathbb{N}$ *est une application appelée la* relation de flot, *qui associe à un arc du graphe son* poids.

Nous dirons qu'un réseau est fini *si* $P \cup T$ *l'est.*
Lorsque W *est définie dans* $\{0, 1\}$, *nous dirons que le réseau est à* arcs simples.

Soit un réseau $N = (P, T, W)$ *et deux ensembles* $P' \subseteq P$ *et* $T' \subseteq T$*, alors le* sous-réseau de N induit par P', T' *est* $N' = (P', T', W')$ *tel que* W' *est* W *restreint à* $(P' \times T') \cup (T' \times P')$.

Remarque : quand l'idée de PETRI atteint les USA, vers 1970, il est rapidement établit que les réseaux de Petri et les systèmes d'addition de vecteurs[2] (SAV) sont mathématiquement équivalents, bien qu'ils soient issus d'approches très différentes (vision informatique du monde physique pour PETRI, modèle formel du parallélisme pour KARP & MILLER).

◇

Dans la suite, nous ne considérerons que des réseaux finis.

Graphiquement, un réseau de Petri est un graphe dont les sommets sont des cercles s'ils sont des places et des rectangles s'ils sont des transitions. Les poids sont adscrits aux arcs qui sont représentés par des flèches. Classiquement, les arcs de poids nul ne sont pas représentées. Comme pour les graphes, si $W(x, y) > 0$, x est appelée la *source* et y la *cible* de l'arc.

Pour alléger le graphe, le poids ne sera adscrit que s'il est supérieur à un.

Remarque : Nous aurons parfois besoin de nous abstraire du réseau pour ne plus considérer que le graphe sous-jacent $\mathcal{G}_N = (S, A)$ tel que :

$$\begin{array}{rcl} S & \stackrel{déf}{=} & P \cup T \\ (x, y) \in A & \stackrel{déf}{\Longleftrightarrow} & W(x, y) > 0 \end{array}$$

◇

Exemple 1.1 : Un réseau de Petri simple.

Prenons l'exemple d'une pile FIFO à une seule ligne et pouvant véhiculer deux messages[3]. Il est facile de voir le confort apporté par la représentation graphique du réseau ! Le réseau $N_s = (P_s, T_s, W_s)$ de la figure 1.1 est défini par :

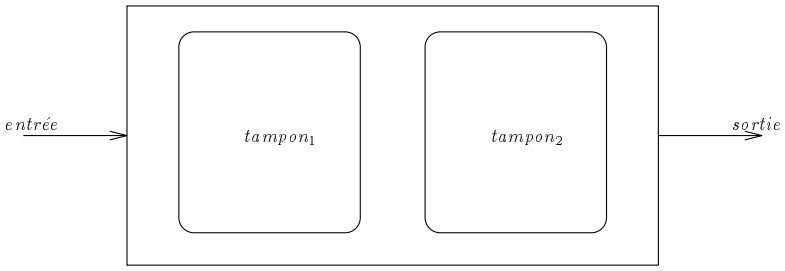
$P_s = \{entrée,\ tampon_1,\ tampon_1\ libre,\ tampon_2,\ tampon_2\ libre,\ sortie\}$;

$T_s = \{t_1,\ t_2,\ t_3\}$;

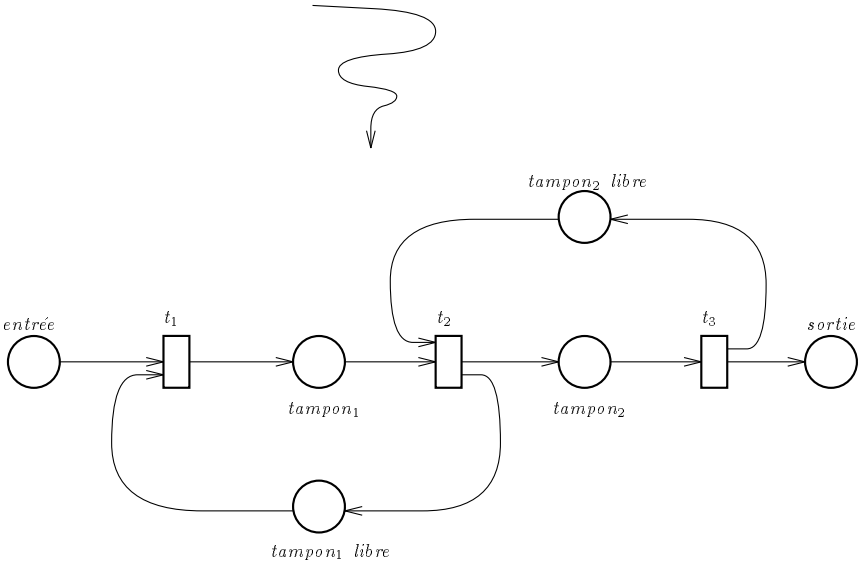
$W_s = \{\ entrée \overset{1}{\frown} t_1,\ tampon_1\ libre \overset{1}{\frown} t_1,\ t_1 \overset{1}{\frown} tampon_1,$
$tampon_1 \overset{1}{\frown} t_2,\ tampon_2\ libre \overset{1}{\frown} t_2,\ t_2 \overset{1}{\frown} tampon_2,\ t_2 \overset{1}{\frown} tampon_1\ libre,$
$tampon_2 \overset{1}{\frown} t_3,\ t_3 \overset{1}{\frown} sortie,\ t_3 \overset{1}{\frown} tampon_2\ libre\}$

[2] Nous ne présentons pas ici les SAV. Le lecteur intéressé pourra se reporter à [Reu89].

[3] Nous ne nous posons pas ici le problème de la spécification d'un système par des réseaux de Petri, le lecteur intéressé pourra se référer au livre de W. REISIG [Rei92].



File FIFO pouvant contenir deux messages du même type.



N

Figure 1.1 : Un réseau de Petri simple : une file FIFO à deux tampons.

Remarque : pour W_s, nous adoptons la notation $source \overset{poids}{\frown} cible$, étant entendu que l'arc apparaît ssi son poids est non nul.

$\diamond$

Dans l'exemple précédent, il est facile de voir que les transitions sont très similaires. Vues de l'extérieur, on peut considérer que, bien qu'étant nommées différemment, elles ont la même fonction. C'est la fonction de l'étiquetage que de définir la partie *visible* (pour un observateur extérieur au système) d'une transition.

1.3.2 L'étiquetage des transitions

L'étiquetage permet de s'abstraire de l'identité des transitions pour ne plus s'intéresser qu'à leurs effets observables. Lorsque le réseau n'est pas étiqueté, il nous faut nous interroger sur la partie *visible* (de l'extérieur) des transitions.

Définition 1.3.2. *Réseau étiqueté*

Nous appellerons réseau de Petri étiqueté *un quadruplet* $N = (P, T, W, l)$ *tel que :*

- (P, T, W) *soit un réseau de Petri au sens de la définition 1.3.1 ;*
- $l : T \to \mathcal{A}$ *est une* application d'étiquetage *(ou plus simplement un* étiquetage*) sur un alphabet (ou ensemble d'*actions $\mathcal{A}$.

Pour $t \in T$, $l(t)$ *est* l'étiquette *de* t.

Graphiquement, les étiquettes sont inscrites dans les rectangles représentant les transitions.

La définition d'un sous-réseau s'étend trivialement en $N' = (P', T', W', l')$ avec $N' = (P', T', W')$ tel que dans la définition 1.3.1 et $l' = l/_{T'}$.

Nous reprenons le réseau N_s de l'exemple 1.1 en ajoutant un étiquetage. Nous considérerons assez globalement que toutes les transitions ont le même effet : elles font progresser un message sur la ligne.

Exemple 1.2 : .

Nous définissons le réseau étiqueté $N_e = (P_e, T_e, W_e, l_e)$ par :

$P_e = P_s$;

$T_e = T_s$;

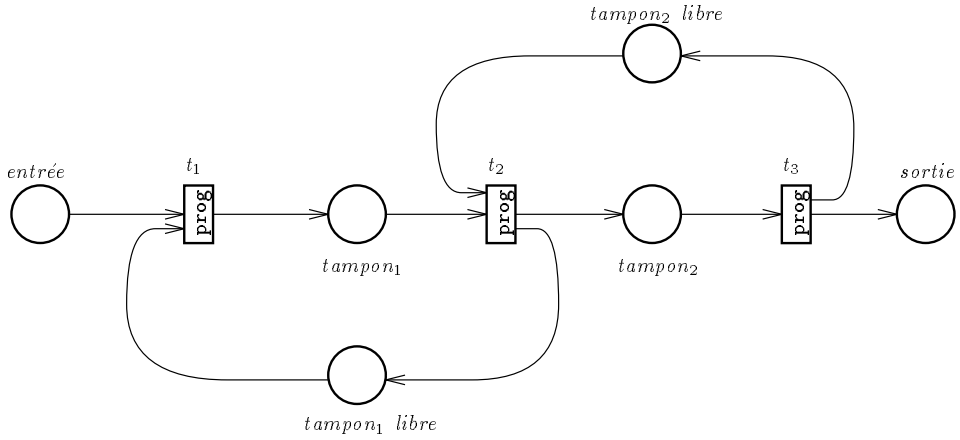
$W_e = W_s$;

l_e défini sur $\mathcal{A} = \{\texttt{prog}\}$ et tel que $l(t_1) = l(t_2) = l(t_3) = \texttt{prog}$.

La figure 1.2 donne la représentation graphique de N.

Nous allons maintenant nous intéresser au comportement des réseaux de Petri (étiquetés).

Figure 1.2 : Un réseau de Petri étiqueté (la file FIFO).



N

1.4 Le comportement dans les réseaux étiquetés

Nous avons choisi comme cadre mathématiques les multi-ensembles (voir la section 1.2). À la structure des réseaux, nous allons ajouter la notion de *marquage* qui nous permettra de modéliser leur comportement.

1.4.1 Les réseaux de Petri marqués

Un *marquage* correspond à un multi-ensemble de places. Il indique les ressources effectivement présentes dans le système et est symbolisé par une distribution de *jetons* dans les places. Formellement :

Définition 1.4.1. *Réseau marqué, système*
Nous appellerons réseau de Petri marqué *(ou* système*) une paire* $\Sigma = (N, M_0)$ *telle que :*

$N = (P, T, W, l)$ *est un réseau étiqueté au sens de la définition 1.3.2 ;*

M_0 *est un un multi-ensemble sur* P *(i.e.* $M \in \mathcal{M}(P)$*).*

M_0 *est appelé le* marquage initial *du système.*
Nous dirons qu'un marquage est sauf *ssi* $\forall p \in P,\ M(p) \leq 1$.

Nous verrons ci-après (parag. 1.4.2) que *l'évolution* du système peut être modélisée par une suite de marquages et de transitions.

Nous noterons souvent $\mathcal{M}_N \stackrel{déf}{=} \mathcal{M}(P)$ l'ensemble des marquages d'un réseaux.

Exemple 1.3 : Une file FIFO à deux tampons.

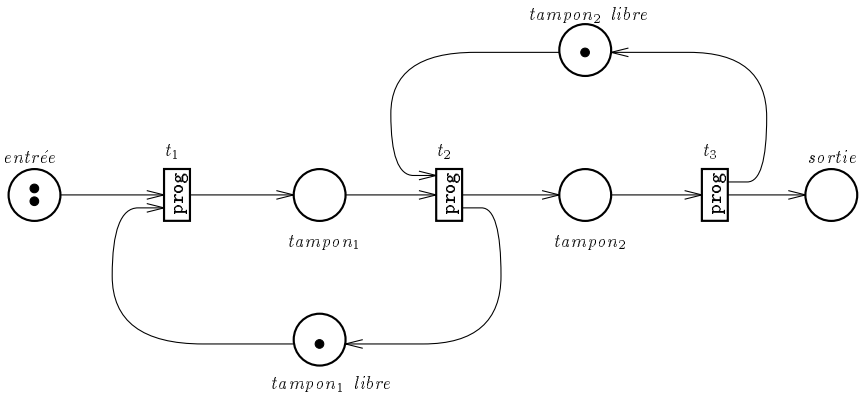
Nous donnons une situation initiale telle que la file soit vide et reçoive deux messages. Nous avons le réseau marqué (N_m, M_m) défini par :

$N_m = N_e$;

$M_m = \{entrée, entrée, tampon_1\ libre, tampon_2\ libre\}$.

La figure 1.3 donne la représentation graphique de (N_m, M_m).

Figure 1.3 : Un réseau de Petri marqué : deux messages en entrée.



(N_m, M_m)

Remarques :

1. Il est évident que l'étiquetage et le marquage initial sont deux compléments distincts et que nous pourrions définir des réseaux marqués mais non étiquetés.
2. Il existe dans la littérature différentes manières de noter les marquages qui sont, de notre point de vue, moins pratiques :
   - une notation vectorielle est fréquemment utilisée, conjointement à la matrice d'incidence (déf. 7.1.4, p. 157). Si cette notation facilite certains calculs (comme celui des invariants), elle fait appel à un ordre sur P qui n'est pas explicité. Le marquage initial de l'exemple 1.3 serait $M = (2, 0, 1, 0, 1, 0)$.
   - GENRICH, dans [Gen90] utilise une notation assez proche de la notre. Pour ce même exemple, nous aurions : $M = 2\langle entrée\rangle + 1\langle tampon_1\ libre\rangle + 1\langle tampon_2\ libre\rangle$
3. L'exemple précédent permet déjà plusieurs réflexions :

- nous aurions pu prendre une infinité de marquages différents ;
- une méthode de réduction prenant en compte le marquage initial devrait donc être relancée à chaque changement de ce marquage (ce qui peut être très lourd si on désire faire de la vérification).

◇

Nous allons maintenant fixer les règles *d'évolution* d'un système.

1.4.2 Comportement du système

Les changements d'états du système sont produits par les tirs des transitions. Pour qu'une transition soit tirable, il faut que les ressources nécessaires soient présentes. Après le tir, ces ressources sont consommées et éventuellement de nouvelles ressources sont produites. Formellement :

Définition 1.4.2. *Pré- et post-conditions*
Étant donné un réseau $N = (P, T, W)$ *et* $x \in P \cup T$, *nous avons les ensembles :*

$$
\begin{aligned}
{}^\circ x &\stackrel{\text{déf}}{=} \{y \in P \cup T \ : \ W(y,x) > 0\} \\
x^\circ &\stackrel{\text{déf}}{=} \{y \in P \cup T \ : \ W(x,y) > 0\}
\end{aligned}
$$

et les multi-ensembles ${}^\bullet x$ *et* $x^\bullet$ *t.q. :*

$$
\begin{aligned}
&\forall p \in P,\ {}^\bullet t(p) = W(p,t) \\
&\forall p \in P,\ t^\bullet(p) = W(t,p)
\end{aligned}
$$

Nous appellerons ${}^\circ x$ *(resp.* x°*) l'ensemble des* prédécesseurs *(resp.* successeurs*) de* x.

Pour $t \in T$, *nous appellerons* ${}^\bullet t$ *(resp.* $t^\bullet$*) les* pré-conditions *(resp. les* post-conditions*) de* t.

Une transition triviale *est telle que* ${}^\bullet t = t^\bullet$.

Remarques :

1. Ces notions sont, dans leur définition, indépendantes des marquages. Cependant, les pré- et post-conditions n'ont pas de sens intuitif hors du comportement des systèmes, c'est pourquoi nous les introduisons ici.
2. Il arrive de trouver dans la littérature des notations différentes : la notation ${}^\bullet t, t^\bullet$ est parfois utilisée pour les *ensembles* de places. Nous préférons nos notations :
   (a) parce qu'elles constituent une extension cohérente à celle des «systèmes élémentaires» (voir [RT86]) où W est une relation incluse dans $(P \times T) \cup (T \times P)$;

(b) parce qu'elles sont cohérentes avec la notation des ensembles de places initiales ou finales d'un réseau causal (voir la section 1.6, p. 44) ;

(c) enfin, parce qu'elles nous semblent plus adaptées au graphisme des réseaux de Petri où les nœuds sont des cercles ou rectangles et les jetons des points noirs.

◇

Nous pouvons maintenant nous intéresser au *tir* proprement dit d'une transition :

Définition 1.4.3. *Règle d'occurrence d'une transition*

Étant donné un réseau $N = (P, T, W, l)$, nous dirons qu'une transition $t \in T$ est tirable dans un marquage $M \in \mathcal{M}_N$ *ssi* ${}^\bullet t \subseteq M$.

Nous noterons cela $M \xrightarrow{t}$ (et $M \overset{t}{\not\rightarrow}$ si t n'est pas tirable depuis M).

Le tir *de la transition t à partir du marquage M mène à un marquage M' défini par :*

$$M' \overset{\text{déf}}{=} M - {}^\bullet t + t^\bullet$$

Nous noterons cela $M \xrightarrow{t} M'$.
Si $l(t) = a$, nous avons les notations $M \xrightarrow{t:a}$ et $M \xrightarrow{a} \overset{\text{déf}}{\Longleftrightarrow} \exists t \ : \ M \xrightarrow{t:a}$.

Par ailleurs, nous dirons qu'un marquage M est *mort*, noté $M \not\rightarrow$, s'il ne permet le tir d'aucune transition.

Sur la figure précédente (1.3), nous avons ainsi :

$$ {}^\bullet t_1 = \{\textit{entrée}, \textit{tampon}_1\ \textit{libre}\} \tag{1.1}$$

$$ t_3{}^\bullet = \{\textit{tampon}_2\ \textit{libre}, \textit{sortie}\} \tag{1.2}$$

$$ M_m \xrightarrow{t_1} \{\textit{entrée}, \textit{tampon}_1, \textit{tampon}_2\ \textit{libre}\} \tag{1.3}$$

$$ M_m \overset{t_2}{\not\rightarrow} \tag{1.4}$$

Ce que nous pouvons traduire par :

- «pour accepter un message, il faut qu'il y en ait un en entrée et que le premier tampon soit libre» pour (1.1) ;
- «lorsqu'on a délivré un message, le deuxième tampon devient libre» pour (1.2)[4] ;
- «l'état initial du système permet d'accepter un message *et* de passer dans un état où le premier tampon n'est plus libre»[5] pour (1.3) ;

[4] Nous ne nous préoccupons pas ici d'en vérifier la correction.
[5] Nous passons sous silence les «non-changements» par simplicité.

- enfin, «dans l'état initial, nous ne pouvons pas faire passer un message du premier tampon dans le second» pour (1.4).

Remarque : de cette règle d'occurrence, il découle différentes relations entre les transitions :

- de causalité : si une transition n'est franchissable qu'à condition qu'une autre soit franchie préalablement ;
- de conflit : si à l'inverse le franchissement d'une transition empêche celui d'une autre ;
- de concurrence : si deux transitions peuvent être franchies simultanément.

◇

1.4.3 Séquences de transitions et atteignabilité

Nous pouvons élargir ces définitions au cas des séquences de transitions (à la base des sémantiques *d'entrelacement*). Formellement, étant donné un réseau $N = (P, T, W)$, nous appellerons *séquence (de transitions)* une séquence sur T (au sens de la définition 1.1.3).

Nous avons alors :

$$\begin{aligned} {}^\bullet\varepsilon &\stackrel{déf}{=} \emptyset \\ {}^\bullet(\rho u) &\stackrel{déf}{=} {}^\bullet\rho + ({}^\bullet u \setminus \rho^\bullet) \end{aligned}$$

et symétriquement :

$$\begin{aligned} \varepsilon^\bullet &\stackrel{déf}{=} \emptyset \\ (\rho u)^\bullet &\stackrel{déf}{=} (\rho^\bullet \setminus {}^\bullet u) + u^\bullet \end{aligned}$$

Nous avons les définitions suivantes :

Définition 1.4.4. *Séquence d'occurrences, atteignabilité Soit M un marquage de $N = (P, T, W, l)$ et $\rho = u_1 \dots u_k \in T^*$:*

$$M \stackrel{\rho}{\longrightarrow\!\!\!\!\!\rightarrow} M' \stackrel{\text{déf}}{\Longleftrightarrow} M \stackrel{u_1}{\rightarrow} M^1 \stackrel{u_2}{\rightarrow} M^2 \cdots M^{k-1} \stackrel{u_k}{\rightarrow} M'$$

Nous dirons alors que la séquence d'occurrences ρ *est* tirable depuis M *et mène au marquage M'.*

En particulier, nous avons $M \stackrel{\epsilon}{\longrightarrow\!\!\!\!\!\rightarrow} M$ pour tout M.

M' est atteignable depuis M, *noté $M \longrightarrow\!\!\!\!\!\rightarrow M'$, ssi il existe une séquence $\rho \in T^*$ telle que $M \stackrel{\rho}{\longrightarrow\!\!\!\!\!\rightarrow} M'$.*

Nous étendons naturellement l'étiquetage aux séquences par :

$$\begin{aligned} l(\varepsilon) &\stackrel{\text{déf}}{=} \varepsilon \\ l(u_1 \dots u_k) &\stackrel{\text{déf}}{=} l(u_1) \dots l(u_k) \in \mathcal{A}^* \end{aligned}$$

Par extension, nous dirons alors qu'une transition t est *atteignable* depuis un marquage M ssi il existe $M' \in \mathcal{M}_N$ tel que ${}^\bullet t \subseteq M'$ et M' atteignable depuis M.

Similairement, nous dirons qu'une place p est *atteignable* depuis un marquage M ssi il existe $M' \in \mathcal{M}_N$ tel que $p \in M'$ et M' atteignable depuis M.

Remarque : un réseau peut être vu comme un automate à compteurs avec un test partiel pour zéro. Une séquence tirable correspond alors à un mot accepté par l'automate. Avec cette interprétation, un réseau non étiqueté est déterministe puisque pour un état donné et une transition donnée au plus un changement est possible. Dans un réseau étiqueté, il peut arriver que plusieurs transitions de même étiquette soient franchissables, ce qui correspond aux automates non-déterministes.

◇

Nous avons le théorème suivant qui établit la cohérence des notations (la preuve est aisée) :

Théorème 1.4.5. *Soit M un marquage de $N = (P, T, W)$ et $\rho \in T^*$:*

$$M \xrightarrow{\rho} M' \stackrel{\text{déf}}{\Longleftrightarrow} {}^\bullet\rho \subseteq M \wedge M' = M - {}^\bullet\rho + \rho^\bullet$$

Une conséquence simple est que pour tout marquage L, $M \xrightarrow{\rho}\!\!\!\!\rightarrow M'$ entraîne $M + L \xrightarrow{\rho}\!\!\!\!\rightarrow M' + L$.

1.4.4 Ensemble et graphe des marquages atteignables

Il est facile de voir que l'atteignabilité est une notion centrale lors de l'étude dynamique des systèmes modélisés par les réseaux. C'est aussi un lien fort utile dans le cadre des sémantiques d'entrelacement, entre les réseaux de Petri et les systèmes de transitions. Formellement :

Définition 1.4.6. *Ensemble des marquages atteignables*

Étant donné un système $\Sigma = (N, M_0)$ où $N = (P, T, W)$, nous appellerons ensemble de marquages atteignables *l'ensemble :*

$$\mathcal{R}(\Sigma) \stackrel{\text{déf}}{=} \{M \in \mathcal{M}(P) \ : \ M_0 \longrightarrow\!\!\!\!\rightarrow M\}$$

Dans la majorité des cas, cet ensemble sera énorme, voire infini. Le graphe des marquages atteignables s'obtient simplement en prenant pour nœuds $\mathcal{R}(\Sigma)$ et en ajoutant les arcs étiquetés par les transitions permettant de passer d'un marquage à un autre :

Définition 1.4.7. *Graphe des marquages atteignables*
Étant donné un système $\Sigma = (N, M_0)$ *où* $N = (P, T, W)$, *nous appellerons* graphe des marquages (atteignables) *le graphe*[6] *orienté et étiqueté* $\mathcal{GR}(N, M_0) = (S, A, E)$ *tel que :*

- $S = \mathcal{R}(\Sigma)$ *est l'ensemble des sommets du graphe ;*
- A *est l'ensemble des arcs et* $E : A \to T$ *un étiquetage. Ils sont définis par :*
$$(M \xrightarrow{a} M' \wedge E(a) = t) \overset{\text{déf}}{\Longleftrightarrow} (M \xrightarrow{t} M') \text{ dans } N.$$
- M_0 *est un nœud distingué appelé la* racine *du graphe.*

Le lien entre les comportements séquentiels du réseau et les chemins orientés de $\mathcal{GR}(N, M_0)$ est donné par le théorème suivant (dont la preuve est aisée) :

Théorème 1.4.8. *Dans un système* (N, M_0), *un marquage* M' *est atteignable depuis un marquage* M *par une séquence* ρ *ssi il existe un chemin orienté de* M *vers* M' *dans* $\mathcal{GR}(N, M_0)$, *étiqueté par* ρ.

Remarques sur $\mathcal{G}R(N, M_0)$ **:**

1. De tels graphes (orientés, étiquetés et ayant une racine) sont des *systèmes de transitions*. Ils sont souvent utilisés pour modéliser le comportement des systèmes.
2. Lorsque le réseau est étiqueté, le graphe des marquages atteignables peut aussi porter ce double étiquetage.

◇

1.4.5 Tirs concurrents

Un des avantages des réseaux de Petri est leur adaptation au vrai parallélisme. Nous allons étendre les notions vues précédemment au cas des pas concurrents. Formellement, étant donné un réseau $N = (P, T, W)$, nous appellerons *pas (concurrent)* un multi-ensemble de transition $\mu = \{u_1, \ldots, u_n\} \in \mathcal{M}(T)$.

[6] Nous avons en fait un *multi*-graphe, i.e. il peut exister plusieurs arcs entre deux sommets. En effet, rien n'interdit que deux transitions aient les mêmes pré- et post-conditions.

Un pas représente donc un ensemble d'actions simultanées[7].
Les notations précédentes deviennent alors, pour un marquage M de N :

$$\begin{aligned} {}^\bullet\mu &\stackrel{déf}{=} \sum_{t\in T} \mu(t).{}^\bullet t \\ \mu^\bullet &\stackrel{déf}{=} \sum_{t\in T} \mu(t).u_i{}^\bullet \\ M \stackrel{\mu}{\Rightarrow} M' &\stackrel{déf}{\Longleftrightarrow} {}^\bullet\mu \subseteq M \wedge M' = M - {}^\bullet\mu + \mu^\bullet \end{aligned}$$

Nous noterons le pas vide ε.
Nous étendons la notion de *support* à un pas :

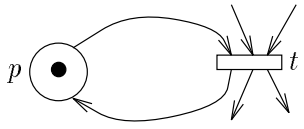
$$\begin{aligned} \mathcal{S}(\varepsilon) &\stackrel{déf}{=} \emptyset \\ \mathcal{S}(\mu \cup \{u\}) &\stackrel{déf}{=} \mathcal{S}(\mu) \cup \{u\} \end{aligned}$$

La figure 1.4 nous montre l'évolution de N_m après les tirs de la séquence $_{t_1}.t_2$ puis du pas $\{_{t_1},t_3\}$.

Nous aurons deux termes particuliers pour désigner certaines formes de parallélisme :

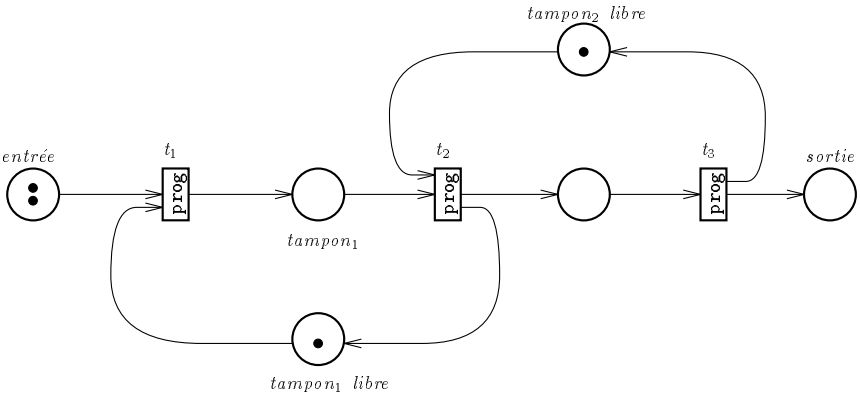
- nous parlerons d'*égo-concurrence* lorsqu'une *transition* est parallèle à elle-même, e.g. $\{t_1, t_1\},\ t_1 \in T$;
- nous parlerons d'*auto-concurrence* lorsqu'une *action* est parallèle à elle-même, e.g. $\{a, a\},\ a \in \mathcal{A}$;

Une différence majeure est que le premier ne peut exprimer qu'un parallélisme dynamique, alors que le second peut aussi exprimer un parallélisme structurel. Nous verrons ultérieurement (voir la section 8.3 sur le raffinage) que le parallélisme dynamique est parfois gênant. Une des façons de l'éviter consiste alors à ajouter une place à chaque transition, comme montré sur la figure ci-dessous. Cet ajout n'altère pas les propriétés essentielles du système ([BD90]).

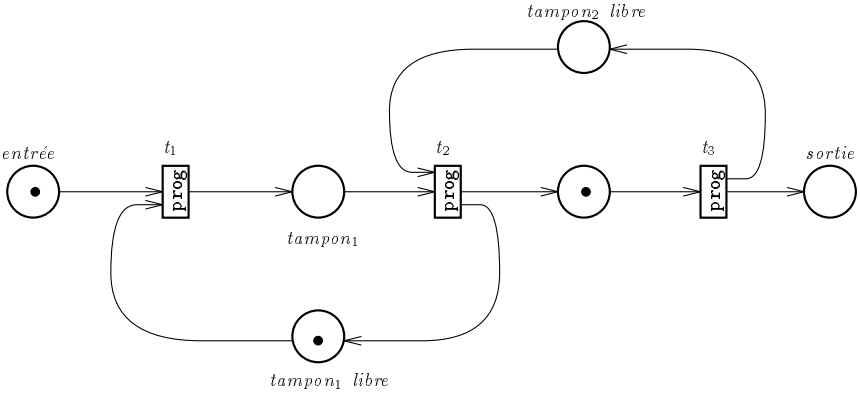


[7] Cette notion n'est pas toujours identique, selon que l'on considère les transitions comme *atomiques* (i.e. elles se font toutes en temps 1) ou non (voir p.ex. les réseaux temporisés [Zub80]).

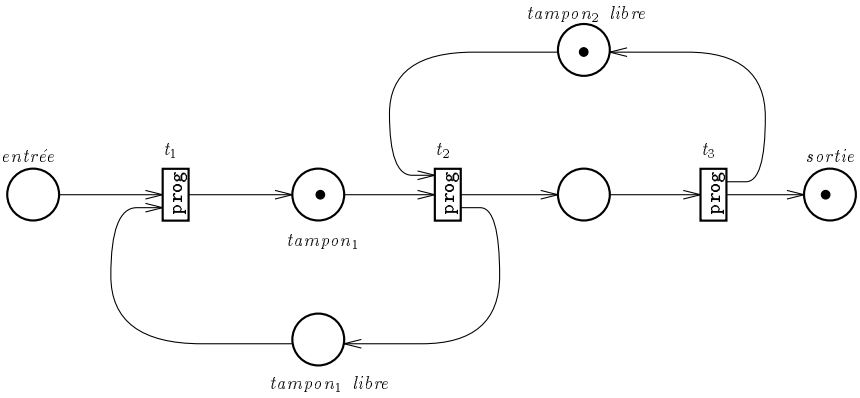
Figure 1.4 : Évolution de N pour $M_{init} \xrightarrow{t_1.t_2.\{t_1,t_3\}} \twoheadrightarrow$.



(N, M_{init})



$(N, M_{init}) \xrightarrow{t_1.t_2} \twoheadrightarrow (N, M')$



$(N, M') \xrightarrow{\equiv \{t_1,t_3\} \rightarrow} (N, M'')$

1.4.6 Des pas concurrents aux séquences

Il existe un lien évident entre les pas et les séquences. Une séquence ayant les mêmes éléments qu'un pas peut être vue comme une réalisation de ce pas. Un unique observateur voit les actions dans un certain ordre et par suite transcrit un pas en une séquence[8]. Ceci légitimise notre notation du pas vide par ε et l'extension de nos définitions.

Nous définissons :

Définition 1.4.9. *Linéarisation d'un pas*

Soit $\mu \in \mathcal{M}_N$ un pas parallèle, nous dirons qu'une séquence $\rho \in T^*$ est une linéarisation de μ *ssi :*

$$\forall t \in T,\ occ_\rho(t) = occ_\mu(t)$$

Nous notons $seq(\mu)$ l'ensemble des linéarisations de μ.

Nous avons la simple proposition suivante :

Proposition 1.4.10. *Soit $\mu \in \mathcal{M}_N$ un pas parallèle, alors :*

$$\forall \rho \in seq(\mu)\ :\ ({}^\bullet\rho \subseteq {}^\bullet\mu) \wedge (\rho^\bullet \subseteq \mu^\bullet)$$

1.5 Quelques remarques sur les sémantiques

Nous pouvons considérer l'évolution des réseaux marqués de nombreuses manières (voir par exemple [Hoo94]). Une des grandes subdivisions est le choix de la sémantique :

1. les sémantiques d'entrelacement réduisent le parallélisme à du non-déterminisme séquentiel, (i.e. $a||b = a.b + b.a$)[9]. Elle identifie, par exemple, les deux systèmes de la figure 1.5. Pourtant, ils sont très différents. D'abord, il est fort probable que Σ soit plus rapide. D'autre part, il est bien connu que son raffinement peu engendrer certains problèmes (voir p.ex. [CMP87, Vog92]).

2. les sémantiques de pas permettent de les distinguer. Intuitivement, elles suppose l'existence d'une horloge globale et l'atomicité des transitions (i.e. chaque action prend exactement une unité de temps. Deux réseaux sont donc équivalents s'ils peuvent effectuer les mêmes séquences de pas parallèles. Les deux systèmes de la figure 1.6 sont équivalents pour ces sémantiques. Pourtant, pour faire $(a \parallel c); b$, le second doit attendre que c soit fini pour exécuter b.

[8] C'est une des raisons invoquées par certains auteurs (e.g. [Mur91]) pour dire qu'il suffit d'un nombre suffisant d'observateurs pour induire le parallélisme et que par suite les sémantiques d'entrelacement sont suffisantes.

[9] Nous utilisons les notations de CCS. $a||b = a.b + b.a$ signifie que «a en parallèle avec b» est sémantiquement équivalent à «a suivi de b ou b suivi de a».

Figure 1.5 : Σ_1 et Σ_2 sont indistinguables (sém. d'entrelacement).

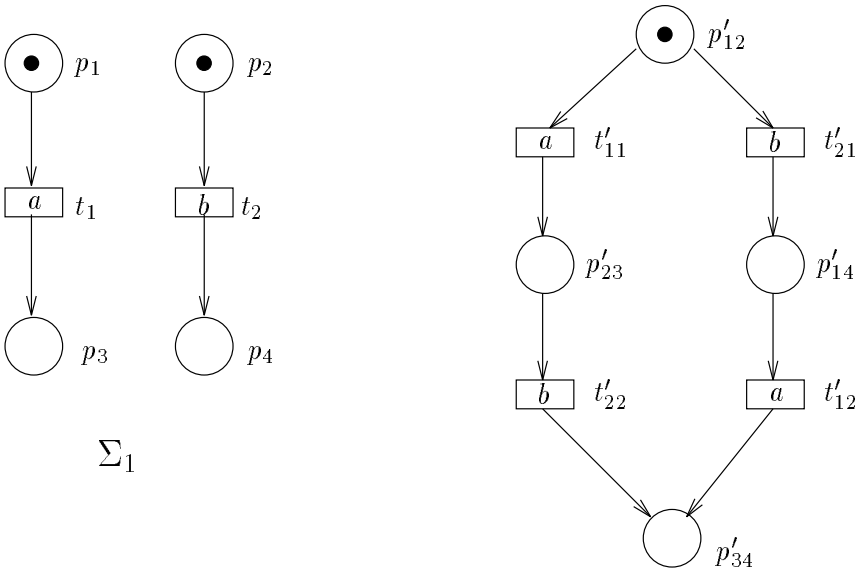
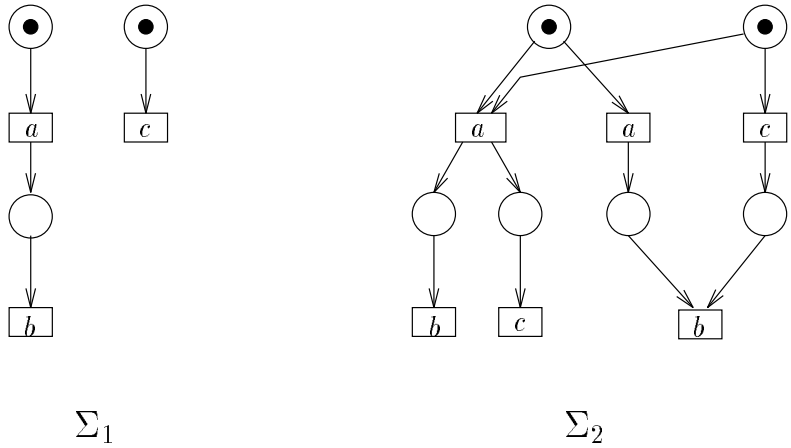


Figure 1.6 : Σ_1 et Σ_2 sont indistinguables (sém. pas).



3. une sémantique du vrai parallélisme se doit de les distinguer. Les réseaux de Petri sont parfaitement adaptés à ce type de sémantique. Pour cela, il est nécessaire de disposer d'une structure telle que deux réseaux ayant les mêmes comportements parallèles mais une structure différente puissent être identifiés ([Pet77, Rei85]). La section 1.6 présente une classe de réseaux, les réseaux causaux, qui nous permet de représenter les comportements séquentiels et concurrents des réseaux de Petri.

1.6 Les réseaux causaux

Ceux-ci sont introduits dans [Old89] afin de montrer qu'une équivalence donnée respecte le comportement concurrent des réseaux. Un réseau causal permet d'exprimer le parallélisme d'une séquence de transitions à travers un réseau dans lequel tous les choix ont été résolus. Pour cela, tout branchement sur les places est interdit : un jeton dans une place ne peut avoir plusieurs origines différentes, de même qu'il ne peut choisir son futur. Nous verrons à travers la définition 1.6.3 comment un réseau causal reflète les comportements d'un réseau N donné.

Définition 1.6.1. *Réseau causal*

Un réseau causal *(fini) est un réseau à arcs simples étiqueté fini* $C \stackrel{\text{déf}}{=} (P_C, T_C, F_C, l_C)$ *tel que :*

1. *pour tout* $u \in T_C$, ${}^\bullet u$ *et* $u^\bullet$ *sont des ensembles (et non des multi-ensembles),*
2. *les places sont sans branchement : pour tout* $p \in P_C$, $|{}^\circ p| \leq 1$ *et* $|p^\circ| \leq 1$,
3. *la relation* $F_C \subseteq (P \times T) \cup (T \times P)$ *est bien-fondée, c'est-à-dire qu'il n'existe pas de chaîne en arrière infinie :*

$$\cdots (p_n, u_n)(u_n, p_{n-1}) \cdots (p_1, u_1)(u_1, p_0)$$

dans F_C.

Le point 3 nous permet d'affirmer que ${F_C}^+$, la fermeture transitive de F_C, est acyclique et constitue un ordre bien-fondé sur $P_C \cup T_C$.

Nous notons ${}^\circ C \stackrel{déf}{=} \{p \in P_C \ : \ {}^\circ p = \emptyset\}$ l'ensemble des *places initiales* de C, c'est-à-dire les places n'étant alimentées par aucune transition. Symétriquement, $C^\circ \stackrel{déf}{=} \{p \in P_C \ : \ p^\circ = \emptyset\}$ désigne l'ensemble des *places finales* de C qui n'alimentent aucune transition.

La propriété suivante indique qu'un réseau causal ne contient pas de places ou de transitions inutiles (par exemple non atteignables) :

Proposition 1.6.2. *Si C est un réseau causal, alors il existe une séquence de transitions $^\circ C = M_0 \stackrel{u_1}{\rightarrow} M_1 \cdots \stackrel{u_k}{\rightarrow} M_k$ telle que*

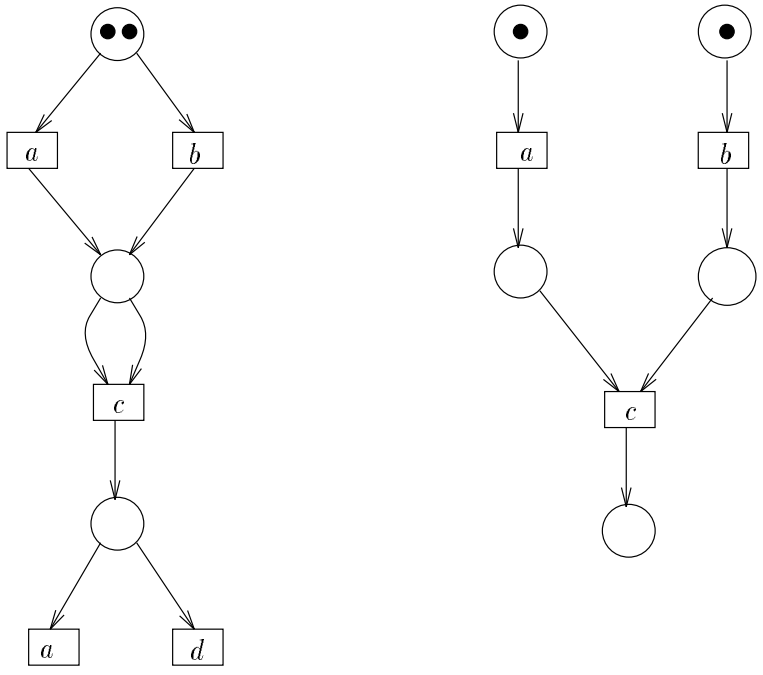
$$P_C = \bigcup_{i=0,\ldots,k} M_i \text{ et } T_C = \{u_1, \ldots, u_k\}$$

La preuve est relativement technique et ne présente pas d'intérêt particulier. Le lecteur pourra se reporter à [Aut93], page 94.

Nous appelons de telles séquences de transitions des *exécutions complètes* du réseau causal C. L'existence de ces exécutions complètes nous permet d'affirmer qu'un réseau causal ne contient pas de places ni de transitions inutiles. Le dernier marquage atteint par une telle exécution est nécessairement C°. De plus, la définition 1.6.1 et $M_0 = {}^\circ C$ nous permettent d'affirmer que tous les marquages M_i sont saufs.

L'exemple de la figure 1.7 montre le réseau causal correspondant au comportement parallèle de N pour $(a \parallel b)c$.

Figure 1.7 : Un réseau causal.



N C

1.6.1 Réseaux causaux et comportements parallèles

Nous pouvons maintenant définir le plongement d'un réseau causal dans un réseau N :

Définition 1.6.3. *Plongement*
Étant donné un réseau causal C et un réseau N, une application

$$f \ : \ P_C \cup T_C \to P_N \cup T_N$$

est appelée un plongement *de C dans N, noté $f : C \to N$ si :*

1. $f(P_C) \subseteq P_N$ *et* $f(T_C) \subseteq T_N$,
2. $\forall u \in T_C,\ l_C(u) = l_N(f(u))$,
3. $\forall u \in T_C,\ f({}^\bullet u) = {}^\bullet f(u)$ *et* $f(u^\bullet) = f(u)^\bullet$.

Le point 3 indique que le plongement respecte la relation de flot, le point 2 que ce plongement respecte l'étiquetage, et le point 1 que ce plongement permet de décrire un comportement de N. Nous reprenons en figure 1.8 l'exemple illustrant un réseau causal, dans lequel nous avons représenté en pointillés un plongement f.

Si ${}^\circ C \stackrel{u_1}{\to} \ldots \stackrel{u_k}{\to} C^\circ$ est une exécution complète de C, alors

$$M = f({}^\circ C) \stackrel{f(u_1)}{\to} \ldots \stackrel{f(u_k)}{\to} M' = f(C^\circ)$$

est une séquence de transitions dans N, notée $M \stackrel{C,f}{\to} M'$. De même, pour toute séquence de transitions $\rho = M \stackrel{t_1}{\to} \ldots \stackrel{t_n}{\to} M'$ de N, il existe un réseau causal C et un plongement $f : C \to N$ tels qu'il existe une exécution complète de C dont l'image par f dans N est ρ.

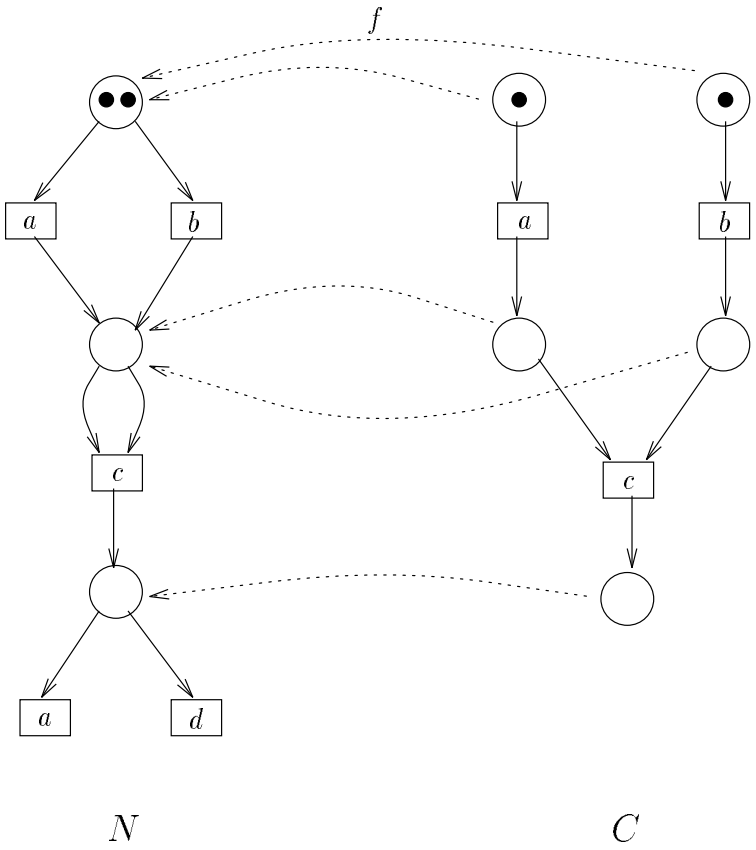
Une paire $\pi = (C, f)$ où $f : C \to N$ et le pas $M \stackrel{\pi}{\twoheadrightarrow} M'$ qui lui est associé sont appelés un *processus* de N.

Nous utiliserons les réseaux causaux dans les preuves que nous ferons ultérieurement sur les équivalences basées sur la bisimulation pomset. Nous donnons une définition des pomsets basée sur les processus en associant à un réseau causal C son pomset $pom(C)$ par

$$pom(C) = (T_C, {F_C}^+/_{T_C \times T_C}, l_C)$$

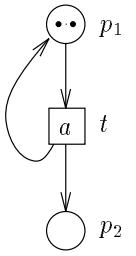
Nous conservons dans $pom(C)$ uniquement les relations de dépendances entre transitions, alors que dans C nous trouvons également une description de la façon dont sont déplacés les jetons dans le réseau. Nous notons $M \stackrel{\rho}{\to} M'$ quand il existe un *pas pomset* $M \stackrel{C,f}{\to} M'$ dans un réseau N avec $\rho = pom(C)$.

Figure 1.8 : Un plongement d'un réseau causal C dans un réseau N.



Remarque sur la définition d'un «pomset» :

Notre définition d'un pomset est différente de celle de [GV87] qui le définit comme une linéarisation d'une séquence de tirs sur laquelle est mise un ordre construit à partir de la relation de flot. Dans le cadre de réseaux non saufs, certains pomsets ne peuvent être construits de cette façon. Dans le réseau ci-contre, la séquence $\{p_1, p_1\} \stackrel{t}{\rightarrow}\stackrel{t}{\rightarrow} \{p_2, p_2\}$ peut donner lieu à deux pomsets : «a en parallèle avec a» et «a suivi de a».

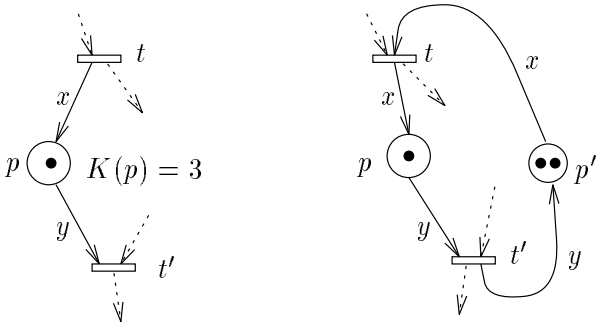


◇

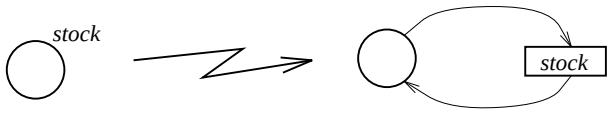
Remarques

Il existe une grande variété d'enrichissements des réseaux de Petri (capacités [Dev89], temporisation [Zub80], etc). Nous faisons ci-après quelques remarques sur des variantes classiques et codables dans les réseaux tels que nous les avons définis.

Réseaux à capacités. Ce sont définis par $N = (P, T, W, K)$, où (P, T, W) est un réseau au sens de la définition 1.3.1, p. 29 et où K est une fonction de P dans $\mathbb{N} \cup \{\omega\}$ fixant une *capacité* pour les places (i.e. le nombre maximal de jetons qu'elles peuvent contenir). Nous pouvons les coder dans les réseaux sans capacités en utilisant la méthode donnée dans [Rei85]. On associe à chaque place $p \in P$ telle que $K(p) \in \mathbb{N}$ une place *complémentaire* p'. La figure ci-dessous illustre cette construction.



Étiquetage sur les places. Cet étiquetage «dual», est utilisé par exemple dans [PS90]. Un tel étiquetage ne gène en rien notre méthode de réduction. En effet, de tels réseaux sont «équivalents» à un réseau étiqueté sur les transitions (voir la figure ci-dessous).



Bibliographie

Le problème de l'atteignabilité est traité dans [May81, Kos82]. [May84] donne un algorithme général. Enfin, [Reu89] expose ce problème de manière très claire.

Les problèmes liés à la structure des réseaux (graphes et hypergraphes) sont traités de manière assez exhaustive par [Ber87].

Chapitre 2

À la recherche d'une équivalence sur les places

Nous allons nous donner comme objectif très général de trouver une méthode de réduction qui :

1. soit une équivalence sur les places nous permettant de les fusionner ;
2. préserve la bisimulation ;
3. est indépendante du marquage initial (i.e. du contexte). En effet, $\mathcal{R}(N, M_0)$ est souvent trop grand pour être utilisable et nous voulons éviter de recommencer la réduction pour chaque marquage initial.

Une première approche de cet objectif est la suivante :

$$p \sim q \Leftrightarrow \forall M \in \mathcal{M}_N \ : \ M + p \underline{\leftrightarrow} M + q$$

Dans ce chapitre, nous allons rappeler la définition et quelques propriétés essentielles de la bisimulation classique (entre les marquages). Nous utiliserons la définition de la bisimulation *forte* [Mil83, Par81].

Nous en dériverons une première relation d'équivalence, notée $\sim$, entre les places du réseau. La section 2.2 étudie $\sim$. En particulier, nous y prouvons un résultat très général pour la correction du quotient (théorème 2.2.8). Ce théorème fondamental donne une caractérisation des relations sur les places telles que le quotient soit correct, i.e. telles que le réseau et le réseau quotienté soient bisimilaires.

La dernière section est dévolue à la calculabilité de $\sim$. Nous utilisons une technique inspirée de [Jan93], mais de manière non triviale.

2.1 La bisimulation classique (entre les marquages)

En principe, deux programmes peuvent être considérés comme équivalents seulement s'ils sont écrits exactement de la même manière. Ceci

est évidemment trop restrictif, nous avons donc à chercher une équivalence «raisonnable» entre les programmes.

Qu'entendons-nous par «raisonnable» ? La condition prépondérante est de respecter les règles de changement d'état, i.e. si d'un marquage M je peux tirer une transition t et arriver dans un marquage M', alors de tout marquage équivalent à M je dois pouvoir tirer une transition équivalente à t et parvenir à un état équivalent à M'. Il y a évidemment un grand nombre de candidates, mais nous allons nous intéresser à l'une des plus étudiée : la bisimulation (forte), introduite dans [Mil80, Par81].

Un certain nombre de propriétés font de la bisimulation une équivalence comportementale «agréable» :

- c'est une équivalence comportementale fondamentale pour les systèmes réactifs ;
- elle préserve les comportements arborescents ([Mil80]) et donc beaucoup de propriétés dynamiques ;
- c'est une congruence pour un grand nombre d'opérateurs (composition séquentielle, parallèle, etc) ;
- elle est basée sur les comportements observables du systèmes (i.e. elle fait abstraction de l'identité des transitions pour ne considérer que leur effet visible) ;
- elle a des liens forts avec les logiques temporelles. Typiquement, deux systèmes sont bisimilaires ssi ils vérifient les mêmes propriétés HML ([HM80]) ;
- enfin, nous verrons qu'il est possible d'adapter sa définition pour obtenir un traitement adéquat des actions internes ou du raffinage.

2.1.1 Définition de la bisimulation

Comme la bisimulation est issue du domaine des automates, il est nécessaire de se représenter le réseau comme son graphe des marquages. Ce faisant, il est évident que nous perdons les informations relatives au parallélisme. Cependant, nous verrons que la bisimulation admet un grand nombre de variantes moins «oublieuses» du parallélisme.

Intuitivement, deux systèmes sont bisimilaires s'ils peuvent s'imiter l'un l'autre, i.e. avoir les mêmes comportements. Une manière classique de voir la bisimulation sont les «jeux de bisimulations» (selon l'expression de C. Stirling).

Dans le cadre des réseaux de Petri, nous pouvons définir les règles comme suit : le joueur 1 choisit un des deux réseaux que l'on veut comparer et tire

une des transitions franchissables. Le joueur 2 doit alors l'imiter en tirant une transition avec la *même* étiquette dans l'autre réseau. Ensuite le joueur 1 choisit un réseau, etc. Si le joueur 2 ne peut pas imiter, il perd.

Formellement :

Définition 2.1.1. *Propriété de transfert*

Soient $N_1 = (P_1, T_1, W_1, l_1)$ et $N_2 = (P_2, T_2, W_2, l_2)$ deux réseaux, une relation $R \subseteq \mathcal{M}(P_1) \times \mathcal{M}(P_2)$ entre les marquages vérifie la propriété de transfert *ssi :*

> *pour tout $M_1 R M_2$ et tout pas $M_1 \xrightarrow{t_1} M_1'$ il existe un pas de même étiquette $M_2 \xrightarrow{t_2 : l(t_1)} M_2'$ tel que $M_1' R M_2'$;*

Nous avons donc le diagramme suivant :

$$\begin{array}{ccc} M_1 & \overset{R}{\text{———}} & M_2 \\ {\scriptstyle t_1}\downarrow & & \downarrow{\scriptstyle t_2 : l(t_1)} \\ M_1 & \overset{R}{\text{———}} & M_2 \end{array}$$

Nous en dérivons la définition suivante :

Définition 2.1.2. *Bisimulation entre les marquages*

Une relation symétrique ayant la propriété de transfert est une bisimulation (entre les marquages). *Les marquages reliés sont dits* bisimilaires. *Nous le noterons R : $M_1 \underline{\leftrightarrow} M_2$.*

Deux systèmes, (N_1, M_1) et (N_2, M_2), dont les marquages initiaux sont reliés par une bisimulation sont dits bisimilaires, *noté :*

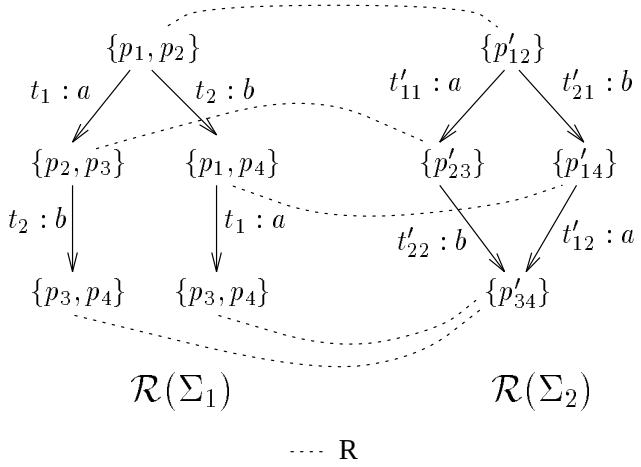
$$R : (N_1, M_1) \underline{\leftrightarrow} (N_2, M_2)$$

Sur l'exemple de la figure 1.5, p. 43, il est facile d'établir le graphe des marquages atteignables et de dessiner une relation de bisimulation entre N et N', comme le montre la figure 2.1.

Remarques :

1. Dans le cas de réseaux non étiquetés, nous aurons à définir ce qui est extérieurement visible, i.e. dans quelle mesure les transitions sont distinguables. Il y a différentes possibilités :
   - soit le tir d'une transition est distinguable et nous les étiquetons toutes différemment. Nous verrons qu'alors notre méthode ne permet que peu réductions.
   - soit les tirs sont indistinguables et nous leur donnons à toutes la même étiquette ;
   - soit autre chose encore.

Figure 2.1 : Un exemple de bisimulation



2. Notre objectif étant de réduire un réseau, nous nous intéressons au cas particulier des *auto-bisimulations*, i.e. entre un réseau et lui-même. Dans ce cas, nous pouvons simplifier la propriété de transfert en omettant la réciproque et en demandant à la bisimulation d'être une relation symétrique.

◇

Parmi les propriétés de la bisimulation nous retiendrons les suivantes :

Proposition 2.1.3. *Soient (N_i, M_i), $i \in \{1,2,3\}$, des systèmes. On note R_{ij}, R'_{ij} des bisimulations entre (N_i, M_i) et (N_j, M_j). Nous avons :*

1. *$Id_{\mathcal{M}(P_1)}$ est une bisimulation sur (N_1, M_1).*
2. *R_{12}^{-1} est une bisimulation entre (N_2, M_2) et (N_1, M_1).*
3. *$R_{12} \cup R'_{12}$ est une bisimulation entre (N_1, M_1) et (N_2, M_2).*
4. *$R_{23} \circ R_{12}$ est une bisimulation entre (N_1, M_1) et (N_3, M_3).*
5. *$\widetilde{R_{12}}$ est une bisimulation entre (N_2, M_2) et (N_1, M_1).*
6. *$\underline{\leftrightarrow} \stackrel{\text{déf}}{=} \cup\{R_{12} : R_{12}$ bisimulation entre (N_1, M_1)et$(N_2, M_2)\}$ est la plus grande bisimulation entre (N_1, M_1) et (N_2, M_2) ;*
7. *$\underline{\leftrightarrow}$ est une relation d'équivalence entre les marquages d'un réseau.*

Preuve.

- 1, 2 et 3 découlent directement de la définition de la propriété de transfert. Il est à noter que l'on peut généraliser 3 à des unions infinitaires.

- (4) vient de la compositionnalité des diagrammes :

$$\begin{array}{ccccc} M_1 & \xrightarrow{R_{12}} & M_2 & \xrightarrow{R_{23}} & M_3 \\ {\scriptstyle t_1}\downarrow & & {\scriptstyle t_2:l(t_1)}\downarrow & & \downarrow{\scriptstyle t_3:l(t_1)} \\ M_1' & \xrightarrow{R_{12}} & M_2' & \xrightarrow{R_{23}} & M_3' \end{array}$$

- 5 est une simple conséquence des précédentes.
- 6 $\underline{\leftrightarrow}$ est une bisimulation d'après 3 et c'est la plus grande par construction.
- 7 découle de 5 et 3.

□

2.2 De l'équivalence ∼

Nous parlerons d'équivalence *structurelle* puisque nous allons nous abstraire du marquage initial, et donc ne reposer que sur la structure du réseau.

Nous allons tout d'abord en donner une définition précise, puis découvrir ses propriétés pour enfin montrer qu'elle est incalculable. Le chapitre suivant montrera que la *bisimulation de place* est une approximation de ∼ (calculable efficacement).

2.2.1 L'équivalence structurelle

Nous reprenons la définition donnée informellement dans l'introduction :

Définition 2.2.1. *Équivalence structurelle*
Étant donné un réseau $N = (P, T, W, l)$, *nous appellerons* équivalence structurelle *la relation* $\sim \subseteq P \times P$ *définie par :*

$$p \sim q \overset{\text{déf}}{\Longleftrightarrow} \forall M \in \mathcal{M}(P) \ : \ M + p \underline{\leftrightarrow} M + q \tag{2.1}$$

Cette définition nous assure que ∼ est une relation d'équivalence sur les places (puisque $\underline{\leftrightarrow}$ en est une sur les marquages). Clairement, ∼ est indépendante du marquage initial.

Remarque : nous aurions pu prendre une définition plus générale en nous basant sur *une* auto-bisimulation et non sur la plus grande. Cependant l'intérêt en est restreint : nous cherchons une méthode de réduction et sommes donc intéressés par la plus grande relation, i.e. celle permettant de faire la plus forte réduction.

◇

Nous pouvons définir une relation sur les marquages à partir d'une relation sur les places :

Définition 2.2.2. *Soit $N = (P, T, W)$ un réseau et $R \subseteq P \times P$ une relation sur ses places. On note $\overline{R}$ la relation dans $\mathcal{M}(P) \times \mathcal{M}(P)$ telle que :*

$$M \overline{R} M' \stackrel{\text{déf}}{\Longleftrightarrow} \exists p_i, p'_i \in P, i \in \{1, \ldots, n\} \ : \ \begin{cases} \forall i \in \{1, \ldots, n\},\ p_i R p'_i \\ M = \{p_1, \ldots, p_n\} \\ M' = \{p'_1, \ldots, p'_n\} \end{cases}$$

Deux propriétés importantes découlent de cette définition :

Proposition 2.2.3. *Soit $N = (P, T, W)$ un réseau et $R \subseteq P \times P$ une relation sur ses places.*

1. *Si R est réflexive, alors :*

   (a) *$\overline{R}$ est* additive[1], *i.e. pour tous $M_1, M'_1, M_2, M'_2 \in \mathcal{M}(N)$:*

   $$(M_1 \,\overline{R}\, M'_1 \ \wedge \ M_2 \,\overline{R}\, M'_2) \ \Rightarrow \ (M_1 + M_2 \,\overline{R}\, M'_1 + M'_2) \quad (2.2)$$

   (b) *et c'est une* congruence, *i.e. pout tous $M_1, M_2, M_3 \in \mathcal{M}(N)$:*

   $$(M_1 \,\overline{R}\, M_2) \ \Rightarrow \ (M_1 + M_3 \,\overline{R}\, M_2 + M_3) \quad (2.3)$$

2. *R est une relation d'équivalence ssi $\overline{R}$ en est une.*

Remarques :

1. Le théorème 4.3 de [Jan93] nous permet de dire que $\overline{R}$ est un semi-linéaire.
2. Le fait d'être additive est équivalent à être une congruence dans notre contexte. (2.2) implique trivialement (2.3). Pour obtenir la réciproque, il suffit d'utiliser la transitivité de $\underline{\leftrightarrow}$.
   De plus, cela implique que $\overline{R}$ est un semi-linéaire (théo. 4.3 de [Jan93]).

◇

Nous avons alors la proposition suivante dont la preuve découle directement de la définition de $\sim$:

Proposition 2.2.4. $\overline{\sim} \subseteq \underline{\leftrightarrow}$.

Remarque : ce résultat peut être généralisé à toute relation incluse dans $\sim$.

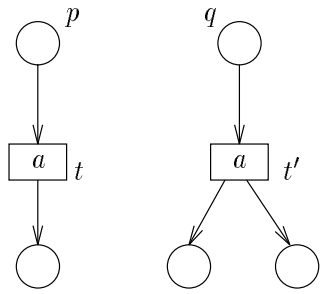
◇

Cependant, en général, $\overline{\sim}$ *n'est pas* une bisimulation, comme le prouve la figure 2.2. Sur cette figure, il est facile de voir que $\{p\} \overline{\sim} \{q\}$, mais que $t^\bullet \not\overline{\sim} t'^\bullet$.

Nous allons maintenant montrer que $\sim$ permet d'obtenir un quotient correct vis-à-vis de notre critère (la bisimulation).

[1] Nous reparlerons de cette propriété lorsque nous étudierons les propriétés des *bisimulations additives* (section 3.9.1, p. 95).

Figure 2.2 : $\overline{\sim}$ n'est pas toujours une bisimulation.



N

2.2.2 Le réseau quotient

Étant donné un réseau $N = (P, T, W)$ et E une équivalence sur P, nous quotientons de manière classique en faisant toutefois un léger abus de notation : les transitions du réseau quotient seront notées $[t]_E$, bien que E ne soit que sur P. Cela rend la définition de $W/_E$ plus aisée et un certain nombre de preuves moins confuses[2].

Définition 2.2.5. *Réseau et marquage quotient*

Soit $E \subseteq P \times P$ une relation d'équivalence sur les places d'un réseau $N = (P, T, W, l)$. Le réseau quotient $N/_E \stackrel{\text{déf}}{=} (P/_E, T/_E, W/_E, l/_E)$ *est tel que :*

- $P/_E \stackrel{\text{déf}}{=} \{[p]_E \ : \ p \in P\}$;
- $T/_E \stackrel{\text{déf}}{=} T$;
- $W/_E([x]_E, [y]_E) \stackrel{\text{déf}}{=} \sum_{u \in [x], v \in [y]} W(u, v)$;
- $l/_E \stackrel{\text{déf}}{=} l$.

Il nous faut maintenant exprimer de quelle manière nous quotientons un marquage. Nous étendons[3] $[\cdot]_E$ *à* $[\cdot]_E \ : \mathcal{M}_N \to \mathcal{M}_{N/E}$, *avec :*

$$M/_E \stackrel{\text{déf}}{=} [M]_E = [\{p_1, p_2, \ldots, p_n\}]_E \stackrel{\text{déf}}{=} \{[p_1]_E, [p_2]_E, \ldots, [p_n]_E\}$$

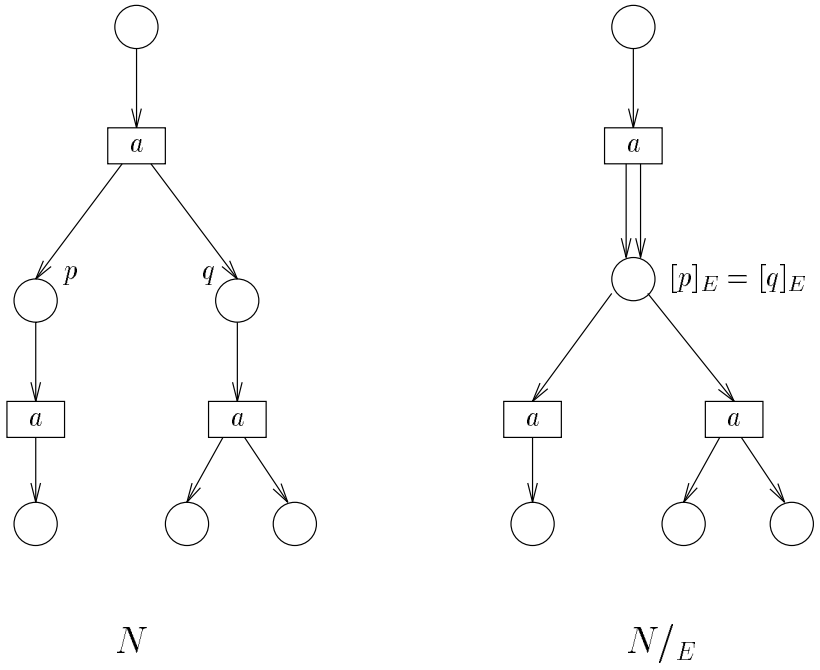
[2]Cela revient en fait à considérer $E' \stackrel{déf}{=} E \cup Id_T$.

[3]Cette définition est cohérente avec celle de $\overline{[\cdot]_E}$ (déf. 2.2.2, p. 56).

Par conséquent, pour un système $\Sigma = (N, M)$, *nous avons* $\Sigma/_E \stackrel{\text{déf}}{=} (N/_E, M/_E)$.

Informellement, nous faisons donc «glisser» tous les arcs sur une place représentant la classe définie par E (voir la figure 2.3).

Figure 2.3 : Un exemple de réseau quotient.



Avec ces définitions, nous aurons ${}^\bullet[t]_E = [{}^\bullet t]_E$. Donc, si on a un pas $M \stackrel{t}{\to} M'$ dans N, alors le pas $[M]_E \stackrel{[t]_E}{\longrightarrow} M'_E$ existe dans $N/_E$. L'inverse n'est pas vrai, comme le montre la figure 2.4 où N et $N/_E$. Par contre, si $[M]_E \stackrel{[t]_E}{\longrightarrow} M'_E$ alors il existe $M_1, M'_1 \in \mathcal{M}_N$ tels que $[M_1]_E = [M]_E, [M'_1]_E = [M']_E$ et $M_1 \stackrel{t}{\to} M'_1$.

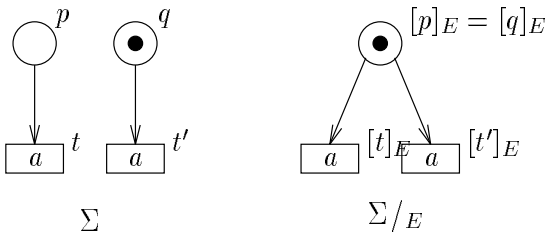
Nous concrétisons cela par un petit lemme technique qui nous sera utile ultérieurement :

Lemme 2.2.6. *Soit* $N = (P, T, W)$ *un réseau,* $t \in T$ *et* $M, M' \in \mathcal{M}(N)$. $[\cdot]_E$ *étant tel qu'en la définition 2.2.5, nous avons :*

$$M \stackrel{t}{\to} M' \;\Rightarrow\; [M]_E \stackrel{[t]_E}{\longrightarrow} M'_E$$

$$[M]_E \stackrel{[t]_E}{\longrightarrow} M'_E \;\Rightarrow\; \exists M_1, M'_1 \;:\; \begin{cases} [M]_E = [M_1]_E,\ [M']_E = [M'_1]_E \\ M_1 \stackrel{t}{\to} M'_1 \end{cases}$$

Figure 2.4 : $\mathcal{R}(\Sigma)$ et $\mathcal{R}(\Sigma/_E)$ n'ont pas toujours isomorphes.



Remarque : en fait, plus généralement, pour tous les opérateurs de comparaison (notations 1.2.1, p. 28), nous avons : $(M_1\ op\ M_2)$ dans Σ implique $(M_1/_E\ op\ M_2/_E)$ dans $\Sigma/_E$. La réciproque est : $(M_1\ op\ M_2)$ dans $N/_E$ implique qu'il existe des marquages M_1', M_2' dans Σ tel que $(M_1'\ op\ M_2')$ et $M_1' = M_1/_E, M_2/_E = M_2'$.

◇

De cette définition, nous dérivons le petit lemme suivant qui établit la cohérence entre $[\cdot]_E$ et $\overline{E}$:

Lemme 2.2.7. *Soit un réseau $N = (P, T, W, l)$ et $E \subseteq P \times P$ une relation d'équivalence sur ses places. Nous avons :*

$$\forall M, M' \in \mathcal{M}_N \ : \ [M]_E = [M']_E \Leftrightarrow M \overline{E} M'$$

Par suite, $(\mathcal{M}_N)/_E = \mathcal{M}_{(N/_E)}$.

Nous donnons maintenant deux théorèmes importants :

- le théo. 2.2.8 qui nous permet de quotienter. C'est une généralisation du lemme 6 de [ABS91b].
- le théo. 2.2.10 qui est sa réciproque et montre que $\sim$ est la plus grande relation nous permettant de quotienter correctement.

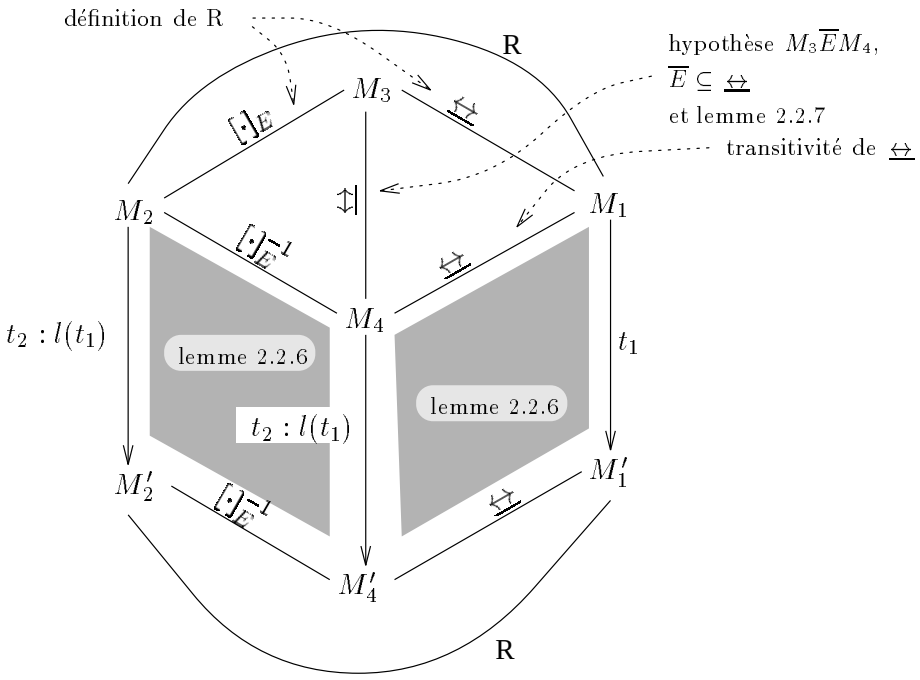
Théorème 2.2.8. *Correction du quotient*

Soit un réseau $N = (P, T, W, l)$, si $E \subseteq P \times P$ est une équivalence telle que $\overline{E} \subseteq \underline{\leftrightarrow}$, alors pour tout marquage initial M, (N, M) et $(N/_E, M/_E)$ sont bisimilaires, i.e. :

$$\forall M \in \mathcal{M}_N,\ (N, M) \underline{\leftrightarrow} (N/_E, M/_E)$$

Preuve. Il suffit de montrer que $R \stackrel{déf}{=} [\cdot]_E \circ \underline{\leftrightarrow} \subseteq \mathcal{M}_N \times \mathcal{M}_{N/_E}$ est une bisimulation entre N et $N/_E$, donc de vérifier la propriété de transfert pour R.

La figure ci-dessous explicite cette démonstration.



1. Soient $M_1 \xrightarrow{t_1} M_1'$ et $M_1 R M_2$. Le diagramme suivant établit le premier point de la propriété de transfert :

$$\begin{array}{ccccc} M_1 & \underline{\leftrightarrow} & M_3 & [\cdot]_E & M_2 = [M_3]_E \\ {\scriptstyle t_1}\downarrow & & {\scriptstyle t_3 : l(t_2)}\downarrow & & \downarrow{\scriptstyle t_3 : l(t_1)} \\ M_1' & \underline{\leftrightarrow} & M_3' & [\cdot]_E & M_2' = [M_3']_E \end{array}$$

   t_3 existe par bisimilarité et t_2 est directement issue du lemme précédent (2.2.6).

2. Soit maintenant un pas $M_2 \xrightarrow{t_2} M_2'$ et $M_1 R M_2$. De $M_1 R M_2$, nous savons qu'il existe un marquage M_3 tel que $M_1 \underline{\leftrightarrow} M_3 [\cdot]_E M_2$. D'après le lemme 2.2.6, nous savons qu'il existe un pas $M_4 \xrightarrow{t_2} M_4'$ fermant le diagramme suivant :

$$\begin{array}{ccc} M_2 & [\cdot]_E^{-1} & M_4 \\ {\scriptstyle t_2}\downarrow & & \downarrow{\scriptstyle t_2} \\ M_2' & [\cdot]_E^{-1} & M_4' \end{array}$$

   Par suite $M_3 \overline{E} M_4$ et, comme $\overline{E} \subseteq \underline{\leftrightarrow}$, $M_3 \underline{\leftrightarrow} M_4$. Comme $M_1 \underline{\leftrightarrow} M_3$, nous avons $M_4 \underline{\leftrightarrow} M_1$ et la propriété de transfert nous permet de compléter le diagramme :

$$\begin{array}{ccccc} M_2 & [\cdot]_E^{-1} & M_4 & \underline{\leftrightarrow} & M_1 \\ {\scriptstyle t_2}\downarrow & & {\scriptstyle t_2}\downarrow & & \downarrow{\scriptstyle t_1 : l(t_2)} \\ M_2' & [\cdot]_E^{-1} & M_4' & \underline{\leftrightarrow} & M_1' \end{array}$$

Ce qui complète la preuve.

□

Remarque sur la preuve du théorème 2.2.8 : il est important de noter que ce théorème ne dépend que de trois hypothèses :

1. la manière de quotienter (lemme 2.2.6) ;
2. la transitivité de la bisimulation ;
3. l'imitation d'*une* transition.

En effet, cela nous permettra de l'étendre ultérieurement aux bisimulations prenant en compte les actions invisibles.

◇

Corollaire 2.2.9. *Soit un réseau $N = (P, T, W, l)$, nous avons :*

$$\forall M \in \mathcal{M}_N,\ (N, M) \underline{\leftrightarrow} (N/_{\sim}, M/_{\sim})$$

De plus, $\sim$ est la plus grande relation d'équivalence qui nous permette de quotienter :

Théorème 2.2.10.

$$\overline{E} \not\subseteq \underline{\leftrightarrow} \Rightarrow \exists M \ :\ (N, M) \underline{\not\leftrightarrow} (N/_E, M/_E)$$

Preuve. Puisque $\overline{E} \not\subseteq \underline{\leftrightarrow}$, nous en déduisons l'existence dans N de M_1, M_2 tels que $M_1 \overline{E} M_2$ et $M_1 \underline{\not\leftrightarrow} M_2$. Par suite, il existe un pas $M_1 \stackrel{t_1}{\rightarrow} M_1'$ qui ne peut être imité par un pas $M_2 \stackrel{t_2 : l(t_1)}{\rightarrow} M_2'$ avec $M_1' \underline{\leftrightarrow} M_2'$.

Or, comme $M_1 \overline{E} M_2$, dans $N/_E$ il existe un pas $[M_2]_E \stackrel{t_1}{\rightarrow} [M_1']$. Par suite, $(N, M_2) \underline{\not\leftrightarrow} (N/_E, M_2/_E)$.

□

$\sim$ est donc *la* relation que nous cherchons, malheureusement nous allons voir qu'elle n'est pas calculable.

2.3 Calculabilité de $\sim$

Si nous pouvions décider de la bisimilarité de deux systèmes, nous serions sûrs de pouvoir calculer $\sim$ (en prenant toutes les relations sur les places). Malheureusement nous avons :

Théorème 2.3.1. *[Jan94]*
La bisimilarité de deux systèmes est indécidable.

Il nous reste donc à chercher si $\sim$ est calculable. Cette section est consacrée au théorème suivant :

Théorème 2.3.2. $\sim$ *n'est pas calculable.*

Une des méthodes les plus directes pour établir l'indécidabilité d'un problème est d'y ramener un des problèmes indécidables pour une machine universelle (machine de Turing, machine à compteurs, etc).

Malheureusement cette méthode échoue dans le cadre des réseaux de Petri, compte-tenu de leur incapacité à simuler un branchement conditionnel (on ne peut tester la vacuité d'une place). Nous allons présenter une manière, pour un réseau, de simuler en un sens plus faible une machine universelle. Les sections qui suivent présentent :

- un bref rappel sur les machines à compteurs de MINSKY (l'annexe A est plus détaillée et présente les résultats d'indécidabilité ainsi que l'idée de leur preuve) ;
- la méthode utilisée par JANCAR dans [Jan94] pour prouver l'indécidabilité de la bisimulation et les modifications nécessaires afin de l'adapter à l'équivalence $\sim$;
- enfin, la preuve de l'indécidabilité de $\sim$.

Remarque : bien que le résultat du théorème 2.3.2 nous suffise, nous avons en plus :

$$\sim \textit{ est co-semidécidable}$$

En effet, $p \not\sim q$ ssi $\exists M \in \mathcal{M}_N \ : \ p + M \not\leftrightarrows q + M$. Or la non-bisimilarité est semi-décidable dans les réseaux de Petri (voir par exemple [Mil89, Chr92]).

$\diamond$

2.3.1 Rappels sur les machines à compteurs

À la différence des réseaux de Petri, les machines à compteurs sont déterministes[4].

Définition 2.3.3. *Machines à compteurs*

Une machine à compteurs *est un triplet* $\mathcal{C} = (Q, C, I)$ *où :*

- $Q = \{s_0, s_1, s_2, \ldots, s_n\}$ *est l'ensemble des* étapes *(i.e. un programme séquentiel) ;*
- $C = \{c_1, c_2, \ldots, c_m\}$ *est l'ensemble des* compteurs *(i.e. des registres à capacité infinie) ;*

[4] C'est du moins cette définition que nous utilisons. [Iba78] définit des machines non-déterministes.

$I : Q \to$ `Ins` *définit pour chaque étape l'opération à effectuer, avec* $I(s_n) = halte$.

Nous prendrons comme types d'instructions possibles (ensemble `Ins`*) :*

ins_1 ::	$c_l++\ ;\ s_i$	*ajoute 1 au compteur* c_l *et saute à l'instruction* s_i
ins_2 ::	*si* $c_l > 0$ *alors* $c_l--\ ;\ s_i$ *sinon* s_j	*si le compteur* c_l *est positif, alors on lui soustrait 1 et on va à* s_i*, sinon on passe à* s_j
ins_3 ::	*halte*	*arrête la machine*

$i, j \in \{0, \ldots, n\}$ *et* $l \in \{1, \ldots, m\}$.

Un état *de la machine est donné par un couple* $\epsilon = (s, \theta)$ *donnant l'étape atteinte et le contenu des compteurs (*$s \in Q$ *et* $\theta : C \to \mathbb{N}$*).*

Le comportement d'une machine (ses changements d'état) est donc donné par :

$$(s_k, \theta) \xrightarrow{ins_1} (s_i, \theta + \delta_{c_l})$$

$$(s_k, \theta) \xrightarrow{ins_2} \begin{cases} (s_i, \theta - \delta_{c_l}) & \text{si } \theta(c_l) > 0 \\ (s_j, \theta) & \text{sinon} \end{cases}$$

où $\delta_{c_l} : C \to \mathbb{N}$ est la fonction caractéristique de c_l (i.e. $\delta(c) = 1$ ssi $c = c_l$).

Parmi les problèmes indécidables de [Min67], nous retiendrons les deux suivants :

le problème de l'arrêt (PBA) pour une machine $\mathcal{C}$ et une configuration θ s'exprime comme suit :

«$\mathcal{C}$ s'arrête-t-elle en partant de (s_0, θ) ?»

le problème de l'arrêt uniforme (PBAU) pour une machine $\mathcal{C}$ s'exprime comme suit :

«$\mathcal{C}$ s'arrête-t-elle en partant de (s_0, θ) quel que soit θ ?»

Il est facile de passer d'une machine $\mathcal{C} = (Q, C, I)$ à un réseau la simulant (faiblement) et noté $N_{\mathcal{C}} = (P_{\mathcal{C}}, T_{\mathcal{C}}, W_{\mathcal{C}}, l_{\mathcal{C}})$:

- $P_{\mathcal{C}} \stackrel{déf}{=} Q \cup C$
- $T_{\mathcal{C}}, W_{\mathcal{C}}$ sont définis comme suit :
  - si $I(s_k)$ est de la forme ins_1, nous ajoutons[5] $s_k \xrightarrow{t_k} s_i + c_l$;

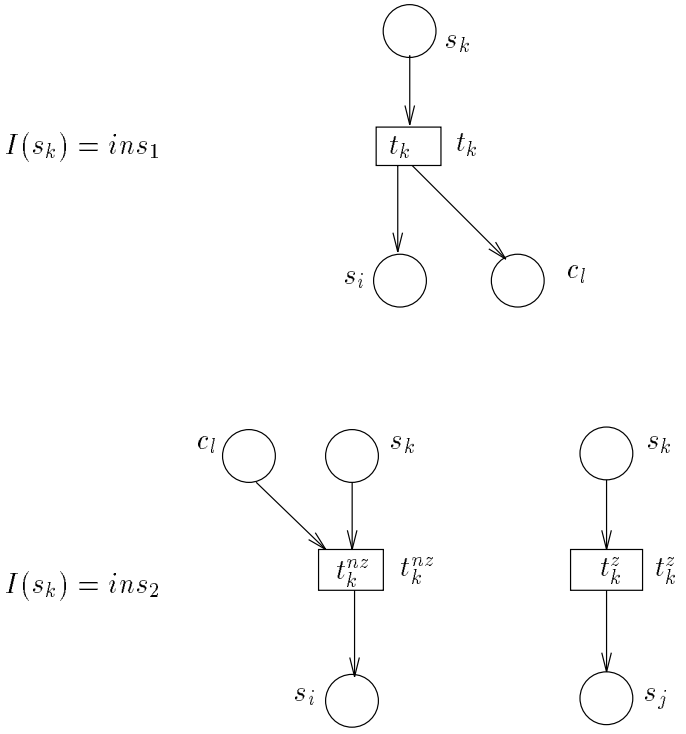
[5] L'ajout de $M \xrightarrow{t} M'$ se fait en ajoutant t à $T_{\mathcal{C}}$ et l'ensemble d'arcs $\{(M(p), t),\ (t, M'(p)),\ p \in P_{\mathcal{C}}\}$ à $W_{\mathcal{C}}$.

– si $I(s_k)$ est de la forme ins_2, nous ajoutons $s_k + c_l \xrightarrow{t_k^{nz}} s_i$ et $s_k \xrightarrow{t_k^z} s_j$;

- enfin, $\forall t \in T_{\mathcal{C}}$: $l_{\mathcal{C}}(t) = t$

La figure 2.5 explicite ces constructions.

Figure 2.5 : Des machines à compteurs aux réseaux de Petri.

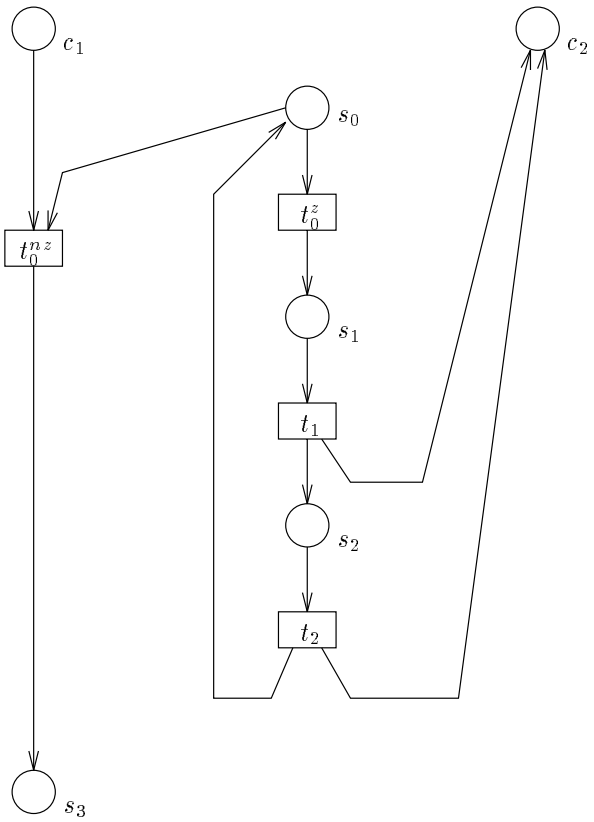


Exemple 2.1 : Une machine à compteurs.

Prenons le petit programme ci-dessous :

s_0 :	si $c > 0$	alors $c_1 - -; s_3$
		sinon s_1
s_1 :	$c_2 + +; s_2$	
s_2 :	$c_2 + +; s_2$	
s_3 :	halte	

Il se traduira par le réseau :



Nous allons maintenant expliciter la manière de simuler une machine à compteur par un réseau. Tout d'abord, nous représentons les états de la machine par les marquages :

Notations 2.3.1. *Soit $\epsilon = (s, \theta)$, un état de $\mathcal{C}$, on note M_ϵ le marquage représentant ϵ dans $N_\mathcal{C}$, i.e. :*

$$M_\epsilon(x) \stackrel{\text{déf}}{=} \begin{cases} 1 & si\ x = s \\ 0 & si\ x \in Q \setminus \{s\} \\ \theta(x) & si\ x \in C \end{cases}$$

$N_\mathcal{C}$ peut simuler $\mathcal{C}$, puisque s'il existe un pas $\epsilon \to \epsilon'$ dans $\mathcal{C}$, alors il existe un pas $M_\epsilon \to M_{\epsilon'}$ dans $N_\mathcal{C}$. Mais il la simule en un sens faible puisque la réciproque n'est pas nécessairement vraie (il suffit de prendre t_j^z qui peut être franchie même si c_l est non-vide).

2.3.2 La méthode de JANCAR

Dans [Jan94], JANCAR utilise PBA et propose une méthode générique pour établir l'indécidabilité d'une équivalence sur les marquages : pour prouver l'indécidabilité d'une X-équivalence, on construit deux variantes d'un

réseau $N_{\mathcal{C}}$ simulant faiblement une machine $\mathcal{C}$ telles que $\mathcal{C}$ s'arrête ssi les deux variantes ne sont pas X-équivalentes.

Nous allons adapter cette méthode au cas de notre relation structurelle $\sim$. Nous aurons ainsi à ajouter un certain nombre d'éléments à $N_{\mathcal{C}}$ pour obtenir que le problème de $p_1 \sim p_2$ soit équivalent à celui de l'arrêt de la machine.

Le problème diffère de celui de JANCAR à cause du «$\forall M \in \mathcal{M}_N$» de notre définition (déf. 2.2.1, p. 55). Pour résoudre ce problème nous allons «diviser pour régner». $\mathcal{M}_N$ contiendra deux classes de marquages vis-à-vis de $\underline{\leftrightarrow}$: ceux correspondant à un état de la machine (que nous appellerons *valides*) et les autres. En rendant tous les marquages non-valides bisimilaires nous ramenons le problème à l'étude des marquages valides, i.e. à ceux simulant un comportement de $\mathcal{C}$. Cependant, il restera à considérer le fait que l'état initial de la machine est spécifié : nous ne sommes pas forcément en s_0. Pour régler ce problème, nous allons faire boucler le réseau au moins une fois sur s_0, mais de manière différente selon que p_1 ou p_2 sera marquée. Par suite, nous pourrons utiliser PBAU.

2.3.3 Preuve du théorème 2.3.2

Pour construire nos réseaux, nous allons ajouter à $N_{\mathcal{C}}$ un certain nombre d'éléments :

- deux places p_1 et p_2 (dont on voudra savoir si $p_1 \sim p_2$) ;
- des transitions nous permettant de tricher à tous les coups et qui, de plus, échangeront les jetons des places p_1 et p_2. Ainsi le réseau simule correctement la machine ssi il la simule sans tricher.
- une transition permettant de distinguer les places p_1 et p_2 à la terminaison de $\mathcal{C}$;
- nous allons ajouter enfin un certain nombre de transitions nous permettant de distinguer les marquages *valides* des autres.

Nous justifierons ces ajouts un peu après.
Formellement, nous créons un réseau $N = (P, T, W, l)$ tel que :

- $P = P_{\mathcal{C}} \cup P'$, $P' = \{p_1, p_2, p_3\}$. La place p_3 nous servira à différencier p_1 et p_2.
- T, W, l incluent $T_{\mathcal{C}}, W_{\mathcal{C}}, l_{\mathcal{C}}$ resp. et nous leur ajoutons :
  1. une transition revenant sur s_0 sous la condition p_1 : $s_n + p_1 \xrightarrow{t_1^b : b} s_0 + p_1$;
  2. son homologue sous la condition p_2 : $s_n + p_2 \xrightarrow{t_2^b : b} s_0 + p_3$.

3. une transition distinguant les conditions d'arrêt : $s_n + p_3 \stackrel{t^f:f}{\rightarrow} \emptyset$;
4. pour tous les s_i tels que $I(s_i) = ins_2$ (branchement), des transitions échangeant un jeton entre p_1 et p_2 et trichant (i.e. faisant croire qu'un compteur est vide) : $c + p_1 + s_i \xrightarrow{t_i^e:t_i^z} c + p_2 + s_j$ et $c + p_2 + s_i \xrightarrow{t_i^{e'}:t_i^z} c + p_1 + s_j$.
5. les transitions distinguant les marquages valides des non-valides :

$$\begin{aligned}
\forall i,j \in \{1,2,3\} \quad &: \quad p_i + p_j \stackrel{t_{ij}^x:x}{\longrightarrow} p_i + p_j \\
\forall i,j \in \{0,\ldots,n\} \quad &: \quad s_i + s_j \stackrel{t_{ij}^{x'}:x}{\longrightarrow} s_i + s_j \\
\forall i \in \{0,\ldots,n\} \quad &: \quad p_1 + (s_n + p_i) \stackrel{t_{1i}^b:b}{\rightarrow} p_3 + (s_0 + p_i) \\
&\qquad\quad p_1 + (s_n + s_i) \stackrel{t_{1i}^{b'}:b}{\rightarrow} p_3 + (s_0 + s_i) \\
&\qquad\quad p_2 + (s_n + s_i) \stackrel{t_{2i}^b:b}{\rightarrow} p_2 + (s_0 + s_i) \\
&\qquad\quad p_2 + (s_n + s_i) \stackrel{t_{2i}^{b'}:b}{\rightarrow} p_2 + (s_0 + s_i)
\end{aligned}$$

Nous allons expliquer ou justifier maintenant ces divers ajouts :

- t_1^b et t_2^b (fig. 2.6) permettent de tenir compte du $\forall M \in \mathcal{M}_N$ de la définition en faisant boucler la machine sur elle-même.

  En effet, le «$\forall M \in \mathcal{M}_N$» de la définition 2.2.1 a pour conséquence de laisser le compte-tenu des compteurs arbitraire (ce que nous résolvons par PBAU) *et* de faire débuter le programme à une étape quelconque. En faisant boucler la machine, nous nous assurons de nous ramener à l'étape s_0 (dans le cas où la machine s'arrêterait).

- t_1^b et t_2^b ont en plus pour objectif, avec t^f, de différencier un retour sur s_0 par p_1 d'un par p_2 grâce au transfert vers p_3.

  Intuitivement, elles seules permettent de distinguer p_1 de p_2, puisque les autres transitions en sont indépendantes ou les utilisent de manière «symétrique».

- les $t_i^e, t_i^{e'}$ (fig. 2.7) sont utiles pour montrer l'indécidabilité. Elles ont le même effet que t_i^z, mais les franchir signifie toujours tricher.

  Intuitivement, à partir de p_2, on ne pourra revenir plusieurs fois sur s_0 qu'en trichant.

- les $t_{1i}^b, t_{1i}^{b'}, t_{2i}^b, t_{2i}^{b'}$ (fig. 2.8) ont pour objectif de rendre tous les marquages non valides bisimilaires en permettant l'imitation de t_1^b et t_2^b.

Figure 2.6 : Des transitions pour boucler de s_n à s_0.

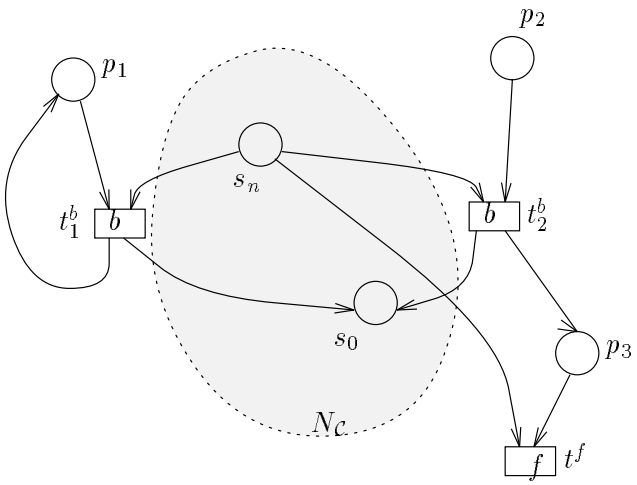
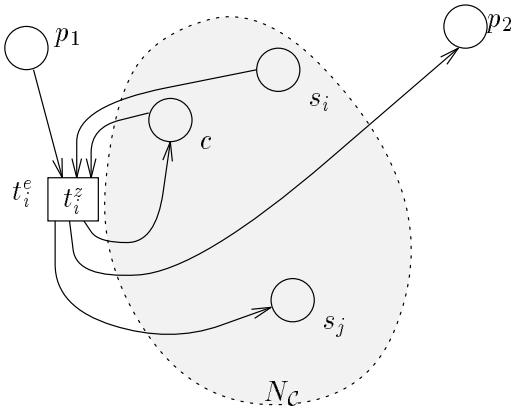
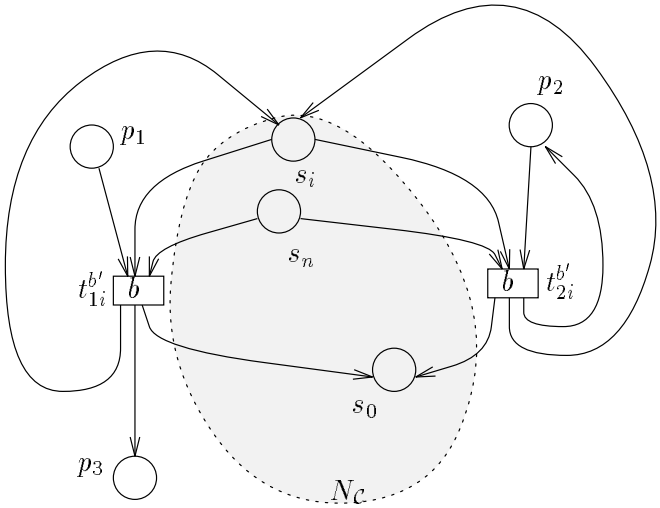


Figure 2.7 : Des transitions pour passer des jetons de p_1 à p_2.



et symétriquement pour p_2 et $t_i^{e'}$.

Figure 2.8 : Des transitions pour séparer les marquages.



et similairement pour t_{1i}^{b} et t_{2i}^{b}

- enfin, les $t_{ij}^{x}, t_{ij}^{x'}$ et t^{f} nous permettront de séparer les marquages valides des autres.

Nous pouvons maintenant aborder le lemme essentiel :

Lemme 2.3.4. *Dans N, $p_1 \sim p_2$ si et seulement si $\mathcal{C}$ ne s'arrête jamais*[6] *en partant de s_0.*

Avant d'aborder sa preuve, nous allons montrer que les marquages valides et non valides sont bien séparés par $\underline{\leftrightarrow}$ dès lors qu'on ajoute un jeton dans une place p_i, $i \in \{1,2,3\}$.

Lemme 2.3.5. *Soient $M, M' \in \mathcal{M}_N$ tels que M soit valide et M' non. Alors :*

$$\forall i,j \in \{1,2,3\} \ : \ p_i + M \not\underline{\leftrightarrow} p_j + M'$$

Preuve.

Nous allons simplement énumérer les cas d'invalidité d'un marquage :

- si $M'(Q) = M'(P') = 0$. Dans ce cas, $p_j + M'$ est un marquage mort, mais $p_i + M$ ne l'est pas :
  - soit $M(s_n) = 0$, alors $p_i + M$ permet de franchir une transition de $N_{\mathcal{C}}$;

[6] i.e. $\mathcal{C}$ ne s'arrête pour aucune valeur des compteurs.

– soit $M(s_n) = 1$ et $p_i + M$ peut tirer une des transitions t_1^b, t_2^b ou t^f.

- si $M'(P') > 0$ ou $M'(Q) > 1$, alors $p_j + M' \xrightarrow{x}$ ne peut être imité par $p_i + M$.

D'où le lemme.

□

Nous avons aussi besoin de montrer que p_1 et p_2 sont «neutres» dans le cas de marquages non valides, formellement :

Lemme 2.3.6. *Pour tout marquage M non-valide, nous avons :*

$$p_1 + M \underline{\leftrightarrow} p_2 + M$$

Preuve. Nous construisons la relation R telle que :

- $Id_{\mathcal{M}_N} \in R$;
- pour tout marquage M non-valide, $(p_1+M)\ R\ (p_2+M)$ et réciproquement.

Il nous reste à vérifier que R est une bisimulation.
Nous étudions les différentes transitions :

- pour $t \in T_{\mathcal{C}} \cup \{t_{ij}^{x'}\}_{i,j\in\{1,2,3\}} \cup \{t^f\}$, nous avons $p_1, p_2 \notin {}^\circ t$, donc :

$$\begin{array}{ccc} p_1 + M & \overset{R}{\text{———}} & p_2 + M \\ {\scriptstyle t}\downarrow & & \downarrow{\scriptstyle t} \\ p_1 + M' & \overset{R}{\text{———}} & p_2 + M' \end{array}$$

- pour $t \in \{t_i^e, t_i^{e'}\}_i$, nous obtenons :

$$\begin{array}{ccc} p_1 + M & \overset{R}{\text{———}} & p_2 + M \\ {\scriptstyle t_i^e}\downarrow & & \downarrow{\scriptstyle t_i^{e'}} \\ p_2 + M' & \overset{R}{\text{———}} & p_1 + M' \end{array}$$

- pour $t \in \{t_{ij}^x\}$, il existe toujours $t' \in \{t_{ij}^x\}$ telle que :

$$\begin{array}{ccc} p_1 + M & \overset{R}{\text{———}} & p_2 + M \\ {\scriptstyle t}\downarrow & & \downarrow{\scriptstyle t'} \\ p_1 + M & \overset{R}{\text{———}} & p_2 + M \end{array}$$

- le cas le plus intéressant est pour $t \in \{t_1^b, t_2^b\} \cup \{t_{1i}^b, t_{1i}^{b'}, t_{2i}^b, t_{2i}^{b'}\}_{i \in \{0,\ldots,n\}}$. Si $M \stackrel{t}{\not\to}$, alors trivialement :

$$\begin{array}{ccc} p_1 + M & \overset{R}{\text{———}} & p_2 + M \\ {\scriptstyle t}\big\downarrow & & \big\downarrow{\scriptstyle t} \\ p_1 + M' & \overset{R}{\text{———}} & p_2 + M' \end{array}$$

Considérons donc le cas $M \stackrel{t}{\not\to}$. Nous allons étudier les différents cas d'invalidité du marquage[7] :

1. si $M(s_n) = 0$ (donc en particulier si $M(Q) = 0$), alors $p_1 + M \stackrel{t}{\not\to}$ et $p_2 + M \stackrel{t}{\not\to}$;
2. si $M(s_n) > 0$, nous considérons les deux cas *non exclusifs* ci-dessous :
   (a) M est de la forme $s_n + p_i + M'$ (cas $M(P') > 0$), nous avons :

$$\begin{array}{ccc} p_1 + s_n + p_i + M' & \overset{R}{\text{———}} & p_2 + s_n + p_i + M' \\ {\scriptstyle t_1^b}\big\downarrow & & \big\downarrow{\scriptstyle t_{2i}^b} \\ p_1 + s_0 + p_i + M' & \overset{R}{\text{———}} & p_2 + s_0 + p_i + M' \end{array}$$

   et d'autre part :

$$\begin{array}{ccc} p_1 + s_n + p_i + M' & \overset{R}{\text{———}} & p_2 + s_n + p_i + M' \\ {\scriptstyle t_{1i}^b}\big\downarrow & & \big\downarrow{\scriptstyle t_2^b} \\ p_3 + s_0 + p_i + M' & \overset{R}{\text{———}} & p_3 + s_0 + p_i + M' \end{array}$$

   (b) M est de la forme $s_n + s_i + M'$ (cas $M(Q) > 1$), nous avons :

$$\begin{array}{ccc} p_1 + s_n + s_i + M' & \overset{R}{\text{———}} & p_2 + s_n + s_i + M' \\ {\scriptstyle t_1^b}\big\downarrow & & \big\downarrow{\scriptstyle t_{2i}^{b'}} \\ p_1 + s_0 + s_i + M' & \overset{R}{\text{———}} & p_2 + s_0 + s_i + M' \end{array}$$

   et d'autre part :

$$\begin{array}{ccc} p_1 + s_n + s_i + M' & \overset{R}{\text{———}} & p_2 + s_n + s_i + M' \\ {\scriptstyle t_{1i}^{b'}}\big\downarrow & & \big\downarrow{\scriptstyle t_2^b} \\ p_3 + s_0 + s_i + M' & \overset{R}{\text{———}} & p_3 + s_0 + s_i + M' \end{array}$$

R vérifie donc la propriété de transfert. Par suite R est une bisimulation et $R \subseteq \underline{\leftrightarrow}$ nous donne le lemme.

□

[7] Il y a trois cas d'invalidité : $M(Q) = 0$, $M(Q) > 1$ et $M(P') > 0$.

La conséquence directe des deux lemmes précédents est :

$$p_1 \sim p_2 \; ssi \; M + p_1 \underline{\leftrightarrow} M + p_2, \; \forall M \; valide$$

Nous pouvons donc nous restreindre aux marquages valides (i.e. N simulant faiblement la machine).

Preuve du lemme 2.3.4.

Soient $\epsilon = (s, \theta)$ l'état de la machine, différents cas se présentent à nous :

1. nous faisons un cas particulier pour $s = s_0$. Nous avons :

   $$s_0 \in M \Rightarrow (p_1 \sim p_2 \text{ ssi } \mathcal{C} \text{ ne s'arrête jamais})$$

   Or ce problème, que nous noterons D-PBAU (dual de PBAU), est indécidable (voir le théorème A.2.1, p. 194).

2. si $\mathcal{C}$ s'arrête en partant de $\epsilon = (s, \theta)$, les transitions t_1^b et t_2^b font que nous relançons la machine dans un état $\epsilon' = (s_0, \theta')$, donc nous sommes dans le cas précédent.

3. si $\mathcal{C}$ ne s'arrête pas en partant de $\epsilon = (s, \theta)$, nous savons qu'alors $M + p_1 \underline{\leftrightarrow} M + p_2$ ([Jan94]).

Il découle que décider de $p_1 \sim p_2$ est équivalent à D-PBAU qui est indécidable.

□

$p_1 \sim p_2$ est donc indécidable.

Conclusion

Nous avons maintenant cerné les principales propriétés désirables pour obtenir notre méthode de réduction :

1. être une congruence pour la mise en contexte ;

2. telle que le quotient obtenu soit bisimilaire au système de départ ;

3. être calculable, si possible efficacement.

La première approche, $\sim$, ne satisfait pas au dernier critère. Nous allons donc chercher une relation d'équivalence incluse dans $\sim$ (ce qui nous assure de la correction du quotient grâce au lemme 2.2.8, p. 59 et est nécessaire puisque $\sim$ est la plus grande permettant de quotienter).

Notes bibliographiques

La bisimulation a été définie dans [Mil83] et [Par81]. Elle est décidable pour quelques classes de réseaux (réseaux d'états finis et certaines classes d'états infinis [BS87, CH93b], «Basic Parallel Process» [Kul94, CHS92]), mais pas en général ([Jan94]).

Elle possède un bon nombre de variantes présentées par exemple dans [BDKP91, Wei89], ou plus généralement les équivalences sur les réseaux sont reliées dans [DDM87, Vog94, PRS92]. Le lecteur pourra aussi consulter la section 3.8 et les chapitres 4 et 5.

[EN94], outre les résultats de JANCAR, présente de nombreux résultats d'indécidabilité dans le cadre des réseaux de Petri. D'autres résultats de complexité sont dans [JLL77].

Sur la calculabilité en général le lecteur pourra consulter avec profit [Min67] et [Bri95].

Chapitre 3

La bisimulation de places stricte

Nous étudions dans ce chapitre une variante de $\sim$: la *bisimulation de places stricte* («stricte» par opposition aux versions dérivées que nous exposerons dans la deuxième partie). Cette relation qui est calculable et pour laquelle il existe un algorithme efficace.

C'est Olderog qui le premier a proposé une notion de bisimulation de places dans [Old89, Old91a] où il utilise le terme de «strong bisimulation». Même si sa présentation mettait en avant d'autres objectifs, Olderog proposait une méthode simple permettant de prouver que deux réseaux avaient les mêmes processus concurrents. Pour cela, Olderog proposait de mettre en relation les places des deux réseaux, d'une façon vérifiant certaines contraintes locales.

Dans [ABS91a], la proposition d'Olderog est corrigée et les propriétés de la notion (revue) de bisimulation de places sont étudiées en détail. En particulier, il y est montré comment la bisimulation de place permet de réduire les réseaux de Petri. C'est dans [AS92] que les applications à la réduction sont mises en avant, et qu'on montre comment la bisimulation de places préserve aussi certains aspects causaux des comportements. Un outil informatique a été réalisé pour implémenter les algorithmes ([Pfi92]).

Dans ce chapitre, la première section établit le lien entre $\sim$ et la bisimulation de places stricte (que nous abrégerons par BdPS) et donne les propriétés de base. Ensuite, nous nous intéressons à l'existence d'une plus grande BdPS, notée $B(N)$.

Les deux sections suivantes présentent la méthode de réduction dans sa globalité ainsi qu'une propriété caractéristique des BdPS. De cette propriété nous dériverons un algorithme de calcul de $B(N)$.

Ensuite, nous étudions le coût théorique de la réduction et présentons un exemple de déroulement de l'algorithme.

Nous terminons par quelques remarques générales sur les bisimulations de places et sur diverses sémantiques d'ordre partiel.

3.1 La bisimulation de places stricte

Nous allons rechercher une équivalence qui soit incluse dans $\sim$, ce qui permettra de conserver le théorème du quotient correct et qui en soit une approximation calculable. Enfin, nous montrerons qu'elles diffèrent.

3.1.1 De $\sim$ à la bisimulation de places stricte

Nous définissons :

Définition 3.1.1. *Bisimulation de place stricte*
Soit N un réseau et $B \subseteq P \times P$ une relation réflexive entre ses places. Nous dirons que B est une bisimulation de place (stricte) *ssi $\overline{B}$ est une bisimulation classique.*

Une première conséquence est que $\overline{B} \subseteq \underline{\leftrightarrow}$.

Remarque sur la réflexivité : il est essentiel de savoir, lorsqu'on réduit, si le résultat est «aussi petit que possible». Dans notre cas, cela se ramène à l'existence d'une plus grande bisimulation de places, i.e. au fait que l'union de deux bisimulations de places est inclue dans une bisimulation de places. Nous allons montrer que la réflexivité est nécessaire et suffisante :

nécessaire : Soient B et B', deux relations sur les places d'un réseau N, telles que $\overline{B}$ et $\overline{B'}$ soient des bisimulations classiques. Si nous n'exigeons pas la réflexivité, nous n'avons pas nécessairement une relation qui inclue B et B' et qui soit une bisimulation de place. La figure 3.1 donne un exemple.

Sur cette figure, nous avons B dessinée en pointillés et nous prenons $B' = Id_P$. Il n'existe pas de relation contenant B et B' qui soit une bisimulation de place.

En fait, nous identifions par B des places non-équivalentes : nous ne pouvons remplacer un jeton de p_1 par un jeton dans p_4 que si nous remplaçons dans le même temps un jeton de p_2 par un de p_3.

suffisante : nous avons la proposition suivante :

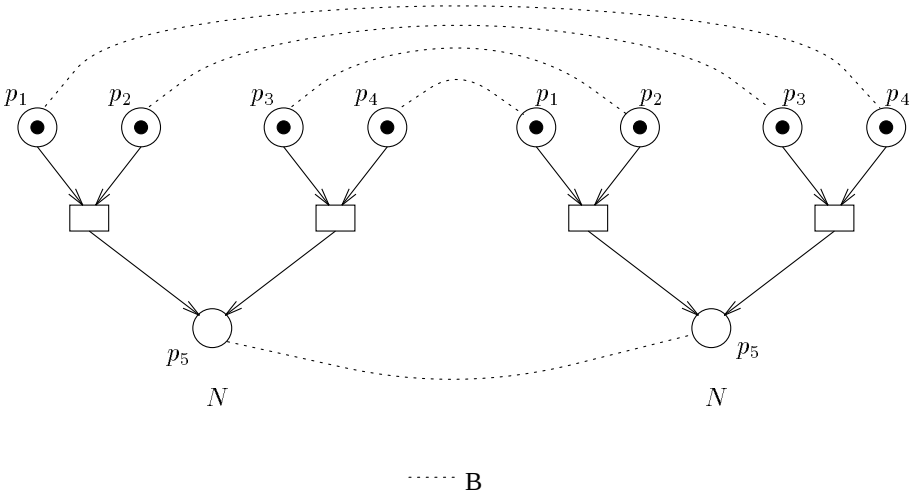
Proposition 3.1.2. *[ABS91a]*
Si B et B' sont deux bisimulations de places, alors la fermeture symétrique et transitive de $B \cup B'$ est une bisimulation de place sur N.

La preuve de cette proposition repose sur le lemme suivant :

Lemme 3.1.3. *[ABS91a]*
Si B_1 et B_2 sont deux relations réflexives sur P, alors :

$$\overline{(B_1 \cup B_2)^*} = (\overline{B_1} \cup \overline{B_2})^*$$

Figure 3.1 : Pourquoi la réflexivité ?



Preuve. La direction "$\supseteq$" est évidente. Nous voyons la preuve de l'autre inclusion.

Notons B pour $(B_1 \cup B_2)^*$. Si $M_1 \overline{B} M_2$, nous savons que M_1 est de la forme $\{p_1, \ldots, p_n\}$ et M_2 de la forme $\{q_1, \ldots, q_n\}$, avec $p_i B q_i$ pour $i = 1, \ldots, n$.

Étant donnée la définition de B, nous savons que si $p_i B q_i$, il existe une séquence $\sigma_i = (p_i^0, p_i^1, \ldots, p_i^{k_i})$ telle que $p_i = p_i^0$, $q_i = p_i^{k_i}$ et $p_i^{j-1} R_i^j p_i^j$ pour $j = 1, \ldots, k_i$, et où $R_i^j \in \{B_1, B_2\}$. Les séquences σ_i n'ont pas forcément toutes la même longueur, mais il est possible de les rendre toutes d'une longueur k commune en insérant des séquences $p_i^j R_i^j p_i^j$, les R_i^j étant des relations réflexives.

Il est possible de la même façon de rendre tous les R_i^j ($i = 1, \ldots, n$, j fixé) égaux à un même R^j, en allongeant éventuellement les séquences à une longueur k'. Si nous notons $M^0, \ldots, M^{k'}$ les marquages définis par $M^j = \{p_1^j, \ldots, p_n^j\}$, alors $M^0 = M_1$, $M^{k'} = M_2$ et $M^{j-1} \overline{R^j} M^j$ pour $j = 1, \ldots, k'$.

Nous en déduisons que $M_1 (\overline{R^{k'}} \circ \cdots \circ \overline{R^1}) M_2$, et donc que $M_1 (\overline{B_1} \cup \overline{B_2})^* M_2$, chaque R^j étant B_1 ou B_2.

□

La preuve de la proposition 3.1.2 se déroule alors comme suit :

Preuve. Supposons que $M_1 \overline{B} M_2$ (où $B = B_1 \cup B_2$). Le lemme 3.1.3 dit que $M_1 \overline{R} M_2$ où R est de la forme $R^k \circ \cdots \circ R^1$ (le choix des R^i dépendant de M_1 et M_2). R est une bisimulation puisqu'elle est la composée de B_i. Donc tout $M_1 \xrightarrow{t_1} M_1'$ peut être imité par un $M_2 \xrightarrow{t_2} M_2'$, tel que $M_1' \overline{R} M_2'$ et $l(t_1) = l(t_2)$. Comme $R \subseteq B$, nous en déduisons $M_1' \overline{B} M_2'$. Un raisonnement similaire peut être effectué afin d'imiter un pas $M_2 \xrightarrow{t_2} M_2'$.

Donc B est une bisimulation de places stricte, et une équivalence par construction.

□

La proposition 3.1.2 se généralise de deux à un nombre quelconque de bisimulations de places.

◇

Nous allons maintenant nous intéresser à l'existence et au calcul d'une plus grande bisimulation de places stricte.

3.2 Plus grande bisimulation de places stricte

Nous avons les propriétés suivantes :

Proposition 3.2.1. *Étant donné un réseau $N = (P, T, W, l)$ et B, B' des BdPS, nous avons :*

1. *Id_P, B^{-1}, $B \circ B'$, $\widetilde{B}$ sont des BdPS sur N ;*
2. *$B(N) \stackrel{\text{déf}}{=} \bigcup\{B \ : \ B \text{ BdPS sur } N\}$ est la plus grande bisimulation de place stricte sur N.*
3. *$B(N)$ est une relation d'équivalence sur P ;*
4. *$B(N) \subseteq \sim$ et parfois $B(N) \neq \sim$.*

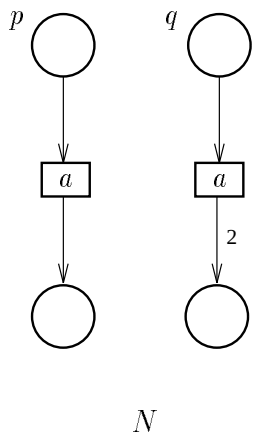
Preuve.

- 1 est une simple conséquence de la définition et de la proposition 3.1.2 puisque $\widetilde{B}$ peut être vue comme la fermeture symétrique et transitive de $B \cup B$.
- 2 vient de $B(N) = \widetilde{B(N)}$ et de la proposition 3.1.2 appliquée à l'union (qui est finie puisque le nombre de relations d'équivalences sur les places l'est).
- 3 découle de :

$$\begin{array}{rcll} pBq & \Rightarrow & \forall M,\ M + p\,\overline{B}\,M + q & (Id_P \subseteq B) \\ & \Rightarrow & \forall M,\ M + p \underline{\leftrightarrow} M + q & (\overline{B} \subseteq \underline{\leftrightarrow}) \\ & \Rightarrow & p \sim q & \end{array}$$

- la figure 3.2 prouve que l'inclusion est parfois stricte. En effet, nous avons $\forall M \in \mathcal{M}_N,\ M + p \underline{\leftrightarrow} M + q$ et donc $p \sim q$. De même, la bisimulation de place réclamant que les marquages atteints aient la même taille, il est facile de voir que $(p, q) \notin B(N)$. C'est un des points essentiels séparant $B(N)$ de $\sim$.

□



N

$p \sim q$
$(p,q) \notin B(N)$

Figure 3.2 : $B(N)$ est plus forte que $\sim$.

3.3 Réduire grâce à $B(N)$

Le théorème du quotient correct (théo. 2.2.8, p. 59) a pour corollaire immédiat :

Corollaire 3.3.1. *Soit N un réseau. Alors pour tout marquage $M \in \mathcal{M}_N$ nous avons :*

$$(N, M) \underline{\leftrightarrow} (N/_{B(N)}, M/_{B(N)})$$

Nous pouvons maintenant réduire le réseau en «fusionnant» les places bisimilaires puis en éliminant les transitions *redondantes*. Formellement :

Définition 3.3.2. *Soit $N = (P, T, W, l)$ un réseau et $t, t' \in T$:*

$$t \sqsubseteq t' \overset{\text{déf}}{\iff} ({}^\bullet t' \setminus {}^\bullet t = t'^\bullet \setminus t^\bullet) \wedge (l(t) = l(t'))$$

Cela impose entre autre que ${}^\bullet t \subseteq {}^\bullet t'$ et $t^\bullet \subseteq t'^\bullet$. Intuitivement, t peut faire la même chose que t', mais en «bouclant moins».

Nous avons les propriétés suivantes :

Proposition 3.3.3. *Soit $N = (P, T, W, l)$ un réseau.*

1. $(t \sqsubseteq t') \Rightarrow (\forall M \xrightarrow{t'} M' \ : \ M \xrightarrow{t} M')$;
2. $\sqsubseteq$ *est un pré-ordre (i.e. réflexif, transitif) partiel sur T ;*

Preuve. La propriété (1) découle directement des définitions.

Pour (2), $\sqsubseteq$ est réflexif et partiel par construction. La transitivité s'obtient facilement :

$$\begin{array}{ll} & (t \sqsubseteq t') \wedge (t' \sqsubseteq t'') \\ \Rightarrow & \left\{ \begin{array}{l} {}^\bullet t' \setminus {}^\bullet t = t'^\bullet \setminus t^\bullet \\ {}^\bullet t'' \setminus {}^\bullet t' = t''^\bullet \setminus t'^\bullet \\ {}^\bullet t \subseteq {}^\bullet t' \subseteq {}^\bullet t'' \end{array} \right. \qquad \text{et } t^\bullet \subseteq t'^\bullet \subseteq t''^\bullet \\ \Rightarrow & {}^\bullet t'' \setminus {}^\bullet t = t''^\bullet \setminus t^\bullet \\ \Rightarrow & t \sqsubseteq t'' \end{array}$$

□

Nous définissons alors comment réduire le nombre de transitions :

Définition 3.3.4. *Transition redondante, élagage*

Étant donné un réseau $N = (P, T, W, l)$, nous dirons qu'une transition $t' \in T$ est redondante (dans T) *si il existe $t \in T$ telle que $t \sqsubseteq t'$.*

Nous dirons que T est élagué *s'il ne contient pas de transition redondante.*

Pour $N = (P, T, W, l)$, nous notons $élag(N) \stackrel{\text{déf}}{=} (P, T_{élag}, W/_{T_{élag}}, l/_{T_{élag}})$ un réseau tel que :

- $T_{élag} \subseteq T$;
- $\forall t \in T,\ \exists t' \in T_{élag}\ :\ t' \sqsubseteq t$;
- $T_{élag}$ *est élagué.*

Remarques :

1. Nous pouvons aussi définir $\sqsubseteq$ par la relation de flot :

$$t \sqsubseteq t' \stackrel{déf}{\Longleftrightarrow} (\stackrel{t}{\rightarrow} \subseteq \stackrel{t'}{\rightarrow}) \wedge (l(t) = l(t'))$$

2. Cette relation s'étend naturellement aux séquences et pas parallèles :

$$\begin{array}{rcl} \sigma \sqsubseteq \sigma' & \stackrel{déf}{\Longleftrightarrow} & ({}^\bullet\sigma' \setminus {}^\bullet\sigma = \sigma'^\bullet \setminus \sigma^\bullet) \wedge (l(\sigma) = l(\sigma')) \\ \mu \sqsubseteq \mu' & \stackrel{déf}{\Longleftrightarrow} & ({}^\bullet\mu' \setminus {}^\bullet\mu = \mu'^\bullet \setminus \mu^\bullet) \wedge (l(\mu) = l(\mu')) \\ \sigma \sqsubseteq \mu & \stackrel{déf}{\Longleftrightarrow} & \exists \sigma' \in seq(\mu)\ :\ \sigma \sqsubseteq \sigma' \\ \mu \sqsubseteq \sigma' & \stackrel{déf}{\Longleftrightarrow} & \forall \sigma \in seq(\mu)\ :\ \sigma \sqsubseteq \sigma' \end{array}$$

◇

La propriété suivante découle de la proposition 3.3.3-1 :

Proposition 3.3.5. *Soit $N = (P, T, W, l)$ un réseau :*

$$\forall M \in \mathcal{M}_N,\ Id\ :\ (N, M) \underline{\leftrightarrow} (élag(N), M)$$

Soit N le réseau à réduire, nous définissons le réseau réduit par :

$$N_r \stackrel{déf}{=} \text{élag}(N/_{B(N)})$$

et pour un système $\Sigma = (N, M)$ nous avons :

$$\Sigma_r \stackrel{déf}{=} (N_r, M/_{B(N)})$$

Remarque : il est facile de voir que la réduction est «maximale», i.e. :

$$(N_r)_r \equiv N_r$$

(où $\equiv$ est l'isomorphisme des graphes).

◇

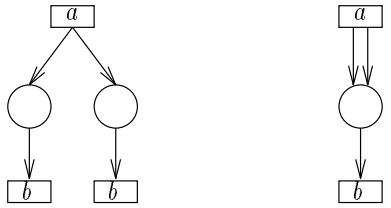
Théorème 3.3.6. *Pour $\Sigma = (N, M_0)$ un système, nous avons :*

$$\mathcal{R}(\Sigma)/_{B(N)} = \mathcal{R}(\Sigma/_{B(N)})$$

La preuve se déduit facilement de la définition de $\Sigma/_{B(N)}$ et du lemme 2.2.6, p. 58.

Remarques :

1. Le théorème 3.3.6 montre que nous conservons un grand nombre de propriétés de $\mathcal{R}(\Sigma)$, en particulier la semi-linéarité (utilisée fréquemment pour des problèmes de décidabilité, voir [EN94]).
2. La fusion de place n'est pas compatible avec la restriction à des réseaux sans égo-concurrence. Nous verrons que cette restriction est souvent nécessaire pour avoir un raffinage correct (vis-à-vis de relations d'équivalence).
3. On pourrait croire, en se restreignant aux réseaux saufs, que la définition de forme quotient fait «inutilement» la somme des marques (et rend le réseau non-sauf). On aimerait pouvoir combiner réduction avec l'opération de diviser tous les poids entrant/sortant dans une place donnée par le même facteur. Très informellement, cela fonctionne sauf dans le cas générique ci-dessous, où deux processus asynchrones sont initiés par la même transition :



N N_r

De fait, les réseaux saufs expriment le parallélisme uniquement dans sa dimension structurelle (i.e. deux transitions concurrentes sont toujours deux nœuds distincts). De ce point de vue, la bisimulation de places stricte fait un repliage passant du parallélisme structurel à celui dynamique.

4. Il est à noter que la bisimulation de places stricte permet de dériver une notion de *quotient canonique* au sens où il offre un représentant unique (modulo isomorphisme) d'une classe de systèmes bisimilaires de places. Ce ne sera pas toujours le cas.

   Étant donné un système (N, M), avec $N = (P, T, W, l)$, nous dirons qu'une place $p \in P$ est statiquement atteignable ssi il existe $p_0 \in M$ telle que $p_0 W^* p$, i.e. qu'il existe un chemin orienté d'une place marquée initialement jusqu'à p.

   Nous notons $\|(N, M)\|$ le système obtenu en éliminant toutes les places non atteignables (statiquement). Nous pouvons définir une relation d'équivalence, notée $\sim_s$, telle que :

   Définition 3.3.7.

   $$(N, M) \sim_s (N', M') \stackrel{\text{déf}}{\Longleftrightarrow} \|(N_r, M_r)\| \equiv \|(N'_r, M'_r)\|$$

   où $\equiv$ est l'isomorphisme entre les réseaux étendu naturellement aux marquages.

   $\diamond$

La BdPS vérifie deux des objectifs que nous nous étions fixés : quotienter par fusion de places en restant bisimilaires. Nous allons maintenant étudier le calcul de $B(N)$ pour un réseau donné.

3.4 La propriété de transfert local

Tester si une relation B est une bisimulation de place est un problème à première vue complexe. En effet, il s'agit de vérifier que $\overline{B}$ est une bisimulation.

Une première amélioration est possible grâce à l'additivité de la BdPS (prop. 2.2.3, p. 56). Informellement, nous pouvons nous restreindre à l'imitation de pas ${}^\bullet t_1 \stackrel{t_1}{\rightarrow} t_1{}^\bullet$ par des pas ${}^\bullet t_2 \stackrel{t_2}{\rightarrow} t_2{}^\bullet$ où ${}^\bullet t_1 \overline{B}{}^\bullet t_2$, $t_1{}^\bullet \overline{B} t_2{}^\bullet$ et $l(t_1) = l(t_2)$. C'est la démarche adoptée par [ABS91a].

Nous allons donner ici un critère nécessaire et suffisant pour caractériser les BdPS et obtenir des algorithmes simples.

Définition 3.4.1. *La propriété de transfert local*

Étant donné un réseau $N = (P, T, W, l)$, nous dirons qu'une relation $R \subseteq P \times P$ a la propriété de transfert local *ssi :*

Pour tout pas ${}^\bullet t \stackrel{t}{\rightarrow} t^\bullet$ et tous $p, q \in P$ tels que $p \in {}^\circ t$ et pRq il existe $t' \in T$ telle que, ${}^\bullet t - p + q \stackrel{t'}{\rightarrow} M' \overline{R} t^\bullet$ et $l(t) = l(t')$

ce qui peut s'exprimer plus simplement par le diagramme suivant :

$$\begin{array}{ccc} {}^\bullet t & \overline{R} & {}^\bullet t - p + q \\ {\scriptstyle \forall t:a}\downarrow & & \downarrow{\scriptstyle \exists t':a} \\ t^\bullet & \overline{R} & M \end{array}$$

La propriété de transfert local est donc la propriété de transfert pour $\overline{R}$ (voir déf. 2.1.1, p. 53) restreinte à l'imitation de pas ${}^\bullet t \stackrel{t}{\to} t^\bullet$ à partir de marquages ne différant que d'une place. Il est évident que dans un réseau fini la vérification de cette propriété peut se faire en un nombre fini d'étapes.

Nous avons le théorème fondamental suivant :

Théorème 3.4.2. *Étant donnés un réseau $N = (P, T, W, l)$ et $B \subseteq P \times P$ une relation réflexive et symétrique vérifiant la propriété de transfert local, B^* (sa fermeture transitive) est une bisimulation de place stricte.*

Preuve. Il nous suffit de prouver que $\overline{B}$ est une bisimulation classique, c'est-à-dire qu'elle vérifie la propriété de transfert. $\overline{B}$ étant symétrique, nous n'avons à prouver qu'un sens de la propriété de transfert.

Considérons deux marquages $M_1 \overline{B^*} M_2$ et un pas $M_1 \stackrel{t_1}{\to} M_1'$. Plusieurs cas se présentent :

1. si $M_1 \overline{B} M_2$:

   (a) supposons que $M_2 = M_1 - p + q$ avec pBq. Si $M_1 = {}^\bullet t_1$, le transfert local nous permet de dire qu'il existe un pas $M_2 \stackrel{t_2}{\to} M_2'$, avec $M_1' \overline{B} M_2'$ avec $l(t_1) = l(t_2)$. Sinon, M_1 est de la forme ${}^\bullet t_1 + M_1''$. Si $p \in M_1''$, alors $M_2 \stackrel{t_1}{\to} M_1' - p + q$ imite $M_1 \stackrel{t_1}{\to} M_1'$. Si $p \in {}^\bullet t_1$, alors par la propriété de transfert il existe un pas ${}^\bullet t_1 - p + q \stackrel{t_2}{\to} M'$, avec $l(t_1) = l(t_2)$. On peut donc imiter $M_1 \stackrel{t_1}{\to} M_1'$ par $M_2 \stackrel{t_2}{\to} M_2'$ avec $M_2' \overline{B} M_1'$, ou‘ $M_2' = M_1'' + M'$ et $M_1' = M_1'' + t_1{}^\bullet$,

   (b) sinon, il existe un ensemble de marquages tels que

   $$M_1 = M^0 \overline{B} M^1 \overline{B} \ldots \overline{B} M^k = M_2,$$

   avec $M^i = M^{i-1} - p_i + q_i$ et $p_i B q_i$, $i = 1, \ldots, k$. Dans ce cas, $M^0 \stackrel{t_1}{\to} M_1'$ peut être imité par un $M^1 \stackrel{u_1}{\to} M'^1$ car on retombe dans le cas précédent. L'imitation peut se propager jusqu'à $M_2 = M^k \stackrel{u_k}{\to} M'^k$. L'imitation se fait avec $t_2 = u_k$, et $M_2' = M'^k$. Comme $M_1' \overline{B} M'^1 \overline{B} \ldots \overline{B} M'^k$, nous avons $M_1' \overline{B^*} M_2'$, soit $M_1' \overline{B^*} M_2'$ puisque B est réflexive (lemme 3.1.3),

2. si $M_1 \overline{B^*} M_2$, il est possible de décomposer en une séquence $M_1 = M^1 \overline{B} M^2 \ldots \overline{B} M^k = M_2$, et d'utiliser le cas précédent pour construire une imitation $M_2 \stackrel{u}{\to} M_2'$.

□

Il est évidemment facile de voir qu'une BdPS vérifie la propriété de transfert local. Nous en dérivons le corollaire suivant :

Corollaire 3.4.3. *Caractérisation des BdPS*
Étant donnés un réseau $N = (P, T, W, l)$ et $B \subseteq P \times P$ une équivalence sur les places du réseau. Alors B est une bisimulation de places stricte ssi elle vérifie la propriété de transfert local.

Comme la propriété de transfert local est vérifiable en un temps fini (pour un réseau fini) et que le nombre de relation d'équivalence sur les places est fini, nous avons :

Proposition 3.4.4. *$B(N)$ est calculable*

Le transfert local est donc une propriété vérifiable en un temps fini et caractéristique des BdPS. Nous avons donc tout ce qui est nécessaire pour calculer $B(N)$ et N_r. Les sections suivantes traitent de ces opérations.

3.5 Calcul de $B(N)$

À la simple lecture de la propriété de transfert local et grâce au corollaire 3.4.3, nous posons l'algorithme suivant :

Algorithme 3.5.1. *Calcul de $B(N)$*
Soit le réseau $N = (P, T, W, l)$:

1. *$B = P \times P$.*
2. *Tester si B a la propriété de transfert local, i.e. :*
   - *s'il existe $t \in T$, $p \in {}^\bullet t$, $q \in P$ avec pBq tels qu'il n'existe pas de $t' \in T$ de même étiquette, tirable à partir de ${}^\bullet t - p + q$ et atteignant un marquage M' t.q. ${}^\bullet t \, \overline{B} M'$, alors éliminer les paires (p, q) et (q, p) de B et recommencer l'étape 2.*
   - *sinon, B a la propriété de transfert local.*
3. *B est la plus grande bisimulation de place, i.e. $B = B(N)$.*

Remarque : ce type d'algorithme, raffinant une relation et cherchant un point fixe, est classique. La même méthode est utilisée pour le calcul de $\underline{\leftrightarrow}$ dans les systèmes d'états finis (voir [BS87]).

◇

Nous étudions maintenant sa terminaison et sa correction.

3.5.1 Terminaison de l'algorithme

Pour tout réseau N, la boucle 1 termine puisqu'elle ne contient que des parcours d'ensembles finis.

L'algorithme termine, car il y a au plus $n_P^2 - n_P$ paires à enlever. Or, après chaque parcours de la boucle sur $t \in T$:

- soit on a éliminé au moins deux paires : (p,q) et (q,p) qui étaient dans B (car nous avons la condition $B(p,q)$), et B a diminué ;
- soit on n'élimine aucune paire, B est stable, le point fixe est atteint et nous avons terminé.

3.5.2 Correction de l'algorithme

De par sa construction, l'algorithme nous assure qu'à l'issue B a la propriété de transfert faible.

Le point important est donc de vérifier que B est la bisimulation de place cherchée, c'est-à-dire la plus grande. $B(N)$ est incluse dans B au départ de l'algorithme puisque $B = P \times P$. De plus, si l'on enlève (p,q) et (q,p) c'est parce que p et q ne sont pas bisimilaires car $B(N) \subseteq B$. Donc, après chaque boucle 2, nous avons $B(N) \subseteq B$ et par conséquent en fin de parcours $B = B(N)$.

3.6 Coût théorique de la réduction

Dans cette section, nous affinons le calcul du coût de la réduction.

Soit un réseau $N = (P,T,W)$, nous introduisons les notations suivantes :

Notations 3.6.1. *Paramètres d'un réseau*

Pour une partition de sommets (P et T), nous définissons les degrés entrant, sortant et global par :

$$\begin{cases} d_{T^\circ} = max_{t \in T}\{|t^\circ|\}, & \textit{le degré sortant des transitions ;} \\ d_{^\circ T} = max_{t \in T}\{|^\circ t|\}, & \textit{le degré entrant des transitions ;} \\ d_T = max_{t \in T}\{|^\circ t| + |t^\circ|\}, & \textit{le degré des transitions ;} \end{cases}$$

et similairement pour $d_{^\circ P}$, d_{P° et d_P.

Nous appellerons degré du réseau *le nombre $d = max\{d_P, d_T\}$.*

Nous notons :

$$\begin{cases} n_P = |P|, & \textit{le nombre de places ;} \\ n_T = |T|, & \textit{le nombre de transitions ;} \\ n_W = |\{(x,y) \in (P \cup T)^2 \ : \ W(x,y) > 0\}|, & \textit{le nombre d'arcs ;} \end{cases}$$

Il est facile de voir que $n_P.d_P \geq n_W \leq n_T.d_T$.

Nous appellerons taille du réseau *le nombre $n_P + n_T$.*

Théorème 3.6.1. *Le coût de la réduction est alors en :*

$$\mathcal{O}(n_T^3.\log(n_T).n_P^3.d_T^{d_T+1}.d_P)$$

soit en $\mathcal{O}(n_T^3.\log(n_T).n_P^3)$ *pour les familles de réseaux où les degrés sont bornés par une constante.*

Nous allons maintenant détailler le coût maximal théorique de chaque étape, et établir la preuve du théorème.

3.6.1 L'algorithme implémenté

Il est simplement calqué sur la recherche d'un point fixe et sur celui donné en 3.5.1, p. 84 :

fonction B (N : un réseau) $\triangleright$ une relation sur P
  B : une relation sur P
  B-stable : un booléen
  B $\triangleleft$ P $\times$ P
  B-stable $\triangleleft$ vrai
  Répéter
    Pour t_1 parcourant T
      Pour p parcourant $^\bullet t_1$
        Pour q parcourant P-$\{p\}$
          Si $((p,q) \in$ B)
            et non (est-imitable $(t_1,p,q,$B))
          Alors B $\triangleleft$ B - $\{(p,q);(q,p)\}$
            B-stable $\triangleleft$ faux
          fsi
        fpour $\{q\}$
      fpour $\{p\}$
    fpour $\{t_1\}$
  $\{$*Invariant :* $B(N) \subseteq$ B *(donc* B *réflexive) et* B *est symétrique*$\}$
  jusqu'à B-stable
  $\{B(N) \subseteq$ B *et* B *a la propriété de transfert local donc* $B(N) =$ B$\}$
  Retourne B

où la fonction est-imitable $(t_1,p,q,$B) teste s'il existe une transition t_2 telle que :

$$\begin{array}{ccc} {}^\bullet t_1 & \overline{\mathtt{B}} & {}^\bullet t_1 - p + q \\ {\scriptstyle t_1:a}\downarrow & & \downarrow{\scriptstyle t_2:a} \\ t_1{}^\bullet & \overline{\mathtt{B}} & M \end{array}$$

Cette fonction est codée comme suit :

fonction `est-imitable` $(t_1, p, q,$ `B`$)$ $\triangleright$ `un booléen`
  Pour t_2 `parcourant T-`$\{t_1\}$
    Si `l(`t_1`)=l(`t_2`)`
      et `est-tirable` $(t_2,\ {}^\bullet t_1 - p + q)$
      et `sont-reliés (B,` $t_1{}^\bullet,\ {}^\bullet t_1 - p + q - {}^\bullet t_2 + t_2{}^\bullet$`)`
    Alors retourne `vrai`
  fpour $\{t_2\}$
  Retourne `faux`

où les fonctions `est-tirable (`t`,` M`)` et `sont-reliés (B,` M`,` M'`)` testent respectivement $M \xrightarrow{t}$ et $M \overline{\mathtt{B}} M'$.

3.6.2 Calcul de la complexité

Nous allons étudier la complexité de chaque étape :

le test du lien entre deux marquages (`sont-reliés (B,M,M')`) revient à tester tous les arrangements possibles pour les supports de M et M', soit $d_T^{d_T}$ possibilités ;

En fait, nous verrons (preuve de la prop. 3.6.2) qu'il est possible de faire les comparaisons avec $\mathtt{B}^*$ qui est une relation d'équivalence. Par suite, nous pouvons le faire en temps d_T.

le test de franchissement (`est-tirable (t,M)`) revient à vérifier l'inclusion de deux multi-ensembles de taille $d_{\bullet T}$ et peut être fait en temps linéaire ;

la recherche de t_2 (`est-imitable (t,p,q,B)`) se fait donc en temps $\mathcal{O}((n_T - 1).(d_T^{d_T} + d_{\bullet T}))$;

les trois boucles imbriquées (sur t_1, p, q) donnent donc une complexité de $n_T.n_P.d_{\bullet T}.(n_T - 1).(d_T + d_{\bullet T})$.

En effet, nous pouvons considérer que les autres tests de l'algorithme sont faits en temps constant.

Dans la suite, nous appellerons *étape* le parcours complet des trois boucles imbriquées.

Il nous reste donc à évaluer le nombre maximal d'étapes nécessaires. Nous allons nous baser sur un version légèrement modifiée de l'algorithme.

Proposition 3.6.2. *On a au plus* $n_p - 1$ *étapes dans l'algorithme 3.5.1.*

Preuve. Remarquons tout d'abord qu'un cas particulier du théorème 3.4.2, p. 83 est $B = B^*$.

Dans l'algorithme 3.5.1 (noté A), nous pouvons donc faire la recherche d'une imitation en utilisant B^*. Puisque `est-imitable (`t_1, p, q`,B*)` implique `est-imitable (`t_1, p, q`,B)`, nous obtenons un algorithme (appelé A^*) a priori plus lent.

Dans A^*, si $\mathtt{B}^{*}$[1] ne change pas au cours d'une étape, $\mathtt{B}$ non plus. Alors nous avons le point fixe et $\mathtt{B}^* = B(N)$. Or le nombre maximal de classes dans $\mathtt{B}^*$ est n_P, donc nous effectuerons au maximum $n_P - 1$ étapes (dans A^*).

Comme A converge au moins aussi rapidement, nous avons au maximum $n_P - 1$ étapes (dans A).

□

En cumulant tous ces résultats, nous avons un coût en $\mathcal{O}(n_T^2.n_P^2.d_T^2)$ pour le calcul de $B(N)$.

3.6.3 Fusion des places bisimilaires

Il s'agit de re-diriger les arcs, soit au plus n_W opérations si $B(N) = P \times P$, donc en $\mathcal{O}(n_P.d_P)$.

3.6.4 Élagage

La relation $\sqsubseteq$ est un ordre partiel sur T. Il est facile de voir que dans le cas où aucune transition n'est redondante, le coût est en $\mathcal{O}(n_T.\log(n_T))$.

En effet, $\sqsubseteq$ revient à trier les transitions selon trois possibilités (plus petite, plus grande, non liée), avec un ordre total. Ce type de tri est en $\mathcal{O}(n.\log(n))$, pour n éléments à trier.

3.7 Un exemple

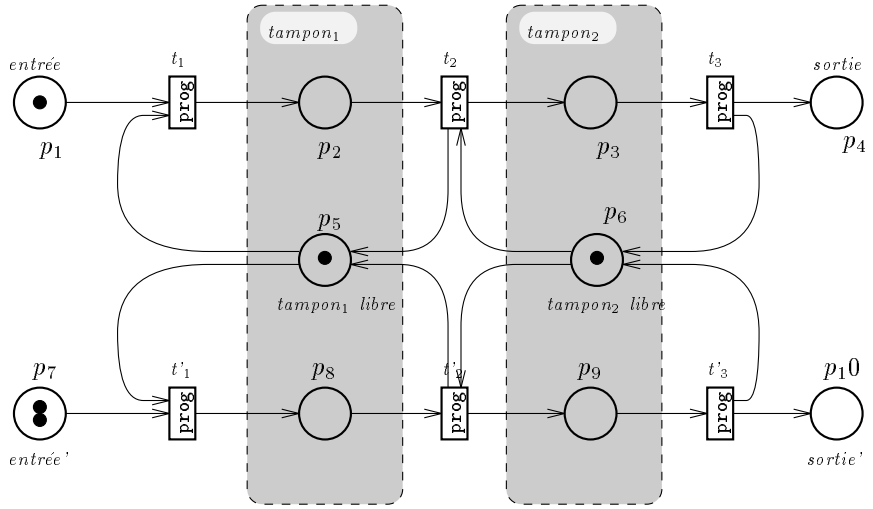
Nous reprenons le réseau décrivant la file FIFO et le doublons afin de simuler deux canaux. Nous avons le réseau de la figure 3.3.

Pour cet exemple, le prototype développé donne le résultat suivant :

```
B(fifo-2)

Strict place bisimulation results :
-----------------------------------
        { p10, p4 }
        { p9, p3 }
        { p8, p2 }
        { p7, p1 }
        { p6 }
        { p5 }
```

[1] Ce qui revient en fait à n'enlever les places qu'à la fin de chaque étape.

Figure 3.3 : Une file FIFO à deux tampons et deux canaux



(N, M)

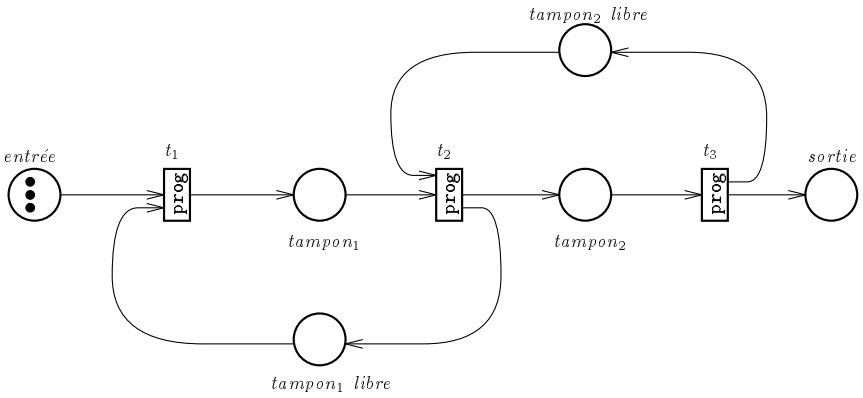
```
Times :
-------
                 Pract   Theo
Nb of loops      3       10
Nb of t2         288     3600

2 seconds, 1 to compute B(N).

Reduction :
-----------
        |P|     |T|     degrees of P,T
N       10      6       4, 4
Nr      6       3       2, 4

[Nr saved to red-fifo-2.net]
```

Nous avons donc, après réduction le système de la figure 3.4.

3.8 Bisimulation de places stricte et sémantiques d'ordre partiel

Nous nous contentons de redonner ici quelques résultats. Le lecteur intéressé pourra se reporter au chapitre 6 de [Aut93]. En fait, nous allons

Figure 3.4 : Le réseau réduit.



(N_r, M_r)

voir que notre méthode de réduction conserve une partie du parallélisme (quitte à transformer un parallélisme structurel en parallélisme dynamique lors du passage de N à N_r). Par contre, il est facile de voir que la fusion des places va modifier les relations de causalité. C'est pour cette raison que la bisimulation pomset ne nous permettra pas de quotienter correctement.

Nous donnons tout d'abord une idée informelle des différentes variantes de la bisimulation :

- bisimulation *concurrente* [NT84, GV87] : la possibilité de tirer simultanément un ensemble de transitions est respectée,
- bisimulation *partial word* [Vog90] : le tir d'un pomset-step ρ est imité par le tir d'un pomset-step ρ' qui peut être plus parallèle que ρ dans la mesure où il contient les mêmes dépendances entre transitions que ρ sauf certaines qui ont pu être omises,
- bisimulation *pomset* [BC87, GV87] : le tir d'un pomset est imité par le tir d'un pomset identique. Les dépendances entre transitions sont strictement respectées,
- bisimulation de *processus* [AS92] : non seulement les dépendances entre transitions sont respectées, mais aussi le mouvement des jetons permettant de tirer ces transitions.

3.8.1 Quelques bisimulations basées sur les ordres partiels

Nous donnons maintenant les définitions formelles de ces bisimulations.

Définition 3.8.1. *$R \subseteq \mathcal{M}_N \times \mathcal{M}_{N'}$ est une $*$ – bisimulation entre deux systèmes $\Sigma = (N, M_0)$ et $\Sigma' = (N', M'_0)$ (pour $* \in \{i, c, pw, pom, pr\}$) ssi $M_0 R M'_0$ et si pour tout $M_1 R M'_1$*

- *(bisimulation* concurrente*) $* = c$ et tout pas $M_1 \stackrel{\mu}{\Longrightarrow} M_2$, $\mu \in \mathcal{M}(T)$ dans N peut être imité par $M'_1 \stackrel{\mu'}{\Longrightarrow} M'_2$ dans N', avec $l_N(\mu) = l_{N'}(\mu')$ et $M_2 R M'_2$, et réciproquement,*

- *(bisimulation* partial word*) $* = pw$ et tout pas pomset $M_1 \stackrel{\rho}{\longrightarrow\!\!\!\!\!\rightarrow} M_2$ dans N peut être imité par $M'_1 \stackrel{\rho'}{\longrightarrow\!\!\!\!\!\rightarrow} M'_2$ dans N', avec $\rho' \preceq \rho$ et $M_2 R M'_2$, et réciproquement,*

- *(bisimulation* pomset*) $* = pom$ et tout pas pomset $M_1 \stackrel{\rho}{\longrightarrow\!\!\!\!\!\rightarrow} M_2$ dans N peut être imité par $M'_1 \stackrel{\rho}{\longrightarrow\!\!\!\!\!\rightarrow} M'_2$ dans N', avec $M_2 R M'_2$, et réciproquement,*

- *(bisimulation de* processus*) $* = pr$ et tout processus $M_1 \stackrel{C,f}{\rightarrow} M_2$ dans N peut être imité par $M'_1 \stackrel{C,f'}{\rightarrow} M'_2$ dans N', avec $M_2 R M'_2$, et réciproquement.*

Nous pouvons établir la hiérarchie suivante entre les bisimulations que nous venons de définir :

Lemme 3.8.2.

$$N \underline{\leftrightarrow}_{pr} N' \Rightarrow N \underline{\leftrightarrow}_{pom} N' \Rightarrow N \underline{\leftrightarrow}_{pw} N' \Rightarrow N \underline{\leftrightarrow}_{c} N' \Rightarrow N \underline{\leftrightarrow} N'$$

Aucune de ces implications ne peut être transformée en une équivalence [GV87, Vog90, AS92].

Pour toutes ces bisimulations nous obtenons des versions adaptées de la bisimulation de places :

Définition 3.8.3. *Pour $* \in \{c, pw, pom, pr\}$, nous dirons qu'une relation $B_* \subseteq P \times P$ est une $*$-bisimulation de places ssi $\overline{B_*}$ est une $*$-bisimulation.*

Nous avons l'analogue du point 2 de la proposition 3.2.1 donnant l'existence d'une plus grande $*$-bisimulation de place sur un réseau, notée $B_*(N)$, qui soit une équivalence[2]. Nous pouvons donc définir de manière analogue à 2.2.5, p. 57 le système quotient $\Sigma/_{B_*(N)}$. Le paragraphe suivant explicite les liens entre les différentes bisimulations de places.

[2]En effet, ce théorème repose sur le lemme 3.1.3, p. 76 qui n'exige que la réflexivité de la relation.

3.8.2 Classification des bisimulations

D'après la définition 3.8.3 et le lemme 3.8.2, il est clair que :

$$B_{pr}(N) \subseteq B_{pom}(N) \subseteq B_{pw} \subseteq B_c(N) \subseteq B(N)$$

Mais de plus, nous avons le théorème suivant :

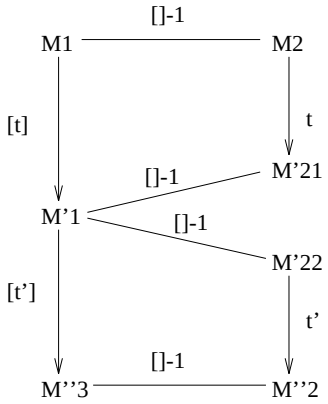
Théorème 3.8.4. *Soit N un réseau de Petri, alors :*

$$B_{pw}(N) = B_c(N) = B(N)$$

Ce théorème découle directement du théorème 6.6 de [Aut93]. La preuve étant longue et technique, nous invitons le lecteur intéressé à se reporter au manuscrit précité.

Nous terminerons cette section en notant que $B_{pom}(N)$ ne nous permet pas de quotienter de manière satisfaisante. En effet, nous ne pouvons donner l'analogue du théorème du quotient correct (théo. 2.2.8, p. 59).

Remarque sur le théorème 2.2.8 : en fait dans les autres cas, ce théorème ne s'étend *que* parce que nous imitons *une* transition. Typiquement, le cas pomset ne marche pas car :



mais on ne peut garantir ${}^\bullet t' \cap t^\bullet \neq \emptyset$ ce qui est contraire à la pomset (voir à ce sujet l'exemple 3.5).

Explication : la fusion perturbe les causalités.

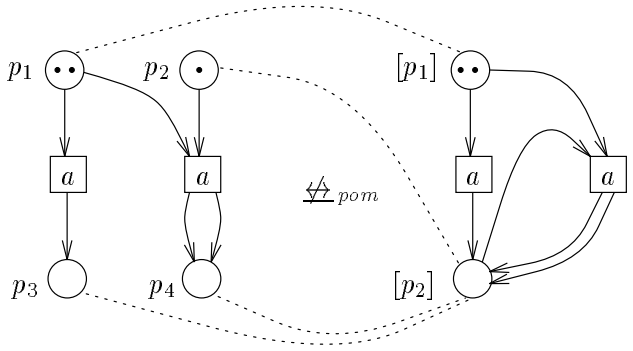
Pourquoi cela fonctionne-t-il avec $B_{pr}(N)$? Dans la *pr*-bisimulation, il faut aussi respecter la façon dont les jetons se déplacent, i.e. on a des structures causales isomorphes.

◇

Sur la figure 3.5, en disposant deux jetons dans p_1 et un dans p_2, le seul pomset tirable est «a et a en parallèle». Par contre, avec deux jetons dans $[p_1]$ et un dans $[p_2]$, il est possible de tirer le pomset «a suivi de a» dans N_2.

La BdPS conserve donc une partie du parallélisme, mais pas les causalités.

Figure 3.5 : On ne peut quotienter avec $B_{pom}(N)$.



N_1 $\qquad\qquad$ $N_2 = N_1/_{B_{pom}(N_1)}$

3.9 Des bisimulations de places en général

Nous pouvons adapter la notion de *bisimulation de place* aux variantes de la bisimulation :

> B_* est une **-bisimulation de places* ssi $\overline{B_*}$ est une *-bisimulation entre les marquages.

Dès lors qu'une *-bisimulation entre les marquages est définie par une «variante» de la propriété de transfert, nous avons :

Théorème 3.9.1. *Lorsqu'elles existent, les plus grandes $*$-bisimulations de places sont calculables.*

Avant d'aborder la preuve du théorème 3.9.1, nous allons préciser quels types de modifications nous nous autorisons apporter à la propriété de transfert :

- le cas de la pomset montre bien qu'il est difficile d' assouplir les conditions $\forall M_1 \overline{B} M_2$ et $\forall M_1 \stackrel{t_1}{\rightarrow} M_1'$;
- nous pouvons par contre modifier notre manière d'imiter :
  - en autorisant des séquences avec des marquages intermédiaires, donc remplacer $\exists t_2 \in T \ : \ M_2 \stackrel{t_2}{\rightarrow} M_2'$ par $\exists \sigma_1, \ldots, \sigma_n \ : \ M_2 = M^0 \stackrel{\sigma_1}{\rightarrow} M^1 \cdots M^{n-1} \stackrel{\sigma_n}{\rightarrow} M^n = M_2'$ avec des contraintes sur n ou sur les σ_i. Nous ne pouvons pas mettre de contrainte dépendant des marquages, puisque les bisimulation de places en font abstraction, mais nous pouvons, p.ex., demander à ce que les séquences aient une certaine étiquette.

– en demandant que certains marquages soient reliés.

Ces possibilités couvrent la gamme des bisimulations prenant en compte les actions internes que nous étudierons dans la partie suivante (τ, bisimulation de (semi-)branchement, η et Δ).

Définition 3.9.2. *Transfert semi-local*

Étant donnés un réseau $N = (P, T, W, l)$, nous dirons qu'une relation d'équivalence $B \subseteq P \times P$ vérifie le transfert semi-local *ssi :*

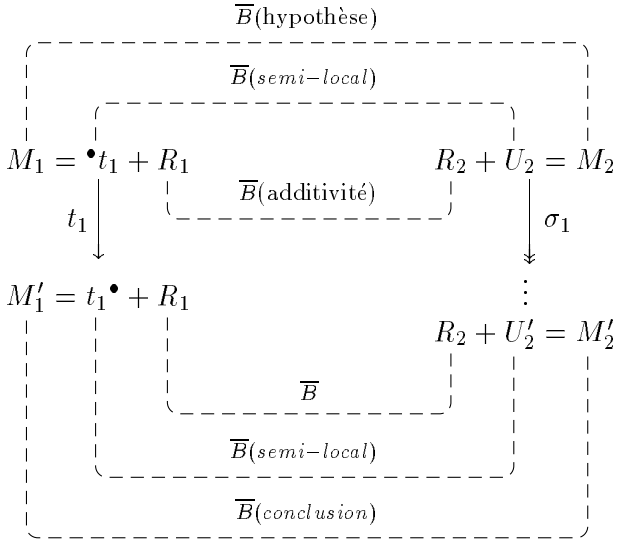
$$\begin{array}{ccc} {}^\bullet t_1 & \overline{B} & M \\ {\scriptstyle \forall t_1}\downarrow & & \downarrow{\scriptstyle \exists t_2 : l(t_1)} \\ t_1{}^\bullet & \overline{B} & M' \end{array}$$

Le théorème 3.9.1 devient alors une simple conséquence du suivant :

Théorème 3.9.3. *Étant donnés un réseau $N = (P, T, W, l)$, nous dirons qu'une relation d'équivalence $B \subseteq P \times P$ vérifie la propriété de transfert ssi elle vérifie celle du transfert semi-local.*

Preuve. Le théorème 3.9.3 découle de l'additivité (prop. 2.2.3, p. 56) et du fait que l'atteignabilité est décidable ([May84, Kos82]). En fait, elle repose sur le fait que B est réflexive, donc que $\overline{B}$ est additive et $Id_{\mathcal{M}_N} \subseteq \overline{B}$.

En effet, soit B une relation sur les places, l'additivité nous permet de ramener la propriété de transfert à l'étude de marquages reliés par $\overline{B}$ aux pré- et post-conditions d'une transition (soit trivialement un nombre fini), comme le montre la figure ci-dessous :



Le nombre de relations entre les places d'un réseau (fini) étant fini, nous avons le théorème.

□

Malheureusement, il est prouvé dans [Lip76] que le problème de l'atteignabilité est «exp-space-hard» et MAYR dans [May84] (parag. 6, p. 459) ne sait pas sous quelles conditions son algorithme est non primitif-récursif (voir p.ex. [May81]).

Dans la suite, nous allons donc nous intéresser à certaines variantes de la bisimulation et donner des algorithmes efficaces pour le calcul des bisimulations de places correspondantes.

3.9.1 Bisimulations additives

Définition 3.9.4. *Bisimulation additive*

Soit $N = (P, T, W, l)$ un réseau, nous dirons que $R \subseteq \mathcal{M}_N \times \mathcal{M}_N$ est une bisimulation additive *ssi :*

$$\forall M_1, M_2, M_3 \in \mathcal{M}_N \ : \ M_1 \overline{B} M_2 \rightarrow M_1 + M_3 \overline{B} M_2 + M_3$$

Il existe une plus grande bisimulation additive que nous noterons $B_a(N)$.

Les définitions respectives nous donnent trivialement :

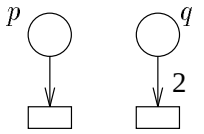
Proposition 3.9.5.

$$\overline{B(N)} \subseteq B_a(N)$$

La comparaison entre $\overline{\sim}$ et $B_a(N)$ est sans objet, puisque nous savons que $\overline{\sim}$ n'est pas toujours une bisimulation (fig. 2.2, p. 57). Cependant, lorsque c'en est une, nous avons $\overline{\sim} \subseteq B_a(N)$.

La figure 3.6 montre que ces inclusions sont parfois strictes.

Figure 3.6 : $B(N)$, $B_a(N)$ et $\sim$ sont parfois différentes.



N

$B(N) = \sim = Id_P$
$(\{p\}, \{q, q\}) \in B_a(N)$

Proposition 3.9.6. *$B_a(N)$ est calculable.*

Preuve. Il suffit de la voir comme un semi-linéaire. Comme la propriété de transfert correspond à une formule de Presburger, nous avons la semi-décidabilité par énumération. La semi-décidabilité de la non-bisimilarité nous donne la décidabilité de $B_a(N)$.

□

Malheureusement, c'est *extrêmement* coûteux, puisque la décision d'une formule de Presburger se fait en $\mathcal{O}(2^{2^{2^{pn}}})$ (voir [Opp]).

Conclusion

La bisimulation de places stricte est donc une notion simple satisfaisant aux trois critères que nous nous étions fixés :

1. permettre la réduction par fusion de places équivalentes (déf. 2.2.5, p. 57) ;
2. donner un réseau réduit qui soit bisimilaire au réseau initial (coro. 3.3.1, p. 79) ;
3. être calculable efficacement (théo. 3.6.1, p. 86).

Il est donc naturel d'étudier son comportement pour différentes variantes de la bisimulation et différents enrichissements de la structure des réseaux. Les chapitres suivants vont s'intéresser successivement aux actions invisibles et aux arcs inhibiteurs.

Notes bibliographiques

La bisimulation de place stricte est étudiée dans [ABS91b, Aut93].

Deuxième partie

Les extensions classiques des réseaux de Petri

Un des grands avantages de la bisimulation de places est sa souplesse. Nous avons considéré deux types d'extensions :

1. utiliser une autre bisimulation comme critère de correction. Cas souvent complexe, mais des modifications superficielles de l'algorithme sont possibles.

2. appliquer la bisimulation de places à des réseaux «décorés». Les remaniements de l'algorithme sont plus importants.

Dans les deux cas, le nombre de possibilités est important. Nous nous sommes restreints aux cas essentiels par le volume de littérature s'y rapportant et par leurs applications :

- abstraction : τ-bisimulation et bisimulation de (semi-)branchement. Les objectifs sont ici faciles à définir, puisque la τ-bisimulation, proposée par Milner pour CCS, ainsi que la bisimulation de branchement de van Glabbeek et Weijland, sont les adaptions de la bisimulation presque universellement utilisées dans la situation où nous nous plaçons. Nous nous sommes placé dans le cadre de la τ-bisimulation ([Mil80]) et de la bisimulation de (semi-)branchement ([GW89b, Wei89]) ;

- réseaux à arcs inhibiteurs : où la présence d'un jeton dans une place peut inhiber une transition ; la représentation d'un tel arc par $-o$ vient de la négation logique dans les cellules de Mc Culloch-Pitts ([Min67]).

- réseaux à arcs de test ([Lak93, CH93b]) : voir remarque du chapitre sur les inhibiteurs ;

- réseaux colorés ([Jen92, CH93a]), temporisés ([Zub80]) et stochastiques ([Ajm89]), voir en conclusion de partie.

De très nombreuses variantes ont été définies sur les réseaux de Petri. [BC92] présente les modèles de bases et diverses familles ; [Bes88] présente les réseaux à poids ; [DE95] étudie les réseaux à choix libres (voir déf. 7.2.1, p. 161) et [Lak94] relie les réseaux colorés et les réseaux à objets. Et il existe encore bien d'autres choses dans la littérature...

Chapitre 4

La τ-bisimulation de places

Une question importante est de savoir comment l'approche «simplification par la bisimulation de places» peut prendre en compte les actions invisibles (notées τ). Ces actions sont un outil conceptuel classique pour obtenir de façon méthodique certains comportements. Typiquement, elles apparaissent lors de la mise en parallèle de deux réseaux avec synchronisation sur un ensembles de transitions donné ; les transitions synchronisées deviennent invisibles pour un observateur extérieur.

De fait, la notation τ vient des algèbres de processus telles que CCS ([Mil80]) qui offrent un mécanisme *d'abstraction* permettant de cacher certaines actions du programme. En «gommant» ainsi certains détails du comportement, nous pouvons rendre des systèmes indistinguables (d'un point de vue externe).

Une idée «naturelle» est donc d'autoriser le système à évoluer de manière interne pour pouvoir imiter. C'est sur ce principe qu'est basée la τ-bisimulation (ou «bisimulation observationnelle») de [Mil80].

Le chapitre est organisé comme suit. Après quelques définitions supplémentaires, nous présentons la τ-bisimulation de places (ou τ-BdP). Il s'avère qu'elle a toutes les propriétés souhaitées, *sauf* l'équivalent du théorème du transfert local qui nous donnait un algorithme efficace (théo. 3.4.2).

La section 4.6 présente une approximation de la τ-bisimulation de places : la τp-bisimulation de places (ou τp-BdP). Celle-ci permet de dériver un théorème analogue à celui du transfert local et donc de calculer efficacement la plus grande τp-BdP : $B_{\tau p}(N)$.

Enfin, la section 4.9 établit le lien entre ces deux notions et montre que pour une large classe de réseaux $B_\tau(N) = B_{\tau p}(N)$.

4.1 Quelques nouvelles notations

L'étiquetage se fait maintenant dans $\mathcal{A}_\tau = \mathcal{A} \cup \{\tau\}$. Nous noterons $T_\tau \subseteq T$ l'ensemble des transitions de T étiquetées par τ.

Nous ne nous intéressons qu'à la partie *visible* des séquences, nous définissons :

Définition 4.1.1. *Soit $N = (P, T, W, l)$ un réseau étiqueté sur $\mathcal{A}_\tau$ et $\sigma, \sigma' \in T^*$, $w \in \mathcal{A}^*$, nous avons :*

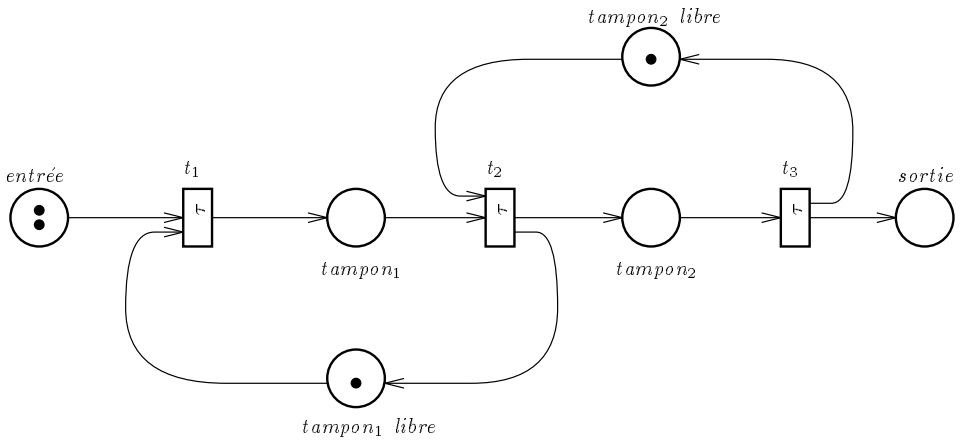
$$\begin{aligned}
\sigma/_{\mathcal{A}} &\stackrel{\text{déf}}{=} \sigma/_{\{t \in T \,:\, l(t) \in \mathcal{A}\}} \\
\sigma =_\tau \sigma' &\stackrel{\text{déf}}{\Longleftrightarrow} \sigma/_{\mathcal{A}} = \sigma'/_{\mathcal{A}} \\
l_\tau(\sigma) = w &\stackrel{\text{déf}}{\Longleftrightarrow} l(\sigma/_{\mathcal{A}}) = w \\
\sigma \sqsubseteq_\tau \sigma' &\stackrel{\text{déf}}{\Longleftrightarrow} ({}^\bullet\sigma' - {}^\bullet\sigma = \sigma'^\bullet - \sigma^\bullet) \wedge (l(\sigma) =_\tau l(\sigma')) \\
M \xrightarrow{\sigma :_\tau w}\!\!\!\!\rightarrow M' &\stackrel{\text{déf}}{\Longleftrightarrow} (M \stackrel{\sigma}{\longrightarrow}\!\!\!\!\rightarrow M') \wedge (l(\sigma) =_\tau w)
\end{aligned}$$

Nous appellerons $w = l_\tau(\sigma)$ l'étiquette modulo τ *de σ.*

Nous définissons les mêmes notations pour $\mu \in \mathcal{M}(T)$ (à l'exception de l'étiquette).

La figure 4.1 montre une manière de transformer la file en «boîte noire» en étiquetant les transitions `prog` par τ. Dans ce cas, un observateur extérieur ne se préoccupe plus du nombre de tampons ni des transferts d'information entre les tampons, mais voit simplement les données entrer et sortir.

Figure 4.1 : Un réseau de Petri avec des actions internes



N

4.2 La τ-bisimulation

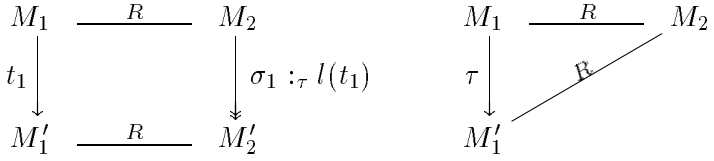
Intuitivement, deux systèmes de transitions sont τ-bisimilaires s'ils peuvent s'imiter l'un l'autre abstraction faite des transitions invisibles, i.e. avoir les mêmes comportements en intercalant autant de transitions étiquetées par τ que voulu. Formellement :

Définition 4.2.1. *Propriété de τ-transfert*

Soient $N_1 = (P_1, T_1, W_1, l_1)$ et $N_2 = (P_2, T_2, W_2, l_2)$ deux réseaux, une relation $R \subseteq \mathcal{M}(P_1) \times \mathcal{M}(P_2)$ entre les marquages vérifie la propriété de τ-transfert *ssi :*

> *pour tout $M_1 R M_2$ et tout pas $M_1 \xrightarrow{t_1} M_1'$ il existe une séquence $\sigma_2 \in T^*$ de même étiquette modulo τ avec $M_2 \xrightarrow{\sigma_2 :_\tau l(t_1)} M_2'$ et telle que $M_1' R M_2'$*

Cette définition est plus explicite sur le schéma suivant :



Nous en dérivons la définition suivante :

Définition 4.2.2. *τ-bisimulation entre les marquages*

Une relation symétrique ayant la propriété de τ-transfert est une τ-bisimulation (entre les marquages). *Les marquages reliés sont dits* τ-bisimilaires. *Nous le noterons R : $M_1 \underline{\leftrightarrow}_\tau M_2$.*

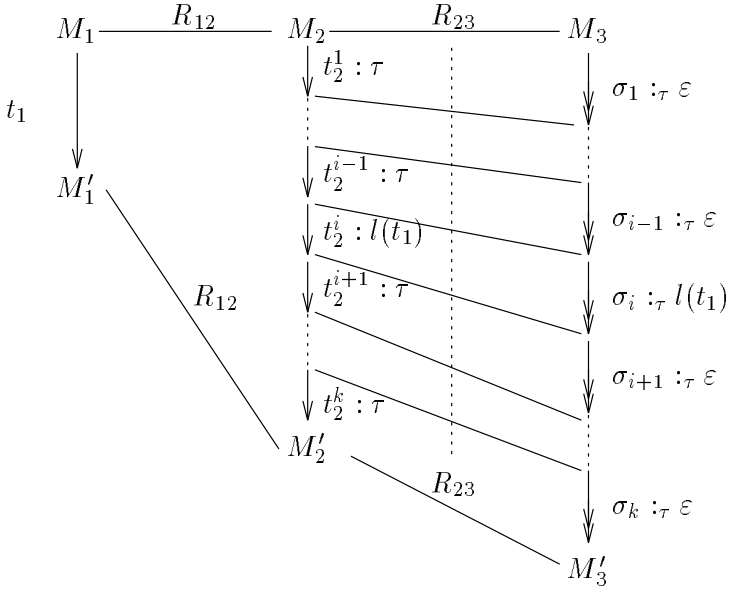
Deux systèmes, $\Sigma_2 = (N_1, M_1)$ et $\Sigma_2 = (N_2, M_2)$, dont les marquages initiaux sont reliés par une τ-bisimulation sont dits τ-bisimilaires. *Nous le notons R : $\Sigma_1 \underline{\leftrightarrow}_\tau \Sigma_2$.*

De la même manière que pour la bisimulation classique, nous avons les propriétés suivantes :

Proposition 4.2.3. *Soient $\Sigma_i = (N_i, M_i)$, $i \in \{1, 2, 3\}$ des systèmes. On note R_{ij}, R'_{ij} des τ-bisimulations entre Σ_i et Σ_j. Nous avons :*

1. *$Id_{\mathcal{M}(P_1)}$, R_{12}^{-1}, $R_{12} \cup R'_{12}$, $\widetilde{R_{12}}$ et $R_{23} \circ R_{12}$ sont des τ-bisimulations.*
2. *$\underline{\leftrightarrow}_\tau \stackrel{\text{déf}}{=} \bigcup\{R_{12} : R_{12}$ τ-bisimulation$\}$ est la plus grande τ-bisimulation entre Σ_1 et Σ_2.*
3. *$\underline{\leftrightarrow}_\tau$ est une relation d'équivalence entre les marquages.*

Preuve. Par rapport à la preuve donnée pour la bisimulation classique, seul $R_{23} \circ R_{12}$ est modifié par la présence de séquences. Nous donnons uniquement une preuve graphique de cette propriété simple :



□

4.3 La τ-bisimulation de place

De la même manière que pour la bisimulation de places stricte, nous dérivons la notion de τ-bisimulation de places :

Définition 4.3.1. *τ-bisimulation de place*

Soit $N = (P, T, W, l)$ un réseau et $B \subseteq P \times P$ une relation réflexive. Nous dirons que B est une τ-bisimulation de place *(ou τ-BdP) ssi $\overline{B}$ est une τ-bisimulation entre les marquages.*

Nous pouvons donner les propriétés qui en découlent :

Proposition 4.3.2. *Étant donné un réseau $N = (P, T, W, l)$*

1. *Id_P est une τ-bisimulation de place sur N ;*
2. *si B, B' sont des τ-bisimulations de place sur N, alors $B \circ B'$ en est une aussi ;*
3. *si B est une τ-bisimulation de place sur N, alors $\widetilde{B}$ (la plus petite relation d'équivalence contenant B) est aussi une τ-bisimulation de place sur N ;*

4. $B_\tau(N) \stackrel{\text{déf}}{=} \bigcup\{B \ : \ B \text{ } \tau\text{-BdP sur } N\}$ *est la plus grande* τ*-BdP sur* N.

5. $B_\tau(N)$ *est une relation d'équivalence ;*

Preuve. Pour 3 et 4, nous pouvons étendre la preuve de la proposition 3.1.2, p. 76.

En effet, si nous considérons deux marquages $M_1 \overline{B} M_2$, où $B = (B_1 \cup B_2)^*$, alors il existe une τ-bisimulation de place R qui est une composée de B_1 et B_2 telle que $M_1 \overline{R} M_2$. Nous imitons par R tout pas $M_1 \stackrel{t_1}{\rightarrow} M_1'$, et atteignons des états intermédiaires et un état final bisimilaires par R à M_1 et M_1', ce qui nous permet de conclure que B est une τ-bisimulation de place, puisque $R \subseteq B$.

Les autres propriétés ont des preuves identiques à celles développées pour la bisimulation de places stricte (prop. 3.2.1, p. 78).

□

4.4 Le réseau quotient

Nous ne rappelons pas les définitions des quotients (voir déf. 2.2.5, p. 57).

Le lemme 2.2.6, p. 58 restant valide, la preuve du théorème de correction du quotient (théo. 2.2.8, p. 59) s'étend facilement au cas de la τ-bisimulation de places.

Nous avons donc :

Corollaire 4.4.1. *Quotient correct pour la* τ*-BdP*

Soient $N = (P, T, W, l)$ *un réseau de Petri et* $E \subseteq P \times P$ *une relation d'équivalence telle que* $\overline{E} \subseteq \underline{\leftrightarrow}_\tau$. *Alors :*

$$\forall M \in \mathcal{M}(N),\ (N, M) \underline{\leftrightarrow}_\tau (N/_{B_\tau(N)}, M/_{B_\tau(N)})$$

4.5 Vers un algorithme efficace

Nous allons montrer ici que la technique utilisée précédemment pour calculer $B(N)$ ne fonctionne pas directement dans le cas de $B_\tau(N)$. Nous donnerons aussi un contre-exemple pour la méthode utilisées dans le cadre des systèmes de transitions.

Les sections suivantes s'attacheront à déterminer un algorithme efficace pour une certaine classe de réseaux.

4.5.1 Calculabilité de $B_\tau(N)$

En effet, grâce au théorème 3.9.1, p. 93, nous avons :

Corollaire 4.5.1. $B_\tau(N)$ *est calculable.*

Comme nous l'avons déjà dit, la méthode sous-jacente est très loin d'être efficace.

4.5.2 Un contre-exemple pour le théorème 3.4.2

Malheureusement, la propriété de transfert local ne suffit plus, i.e. le théorème 3.4.2, p. 83 n'est plus valable.

La figure 4.2 fournit un contre-exemple simple et instructif. En effet, il est facile de voir que

$$B = \{(p_1, p_2), (p_2, p_1), (p_3, p_4), (p_4, p_3)\}$$

vérifie la propriété de transfert local (prop. 3.4.1, p. 82). Néanmoins, dans le marquage $\{p_1, p_3\}$ aucune transition n'est tirable, alors que $\{p_2, p_4\}$ autorise t_3.

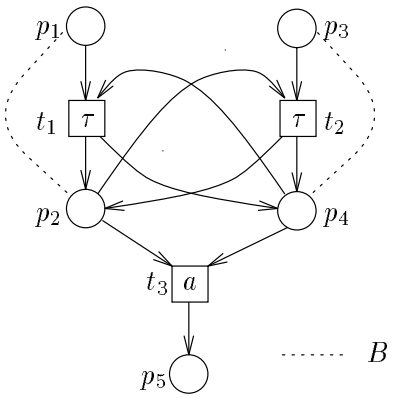


Figure 4.2 : Le théorème 3.4.2, p. 83 n'est plus valable.

4.5.3 La saturation

Une autre approche permettant de calculer $B_\tau(N)$ s'inspire de méthodes développées pour les systèmes de transitions. L'idée est de ramener la τ-bisimulation à une bisimulation stricte en saturant le système ([KS90]). Un réseau τ-saturé est défini par :

Définition 4.5.2. *τ-saturé, τ-saturable*

Étant donné un réseau N, nous dirons que N est τ-saturé *ssi pour toute séquence $\sigma \in T_\tau^*$, il existe une transition $t \in T_\tau$ telle que $t \sqsubseteq \sigma$.*

Nous dirons que N est τ-saturable *si en ajoutant un nombre fini de transitions à N nous obtenons un réseau N' t.q. Id : $N' \underline{\leftrightarrow}_\tau N$.*

L'intérêt de la τ-saturation est de ramener la plus grande τ-BdP à la plus grande BdPS. En effet, nous déduisons directement de la définition de la τ-saturation :

Théorème 4.5.3. *Si N est un réseau τ-saturé, alors $B_\tau(N) = B(N)$.*

Si nous pouvons saturer un réseau N afin d'obtenir un réseau $N' \underline{\leftrightarrow}_\tau N$, alors il est possible de calculer $B_\tau(N) = B(N')$. Cela n'est pas possible en général :

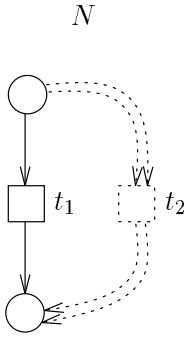
Théorème 4.5.4. *Si N est τ-saturé et N a des τ-transitions non triviales, alors N est infini.*

Preuve. Il suffit de considérer une transition t d'étiquetage τ qui déplace des jetons, et de montrer qu'il n'est pas possible d'imiter dans un réseau fini des séquences composées de t tirée k fois consécutives pour un k quelconque (fig. 4.3).

Formellement, supposons $t \in T_N$ avec ${}^\bullet t \neq t^\bullet$. Nous pouvons supposer que ${}^\bullet t \setminus t^\bullet$ est non vide (ou indifféremment $t^\bullet \setminus {}^\bullet t$), et considérer une place $p \in ({}^\bullet t \setminus t^\bullet)$. Notons t^k la séquence constituée de la transition t tirée k fois consécutives. La place p apparaît au moins k fois dans ${}^\bullet t^k \setminus t^{k\bullet}$. Comme N est τ-saturé, il existe pour chaque séquence t^k une transition t_k telle que $M \xrightarrow{t^k} M'$ implique $M \xrightarrow{t_k} M'$. Il existe une telle transition t_k pour chaque $k \geq 0$, et les t_k sont toutes différentes puisque les ${}^\bullet t^k \setminus t^{k\bullet}$ le sont. Il existe donc une infinité de t_k.

□

Figure 4.3 : N n'est pas τ-saturable.



Ces contre-exemples sont faciles à analyser : la séquence de transitions étiquetées par τ déplace le problème de l'équivalence. On peut donc supposer qu'en se restreignant à des pas parallèles, ce problème disparaîtra. La

section suivante explique cette nouvelle approche et montre que c'est une approximation de $B_\tau(N)$.

En s'inspirant des systèmes de transition et de l'aptitude de la bisimulation de places à conserver le parallélisme (voir section 3.8, p. 89), nous allons imaginer une méthode de saturation.

4.6 Une approximation : $B_{\tau p}$

Les deux voies explorées ci-dessus (transfert faible et saturation) ne peuvent aboutir à un calcul efficace de $B_\tau(N)$. A partir de la seconde d'entre elles, nous examinons une nouvelle technique de saturation de réseaux qui ne conduit pas toujours à la construction de réseaux infinis.

Le problème évoqué dans le théorème 4.5.4 est lié à l'impossibilité d'obtenir dans certains cas un réseau fini dans lequel toute séquence de longueur quelconque de τ-transitions peut être imitée par une seule transition. Une façon de contourner ce problème est d'autoriser l'imitation d'une telle séquence par le tir concurrent d'un multi-ensemble de transitions. Nous appelons la relation permettant de telles imitations une τp-bisimulation.

L'intérêt essentiel de cette relation est d'avoir une caractérisation analogue à la BdPS, i.e. par une propriété locale (théo. 4.7.3, p. 111). Il en résulte un algorithme de calcul de $B_{\tau p}(N)$ efficace.

4.6.1 La τp-bisimulation

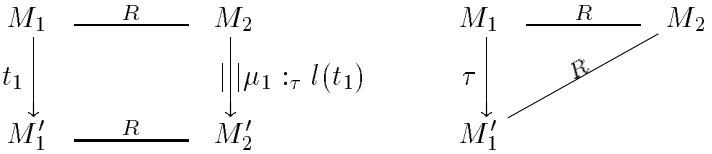
Nous allons donc définir la τp-bisimulation en remplaçant la séquence de la définition de la τ-bisimulation par un pas parallèle, formellement :

Définition 4.6.1. *Propriété de τp-transfert*

Soient $N_1 = (P_1, T_1, W_1, l_1)$ et $N_2 = (P_2, T_2, W_2, l_2)$ deux réseaux, une relation $R \subseteq \mathcal{M}(P_1) \times \mathcal{M}(P_2)$ entre les marquages vérifie la propriété de τp-transfert *ssi :*

- *pour tout $M_1 R M_2$ et tout pas $M_1 \xrightarrow{t_1} M_1'$ il existe un pas $\mu_2 \in \mathcal{M}(T)$ de même étiquette modulo tel que $\tau\ M_2 \equiv \xrightarrow{\mu_2 :_\tau l(t_1)} M_2'$ et $M_1' R M_2'$;*
- *et réciproquement.*

Cette définition est plus explicite sur le schéma suivant :



Remarque : de même que pour la τ-bisimulation, si $l(t_1) = \tau$, alors on peut avoir un pas vide, soit $M_1' R M_2$.

$\diamond$

Nous dérivons :

Définition 4.6.2. *τp-bisimulation entre les marquages*

Une relation ayant la propriété de τp-transfert est une τp-bisimulation (entre les marquages). *Les marquages reliés sont dits* τp-bisimilaires.

Deux systèmes, $\Sigma_2 = (N_1, M_1)$ et $\Sigma_2 = (N_2, M_2)$, dont les marquages initiaux sont reliés par une τp-bisimulation sont dits τp-bisimilaires.

De la même manière que pour la bisimulation classique, nous avons les propriétés suivantes (dont les preuves sont similaires à celles développées pour la τ-bisimulation) :

Proposition 4.6.3. *Soient $\Sigma_i = (N_i, M_i)$, $i \in I_3$ des systèmes. On note R_{ij}, R'_{ij} des τp-bisimulations entre Σ_i et Σ_j. Nous avons :*

1. *$Id_{\mathcal{M}(P_1)}$, R_{12}^{-1}, $R_{23} \circ R_{12}$, $R_{12} \cup R'_{12}$ et $\widetilde{R_{12}}$ sont des τp-bisimulations.*
2. *$\underline{\leftrightarrow}_{\tau p} \stackrel{\text{déf}}{=} \bigcup\{R_{12} \ : \ R_{12}\ \tau p - bisimulation\}$ est la plus grande τp-bisimulation entre Σ_1 et Σ_2.*
3. *$\underline{\leftrightarrow}_{\tau p}$ est une relation d'équivalence entre les marquages.*

Enfin, nous avons surtout la propriété suivante (triviale en comparant les propriétés de transfert) :

Proposition 4.6.4.

$$\underline{\leftrightarrow} \subseteq \underline{\leftrightarrow}_{\tau p} \subseteq \underline{\leftrightarrow}_{\tau}$$

4.6.2 La τp-bisimulation de places

Nous définissons alors la τp-bisimulation de places par :

Définition 4.6.5. *τp-bisimulation de place*

Soit $N = (P, T, W, l)$ un réseau et $B \subseteq P \times P$ une relation réflexive. Nous dirons que B est une τp-bisimulation de place *ssi $\overline{B}$ est une τp-bisimulation entre les marquages.*

De la même manière que pour la τ-bisimulation de places, nous obtenons les propriétés suivantes :

Proposition 4.6.6. *Étant donné un réseau $N = (P, T, W, l)$*

1. *Id_P est une τp-bisimulation de place sur N ;*

2. *si B, B' sont des τp-bisimulations de place sur N, alors $B \circ B'$ en est une aussi ;*

3. *si B est une τp-bisimulation de place sur N, alors $\widetilde{B}$ (la plus petite relation d'équivalence contenant B) est aussi une τ-bisimulation de place sur N ;*

4. *il existe une plus grande τp-bisimulation de place sur N que nous noterons $B_{\tau p}(N)$. De plus, $B_{\tau p}(N) = \bigcup\{B \ : \ B$ bisimulation sur $N\}$.*

5. *$B_{\tau p}(N)$ est une relation d'équivalence ;*

La propriété 4.6.4 nous donne évidemment comme corollaire :

Corollaire 4.6.7.

$$B(N) \subseteq B_{\tau p}(N) \subseteq B_{\tau}(N)$$

Nous allons nous intéresser au calcul de $B_{\tau p}(N)$ et montrer ensuite sous quelles conditions $B_{\tau p}(N)$ permet de calculer $B_{\tau}(N)$ efficacement.

4.6.3 Le réseau quotient

Le lemme 2.2.6, p. 58 restant valide, la preuve du théorème de correction du quotient (théo. 2.2.8, p. 59) s'étend facilement au cas de la τp-bisimulation de places.

Corollaire 4.6.8. *Soit N un réseau, alors :*

$$\forall M \in \mathcal{M}(N), (N, M) \underline{\leftrightarrow}_{\tau p} (N/_{B_{\tau p}(N)}, M/_{B_{\tau p}(N)})$$

4.7 Calculabilité de $B_{\tau p}(N)$

Nous obtenons de manière identique un corollaire du théorème 3.9.1, p. 93 :

Corollaire 4.7.1. *$B_{\tau p}(N)$ est calculable.*

Cependant, nous allons montrer que nous pouvons adapter la propriété de transfert local et dériver un théorème de caractérisation analogue au théorème 3.4.2, p. 83. Ainsi, nous obtiendrons un algorithme efficace.

4.7.1 La propriété de τp-transfert local

De la même manière que pour la bisimulation de places stricte, nous donnons un critère local pour la τp-bisimulation de places :

Définition 4.7.2. *τp-transfert local*

Étant donné un réseau $N = (P, T, W, l)$, nous dirons qu'une relation $R \subseteq P \times P$ a la propriété du τp-transfert local *ssi :*

> *pour tout pas ${}^\bullet t \xrightarrow{t} t^\bullet$ et tous $p, q \in P$ tels que $p \in {}^\circ t$ et pRq il existe un pas $\mu \in \mathcal{M}(T)$ tel que ${}^\bullet t - p + q \xrightarrow{\mu :_\tau l(t)} M'$, $t^\bullet \overline{R} M'$ et $l(\mu) =_\tau l(t)$.*

Ce qui peut s'exprimer plus simplement par le diagramme suivant :

$$\begin{array}{ccc} {}^\bullet t & \overline{R} & {}^\bullet t - p + q \\ t \downarrow & & \Vert \mu :_\tau l(t) \\ t^\bullet & \overline{R} & M \end{array}$$

Par rapport au transfert local, nous ne faisons que remplacer une transition par un pas concurrent. Il suffit de remarquer que ce pas doit avoir des pré- et post-conditions de la même taille que celles de t (donc finis) pour en déduire que cette propriété est vérifiable en un temps fini.

Le théorème central pour utiliser cette propriété est :

Théorème 4.7.3. *Si B est réflexive et a la propriété de τp-transfert local, alors B^* est une τp-bisimulation de places.*

Remarque : une τp-bisimulation de places vérifie trivialement la propriété de τp-transfert local, nous avons donc une propriété caractéristique pour les τp-bisimulations de places.

◇

Preuve. Considérons deux marquages $M_1 \overline{B^*} M_2$ et un pas $M_1 \xrightarrow{t_1} M_1'$. Plusieurs cas se présentent :

1. Si $M_1 \overline{B} M_2$:
   (a) supposons que $M_2 = M_1 - p + q$ avec pBq. Si $M_1 = {}^\bullet t_1$, le τp-transfert local nous permet de dire qu'il existe un pas $\mu \in \mathcal{M}(T)$ tel que $M_2 \xrightarrow{\mu} M_2'$, avec $M_1' \overline{B} M_2'$. Sinon, M_1 est de la forme ${}^\bullet t_1 + M_1''$. Si $p \in M_1''$, alors $M_2 \xrightarrow{t_1} M_1' - p + q$ imite $M_1 \xrightarrow{t_1} M_1'$. Si $p \in {}^\bullet t_1$, alors par la propriété de τp-transfert local il existe un pas ${}^\bullet t_1 - p + q \xrightarrow{\mu} M'$. On peut donc imiter $M_1 \xrightarrow{t_1} M_1'$ par $M_2 \xrightarrow{\mu} M_2'$ avec $M_2' \overline{B} M_1'$, où $M_2' = M_1'' + M'$ et $M_1' = M_1'' + t_1{}^\bullet$,

(b) sinon, il existe un ensemble de marquages tels que

$$M_1 = M^0 \overline{B} M^1 \overline{B} \ldots \overline{B} M^k = M_2$$

avec $M^i = M^{i-1} - p_i + q_i$ et $p_i B q_i$, $i = 1, \ldots, k$. Dans ce cas, $M^0 \stackrel{t_1}{\rightarrow} M'_1$ peut être imité par un $M^1 \stackrel{\mu_1}{\rightarrow} M'^1$. Chaque μ_j étant fini, l'imitation peut se propager jusqu'à $M_2 = M^k \stackrel{\mu_k}{\rightarrow} M'^k$. L'imitation se fait avec $t_2 = \mu_k$, et $M'_2 = M'^k$. Comme $M'_1 \overline{B} M'^1 \overline{B} \ldots \overline{B} M'^k$, nous avons $M'_1 \overline{B^*} M'_2$, soit $M'_1 \overline{B^*} M'_2$ puisque B est réflexive (lemme 3.1.3, p. 76),

2. si $M_1 \overline{B^*} M_2$, il est possible de décomposer en une séquence

$$M_1 = M^1 \overline{B} M^2 \ldots \overline{B} M^k = M_2$$

et d'utiliser le cas précédent pour construire une imitation $M_2 \stackrel{\mu}{\rightarrow} M'_2$.

□

Une conséquence immédiate est d'avoir l'algorithme suivant :

Algorithme 4.7.1. *Calcul de $B_{\tau p}(N)$*
Soit le réseau $N = (P, T, W, l)$:

1. *$B = P \times P$.*

2. *Tester si B a la propriété de τp-transfert local, i.e. :*

   - *s'il existe $t \in T$, $p \in {}^\bullet t$, $q \in P$ avec pBq tels qu'il n'existe pas de pas $\mu \in \mathcal{M}(T)$ de même étiquette modulo τ, tirable à partir de ${}^\bullet t - p + q$ et atteignant un marquage M' t.q. $t^\bullet \overline{B} M'$, alors éliminer les paires (p, q) et (q, p) de B et recommencer l'étape 2.*
   - *sinon, B a la propriété de τp-transfert local.*

3. *B est la plus grande τp-bisimulation de place, i.e. $B = B_{\tau p}(N)$.*

Cet algorithme nous donne un moyen assez simple de calculer $B_{\tau p}(N)$ pour un N donné. Les preuves de terminaison et de correction sont en tout point similaires à celle données pour l'algorithme de calcul de $B(N)$ (voir la section 3.5.2, p. 85). La complexité du calcul de $B_{\tau p}(N)$ est cependant supérieure à celle de $B(N)$, puisqu'il nous faut examiner des multi-ensembles de transitions afin d'imiter t.

4.8 Coût théorique du calcul de $B_{\tau p}(N)$

Nous ne reprenons pas ici tout le calcul effectué dans la section 3.6, mais simplement la fonction `est-imitable (...)`.

Théorème 4.8.1. *$B_{\tau p}(N)$ est calculable en $\mathcal{O}(n_P^2.n_T^{d_T+2})$.*
Le coût de la réduction est alors :

$$\mathcal{O}(n_P^3.n_T^{d+3}.\log(n_T))$$

Preuve. Dans le cas de $B(N)$, nous avions $\mathcal{O}(n_P^3.n_T^3.\log(n_T))$ pour toute la réduction et $B(N)$ se calculait en $\mathcal{O}(n_P^2.n_T^2)$.

La recherche d'une imitation se faisait en $\mathcal{O}(n_T)$ puisque nous cherchions une seule transition. Maintenant, nous cherchons un pas parallèle. Or, pour chaque transition $t \in T$ à imiter, nous savons que le nombre total de places en entrée et en sortie de t est au plus d_T. Il y a donc au plus n_T choix possibles pour consommer (resp. produire) chaque jeton de M_2 (resp. de M_2'). Nous avons donc $n_T^{d_T}$ choix possibles. D'autre part, comme il y a n_T transitions possibles, nous obtenons le facteur $n_T^{d_T+1}$.

Le reste du calcul du coût de la réduction est inchangé.

□

4.9 De $B_{\tau p}(N)$ à $B_\tau(N)$

L'objet de ce chapitre reste le calcul de $B_\tau(N)$. Nous savons que ce calcul n'est pas possible directement à partir des propriétés liées au transfert faible, et qu'il n'est pas possible de τ-saturer n'importe quel réseau.

Nous savons cependant qu'il nous est possible de calculer la plus grande bisimulation pour une définition alternative où une transition est imitée par un pas concurrent. Il s'agit maintenant de relier ces résultats entre eux.

Afin d'alléger la structure de ce chapitre, **cette section ne fait que présenter les résultats essentiels**. Pour les preuves et d'autres résultats (en particulier sur les méthodes de p-saturation[1] d'un réseau, théo. B.3.3, p. 204), nous renvoyons le lecteur à l'annexe B.

Nous allons présenter ici une nouvelle notion, la *p-saturation*, qui est fondamentale pour obtenir un calcul efficace de $B_{\tau p}(N)$. Intuitivement, un réseau est p-saturé si une séquence de transition ayant au plus une étiquette visible peut être remplacée par un pas parallèle, formellement :

Définition 4.9.1. *Réseau p-saturé*

Nous notons $T^\times \stackrel{\text{déf}}{=} \{\sigma \in T^* \ : \ l_\tau(\sigma) =\in \mathcal{A} \cup \{\varepsilon\}\}$, l'ensemble des séquences ayant au plus une étiquette visible.

[1] Cette notion est assez proche de la Δ-saturation de [BK89].

Étant donné un réseau de Petri étiqueté $N = (P, T, W, l)$, nous dirons que N est p-saturé ssi :

$$\forall \sigma \in T^{\times},\ \exists \mu \in \mathcal{M}(T)\ :\ {}^{\bullet}\sigma \xrightarrow{\mu :_{\tau}\ l_{\tau}(\sigma)} \sigma^{\bullet}$$

Nous avons alors le théorème suivant qui découle de la définition :

Théorème 4.9.2. *Si N est p-saturé, alors $B_{\tau}(N) = B_{\tau p}(N)$.*

En complément, nous avons la conjecture forte suivante dont la preuve est en annexe (section B.4, p. 206) :

Conjecture 4.9.1. *Il est décidable si N est p-saturé.*

Une question immédiate est : «comment faire pour un réseau N non p-saturé ?»

Nous allons voir que :

- il existe toujours un réseau p-saturé N^* tel que $B_{\tau}(N) = B_{\tau}(N^*)$ (théo. 4.9.3) ;
- qu'il est possible de construire de façon incrémentale ce réseau et ce en respectant $\underline{\leftrightarrow}_{\tau}$. Donc pour tous les réseaux intermédiaires N_i, nous avons $B_{\tau p}(N_i) \subseteq B_{\tau}(N)$. Malheureusement, cette suite n'est pas croissante (prop. B.2.3, p. 201).
- qu'il est décidable si un N^* peut être construit finiment.

Nous citons ci-dessous les deux théorèmes principaux et détaillons leur preuves dans l'annexe B.

Théorème 4.9.3. *Pour tout réseau N il existe un réseau N^* (fini ou non) p-saturé tel que :*

$$B_{\tau}(N) = B_{\tau}(N^*) = B_{\tau p}(N^*)$$

De plus, N^* peut être construit incrémentalement depuis N en respectant $\underline{\leftrightarrow}_{\tau}$. Nous appellerons *p-saturation* cette construction et nous noterons $N \rightsquigarrow^* N'$ si N' est une p-saturation de N. Nous avons :

Théorème 4.9.4. *(prop. B.2.2, p. 200)*
Si $N \rightsquigarrow^* N'$, alors :

$$B_{\tau p}(N') \subseteq B_{\tau}(N) = B_{\tau}(N')$$

Ce résultat est toutefois relativement optimiste, car s'il existe, N^* n'est pas toujours fini. Nous dirons qu'un réseau est p-saturable s'il existe une p-saturation finie N^* de N. Nous avons le résultat suivant (voir la preuve dans la section B.4, p. 206) :

Théorème 4.9.5. *La p-saturabilité est décidable.*

Nous pouvons donc donner un algorithme pour le calcul de $B_\tau(N)$:

Algorithme 4.9.1. *Calcul de $B_\tau(N)$*

1. *Calculer une p-saturation N^* de N,*
2. *Calculer $B_{\tau p}(N^*)$ en utilisant l'algorithme 4.7.1. $B_\tau(N)$ est $B_{\tau p}(N^*)$.*

La classe des réseaux p-saturables[2] est vaste. Elle contient entre autres les réseaux tels que N_τ soit acyclique (où N_τ est N restreint aux transitions étiquetées par τ). C'est la classe des systèmes ne pouvant pas diverger (en bouclant ou en se bloquant) de manière invisible.

Conclusion

$B_\tau(N)$ est toujours calculable, mais pas toujours de manière très efficace. En particulier, décider si un réseau est p-saturable est très coûteux. En contrepartie, le théorème 4.9.4 nous permet d'utiliser une heuristique différente :

- p-saturer jusqu'à un certain point, on obtient alors une p-saturation N' ;
- calculer $B_{\tau p}(N)$, qui est une approximation de $B_\tau(N)$.

[2]L'annexe B contient quelques éléments pour le coût de la p-saturation

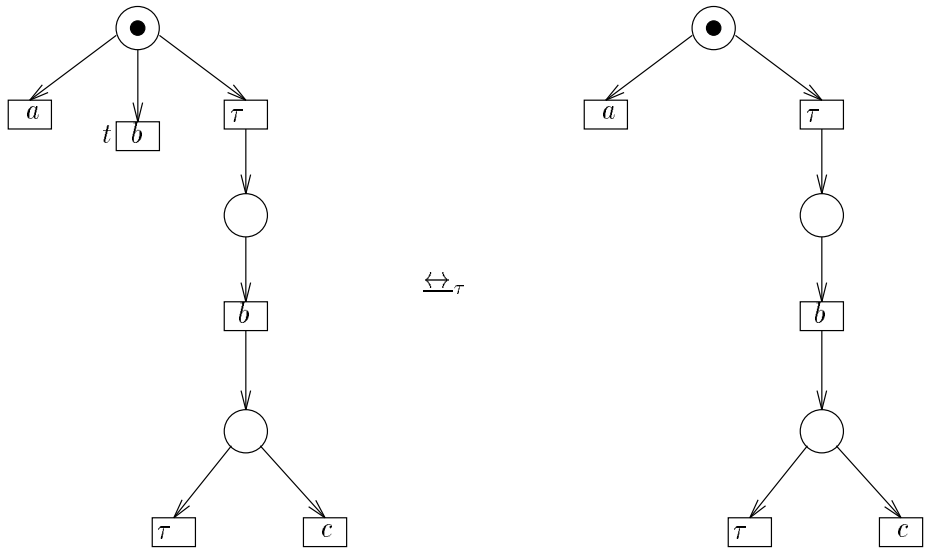
Chapitre 5

Les bisimulations de branchement

En fait, la manière dont la τ-bisimulation voit (et abstrait) les actions internes n'est pas toujours satisfaisante. En particulier, elle ne respecte pas la structure arborescente des choix comme le montre la figure 5.1.

Sur cet exemple, t est imité par $\tau b \tau$. Cependant, dans le second réseau, on choisit de ne pas faire c qu'après avoir fait b.

Figure 5.1 : La τ-bisimulation ne respecte pas la structure des choix.



N_1 N_2

Dans deux réseaux bisimilaires, toute exécution de l'un devrait correspondre à une exécution de l'autre de manière que les états intermédiaires correspondent aussi, de sorte que ceux de la figure 5.1 ne le sont pas.

C'est sur cette base que VAN GLABBEEK & WEIJLAND introduisent la bisimulation de branchement ([GW89b]). Par ailleurs, pour des raisons techniques, WEIJLAND introduit la bisimulation de semi-branchement ([Wei89]).

En-deçà de cet aspect conceptuel, les bisimulations de branchement ont un certain nombre de propriétés intéressantes :

1. à la différence de la τ-bisimulation, elle sont préservées par le raffinement d'action s'il n'introduit pas de parallélisme ([GW89c] ;
2. pour une large classe de réseaux, elles coïncident avec la τ-bisimulation ([GW89a]) ;
3. elles ont des liens forts avec les logiques temporelles (comme CTL*, voir p.ex. [DV90]).

Nous allons étudier ces deux bisimulations et les porter dans le contexte des bisimulations de places.

Dans la première section, nous rappelons les différentes bisimulations de branchement et expliquons leurs liens. L'égalité des plus grandes bisimulations nous permettra de nous restreindre à l'étude de la plus simple (techniquement parlant).

La deuxième section présente donc la *sb*-bisimulation de places, dérivée de la bisimulation de semi-branchement. Suivant un plan maintenant connu, nous présentons les théorèmes permettant de quotienter puis de réduire. Nous montrons aussi comment *dans certains cas*, elle peut être calculée de manière similaire à la τ-bisimulation de places. Nous terminons avec le coût théorique de cette réduction.

5.1 Les bisimulations de branchement

Nous avons les définitions suivantes :

Définition 5.1.1. *Propriété de branchement*

Étant donnés un réseau $N = (P, T, W, l)$, nous dirons qu'une relation $R \subseteq \mathcal{M}(P_1) \times \mathcal{M}(P_2)$ entre les marquages est vérifie la propriété de *b*-transfert *ssi pour tout $M_1 R M_2$ et pour tout pas $M_1 \xrightarrow{t_1} M_1'$:*

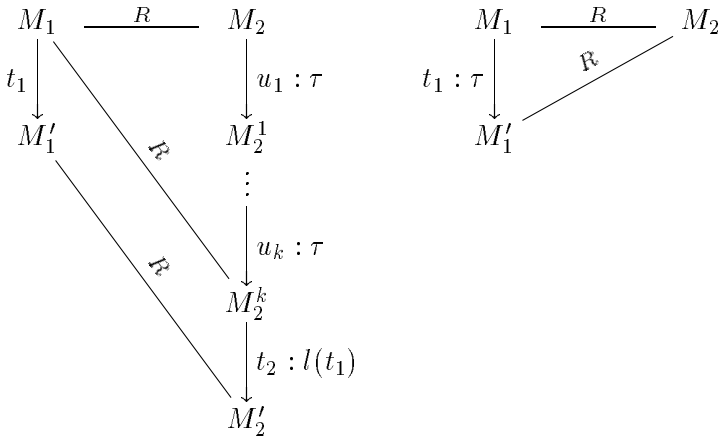
- *il existe une séquence*

$$M_2 \xrightarrow{u_1:\tau} M_2^1 \xrightarrow{u_2:\tau} \ldots \xrightarrow{u_k:\tau} M_2^k \xrightarrow{t_2} M_2'$$

avec $l(t_2) = l(t_1)$ telle que $M_1 R M_2^k$ et $M_1' R M_2'$;

- *si* $l(t_1) = \tau$, *on peut avoir à la place :* $M_1' R M_2$;

Ce qui est plus explicite sur les diagrammes suivants :



Définition 5.1.2. *Bisimulation de branchement [GW89b]*

Une relation symétrique ayant la propriété de b-transfert est une bisimulation de branchement (entre les marquages) *(ou b-bisimulation). Les marquages reliés sont dits b-*bisimilaires. *Nous le noterons* $R : M_1 \underline{\leftrightarrow}_b M_2$.

Deux systèmes, $\Sigma_1 = (N_1, M_1)$ *et* $\Sigma_2 = (N_2, M_2)$, *dont les marquages initiaux sont reliés par une b-bisimulation sont dits b-*bisimilaires. *Nous le notons* $R : \Sigma_1 \underline{\leftrightarrow}_b \Sigma_2$.

En particulier il y a une plus grande b-bisimulation $\underline{\leftrightarrow}_b \stackrel{\text{déf}}{=} \bigcup\{R : R$ *b-bisimulation*$\}$.

La définition de la bisimulation de semi-branchement est légèrement différente : elle ne restreint pas la deuxième alternative aux séquences de longueur 0. Elle a été introduite essentiellement pour ses facilités théoriques. Formellement :

Définition 5.1.3. *Propriété de sb-transfert*

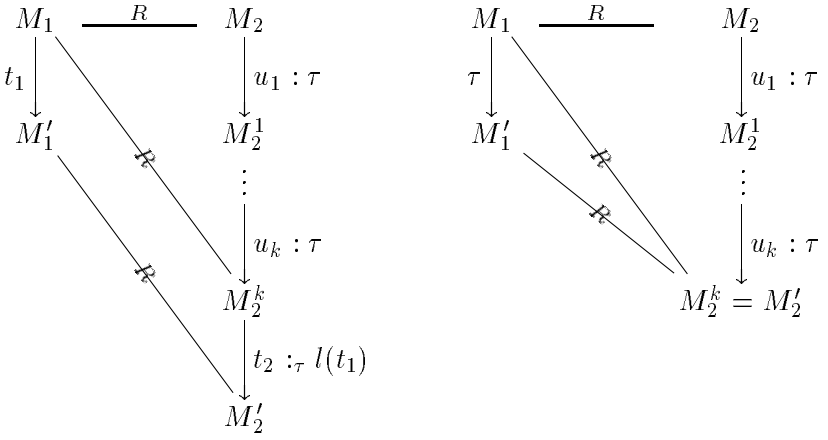
Soient $N_1 = (P_1, T_1, W_1, l_1)$ *et* $N_2 = (P_2, T_2, W_2, l_2)$ *deux réseaux, une relation* $R \subseteq \mathcal{M}(P_1) \times \mathcal{M}(P_2)$ *entre les marquages vérifie la* propriété de *sb-*transfert *ssi pour tout* $M_1 R M_2$ *il existe une séquence*

$$M_2 \stackrel{u_1:\tau}{\longrightarrow} M_2^1 \stackrel{u_2:\tau}{\longrightarrow} \ldots \stackrel{u_k:\tau}{\longrightarrow} M_2^k \stackrel{t_2}{\longrightarrow} M_2'$$

avec $l(t_2) = l(t_1)$ *telle que et pour tout pas* $M_1 \stackrel{t_1}{\longrightarrow} M_1'$:

- $M_1 R M_2^k$ *et* $M_1' R M_2'$;
- *si* $l(t_1) = \tau$ *on peut avoir à la place :* $M_1 R M_2'$ *et* $M_1' R M_2'$;

Ce qui est plus explicite sur les diagrammes suivants :



D'où nous dérivons la définition suivante :

Définition 5.1.4. *Bisimulation de semi-branchement [Wei89]*

*Une relation symétrique ayant la propriété de sb-transfert est une sb-*bisimulation (entre les marquages). *Les marquages reliés sont dits sb-*bisimilaires. *Nous le noterons* $R : M_1 \underline{\leftrightarrow}_{sb} M_2$.

Deux systèmes, $\Sigma_1 = (N_1, M_1)$ *et* $\Sigma_2 = (N_2, M_2)$, *dont les marquages initiaux sont reliés par une sb-bisimulation sont dits sb-*bisimilaires. *Nous le notons* $R : \Sigma_1 \underline{\leftrightarrow}_{sb} \Sigma_2$.

En particulier il y a une plus grande sb-bisimulation $\underline{\leftrightarrow}_{sb} \stackrel{\text{déf}}{=} \bigcup\{R : R \text{ sb-bisimulation}\}$.

Les deux définitions sont très proches et il est facile de voir que toute bisimulation de branchement est une bisimulation de semi-branchement.

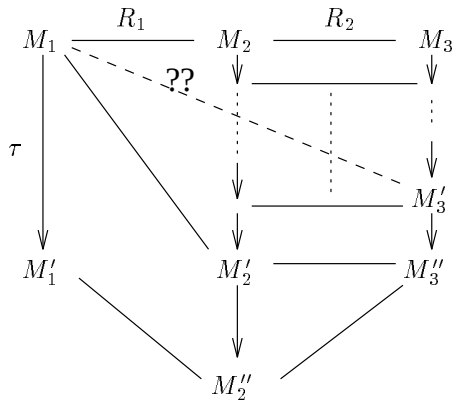
Nous avons une plus grande bisimulation de (semi-)branchement (notées $\underline{\leftrightarrow}_b$ et $\underline{\leftrightarrow}_{sb}$ resp.). De plus, nous avons la proposition suivante :

Proposition 5.1.5. *[Wei89]*

$$\underline{\leftrightarrow}_b = \underline{\leftrightarrow}_{sb}$$

Remarques :

1. La composée de deux bisimulations de branchement n'est pas forcément une bisimulation de branchement. La figure ci-dessous explique pourquoi. L'idée est de prendre deux relations telles que leur composition ne nous permette pas d'utiliser le cas particulier pour les transitions étiquetées par τ non plus que le cas général.



Heureusement, la plus grande bisimulation de branchement coïncide avec la plus grande bisimulation de semi-branchement.

2. Nous avons le lemme suivant ([Wei89]) :

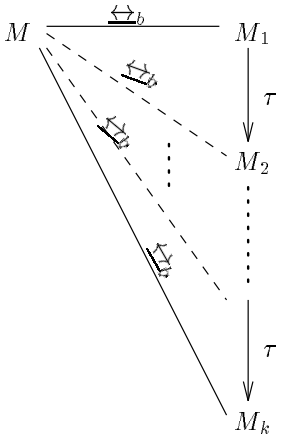
Lemme 5.1.6. «*stuttering lemma*»

Pour tout marquage M et toute séquence de τ :

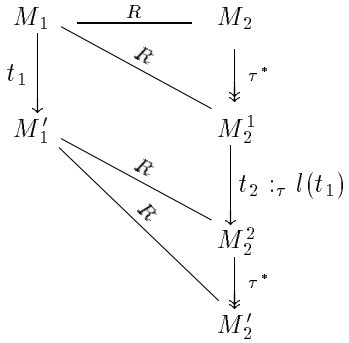
$$\forall M_1 \xrightarrow{\tau} M_2 \xrightarrow{\tau} M_3 \cdots M_{k-1} \xrightarrow{\tau} M_k$$

si $M \underline{\leftrightarrow}_b M_1$ et $M \underline{\leftrightarrow}_b M_k$, alors $M \underline{\leftrightarrow}_b M_i$, $\forall i \in \{1, \ldots, k\}$.

D'où le diagramme suivant :



3. WEIJLAND donne des définitions légèrement différentes en autorisant des séquences de la forme :



Il est facile de voir que les deux définitions sont équivalentes. Cette forme rend néanmoins plus compréhensibles les liens entre b-, η-, Δ et τ-bisimulations (voir ci-après).

4. Nous citons pour mémoire les $\eta-$ et Δ-bisimulations, introduites dans [BBK87] et [Mil80] respectivement et dont les définitions sont données graphiquement sur la figure 5.2.

   Nous avons ([Wei89]) :

$$\underline{\leftrightarrow}_{sb} \subseteq \begin{matrix}\underline{\leftrightarrow}_\eta \\ \underline{\leftrightarrow}_\Delta\end{matrix} \subseteq \underline{\leftrightarrow}_\tau$$

◇

Nous allons donc nous restreindre à l'étude de la *sb*-bisimulation. Nous avons les propriétés suivantes dont la preuve est similaire au cas de la τ-bisimulation (prop. 4.2.3, p. 103, le point 4 découle de la définition) :

Proposition 5.1.7. *Soient $\Sigma_i = (N_i, M_i)$, $i \in \{1,2,3\}$ des systèmes. On note R_{ij}, R'_{ij} des sb-bisimulations entre Σ_i et Σ_j. Nous avons :*

1. *$Id_{\mathcal{M}(P_i)}$, R_{12}^{-1}, $R_{23} \circ R_{12}$, $R_{12} \cup R'_{12}$ et $\widetilde{R_{12}}$ sont des sb-bisimulations.*
2. *$\underline{\leftrightarrow}_{sb} \stackrel{\text{déf}}{=} \bigcup\{R_{12} \ : \ R_{12}$ sb−bisimulation$\}$ est la plus grande sb-bisimulation entre Σ_i et Σ_j.*
3. *$\underline{\leftrightarrow}_{sb}$ est une relation d'équivalence entre les marquages.*
4. *$\underline{\leftrightarrow}_{sb} \subseteq \underline{\leftrightarrow}_\tau$*

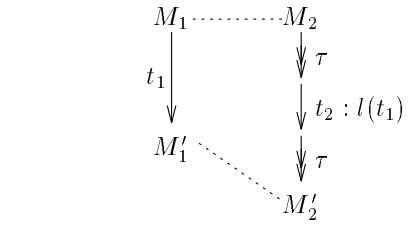
5.2 Les *b*- et *sb*-bisimulations de places

Nous dérivons les définitions des *b*- et *sb*-bisimulations de places :

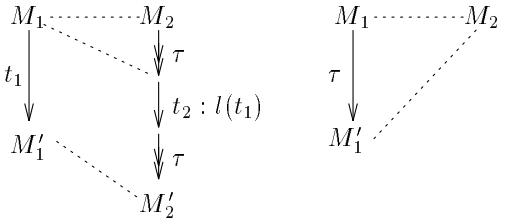
Définition 5.2.1. *b- et sb-bisimulation de places*

Étant donné un réseau $N = (P, T, W, l)$, nous dirons qu'une relation réflexive $B \subseteq P \times P$ est une b-bisimulation de places *(resp. une* sb-bisimulation de places*) ssi $\overline{B}$ est une b-bisimulation (resp. une sb-bisimulation).*

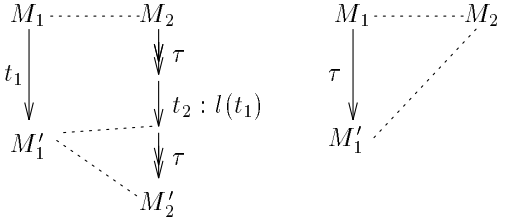
Figure 5.2 : Des bisimulations pour les actions internes.



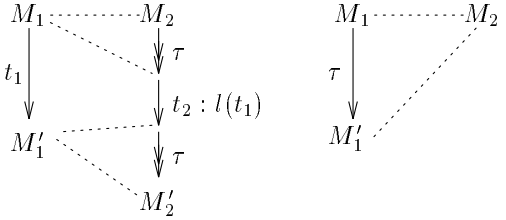
τ-bisimulation [Mil80]



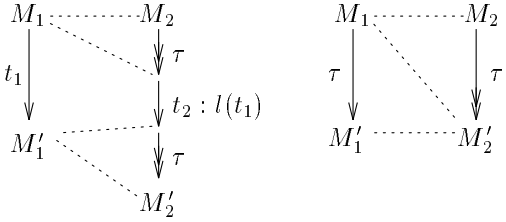
η-bisimulation [BBK87]



Δ-bisimulation [Mil80]



bisimulation de branchement [GW89b]

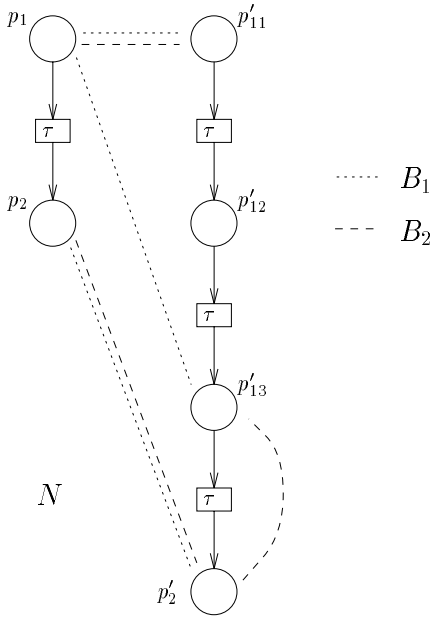


sb-bisimulation [Wei89]

Nous abrégerons ces définitions par b-BdP et sb-BdP respectivement.

Remarque sur la composition de b-bisimulations de places : la figure 5.3 montre que, comme pour la bisimulation de branchement, la composée de b-BdP n'est pas forcément une b-BdP. Sur cette figure, nous prenons $B_1 = Id_P \cup \{(p_1, p'_{11}); (p_1, p'_{13}); (p_2, p'_2)\}$ et $B_2 = Id_P \cup \{(p_1, p'_{11}); (p_2, p'_{13}); (p_2, p'_2)\}$. Il est facile de voir que $\{p_1\}$ n'est pas reliée à $\{p'_{12}\}$ par $(B_1 \circ B_2)$, donc $(B_1 \circ B_2)$ n'est pas une bisimulation de branchement.

Figure 5.3 : $B_1 \circ B_2$ n'est pas une b-bisimulation de places.



◇

Pour la sb-bisimulation de places, nous obtenons les propriétés suivantes en suivant la même démarche que pour la τ-BdP :

Proposition 5.2.2. *Étant donné un réseau $N = (P, T, W, l)$*

1. *Id_P est une sb-bisimulation de places sur N ;*
2. *si B, B' sont des sb-bisimulations de places sur N, alors $B \circ B'$ en est une aussi ;*
3. *si B est une sb-bisimulation de places sur N, alors $\widetilde{B}$ (la plus petite relation d'équivalence contenant B) est aussi une sb-bisimulation de places sur N ;*
4. *$B_{sb}(N) = \bigcup\{B \ : \ B$ sb-BdP sur $N\}$ est la plus grande sb-BdP sur N.*

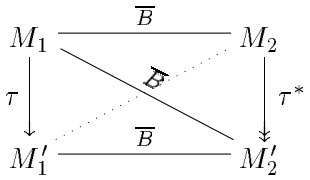
5. $B_{sb}(N)$ *est une relation d'équivalence ;*

Les résultats suivant vont nous permettre de ramener l'étude de la bisimulation de branchement à celle de la *sb*-bisimulation :

Proposition 5.2.3. *$B_{sb}(N)$ est une b-bisimulation de places.*

Preuve. La seule différence est dans l'imitation d'un τ.

Nous notons $B = B_{sb}(N)$. Le diagramme suivant exprime clairement que *la plus grande sb*-bisimulation de places, qui est transitive relie les marquages $M'1$ et M_2 :

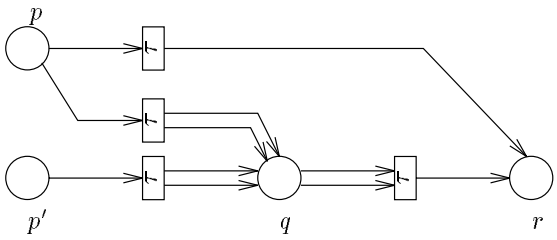


□

D'autre part, il suffit de remarquer que toute bisimulation de branchement est une semi-branchement pour déduire que :

Corollaire 5.2.4. *La plus grande bisimulation b-bisimulation de places sur un réseau N existe et est $B_b(N) = B_{sb}(N)$.*

Remarque sur le «stuttering lemma» (lemme 5.1.6, p. 121) : en général, $\overline{B_{sb}(N)}$ ne vérifie pas ce lemme, comme le montre l'exemple très simple ci-dessous :



$(\{p\}, \{p'\}) \in \overline{B_{sb}(N)}$, mais $(\{p\}, \{q, q\}) \notin \overline{B_{sb}(N)}$

Une conséquence importante est que les techniques classiques utilisées dans les systèmes de transitions ne peuvent être utilisées dans notre cadre (voir p.ex. [GV90]).

◇

5.3 La *sb*-bisimulation de places

Nous allons donc nous restreindre à l'étude de $B_{sb}(N)$ et établir les deux résultats principaux : le théorème du quotient correct et un critère local.

Nous avons le théorème du quotient correct, dont la preuve suit la même démarche que celle du théorème du quotient correct pour $\sim$ (théo. 2.2.8, p. 59) :

Théorème 5.3.1. *Soient $N = (P, T, W, l)$ un réseau de Petri et $E \subseteq P \times P$ une relation d'équivalence telle que $\overline{E} \subseteq \underline{\leftrightarrow}_{sb}$. Alors :*

$$\forall M \in \mathcal{M}(N),\ (N, M) \underline{\leftrightarrow}_{sb} (N/_E, M/_E)$$

Preuve. Il suffit de montrer que $R \stackrel{déf}{=} [\cdot]_E \circ \underline{\leftrightarrow} \subseteq \mathcal{M}_N \times \mathcal{M}_{N/E}$ est une bisimulation entre N et $N/_E$, donc de vérifier la propriété de *sb*-transfert pour R.

Tout d'abord, le lemme 2.2.6, p. 58 reste valable dans le cadre de la *sb*-BdP.

1. Soient $M_1 \stackrel{t_1}{\rightarrow} M_1'$ et $M_1 R M_2$. Le diagramme suivant établit le premier point de la propriété de transfert :

$$\begin{array}{ccccc}
M_1 & \underline{\leftrightarrow}_{sb} & M_3 & [\cdot]_E & M_2 = [M_3]_E \\
{\scriptstyle\tau}\downarrow & \swarrow_{sb} & {\scriptstyle\tau^*}\downarrow & & \downarrow{\scriptstyle\tau^*} \\
M_1' & \underline{\leftrightarrow}_{sb} & M_3' & [\cdot]_E & M_2' = [M_3']_E
\end{array}$$

et

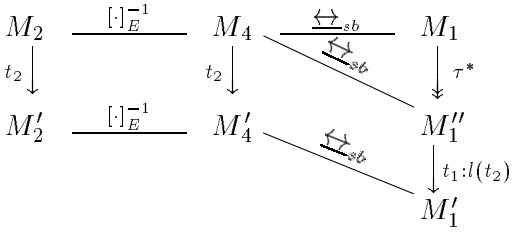
$$\begin{array}{ccccc}
M_1 & \underline{\leftrightarrow}_{sb} & M_3 & [\cdot]_E & M_2 = [M_3]_E \\
{\scriptstyle t_1}\downarrow & \swarrow_{sb} & {\scriptstyle\tau^*}\downarrow & & \downarrow{\scriptstyle\tau^*} \\
M_1' & & M_3'' & [\cdot]_E & M_2'' \\
 & \swarrow_{sb} & {\scriptstyle t_3:l(t_1)}\downarrow & & \downarrow{\scriptstyle t_3} \\
 & & M_3' & [\cdot]_E & M_2' = [M_3']_E
\end{array}$$

$M_3 \stackrel{\tau^*}{\rightarrow\!\!\!\!\!\rightarrow} \stackrel{t_3}{\rightarrow} M_3'$ existe par bisimilarité et $M_2 \stackrel{\tau^*}{\rightarrow\!\!\!\!\!\rightarrow} \stackrel{t_3}{\rightarrow} M_2'$ est directement issue du lemme 2.2.6.

2. Soit maintenant un pas $M_2 \stackrel{t_2}{\rightarrow} M_2'$ dans $N/_E$ et $M_1 R M_2$. De $M_1 R M_2$, nous déduisons qu'il existe un marquage M_3 tel que $M_1 \underline{\leftrightarrow}_{sb} M_3\, [\cdot]_E\, M_2$. D'après le lemme 2.2.6, nous savons qu'il existe un pas $M_4 \stackrel{t_4}{\rightarrow} M_4'$ dans N fermant le diagramme suivant :

$$\begin{array}{ccc}
M_2 & \stackrel{[\cdot]_E^{-1}}{\longrightarrow} & M_4 \\
{\scriptstyle t_2}\downarrow & & \downarrow{\scriptstyle t_2} \\
M_2' & \stackrel{[\cdot]_E^{-1}}{\longrightarrow} & M_4'
\end{array}$$

Par suite $M_3 \overline{E} M_4$ et, comme $\overline{E} \subseteq \underline{\leftrightarrow}$, $M_3 \underline{\leftrightarrow} M_4$. Comme $M_1 \underline{\leftrightarrow} M_3$, nous avons $M_4 \underline{\leftrightarrow} M_1$ et la propriété de *sb*-transfert nous permet de compléter le diagramme :



et similairement pour la deuxième manière d'imiter un τ. Ce qui complète la preuve.

□

La propriété de *sb*-transfert correspond au cadre du théorème 3.9.1, p. 93 et nous avons :

Corollaire 5.3.2. *(du théo. 3.9.1)*
$B_{sb}(N)$ est calculable.

Groote et Vaandrager dans [GV90], utilisent explicitement la finitude de l'espace d'état et le «stuttering lemma» pour calculer efficacement $\underline{\leftrightarrow}_{sb}$ dans le cadre des systèmes de transitions. Leur méthode n'est donc pas transférable à notre cas.

Nous allons voir que pour une certaine classe de réseaux, il existe un algorithme efficace pour une famille de réseaux. De la même manière que nous avons approché la τ-bisimulation par la τp-bisimulation, nous allons travailler avec une approximation de la *sb*-bisimulation : la *sbp*-bisimulation.

Nous pourrons alors établir les corollaires nécessaires pour montrer que $B_{sb}(N)$ et $B_{sbp}(N)$ coïncident pour la classe des réseaux τp-saturés.

5.4 La *sbp*-bisimulation

Dans cette section, nous allons montrer que :

- pour la classe des réseaux *τp-saturés* (déf. 5.5.5), $B_{sb}(N)$ est calculable efficacement ;
- qu'il n'est pas possible, a priori, de τp-saturer un réseau en respectant $\underline{\leftrightarrow}_{sb}$.

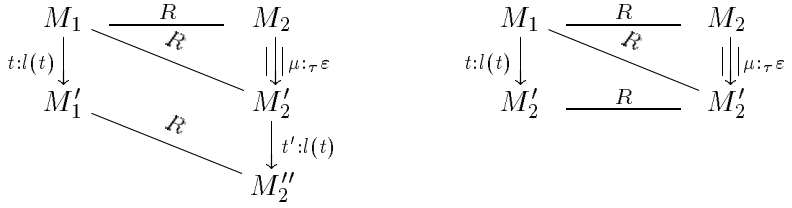
La *sbp*-bisimulation s'obtient simplement en remplaçant les séquences de τ de la définition 5.1.4 par un pas concurrent. Formellement :

Définition 5.4.1. *Propriété de sbp-transfert*

Soient $N_1 = (P_1, T_1, W_1, l_1)$ *et* $N_2 = (P_2, T_2, W_2, l_2)$ *deux réseaux, une relation* $R \subseteq \mathcal{M}(P_1) \times \mathcal{M}(P_2)$ *entre les marquages vérifie la* propriété de *sbp*-transfert *ssi pour tout* $M_1 R M_2$ *et pour tout pas* $M_1 \xrightarrow{t_1} M_1'$ *:*

- *il existe un pas* $M_2 \equiv\!\!\xrightarrow{\mu:\tau\varepsilon}\!\!\rightarrow M_2^1$ *et une transition* $M_2^1 \xrightarrow{t_2} M_2'$ *avec* $l(t_2) = l(t_1)$ *tels que* $M_1 R M_2^1$ *et* $M_1' R M_2'$ *;*
- *si* $l(t_1) = \tau$ *on peut avoir aussi un pas* $M_2 \equiv\!\!\xrightarrow{\mu:\tau\varepsilon}\!\!\rightarrow M_2'$ *tel que* $M_1 R M_2'$ *et* $M_1' R M_2'$ *;*

Ce qui est plus explicite sur les diagrammes suivants :



D'où les bisimulations correspondantes :

Définition 5.4.2. *sbp-bisimulation entre les marquages*

*Une relation symétrique ayant la propriété de sbp-transfert est une sbp-*bisimulation (entre les marquages). *Les marquages reliés sont dits sbp-*bisimilaires.

Deux systèmes, $\Sigma_2 = (N_1, M_1)$ *et* $\Sigma_2 = (N_2, M_2)$*, dont les marquages initiaux sont reliés par une sbp-bisimulation sont dits sbp-*bisimilaires.

De cette définition découle que toute *sbp*-bisimulation est une *sb*-bisimulation.

Définition 5.4.3. *sbp-bisimulation de place*

Soit $N = (P, T, W, l)$ *un réseau et* $B \subseteq P \times P$ *une relation réflexive. Nous dirons que* B *est une sbp-*bisimulation de places *(ou sbp-BdP) ssi* $\overline{B}$ *est une sbp-bisimulation entre les marquages.*

Malheureusement, la *sbp*-bisimulation (de places) n'a pas toutes les «bonnes» propriétés de la τp-bisimulation (de places). Particulièrement, la composée de deux *sbp*-bisimulation (de places) n'est en général pas une *sbp*-bisimulation (de places).

Nous allons voir que cela n'est pas gênant dans la mesure où notre objectif est le calcul de $B_{sb}(N)$.

5.5 Correction et calculabilité de la *sbp*-BdP

Nous avons la propriété fondamentale suivante, qui découle directement des définitions :

Proposition 5.5.1.

$$\underline{\leftrightarrow} \subseteq \underline{\leftrightarrow}_{sbp} \subseteq \underline{\leftrightarrow}_{sb} \subseteq \underline{\leftrightarrow}_{\tau}$$

En suivant la même démarche que pour le théorème 5.3.1, nous avons :

Théorème 5.5.2. *Soient $N = (P, T, W, l)$ un réseau de Petri et $E \subseteq P \times P$ une relation d'équivalence telle que $\overline{E} \subseteq \underline{\leftrightarrow}_{sbp}$. Alors :*

$$\forall M \in \mathcal{M}_N,\ (N, M)\, \underline{\leftrightarrow}_{sbp}\, (N/_{B_{sbp}(N)}, M/_{B_{sbp}(N)})$$

Nous pouvons alors dérouler le même canevas que pour $B_{\tau p}(N)$:

- définir une propriété locale (déf. 5.5.7) ;
- montrer qu'elle caractérise les *sbp*-BdP *pour la classe des réseaux τp-saturés* ;
- montre que dans cette classe $B_{sbp}(N) = B_{sb}(N)$;
- en dériver un algorithme et le coût théorique du calcul de $B_{sb}(N)$.

Nous avons la propriété suivante, conséquence de la proposition 5.5.1 :

Corollaire 5.5.3.

$$B(N) \subseteq B_{sbp}(N) \subseteq B_{sb}(N) \subseteq B_{\tau}(N)$$

Le théorème 3.9.1, p. 93 a pour conséquence :

Corollaire 5.5.4. *$B_{sbp}(N)$ est calculable.*

Il nous reste à donner une caractérisation de la *sbp*-bisimulation de places.

5.5.1 τp-saturation et *sb*-bisimulation

Nous avons :

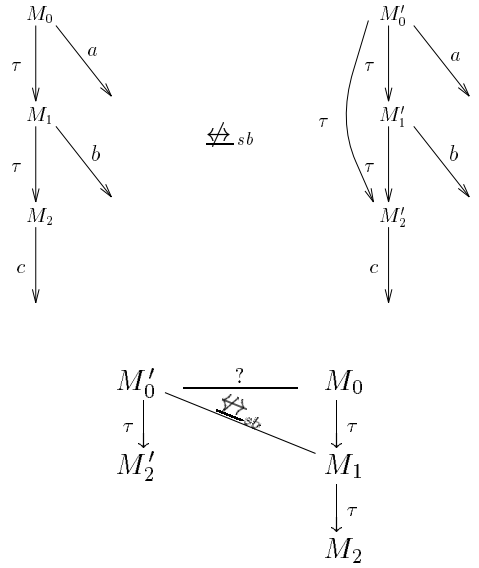
Définition 5.5.5. *Réseau τp-saturé*
Soit $N = (P, T, W, l)$ un réseau, N est τp-saturé *ssi :*

$$\forall \sigma \in T_{\tau}^{*},\ \exists \mu \in \mathcal{M}(T_{\tau})\ :\ \mu \sqsubseteq \sigma$$

Dans le cadre de la τ-bisimulation, un intérêt fondamental de la p-saturation est de pouvoir se faire algorithmiquement en respectant $\underline{\leftrightarrow}_\tau$ (prop. B.2.2, p. 200). Malheureusement, nous ne pouvons pas agir aussi mécaniquement avec la sb-bisimulation :

Proposition 5.5.6. *La τp-saturation ne respecte pas toujours $\underline{\leftrightarrow}_{sb}$.*

Preuve. En effet, il n'est pas toujours de τp-saturer en restant sb-bisimilaires comme le montre la figure ci dessous :



□

5.5.2 Calculer $B_{sbp}(N)$ dans les réseaux τp-saturés

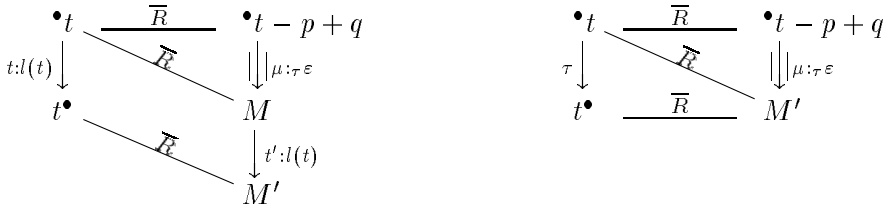
Nous prenons une adaptation naturelle de la propriété de τp-transfert local (déf. 4.7.2, p. 111) :

Définition 5.5.7. *Propriété de sbp-transfert local*

Étant donné un réseau $N = (P, T, W, l)$, nous dirons qu'une relation $R \subseteq P \times P$ a la propriété du *sbp*-transfert local *ssi :*

> *pour tout pas ${}^\bullet t \xrightarrow{t} t^\bullet$ et tous $p, q \in P$ tels que $p \in {}^\bullet t$ et pRq il existe un pas $\mu \in \mathcal{M}(T_\tau)$ telle que, ${}^\bullet t - p + q \overset{\mu}{\Longrightarrow} M$, ${}^\bullet t \overline{R} M$ et :*
>
> - *il existe une transition t' telle que $l(t') = l(t)$ et $M \xrightarrow{t'} M'$ avec $t^\bullet \overline{R} M'$.*
> - *si $l(t) = \tau$, on peut aussi avoir $t^\bullet \overline{R} M$.*

ce qui peut s'exprimer plus simplement par les diagrammes suivant :



L'extension du théorème de caractérisation n'est pas aussi directe. En effet, nous avons toujours besoin de conserver un «ordre» entre μ et t'. Intuitivement, nous savons déjà que la *sbp*-BdP ne nous permettra de calculer $B_{sb}(N)$ que dans le cas d'un réseau saturé en un certain sens. Nous allons montrer que pour ces réseaux, la *sbp*-BdP est caractérisée par un critère local.

Théorème 5.5.8. *(Caractérisation des sbp-BdP pour les réseaux τp-saturés)*

Soit N un réseau τp-saturé et $B \subseteq P \times P$ une relation réflexive ayant la propriété de sbp-transfert local, alors B^* est une sbp-bisimulation de places.

Preuve. Considérons deux marquages $M_1 \overline{B^*} M_2$ et un pas $M_1 \xrightarrow{t_1} M_1'$. Nous notons $M_1 = {}^\bullet t_1 + M_1''$. Plusieurs cas se présentent :

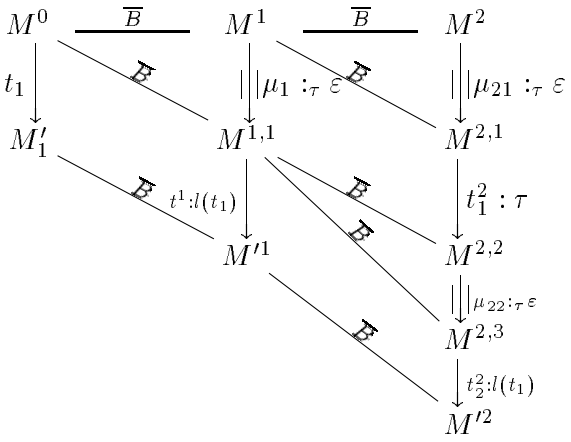
1. Si $M_1 \overline{B} M_2$:

   - supposons que $M_2 = M_1 - p + q$ avec pBq.
     - soit $p \in {}^\bullet t_1$ et la propriété de *sbp*-transfert local pour B est directement utilisable ;
     - soit $p \in M_1''$ et t_1 est sa propre imitation.
   - sinon, il existe un ensemble de marquages tels que

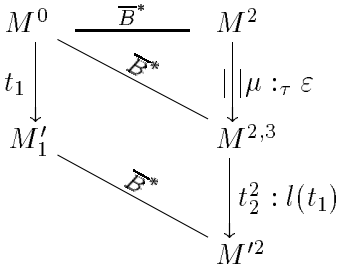
   $$M_1 = M^0 \overline{B} M^1 \overline{B} \ldots \overline{B} M^k = M_2$$

   avec $M^i = M^{i-1} - p_i + q_i$ et $p_i B q_i$, $i = 1, \ldots, k$. Dans ce cas, nous avons les diagrammes ci-dessous (le cas $l(t_1) = \tau$ étant traité de

la même manière) :



Le réseau étant par hypothèse τp-saturé, nous avons un pas concurrent $\mu \sqsubseteq \mu_{21}.t_1^2.\mu_{22}$ et par suite nous avons :



Chaque μ_* étant fini, l'imitation peut se propager jusqu'à M_2. Comme $M'_1\overline{B}M'^1\overline{B}\ldots\overline{B}M'^k$, nous avons $M'_1\overline{B^*}M'_2$, soit $M'_1\overline{B^*}M'_2$ puisque B est réflexive (lemme 3.1.3, p. 76). Le même raisonnement est applicable pour les marquages intermédiaires.

2. Dans le cas général, si $M_1\overline{B^*}M_2$, il est possible de décomposer en une séquence
$$M_1 = M^1\overline{B}M^2\ldots\overline{B}M^k = M_2$$
et d'utiliser les résultats ci-dessus pour construire une imitation $M_2 \stackrel{\sigma}{\rightarrow} M'_2$.

□

Une conséquence immédiate est d'avoir l'algorithme suivant :

Algorithme 5.5.1. *Calcul de $B_{sbp}(N)$*
Soit le réseau τp-saturé $N = (P, T, W, l)$:

1. $B = P \times P$.

2. *Tester si B a la propriété de sbp-transfert local, i.e. :*

   - *s'il existe $t \in T$, $p \in {}^\bullet t$, $q \in P$ avec pBq tels que le diagramme de la définition ne puisse être fermé, alors éliminer les paires (p,q) et (q,p) de B et recommencer l'étape 2.*
   - *sinon, B a la propriété de sbp-transfert local.*

3. *B est la plus grande sbp-bisimulation de place, i.e. $B = B_{sbp}(N)$.*

Cet algorithme nous donne un moyen assez simple de calculer $B_{sbp}(N)$ pour un N donné. L'algorithme est correct et termine pour les mêmes raisons que les précédents (voir la section 3.5.2, p. 85). Sa complexité est cependant supérieure à celle du calcul de $B_{\tau p}(N)$, puisqu'il nous faut examiner en plus une transition pour imiter t.

5.6 Coût théorique du calcul de $B_{sbp}(N)$

Nous ne reprenons pas ici tout le calcul effectué dans la section 3.6.2, mais simplement la fonction `est-imitable (...)`.

Théorème 5.6.1. *Si nous supposons que le degré des transitions de N est borné par une valeur d, $B_{sbp}(N)$ est calculable en $\mathcal{O}(n_P^2.n_T^{d+3})$.*

Le coût de la réduction est alors :

$$\mathcal{O}(n_P^3.n_T^{d+4}.\log(n_T))$$

pour des familles de réseaux où les degrés sont bornés par une constante.

Preuve. Le calcul de $B_{\tau p}(N)$ se fait en $\mathcal{O}(n_P^2.n_T^{d_T+2})$.

Or pour $B_{sbp}(N)$, nous avons éventuellement à chercher une transition supplémentaire, donc $B_{sbp}(N)$ a un coût théorique de $\mathcal{O}(n_P^2.n_T^{d_T+3})$.

Le reste du calcul du coût de la réduction est inchangé.

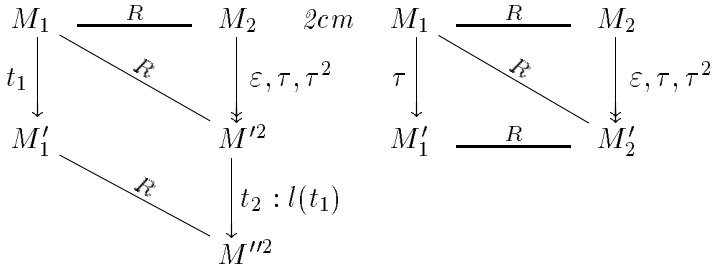
□

5.7 Une autre approche : la sb_2-bisimulation

Nous avons vu que nous ne pouvions pas «brutalement» τp-saturer un réseau en restant sb-bisimilaires (prop. 5.5.6). Nous allons voir qu'en prenant quelques précautions, nous pouvons calculer une approximation de $B_{sb}(N)$.

Définition 5.7.1. *Propriété de sb_2-transfert*

Soient $N_1 = (P_1, T_1, W_1, l_1)$ et $N_2 = (P_2, T_2, W_2, l_2)$ deux réseaux, une relation $R \subseteq \mathcal{M}(P_1) \times \mathcal{M}(P_2)$ entre les marquages vérifie la propriété de sb_2-transfert *ssi :*



D'où les bisimulations correspondantes :

Définition 5.7.2. *sb_2-bisimulation entre les marquages*

Une relation symétrique ayant la propriété de sb_2-transfert est une sb_2-bisimulation (entre les marquages). *Les marquages reliés sont dits* sb_2-bisimilaires.

Deux systèmes, $\Sigma_2 = (N_1, M_1)$ et $\Sigma_2 = (N_2, M_2)$, dont les marquages initiaux sont reliés par une sb_2-bisimulation sont dits sb_2-bisimilaires.

En limitant la longueur des séquences de τ, nous conservons certaines propriétés, mais perdons la transitivité :

Proposition 5.7.3. *Soient R, R' des sb_2-bisimulations, nous avons :*

1. *$Id_{\mathcal{M}_N}$, R^{-1}, $R \cup R'$ sont des sb_2-bisimulations ;*
2. *$\underline{\leftrightarrow}_{sb_2} \stackrel{\text{déf}}{=} \bigcup\{R : sb_2\text{-bisimulations}\}$ est une sb_2-bisimulation. En général, $\underline{\leftrightarrow}_{sb_2}$ n'est pas une relation d'équivalence.*
3. *$\widetilde{R} \subseteq \underline{\leftrightarrow}_{sb}$, en particulier $\widetilde{\underline{\leftrightarrow}_{sb}}2 \subseteq \underline{\leftrightarrow}_{sb}$.*

Preuve.

- 1 et 2 sont de simples conséquences de la définition.
- 3 vient simplement de ce que $R \subseteq \underline{\leftrightarrow}_{sb}$.

□

De cette définition découle que toute sb_2-bisimulation est une sb-bisimulation.

Définition 5.7.4. *sb_2-bisimulation de places*

Soit $N = (P, T, W, l)$ un réseau et $B \subseteq P \times P$ une relation réflexive. Nous dirons que B est une sb_2-bisimulation de places *(ou sb_2-BdP) ssi $\overline{B}$ est une sb_2-bisimulation entre les marquages.*

Nous retrouvons donc les propriétés classiques, a l'exception de celles basées sur la transitivité :

Proposition 5.7.5. *Soient B et B' des sb_2-BdP, nous avons :*

1. *Id_P, B^{-1} et $B \cup B'$ sont des sb_2-BdP ;*
2. *$B_{sb_2}(N) \stackrel{\text{déf}}{=} \bigcup\{B \ : \ Bsb_2\text{-}BdP\}$ est une sb_2-BdP. En général, $B_{sb_2}(N)$ n'est pas une relation d'équivalence.*
3. *$\widetilde{B_{sb_2}(N)} \subseteq B_{sb}(N)$*

Preuve.

- 1 et 2 sont de simple conséquences de la définition et de la propriété 5.7.3.
- 3 vient de $B_{sb_2}(N)$ réflexive et symétrique, donc :

$$\widetilde{B_{sb_2}(N)} = (B_{sb_2}(N))^*$$

or

$$\overline{(\widetilde{B_{sb_2}(N)})} = \overline{(B_{sb_2}(N))^*} = (\overline{B_{sb_2}(N)})^* \subseteq \underline{\leftrightarrow}_{sb}$$

en utilisant le lemme 3.1.3, p. 76 et la propriété 5.7.3.

□

Nous allons maintenant nous intéresser à un procédé de couverture qui permet d'une part de rester sb-bisimilaire, d'autre part d'alterner les étapes de couverture et de calcul de la $B_{sb_2}(N)$ pour obtenir une approximation de $B_{sb}(N)$. Formellement :

Définition 5.7.6. *sb_2-couverture, sb_2-saturé*

Étant donnés $N = (P, T, W, l)$ et $t_1, t_2 \in T$ telles que :

$$M \xrightarrow{t_1:\tau} \xrightarrow{t_2:\tau} M' \ \wedge \ M \, \overline{B_{sb_2}(N)} \, M'$$

*nous appellerons sb_2-*couverture de $t_1.t_2$ *la transition $t \equiv t_1.t_2$. Nous noterons $N \rightsquigarrow_{sb2\ t_1,t_2} N'$ si N' est N auquel on a ajouté la transition t.*

De plus : $N \rightsquigarrow^*_{sb2}\ N' \stackrel{\text{déf}}{\Longleftrightarrow} N \rightsquigarrow_{sb2\ t_1,t'_1} N_1 \rightsquigarrow_{sb2\ t_2,t'_2} \ldots \rightsquigarrow_{sb2\ t_k,t_k} N'$.

*Enfin, nous avons $N \not\rightsquigarrow_{sb2}$ ssi aucune sb_2-couverture n'est possible. Nous dirons alors que N est sb_2-*saturé*.*

Remarque sur la sb_2-couverture : quelles conséquences cette nouvelle forme[1] de couverture entraîne-t-elle ? Tout d'abord, les classes de réseaux sb_2-saturables sont les mêmes, puisque fondamentalement la méthode ne change pas.

[1] Par rapport à la définition B.0.1, p. 197.

Il reste à prouver un théorème équivalent à B.3.3, p. 204, i.e. que toute stratégie équitable est complète (voir les déf. B.3.2, p. 204 et B.3.1, p. 203 resp). Cela vient simplement de ce que la stratégie est complètement déterministe.

◇

Nous allons maintenant nous intéresser à la correction de la méthode de couverture. En fait, $B_{sb}(N)$ étant notre objectif, la proposition suivante nous suffit :

Proposition 5.7.7.

$$N \rightsquigarrow^*_{sb2} N' \Rightarrow \forall M_0 \in \mathcal{M}_N = \mathcal{M}_{N'} \ : \ (N, M_0) \underline{\leftrightarrow}_{sb} (N', M_0)$$

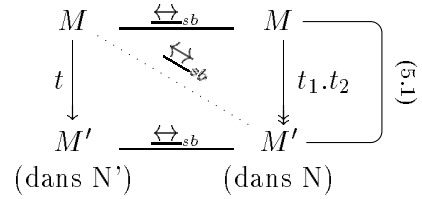
Preuve.

Il suffit de vérifier pour $N \rightsquigarrow_{sb2 \ t_1,t_2} N'$, le cas général se déduisant par transitivité de $\underline{\leftrightarrow}_{sb}$.

Remarquons que N' est auquel nous avons rajouté t. Pour vérifier la propriété de sb-transfert, il suffit de pouvoir imiter $M \xrightarrow{t} M'$ dans N, ce qui découle simplement de :

$$\overline{B_{sb_2}(N)} \subseteq \underline{\leftrightarrow}_{sb_2} \subseteq \underline{\leftrightarrow}_{sb} \tag{5.1}$$

ce qui nous donne :



et nous concluons grâce à la transitivité de $\underline{\leftrightarrow}_{sb}$.

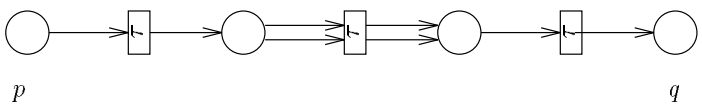
□

Corollaire 5.7.8. *(du théorème 5.3.1, p. 126)*
Si $N \rightsquigarrow^*_{sb2} N'$, *alors :*

$$\forall M_0 \in \mathcal{M}_N = \mathcal{M}_{N'} \ : \ (N'/_{B_{sb_2}(N')}, M_0/_{B_{sb_2}(N')}) \underline{\leftrightarrow}_{sb} (N, M_0)$$

Remarques sur la calculabilité et la convergence vers $B_{sb}(N)$:

1. Nous avons les mêmes problèmes qu'avec la p-saturation : la suite des réseaux obtenus par sb_2-couverture n'est pas une suite croissante (voir la prop. B.2.3, p. 201).
2. Lorsqu'un réseau est sb_2-saturé (i.e il n'y a plus de sb_2-couverture possible), nous n'avons pas toujours $B_{sb_2}(N) = B_{sb}(N)$ (exemple ci-dessous).



$$B_{sb}(N) = P \times P, \text{ mais } (p, q) \notin B_{sb_2}(N)$$

3. Cela nous amènerait à étudier les $sb_3, \ldots$-BdP. Ce nous semble une approche viable pour les réseaux τ-conservateurs.
4. Pour calculer, nous ne pouvons utiliser le transfert local, mais le semi-local fonctionne et reste assez efficace puisque nous nous limitons aux séquences de longueur 2.
5. Pour quelles classes de réseaux a-t-on $B_{sb_2}(N) = B_{sb_k}(N)$?

$\diamond$

Conclusion

Comme pour $B_\tau(N)$, il existe un algorithme efficace pour $B_{sb}(N)$ pour une classe de réseaux, mais nous ne savons pas précisément, actuellement, comment étendre cette classe. Particulièrement, il reste à explorer l'utilité des sb_n-BdP, où la couverture se ferait pour des séquences de longueur $i \leq n$.

Pour les η- et Δ-bisimulations, il est possible d'utiliser les mêmes techniques.

Nous allons nous intéresser à une extension classique des réseaux de Petri : les arcs inhibiteurs.

Chapitre 6

Bisimulation de place stricte et arcs inhibiteurs

Il existe de nombreuses extensions des réseaux de Petri. Parmi celles-ci, les arcs inhibiteurs ont un place particulière puisqu'ils leur donnent la puissance des machines de Turing. Il s'agit donc bien d'une extension et non d'une simple «facilité» syntaxique.

Intuitivement, un *arc inhibiteur* va d'une place à une transition et permet d'interdire le franchissement de cette transition si cette place est marquée[1].

Remarques sur les arcs de test :

1. Un autre type d'arcs est également fréquemment utilisé, les *arcs de test*. Ces arcs ont un but dual des arcs inhibiteurs : interdire le franchissement d'une transition si un certain marquage n'est pas atteint.

   Dans [CH93a] et [Lak93], un certain nombre de méthodes permettent de passer des arcs de test aux arcs inhibiteurs. Elles sont basées sur le fait que *pratiquement* de tels arcs sont reliés à des places ayant une capacité.

   Ces deux types d'arcs, ajoutés aux capacités de places, devraient fournir un ensemble suffisant pour spécifier tous les systèmes possibles avec un minimum de confort (voir [Bil88, CH93a, Lak93]).

2. Ce genre d'arc pose de nombreux problèmes intéressants pour les sémantiques d'ordre partiel (de la même manière que les places avec capacité). Cela sort des objectifs de cette thèse, mais on pourra consulter avec profit [Dev89].

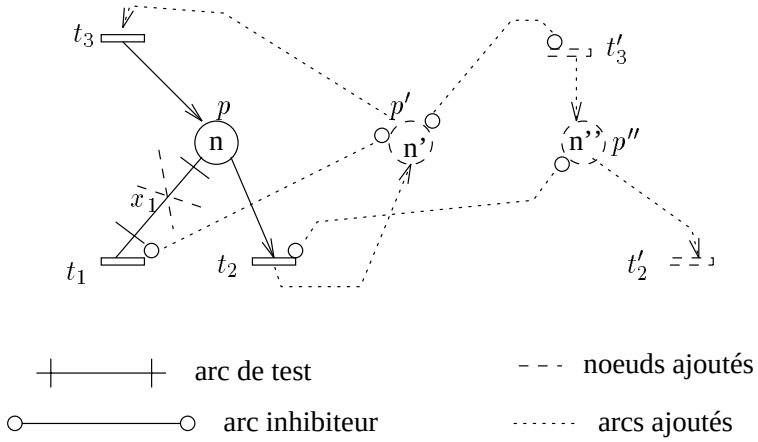
3. Si l'on s'abstrait de ces problèmes, on peut sans doute ramener l'étude des arcs de test à ceux des arcs inhibiteurs par la construction[2] de la figure 6.1, sous réserve que $p \notin {}^\bullet t_1$. Cette dernière réserve peut sans doute sauter si on remplace t_1 par un t_1'.

   p' est une place complémentaire classique et joue en même temps le rôle de capacité. p'' est une *place supplémentaire* destinée à stocker les jetons de p de $x+1$ à l'infini.

[1] La relation de priorité introduite par [JG88] et utilisée dans [DGV93] peut être vue comme un dérivé des arcs inhibiteurs.

[2] Cas des réseaux à arcs simples.

Figure 6.1 : Des arcs de test aux arcs inhibiteurs.

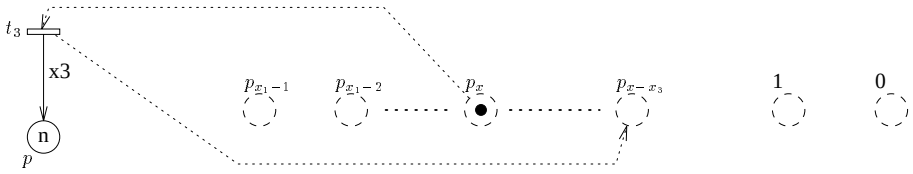


Les transitions t'_i ont les même pré- et post-conditions que leurs homologues t_i.

L'utilisation des transitions est résumée par le tableau suivant :

$M(p)$	$M(p')$	$M(p'')$	t_1	t_2	t_3	t'_2	t'_3
$< x$	> 0	0		$\times$	$\times$		
$= x$	0	0	$\times$	$\times$			$\times$
$= x$	0	≥ 1	$\times$			$\times$	$\times$

Dans le cadre des réseaux à poids, la solution est plus complexe, puisqu'elle doit prendre en compte les problèmes de répartition des jetons entre les différentes places. La figure ci-dessous donne une idée de ce qui pourrait se faire.



◇

6.1 Les réseaux à arcs inhibiteurs

Nous les présentons d'une manière analogue au réseaux de Petri classiques, en séparant la structure du comportement.

6.1.1 Structure des réseaux à arcs inhibiteurs

Puisque nous ne nous intéressons qu'aux systèmes étiquetés, nous définissons :

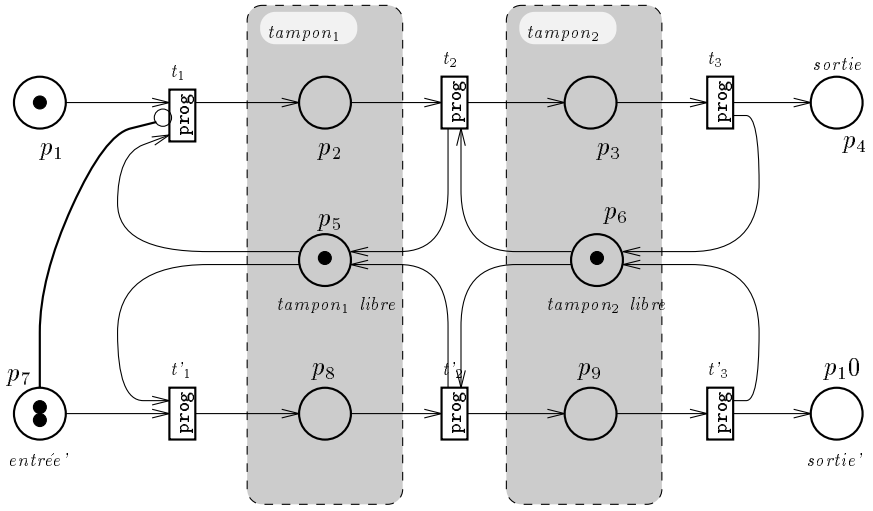
Définition 6.1.1. *Réseau de Petri à arc inhibiteurs*
Un réseau de Petri à arc inhibiteurs *est un quintuplet* $N = (P, T, W, l, I)$ *tel que :*

- (P, T, W, l) *soit un réseau de Petri au sens de la définition 1.3.2, p. 32 ;*
- $I \subseteq P \times T$ *est une fonction d'inhibition.*

De même, nous définissons un système (à arcs inhibiteurs) *par* $\Sigma = (N, M_0)$*, où* N *est un réseau de Petri à arc inhibiteurs et* M_0 *le marquage initial.*
Nous noterons $I(t), t \in T$ *l'ensemble des places inhibant* t *ou sa représentation vectorielle dans* $\{0,1\}^{|P|}$*.*

Graphiquement, un arc inhibiteur est représenté par une courbe dont l'extrémité (côté transition) est un cercle. En cela, nous retrouvons la notation usuelle de la négation.
Le réseau de la figure 6.2 montre un exemple de réseau à arcs inhibiteurs. Dans ce contexte, une des lignes est prioritaire sur l'autre. C'est typiquement un comportement qui n'est pas codable dans les réseaux classiques.

Figure 6.2 : Un réseau de Petri à arcs inhibiteurs.



(N, M)

Le fait principal va être la modification du comportement.

6.1.2 Comportement dans les réseaux à arcs inhibiteurs

Les arcs inhibiteurs n'allant que d'une place vers une transition, nous n'avons besoin d'étendre que la notion de tirabilité. Nous le faisons de la manière suivante :

Définition 6.1.2. *Tirabilité*

Étant donné un système $\Sigma = (N, M)$, nous dirons qu'une transition est tirable depuis M, *noté $M \xrightarrow{t}$, ssi ses pré-conditions sont remplies et qu'elle n'est pas inhibée, i.e. :*

$$^\bullet t \subseteq M \ \wedge \ I(t).M \stackrel{\text{déf}}{=} \sum_{p \in P} I(p,t).M(p) = 0$$

Nous l'étendons naturellement aux séquences.

Remarque sur le comportement concurrent : les arcs inhibiteurs posent des problèmes intéressants dans le cadre des sémantiques concurrentes. Par exemple, le réseau N_1 de la figure 6.4 peut tirer $t_1 \parallel t_2$, mais ni $t_1.t_2$, ni $t_2.t_1$. Par contre, le réseau N_2 peut tirer $t_1.t_2$. On se rend aisément compte des problèmes que poserait le calcul de la τp-bisimulation de places pour ces réseaux.

$\diamond$

6.2 La bisimulation de places stricte et les arcs inhibiteurs

Nous reprenons les définitions de la section 3.1, p. 76 et les étendons au cas des réseaux à arcs inhibiteurs. La bisimulation entre les marquages est définie de la même manière, nous aurons donc :

Définition 6.2.1. *Bisimulation de places stricte*

Soit $\Sigma = (N, M_0)$ un système et $B \subseteq P \times P$ une relation réflexive entre ses places. Nous dirons que B est une bisimulation de place (stricte) *ssi $\overline{B}$ est une bisimulation classique.*

Nous avons les propriétés suivantes (avec les mêmes preuves que pour le cas sans inhibiteurs) :

Proposition 6.2.2. *Étant donné un réseau $N = (P, T, W, l, I)$*

1. *Id_P est une bisimulation de place stricte sur N ;*
2. *si B, B' sont des bisimulations de place stricte sur N, alors $B \circ B'$ en est une aussi ;*

3. *si B est une bisimulation de place stricte sur N, alors $\widetilde{B}$ (la plus petite relation d'équivalence contenant B) est aussi une bisimulation de place stricte sur N ;*

4. *il existe une plus grande bisimulation de place stricte sur N que nous noterons $B(N)$. De plus, $B(N) = \bigcup\{B \ : \ B$ bisimulation de place stricte sur $N\}$.*

5. *$B(N)$ est une relation d'équivalence ;*

Nous avons maintenant les outils nécessaires pour définir le réseau quotient et déterminer une méthode de réduction.

6.3 Le réseau quotient

La méthode utilisée dans le cas des réseaux sans arcs inhibiteurs était de fusionner les places équivalentes, ou, ce qui revient au même, de déplacer tous les arcs sur un représentant de la classe puis d'éliminer toutes les places isolées du réseau.

La figure 6.3 montre que cette technique n'est plus utilisable dans le cas des réseaux à arcs inhibiteurs. En effet, si nous fusionnons p_1 et p_2, aucune transition n'est tirable dans le système réduit. Le théorème du quotient correct (théo. 2.2.8, p. 59) ne peut donc être généralisé si nous conservons cette définition du réseau quotient (déf. 2.2.5, p. 57).

La solution consiste à choisir un représentant de la classe d'équivalence et à ne conserver que les arcs inhibiteurs en partant. Nous pouvons alors généraliser le théorème 2.2.8. Formellement :

Définition 6.3.1. *Réseau quotient en présence d'arcs inhibiteurs*

Étant donné un réseau $N = (P, T, W, l, I)$ et une relation d'équivalence $E \subseteq P \times P$, nous notons $[p]_E$ la classe d'équivalence d'une place $p \in P$ et une fonction de choix *$c : P \to P$ telle que $pEp' \Leftrightarrow c(p) = c(p')$.*

Nous définissons le réseau quotient *$N/_E = (P/_E, T/_E, W/_E, l/_E, I/_E)$ de N par E de la manière suivante :*

- $P/_E \stackrel{\text{déf}}{=} \{c(p) \ : \ p \in P\}$
- $T/_E \stackrel{\text{déf}}{=} T$
- $\begin{cases} W/_E(c(p), t) \stackrel{\text{déf}}{=} \sum_{u \in [p]_E} W(u, t) \\ W/_E(t, c(p)) \stackrel{\text{déf}}{=} \sum_{u \in [p]_E} W(t, u) \end{cases}$
- $I/_E(c(p), t) \stackrel{\text{déf}}{=} I(c(p), t)$
- $l/_E \stackrel{\text{déf}}{=} l$

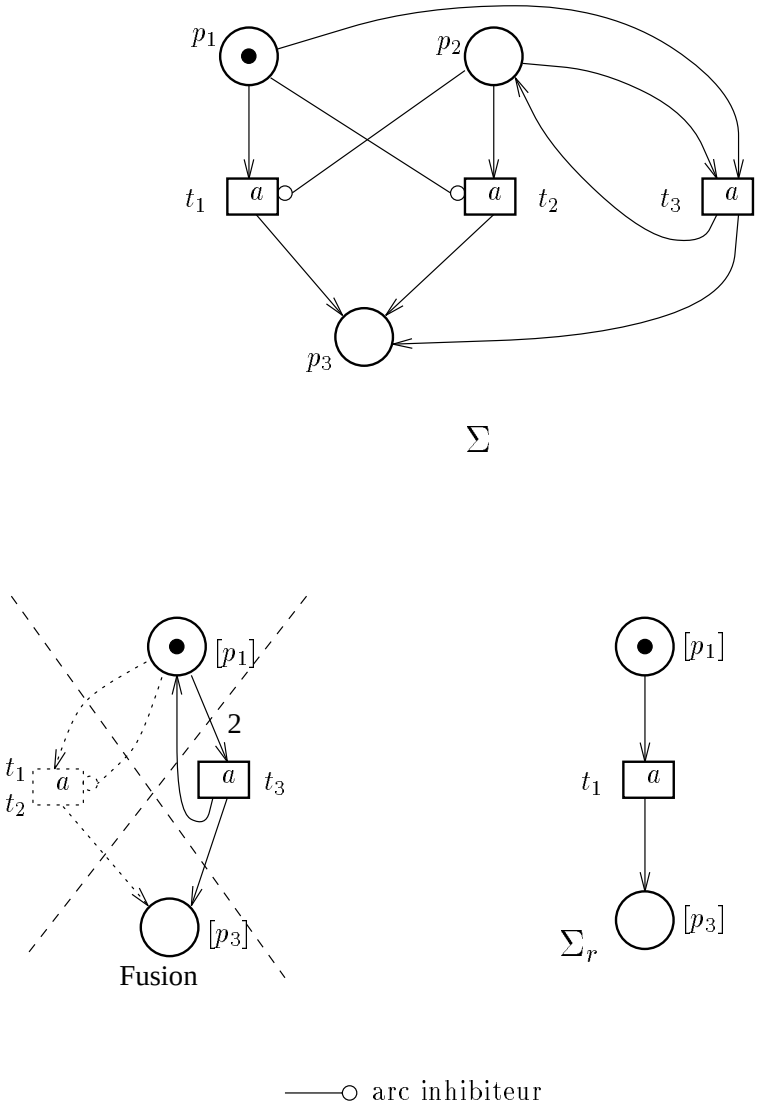


Figure 6.3 : La définition 2.2.5 est inutilisable en présence d'inhibiteurs.

Pour quotienter les marquages :

$$M/_E \stackrel{\text{déf}}{=} [M]_E \stackrel{\text{déf}}{=} [\{p_1, p_2, \ldots, p_n\}]_E \stackrel{\text{déf}}{=} \{c(p_1), c(p_2), \ldots, c(p_n)\}$$

Remarques sur la définition du réseau quotient :

1. Ceci implique que pour un même réseau, nous avons autant de manières de quotienter que de fonctions de choix.
2. Si I est vide, nous retrouvons la définition 2.2.5, p. 57.
3. Les arcs de I ne sont pas traités de la même manière que ceux de W, c'est ce qui nous permettra de quotienter correctement.

◇

Nous avons alors le théorème suivant :

Théorème 6.3.2. *Quotient correct pour les inhibiteurs*
Soit N un réseau et E une relation d'équivalence sur ses places telle que $\overline{E}$ soit une bisimulation. Alors pour tout marquage $M \in \mathcal{M}_N$ nous avons :

$$(N/_E, M/_E) \textit{ et } (N, M) \textit{ sont bisimilaires}$$

La preuve de ce théorème repose essentiellement sur les adaptions du lemme technique établissant le lien entre les pas du réseau et ceux du réseau quotienté (lemme 2.2.6, p. 58). Nous donnons tout d'abord son équivalent dans le cadre des réseaux à arcs inhibiteurs :

Lemme 6.3.3. *Soit N un réseau à arcs inhibiteurs, $E \subseteq P \times P$ une relation d'équivalence entre ses places t.q. $\overline{E}$ soit une bisimulation. Soit $N/_E$ le réseau quotient tel que dans la définition 6.3.1, nous avons :*

1. *Pour tout pas $M \stackrel{t}{\rightarrow} M'$ dans N, il existe un pas $[M]_E \xrightarrow{t':l(t)} [M']_E$ dans $N/_E$.*
2. *Pour tout pas $[M]_E \stackrel{t'}{\rightarrow} [M']_E$ dans $N/_E$, il existe un pas $M \stackrel{t}{\rightarrow} M''$ avec $M''\overline{E}M'$ dans N.*

Preuve.

- Pour prouver 1, nous utilisons le fait que $\overline{E}$ est une bisimulation. De $M\overline{E}[M]_E$ et $M'\overline{E}[M']_E$, la propriété de transfert assure qu'il existe un pas $[M]_E \xrightarrow{t':l(t)} [M']_E$.

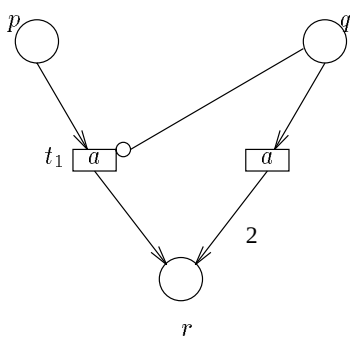
- Pour (2), nous pouvons reprendre la technique utilisée au cas précédent. Considérons un pas $[M]_E \xrightarrow{t'} [M']_E$ dans $N/_E$. $\overline{E}$ étant une bisimulation nous avons un t tel que :

$$\begin{array}{ccc} [M]_E & \overline{E} & M \\ {\scriptstyle t'}\downarrow & & \downarrow{\scriptstyle t:l(t')} \\ [M']_E & \overline{E} & M'' \end{array}$$

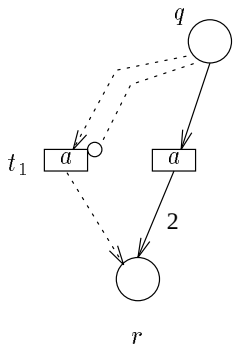
avec $M''\overline{E}M'$

Remarque : la condition $\overline{E} \subseteq \underline{\leftrightarrow}$ ne suffit pas dans ce cas. En effet, $M \xrightarrow{t} M'$ n'entraîne pas $[M]_E \xrightarrow{t} [M']_E$, comme le montre l'exemple ci-dessous. Nous avons pEq et $\overline{E} \subseteq \underline{\leftrightarrow}$. Cependant $\overline{E}$ n'est pas une bisimulation car :

$$\begin{array}{ccc} \{p\} & \overline{E} & \{q\} \\ {\scriptstyle t_1:a}\downarrow & & \downarrow{\scriptstyle t_2:a} \\ \{r\} & \overline{E} & \{r,r\} \end{array}$$



N



N_r où $c(p) = c(q) = q$

◇

□

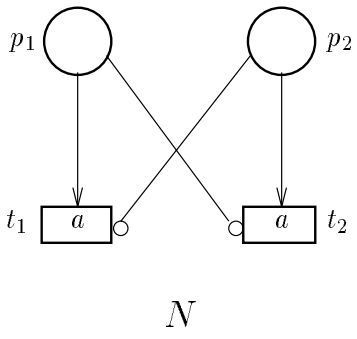
Nous avons maintenant l'outil nécessaire pour prouver le théorème 6.3.2. La preuve est en tout point identique à celle développée pour les réseaux classiques (voir page 59).

Nous pouvons alors réduire en éliminant des transitions redondantes et celles structurellement mortes.

6.4 Calculabilité de $B(N)$

Il reste maintenant à calculer cette plus grande relation d'équivalence de manière efficace. Malheureusement, le théorème de caractérisation (théo.

3.4.2, p. 83) ne peut être généralisé, comme le montre l'exemple de la figure 6.4.



N

Figure 6.4 : Le théorème 3.4.2 n'est plus valable.

Dans cet exemple, $B = P \times P$ satisfait trivialement aux conditions du théorème (réflexivité, symétrie et transfert local). Cependant, alors que le marquage $\{p_1, p_1\}$ permet de tirer t_1, $\{p_1, p_2\}$ inhibe toute transition.

Intuitivement, imiter dans un marquage donné ne suffit plus puisqu'un jeton peut inhiber la transition imitatrice. De plus, le transfert local ne se préoccupe que des arcs classiques, or il est évident qu'un jeton peut devenir inhibiteur en passant d'une place à l'autre.

6.4.1 Propriété de transfert local généralisé

Il est facile de voir que la propriété du transfert local (prop. 3.4.1, p. 82) ne relie les places du réseau qu'en fonction de ce qu'elles autorisent. Ces difficultés peuvent être contournées en prenant soin de conserver récursivement la valeur «sémantique» d'un jeton. Formellement, nous généralisons la propriété comme suit :

Définition 6.4.1. *Propriété de transfert local généralisée (PTLG)*
Soit un réseau $N = (P, T, W, I, l)$ et une relation $B \subseteq P \times P$.

- *Nous dirons que $M_1 + r \xrightarrow{t_1} M_1'$* est imitable depuis *$M_1 + s$* relativement à *$B$ ssi il existe une transition $t_2 \in T$ de même étiquette que t_1 t.q. $M_1 + s \xrightarrow{t_2} M_2'$ $\overline{B}$ M_1' et si $\forall i \in I(t_2)$[3], $M_1 + i + r \xrightarrow{t_1} M_1' + i$ est imitable à partir de $M_1 + i + s$ relativement à B.*

- *B a la* propriété de transfert local généralisée *ssi pour toute transition $t_1 \in T$, pour toute place $p \in {}^\circ t_1$ et toute place $q \in P$ avec q B p:*

  1. *${}^\bullet t_1 \xrightarrow{t_1} t_1{}^\bullet$ est imitable depuis ${}^\bullet t_1 - p + q$ relativement à B ;*

[3] Il est évident que si t_1 est inhibée, il n'y a plus lieu de l'imiter.

2. $\forall t_2 \in T \ : \ (p \notin I(t_2) \wedge q \in I(t_2)),\ {}^\bullet t_2 + p \xrightarrow{t_2} t_2{}^\bullet + p$ *est imitable depuis* ${}^\bullet t_2 + q$ *relativement à* B *;*
3. *symétriquement,* $\forall t_2 \in T \ : \ (p \in I(t_2) \wedge q \notin I(t_2)),\ {}^\bullet t_2 + q \xrightarrow{t_2} t_2{}^\bullet + q$ *est imitable depuis* ${}^\bullet t_2 + p$ *relativement à* B*.*

Il est facile de voir que si I est vide (cas des réseaux classiques), nous retrouvons la propriété de transfert local (déf. 3.4.1, p. 82).

Remarque sur la notion «est imitable» : clairement, cette notion est récursive. Cependant, il est facile de voir qu'a chaque appel, une nouvelle place est introduite (i.e. $i \notin M_1$, puisque $i \in I(t_2)$ et $M_1 + s \xrightarrow{t_2}$). Ceci nous assure que la récursion s'arrête dès lors que P est fini.

◇

Nous présentons maintenant la caractérisation des bisimulations de places dans le cadre des réseaux à arcs inhibiteurs :

Théorème 6.4.2. *Caractérisation des BdPS en présence d'arcs inhibiteurs*

Étant donné un réseau $N = (P, T, W, l, I)$ *et* $B \subseteq P \times P$ *une relation réflexive et symétrique vérifiant la propriété de transfert local généralisée, alors* B^* *(sa fermeture transitive) est une bisimulation de place stricte.*

Preuve. Il nous suffit de prouver que $\overline{B^*} = \overline{B}^*$ (lemme 3.1.3, p. 76) est une bisimulation classique, c'est-à-dire qu'elle vérifie la propriété de transfert. $\overline{B}^*$ étant symétrique, nous n'avons à prouver qu'un sens de la propriété de transfert.

Considérons deux marquages $M_1 \overline{B^*} M_2$ et un pas $M_1 \xrightarrow{t_1} M_1'$. Plusieurs cas se présentent :

1. si $M_1 \overline{B} M_2$:
   - Supposons que $M_2 = M_1 - p + q$ avec pBq.
     - Si $M_1 = {}^\bullet t_1$, le transfert local généralisé nous permet de dire qu'il existe un pas $M_2 \xrightarrow{t_2} M_2'$, avec $M_1' \overline{B} M_2'$ et $l(t_1) = l(t_2)$.
     - Sinon, M_1 est de la forme ${}^\bullet t_1 + M_1''$.
       * si $p \in M_1''$, soit $I(t_1) \cap M_1'' = \emptyset$ et $M_2 \xrightarrow{t_1} M_1' - p + q$ imite $M_1 \xrightarrow{t_1} M_1'$, soit $\exists r \in I(t_1) \cap M_1''$. Dans ce cas, la PTLG, par récursivité, nous donne un pas ${}^\bullet t_1 - p + q + r \xrightarrow{u_2} M'^2$, etc jusqu'à $M_2 \xrightarrow{u_k} M_2'$, avec $M_1' \overline{B} M_2'$ et $l(t_1) = l(u_k)$.
       * si $p \in {}^\bullet t_1$, alors par la PTLG il existe un pas ${}^\bullet t_1 - p + q \xrightarrow{u_1} M'$, avec $l(t_1) = l(u_1)$. En l'appliquant récursivement comme ci-avant, nous obtenons un pas $M_2 \xrightarrow{u_k} M_2'$, avec $M_1' \overline{B} M_2'$ et $l(t_1) = l(u_k)$.

- Sinon, il existe un ensemble de marquages tels que

$$M_1 = M^0 \overline{B} M^1 \overline{B} \ldots \overline{B} M^k = M_2,$$

avec $M^i = M^{i-1} - p_i + q_i$ et $p_i B q_i$, $i = 1, \ldots, k$. Dans ce cas, $M^0 \stackrel{t_1}{\rightarrow} M'_1$ peut être imité par un $M^1 \stackrel{u_1}{\rightarrow} M'^1$, et l'imitation peut se propager jusqu'à $M_2 = M^k \stackrel{u_k}{\rightarrow} M'^k$. L'imitation se fait avec $t_2 = u_k$, et $M'_2 = M'^k$. Comme $M'_1 \overline{B} M'^1 \overline{B} \ldots \overline{B} M'^k$, nous avons $M'_1 \overline{B^*} M'_2$, soit $M'_1 \overline{B^*} M'_2$ puisque B est réflexive (lemme 3.1.3).

$$\begin{array}{ccccc} M^0 & \overline{B} & M^1 & \cdots & M^k \\ {\scriptstyle t_1}\downarrow & & \downarrow{\scriptstyle u_2:l(t_1)} & & \downarrow{\scriptstyle u_k:l(t_1)} \\ M'^0 & \overline{B} & M'^1 & \cdots & M'^k \end{array}$$

soit

$$\begin{array}{ccc} M_1 & \overline{B^*} & M_2 \\ {\scriptstyle t_1}\downarrow & & \downarrow{\scriptstyle u_k:l(t_1)} \\ M'_1 & \overline{B^*} & M'_2 \end{array}$$

2. Dans le cas général, si $M_1 \overline{B^*} M_2$, il est possible de décomposer en une séquence $M_1 = M^1 \overline{B} M^2 \ldots \overline{B} M^k = M_2$, et d'utiliser le résultat ci-dessus pour construire une imitation $M_2 \stackrel{u}{\rightarrow} M'_2$.

□

Nous avons donc le résultat suivant (prouvé dans la section 6.5) :

Théorème 6.4.3. *Calculabilité de $B(N)$ en présence d'inhibiteurs*
$B(N)$ est calculable en temps exponentiel (en la taille de P).

L'algorithme est modifié en remplaçant le test par «*Tester si B a la propriété de transfert local généralisée*». Soit :

Algorithme 6.4.1. *Calcul de $B(N)$ en présence d'inhibiteurs*
Soit le réseau $N = (P, T, W, l)$:

1. $B = P \times P$.
2. *Tester si B a la propriété de transfert local généralisée, i.e. :*
   - *s'il existe $t \in T$, $p \in {}^\bullet t$, $q \in P$ avec pBq tels qu'un des points de la définition 6.4.1 n'est pas vérifié, alors éliminer les paires (p,q) et (q,p) de B et recommencer l'étape 2.*
   - *sinon, B a la propriété de transfert local généralisée.*
3. *B est la plus grande bisimulation de place stricte, i.e. $B = B(N)$.*

Cette algorithme est correct pour les mêmes raisons que celui calculant $B(N)$ pour un réseau sans arcs inhibiteurs (algo. 3.5.1, p. 84).
La section suivante montre qu'il termine et en donne le coût théorique.

6.5 Coût théorique du calcul de $B(N)$

À nouveau, le coût algorithmique ne nécessite pas une réévaluation complète, puisque seule la fonction de recherche est modifiée (les transitions structurellement mortes peuvent être détectées en même temps que les transitions redondantes).

Théorème 6.5.1. *Le coût de la réduction est alors en :*

$$\mathcal{O}(n_T^{n_P+3}.n_P^3)$$

pour les familles de réseaux où les degrés sont bornés par une constante.

Preuve. La structure de l'algorithme est inchangée jusqu'à la recherche d'une imitation. Pour imiter le pas $M_1+r \xrightarrow{t} M_1'$ depuis M_1+s relativement à B, nous codons la fonction :

fonction est-imitable (M_1+r, t_1, M_1+s,B) $\triangleright$ un booléen
  t_2-ok : un booléen
  Pour t_2 parcourant T-$\{t_1\}$
    Si l(t_1)=l(t_2)
      et est-tirable (t_2, M_1+s)
      et sont-reliés (B, $t_1^\bullet$, $M_1+s-{}^\bullet t_2+t_2^\bullet$)
    Alors t_2-ok $\lhd$ vrai
      Pour i parcourant $I(t_2)$
        t_2-ok $\lhd$ t_2-ok $\wedge$ est-imitable (M_1+i+r, t_1, M_1+i+s,B)
        {*Les supports vérifient :* $P_{M_1+r} \subsetneq P_{M_1+i+r}$}
      fpour $\{i\}$
    Si t_2-ok
    Alors retourne vrai
  fpour $\{t_2\}$
  $\{M_1+r \xrightarrow{t_1}$ *n'est pas imitable depuis* M_1+s *relativement à* B$\}$
  Retourne faux

Seul le parcours des places inhibitrices diffère du cas sans inhibiteurs. L'appel récursif de est-imitable (...) peut se produire au plus n_P fois, puisque nous avons trivialement $I(t_2) \cap (M_1+r) = \emptyset$. Ceci nous garantit la terminaison.

Nous introduisons $d_i = max_{t \in T}\{|I(t)|\}$. Nous avons donc au plus d_i places dans $I(t_2)$. Ce facteur n'apparaît pas compte-tenu de notre hypothèse sur les degrés.

Le coût de l'appel initial de cette fonction est donc en $\mathcal{O}(n_T^{n_P})$.

Le reste du calcul du coût est inchangé.

□

Conclusion

Pour les réseaux à arcs inhibiteurs, l'adaptation de la bisimulation de places est donc non triviale. De fait, les résultats sont un peu moins bons, puisqu'ils sont en temps exponentiel. Intuitivement, nous pourrions «croiser» la τ-bisimulation avec les réseaux à arcs inhibiteurs. Les problèmes posés auraient les mêmes solutions : très informellement, il suffirait d'ajouter la condition $I(t) \subseteq I(t')$ à la définition de $t \sqsubseteq t'$ pour conserver les mêmes propriétés.

À notre connaissance, la bisimulation de places est la seule méthode de réduction pour les réseaux à arcs inhibiteurs.

Plus généralement, les problèmes posés par les extensions des réseaux de Petri posent deux problèmes distincts :

1. l'adaptation de la notion de bisimilarité. C'est par exemple récemment abordé par [Buc95] pour les réseaux stochastiques.

2. la correction du quotient.

Les extensions *syntaxiques*, i.e. où la «décoration» a pour seul objectif de donner un modèle plus compact, ont une faible incidence (p.ex. pour certaines définitions des réseaux colorés[4], il n'y a qu'une augmentation de la complexité).

Les extensions *sémantiques*, qui visent à accroître la puissance du modèle (réseaux avec arcs inhibiteurs) ou à intégrer d'autres paramètres (réseaux temporisés de [Zub80] ou stochastiques de [Ajm89]) ont un impact beaucoup plus fort puisqu'ils peuvent apporter de profondes modifications pour la définition de la bisimulation.

[4]Voir p.ex. [Jen92, Jen95].

Troisième partie

Les bisimulations de places en pratique

Chapitre 7

Bisimulation de places et comportements

Nous savons que la bisimulation (qui est notre critère de correction) conserve de nombreuses propriétés de vivacité et de sécurité. Ces propriétés sont définies *dynamiquement*, i.e. elles s'appuient sur les comportements du réseau. Mais nous avons vu que la bisimulation de places préservait parfois plus (théo. 3.8.4, p. 92). L'objectif de ce chapitre est donc de voir de quelle manière la BdPS se mêle aux théories développées par ailleurs.

La vérification de ces propriétés étant très complexe, certains auteurs ont naturellement cherché à les caractériser par des contraintes structurelles, moins coûteuses car ne s'appuyant que sur la structure du réseau ([VV81, Yam84, BD90]). La première section présente diverses définitions et étudie l'impact de notre méthode de réduction.

Ces propriétés, comportementales ou structurelles, ont permis de définir diverses familles de réseaux. De nombreux outils utilisent les propriétés de ces familles pour optimiser leurs calculs. Dans le contexte d'un interfaçage de la bisimulation de places, il est intéressant de savoir si ces familles sont stables. La deuxième section aborde ce point.

Nous donnons en conclusion un tableau récapitulant ces divers résultats.

7.1 Les propriétés des réseaux de Petri

Il existe de nombreuses notions, nous nous attacherons à celles qui sont le plus utilisées. [EN94], par exemple, liste un certain nombre de propriétés avec la complexité de leur vérification. [Hac76, Ber83] en donnent aussi plusieurs autres.

7.1.1 Propriétés comportementales

Il s'agit donc de propriétés qui, pour être vérifiées, demandent que l'on étudie l'ensemble des comportements du réseau, i.e. son graphe d'état. La bisimulation de places est un moyen efficace de réduire ce graphe, il est donc important de savoir sous quelles conditions elle préserve ou non ces propriétés.

Définition 7.1.1. *Vivant, pseudo-vivant*

Soit M un marquage sur un réseau N, nous dirons que M est vivant *si toute transition t est atteignable depuis M.*

Nous dirons que M est pseudo-vivant *s'il permet de tirer au moins une transition.*

Soit (N, M) un système, nous dirons qu'il est (pseudo-)vivant *si $\forall M' \in \mathcal{R}(N)$, M' l'est.*

Intuitivement, un système est vivant si n'importe quelle transition est toujours atteignable. Il est facile de voir qu'un système vivant est pseudo-vivant, i.e. sans blocage.

Définition 7.1.2. *Borné, sauf*

Soit (N, M_0) un système, nous dirons qu'il est n-borné *si il existe $n \in \mathbb{N}$ tel que :*

$$\forall M \in \mathcal{R}(M_0),\ |M(p)| \leq n$$

Nous dirons qu'un système est sauf *ssi il est 1-borné.*

Il est facile de voir que pour un système borné $\mathcal{R}(M_0)$ est fini et que $\forall M \in \mathcal{R}(M_0),\ M_0 \not\subset M$ ($M = M_0$ est possible).

Ces réseaux sont de ce fait souvent utilisés. Ils sont utilisés comme systèmes «condition/événement» (voir p.ex. [DDM88]). En particulier, l'ensemble des marquages atteignables étant fini, presque toutes les propriétés deviennent décidables. En particulier, le langage LOTOS permet de produire des réseaux saufs (voir [GS90]).

Nous avons trivialement, comme conséquence du théorème 3.3.6, p. 81 :

Proposition 7.1.3. *Soit Σ un système et Σ_r sa réduction.*

Σ est vivant (resp. pseudo-vivant, n-borné) ssi Σ_r est vivant (resp. pseudo-vivant, n'-borné, $n' \leq n.\max_{p \in P}\{|[p]|\}$).

7.1.2 Les invariants

Ce sont sans doute parmi les plus utilisés des méthodes d'analyse.

Informellement, un P-invariant d'un système est une propriété vraie quelque soit le marquage atteint. Il permet de vérifier un grand nombre de propriété dynamiques (vivant, borné, atteignable, etc). Dans l'exemple

de la file FIFO (1.3, p. 34), nous n'aurons jamais un tampon occupé avec un jeton dans la place *tampon libre* correspondante.

Un T-invariant correspond à des séquences «nulles», i.e. telles que ${}^\bullet\sigma = \sigma^\bullet$. Formellement :

Définition 7.1.4. *Invariants*

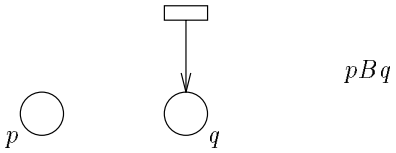
Étant donné un réseau $N = (P, T, W)$, nous notons sa matrice d'incidence $\mathbf{N} \in \mathbb{Z}^P \times \mathbb{Z}^T$, *où* $\mathbf{N}(p,t) \stackrel{\text{déf}}{=} W(t,p) - W(p,t)$.

Nous appelons P-invariant *(resp. T-invariant) une solution de :*

$$X.\mathbf{N} = \mathbf{0}$$

(resp. $\mathbf{N}.Y = \mathbf{0}$*).*

Remarque : il n'y a aucun lien entre P-invariants et places bisimilaires comme le montre très simplement la figure ci-dessous, où tout vecteur tel que $X(q) = 0$ est un P-invariant de N, mais où aucun n'est un P-invariant de N' (dans lequel p et q sont fusionnées) :



◇

Regardons l'impact de la méthode de réduction sur la matrice d'incidence : la fusion de places va correspondre à l'addition de lignes et la suppression des transitions redondantes au retrait de colonnes.

Pour la figure 7.1, nous avons :

$$\mathbf{N} = \begin{pmatrix} -1 & +1 \\ 0 & 0 \\ +1 & -1 \end{pmatrix}$$

Les invariants sont donc de la forme :

$$X = \begin{pmatrix} x & y & x \end{pmatrix} \qquad Y = \begin{pmatrix} x \\ x \end{pmatrix}$$

Après fusion de p_1 et p_3 puis élagage, nous obtenons respectivement :

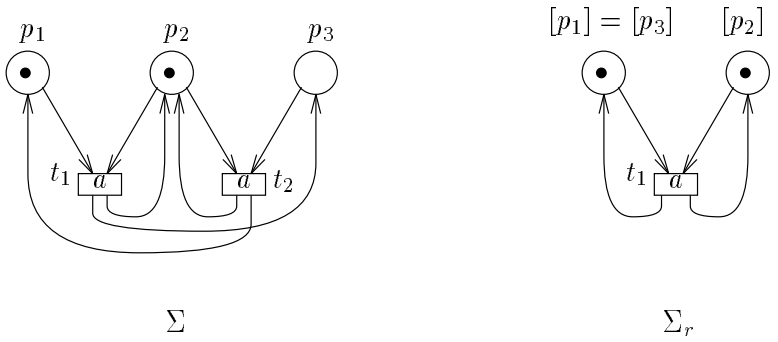
$$\mathbf{N}' = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \qquad \mathbf{N}_r = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Les invariants de N_r sont alors de la forme :

$$X = \begin{pmatrix} x & x \end{pmatrix} \qquad Y = \begin{pmatrix} x \end{pmatrix}$$

Nous avons les résultats suivants :

Figure 7.1 : Comment lier les invariants de Σ et Σ_r ?



Σ Σ_r

Proposition 7.1.5. *Soit N un réseau, N' le réseau où les places sont fusionnées et N_r le réseau réduit :*

1. *La fusion de places préserve les T-invariants.*
2. *L'élagage préserve les P-invariants.*
3. *Soit X_r est un P-invariant de N_r, alors le vecteur :*

$$X : X(p) \stackrel{\text{déf}}{=} X_r(p)$$

   est un P-invariant de N.

4. *Soit Y est un T-invariant de N, alors le vecteur :*

$$Y_r(t) \stackrel{\text{déf}}{=} \sum_{t' \in lag(t)} Y(t)$$

   est un T-invariant de N_r.

Preuve.

- Le point 1 vient de ce que la fusion des places revient à additionner les bilans $t^\bullet(p) - {}^\bullet t(p)$ de chaque transition pour chaque place, i.e. :

$$\mathbf{N}_r(t,[p]) = \sum_{q \in [p]} \mathbf{N}(t,q)$$

  d'où :

$$X_r.(\mathbf{N}_r(t)) = \sum_{[p]} \left(X_r([p]).\sum_{q \in [p]} \mathbf{N}(t,q) \right) = \sum p \in PX_r([p]).\mathbf{N}(t,q)$$

- pour 2, il suffit de noter que la fusion de places correspond à une addition de lignes :

$$\mathbf{N}.Y = 0 \rightarrow \forall P' \subseteq P \ : \ \sum_{p \in P'} (\mathbf{N}(p)).Y = 0$$

- Les deux dernières propriétés découlent de celles ci-dessus.

□

7.1.3 Semi-linéarité de $\mathcal{R}(\Sigma)$

Nous avons les définitions suivantes :

Définition 7.1.6. *Semi-linéaire*

Un sous-ensemble de $\mathbb{N}^n$ *est* linéaire *s'il est de la forme :*

$$\{u + \sum_{i=1}^{p} n_i v_i, \ n_i \in \mathbb{N}\}$$

où $u, v_1, \ldots, v_p$ *sont des vecteurs de* $\mathbb{N}^n$.

Un sous-ensemble de $\mathbb{N}^n$ *est un* semi-linéaire *s'il est une union finie d'ensembles linéaires.*

La semi-linéarité de l'ensemble de marquages atteignables rend un grand nombre de propriétés décidables, particulièrement celles de l'inclusion et de l'égalité des ensembles de marquages atteignables. [Hau90] montre que la semi-linéarité de $\mathcal{R}(\Sigma)$ est décidable.

Le théorème 3.3.6, p. 81 nous donne trivialement :

Proposition 7.1.7. *$\mathcal{R}(\Sigma)$ est semi-linéaire ssi $\mathcal{R}(\Sigma_r)$ l'est.*

7.1.4 Régularité du langage

Le langage du réseau est l'ensemble $L(N) \stackrel{déf}{=} \{\rho \in T^* : M_0 \stackrel{\rho}{\rightarrow}\}$. [VV81] donne une caractérisation structurelle de la régularité du langage d'un réseau.

La régularité du langage rend un grand nombre de propriétés plus facilement décidables (atteignabilité, vivacité, inclusion des ensembles de marquages atteignables).

Dans le cadre de la bisimulation de places, le graphe des marquages atteignables de Σ_r est obtenu par un «repliage» correspondant à la fusion des places, avec la suppression des arcs correspondant aux transitions redondantes. Nous avons :

Proposition 7.1.8. *Si $L(N)$ est régulier alors $L(N_r)$ l'est.*

Remarque sur le langage d'actions : Si nous notons $LA(N) \stackrel{déf}{=} \{l(\rho), \rho \in T^* : M_0 \stackrel{\rho}{\rightarrow}\}$ le langage d'actions, nous avons trivialement $LA(N) = LA(N_r)$.

◇

7.2 Quelques grandes familles de réseaux

Les réseaux de Petri forment un cadre très large pour l'étude du parallélisme. Ainsi, pour s'attacher à certaines propriétés, un certain nombre de familles ont été définies. Nous présentons ici les plus importantes ou celles qui nous serons utiles. [EN94], p.ex. en cite quelques autres.

Plusieurs de ces classes sont basées sur des propriétés structurelles (réseaux à choix libre [Hac72], réseaux sans conflits de [LR78], réseaux saufs, etc.

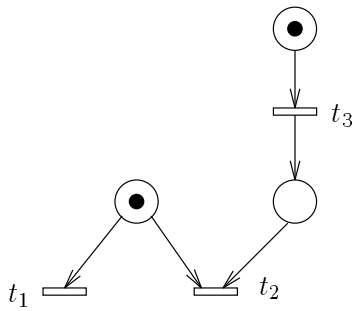
7.2.1 Les réseaux à choix libres

Les réseaux de Petri ont un grand pouvoir d'expression. Malheureusement, un grand nombre d'algorithmes d'analyse de leur comportement ont des bornes de complexité élevées, voire ne terminent pas forcément. Une théorie liant la structure et le comportement d'un réseau donné a donc une portée pratique très limitée.

Ces obstacles peuvent être contournés en se restreignant à certaines classes de réseaux, avec certaines contraintes sur leur structure.

Une des classes «limites» est celle des réseaux à choix libres. Celle-ci autorise la synchronisation et le conflit[1] de manière qu'ils n'interfèrent pas. Typiquement, dans la séquence $t_1.t_3$, il n'apparaît pas explicitement de conflit. Par contre, si t_3 est tirée en premier, alors nous aurons un conflit entre t_1 et t_2. Grossièrement, à cause de la synchronisation sur t_2, la transition t_3 influe sur l'occurrence de t_1 ou t_2.

Figure 7.2 : Un réseau où conflit et synchronisation interfèrent.



Ces réseaux à choix libres (FC) ont été étendus de deux manières :

1. en assouplissant les conditions structurelles, ce qui a donné les réseaux à choix libres *étendus* (EFC) ;

[1] Deux transitions t_1, t_2 sont en conflit s'il existe un marquage M avec $M \xrightarrow{t_1} M_1$ et $M \xrightarrow{t_2} M_2$, mais tel que $M_1 \not\xrightarrow{t_2}$ ou $M_2 \not\xrightarrow{t_1}$, i.e. le tir d'une transition empêche le tir de l'autre.

2. en les définissant sur les réseaux de Petri à poids : les réseaux à *conflits équilibrés* (ECS).

Définition 7.2.1. *Réseaux à choix libres [Hac72, BD90, TS93]*
Un réseau à arcs simples $N = (P, T, A)$ est un FC-réseau ssi :

$$\forall p \in P, \forall t \in T \ : \ A(p,t) > 0 \Rightarrow (p^\circ = \{t\}) \vee ({}^\circ t = \{p\})$$

Un réseau à arcs simples $N = (P, T, A)$ est un EFC-réseau ssi :

$$\forall t, t' \in T \ : \ {}^\circ t \cap {}^\circ t' \neq \emptyset \Rightarrow {}^\circ t = {}^\circ t'$$

Un réseau à poids $N = (P, T, W)$ est un ECS ssi :

$$\forall t, t' \in T \ : \ {}^\bullet t \cap {}^\bullet t' \neq \emptyset \Rightarrow {}^\bullet t = {}^\bullet t'$$

Dans ces réseaux si un marquage autorise une transition en sortie d'une place p, il autorise toutes les transitions en sortie de p.

Proposition 7.2.2. *Les familles FC, EFC et ECS sont stables pour notre réduction.*

Preuve. Nous travaillons avec les définitions des EFC et ECS, la preuve s'étend aisément au cas des FC.

Supposons que N soit un ECS et montrons que N_r en est alors un aussi, i.e. :

$$ {}^\bullet[t] \cap {}^\bullet[t'] \neq \emptyset \Rightarrow {}^\bullet[t] = {}^\bullet[t'] $$

Nous avons trivialement :

$$ {}^\bullet t = {}^\bullet t' \ \Rightarrow \ [{}^\bullet t] = [{}^\bullet t'] \tag{7.1} $$
$$ {}^\bullet[t] \cap {}^\bullet[t'] = \emptyset \ \Rightarrow \ {}^\bullet t \cap {}^\bullet t' = \emptyset \tag{7.2} $$

- Nous supposerons dans un premier temps, que T *ne contient pas de transitions de précondition vide.* Nous étudierons ce cas en fin de démonstration.

  Soient $[{}^\bullet t_1] \cap [{}^\bullet t_2] \neq \emptyset$.

  1. Si ${}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset$, comme N est un EFC, nous avons ${}^\bullet t_1 = {}^\bullet t_2$ et par (7.1), ${}^\bullet t = {}^\bullet t'$.
  2. Supposons ${}^\bullet t_1 \cap {}^\bullet t_2 = \emptyset$. Nous avons ${}^\bullet t_1 = M_1 + p_1$ et ${}^\bullet t_2 = M_2 + p_2$, avec $p_1 \, B \, p_2$. Par la propriété de transfert, nous savons qu'il existe un pas $M_1 + p_2 \stackrel{t'_1}{\rightarrow}$:
     (a) Si $M_1 \cap {}^\bullet t'_1 \neq \emptyset$, alors nous avons ${}^\bullet t_1 = {}^\bullet t'_1$ ce qui est contradictoire avec $M_1 + p_2 \stackrel{t'_1}{\rightarrow}$.

(b) Donc ${}^\bullet t'_1 = p_2$, puisque $\forall t, {}^\bullet t \neq \emptyset$. N étant un ECS, nous avons ${}^\bullet t'_1 = {}^\bullet t_2 = p_2$. Alors, nous imitons t'_1 à partir de p_1 par un pas $p_1 \xrightarrow{t''_1}$ et, N étant un ECS, ${}^\bullet t_1 = {}^\bullet t''_1 = p_1$ et par suite $[{}^\bullet t_1] = [{}^\bullet t_2]$.

- Il nous reste à traiter le cas des transitions à pré-conditions vides. Seul le cas 2b est modifié par un sous-cas supplémentaire : ${}^\bullet t'_1 = \emptyset$.

  Nous pouvons supposer que :

$$p_1{}^\circ = \{t_1\} \tag{7.3}$$
$$p_2{}^\circ = \{t_2\} \tag{7.4}$$
$$l(t_1) = l(t_2) \tag{7.5}$$

  (7.5) est une simple conséquence de $p_1 \, B \, p_2$. Si nous combinons cela avec le fait que N est un ECS, alors toutes les transitions consommant des jetons de p_1 (resp. p_2) ont les mêmes pré-conditions. De ce point de vue, vérifier la propriété de transfert pour *deux* transitions (une partant de p_1, l'autre de p_2) est suffisant. Nous pouvons donc poser les restrictions (7.3) et (7.4).

  En fait, nous allons montrer que dans ces conditions, t'_1 permet de rendre équivalentes toutes les places de ${}^\bullet t_1$. Prenons simplement la relation $B = Id_P \cup \{(p_1, p_j)\}_{p_i, p_j \in {}^\circ t_1}$ et montrons que $\overline{B}$ est une bisimulation.

  Le point essentiel, qui se prouve aisément, est le suivant :

$$\forall M, M^+, M^- \in \mathcal{M}_N : \quad \begin{array}{ccc} M & \overline{B} & M + M^- - M^+ \\ {\scriptstyle t'_1}\downarrow & & \downarrow{\scriptstyle t'_1} \\ M' & \overline{B} & M' + M^- - M^+ \end{array}$$

  Donc en particulier nous avons :

$$\forall q \in {}^\circ t_1 : \quad \begin{array}{ccc} M & \overline{B} & M - p_1 + q \\ {\scriptstyle t_1}\downarrow & & \downarrow{\scriptstyle t'_1} \\ M' & \overline{B} & M' - p_1 + q \end{array}$$

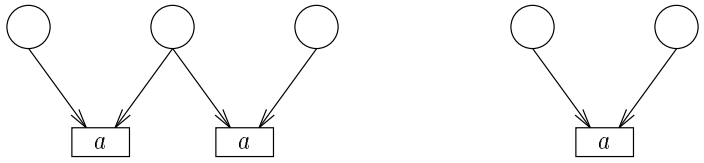
  Pour toutes les autres transitions de T, puisqu'elles n'utilisent ni p_1, ni q, nous avons trivialement :

$$\forall q \in {}^\circ t_1 : \quad \begin{array}{ccc} M & \overline{B} & M - p_1 + q \\ {\scriptstyle t}\downarrow & & \downarrow{\scriptstyle t} \\ M' & \overline{B} & M' - p_1 + q \end{array}$$

  par suite, $\overline{B}$ est une bisimulation, donc B est une bisimulation de place, ce qui conclue la démonstration.

□

Remarque : la réciproque est évidemment fausse comme le montre la figure ci-dessous où N_r est un ECS, mais pas N.



N N_r

◇

7.2.2 Les «Basic Parallel Process»

Il s'agit d'une classe d'expressions CCS basée sur le choix et le parallélisme asynchrone (voir [CH93b]). Exprimée dans les réseaux de Petri, nous avons :

Définition 7.2.3. *Un réseau $N = (P, T, W, l)$ est un* Basic Paralell Process *(*BPP*) ssi :*

$$\forall t \in T,\ |{}^\bullet t| = 1$$

La réduction ne change pas la taille des pré-conditions, d'où la proposition :

Proposition 7.2.4. *Si N est un BPP, alors N_r en est un aussi.*

Remarque : la réciproque est trivialement fausse, comme le montre la figure ci-dessous :

N N_r

◇

Conclusion

Nous donnons simplement un tableau récapitulant les différents résultats :

Σ est		Σ_r est
Vivant	$\Leftrightarrow$	Vivant
Quasi-vivant	$\Leftrightarrow$	Quasi-vivant
n-borné	$\Leftrightarrow$	n'-borné avec $n' = n.\max_{p \in P}\{\lvert[p]\rvert\}$
$\mathcal{R}(N)$ semi-linéaire	$\Leftrightarrow$	$\mathcal{R}(N)$ semi-linéaire
$LA(N)$	$=$	$LA(N_r)$
$L(N)$ régulier	$\Rightarrow$	$L(N_r)$ régulier
FC, EFC, ECS	$\Rightarrow$	FC, EFC, ECS
BPP	$\Rightarrow$	BPP

Les différents résultats s'étendent au cas de la relation structurelle, à l'exception de celui sur les ECS (puisque nous utilisons le fait que $\overline{B}$ est une bisimulation). Dans ce dernier cas cependant, un résultat similaire peut être obtenu dans le cadre des réseaux dont les transitions ont des préconditions non-vides.

Notes bibliographiques

[EN94] et [Hac76] fournissent de nombreux résultats de décidabilité et de complexité pour ces propriétés.

Chapitre 8

Bisimulation de places et composition de réseaux

Il existe principalement deux méthodes de conception :

- ascendante : où le systèmes est construit par assemblage de plus petites composantes. Dans les réseaux, cela correspond à la mise en parallèle de réseaux. Deux opérateurs viennent la compléter, d'une part le renommage, de l'autre l'abstraction.

  la mise en parallèle , synchrone ou non, notée $N_1 \|_A N_2$, $A \subseteq \mathcal{A}$. Quand deux réseaux sont composés de cette manière, ils doivent se synchroniser pour effectuer une action de A. Graphiquement, cela se représente en fusionnant deux à deux les transitions étiquetées dans A. La première section étudie cet opérateur.

  le renommage , noté $N[f]$. Il s'agit ici de changer le nom visible de certaines transitions. Par exemple, si nous voulons modéliser une file FIFO avec deux canaux distincts, nous pouvons changer les noms des actions en ajoutant la désignation du canal (p.ex. «reçoit» devient «reçoit sur le canal 1»).

  l'abstraction , notée $N \setminus A$. Dans ce cas, les actions de A deviennent invisibles. Cela permet de dissimuler des actions qui étaient importantes à un certain niveau de détail, mais ne le sont plus. Ces deux derniers opérateurs sont étudiés dans la deuxième section.

- descendante : où partant d'un modèle très générale, nous détaillons chaque élément du système jusqu'à un niveau de détail voulu. Dans les réseaux, c'est le raffinage. Nous terminons par quelques remarques sur **le raffinage**. En effet, si la réduction a un comportement «agréable» pour les trois opérateurs précédents, il n'est pas possible d'alterner raffinage et réduction.

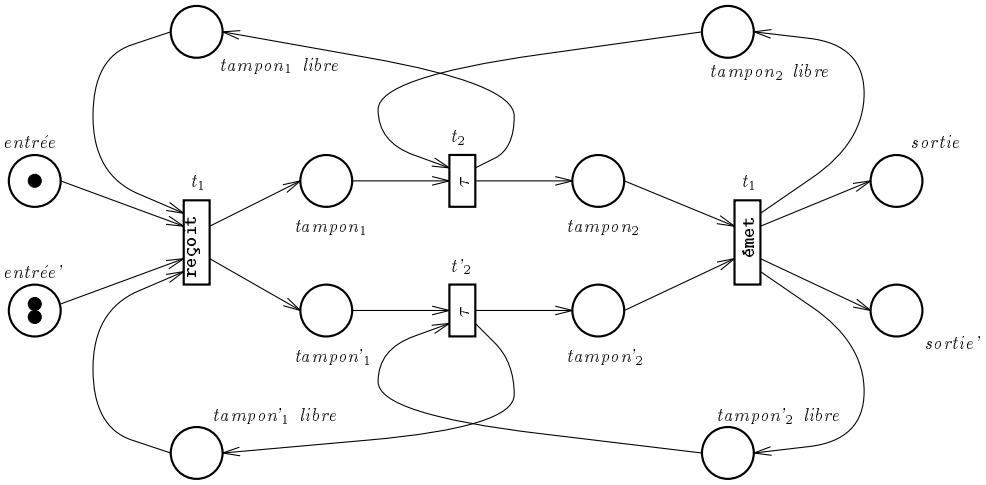
8.1 La composition parallèle

La composition parallèle synchrone est l'outil de composition ascendante par excellence.

8.1.1 L'opérateur $\|_A$

Supposons que nous voulions créer deux files dont les entrées et sorties soient synchrones. Nous allons donc mettre en parallèle deux réseaux N_e en leur demandant de se synchroniser sur `recoit` et `émet`, i.e. $N_{par} = N_e \|_{\{\texttt{recoit},\texttt{émet}\}} N_e$. La figure 8.1 montre le réseau résultant.

Figure 8.1 : La composition parallèle synchrone.



$$N = N_e \|_{\{\texttt{recevoir},\texttt{émettre}\}} N_m$$

Nous avons la définition suivante :

Définition 8.1.1. *Composition parallèle*
Soient $N_1 = (P_1, T_1, W_1, l_1)$ et $N_2 = (P_2, T_2, W_2, l_2)$ deux réseaux étiquetés dans $\mathcal{A}$. Soit $A \subseteq \mathcal{A}$, la composition parallèle de N_1 et N_2 synchronisés sur A *est le réseau $N = (P, T, W, l) = N_1 \|_A N_2$ tel que :*

$$\begin{aligned}
P &\stackrel{\text{déf}}{=} P_1 \times \{\varepsilon\} \cup \{\varepsilon\} \times P_2 \\
T &\stackrel{\text{déf}}{=} \{(t_1, t_2) \ : \ t_1 \in T_1, t_2 \in T_2, l_1(t_1) = l_2(t_2) \in A\} \\
&\quad \bigcup \{(t_1, \varepsilon) \ : \ t_1 \in T_1, l_1(t_1) \notin A\} \\
&\quad \bigcup \{(\varepsilon, t_2) \ : \ t_2 \in T_2, l_2(t_2) \notin A\}
\end{aligned}$$

$$W((p_1,p_2),(t_1,t_2)) \stackrel{\text{déf}}{=} \begin{cases} W_1(p_1,t_1) & si\ p_1 \in P_1\ et\ t_1 \in T_1 \\ W_2(p_2,t_2) & si\ p_2 \in P_2\ et\ t_2 \in T_2 \\ 0 & sinon \end{cases}$$

$$\text{idem pour } W((t_1,t_2),(p_1,p_2))$$

$$l(t_1,t_2) \stackrel{\text{déf}}{=} \begin{cases} l_1(t_1) & si\ t_1 \in T_1 \\ l_2(t_2) & si\ t_2 \in T_2 \end{cases}$$

De la même manière, nous définissons la composition parallèle pour des systèmes par : $(N_1,M_1) \parallel_A (N_2,M_2) \stackrel{\text{déf}}{=} (N_1 \parallel_A N_2, M)$*, avec :*

$$M(p_1,p_2) \stackrel{\text{déf}}{=} \begin{cases} M_1(p_1) & si\ p_1 \in P_1 \\ M_2(p_2) & si\ p_2 \in P_2 \end{cases}$$

Remarques :

1. Dans le cas où un réseau doit se synchroniser avec une action inexistante dans le second réseau, cette action disparaît.
2. Il est évident que nous pouvons définir la composition parallèle asynchrone comme le cas particulier $A = \emptyset$.

◇

8.1.2 Composition parallèle et bisimulations

Nous avons le théorème fondamental suivant :

Théorème 8.1.2. *La bisimulation et la bisimulation de semi-branchement sont des congruences pour la composition parallèle.*

Preuve. Supposons que $N_1 \underline{\leftrightarrow}_1 N_1'$ et $N_2 \underline{\leftrightarrow}_2 N_2'$. Les marquages atteignables de $(N_1,M_1) \parallel_A (N_2,M_2)$ peuvent être vus comme des paires de marquages atteignables de (N_1,M_1) et (N_2,M_2) (et de même pour $(N_1',M_1') \parallel_A (N_2',M_2')$). Alors, on peut définir B comme la restriction de $\{(M_1 \cup M_2, M_1' \cup M_2') \ : \ M_i \underline{\leftrightarrow}_i M_i', i = 1,2\}$ aux marquages atteignables de $(N_1,M_1) \parallel_A (N_2,M_2)$ et $(N_1',M_1') \parallel_A (N_2',M_2')$. B est une bisimulation entre ces réseaux.

□

8.1.3 Un résultat en demi-teinte

Évidemment, le résultat précédent est obtenu grâce à une bisimulation autre que la simple union, avec une restriction aux marquages atteignables. Cependant, la bisimulation de places est indépendante de la mise en contexte, on peut donc espérer pouvoir l'utiliser facilement.

Théorème 8.1.3. *Soient N_1 et N_2 deux réseaux, alors $B = B(N_1) \cup B(N_2)$ est une bisimulation de places sur $N = N_1 \parallel_A N_2$.*

Preuve. Nous notons $N = (P, T, W, l)$.

Il suffit de montrer que $B(N_i) \cup Id_P, i = 1, 2$ est une bisimulation sur N et d'utiliser la propriété 3.1.2, p. 76 disant que :

> «si B et B' sont des BdPS sur N, alors la fermeture symétrique et transitive de $B \cup B'$ est une BdPS sur N.»

Or $B(N_1) \cup B(N_2)$ est trivialement une équivalence.
Par la définition 8.1.1, nous avons :

- pour M de N : $M = M^1 + M^2$ avec $M^i \in \mathcal{M}_{N_i}$;
- pour t de N : $t = (t^1, t^2) \in (T_1 \cup \{\varepsilon\}) \times (T_2 \cup \{\varepsilon\})$;
- pour un pas $M \xrightarrow{t} M'$ de N : $M^1 \xrightarrow{t^1} M'^1$ et $M^2 \xrightarrow{t^2} M'^2$.

Nous vérifions la propriété de transfert pour $M_1 \overline{B(N_1)} M_2$ et $M_1 \xrightarrow{t_1} M'_1$ *dans* N. Avec les notations précédentes :

- soit $t_1^1 = \varepsilon$ (resp. $t_1^2 = \varepsilon$), donc $l(t_1) \notin A$. Alors trivialement le transfert est vérifié par (ε, t_1^2) (resp. par un (t_2^1, ε), issu de la propriété de transfert dans N_1) ;
- soit $t_1^1 \in T_1$ et $t_1^2 \in T_2$, donc $l(t_1) \in A$. Par hypothèse, nous savons que *dans* N_1, il existe un pas $M_2^1 \xrightarrow{t_2^1 : l_1(t_1^1)} M_2'^1$ avec $M_2'^1 \overline{B(N_1)} M_1'^1$. Par suite, *dans* N nous avons par définition le pas $M_2 \xrightarrow{(t_2^1, t_1^2) : l(t_1)} M_2'^1 + M_2'^2$ et le transfert est vérifié.

$$\begin{array}{ccccc} M_1^1 & + & M_1^2 & \overline{B(N_1)} & M_2^1 & + & M_1^2 \\ {\scriptstyle t_1^1}\downarrow & & \downarrow{\scriptstyle t_1^2} & & {\scriptstyle t_2^1}\downarrow & & \downarrow{\scriptstyle t_1^2} \\ M_1'^1 & + & M_1'^2 & \overline{B(N_1)} & M_2'^1 & + & M_1'^2 \end{array}$$

Nous prouvons la même chose pour $B(N_2)$, d'où le résultat.

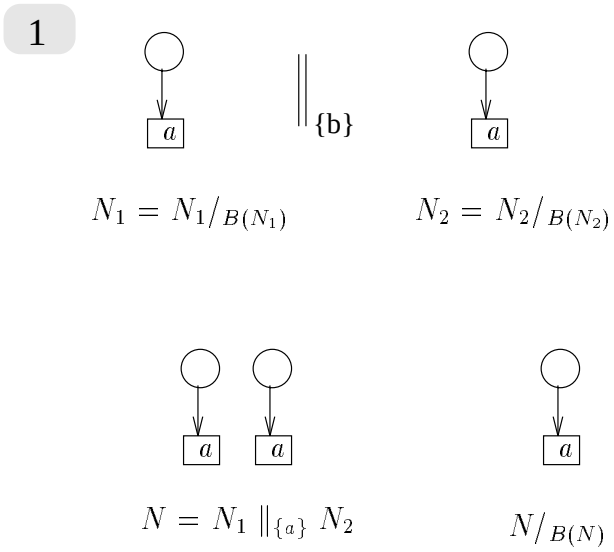
□

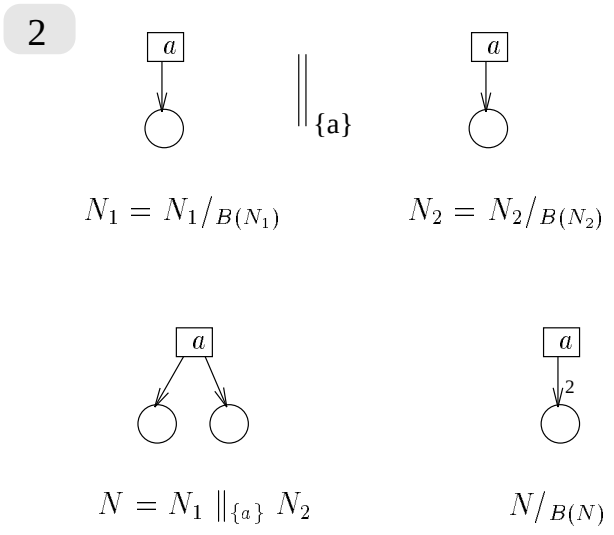
Évidemment, B n'est pas forcément la plus grande BdPS, ainsi que le montrent les exemples triviaux de la figure 8.2, où la synchronisation se fait sur a ou b.

8.2 L'abstraction et le renommage

Nous nous contenterons de présenter brièvement ces deux opérateurs et le théorème de congruence pour la bisimulation. Ensuite, nous montrerons de quelle manière ils changent la plus grande bisimulation de place.

Figure 8.2 : Des exemples où $B(N_1) \cup B(N_2) \subsetneq B(N)$.





Définition 8.2.1. *Renommage et abstraction*

Soit $N = (P, T, W, l)$ un réseau et $f : \mathcal{A}_\tau \to \mathcal{A}_\tau$ une fonction telle que $f(\tau) = \tau$ et $f^{-1}(\tau) = \{\tau\}$. Le renommage de N par f, *noté $N[f]$ est obtenu en changeant l'étiquetage par :*

$$l_{N[f]}(t) = f(l(t))$$

De la même manière, l'abstraction *d'un ensemble d'actions $A \subseteq \mathcal{A}$ est obtenu en changeant l'étiquetage par :*

$$l_{N[f]}(t) = \begin{cases} l(t) & si\ t \notin A \\ \tau & sinon \end{cases}$$

Informellement, le renommage permet donc «d'instancier» une composante du réseau alors que l'abstraction permet de la dissimuler. Pour ces deux opérations nous avons :

Théorème 8.2.2. *La bisimulation et la bisimulation de semi-branchement sont des congruences pour le renommage et l'abstraction. De plus, si B est une (sb-)bisimulation entre N_1 et N_2, alors c'en est une aussi entre $N_1[f]$ (resp. $N_1 \setminus A$) et $N_2[f]$ (resp. $N_2 \setminus A$).*

Nous en dérivons sans peine le corollaire suivant :

Corollaire 8.2.3.

$$\begin{aligned} B(N) &\subseteq B(N[f]) \\ B(N) &\subseteq B(N \setminus A) \\ B_{sb}(N) &\subseteq B_{sb}(N[f]) \\ B(N) &\subseteq B_{sb}(N \setminus A) \end{aligned}$$

Il est facile de voir que $B(N) \neq B(N[f])$ (resp. $B_{sb}(N) \neq B_{sb}(N[f])$) n'est possible que si le renommage «identifie» des actions, i.e. $\exists a \in \mathcal{A} : |f^{-1}(a)| > 1$.

8.3 Le raffinage

Notre méthode de réduction est incompatible avec le raffinage. En effet, ce qui est sous-jacent à la fusion de place est de ramener une partie du parallélisme structurel à du parallélisme dynamique. Or il est connu que l'égo-concurrence est incompatible avec le raffinement, comme le montre l'exemple de la figure 8.3

Ce résultat était prévisible, dans la mesure où $\underline{\leftrightarrow}$ n'est pas une congruence pour le raffinement.

Les réseaux ont les mêmes processus. Cependant, $raf_1(N_r)$ peut faire *cce* et pas $raf_1(N)$. Si en plus on veut travailler avec les mots partiels (ou les processus), seul $raf_2(N_r)$ peut faire :

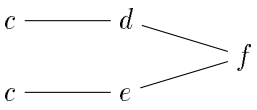
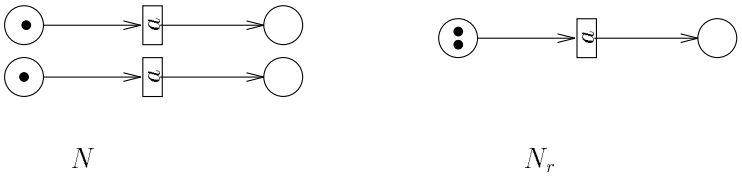
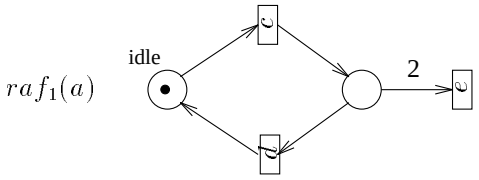
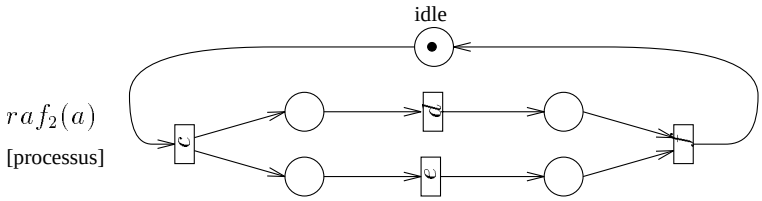


Figure 8.3 : Égo-concurrence et raffinement.







Il existe plusieurs solutions :

- interdire l'égo-concurrence dans les réseaux de manière dynamique, p.ex. en utilisant des réseaux saufs : malheureusement, la classe des réseaux saufs n'est pas stable pour notre méthode de réduction ;
- ne raffiner que des transitions non égo-concurrentes : ce qui fait appel à un calcul dynamique et n'est pas «rentable» ;
- empêcher de manière structurelle l'égo-concurrence en ajoutant une place marquée d'un unique jeton en boucle sur la transition. Cela augmente le nombre de places, ce qui est peu intéressant.

- ne pas raffiner un réseau réduit ;
- la voie la plus ouverte semble être d'utiliser la bisimulation de places pour effectuer un repliage coloré du réseau : c'est actuellement un problème ouvert.

Conclusion

Dans le cadre général d'une insertion dans un outil de conception, nous avons vu que la bisimulation de places peut être utilisée à n'importe quel moment tant que l'on se contente des opérateurs $\|_A$, $N[f]$ et $N \setminus A$. Par contre, elle ne peut être utilisée qu'à l'étape «finale» si nous utilisons le raffinage.

Outre l'adaptation au raffinage, une voie de recherche, est de voir de quelle manière cela se combine avec la *sb*-bisimulation, puisque tous les théorèmes s'appliquent aussi à $\underline{\leftrightarrow}_{sb}$.

Cependant, l'ensemble des résultats sont suffisant pour chercher à savoir ce que la BdPS vaut sur des cas réels. Le dernier chapitre présente brièvement le prototype `PetriS` qui implémente les bisimulations de places et les réseaux de Petri.

Chapitre 9

Le logiciel `PetriS`

Le logiciel développé durant la thèse répondait à différents objectifs :

- vérifier la portée pratique des méthodes ;
- pouvoir tester sur des cas réels les différentes réductions ;
- posséder une structure assez souple pour s'adapter aux différentes variantes des bisimulations et des réseaux ;
- disposer d'une interface suffisamment souple pour pouvoir communiquer avec l'utilisateur et d'autres logiciels.

Il s'inscrivait donc en prolongement du prototype développé en DEA et du module de composition développé par ailleurs ([Kup93]).

Dans ce chapitre nous allons présenter d'une part la structure du logiciel (avec les outils associés), d'autre part les différentes grammaires de l'interpréteur.

Nous terminons par un exemple de session et quelques résultats expérimentaux basés soit sur une génération aléatoire de réseaux, soit sur des réseaux issus du logiciel LOTOS.

9.1 Les concepts

Il ne s'agit pas ici de détailler le code ni de parler de choix d'implémentation (voir pour cela le document de référence [1]), mais de présenter les modules de base.

Pour développer les différents modules, nous avons utilisé des logiciels «classiques». Le langage choisi est C++.

- Pour l'interpréteur de commande, nous avons utilisé `fLex++` comme analyseur lexical et `Bison` pour la syntaxe.

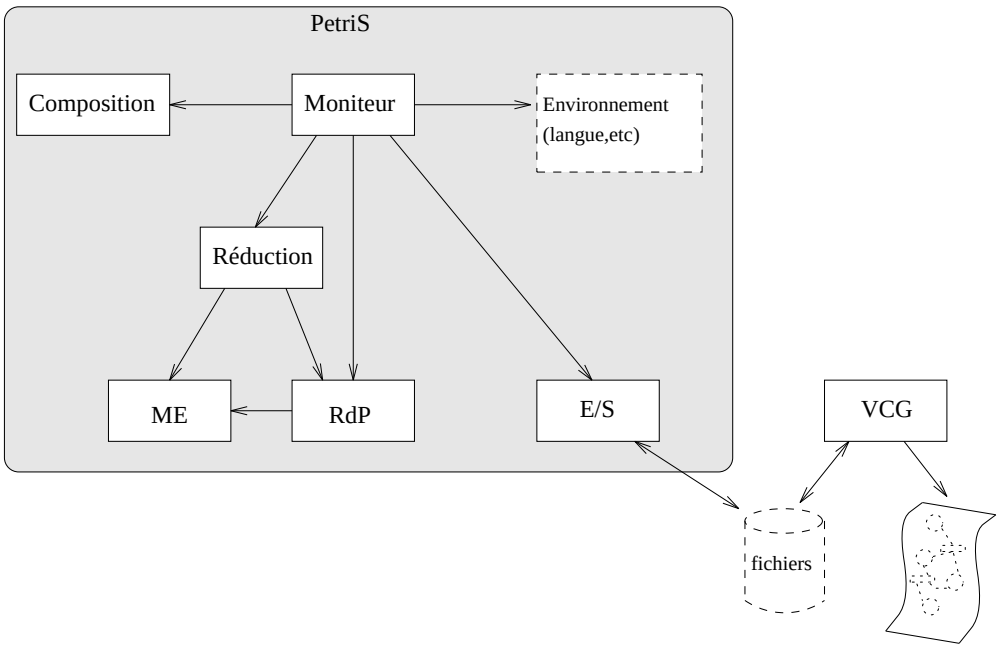
[1] C++ [Del93], fLex++, Bison, G++, norme ANSI, [Kup93], POO [Mey88], VCG [LS]

- Le compilateur choisi est `G++`[2], actuellement suivant la norme 3.0 du langage C++.
- Pour visualiser les réseaux, nous avons choisi d'interfacer le logiciel avec `VCG` (Visualisation of Compiler Graphs) qui offre un vaste choix d'optimisations.
- Enfin, l'ensemble du programme est auto-documenté grâce à LaTeX 2_ε et HTML (pour la grammaire).

9.1.1 L'architecture du logiciel

Les différents modules sont représentés sur la figure 9.1 avec leurs liens.

Figure 9.1 : Architecture de `PetriS`.



L'interpréteur de commandes peut fonctionner soit en interactif, soit en commande par lot. Chaque commande est décrite dans une des sections suivantes.

Le module de composition permet de calculer un nouveau réseau en appliquant les différents opérateurs vu au chapitre 8 (composition parallèle, renommage, etc) plus quelques autres.

[2]Qui a connu trois normes et au moins vingt versions...

Le module de réduction effectue le calcul des bisimulations et des étapes de réduction. Il contient aussi la procédure de p-saturation d'un réseau.

Le module RdP implémente les réseaux de Petri. D'une part leur structure, d'autre part leur comportement.

Le module ME concerne les multi-ensembles et les opérateurs associés.

Le module E/S regroupe les fonctions d'entrée/sortie. En particulier, nous y trouvons l'interface avec le logiciel de visualisation VCG.

Le module environnement permettrait de gérer des paramètres d'initialisation du logiciel tels que le choix d'une langue ou d'une famille de réseaux.

9.1.2 Quelques commandes

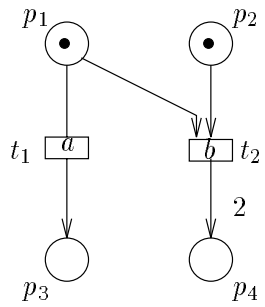
Globalement, le logiciel permet d'effectuer des opérations sur une liste de réseaux. Le tableaux ci-dessous donne quelques commandes. Il existe en plus des commandes de simulation (permettant de tracer des séquences) et de modification (permettant d'ajouter des nœuds ou arcs, de modifier l'étiquetage ou le marquage).

Type de commande	Syntaxe	Remarques
Gestion de la liste	`charger` *réseau* `supprimer` *réseau* `afficher` *réseau* `lister`	Ce sont des fonctions classiques. `lister` permet d'avoir divers renseignements en plus du nom du réseau (n_P, n_T, d_P, d_T, d entre autres).
Définition d'un réseau	`N:=` *définition* `N=` *calcul*	Nous présentons plus loin un exemple de définition extensive de réseau, la syntaxe est assez intuitive. Les calculs possibles regroupent un sous-ensemble de CCS (mise en parallèle synchrone ou non, abstraction, fusion de places ou transitions) et bien sur les opérations liées à la bisimulation de places.
Calculs de BdP	`bdp-s` *réseau* `bdp-tau` *réseau borne*	Dans le cas de la τ-BdP, la borne est une limite de p-saturation.
Entrées/sorties	`sauver` *réseau* `afficher` *réseau* `dessiner` *réseau*	La sauvegarde se fait au format du logiciel[3]. Le dessin se fait au format de VCG afin de pouvoir ultérieurement visualiser les réseaux.

9.1.3 Définir un réseau

Nous ne présentons ici qu'un exemple simple et complet, la syntaxe est très simple. Le réseau décrit ci-dessous correspond à celui de la figure 9.2

Figure 9.2 : Un petit exemple.



N

```
/*=-=-=-=-=-=-=-=-= Net petit-exemple =-=-=-=-=-=-=-=-=*/

/*
         Saved to : petit-exemple.net
         Modified since last save
         Thu Nov 16 23:45:43 1995


        Nb of places : 4
        Nb of transitions : 2
        Degree of P : 2
        Degree of T : 4
        Degree : 4 */

P = { p1, p2, p3, p4 }

T = { t1, t2 }

W = {
   p1:1
        -[t1:a]->
   p3:1;

   p1:1,p2:1
        -[t2:b]->
   p4:2
}
```

```
l = { t1 : a ;
t2 : b
}

M0 = { p1:1, p2:1 }

M = {  }

/*=-=-=-=-=-=-=-= End of petit-exemple =-=-=-=-=-=-=-=*/
```

Dans l'état actuel, seuls les réseaux «classiques» sont implémentés. Cependant, le logiciel a été construit pour s'adapter aux autres types de réseaux (grâce aux patrons et à la sur-définition).

9.2 Une session

En général, le traitement des erreurs n'étant pas implémenté, il est préférable de charger les réseaux et d'utiliser un fichier de commandes par lots.

9.2.1 Réduire la ligne FIFO à deux entrées

Nous utilisons la suite de commandes :

```
/* chargement du reseau */
load fifo-2

/* calcul de B(fifo-2) */
bdps (fifo-2)

/* quitte */
quit
```

Un fichier `log` est produit :

```
B(fifo-2)

Strict place bisimulation results :
-----------------------------------
        { p10, p4 }
        { p9, p3 }
        { p8, p2 }
        { p7, p1 }
```

```
        { p6 }
        { p5 }

Times :
-------
                 Pract   Theo
Nb of loops      3       10
Nb of t2         288     3600

2 seconds, 1 to compute B(N).

Reduction :
-----------
        |P|     |T|     degrees of P,T
N       10      6       4, 4
Nr      6       3       2, 4

[Nr saved to red-fifo-2.net]
```

Le réseau réduit est alors :

```
/*=-=-=-=-=-=-=-= Net red-fifo-2 =-=-=-=-=-=-=-=-=*/

/*
         Saved to : red-fifo-2.net
         Modified since last save
         Fri Nov 17 15:44:04 1995


        Nb of places : 6
        Nb of transitions : 3
        Degree of P : 2
        Degree of T : 4
        Degree : 4 */

P = { p10, p9, p8, p7, p6, p5}

T = { t4, t5, t6}

W = {
   p5:1, p7:1
        -[t4:prog]->
   p8:1;
```

```
   p6:1, p8:1
        -[t5:prog]->
   p5:1, p9:1;

   p9:1
        -[t6:prog]->
   p10:1, p6:1
}

l = { t4, t5, t6 : prog
}

M0 = {  }

M = {  }

/*=-=-=-=-=-=-=-=-= End of red-fifo-2 =-=-=-=-=-=-=-=-=*/
```

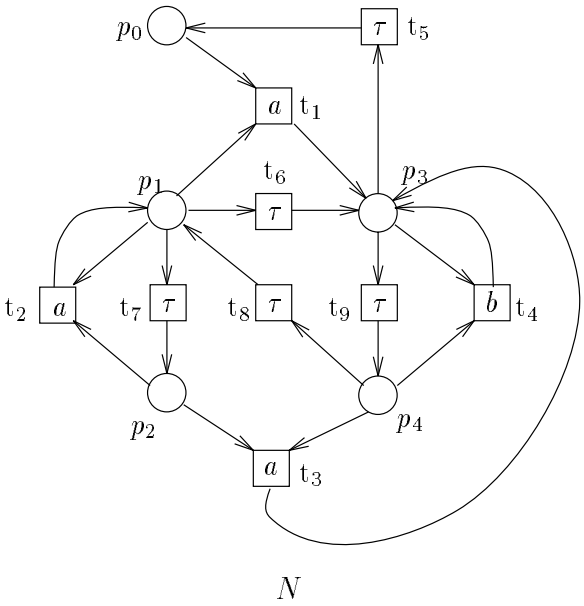
9.2.2 Un exemple pour la τ-bisimulation

Nous prenons le réseau de la figure 9.3. Nous allons constater l'explosion du nombre de transitions sur la figure 9.4. Cette figure est bien sûr illisible, mais un long discours vaut moins qu'une belle image, surtout lorsque nous la comparons au réseau réduit (fig. 9.5).

Nous avons les résultats suivants :

```
tau-p-saturation
tau-p-saturation complete : 2 iterations et 7 ajouts.

p-saturation
p-saturation complete : + 86 transitions (759 etudiees).

B_tau(ICCI)


p-saturation :
--------------
          |P|      |T|      degrees of P, T
N         5        9        7, 3
Nsat      5        102      64, 3
```

Figure 9.3 : Un petit réseau à réduire avec $B_\tau(N)$.




```
Tau place bisimulation results :
--------------------------------
        { p4, p3, p1 }
        { p2, p0 }

Times :
-------
                 Pract   Theo
Nb of loops      3       5
Nb of t2         9239    8.28061e+11

Nb of U          0       2.48418e+12

111 seconds :
        2 to tau-p-saturate,
        57 to p-saturate,
        47 to compute B_tau(N).

Reduction :
-----------
        |P|     |T|     degrees of P,T
```

```
N       5       9       7, 3
Nr      2       4       7, 3

[Nr saved to red-ICCI.net]
```

9.3 Quelques résultats

Ces réseaux d'exercice sont évidemment un peu juste pour juger de la validité d'une telle construction. Nous avons donc testé l'algorithme pour quelques programmes issus de LOTOS (donc déjà «optimisés») :

Réseau	n_P	n_T	$\|[P]\|$	Facteur de réduction
daemon_1	7	7	7	1.0
daemon_2	9	11	7	0.8
daemon_4	13	19	7	0.5
daemon_8	21	35	7	0.3
daemon_16	37	67	7	0.2
daemon_32	69	131	7	0.1
daemon_64	133	259	7	0.05
real001	25	37	8	0.3
real002	5	6	4	0.8
real003	11	13	9	0.8
real004	16	21	8	0.5
real005	21	25	17	0.8
real006	33	62	17	0.5
real007	31	37	25	0.8
real008	49	102	25	0.5
real009	41	49	33	0.8
real010	65	142	33	0.5
real011	5	6	5	1.0
real012	7	5	7	1.0

Une moyenne n'a pas de sens, mais on voit que les réductions peuvent être très fortes.

Conclusion

L'implémentation par un langage orienté objet a permit de séparer efficacement la structure et le comportement du réseau, tout en laissant de grandes possibilités d'adaptation. Le code est largement commenté et l'ensemble est doté d'une documentation élaborée.

Figure 9.4 : La p-saturation du réseau de la figure 9.3.

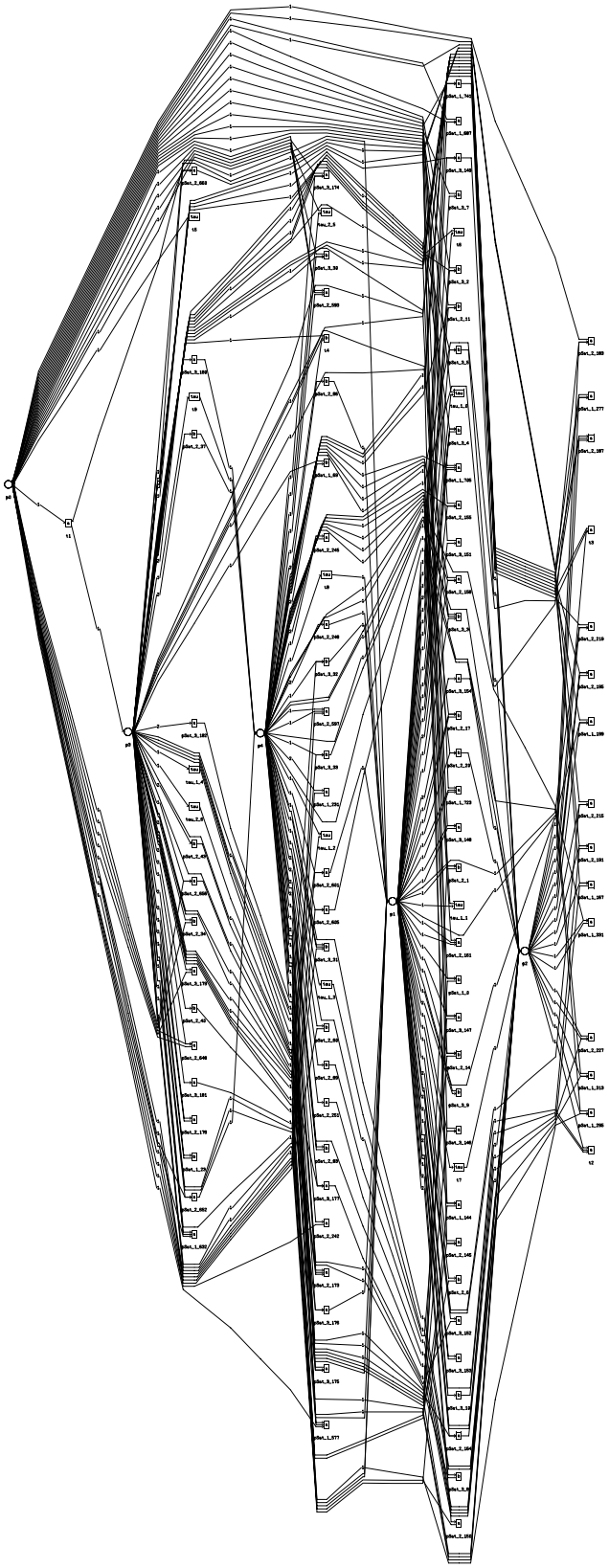
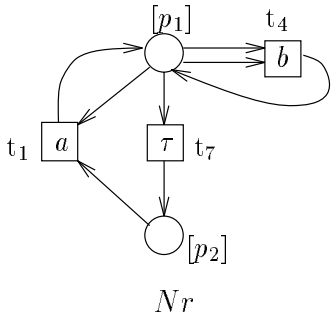


Figure 9.5 : Le réseau de la figure 9.3 réduit.



Nr

Parmi les améliorations possibles, nous retiendrons essentiellement le passage sous PVM [Oak94] et la gestion des erreurs par l'analyseur syntaxique.

Conclusion

Nous avions comme objectif de trouver une méthode de réduction telle que :

1. la réduction se fasse en fusionnant les places du réseau ;
2. que le réseau réduit soit bisimilaire au réseau original ;
3. que la notion sous-jacente soit souple pour s'adapter à d'autres contextes ;
4. que la relation sur les places soit calculable.

Dans le premier chapitre, nous avons montré que la plus grande relation sur les places ($\sim$) vérifiant les deux premiers points n'était pas calculable. Cependant, nous avons vu que la bisimulation de places, issue d'une autre voie de recherche, était une approximation de $\sim$.

Dans les chapitres suivants, nous avons montré qu'elle s'adaptait à la τ-bisimulation avec de légères restrictions sur les réseaux. Cela nous a permis de définir la τp-bisimulation de places, qui est une approximation aisément calculable de la τ-BdP. Nous avons vu qu'a priori nous ne savions pas comment étendre ces résultats aux bisimulations de branchement.

Nous nous sommes ensuite intéressé aux extensions des réseaux de Petri, particulièrement aux arcs inhibiteurs. Dans ce cadre, nous avons étendu la BdPS avec des algorithmes non-triviaux et une manière de quotienter adaptée.

Nous avons terminé sur quelques résultats pratiques permettant d'intégrer la réduction à un outil de génération de réseaux de Petri.

Bien que de nombreuses voies aient été explorées et aient donné de nombreuses indications quant à la manière dont la bisimulation de places s'adapterait à d'autres extensions, ils reste un certain nombre de champs non explorés :

- des problèmes généraux :
  - Dans quelles classes de réseaux les problèmes posés seraient-ils plus simples ? En particulier, avoir $\mathcal{R}(N)$ semi-linéaire nous permettrait-il de mieux prendre en compte le marquage initial ?

  - Comment combiner les différentes méthodes de réduction ?
  - Améliorer les algorithmes et passer à une application pratique «grandeur nature».

- des problèmes plus particuliers :
  - Quelle est la limite exacte entre la bisimulation de places et $\sim$? Et plus précisément, quels résultats pouvons-nous dériver pour les bisimulations additives ?
  - Quels liens pouvons nous établir entre notre méthode de réduction et les repliages colorés des réseaux de Petri ? Cela offre-t-il une alternative pour le raffinage ?
  - Nous avons montré que les bisimulation de branchement n'étaient pas calculables pour des réseaux qui ne soient pas τp-saturés, existe-t-il une autre méthode qui le permette ?

D'autres méthodes de réduction

Il existe d'autres méthodes de réduction pour certaines classes de réseaux. En général, elles visent à réduire l'espace des états tout en conservant un certain nombre de propriété (vivant, borné, etc) plutôt que le comportement du réseau. Elles sont définies sur des réseaux non étiquetés. Les différences avec la bisimulation de places sont les suivantes :

- elles sont locales, i.e. ne considèrent qu'un sous-réseau relativement restreint. Elles nécessitent souvent plusieurs «passages» sur le réseau et on ne peut *a priori* dire à quel moment la réduction sera maximale.
- elles ne préservent que quelques propriétés dynamiques du système ;
- leurs coûts sont relativement élevés.

Il est très difficile de comparer des méthodes ayant des objectifs si différents, nous allons donc simplement donner quelques pointeurs :

- [Ber86] présente de nombreuses méthodes visant à réduire le graphe d'états des réseaux à arcs simples en conservant plusieurs propriétés comportementales. Nous retiendrons en particulier, la bisimulation de places est une généralisation de la notion de «places équivalentes».
- [Had90] est une généralisation des méthodes précédentes aux réseaux colorés (évitant ainsi les coûteux dépliages). Nous y trouvons en plus une méthode de réduction du nombre de couleurs (l'*orthonormalisation*).
- [LFB87] fait appel à une notion de *macro-nœuds* pour réduire des réseaux quelconques. Cependant, seuls les propriétés «vivant» et «borné» sont conservées.

- [DE95] donne une liste de réductions pour les réseaux à choix libres. Il qualifie de «strongly sound», les réductions telles que «N est bien-formé ssi N_r est bien formé», où un réseau N est *bien formé* ssi il existe un marquage M_0 tel que le système (N, M_0) soit borné et vivant. Les résultats du chapitre 7 nous permettent d'ajouter notre méthode de réduction aux réductions «strongly sound»[4].

[4] Les réductions de [Ber86] sont aussi «strongly sound».

Annexes

Annexe A

Les machines de Minsky

Voir le théorème 2.3.2.

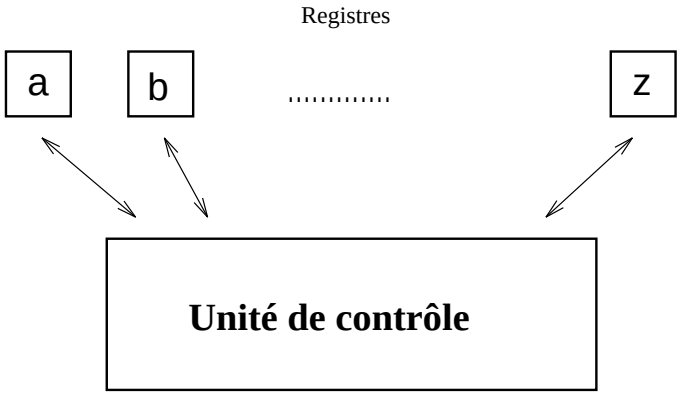
La majeur partie de ce qui est décrit ici est tiré de [Min67], plus particulièrement des chapitres 8 (indécidabilité de l'arrêt) et 11 (description des machines). Nous utiliserons indifféremment les noms de *machines de Minsky* et de *machines à compteurs*.

A.1 Description des machines à compteurs

Le modèle des machines à compteurs est équivalent à celui des machines de Turing ou au formalisme des fonctions «general-recursive», MINSKY utilise ce modèle car il est «plus proche de ce qui se passe dans les ordinateurs digitaux». Dans cette formulation, le problème de la finitude du modèle est éliminé en prenant des machines possédant un nombre fini de registres infinis (i.e. pouvant contenir des nombres arbitrairement grands).

La structure d'une machine à compteurs est donné par la figure A.1.

Figure A.1 : Structure d'une machine à compteurs.



L'unité de contrôle est d'états finis. Il est donc impossible de gérer directement les nombres arbitrairement grands des registres. L'astuce consiste à ne pouvoir distinguer que deux cas : un registre est soit vide, soit plein, i.e. il contient soit 0, soit un nombre positif.

La table A.1 donne la liste commentée des opérations possibles. Un *programme* va donc être une liste de lignes *numérotées* portant chacune une instruction.

Opération	Abréviation	Sémantique
$r := 0$	$r := 0$	Affecte 0 au registre r puis passe à l'instruction suivante
$r := r + 1$	$r++$	Incrémente r de 1 et passe à l'instruction suivante
Si $r > 0$ Alors $r = r - 1$ Sinon aller à n	$r--(n)$	Si $r > 0$ alors on décrémente r et on passe à l'instruction suivante, sinon on passe directement à la ligne n
Arrêt	`H`	Arrête la machine

Table A.1 : Les instructions des machines à compteurs.

Exemple A.1 : Une machine à compteurs.

Soit une machine disposant de sept compteurs : `o`, `e`, `t1`, `t1l`, `t2`, `t2l` et `s`. Nous lui fournissons le programme suivant :

```
début   ( 1 ) o := 0
vide_e  ( 2 ) t1l−− (vide_t1)
        ( 3 ) e−− (e_vide)
        ( 4 ) t1++
vide_t1 ( 5 ) t2l−− (vide_t2)
        ( 6 ) t1−− (t1_vide)
        ( 7 ) t2++
        ( 8 ) t1l++
vide_t2 ( 9 ) t2−− (itère)
        (10) s++
        (11) t2l++
itère   (12) e−− (fin)
        (13) e++
        (14) o−− (début)
e_vide  (15) t1l++
        (16) o−− (vide_t1)
t1_vide (17) t2l++
        (18) o−− (vide_t2)
fin     (19) H
```

Supposons que l'état initial de la machine soit : `e` contient 2, `t1l` et `t2l` contiennent tous deux 1. Le déroulement serait alors le suivant[1] :

Instruction	Effet	`e`	`t1`	`t1l`	`t2`	`t2l`	`s`
1	Met `o` à zéro	2	0	1	0	1	0
2	Décrémente `t1l`	2	0	0	0	1	0
3	Décrémente `e`	1	0	0	0	1	0
4	Incrémente `t1`	1	1	0	0	1	0
5	Décrémente `t2l`	1	1	0	0	0	0
6	Décrémente `t1`	1	0	0	0	0	0
7	Incrémente `t2`	1	0	0	1	0	0
8	Incrémente `t1l`	1	0	1	1	0	0
9	Décrémente `t2`	1	0	1	0	0	0
10	Incrémente `s`	1	0	1	0	0	1
10	Incrémente `t2l`	1	0	1	0	1	1
12	Décrémente `e`	0	0	1	0	1	1
13	Incrémente `e`	1	0	1	0	1	1
14	Itère à l'instruction 2	1	1	0	0	1	1
...	...						
9	Décrémente `t2`	0	0	1	0	0	1
10	Incrémente `s`	0	0	1	0	0	2
10	Incrémente `t2l`	0	0	1	0	1	2
12	Saute à l'instruction 19	0	0	1	0	1	2
19	Fin	0	0	1	0	1	2

A.2 Le problème de l'arrêt

L'essentiel de cette section est issu du chapitre 8 de [Min67].

Parmi les problèmes indécidables de [Min67], nous retiendrons les deux suivants :

le problème de l'arrêt (PBA) pour une machine $\mathcal{C}$ et une configuration θ s'exprime comme suit :

> «$\mathcal{C}$ s'arrête-t-elle en partant de (s_0, θ) ?»

le problème de l'arrêt uniforme (PBAU) pour une machine $\mathcal{C}$ s'exprime comme suit :

> «$\mathcal{C}$ s'arrête-t-elle en partant de (s_0, θ) quel que soit θ ?»

le dual de PBAU (D-PBAU) pour une une machine $\mathcal{C}$ s'exprime comme suit :

> «$\mathcal{C}$ ne s'arrête-t-elle jamais en partant de (s_0, θ) quel que soit θ ?»

[1] Nous ne faisons pas figurer le registre `o`, puisqu'il est constamment à 0 (en fait il sert à faire un saut inconditionnel).

Ces trois problèmes sont indécidables. PBA et PBAU sont traités dans [Min67], nous allons montrer que D-PBAU est indécidable en utilisant la même technique que MINSKY pour PBAU. Formellement :

Théorème A.2.1. *D-PBAU est indécidable.*

Preuve. Pour la preuve, nous repassons aux machines de Turing. Nous renvoyons le lecteur à [Min67] pour une présentation de ces machines et pour leur équivalence avec les machines à compteurs.

Soit T une machine quelconque démarrant en Q_0. Nous construisons T^B, démarrant en q_0^*, en ajoutant à T quelques quintuplets. A et B étant deux symboles inusités, pour tout q_i de T nous ajoutons :

$$\begin{array}{ll} (q_i, A, q_i', 0, L) & \{q_i', \Phi, q_i, A, R\} \\ (q_i, B, q_i'', 0, R) & \{q_i'', \Phi, q_i, B, L\} \end{array}$$

q_i', q_i'' étant de nouveaux états.

Nous ajoutons alors les états :

$$\begin{array}{c} \{q_0^*, \Phi, q_1^*, A, R\} \\ \{q_1^*, \Phi, q_3^*, 0, R\} \\ \{q_2^*, \Phi, Q_0, B, L\} \end{array}$$

Informellement, nous avons ajouté à T une mécanique effaçant la bande avant de la lire. Par la même méthode que [Min67], nous avons :

Lemme A.2.2. «*T ne s'arrête pas pour le ruban vide*» *est équivalent à* «*T ne s'arrête jamais*».

Preuve informelle : T^B est T plus un mécanisme d'effaçage : elle commence par écrire `A0B` puis démarre sur ce `0`. Ensuite, à chaque fois que de besoin, elle pousse `A` ou `B` en effaçant le ruban, quoiqu'il s'y trouve. L'indécidabilité de l'arrêt pour le ruban vide implique donc celle de D-PBAU (puisque «T ne s'arrête pas pour le ruban vide» est équivalent à «T s'arrête pour le ruban vide»). La réciproque est classique.

Remarque : ce problème est équivalent au problème 8.8-2(1), p. 155 de [Min67].

◇

□

A.3 Machines à compteurs et réseaux de Petri

Il est évident que les réseaux de Petri ne sont pas équivalents aux machines à compteur (à moins de leur ajouter des *arcs inhibiteurs*, voir le chapitre 6).

Le principal obstacle est le non-déterminisme induit par les réseaux de Petri. Il est cependant possible de tricher un peu, en admettant que l'exécution du programme correspond à un comportement du réseau. Nous aurons donc un «contrôle» sur le réseau. Une traduction des instructions est donnée par la figure A.2.

Le réseau ainsi constitué a donc :

- une place par registre,
- une place par instruction,
- une ou deux transitions par instructions.

Remarques :

1. Nous pouvons parfaitement établir la correspondance avec la machine telle que définie en A.1 :

   i_1 correspond à la séquence $\begin{array}{l} o := 0 \\ o - - (s) \end{array}$

   i_2 est codé par $\begin{array}{l} c - - s' \\ o := 0 \\ o - - (s) \end{array}$

   i_3 est simplement $\begin{array}{l} o := 0 \\ o - - (s_0) \end{array}$

2. Nous pouvons parfaitement établir la correspondance avec la machine telle que définie en 2.3 :

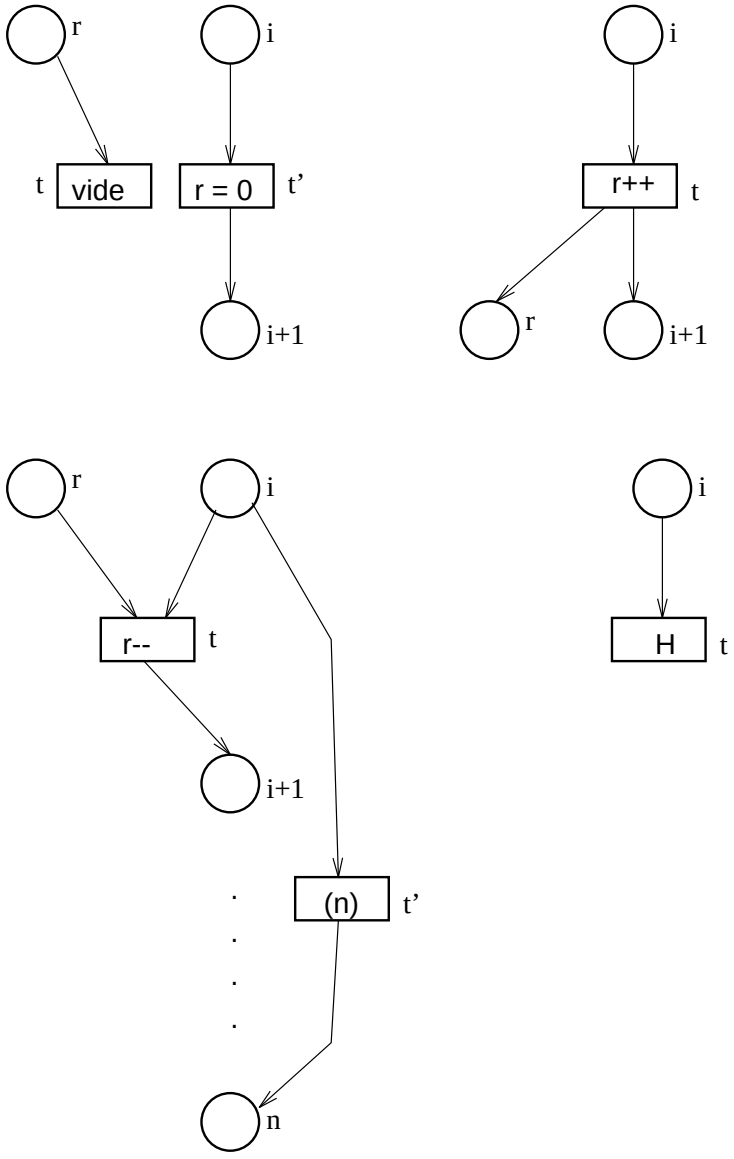
   i_1 correspond à la séquence $\begin{array}{l} o := 0 \\ o - - (s) \end{array}$

   i_2 est codé par $\begin{array}{l} c - - s' \\ o := 0 \\ o - - (s) \end{array}$

   i_3 est simplement $\begin{array}{l} o := 0 \\ o - - (s_0) \end{array}$

$\diamond$

Figure A.2 : Des machines à compteurs vers les réseaux de Petri



Annexe B

La p-saturation

La p-saturation est une étape cruciale pour le calcul de $B_\tau(N)$ puisque nous avons (théo. 4.9.2, p. 114) :

Si N est un réseau p-saturé, alors :

$$B_\tau(N) = B_{\tau p}(N) \qquad \text{(B.1)}$$

Dans ce chapitre, nous allons explorer la p-saturation et son utilisation pratique. Particulièrement, nous allons montrer que :

1. nous pouvons passer d'un réseau N à un réseau N' qui soit p-saturé et tel que $B_\tau(N) = B_\tau(N')$ (coro. B.2.5, p. 201) ;
2. qu'il est parfois possible de décider, étant donné un réseau N, de l'existence d'un réseau fini N' p-saturé vérifiant $B_\tau(N) = B_\tau(N')$ (section B.4, p. 206) ;
3. que lorsqu'il existe ce réseau peut être construit et comment il peut l'être (théo. B.3.3, p. 204) ;
4. enfin d'estimer la taille de N' en fonction de caractéristiques structurelles de N (section B.5, p. 216).

Nous introduisons (ou rappelons) tout d'abord quelques définitions et notations :

Définition B.0.1. *Réseau τp-saturé, couverture*
Soit $N = (P, T, W, l)$ un réseau, nous avons (déf. 4.9.1, p. 113 et 4.1.1, p. 102 resp.) :

$$T^\times \stackrel{\text{déf}}{=} \{\sigma \in T^* \ : \ l_\tau(\sigma) \in \mathcal{A} \cup \{\varepsilon\}\}$$

$$\sigma_1 \sqsubseteq \sigma_2 \stackrel{\text{déf}}{\iff} ({}^\bullet\sigma_2 - {}^\bullet\sigma_1 = \sigma_2{}^\bullet - \sigma_1{}^\bullet \wedge (l(\sigma_1) =_\tau l(\sigma_2))$$

Un réseau est p-saturé *(déf. 4.9.1, p. 113) ssi pour toute séquence* $\sigma \in T^{\times}$*, il existe un multi-ensemble* $U \in \mathcal{M}(T)$ *tel que* $U \sqsubseteq_\tau \sigma$*.*

Nous dirons qu'un réseau est τp-saturé *ssi :*

$$\forall \sigma \in T_\tau^*,\ \exists \mu \in \mathcal{M}(T_\tau)\ :\ \mu \sqsubseteq \sigma$$

Nous dirons alors que σ *est* couverte *par* μ *(resp.* U*).*

Il est facile de voir que N est τp-saturé ssi N_τ l'est.

B.1 p-saturation d'un réseau

A priori, nous ne savons pas p-saturer un réseau. Tout au moins, une approche qui consisterait à chercher à saturer toutes les séquences $\sigma \in T_N^{\times}$ de longueur quelconque ne semble pas raisonnable. En fait, nous allons voir que nous pouvons nous ramener à la saturation des séquences de longueur 2.

B.1.1 De la p-saturation à la $2p$-saturation

Définition B.1.1. *Un réseau sera dit* $2p$*-saturé ssi toutes les séquences* $\sigma \in T_N^{\times}$ *de longueur* 2 *sont couvertes dans* N*.*

Un résultat fondamental sur la p-saturation d'un réseau est le suivant :

Théorème B.1.2. *Un réseau est* p*-saturé ssi il est* $2p$*-saturé.*

Preuve. "$\Rightarrow$" est trivial. Pour prouver "$\Leftarrow$", nous considérons un réseau N $2p$-saturé.

Nous allons montrer comment construire pour chaque séquence $M = {}^\bullet\sigma \xrightarrow{\sigma} \sigma^\bullet = M'$ un pas concurrent $M \xrightarrow{U} M'$ avec $U \sqsubseteq \sigma$. Supposons qu'au pas $M \xrightarrow{\sigma} M'$ correspond le processus $\pi = (C, f)$, où $C = (P_C, T_C, F_C)$. Nous montrons comment construire un processus $\pi' = (C', f')$ dans N tel que :

(1) $f'({}^\circ C') = M$,

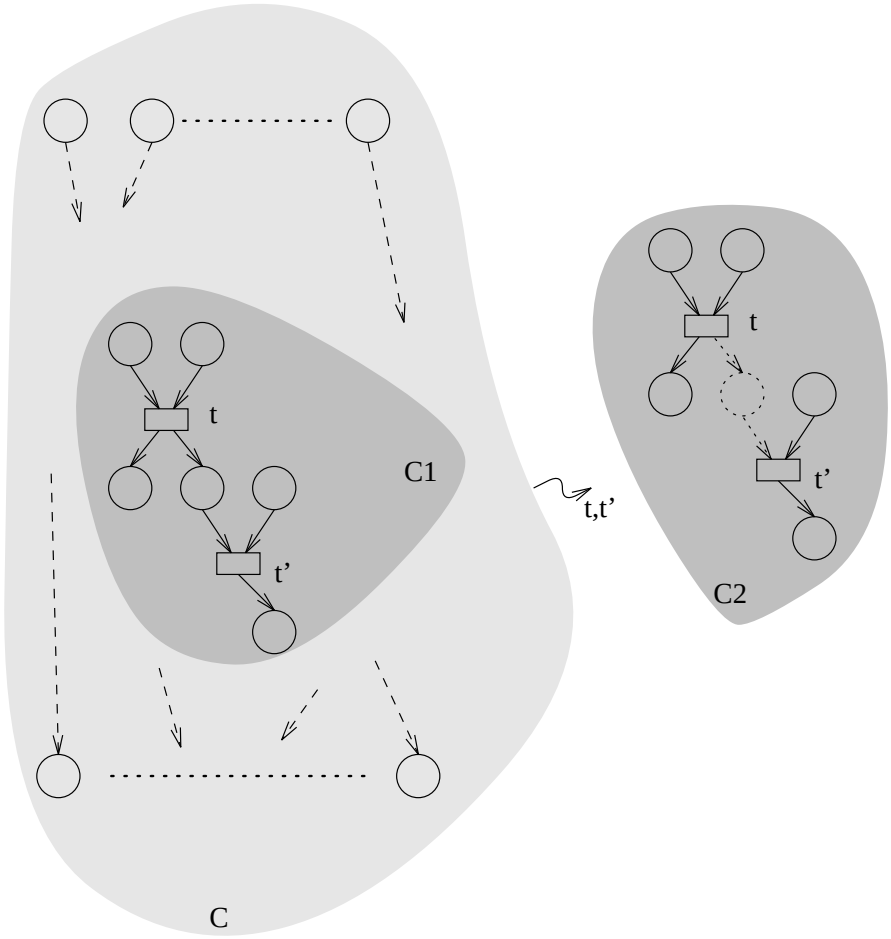
(2) $f'(C'^\circ) = M'$,

(3) $l_N(f'(T_{C'})) =_\tau l_N(\sigma)$,

(4) $T_{C'} = {}^\circ C' \cup C'^\circ$.

Le point (4) nous assure que π' correspond à un pas concurrent U dans N. Les points (1) et (2) que les pré- et post-conditions de σ contiennent celles de U, et le point (3) que $l_N(\sigma) =_\tau l_N(U)$.

π' est construit en modifiant π. Si π satisfait (4), alors c'est déjà un pas concurrent. Sinon, il existe deux transitions $t, t' \in T_C$ et une place $p \in P_C$ telles que $p \in (t^\bullet \cap {}^\bullet t')$. Considérons le réseau causal C_1 extrait de C en se restreignant aux places $P_{C_1} = {}^\bullet t^\bullet \cup {}^\bullet t'^\bullet$ et aux transitions $T_{C_1} = \{t, t'\}$. Le processus (C_1, f_1) correspond à un pas $M_1 \stackrel{t.t'}{\rightarrow} M_1'$ dans N. Puisque N est $2p$-saturé, il existe un pas concurrent $M_1 \stackrel{V}{\rightarrow} M_1'$, qui correspond à un processus concurrent $\pi_2 = (C_2, f_2)$.



Nous pouvons choisir ${}^\circ C_2 = {}^\circ C_1$ et $C_2{}^\circ = C_1{}^\circ$, et remplacer (C_1, f_1) par (C_2, f_2) dans π. Nous obtenons un nouveau processus π' qui satisfait les conditions (1) – (3) et qui est clairement moins séquentiel que π. Nous pouvons ainsi répéter la même opération jusqu'à obtenir un processus qui satisfasse la condition (4), et donc qui soit un pas concurrent.

□

Nous déduisons de ce théorème :

Corollaire B.1.3. *Il existe un algorithme simple qui indique si un réseau fini est p-saturé.*

Preuve. il suffit de regarder pour chaque séquence $\sigma = t_1.t_2 \in T^\times$ si elle est couverte. Clairement, il n'est utile que de considérer les séquences où $t_1{}^\bullet \cap {}^\bullet t_2 \neq \emptyset$. Pour chaque σ, nous vérifions s'il existe $U = \{u_1, \ldots, u_n\}$ tel que ${}^\bullet\sigma \stackrel{U}{\to} \sigma^\bullet$ et $l(U) =_\tau l(\sigma)$. Comme ${}^\bullet t_1$ et ${}^\bullet t_2$ sont finis, n est borné par $\sum |{}^\bullet t_i| + |t_i{}^\bullet| = 4d_T$.

□

B.2 p-saturer un réseau

Comme nous savons qu'il suffit qu'un réseau soit $2p$-saturé pour être p-saturé, nous développons une méthode basée sur la saturation des séquences de longueur 2.

Définition B.2.1. *Soit $t_1.t_2 \in T^\times$ non couverte dans N. Nous notons $N \rightsquigarrow_{t_1,t_2} N'$ si N' est N auquel a été ajoutée une transition t telle que :*

$${}^\bullet t \stackrel{\text{déf}}{=} {}^\bullet(t_1.t_2), \quad t^\bullet \stackrel{\text{déf}}{=} (t_1.t_2)^\bullet \quad \textit{et} \quad l_N(t) =_\tau l_N(t_1.t_2)$$

Nous notons $N \rightsquigarrow N'$ si $N \rightsquigarrow_{t_1,t_2} N'$ pour un (t_1, t_2). Clairement, N est $2p$-saturé ssi $N \not\rightsquigarrow$ (c'est-à-dire qu'il n'existe pas de N' tel que $N \rightsquigarrow N'$).

Le fait de couvrir une séquence de transitions ne modifie pas les propriétés de τ-bisimulation d'un réseau :

Proposition B.2.2. *Si $N \rightsquigarrow N'$ alors :*

$$B_\tau(N) = B_\tau(N')$$

Par contre, une séquence de saturation $N_0 \rightsquigarrow N_1 \rightsquigarrow N_2 \rightsquigarrow \cdots$ ne converge pas forcément de $B_\tau(N)$ comme le montre la figure B.1. En effet, sur cette figure, nous avons les séquences :

$$N_1 \rightsquigarrow_{t_1,t_1} N_1^1 \rightsquigarrow_{t_2,t_2} N_1^2 \rightsquigarrow_{t_3,t_3} N_1^3 \rightsquigarrow \cdots$$

et

$$N_2 \rightsquigarrow_{t_1,t_1'} N_2^1 \rightsquigarrow_{t_2,t_2} N_2^2 \rightsquigarrow_{t_3,t_3} N_2^3 \rightsquigarrow_{t_1,t_1'} \cdots$$

qui sont équitables. Malheureusement :

$$p_1 B_{\tau p}(N_1) p_2 \quad \textit{mais} \quad p_1 \neg B_{\tau p}(N_1^1) p_2$$
$$p_1 B_{\tau p}(N_1^3) p_2 \quad \textit{mais} \quad p_1 \neg B_{\tau p}(N_1^2) p_2$$

et

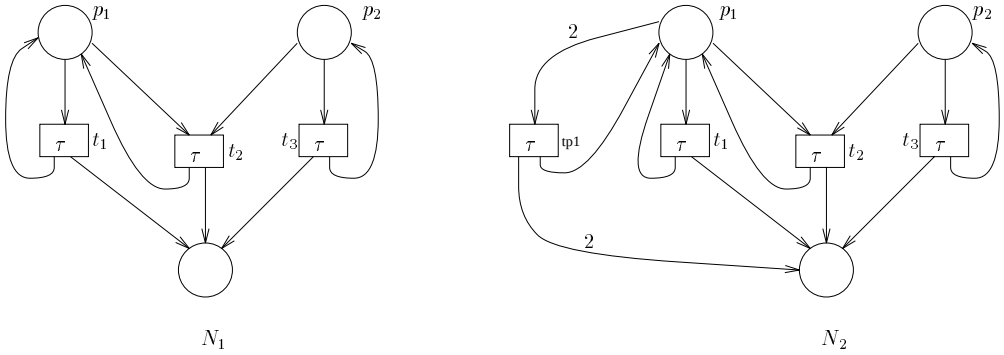
$$\forall i,\ p_1 \neg B_{\tau p}(N_2^i) p_2 \quad \textit{mais} \quad p_1 B_\tau(N_2) p_2$$

Formellement :

Proposition B.2.3. *Étant donnée une séquence de saturation $N_0 \rightsquigarrow N_1 \rightsquigarrow N_2 \rightsquigarrow \cdots$, nous avons :*

1. *$((B_{\tau p}(N_i))_{i=0,1,\ldots}$ ne converge pas forcément (fig. B.1, cas 1) ;*
2. *$\cup_{i=0,1,\ldots} B_{\tau p}(N_i)$ ne converge pas forcément vers $B_\tau(N_0)$ (fig. B.1, cas 2).*

Figure B.1 : Les stratégies ne convergent pas.



btp non inc

Définition B.2.4. *p-saturation,p-saturable*

Nous dirons qu'un réseau N^* est une p-saturation *de N si $N \rightsquigarrow \ldots \rightsquigarrow N^* \not\rightsquigarrow$, et que N est* p-saturable *si il admet une p-saturation finie.*

Le théorème 4.9.2, p. 114 et la proposition B.2.2 ont pour corollaire :

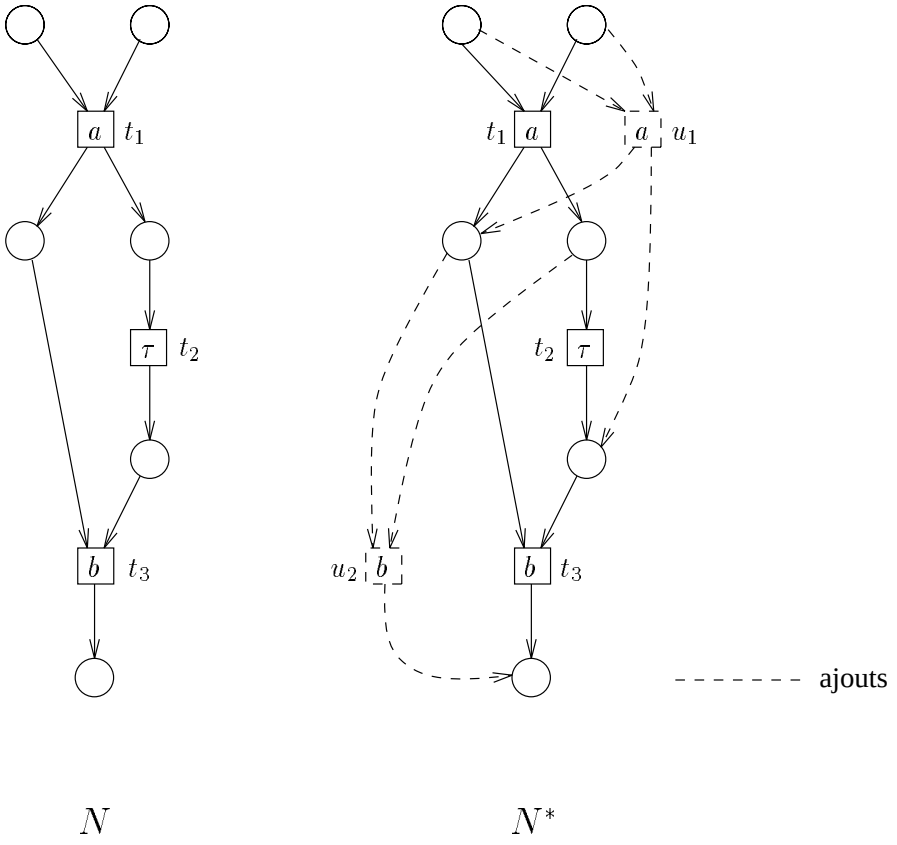
Corollaire B.2.5. *Soit N un réseau et N^* sa p-saturation :*

$$B_\tau(N) = B_{\tau p}(N^*)$$

C'est ce qui nous donne l'algorithme de calcul de $B_\tau(N)$ (algo. 4.9.1, p. 115)

La figure B.2 montre un exemple de p-saturation d'un réseau N. La p-saturation de N mène à l'ajout des transitions u_1 et u_2, qui permettent l'imitation des séquences $t_1.t_2$ et $t_2.t_3$.

Figure B.2 : N^* est N p saturé.



Le dernier point à éclaircir afin d'envisager par exemple une implémentation de l'algorithme est de trouver une méthode de p-saturation. Une approche naturelle basée sur les résultats précédents consiste à bâtir à partir de N une séquence de réseaux $N = N_0 \rightsquigarrow N_1 \rightsquigarrow N_2 \rightsquigarrow \cdots$, en examinant à chaque étape quels sont les couples (t_k, t'_k) non couverts dans N_{k-1}.

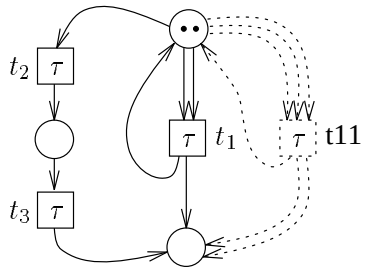
Les sections suivantes vont répondre à :

- toutes les stratégies de p-saturation sont-elles valables ?
- dans quels cas N^* est-il fini ?
- lorsque N^* est fini, quelle est sa taille ?

B.3 Stratégies de p-saturation

Nous pouvons exhiber des exemples de réseaux pour lesquels des séquences de saturation différentes mènent dans un cas à l'obtention de N^*, dans l'autre cas à la construction d'un réseau infini. Considérons l'exemple de la figure B.3. Il est basé sur celui de la figure B.4. Une séquence consistant à saturer N en saturant la séquence $t_1.t_1$ par une transition t_1^1, puis $t_1.t_1^1$ par t_1^2 etc...mène comme nous venons de le voir à un réseau infini. Par contre, il suffit de construire une transition t'_2 qui couvre la séquence $t_2.t_3$ pour saturer N.

Figure B.3 : Certaines stratégies ne permettent pas de saturer N



N

Nous définissons une *stratégie de saturation* comme un critère permettant de choisir (éventuellement de façon non-déterministe) une façon d'étendre une séquence $N_0 \rightsquigarrow_{t_1.t'_1} N_1 \rightsquigarrow_{t_2.t'_2} \ldots \rightsquigarrow_{t_k.t'_k} N_k$ par un $N_k \rightsquigarrow_{t_{k+1}.t'_{k+1}} N_{k+1}$ lorsque N_k n'est pas saturé. Nous dirons que :

Définition B.3.1. *Stratégie complète*

Une stratégie de saturation est complète *quand elle ne mène jamais à une séquence de saturation infinie à partir d'un réseau N p-saturable.*

Clairement, l'exemple précédent montre que toutes les stratégies ne sont pas complètes. Par contre, dans cet exemple, il est clair que la séquence de saturation est infinie parce que l'on repousse indéfiniment la couverture de la séquence $t_2.t_3$. Nous dirons que :

Définition B.3.2. *Stratégie équitable*
Une séquence de saturation $N_0 \rightsquigarrow N_1 \cdots \rightsquigarrow N_k \cdots$ est équitable *si il n'existe pas de paire $t_k.t'_k$ de N_k qui reste non couverte dans cette séquence. Une stratégie est* équitable *ssi elle ne produit que des séquences équitables.*

Nous avons le théorème suivant dont la preuve est donnée dans la section B.3.1 :

Théorème B.3.3. *Toutes les stratégies de p-saturation équitables sont complètes.*

B.3.1 Preuve du théorème B.3.3

Nous procéderons en plusieurs étapes, en introduisant quelques définitions intermédiaires.

Proposition B.3.4. *Si une séquence maximale de saturation $N_0 \rightsquigarrow N_1 \rightsquigarrow \cdots$ est équitable, alors il n'existe pas de $\sigma \in T_k^\times$ qui ne soit pas couverte dans tous les N_m, $m \geq k$.*

Preuve. La séquence $N_0 \rightsquigarrow N_1 \ldots$ est construite en ajoutant à chaque N_i une transition par rapport à N_{i-1}. Appelons N_ω la limite de cette séquence. N_ω peut avoir un nombre infini de transitions. La séquence de saturation considérée est équitable, donc N_ω est $2p$-saturé, et donc p-saturé (théorème B.1.2). Donc toute séquence σ est couverte sans N_ω, et donc dans un N_m.

□

Définition B.3.5. *Étant donnés deux réseaux N_1 et N_2 avec le même ensemble de places, nous dirons que N_1 est* moins p-saturé *que N_2, noté $N_1 \preceq N_2$, ssi :*

1. *pour tout $U_1 \in \mathcal{M}(T_1)$, il existe un $U_2 \in \mathcal{M}(T_2)$ tel que $U_1 \sqsubseteq_\tau U_2$, et*
2. *pour tout $\sigma_2 \in T_2^\times$, il existe un $\sigma_1 \in T_1^\times$ tel que $\sigma_2 \sqsubseteq_\tau \sigma_1$.*

Le point (1) signifie clairement que N_1 ne peut pas avoir de comportements (même concurrents) qui ne soient imités par N_2. Le point (2) assure que N_2 ne peut se distinguer de N_1 par l'existence de séquences de transitions qui n'ont pas d'équivalents dans N_1.

Proposition B.3.6. *$\preceq$ est un pré-ordre.*

par définition de $\preceq$.

L'appellation «moins p-saturé» se justifie par les deux propositions suivantes :

Proposition B.3.7. *Si $N \rightsquigarrow N'$ alors $N \preceq N'$.*

Preuve. Supposons que $N \rightsquigarrow_{t_1.t_2} N'$ avec $T' = T + \{t\}$. Puisque $T \subseteq T'$, le point (1) de la définition de $\preceq$ est satisfait. (2) est également satisfait puisque $\xrightarrow{t} = \xrightarrow{t_1.t_2}$.

□

Proposition B.3.8. *Si $N \preceq N'$ et N est p-saturé, alors N' est τp-saturé.*

Preuve. Supposons que σ' est une séquence dans $T'^{\times}$. Alors il existe une séquence σ dans $T^{\times}$ telle que $\sigma' \sqsubseteq_\tau \sigma$. Il existe donc un $U \in \mathcal{M}(T)$ tel que $\sigma \sqsubseteq_\tau U$ (puisque N est p-saturé), et donc un $U' \in \mathcal{M}(T')$ tel que $U \sqsubseteq_\tau U'$. Finalement, $\sigma' \sqsubseteq_\tau U'$.

□

Nous pouvons maintenant passer à la preuve du théorème B.3.3.

Preuve. Supposons que N est p-saturable. Il existe donc une séquence de saturation $N = N_0 \rightsquigarrow N_1 \cdots \rightsquigarrow N_k = N^*$. Supposons que $N = N'_0 \rightsquigarrow N'_1 \rightsquigarrow \cdots$ soit une autre séquence maximale. Nous allons montrer que si cette séquence est équitable, alors elle permet d'obtenir un réseau τp-saturé.

Nous prouvons par induction que nous pouvons extraire de la séquence $N'_0, N'_1 \ldots$ une sous-séquence $N'_{i_0}, N'_{i_1}, \ldots, N'_{i_k}$ telle que $N_j \preceq N'_{i_j}$ pour $j = 1, \ldots, k$. Tout d'abord, posons $i_0 = 0$. Clairement, $N_0 \preceq N'_0$.

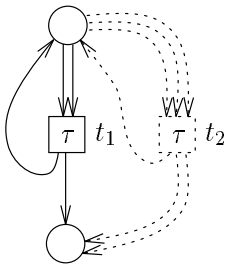
Supposons que $N_j \preceq N'_{i_j}$ et $N_j \rightsquigarrow_{t.t'} N_{j+1}$, et notons t'' la transition telle que $t'' \equiv t.t'$. Puisque $N_j \preceq N'_{i_j}$, il existe un U et un U' dans $\mathcal{M}(T'_{i_j})$ tels que $t \sqsubseteq_\tau U$ et $t' \subseteq_\tau U'$. Soit σ une séquentialisation donnée de $U.U'$, telle que $U.U' \sqsubseteq_\tau \sigma$. La proposition B.3.4 implique que σ est couverte par un V dans un réseau N'_m (puisque la séquence des N'_i est équitable). Posons $i_{j+1} = m$.

Maintenant, considérons un $S \in \mathcal{M}(T_{j+1})$. S est de la forme $S_0 + a \cdot \{t''\}$, où $S_0 \in \mathcal{M}(T_j)$ et $a \in \mathbb{N}$. Puisque $N_j \preceq N'_{i_j}$, il existe un $V_0 \in \mathcal{M}(T'_{i_j})$ tel que $S_0 \sqsubseteq_\tau V_0$. Nous savons que $t'' \sqsubseteq_\tau V$ donc $U \sqsubseteq_\tau (V_0 + a \cdot V)$ avec $V_0 + a \cdot V \in \mathcal{M}(T'_{i_{j+1}})$. Finalement, $N_{j+1} \preceq N'_{i_{j+1}}$.

Il suffit de remarquer que $N_k \preceq N'_{i_k}$ implique que N'_{i_k} est p-saturé (proposition B.3.8) pour conclure la preuve.

□

Figure B.4 : N n'est pas p-saturable.



N

B.4 p-saturabilité

Le réseau de la figure B.4 montre un réseau N qui ne peut être p-saturé. La séquence $M \stackrel{t_1.t_1}{\rightarrow} M'$ ne peut être imitée par un ensemble concurrent. Il nous faut ajouter une transition t_2 telle que ${}^\bullet t_2 = {}^\bullet(t_1.t_1) = \{p_1, p_1, p_1\}$ et $t_2{}^\bullet = (t_1.t_1)^\bullet = \{p_1, p_2, p_2\}$. Notons N' ce réseau défini par $N \rightsquigarrow_{t_1.t_1} N'$. Dans N', il est possible de tirer la séquence $t_1.t_2$, qui ne peut être imitée par un ensemble parallèle. D'une façon générale, si nous notons t^k la séquence constituée de la transition t tirée k fois successives, et kp le marquage de la place p par k jetons, alors $({}^\bullet t_1)^k = (k+1)p_1$ et $(t_1)^{k\bullet} = p_1 + kp_2$. Il n'est pas possible d'imiter ${t_1}^{k+1}$ comme une somme des t_i, $i \leq k$, et ce pour tout k. Il n'existe donc pas de réseau fini qui soit une p-saturation de N.

Nous allons donc chercher à quelle condition un réseau est p-saturable. L'exemple ci-avant nous incite à penser que les réseaux *acycliques* le sont. C'est le corollaire B.4.7, p. 211.

Nous donnons ensuite une condition nécessaire et suffisante pour qu'un *circuit* soit p-saturable (théorème B.4.10, p. 212).

Enfin, nous donnons quelques résultats sur la p-saturabilité dans le cas général.

Avant d'aborder ces différents résultats, nous présentons deux théorèmes importants quant à la réduction du problème de la p-saturabilité.

B.4.1 Se ramener aux séquences connexes de N_τ

Nous procédons en deux étapes. Tout d'abord, nous allons considérer le réseau réduit à ses τ-transitions :

Définition B.4.1. N_τ

Étant donné un réseau étiqueté $N = (P, T, W)$, nous noterons N_τ le réseau obtenu en restreignant N à l'ensemble $T_\tau \stackrel{\text{déf}}{=} \{t \in T \ : \ l(t) = \tau\}$ des transitions étiquetées par τ.

Nous dirons que N est un τ-réseau *ssi $N = N_\tau$.*

Il découle de cette définition le premier résultat suivant :

Théorème B.4.2. *N est p-saturable ssi N_τ est p-saturable.*

Preuve.

$\boxed{\Rightarrow}$ découle simplement du fait qu'une p-saturation de N est une p-saturation de N_τ.

$\boxed{\Leftarrow}$ Remarquons tout d'abord que N_τ est p-saturable ssi il est τp-saturable. Considérons N tel que N_τ est τp-saturé. Alors toute séquence

$$\sigma = u_1 \ldots u_k.t.u_{k+1} l_\tau(\sigma) = l(t) \in \mathcal{A}$$

peut se réécrire comme $\{\mu_1, \mu_1'\}.t.\{mu_2, \mu_2'\}$ avec $\mu_i, \mu_i' \in \mathcal{M}(T_\tau)$ tels que $\{\mu_1.a.\mu_2, \mu_1'.\mu_2'\} \sqsubseteq \sigma$. Soit $\{\mu_1.a.\mu_2, \mu_3\}$ puisque N_τ est τp saturé. Nous avons $|\mu_i| \leq d_T$, $\forall i$. Donc nous aurons au plus $|T_\tau|^{2d_T}$ paires μ_1, μ_2 pour chaque transition visible et par suite au plus $\mathcal{O}(n_T^{2d_T+1})$ séquences à couvrir pour p-saturer N.

□

Remarque : de la preuve, nous pouvons déduire que la taille de N^* est en $\mathcal{O}(n_T^{2b+1})$ pour les familles de réseaux dont les degrés sont bornées par la constante b lorsque N_τ est τp-saturé. Nous nous servirons de cela pour estimer la taille de N^* dans la section B.5

◇

Un second résultat nous permet de simplifier encore les résultats. Il repose simplement sur le fait que la p-saturation d'un pas est équivalente à celle de ses éléments. Formellement :

Définition B.4.3. *Séquences connexes, $\mathcal{K}_N$*

Étant donnés un réseau $N = (P, T, W, l)$ et une séquence $\rho \in T^*$, nous dirons que ρ est connexe *ssi tous ses processus le sont.*

Nous noterons $\mathcal{K}_N$ l'ensemble des séquences connexes de T^*.

Nous avons alors :

Théorème B.4.4. *N est p-saturable ssi $\mathcal{K}_N$ est p-saturable.*

Preuve.

$\boxed{\Rightarrow}$ vient de $\mathcal{K} \subseteq T^*$.

$\boxed{\Leftarrow}$ Il suffit de montrer que toute séquence $\rho \in T^*$ a la même p-saturation qu'une séquence $\rho' \in \mathcal{K}_N^*$. Soit $\rho \in T^*$ telle que $\rho \notin \mathcal{K}_N$. Alors il existe un processus π de ρ tel que π se plonge dans $\mu \in \mathcal{M}(\mathcal{K}_N)$ (trivialement par l'absurde). Donc $\rho \overset{\circ}{\equiv} \mu$ et elles ont la même p-saturation (puisqu'elles ont le même shunt).

□

Dans la suite, nous pouvons donc nous restreindre à l'étude des τ-séquences connexes.

B.4.2 p-saturabilité dans les réseaux acycliques

Nous allons montrer que «N acyclique» est équivalent à «$\mathcal{K}_N$ borné». Intuitivement, il nous suffit de montrer que les séquences de N ont une longueur bornée.

Théorème B.4.5. *Un réseau N est acyclique ssi $\mathcal{K}_N$ est de taille bornée (en fonction de la taille et du degré de N.*

Preuve.

$\boxed{\Leftarrow}$ est assez évident. Supposons que N ait un cycle σ, alors les séquences $\sigma, \sigma^2, \sigma^3, \ldots$ sont toutes connexes (elles partagent les jetons régénérés), et par suite elles appartiennent à $\mathcal{K}_N$ qui est alors infini.

$\boxed{\Rightarrow}$ est plus long à prouver. Nous avons besoin du résultat intermédiaire suivant :

Lemme B.4.6. *Soit $N = (P, T, W)$ un réseau acyclique, pour tous $t_1, t_2 \in T$, il existe $b_1, b_2 \in \mathbb{N}$ tels que :*

$$(k_1 > b_1 \vee k_2 > b_2) \Rightarrow \left(t_1^{k_1}.t_2^{k_2} \equiv \left\{\{t_1^{b_1}.t_2^{b_2}\}^k,\ t_1^{k'_1}.t_2^{k'_2},\ \{t_1\}^{k''_1},\ \{t_2\}^{k''_2}\right\}\right)$$

avec $k'_1.k'_2 < b_1.b_2$.

Preuve. Dans toute cette démonstration, nous utilisons implicitement le fait que N est acyclique, i.e. que $t_1{}^\bullet \cap {}^\bullet t_2 \neq \emptyset \Rightarrow {}^\bullet t_1 \cap t_2{}^\bullet = \emptyset$. Soient donc deux transitions t_1, t_2 telles que $t_1{}^\bullet \cap {}^\bullet t_2 \neq \emptyset$. Nous définissons :

$$D^>(t_1, t_2) \overset{déf}{=} \{p \in t_1{}^\bullet \cap {}^\bullet t_2 \ :\ t_1{}^\bullet(p) > {}^\bullet t_2(p)\}$$

- Le cas $D^>(t_1, t_2) = \emptyset$ est trivial ($b_1 = b_2 = 0$) et nous avons :

$$t_1^{k_1}.t_2^{k_2} \equiv \left\{\{t_1\}^{k_1},\ \{t_2\}^{k_2}\right\}$$

- Considérons le cas le plus simple : $D^{>}(t_1, t_2) = \{p\}$. Nous posons :

$$(b_1, b_2) \stackrel{déf}{=} \min_{x_1, x_2 > 0} \{x_1.t_1{}^{\circ}(p) = x_2.{}^{\circ}t_2(p)\}$$

Le couple (b_1, b_2) est donc celui des plus petits entiers tels que tous les jetons produits par $t_1^{b_1}$ dans la place p soient consommés par $t_2 b_2$.

Intuitivement, la séquence $t_1^{b_1}.t_2^{b^2}$ est la plus séquentielle. Nous avons alors :

$$t_1^{k_1}.t_2^{k_2} \equiv \left\{ \{t_1^{b_1}.t_2^{b_2}\}^k,\ t_1^{k'_1},\ t_2^{k'_2},\ \{t_1\}^{k''_1},\ \{t_2\}^{k''_2} \right\}$$

où :

$$\begin{aligned} k &\stackrel{déf}{=} min(k_1\, div\, b_1,\ k_2\, div\, b_2) \\ k'_i &\stackrel{déf}{=} (k_i - k.b_i)\ mod\, b_i,\ i = 1, 2 \\ k''_i &\stackrel{déf}{=} k_i - k.b_i - k'_i,\ i = 1, 2 \end{aligned}$$

De fait, nous avons bien $k_i = k.b_i + k'_i + k''_i$, $i = 1, 2$. Il nous reste à montrer l'équivalence de la séquence et du pas, ce qui ce fait en montrant que chaque élément du pas est indépendant des autres (i.e. qu'il n'a pas besoins des jetons qu'ils produisent).

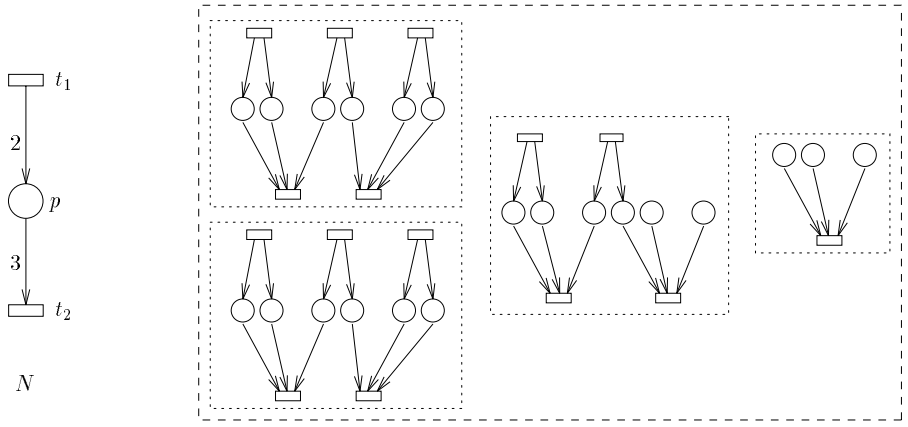
Le calcul de k implique $k'_1 = k_1 - k.b_1$ ou $k'_2 = k_2 - k.b_2$, par conséquent $k''_1.k''_2 = 0$. De plus, nous vérifions bien $k'_1.k'_2 < b_1, b_2$.

Enfin, N étant acyclique et grâce à notre construction, nous avons :

$$\begin{aligned} (t_1^{b_1}.t_2^{b_2})^+ \cap t_2^- &= \emptyset \\ (t_1^{b_1}.t_2^{b_2})^- \cap t_1^+ &= \emptyset \\ (t_1^{k'_1}.t_2^{k'_2})^+ \cap t_2^- &= \emptyset \\ (t_1^{k'_1}.t_2^{k'_2})^- \cap t_1^+ &= \emptyset \end{aligned}$$

ce qui nous garantit la concurrence.

La figure ci-dessous explicite quelque peu cette construction. Nous donnons le processus correspondant à la décomposition de la séquence $t_1^8.t_2^7$ sous la forme $\{\{t_1^3.t_2^2\}^2,\ t_1^2.t_2^2,\ \{t_2\}\}$.



- Avant d'aborder le cas général, nous allons montrer comment nous passons au cas $D^{>}(t_1, t_2) = \{p_1, p_2\}$.

  L'idée, dans notre contexte, est d'utiliser deux fois la méthode précédente : une fois pour p_1 et une autre pour p_2.

  Nous définissons donc comme précédemment des bornes b_{11}, b_{12} basées sur $t_1{}^{\bullet}(p_1)$ et ${}^{\bullet}t_2(p_1)$. Nous obtenons donc une séquence $\rho = t_1^{b_{11}}.t_2^{b_{12}}$ telle que ${}^{\bullet}\rho(p_1) = \rho^{\bullet}(p_1) = 0$. Nous avons alors trois cas :

  1. Si ${}^{\bullet}\rho(p_2) = \rho^{\bullet}(p_2) = 0$, alors nous avons :

     $$t_1^{k_1}.t_2^{k_2} \equiv \left\{ \{t_1^{b_1}.t_2^{b_2}\}^{k}, t_1^{k'_1}.t_2^{k'_2}, \{t_1\}^{k''_1}, \{t_2\}^{k''_2} \right\}$$

     où :

     $$\begin{aligned} b_i &\stackrel{déf}{=} b_{1i},\ i = 1,2 \\ k &\stackrel{déf}{=} min(k_1\, div\, b_1,\ k_2\, div\, b_2) \\ k'_i &\stackrel{déf}{=} (k_i - k.b_i)\ mod\, b_i,\ i = 1,2 \\ k''_i &\stackrel{déf}{=} k_i - k.b_i - k'_i,\ i = 1,2 \end{aligned}$$

     et le résultat est obtenu.

  2. Sinon, nous avons les deux cas symétriques et *exclusifs* (N étant acyclique) : soit ${}^{\bullet}\rho(p_2) > 0$, soit $\rho^{\bullet}(p_2) > 0$. Nous ne traitons que le premier cas, le second s'en déduisant trivialement.

     Nous pouvons à nouveau calculer les bornes b_{21}, b_{22} pour t_1 et ρ, puisque $D^{>}(t_1, \rho) = \{p_2\}$. Nous obtenons aisément en

combinant les deux couples de bornes :

$$\begin{aligned} t_1^{k_1}.t_2^{k_2} &\equiv \left\{\{t_1^{b_{21}}.\rho^{b_{22}}\}^k, t_1^{k'_{11}}, \rho^{k'_{22}}, \{t_1\}^{k''_1}, \{t_2\}^{k''_2}\right\} \\ &\equiv \left\{\{t_1^{b_1}.t_2^{b_2}\}^k, t_1^{k'_1}, t_2^{k'_2}, \{t_1\}^{k''_1}, \{t_2\}^{k''_2}\right\} \end{aligned}$$

et le résultat est obtenu.

- Le cas général se traite de la même manière.

Il nous reste à prouver que les places hors de $D^>(t_1, t_2)$ n'influent pas sur le résultat. Ceci est trivial puisque cela signifie que le tir d'une seule transition t_1 suffit à produire suffisamment de jetons pour le tir d'une transition t_2 (on aurait donc une dépendance du type $(b_1, b_2) = (1, n)$).

□

Ce lemme nous permet donc de dire que la longueur des séquences de $\mathcal{K}_N$ est bornée, N étant fini, $\mathcal{K}_N$ est de taille bornée.

□

Nous avons alors aisément le corollaire suivant s'appuyant sur le théorème B.4.4 :

Corollaire B.4.7. *Si N est un réseau fini et acyclique, alors N est p-saturable.*

Ce résultat nous permet de considérer la très vaste classe des réseaux sans $\tau - cycles$. Est-il possible de l'étendre ? Nous allons présenter ci-après une méthode qui permet de se ramener à l'existence de séquences particulières (les *shunts*).

B.4.3 *p*-saturabilité d'un circuit

Définition B.4.8. *Circuit*

Soit $N = (P, T, W)$ un réseau et $\sigma \in T^*$ une séquence. Nous dirons que σ est un circuit *ssi $N/_{\mathcal{S}_\sigma}$ (N restreint au support de σ) est cyclique.*

Définition B.4.9. *Équivalence de bilan, shunt*

Soit $N = (P, T, W)$ un réseau. Nous dirons que deux séquences ont le même bilan, *noté $\rho \overset{\circ}{\equiv} \rho'$, ssi $\rho^+ = \rho'^+$ et $\rho^- = \rho'^-$, où $\rho^+ \overset{\text{déf}}{=} \rho^\bullet - {}^\bullet\rho$ (resp. $\rho^- \overset{\text{déf}}{=} {}^\bullet\rho - \rho^\bullet$). Intuitivement, ρ^- (resp. ρ^+) représentent les jetons* réellement *consommés (resp. produits).*

Trivialement, $\overset{\circ}{\equiv}$ est une relation d'équivalence sur T^*.

D'autre part, si μ est une p-saturation de ρ, alors $\mu \overset{\circ}{\equiv} \rho$.

$\xi \in T^*$ est un shunt *de ρ ssi $\exists k \in \mathbb{N} : \xi \overset{\circ}{\equiv} \rho^k$ et ${}^\bullet\xi \cap \xi^\bullet \neq \emptyset$, ce qui équivaut à $(\rho^k)^- \overset{\xi}{\rightarrow} (\rho^k)^+$.*

Nous avons le théorème suivant :

Théorème B.4.10. *Soit $N = (P, T, W)$ un réseau et $\rho \in T^*$ une séquence.* $\mathbf{R} \stackrel{\text{déf}}{=} \{\rho, \rho^2, \ldots\}$ *est p-saturable ssi ρ est shuntée.*

Preuve.

- Le cas où ρ n'est pas un circuit est trivial puisqu'alors $\rho^k \equiv \{\rho\}^k$.
- Si ρ est un circuit :

  $\boxed{\Rightarrow}$ Nous allons montrer que si les ρ^k sont p-saturés par un ensemble *fini* $\Xi_\rho \in T^*$ de transitions alors ρ est shunté.
  Nous avons :

  $$\begin{array}{l} \rho \text{ est } p\text{-saturé par } \mu_1 \in \mathcal{M}(\Xi_\rho) \\ \rho^2 \text{ est } p\text{-saturé par } \mu_2 \in \mathcal{M}(\Xi_\rho) \\ \text{etc} \end{array}$$

  On a donc $\mu_i = \mu_i^\sigma + \mu_i^\xi$, où μ_i^σ regroupe toutes les séquences de μ_i qui sont un circuit, et μ_i^ξ regroupe les autres. En particulier, μ_i^ξ est son propre shunt. Nous étendons trivialement cette notation à Ξ_ρ^ξ et Ξ_ρ^σ.
  Par ailleurs, comme $\forall t \in \Xi_\rho$, ${}^\bullet t \cap t^\bullet \subseteq {}^\bullet\rho \cap \rho^\bullet$ (puisque $\forall k > 0$, ${}^\bullet(\rho^k) \cap (\rho^k)^\bullet = {}^\bullet\rho \cap \rho^\bullet$), chaque $t \in \Xi_\rho^\sigma$ est utilisable un nombre fini de fois dans chaque μ_i.
  Nous pouvons donc trouver $k_1 < k_2$ tels que $\mu_{k_1}^\sigma = \mu_{k_2}^\sigma$. Donc $\rho^{k_2} \stackrel{\circ}{\equiv} \rho^{k_1}.\rho^{k_2-k_1}$ est p-saturé par $\mu_{k_1} + \mu$ avec μ acyclique et $\mu \stackrel{\circ}{\equiv} \rho^{k_2-k_1}$. Donc par définition, μ est un shunt pour ρ.

  $\boxed{\Leftarrow}$ il suffit de montrer que si ρ est shuntée, alors $\mathbf{R}$ est p-saturé par un Ξ_ρ fini.

  Il suffit de constater que le shunt ξ de ρ vérifie $\xi \stackrel{\circ}{\equiv} \rho^k$, nous avons alors :

  $$\begin{array}{rcl} \rho^n & \equiv & \rho^{n \bmod k}.(\rho^k)^{n \operatorname{div} k} \\ & \equiv & \rho^{n \bmod k}.\xi^{n \operatorname{div} k} \\ & \equiv & \left\{\rho^{n \bmod k}, \{\xi\}^{n \operatorname{div} k}\right\} \end{array}$$

  puisque ${}^\bullet\xi \cap \rho^\bullet = {}^\bullet\rho \cap \xi^\bullet = \emptyset$.
  Donc l'ensemble de transitions

  $$\Xi_\rho \stackrel{déf}{=} \{t_i,\ i = 1...k\ :\ \forall j = 1, \ldots, k-1,\ t_j \equiv \rho^j \wedge t_k \equiv \xi\}$$

  est tel que pour tout entier n :

  $$\rho^n \equiv \left\{t_{n \bmod k}, \{t_k\}^{n \operatorname{div} k}\right\}$$

  Donc Ξ_ρ est fini et p-sature $\mathbf{R}$.

□

Un premier résultat en découle :

Corollaire B.4.11. *Un réseau N est p-saturable ssi $\mathcal{K}_N$ est shunté par un ensemble fini.*

Il nous reste donc à décider de ce que $\mathcal{K}_N$ est shunté. Nous allons tout d'abord montrer que l'existence d'un shunt pour un circuit est décidable.

B.4.4 Décidabilité de l'existence d'un shunt

Nous avons le théorème suivant :

Théorème B.4.12. *Soit $N = (P, T, W)$ un réseau et $\rho \in T^*$. Il est décidable s'il existe un entier $k > 0$ tel que $(\rho^k)^+$ soit atteignable depuis $(\rho^k)^-$.*

Preuve. Nous utilisons une construction inspirée par [Had] et décrite par la figure B.5. L'existence d'un tel k revient à l'atteignabilité dans le réseau N' de la figure B.5 du marquage $\{fin\}$ depuis le marquage[1] $\rho^- + \{compteur, début\}$. Et ceci est décidable ([May81, May84]).

Nous obtenons N' en ajoutant à N :

- les places *début*, *en-cours*, *fin* et *compteur* ;
- les transitions :

$$\begin{aligned} \{début\} &\xrightarrow{\texttt{init}} \rho^- + \{début\} + \{compteur\} \\ \{début\} &\xrightarrow{\texttt{démar}} \{en\text{-}cours\} \\ \{en\text{-}cours\} &\xrightarrow{\texttt{arrêt}} \{fin\} \\ \rho^+ + \{fin\} + \{compteur\} &\xrightarrow{\texttt{final}} \{fin\} \end{aligned}$$

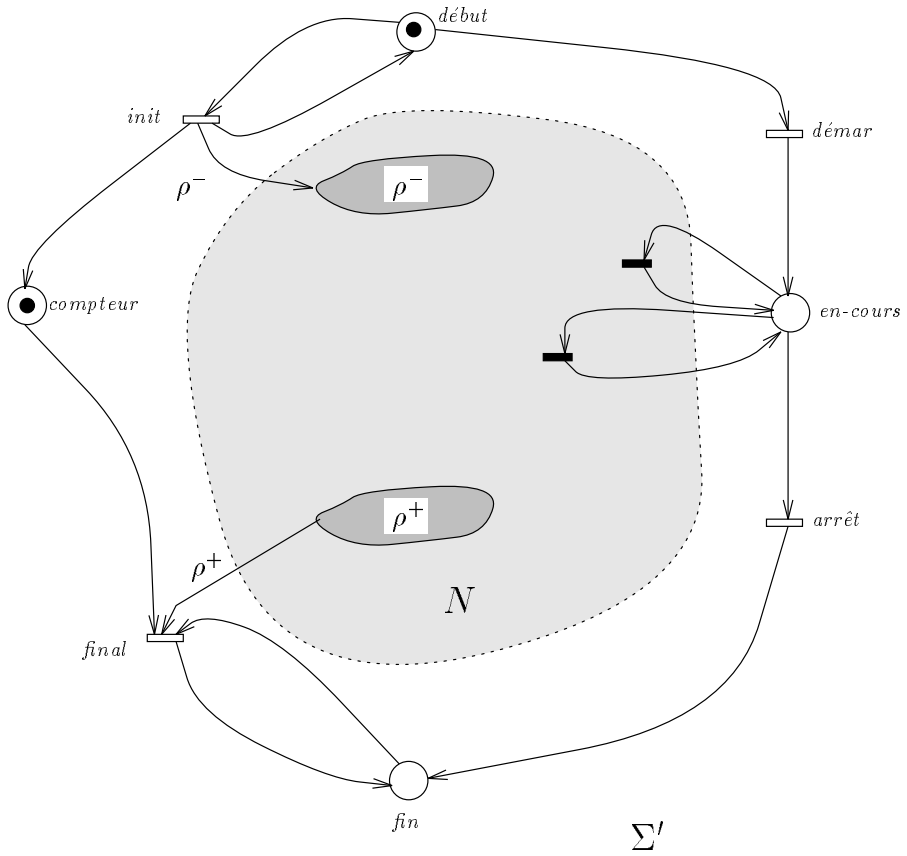
- les arcs $en\text{-}cours \overset{1}{\frown} t$ et $t \overset{1}{\frown} en\text{-}cours$ pour que chaque chaque transition $t \in T$ boucle sur la place *en-cours*.

De plus, nous connaissons la valeur k_1 et le shunt ξ_1, puisque l'algorithme d'atteignabilité donne une séquence.

□

[1] $\rho^- + \{début\}$ ne servent qu'à assurer que l'on a fait au moins une fois $\rho^- \to \rho^+$.

Figure B.5 : De l'atteignabilité aux shunts.



B.4.5 La *p*-saturabilité en général

Le théorème précédent a immédiatement comme conséquence :

Corollaire B.4.13. *La p-saturabilité est co-semi-décidable.*

Preuve. Il suffit de générer les séquences connexes de plus en plus longues. Dès qu'il en existe une non shuntée, N n'est pas p-saturable.

□

Le problème est de savoir si, par exemple, nous pourrions définir dans N une *base minimale de circuits*, à l'analogue de ce qui se fait dans les graphes. Je n'ai malheureusement pas trouvé ce résultat dans la littérature.

Toutefois nous avons comme conjecture forte que la p-saturabilité est décidable. L'idée est la suivante :

- Nous voudrions montrer que, même s'il existe des cycles, en faisant abstraction des jetons régénérés, les transitions ont une séquentialité bornée.
- Nous utilisons donc le théorème B.4.5, une fois pour déterminer des bornes pour $t_1^{b_1}.t_2^{b_2}$ et une autre pour $t_2^{b'_2}.t_1^{b'_1}$.
- Ensuite, il reste à montrer que :

$$\begin{aligned}
(t_1^{b_1.b'_1.b'_2}.t_2^{b_2.b'_1.b'_2})^+ \cap t_1^- &= \emptyset \\
(t_1^{b_1.b'_1.b'_2}.t_2^{b_2.b'_1.b'_2})^+ \cap t_2^- &= \emptyset \\
(t_1^{b_1.b'_1.b'_2}.t_2^{b_2.b'_1.b'_2})^- \cap t_1^+ &= \emptyset \\
(t_1^{b_1.b'_1.b'_2}.t_2^{b_2.b'_1.b'_2})^- \cap t_2^+ &= \emptyset
\end{aligned}$$

  Comme par ailleurs l'équivalence de bilan nous permet de nous ramener à $\mathcal{KO}_N$[2] on a que toute séquence dont les coefficients sont supérieurs aux bornes ci-dessus ne sont pas fortement dépendantes (i.e. $\rho_1^+ \cap \rho_2^- = \emptyset$, ce qui implique que $\xi_1{}^\bullet \cap {}^\bullet\xi_2 = \emptyset$ où ξ_i est le shunt de ρ_i).

  On peut alors passer de deux transitions à n en conservant ses propriétés. Nous avons alors à vérifier l'existence d'un nombre fini de shunts pour montrer que $\mathcal{K}_N$ est shunté, ce qui nous donne la décidabilité de la p-saturabilité.

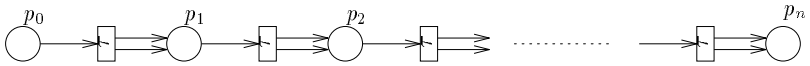
[2]Soit $\rho = u_1 \dots u_k$, nous dirons que ρ est *ordonnée* si nous pouvons l'écrire $v_1^{e_1} \dots v_m^{e_m}$ avec $v_i \in T$, $\forall i$ et $v_i = v_j \Leftrightarrow i = j$.

Trivialement, pour toute séquence ρ il existe une séquence ordonnée ρ' telle que ρ' est un *réarrangement* de ρ (i.e. $\forall t \in T,\ occ_\rho(t) = occ_{\rho'}(t)$) et que $\rho \overset{\circ}{\equiv} \rho'$. Cela vient simplement de ce que $\rho \overset{\circ}{\equiv} \{u_1, \dots, u_k\}$.

Nous notons $\mathcal{KO}_N \subseteq \mathcal{K}_N$ l'ensemble des séquences connexes ordonnées.

B.5 Déterminer la taille de N^*

Tout d'abord, il existe des réseaux très simples tels que la taille de leur p-saturation soit énorme. Le réseau ci-dessous



est p-saturable. Cependant, sa plus petite p-saturation a $S(0)+S(1)+\cdots+S(n-1)$ transitions, où $S(k)$ est le nombre de solutions entières positives de :

$$x_0 + 2x_1 + \cdots + 2^k x_k = 2^{k+1}$$

Ce problème a été étudié dans les années quarante par K. MALHER. Grossièrement, $S(k)$ croît à la même vitesse que 2^{k^2}.

Nous donnerons d'abord une caractérisation pour les réseaux acycliques, puis nous nous appuierons sur la preuve de la décidabilité de la p-saturabilité pour donner une conditions nécessaire et suffisante pour que la taille de N^* soit polynomiale en la taille de N.

Le théorème B.4.2 a comme corollaire l'équivalence entre la «polynomialité» de N^* et celle de N^*_τ.

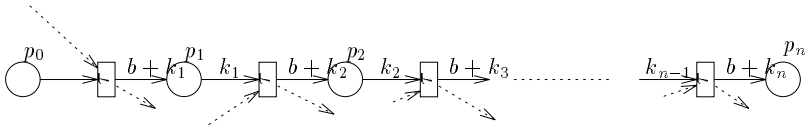
B.5.1 Les réseaux acycliques

Nous dirons qu'un réseau $N = (P, T, W)$ est *acyclique* ssi son graphe sous-jacent $\mathcal{G}(N)$ soit acyclique (i.e. $\forall x \in P \cup T, (x,x) \notin W^*$).

Lemme B.5.1. *Soit N t.q. N_τ est acyclique. Si N^* est polynomial alors :*

$$\forall t_1, t_2 \in T_\tau,\ \forall p \in t_1{}^\circ \cap {}^\circ t_2\ : t_1{}^\bullet(p) = {}^\bullet t_2(p) \tag{B.2}$$

Preuve. Supposons que (B.2) ne soit pas vérifiée, alors nous pouvons trouver un sous-réseau dans N (ou une de ses p-saturation) un sous-réseau du type :



Or nous avons vu que sa p-saturation ajoutait $\mathcal{O}(b^{n^2})$ transitions.

□

Nous introduisons diverses notions :

Définition B.5.2. *Graphe de dépendance*

Étant donné un réseau $N = (P, T, W)$, nous définissons le multi-graphe de dépendances $\mathcal{D}(N) = (S, A, e)$ par :

$$\begin{array}{lcl} A & \stackrel{\text{déf}}{=} & T \\ S & \stackrel{\text{déf}}{=} & \{a_{ij}^p = (t_i, t_j) \ : \ p \in t_i^\bullet \cap {}^\bullet t_j\} \\ e & : & A \to P, \ e(a_i j^p) = p \end{array}$$

Nous appellerons chemin maximal *un chemin qui part d'un nœud sans prédécesseur pour arriver à un nœud sans successeur.*

Nous noterons $\mathbb{P}$, la famille des graphes tels que le nombre de chemins maximaux soit polynomial en la taille de A.

Trivialement, $\mathcal{D}(N)$ est acyclique ssi N l'est.

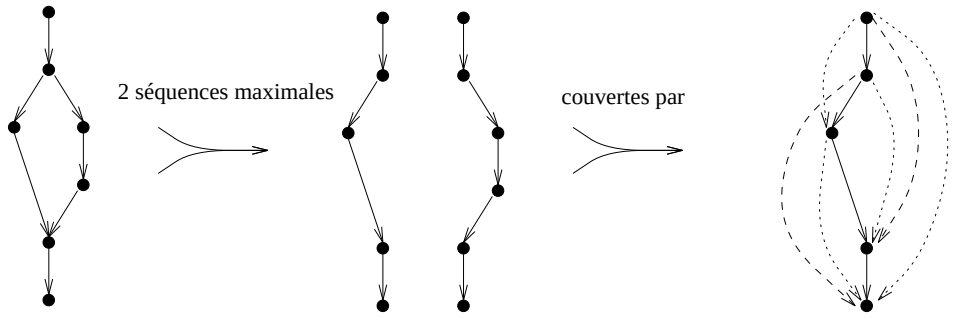
Nous avons alors le lemme suivant :

Lemme B.5.3. *Pour un réseau acyclique $N = (P, T, W)$, N^* est de taille polynomiale en n_T ssi $\mathcal{D}(N)$ est dans $\mathbb{P}$ et s'il vérifie (B.2).*

Preuve.

$\boxed{\Leftarrow}$ découle simplement du lemme B.5.1.

$\boxed{\Rightarrow}$ vient de ce que pour couvrir les séquences de T^*, N vérifiant (B.2), il suffit de couvrir les séquences de $\mathcal{D}(N)$. Or les séquences maximales sont au plus de longueurs n_T et chacune peut donner au plus $\mathcal{O}(n_T^2)$ séquences à couvrir (preuve graphique) :



Comme $\mathcal{D}(N)$ est dans $\mathbb{P}$, N^* est de taille polynomiale en la taille de T.

□

B.5.2 Le cas général

D'après le théorème sur la décidabilité de la p-saturabilité, nous savons que tout circuit de N est imité à partir d'un certain moment (k_σ) par un shunt (i.e. suite de transition dont le support n'est pas fortement connexe). Nous avons : $\exists k \in \mathbb{N} \ : \ \forall \sigma,\ k_\sigma \leq k$. Considérons le réseau N_π obtenu à partir de N en «déroulant» ses cycles jusqu'à l'ordre k. Nous avons alors un réseau N_π τ-bisimilaire à N et sans composante fortement connexe. La taille de N_π est polynomiale en fonction de celle de N avec un facteur k.

Nous avons donc comme corollaire du lemme B.5.3 :

Corollaire B.5.4. *Pour un réseau $N = (P, T, W)$, N^* est de taille polynomiale en n_T ssi $\mathcal{D}(N_\pi)$ est dans $\mathbb{P}$ et s'il vérifie (B.2).*

Bibliographie

[ABS91a] Autant (C.), Belmesk (Z.) & Schnoebelen (Ph.). – Strong bisimilarity on nets revisited (extended abstract). *In: Proc. PARLE'91, vol. II: Parallel Languages, Eindhoven, LNCS 506.* pp. 295–312. – Springer-Verlag.

[ABS91b] Autant (C.), Belmesk (Z.) & Schnoebelen (Ph.). – *Strong Bisimilarity on Nets Revisited.* – Research Report n° 847-I, Grenoble, LIFIA-IMAG, mars 1991.

[AFN92] Alimonti (P.), Feuerstein (E.) & Nanni (U.). – Linear time algorithm for liveness and boundedness in conflict-free Petri nets. *In: Proc. First Latin American Symposium on Theoretical Informatics, Sao Paulo, Brazil, LNCS 583.* pp. 1–14. – Springer-Verlag.

[Ajm89] Ajmone Marsan (M.). – Stochastic petri nets: An elementary introduction. *In: Advances in Petri Nets 1989, LNCS 424.* pp. 1–29. – Springer-Verlag.

[APS94] Autant (C.), Pfister (W.) & Schnoebelen (Ph.). – Place bisimulations for the reduction of labeled Petri nets with silent moves. *Journal of Computing and Information*, vol. 1(1), Special issue: Proceedings of the 6th Int. Conf. on Computing and Information, Trent University, Canada, mai 1994, pp. 230–246. – Proceedings available on http://blaze.trentu.ca/icci.

[AS92] Autant (C.) & Schnoebelen (Ph.). – Place bisimulations in Petri nets. *In: Proc. 13th Int. Conf. Application and Theory of Petri Nets, Sheffield, UK, LNCS 616.* pp. 45–61. – Springer-Verlag.

[Aut93] Autant (C.). – *Réseaux de Petri pour la sémantique et l'implémentation de processus parallèles.* – Thèse de Doctorat, I.N.P. de Grenoble, France, mai 1993.

[BB] Berry (G.) & Boudol (G.). – The chemical abstract machine. *Theoretical Computer Science*, no96, pp. 217–248.

[BBK87] Baeten (J. C. M.), Bergstra (J. A.) & Klop (J. W.). – Term rewriting systems with priorities. *In: Proc. Rewriting Techniques and Applications 87, Bordeaux, LNCS 256.* pp. 83–94. – Springer-Verlag.

[BC87] Boudol (G.) & Castellani (I.). – On the semantics of concurrency: Partial orders and transition systems. *In: Proc. CAAP'87, Pisa, LNCS 249.* pp. 123–137. – Springer-Verlag.

[BC92] Bernardinello (L.) & Cindio (F. De). – A survey of basic net models and modular net classes. *In: Advances in Petri Nets 1992, LNCS 609.* pp. 304–351. – Springer-Verlag.

[BD90] Best (E.) & Desel (J.). – Partial order behaviour and structure of Petri nets. *Formal Aspects of Computing*, vol. 2, n° 2, 1990, pp. 123–138.

[BDKP91] Best (E.), Devillers (R.), Kiehn (A.) & Pomello (L.). – Concurrent bisimulations in Petri nets. *Acta Informatica*, vol. 28, 1991, pp. 231–264.

[Ber83] Berthelot (G.). – *Transformations et analyse de réseaux de Petri - Applications aux protocoles.* – Thèse d'Etat, Univ. Paris 6, octobre 1983.

[Ber85] Berthelot (G.). – Checking properties of nets using transformation. *In: Advances in Petri Nets 1985, LNCS 222.* pp. 19–40. – Springer-Verlag.

[Ber86] Berthelot (G.). – Transformations and decompositions of nets. *In: Petri Nets: Central Models and Their Properties, Bad Honnef, LNCS 254.* pp. 359–376. – Springer-Verlag.

[Ber87] Berge (C.). – *Hypergraphes.* – Gauthier-Villars, 1987.

[Ber93] Berry (D.M.). – Formal specification and verification of concurrent programs. – SEI Curriculum Module, Feb 1993.

[Bes88] Best (E.). – Weighted basic Petri nets. *In: Proc. Concurrency'88, LNCS 335.* pp. 257–276. – Springer-Verlag.

[Bil88] Billington (J.). – *Extending Coloured Petri Nets.* – UK, Phd thesis, also appeared as tr 148, U. of Cambridge, Oct 1988.

[BK89] Bergstra (J. A.) & Klop (J. W.). – Process theory based on bisimulation semantics. *In: Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, Noordwijkerhout, LNCS 354.* pp. 50–122. – Springer-Verlag.

[BM93] Banâtre (J.P.) & Métayer (D. Le). – Programming by multiset transformation. *Communications of the ACM*, vol. 36, n° 1, 1993, pp. 98–111.

[Bri95] Bridges (D.S.). – *Computability - A Mathematical Sketchbook.* – Springer-Verlag, 1995, *Grad. Texts in Math.*, volume 146.

[BS87] Bolognesi (T.) & Smolka (S.A.). – Fundemental results for the verification of observational equivalence : a survey. *In: Protocol Specification, Testing and Verification 7*, éd. par Rudin (H.) & West (C.). pp. 165–179. – North Holland. Cité dans buchholz95.

[Buc95] Buchholz (P.). – A notion of equivalence for stochastic Petri nets. *In: Application and Theory of Petri Nets.* pp. 161–180. – Springer-Verlag.

[CH93a] Christensen (S.) & Hansen (N.D.). – Coloured Petri nets extended with places capacities, test arcs and inhibitor arcs. *In: Proc. 14th Int. Conf. Application and Theory of Petri Nets, Chicago, LNCS 691.* pp. 186–205. – Springer-Verlag.

[CH93b] Christensen (S.) & Hüttel (H.). – Decidability issues for infinite-state processes - a survey. *EATCS Bull.*, vol. 51, octobre 1993, pp. 156–166.

[Chr92] Christensen (S.). – A logical characterisation of asynchronously communicating agents. *In: Proc. Logical Found. Comp. Sci., Tver, Russia, LNCS 620.* pp. 93–104. – Springer-Verlag.

[CHS92] Christensen (S.), Hüttel (H.) & Stirling (C.). – Bisimulation equivalence is decidable for all context-free processes. *In: Proc. CONCUR'92, Stony Brook, NY, LNCS 630.* pp. 138–147. – Springer-Verlag.

[CMP87] Castellano (L.), Michelis (G. De) & Pomello (L.). – Concurrency vs interleaving: an instructive example. *EATCS Bull.*, vol. 31, février 1987, pp. 12–15.

[DDM87] Degano (P.), De Nicola (R.) & Montanari (U.). – Observational equivalences for concurrency models. *In: Formal Description of Programming Concepts - III, Proc. of the third IFIP WG 2.1 working conf., Ebberup 1986*, éd. par Wirsing (M.). pp. 105–129. – North-Holland.

[DDM88] Degano (P.), De Nicola (R.) & Montanari (U.). – A distributed operational semantics for CCS based on condition/event systems. *Acta Informatica*, vol. 26, 1988, pp. 59–91.

[DE95] Desel (J.) & Esparza (J.). – *Free Choice Petri Nets.* – Cambridge Tracts, 1995, *Theo. Comp. Sci.*

[Del93] Delannoy (C.). – *Programmer en langage C++.* – Eyrolles, 1993.

[Dev89] Devillers (R.). – The semantics of capacities in P/T nets. *In : Advances in Petri Nets 1989, LNCS 424.* pp. 128–150. – Springer-Verlag.

[DGV93] Degano (P.), Gorrieri (R.) & Vigna (S.). – On relating some models of concurrency. – Invited lecture to TAPSOFT 1993, 1993.

[DV90] De Nicola (R.) & Vaandrager (F.). – Three logics for branching bisimulation (extended abstract). *In : Proc. 5th IEEE Symp. Logic in Computer Science, Philadelphia, PA*, pp. 118–129.

[EN94] Esparza (J.) & Nielsen (M.). – Decidability issues for Petri nets. *EATCS*, vol. 52, 1994, pp. 245–262.

[Eng93] Engelfriet (J.). – A multiset semantics for the π-calculus with replication. pp. 7–21. – Springer-Verlag.

[Fer89] Fernandez (J.-C.). – An implementation of an efficient algorithm for bisimulation equivalence. *Science of Computer Programming*, vol. 13, 1989, pp. 219–236.

[Gen90] Genrich (H.J.). – Equivalence transformations of Pr/T nets. *In : Advances in Petri Nets 1989, LNCS 424.* pp. 179–208. – Springer-Verlag.

[Gla94] Glabbeek (R. J. van). – What is branching time semantics and why to use it ? *EATCS Bull.*, vol. 53, juin 1994, pp. 191–198.

[GS90] Garavel (H.) & Sifakis (J.). – Compilation and verification of LOTOS specifications. *In : Proc. 10th IFIP Conf. Protocol Specification, Testing and Verification, Ottawa.*

[GV87] Glabbeek (R. J. van) & Vaandrager (F.). – Petri net models for algebraic theories of concurrency. *In : Proc. PARLE'87, vol. II: Parallel Languages, Eindhoven, LNCS 259.* pp. 224–242. – Springer-Verlag.

[GV90] GROOTE (J. F.) & VAANDRAGER (F.). – An efficient algorithm for branching bisimulation and stuttering equivalence. *In: Proc. 17th ICALP, Warwick, LNCS 443.* pp. 626–638. – Springer-Verlag.

[GW89a] GLABBEEK (R. J. VAN) & WEIJLAND (W. P.). – *Branching Time and Abstraction in Bisimulation Semantics (extended abstract).* – Research Report n° CS-R8911, CWI, avril 1989.

[GW89b] GLABBEEK (R. J. VAN) & WEIJLAND (W. P.). – Branching time and abstraction in process algebra. *In: Information Processing 89, Proc. IFIP 11th World Computer Congress, San Francisco.* pp. 613–618. – North-Holland.

[GW89c] GLABBEEK (R. J. VAN) & WEIJLAND (W. P.). – Refinement in branching time semantics. *In: Proc. AMAST Conf., Iowa City*, pp. 197–201. – Available as CWI Report CS-R8922.

[Hac72] HACK (M.H.T). – *Analysis of Production Schemata by Petri Nets.* – Thèse, MIT, Project MAC, 1972.

[Hac76] HACK (M.). – *Decidability Questions for Petri Nets.* – Tech. Report n° 161, MIT, juin 1976.

[Had] HADDAD (S.). – communication personnelle.

[Had90] HADDAD (S.). – A reduction theory for coloured nets. *In: Advances in Petri Nets 1990, LNCS 483.* pp. 209–235. – Springer-Verlag.

[Hau90] HAUSHILDT (D.). – *Semilinearity of the Reachability Set is Decidable for Petri Nets.* – Research Report n° RR 146, Uni. Hamburg, FBI, 1990.

[HKT92] HOOGERS (P.W.), KLEIJN (H.C.M.) & THIAGARAJAN (P.S.). – A trace semantics for Petri nets. *In: Proc. 19th ICALP, Vienna, LNCS 623.* pp. 595–604. – Springer-Verlag.

[HM80] HENNESSY (M.) & MILNER (R.). – On observing nondeterminism and concurrency. *In: Proc. 7th ICALP, Noordwijkerhout, LNCS 85.* pp. 299–309. – Springer-Verlag.

[HM85] HENNESSY (M.) & MILNER (R.). – Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, vol. 32, n° 1, janvier 1985, pp. 137–161.

[Hoa85] HOARE (C. A. R.). – *Communicating Sequential Processes.* – Prentice Hall Int., 1985.

[Hoa94] HOARE (C.A.R.). – Mathematical models for computing science. – FTP, Aug 1994.

[Hoo94] HOOGERS (P.W.). – *Behavioural Aspects of Petri Nets.* – Thèse de PhD, Riskjt Univ., 1994.

[Iba78] IBARRA (O. H.). – Reversal-bounded multicounter machines and their decision problems. *Journal of the ACM*, vol. 25, 1978, pp. 116–133.

[Jan93] JANČAR (P.). – *Decidability Questions for Bisimilarity of Petri Nets and Some Related Problems.* – Research Report n° ECS-LFCS-93-261, Edinburgh, Lab. for Foundations of Computer Science, avril 1993.

[Jan94] JANČAR (P.). – Decidability questions for bisimilarity of Petri nets and some related problems. *In: Proc. STACS'94, Caen, France, LNCS 775.* pp. 581–592. – Springer-Verlag.

[Jen92] JENSEN (K.). – *Coloured Petri nets : Basic concepts, analysis methods and practical use.* – Springer Verlag, 1992, *EATCS monographs*, volume 1.

[Jen95] JENSEN (K.). – *Coloured Petri nets : Basic concepts, analysis methods and practical use.* – Springer Verlag, 1995, *EATCS monographs*, volume 2.

[JG88] JONES (G.) & GOLDSMITH (M.). – *Programming in* OCCAM-2. – Prentice Hall, 1988. Cité dans degano93a.

[JLL77] JONES (N. D.), LANDWEBER (L. H.) & LIEN (Y. E.). – Complexity of some problems in Petri nets. *Theoretical Computer Science*, vol. 4, 1977, pp. 277–299.

[Kos82] KOSARAJU (S. R.). – Decidability of reachability in vector addition systems. *In: Proc. 14th ACM Symp. Theory of Computing, San Francisco*, pp. 267–281.

[KS90] KANELLAKIS (P. C.) & SMOLKA (S. A.). – CCS expressions, finite state processes and three problems of equivalence. *Information and Computation*, vol. 86, 1990, pp. 43–68.

[Kul94] KULICK (S.). – *Process Algebra, CCS and Bisimulation Decidability.* – Rapport technique n° IRCS 94-06, U. of Pennsylvania, April 1994.

[Kup93] KUPPRION (R.). – Hierarchical construction of labeled Petri nets. – 02 1993. Rapport de stage, LIFIA/Parallélisme.

[Lak93] LAKOS (C.A.). – *A general Systematic Approch to Arc Extension for CPN.* – Rapport technique n° TR93-8, Dept. of Comp. Sci., U. of Tasmania, 1993.

[Lak94] LAKOS (C.A.). – *From Coloured Petri Nets to Objects Petri Nets.* – Rapport technique n° TR94-9, Dept. of Comp. Sci., U. of Tasmania, 1994. Mise à jour de TR94-3.

[LFB87] LEE-KWANG (H.), FAVREL (J.) & BAPTISTE (P.). – Generalized Petri net reduction method. *IEEE Trans. Systems, Man and Cybernetics*, vol. 17, n° 2, mars 1987, pp. 297–303.

[Lip76] LIPTON (R.). – *The Reachability Problem is exponential-space-hard.* – Dept. Comp. Sci. Rep. n° 62, New Haven, Yale Univ., 1976. Cité dans mayr84.

[LR78] LANDWEBER (L.H.) & ROBERTSON (E.L.). – Properties of conflict-free and persistent Petri nets. *J. of the ACM*, vol. 25, n° 3, 1978, pp. 352–364.

[LS] LEMKE (I.) & SANDER (G.). – *Visualization of Compiler Graphs.* – Uni. des Saarlandes.

[MAS92] MIRIYALA (S.), AGHA (G.) & SAMI (Y.). – Visualizing actor programs using predicate transition nets. *Journal of Visual Languages and Computation*, vol. 3, n° 2, juillet 1992, pp. 195–220.

[May81] MAYR (E.W.). – Persistence of vector replacement systems is decidable. *Acta Informatica*, vol. 15, 1981, pp. 309–318. – Cité dans mayr84.

[May84] MAYR (E. W.). – An algorithm for the general Petri Net reachability problem. *SIAM J. Comput.*, vol. 13, n° 3, août 1984, pp. 441–460.

[Mes90] MESEGUER (J.). – *Rewriting as a Unified Model of Concurrency.* – Tech. Report n° SRI-CSL-90-02R, Computer Science Lab., SRI International, juin 1990.

[Mey88] MEYER (B.). – *Obbject-oriented Software Construction.* – Prentice Hall, 1988.

[Mil80] MILNER (R.). – *A Calculus of Communicating Systems.* – Springer-Verlag, 1980, *Lecture Notes in Computer Science*, volume 92.

[Mil83] MILNER (R.). – Calculi for synchrony and asynchrony. *Theoretical Computer Science*, vol. 23, n° 3, 1983, pp. 267–310.

[Mil89] MILNER (R.). – A complete axiomatisation for observational congruence of finite-state behaviours. *Information and Computation*, vol. 81, n° 2, 1989, pp. 227–247.

[Min67] MINSKY (M.). – *Computation : Finite and Infinite Machines.* – Prentice Hall, 1967, *Automatic Computation.*

[Mur91] MURPHY (D.). – The physics of observation : A perspective for concurrency theorists. *Bull. of EATCS*, vol. 44, Jun 1991, pp. 192–200.

[NPW81] NIELSEN (M.), PLOTKIN (G. D.) & WINSKEL (G.). – Petri nets, event structures and domains, Part I. *Theoretical Computer Science*, vol. 13, n° 1, 1981, pp. 85–108.

[NT84] NIELSEN (M.) & THIAGARAJAN (P. S.). – Degrees of nondeterminism and concurrency: A Petri net view. *In: Proc. 4th Conf. on Foundations of Software Technology and Theor. Comp. Sci., Bangalore, India, LNCS 181.* pp. 89–117. – Springer-Verlag.

[Oak94] Oak Ridge Nat. Lab., Tennessee. – *PVM 3 User's guide and Reference Manual*, Sept 1994.

[Old89] OLDEROG (E.-R.). – Strong bisimilarity on nets: a new concept for comparing net semantics. *In: Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, Noordwijkerhout, LNCS 354.* pp. 549–573. – Springer-Verlag.

[Old91a] OLDEROG (E.-R.). – Correctness of concurrent processes. *Theoretical Computer Science*, vol. 80, 1991, pp. 263–288.

[Old91b] OLDEROG (E.-R.). – *Nets, Terms and Formulas.* – Cambridge Univ. Press, 1991, *Cambridge Tracts in Theoretical Computer Science*, volume 23.

[Opp] OPPEN (D.C.). – A "$2^{2^{2^{pn}}}$ upper bound on the complexity of Presburger arithmetic, optcrossref =.

[Par81] PARK (D.). – Concurrency and automata on infinite sequences. *In: Proc. 5th GI Conf. on Th. Comp. Sci., LNCS 104.* pp. 167–183. – Springer-Verlag.

[Pet62] PETRI (C. A.). – *Kommunikation mit Automaten.* – Thèse de PhD, Univ. Bonn, 1962. Schriften des Instituts für Instrumentelle Mathematik.

[Pet77] PETRI (C. A.). – *Non-sequential processes.* – Tech. Report n° GMD-ISF-77-5, St Augustin, Gesellschaft Math. Datenverarb., 1977.

[Pfi92] PFISTER (W.). – *Simplification sémantique des réseaux de Petri par la bisimulation de places.* – Rapport de DEA, Univ. Grenoble, juin 1992.

[Pfi94] PFISTER (W.). – Bisimulations de places et réductions des réseaux de Petri. *In: Proc. Renpar'94, Lyon*, pp. 17–23.

[Pra86] PRATT (V. R.). – Modeling concurrency with partial orders. *Int. J. Parallel Programming*, vol. 15, n° 1, 1986, pp. 33–71.

[PRS92] POMELLO (L.), ROZENBERG (G.) & SIMONE (C.). – A survey of equivalence notions for net based systems. *In: Advances in Petri Nets 1992, LNCS 609.* pp. 410–472. – Springer-Verlag.

[PS90] POMELLO (L.) & SIMONE (C.). – A state transformation preorder over a class of EN systems. *In: Advances in Petri Nets 1990, LNCS 483.* pp. 436–456. – Springer-Verlag.

[PX95] PROTH (J.-M.) & XIE (X.). – *Les réseaux de Petri pour la conception et la gestion des systèmes de production.* – Masson, 1995, *MIM.*

[Rei85] REISIG (W.). – *Petri Nets. An Introduction.* – Springer-Verlag, 1985, *EATCS Monographs on Theoretical Computer Science*, volume 4.

[Rei92] REISIG (W.). – *A Primer in Petri Net Design.* – Springer-Verlag, 1992.

[Reu89] REUTENAUER (C.). – *Aspects Mathématiques des réseaux de Petri.* – Paris, Masson, 1989.

[RT86] ROZENBERG (G.) & THIAGARAJAN (P. S.). – Petri nets: Basic notions, structure, behaviour. *In: Current Trends in Concurrency, LNCS 224.* pp. 585–668. – Springer-Verlag.

[SM83] SUZUKI (I.) & MURATA (T.). – A method for stepwise refinement and abstraction of Petri nets. *Journal of Comp. and Sys. Sci.*, vol. 27, 1983, pp. 51–76.

[TS93] TERUEL (E.) & SILVA (M.). – Liveness and home-states in Equal Conflict Systems. pp. 415–432.

[Vog90] VOGLER (W.). – *Failures Semantics of Petri Nets and the Refinement of Places and Transitions.* – Tech. Report n° TUM-I9003, Munich, Institut für Informatik, TUM, janvier 1990.

[Vog92] VOGLER (W.). – *Modular Construction and Partial Order Semantics of Petri Nets.* – Springer-Verlag, 1992, *Lecture Notes in Computer Science*, volume 625.

[Vog94] VOGLER (W.). – Generalized OM-bisimulations. *Information and Computation*, 1994. – To appear.

[VV81] VALK (R.) & VIDAL-NAQUET (G.). – Petri nets and regular languages. *Journal of Computer and System Sciences*, vol. 23, 1981, pp. 299–325.

[Wei89] WEIJLAND (W. P.). – *Synchrony and Asynchrony in Process Algebra.* – Thèse de PhD, Univ. Amsterdam, juin 1989.

[Win87] WINSKEL (G.). – Petri nets, algebras, morphisms and compositionality. *Information and Computation*, vol. 72, n° 3, mars 1987, pp. 197–238.

[Yam84] YAMASAKI (H.). – Normal petri nets. *Theoretical Computer Science*, vol. 31, 1984, pp. 307–315.

[Zub80] ZUBEREK (W.M.). – Timed Petri nets and preliminary performance evaluation. *In: 7th Annual Symposium on Computer Architecture.* – La Baule, France, May 1980.

Index

Dans l'index, les pages sont en *italique* lorsqu'il s'agit d'une définition, d'un théorème ou d'un algorithme.

Les symboles ne sont indexés que par rapport à leur définition.

Symboles

V

Résumé

Cette thèse concerne une nouvelle méthode de réduction des réseaux de Petri. Le problème de la réduction des réseaux a d'abord été considéré comme le moyen de réduire l'espace des états sans modifier un certain nombre de propriétés (telles que «être borné», «être vivant», etc) afin de faciliter leur étude. Nous nous plaçons ici dans une toute autre optique, puisque nous allons chercher une méthode de réduction non pas en voulant conserver un ensemble de propriétés, mais préservant le comportement du réseau.

Dans les premières parties, nous recherchons la relation la plus grande possible qui soit une équivalence sur les places du réseaux nous permettant de les fusionner, telle que le réseau réduit soit bisimilaire au réseau original. Nous montrons que la *bisimulation de places* vérifie ces contraintes et qu'il existe des algorithmes efficaces (polynomiaux) pour la calculer.

Dans la suite, des cas d'extensions classiques des réseaux (arcs inhibiteurs) comme des équivalences (bisimulations «observationnelles») sont abordés. La dernière partie de la thèse s'intéresse à l'équivalence de propriétés entre le réseau initial et le réseau quotient. Ce travail se termine par la présentation succincte du logiciel `PetriS` qui met en œuvre les différents algorithmes.

Mots clés : théorie du parallélisme, réseaux de Petri, réduction, bisimulation

Abstract

This thesis concerns a new reduction method for Petri nets, *place bisimulation*. Formerly, reduction was considered as a technic to shrink the state-space without changing some behavioural properties (such as liveness, boudedness, etc) or as algorithms to state wether or not a net has such properties.

We took a different approch since we granted to preserve the whole behavior and to make a structural reduction (as place merging). The first parts of this work presents solution : *place bisimulation*. For this method, we derive polynomial algorithms.

Next, we extend our results to other net types (with inhibitor arcs) or bisimulation types (τ-bisimulations). We conclude our theoretical approch by linking our reduction method to some other theories (such as conflict-free nets).

Finally, we conclude with some remarks and examples from the `PetriS` system.

Keywords : concurrency theory, Petri nets, reduction method, bisimulation