UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE

Année 2003

N° d'ordre : 3 2 9 9

THESE

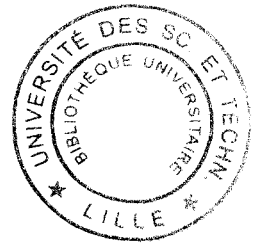
pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE LILLE 1
Discipline : PRODUCTIQUE AUTOMATIQUE et INFORMATIQUE INDUSTRIELLE

présentée

par

Benoît TROUILLET



SUR L'ÉVALUATION DU COMPORTEMENT LOGIQUE DES SYSTÈMES DE PRODUCTION MANUFACTURIÉRE PAR MÉTHODES EXACTES

Directeur de thèse :

M. Jean-Claude GENTINA Professeur à l'École Centrale de Lille

JURY :

M^me^ Mireille BAYART	Examinateur	Professeur à l'Université de Lille 1
M. Ahmer BENASSER	Examinateur	Maître de Conférence à l'I.U.T. de Béthune
M. Philippe CHRÉTIENNE	Examinateur	Professeur à l'Université Pierre et Marie Curie
M. Jean-Louis FERRIER	Rapporteur	Professeur à l'Université d'Angers
M. Robert VALETTE	Rapporteur	Directeur de Recherche au LAAS-CNRS
M. Pascal YIM	Examinateur	Professeur à l'École Centrale de Lille

Thèse préparée au Laboratoire d'Automatique et d'Informatique industrielle de Lille
L.A.I.L., UMR 8021 - École Centrale de Lille

Remerciements

Cette thèse a été préparée au LAIL au sein de l'équipe PFM. Dans cette équipe, les premiers remerciements vont à mon directeur de thèse : Jean-Claude GENTINA, professeur et directeur de l'école Centrale de Lille. Il s'est chargé de mon encadrement, avec le concours d'Ahmer BENASSER, maître de conférence à l'I.U.T de Béthune. Je les remercie sincèrement pour leur disponibilité, leur soutien et leur patience à mon égard.

Je remercie :
Monsieur Jean-Louis FERRIER, Professeur à l'université d'Angers,
Monsieur Robert VALETTE, Directeur de recherche au LAAS-CNRS.
Pour l'honneur qu'ils me font en acceptant d'examiner ce travail et d'être les rapporteurs de cette thèse. Leurs critiques ont permis d'améliorer l'écriture de ce mémoire.

Je suis également très reconnaissant à :
Monsieur Philippe CHRETIENNE, professeur de l'université Pierre et Marie CURIE,
Monsieur Pascal YIM, professeur de l'école Centrale de Lille.
Pour l'honneur qu'il me fait en examinant ce travail et en acceptant de participer à mon jury de thèse.

Je remercie sincèrement les membres de l'équipe PFM pour le temps qu'ils m'ont consacré. La bonne humeur jointe aux précieux conseils et critiques ont contribués au développement de mes recherches et de mes compétences en enseignement dans un climat studieux et convivial (Nathalie, Ouajdi, Pascal ...). Je suis également reconnaissant envers ceux qui m'ont fait confiance et m'ont initié aux joies de l'informatique (Christian, Didier, Samir ...).

Je tiens à exprimer toute ma sympathie à l'ensemble du personnel de l'école centrale en général, et à l'ensemble du 3éme étage du bâtiment C en particulier. Cette « grande famille » m'a accueilli et intégré dans la bonne humeur qui la caractérise. J'adresse une pensée particulière aux membres de l'association des doctorants.

Enfin, je remercie Monsieur M. VANGREVENINGE pour la reprographie de ce mémoire ainsi que Mme E. VÉRIN pour sa gentillesse et sa disponibilité.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

Depuis quelques décennies le monde de l'industrie est soumis à une évolution sensible du contexte économique due à une forte concurrence mondiale et aux variations de la consommation.

Ces pressions entraînent une évolution de la physionomie des entreprises dont le souci est de continuer de réaliser des projets au sein de ce milieu économique.

D'une part, la situation de crise cyclique induit une réduction des coûts. D'autre part les variations fréquentes de la demande des consommateurs exigent une adaptation rapide de la production. Dans ce sens, les entreprises ont évolué en adoptant de nouvelles stratégies de production avec des budgets limités. Ces différentes stratégies de production se traduisent par différents profils d'ateliers de production. C'est par exemple le cas avec l'apparition du « flux tendu » (ou « zéro stock ») méthode pour laquelle le stock est réduit au minimum requis.

Concrètement, ces contraintes induisent des mutations internes. Les ateliers de production, selon une vision globale, sont constitués d'une composante humaine et d'une composante matérielle. Les effets des restrictions budgétaires et de la demande d'une productivité accrue ont induit dans un premier temps la diminution de la composante humaine au profit de la composante « matérielle », notamment pour les tâches simples et répétitives. Dans un second temps, l'évolution récente des techniques et de la consommation a engendré des machines moins spécialisées, permettant d'introduire la flexibilité au sein des ateliers de production donnant ainsi naissance aux : « ateliers flexibles de production ». Cette flexibilité opératoire, à laquelle l'homme est à nouveau associé, est un vecteur de complexité à tous les niveaux et en particulier lors de l'élaboration de la planification des tâches, phase qui

demande une bonne connaissance de l'atelier flexible de production.

La complexité inhérente à la planification des tâches est source de partenariat entre les entreprises et les centres de recherche afin de résoudre des problèmes de plus en plus difficiles, comme c'est le cas pour le problème de l'ordonnancement connu pour être de très grande complexité [CC88a, SU89a].

1.1 Contexte de production manufacturière

Ce type de système est l'un des centres d'intérêt majeur de l'équipe **S**ystème à **É**vénements **D**iscret (**SED**) du **L**aboratoire d'**A**utomatique et d'**I**nformatique de **L**ille (**LAIL**). Nous considérons dans cette thèse la classe des **S**ystèmes **F**lexible de **P**roduction **M**anufacturière (**SPFM**). Ce type d'atelier respecte des hypothèses de fonctionnement que nous décrirons plus précisément par la suite.

Les SPFM transforment des matières premières pour obtenir un produit manufacturé [Bon87]. Une étude préliminaire permet de mettre en oeuvre un processus de fabrication, c'est-à-dire une suite d'opérations successives que doivent subir les matières premières pour aboutir au produit final. Cette phase préliminaire correspond à la planification des tâches. Le résultat de cette étude permet d'appréhender les contraintes matérielles et les contraintes structurelles liées à la production. Celles-ci font apparaître deux thèmes de réflexion. Le premier concerne le matériel nécessaire pour la phase de production avec notamment l'utilisation de machines multi-tâches et des moyens de transport caractérisant la flexibilité de l'atelier étudié. Puis, la planification des usinages met en évidence les ordres partiels des usinages à réaliser. Ainsi de nombreuses décisions devront être prises pour l'élaboration d'une commande de notre système.

La flexibilité, point central de notre étude, se situe à plusieurs niveaux : que ce soit dans la variation de la demande ou dans le fait que les pièces à produire ne subissent plus la même planification, ce qui conduit vers une étude fondamentale de l'atelier afin de faciliter l'adaptation de l'atelier de production aux différentes variations possibles. Au niveau des machines multi-tâches, le séquencement de celles-ci n'est pas unique. Enfin, lors de l'utilisation de moyens de transport qu'il convient d'organiser et de piloter selon une stratégie réfléchie.

Cette brève description permet d'introduire d'emblée la complexité inhérente à ce type de problème. En effet, la combinatoire des choix est très importante au sein de l'atelier

dont l'objectif est de produire différents types de produits manufacturés. Mais ceci peut être répété un grand nombre de fois lors d'une production en moyenne ou grande série.

1.2 Le système de production flexible et manufacturier

Après une brève description du contexte et des raisons qui justifient l'intérêt d'un tel modèle, nous définissons la notion de SFPM telle qu'elle sera utilisée par la suite :

Définition 1 *Un SFPM est caractérisé par :*

- *les différentes pièces produites selon des gammes bien définies ;*
- *les ressources machines qui peuvent être partagées entre plusieurs tâches.*

On distingue les ressources opératoires qui modifient l'état de la pièce (notamment les assemblages) et les ressources de transport qui n'affectent que la position des pièces.

Ainsi, les SFPM disposent-ils de ressources finies telles que des stocks intermédiaires, outils, palettes ... L'élaboration du pilotage sur ce type de systèmes joue un rôle essentiel. C'est le cas notamment du séquencement des ressources partagées qui assure un comportement logique correct pour approcher la « meilleure performance ». Dans ce sens, une bonne représentation du comportement dynamique de ces systèmes est requise pour optimiser un fonctionnement parallèle d'entités indépendantes qu'il convient de synchroniser de façon rigoureuse.

Nous sommes donc amenés à choisir un modèle qui intègre toutes les caractéristiques et tous les paramètres de pilotage du SFPM :

- les flexibilités des gammes opératoires : parallélisme, séquencement des opérations ;
- les flexibilités au niveau des ressources : paramétrage des machines et affectation ;
- les flexibilités du transport: palettes, chariots filoguidés ...

Ceci nécessite une modélisation claire et susceptible de prendre en compte globalement ces différents points de vue.

Les réseaux de Petri (RdP), utilisés depuis plus de vingt ans par l'équipe SED du LAIL, sont très certainement l'un des principaux modèles de références pour spécifier, analyser et optimiser les performances des SFPM. C'est dans ce sens que le LAIL a contribué au paradigme RdP. Leur formalisme est présenté en annexe.

1.3 Cadre de l'étude

1.3.1 Introduction

Les hypothèses de production sont essentielles lors de la phase d'étude préliminaires d'un système de production. Elles possèdent deux intérêts majeurs :

- elle participe à la conception des SFPM ;
- elle permet de rendre les résultats accessibles et effectivement utilisables.

Ce fut l'objet d'un projet développé à partir de 1981 au sein du laboratoire d'automatique et d'informatique de Lille, nommé **C**onception **A**ssistée de **S**ystèmes de **P**roduction **A**utomatisés dans l'**I**ndustrie **M**anufacturière (CASPAIM). L'outil de modélisation est basé sur les réseaux de Petri [ST96a, ST96b]. Le but de ce projet est donc de concevoir un SFPM et d'en analyser les propriétés (vivacité, finitude ...) avec un objectif final d'évaluation des performances. Ainsi, l'idée directrice de ce projet est l'élaboration du pilotage d'un SFPM. L'élaboration de la commande est réalisée par l'ordonnancement des machines et se réduit à la transformation du modèle RdP initial en un graphe d'événements. Nous pouvons citer [GBK88, Kas88, Boi91, Ham91, Cru91, Tog92, Bou93, EK93, Ama94, Aus94, Cra94, Huv94, Ohl95, Taw95, Cas96, Ker96, Cam97].

1.3.2 Hypothèses de production

Lors de la modélisation, plusieurs étapes permettent d'élaborer le réseau de Petri final. La première phase consiste en l'identification des différents systèmes opératoires selon trois points de vue complémentaires :

- le point de vue *pièce* dont le processus de fabrication est représenté par des gammes flexibles modélisées par RdP ;
- le point de vue *fonctionnel* et opérationnel qui utilise également les RdP dans la caractérisation du comportement dynamique des ressources opératoires et du réseau de transport ;
- le point de vue *pilotage* dont le but est d'élaborer une commande permettant une production spécifiée.

Commençons par rappeler les principales caractéristiques de la modélisation des gammes opératoires. Nous supposons que l'on dispose de la planification de production de chacune des pièces, nous partons ainsi des gammes flexibles.

Cette première phase constitue un problème de pilotage. Les gammes opératoires sont flexibles, c'est-à-dire que plusieurs opérations peuvent être permutées faisant apparaître un ordre partiel au sein de la gamme de production. Donc le but de cette phase est d'élaborer une commande globale dans les gammes du système du SPFM. Cette commande sera fonction de la nature ou des types de pièces à produire et du nombre d'exemplaires à produire selon le carnet de commande. L'effet de l'élaboration d'une commande sera la minimisation du temps de cycle mais aussi la minimisation des coûts de production (des en-cours utilisés) et aussi une diminution significative de la complexité de la recherche d'un ordonnancement.

Cette problématique, qui prend en compte les principaux objectifs d'une production industrielle, a fait l'objet de recherches au sein de l'équipe SED du LAIL. Elle fut abordée par Harald Ohl [Ohl95] poursuivie par Hervé Camus [Cam97] avec un objectif d'informatisation de la méthode et Ouajdi Korbaa [Kor98] dans le but d'étendre et compléter les résultats en vue de l'implantation dans le projet CASPAIM. Ces travaux ont permis de linéariser des gammes à partir d'optimisations fines qui conservent dans l'optimisation réalisée tous les paramètres de décisions stratégiques devant conduire à l'optimisation dans une seconde phase des ressources de production et de transport. Le principe de ces travaux repose sur une méthodologie structurée en vue de l'évaluation de performances et de l'ordonnancement des SFPM. La résolution des flexibilités est réalisée à l'aide d'une heuristique, celle-ci s'applique de manière progressive par la mise en œuvre de politiques de placements dont le but est de résoudre les différents choix à effectuer tout au long de la modélisation, en commençant au niveau des prises de décisions sur les ordres partiels des gammes, puis sur le routage des ressources et enfin lors de la prise en compte des moyens de transport. Cette démarche aboutit à l'obtention d'un graphe d'événements non pondérés et marqué dont la caractéristique majeure est de représenter une commande globale des ressources du SFPM sous jacent.

Nous supposons dans cette thèse que pour les SFPM considérés, nous connaissons le processus de fabrication des différents produits ainsi que la quantité de fabrication et les machines disponibles. Nous supposons que le processus de fabrication est complètement déterminé, c'est-à-dire que nous sommes en présence de gammes linéaires ; si tel n'est pas le cas nous ferons références aux travaux précédents pour nous ramener à cette hypothèse. De

plus, **un objectif supplémentaire est de prendre en compte le processus d'assemblage et de désassemblage complexe** (courant dans ce type de modèle), objectif non retenu dans les travaux antérieurs de H. Ohl, H. Camus et O. Korbaa.

À partir de l'hypothèse faite précédemment, nous pouvons maintenant modéliser un SFPM à l'aide des réseaux de Petri. Notre objectif est de réaliser **une étude approfondie de son comportement logique** en « optimisant » les performances associées, afin que le SFPM sous jacent respecte un fonctionnement de référence optimal. L'approche utilisée se démarque des études précédentes où l'outil utilisé était basé sur une heuristique, nous étudierons de manière exacte le comportement de notre modèle en utilisant des structures algébriques appropriées ; l'approche proposée est donc par essence formalisée et permet, nous en développerons la thèse, de réduire la complexité de l'optimisation d'un séquencement dans l'hypothèse d'un comportement cyclique du SFPM.

1.4 Différentes approches pour l'étude des SFPM

1.4.1 Introduction

Une étude des SFPM est motivée par des objectifs divers :

- minimisation d'en-cours ;
- optimisation de temps de cycle ;
- ordonnancement cyclique ...

Le but est l'obtention d'une commande spécifique pour l'atelier de production afin de répondre au mieux à l'optimisation d'un ou plusieurs critères. La complexité inhérente à cette classe de problèmes est évoquée [BEN82, SU89b, CC88a]. En effet, le problème de l'ordonnancement est plusieurs fois NP-Difficile [SU89b], les causes en sont multiples :

- ressources partagées ;
- l'utilisation du parallélisme : son utilisation tend cependant à augmenter les performances de l'atelier de production ;
- le traitement par lots (groupage et dégroupage des produits) ;
- le travail en flux tendu : minimisation des en-cours (WIP).

Ainsi, l'utilisation du parallélisme et le traitement par lots des en-cours contribue à l'amélioration des performances de l'atelier de production, par traitement simultané de

plusieurs pièces, il est cependant source de complexité dans la recherche de performances optimisées.

De plus, les en-cours sont assimilables à des stocks dynamiques. Ils correspondent le plus souvent au nombre de pièces produites au cours d'un cycle de production. La minimisation d'en-cours contribue de façon évidente à la minimisation des coûts. Ce problème est également complexe. En effet, un nombre insuffisant d'en-cours ne permet pas d'atteindre une performance optimale tandis qu'un nombre trop important d'en cours peut induire un blocage du système par saturation. Ceci est illustré sur la figure 1.1 dans laquelle nous pouvons distinguer trois classes de comportement d'un SFPM :

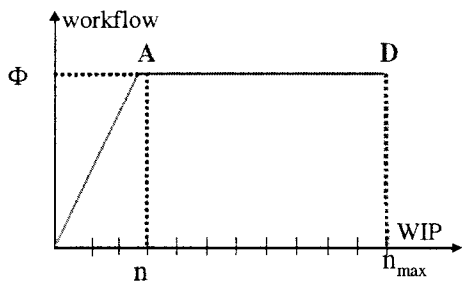


FIG. 1.1 – *relation entre en-cours et flux de production*

Le premier comportement caractérise une augmentation du flux de production en relation avec les en-cours, sur la figure 1.1 ceci est représenté par une progression linéaire. Puis le flux maximal de production est atteint, les en-cours ajoutés n'ont aucune influence sur le flux de production, les machines fonctionnent à leur régime maximum. Enfin, lorsque le système a atteint sa capacité maximale, l'ajout d'un en-cours se traduit par une situation de blocage et le flux de production cesse immédiatement.

Aussi, le but de la minimisation des en-cours est d'en trouver le nombre minimum correspondant à un flux optimal, représenté par le point A de la figure 1.1.

1.4.2 Intérêt des réseaux de Petri

Le modèle réseau de Petri développé par K. A. Petri [Pet62] est défini dans l'annexe A. Ce formalisme permet d'abstraire un système concret en un modèle graphique. Dans le contexte des SFPM, les réseaux de Petri sont reconnus comme un excellent outil de modélisation [DHP+93] et permettent de mener une étude fondamentale du comportement. Aussi

convient-il de répondre aux problèmes définis précédemment pour la classe des SFPM qui appartiennent à la classe plus générale des systèmes dynamiques à événements discrets en utilisant le modèle générique des réseaux de Petri.

Plus précisément, le caractère discret est pris en compte lors des tirs des transitions, un franchissement constitue un événement dans l'évolution d'un réseau de Petri. Puis le caractère dynamique est traduit par l'évolution du marquage. La puissance de modélisation de ce formalisme réside dans la caractérisation de l'ensemble des comportements d'un système complexe. Les synchronisations, les partages de ressources, les choix peuvent être modélisés de manière simple et visuelle. Aussi, l'utilisation de ce formalisme ne se limite-t-il pas aux SFPM mais possède de nombreuses applications aux réseaux de communications, systèmes de transport, systèmes d'exploitation informatiques, systèmes logistiques ...

Pour l'abstraction de modèles plus complexes, des extensions sont disponibles. La prise en compte du temps est possible sous différentes hypothèses, de même que l'utilisation de ratios de production, des lois stochastiques associées aux durées opératoires ...

1.5 Modélisation par RdP des SFPM

Nous nous intéressons à la modélisation d'un atelier de production flexible [PX95] pour lequel les gammes sont, ou ont été, linéarisées. De plus, un SFPM est un système dynamique discret par nature et son état change suivant l'occurrence d'événements discrets. Son comportement peut intégrer des évolutions parallèles, avec synchronisations et partages de ressources.

Aussi, le choix des réseaux de Petri est justifié en raison de son pouvoir d'expression. Il permet de représenter simplement et de manière graphique un tel système. Pour une meilleure compréhension, la modélisation sera couplée à une description concrète de chaque entité du réseau de Petri. Pour cela, nous allons proposer un exemple de référence, fil rouge, illustrant notre propos tout au long de cette thèse.

Les gammes opératoires

Les SFPM sont d'abord abordés selon deux points de vue complémentaires. D'une part, le processus de transformation du produit : il nécessite la connaissance d'un ensemble d'opérations successives (gamme); l'hypothèse faite sur la linéarité des gammes impose un ordre total sur les tâches. D'autre part, le processus opératoire de fabrication, qui com-

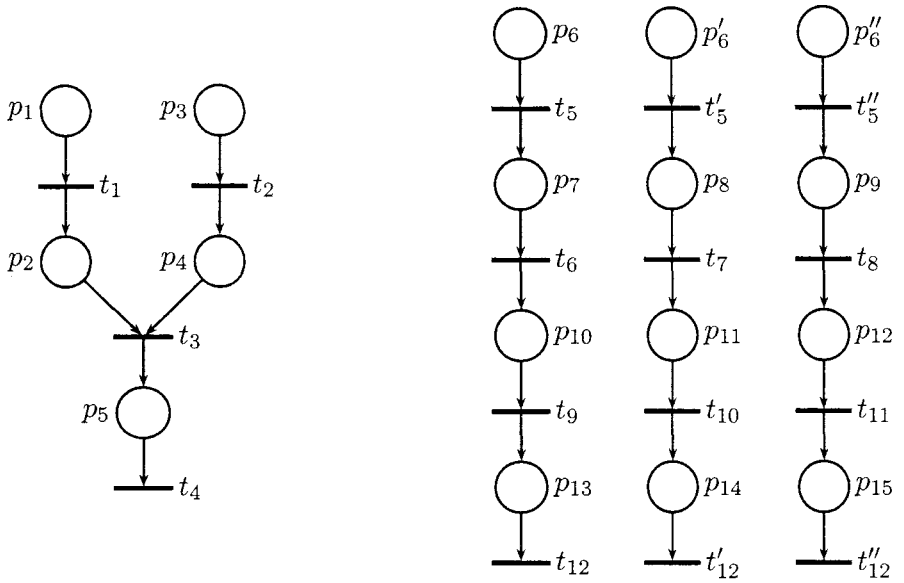


FIG. 1.2 – *Les gammes opératoires*

prend les aspects transformation atomique et d'assemblage/désassemblage pour l'obtention du produit final.

Les produits Le but de ce paragraphe est de représenter le processus de fabrication de chaque produit manufacturé désiré. Les tâches à effectuer sont supposées partiellement ordonnées, la modélisation consiste à abstraire chaque opération à l'aide du formalisme Réseau de Petri.

Nous proposons donc d'introduire ici l'exemple d'illustration de référence utilisé dans cette thèse. L'atelier flexible évoqué a pour objectif de fabriquer un produit de type A pour un lot de produits B. De plus, le produit A est le résultat de l'assemblage de deux pièces de type A_1 et A_2 . Le lot B est constitué de trois pièces B_1, B_2 et B_3. Chaque pièce subit différents usinages,

Les pièces de type A_1 et A_2 subissent trois opérations : une transformation, un assemblage et une transformation finale.

Les pièces de types B_1, B_2 et B_3 en subissent quatre, les usinages sont représentés par les transitions. Les places du RdP modélisent les stocks intermédiaires placés entre chaque lieu d'usinage. La figure 1.2 représente le graphe obtenu.

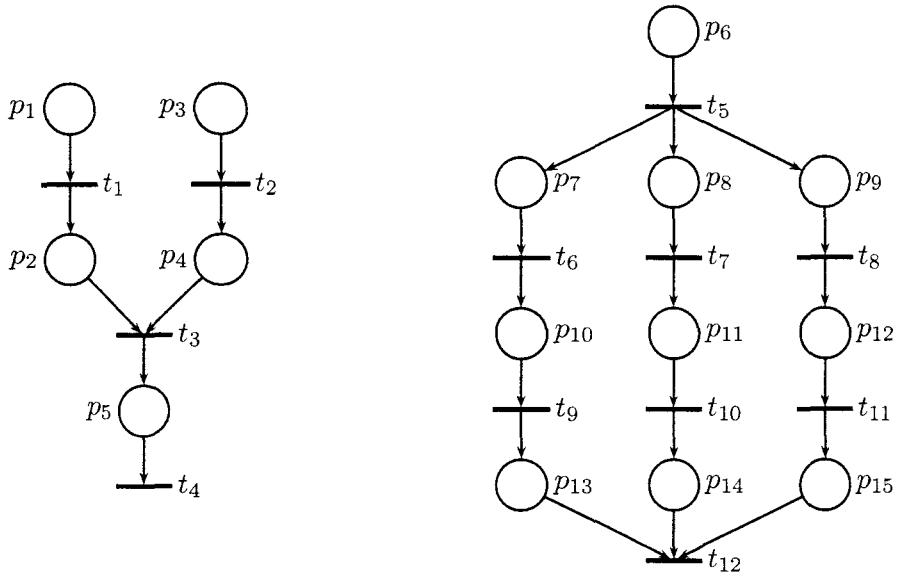


FIG. 1.3 – *Les gammes avec assemblage*

Assemblage et désassemblage Les lignes d'assemblage et désassemblage permettent le traitement par lot ou encore la production de produits composites. L'intérêt est ici de pouvoir favoriser le parallélisme lors de la production afin d'optimiser le temps de cycle.

Pour notre exemple, une pièce de type A est le résultat de l'assemblage des produits A_1 et A_2, la transition t_3 symbolise l'assemblage de ces deux produits. De même, le lot B résulte du traitement d'un lot. Le lot B est désassemblé en trois pièces B_1, B_2 et B_3 pour subir séparément des usinages (transitions t_5, t'_5 et t''_5), puis réassembler pour former le lot final B (transition t_{12}, t'_{12} et t''_{12}).

Le réseau de Petri représenté par la figure 1.3 décrit le séquencement des opérations pour chaque produit.

Les machines Ce paragraphe est destiné à la modélisation des partages de ressources au sein de notre atelier flexible. Il existe différents types de machines adaptées à autant de fonctionnalités requises pour une production. Nous allons en présenter différents types génériques sans être exhaustif de tous les types existants.

Les machines dédiées Dans les systèmes de production, les usinages ou transformations requièrent des machines dédiées ou spécifiques. Dans cette configuration, une machine dédiée est utilisée pour la tâche ou les tâches successives. Le modèle d'étude contient

une machine r_6 dédiée pour la tâche t_9.

Les machines multi-tâches Ce type de machine permet la fabrication ou l'usinage de produits différents. Elles constituent la caractéristique flexible des SFPM. Dans ce sens, l'atelier flexible considéré contient plusieurs machines multi-tâches, les machines multi-tâches : r_1, r_4 et r_5.

Les machines parallèles Si une tâche répétitive requiert une machine spécifique, la possibilité d'utiliser plusieurs machines identiques permet d'augmenter le parallélisme et éventuellement de diminuer le temps de cycle. Nous considérons deux machines r_1, fonctionnant en parallèles pour réaliser la tâche t_1.

Les machines d'assemblages et désassemblage Ce type de machine permet de regrouper en lot pour un traitement commun, ou au contraire de dissocier un lot pour effectuer des traitements en parallèle de chacune des pièces. Deux machines possèdent cette propriété au sein de notre atelier de production ; elles sont symbolisées par les ressources r_2 et r_3.

Fonctionnement cyclique Les ateliers de production considérés fabriquent des produits manufacturés en moyenne ou grande série. L'objectif est d'optimiser les performances de ce type de production sur plusieurs points essentiels : temps de cycle, respect des ratios de production, minimisation d'en-cours ...

Pour une production en moyenne ou grande série l'ordonnancement des tâches s'avère d'une très grande complexité (plusieurs fois NP-difficile). D'où l'idée de réduire cette complexité par l'étude et l'optimisation d'un cycle de production.

Toutefois, notons que l'ordonnancement optimal d'un cycle ne garantit pas l'optimalité de la production totale. Par contre, l'ordonnancement d'un nombre limité de tâches durant un cycle permet de faire chuter considérablement la complexité du problème.

Ainsi le caractère cyclique de notre SFPM est caractérisé dans le modèle RdP par les arcs joignant les transitions de sorties t_4 et t_{12} aux places d'entrée p_1, p_3 et p_6.

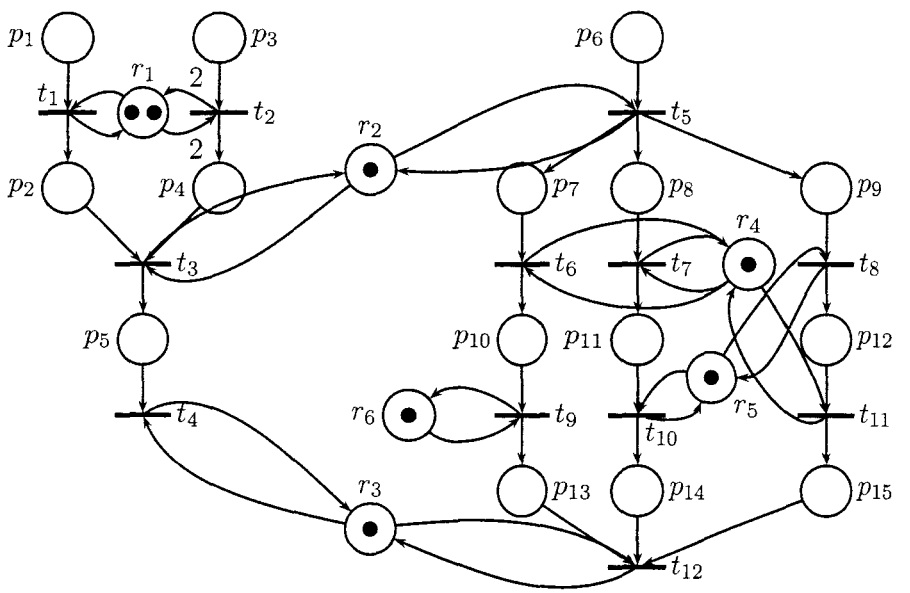


FIG. 1.4 – *Intégration des machines*

1.6 Méthode d'étude des réseaux de Petri

Lors d'une étude fondamentale, les ateliers de production sont généralement représentés par un modèle abstrait (Réseau de Petri, automate ...) permettant l'analyse de critères prédéfinis :

- l'ordonnancement [CC82] ;
- l'évaluation des performances ;
- le comportement de l'atelier.

Pour l'analyse des modèles plusieurs types de méthodes sont à notre disposition.

1.6.1 La simulation

Lors des premières études fondamentales des ateliers de production, les modèles d'atelier obtenus nécessitaient une validation du comportement et des performances associés. L'idée fut de les animer, ce qui constitue les premiers fondements de la simulation. Puis des simulateurs sont apparus avec les premiers ordinateurs.

La simulation [Cer88] permet une étude de procédés et de systèmes fortement complexes, c'est-à-dire pour lesquels la complexité inhérente rend difficile une approche analytique du problème, ce qui justifie l'utilisation de la simulation pour les systèmes de grande taille, puisque la complexité d'un problème est souvent proportionnelle à sa taille, dû à la forte interaction entre les sous-systèmes qui le composent.

Cette technique consiste en une modélisation du système physique, par exemple à l'aide d'une modélisation par réseaux de Petri. Le modèle muni de ses conditions initiales, (chargement initial du modèle) est alors simulé (calculé), les résultats peuvent concerner la validation du comportement, l'évaluation de performance ... Suivant le modèle considéré un simulateur sera dédié à différentes classes problèmes souvent spécifiques. À titre d'exemple, nous pouvons citer les simulateurs suivants :

- GreatSPN pour les réseaux de Petri ;
- QPN pour les réseaux de files d'attentes ;
- SPNP pour les réseaux de Petri stochastiques ;
- Sirphyco pour les réseaux de Petri hybrides ...

La simulation par RdP Pour le cas des réseaux de Petri, la simulation prend comme données le modèle RdP (représentatif de la structure du réseau) et son marquage initial (représentatif des pièces présentes dans le système).

Dans ce type de modèle, chaque changement d'état se produit de manière discrète dans le temps, à des instants déterminés. L'ensemble des événements constitue un échéancier ordonné par une relation d'ordre partiel qui représente l'occurrence chronologique des événements. Aussi, l'évolution du système se fait elle par incrémentation du temps discret.

Si nous considérons l'utilisation de la simulation pour la classe des SFPM, nous sommes amenés à vérifier les performances vis à vis de nombreux scénarii de production générés par la flexibilité de l'atelier de production, ce qui motive l'élaboration d'un logiciel dédié comme GreatSPN. Lors de l'élaboration de différents scénarii de production, la simulation permet de tester et de vérifier le comportement et l'évaluation des performance des solutions obtenues.

Un domaine d'application est la prise de décisions en temps réel. La simulation constitue un outil puissant et surtout rapide pour évaluer les conséquences d'un événement imprévu, tel qu'une panne ou un changement de production, sur le comportement et les performances du système de production.

Enfin, la simulation s'avère un excellent outil de validation et d'évaluation de performances de scénarii prédéfinis.

Cependant, l'évaluation d'un scénario aux performances optimales à l'aide de la simulation implique la prise en compte de tous les scénarii. Or la flexibilité et la taille du système de production induit un très grand nombre de scénarii possibles, ce qui rend cette méthode difficilement utilisable pour une telle problématique.

1.6.2 Les méthodes analytiques

L'obtention d'un comportement spécifié demande une analyse et une modélisation de l'atelier de production. Le but est ici l'ordonnancement des ressources, mais aussi la minimisation des en-cours et du temps de cycle en respectant un comportement spécifié en terme de production, par exemple 10 pièces de type A pour 20 pièces de type B.

Précisons à ce niveau de notre exposé les principaux objectifs de cette étude :

Dans le contexte d'une production flexible manufacturière nous considérons le cas d'une production de moyenne à grande série de différents produits lancés simultanément sur

l'atelier. L'ordonnancement acyclique des ressources de production est trop complexe pour pouvoir être évalué. À fortiori la prise en compte d'un ou plusieurs critères d'optimisation est irréalisable. Dans ce sens, la mise en œuvre d'un ordonnancement cyclique permet selon les travaux de H. Ohl et al. [OCCG95] de caractériser un ordonnancement réalisable calculable et relativement optimisé.

Les techniques mises en œuvre jusqu'alors ont toutes consisté à linéariser plus ou moins la production pour aboutir à une technique d'ordonnancement basée sur l'utilisation d'heuristiques.

Nous allons en rappeler les principales caractéristiques pour mettre en évidence les limites de ces approches justifiant ainsi une autre technique d'optimisation plus formelle que nous présenterons au chapitre suivant.

Les heuristiques Une heuristique est une méthodologie structurée permettant de résoudre progressivement les problèmes rencontrés et dont l'objectif est l'obtention d'une commande de l'atelier.

Dans le cadre de l'évaluation des performances et de l'optimisation d'un atelier de production flexible, la commande correspond à un graphe d'événements obtenu après la résolution des flexibilités du système. Le but est l'optimisation de la production et d'obtenir une commande cyclique qui permette d'approcher les performances optimales du système. Le principe des heuristiques consiste à scinder le problème global en phases élémentaires pour résoudre par étapes successives les flexibilités rencontrées. De nombreuses heuristiques existent, celles-ci sont spécialisées selon les classes de problèmes abordés.

La production de type manufacturière constitue un champ d'application pour les heuristiques. En effet, les SFPM sont des systèmes complexe qui intègrent des degrés de liberté qu'il convient de résoudre pour approcher le procédé de fabrication optimal. Il existe différents types de flexibilités à résoudre, nous pouvons citer à titre d'exemple :

- la production en parallèle ;
- les ratios de production ;
- les flexibilités de gammes ;
- l'affectation des ressources ;
- l'ordonnancement ;
- l'en-cours ;

- les transitoires.

Il existe de nombreux travaux sur l'ordonnancement des systèmes de type SFPM dont l'objectif est d'atteindre un temps de cycle optimal avec le souci de la minimisation des en-cours. Nous proposons d'évoquer quelques-unes des principales heuristiques développées pour résoudre de ce type de problème relatif à l'évaluation de l'ordonnancement cyclique d'un SFPM.

L'heuristique de J. Ershler Parmi les critères d'optimisation, la minimisation des stocks dynamiques permet de réduire le coût économique de production. Aussi, J. Ershler [ER82] propose une heuristique pour un problème avec une machine critique. L'idée consiste en l'ordonnancement des tâches sur la machine critique, puis, de rechercher les ordonnancements des machines en amont et en aval de cette machine.

L'heuristique de H. P. Hillion Le but de cette heuristique est la minimisation du temps de cycle et des encours. H. P. Hillion et al. [HP89] étudie un problème pour lequel chaque processus de fabrication est représenté par une seule gamme opératoire et tel qu'une palette est dédiée à une seule gamme opératoire.

Le principe consiste à placer progressivement, éventuellement en parallèle, les opérations suivant l'ordre logique d'exécution et ceci pour toutes les opérations de la séquence. Aussi une fonction de placement choisit une tâche dans l'ensemble des opérations et la place définitivement, puis le processus est répété jusqu'au placement de toutes les opérations.

Soit r_j le nombre de cycle restant (CT) pour la $j^{\text{ème}}$ exécution d'une gamme opératoire(si ce nombre est supérieur au nombre initialement prévu alors il est égal à 0).

$At_j = (1+r_j)\times CT - \{$date de fin de la dernière opération de la $j^{\text{ème}}$ opération ordonnancée$\}$

At_j est le temps disponible (**A**vailable **t**ime) pour effectuer la $j^{\text{ème}}$ exécution de la gamme opératoire.

Soit $Ro_j = \sum_{k \in \text{opérations restantes}} D(t_{jk})$, la somme des durées des opérations restantes dans la $j^{\text{ème}}$ exécution de la gamme opératoire.

Le critère est une fonction de coût :

$$d_j = 1 - \left(\frac{Ro_j}{At_j}\right) \tag{1.1}$$

Cette fonction traduit le degré de liberté pour le placement des opérations de la $j^{ème}$ exécution d'une gamme opératoire. En fait, le plus grand degré correspond à une petite durée opératoire restante comparée au temps de disponibilité.

Le résultat de cette heuristique peut aboutir à un ordonnancement « non réalisable », dû au fait qu'une opération ne peut plus être placée quelque soit le nombre d'en-cours considéré.

La technique développée par H. P. Hillion consiste à relaxer la contrainte du temps de cycle par une augmentation temporaire de celui-ci pour reprendre la résolution. Cette méthode est utilisée jusqu'à obtention de l'ordonnancement des ressources.

Par contre, l'ordonnancement obtenu ne respecte pas le temps de cycle et l'encours n'est obtenu que pour un temps de cycle associé non minimal. Aussi l'heuristique mémorise l'ordonnancement des ressources et recalcule l'encours, en cherchant à minimiser l'en-cours tant que le temps de cycle optimal est respecté.

Par conséquent, la méthode d'ordonnancement d'Hillion comporte deux phases. Une première phase consiste à calculer l'ordonnancement des opérations sur les machines (ce qui est évident sur un graphe d'événements). La deuxième phase concerne la minimisation des encours, celle-ci requiert le calcul des circuits minimaux.

L'heuristique de C. Valentin Le principal inconvénient de la méthode précédente est quelle ne prend pas en compte les contraintes temporelles associées aux machines. En effet, le degré de liberté est seulement fonction du temps opératoire restant et du temps requis pour la séquence opératoire.

L'heuristique de C. Valentin [Val94] essaie d'améliorer l'algorithme précédant en associant à chaque machine un ensemble d'intervalles de **disponibilité des ressources** tel que chaque intervalle est plus grand que le plus petit temps des opérations restantes. Par conséquent, chaque intervalle peut contenir au moins une des opérations restantes.

Ce critère supplémentaire pour l'heuristique permet l'obtention de meilleurs ordonnancements que précédemment. Ainsi, l'heuristique de C. Valentin fonctionne sur le même principe que l'heuristique de H. P. Hillion, c'est à dire une élaboration de l'ordonnancement en deux étapes : l'élaboration de la séquence d'entrée et si nécessaire la minimisation des encours.

Cette méthode intègre plusieurs inconvénients. Le premier est de diviser le problème de l'ordonnancement en deux sous-problèmes, ce qui ajoute des contraintes lors de la

résolution et ne garantit pas l'obtention de l'optimalité de la solution obtenue. De plus, un autre problème concerne la fonction de placement. En effet, lors de l'étude de la fonction de coût et lorsque la fin du placement des opérations coïncide avec la fin du cycle, le dénominateur de l'équation (1.1) est égal à zéro et une division par zéro apparaît (quand At_j est égal à zéro : ce qui se produit lorsque la fin de la dernière opération placée coïncide avec la date CT du nombre de palettes $(1 + r_j)$).

Enfin, si le critère prend en compte les machines, il existe une seule fonction de coût pour prendre en compte deux problèmes différents :

- le respect du temps de cycle ;
- la minimisation des encours.

L'heuristique de H. Ohl En comparaison avec les heuristiques précédentes, la première différence réside sur le fait qu'au lieu d'ordonnancer chaque séquence d'opérations une à une, il est possible de **regrouper plusieurs séquences d'opérations**, en utilisant le même type de palettes. Par conséquent, chaque palette sera dédiée à un groupe spécifique de séquences d'opérations. Bien sur, il convient de considérer tous les regroupements possibles pour obtenir la meilleure solution possible.

Un intérêt supplémentaire de cette méthode est la fonction de coût. En effet, la fonction de placement est basée sur deux critères, ce qui est logique car nous avons deux contraintes : le temps de cycle et l'encours.

- soit $RTM[i]$ la date de disponibilité de la ressource i ;
- soit $RTT[i]$ le temps opératoire restant pour la ressource i ;
- $CT - RTM[i]$ représente la date de disponibilité pour la ressource i ;
- $RMM[i] = CT - RTT[i] - RTM[i]$ est la marge de la ressource i.

La fonction caractérise le respect du temps de cycle et ne doit jamais être négative pour que l'ordonnancement obtenu soit réalisable. Ce terme à été défini pour garantir un ordonnancement en utilisant l'heuristique **une seule fois**. En fait, durant le calcul, tant que $RMM[i]$ n'est pas négatif, l'ordonnancement est admissible et sans problème de division par zéro.

Le second critère est une fonction de coût réelle. Elle dépend de la marge de l'opération en cours d'exécution et en particulier de l'encours considéré pour le respect du critère de minimisation de l'encours.

L'heuristique de H. Ohl [Ohl95] est basée sur un placement progressif des opérations. Pour chaque itération, la fonction de placement considère la séquence d'opérations de coût minimal et place la première opération de la séquence.

1. l'opération est ordonnancée dés que possible avec l'encours correspondant au regroupement considéré ;
2. ce placement engendre un temps δ de non utilisation de la ressource ($\delta \leqslant 0$). Ainsi, la marge opérationnelle peut être évaluée ;
3. Si la marge de la ressource est positive ou nulle (ce qui est le cas généralement) alors l'ordonnancement considéré est valide et l'on effectue l'itération suivante ;
4. si la marge de la ressource est négative, alors la ressource est utilisée durant un temps trop long et l'ordonnancement doit être fait dès que possible pour cette machine. Par conséquent, il est nécessaire d'ajouter un encours. Il en résulte une perte de marge pour le processus mais la marge de la machine reste la même et l'ordonnancement devient admissible ; alors on effectue l'itération suivante de l'heuristique.

L'ordonnancement obtenu est dit avec cycles indépendants. C'est-à-dire qu'une opération doit être exécutées dans un même cycle.

L'heuristique de Camus-Korbaa Cette heuristique résulte des trois heuristiques précédentes. L'idée des auteurs H. Camus et O. Korbaa est de capitaliser les avantages des travaux précédents. Par conséquent, la politique de placement progressif (Hillion) est considéré, ainsi que les intervalles de disponibilité des ressources (Valentin) et l'idée d'obtenir une solution admissible après une seule exécution de l'algorithme (Ohl). Le caractère innovant est la prise en compte des chevauchements de cycles (une opération peut commencer dans un cycle et se terminer durant le cycle suivant), ainsi que les différentes politiques de placement (le plus tôt possible dans le système, au plus tôt dans un intervalle et au plus tard dans un intervalle).

Les algorithmes génétiques Cette approche est basée sur le codage en utilisant une représentation discrète du temps. Chaque chromosome définit un ordonnancement des opérations sur une période correspondant à un temps de cycle. Un chromosome est représenté par une matrice. Chaque élément de la matrice correspond à une opération de la séquence opératoire d'une pièce particulière. Chaque ligne de la matrice contient les opérations que

doit effectuer une machine particulière. Chaque gène est une opération caractérisée par une séquence d'opérations particulière, son ordre d'exécution et sa date de départ.

Cette méthode possède plusieurs avantages :

- un chromosome représente un ordonnancement réalisable et les conflits de ressources n'existent pas ;
- les contraintes de précédence de l'ordonnancement opératoire n'ont pas besoin d'être respectées dans le codage. Quand les contraintes de précédence ne sont pas respectées, les opérations de la séquence considérée sont alors effectuées avec un encours plus grand.

1.6.3 L'heuristique de J. K. Lee

Cette méthode, développée par Lee et al. [Lee02], basée sur le dépliage des réseaux de Petri, est utilisée pour vérifier les propriétés des réseaux de Petri. Dans un premier temps, cette approche peut permettre l'analyse du problème de l'ordonnancement; ce qui nécessite deux étapes, la première consiste à diviser le réseau de Petri en sous réseaux (BUC) et la seconde étape en une analyse des BUC en utilisant un algorithme de dépliage. Les dates de franchissements des transitions peuvent être calculées par évaluation de celles-ci sur le réseau déplié. Le meilleur ordonnancement du réseau de Petri temporisé déplié est obtenu par minimisation d'une fonction de coût; la méthode de dépliage considère une évolution du temps opératoire d'une machine basée sur le nombre de jetons (nombre d'exemplaire de la ressource partagée), et le temps opératoire de la machine. Les caractéristiques critiques de cette méthode sont :

1. le choix du BUC basé sur une matrice transitive ;
2. la notion d'ordre dans le BUC est basé sur un degré associé aux ressources ;
3. l'analyse du BUC basée sur le dépliage ;
4. une solution d'un BUC est obtenue à l'aide de solutions de sous BUC.

1.6.4 Les approches exactes

Ce paragraphe fait référence aux théories mathématiques élaborées pour l'évaluation des performances des réseaux de Petri temporisés. Pour cette étude, le modèle réseau de Petri est utilisé uniquement comme outil de représentation, puis une abstraction à l'aide

d'un formalisme mathématique permet l'évaluation du comportement temporel des systèmes de la classe des systèmes dynamiques à événements discrets (DEDS).

La théorie des DEDS, à l'aide d'une vision mathématique, date du début des années 1980 [CG79, Zim81, GM84], elle est motivée par une caractéristique d'une sous classe des DEDS : les **G**raphes d'**E**vénements **T**emporisés (GET), *leur comportement est complètement déterminé par la connaissance des dates de début et de fin des tâches.* Aussi, le point de vue exposé est une abstraction du comportement des GET en un système d'équations linéaires à l'aide d'une structure algébrique particulière. Chaque équation caractérise le franchissement d'une transition en fonction du marquage des places et des tirs des transitions en amont de celle-ci.

L'étude d'un tel système est complexe en général. En effet, il représente une traduction du comportement du réseau de Petri dans un langage mathématique. Par contre, la **linéarité** est une propriété permettant une analyse aisée d'un tel système [Plu90]. Cette propriété n'est hélas validée que pour une seule classe de réseau de Petri : les graphes d'événements simples (GES). L'absence de conflits et de pondérations caractérisent leur comportement essentiellement déterministe.

Pour les GES, un événement (arrivée d'un client, envoi d'un signal, achèvement d'une tâche ...) donne lieu à des phénomènes de synchronisation, de saturation ou de concurrence. Ces phénomènes peuvent caractériser les systèmes de production (ateliers flexibles) les systèmes multiprocesseurs, les réseaux informatiques, réseaux de transport ... Ce qui constitue une large classe d'applications. Toutefois la flexibilité des systèmes manufacturiers qui se traduit par des conflits et des indéterminismes ne peut être prise en compte par les GES.

La problématique qui concerne cette thèse est relative à l'étude du comportement des GET contenant des tâches répétitives. À l'aide du formalisme algébrique basé sur les dioïdes [Gau92a], le comportement temporel du système est abstrait en un système d'équations linéaires.

Des problématiques différentes existent et plusieurs dioïdes leurs sont associés. Nous allons présenter deux des dioïdes dédiés à l'évaluation de performances dans les GET non pondérés. L'objet est d'aboutir à la détermination des dates de franchissement des transitions.

(max,+) : Ce dioïde est utilisé pour répondre à la problématique suivante : « *déterminer la date de franchissement au plus tôt de la $k^{ème}$ exécution de la tâche i pour les graphes d'événements temporisés* ». Concrètement, toute date $x_i(k)$ correspond au franchissement de la transition i du graphe d'événements pour la k^{eme} occurrence. Le principe de cette étude est de considérer les contraintes déterministes de précédences du modèle afin d'exhiber le système d'équations linéaires utilisant les deux opérateurs primitifs max et $+$. Le système obtenu capture l'ensemble des ordonnancements possibles. En effet, si l'on note :

- $x_i(k)$ la date du $k^{\text{ème}}$ franchissement de la transition t_i ;
- τ_{ij} le temps de transport ou de production d'une pièce entre les tâches j et i ;
- m_{p_i} la valeur qui donne le nombre d'éléments constitutifs du stock initial dans la place p_i.

Le formalisme peut être illustré par une formule générique permettant de déterminer la date de franchissement d'une transition t_i du réseau de Petri en fonction de ses transitions amont selon la relation (1.2) :

$$x_i(k) = \max_{t_j \in {}^{\bullet\bullet}t_i} \{\tau_{ij} + x_j(k - m_{\bullet t_i \cap t_j \bullet})\} \tag{1.2}$$

Avec les conventions suivantes :

$$\forall i \in \mathbb{N}, \forall k < 0, x_i(k) = -\infty = \varepsilon \text{ et } x_i(0) = 0$$

En considérant l'exemple d'illustration de la figure 1.5, le principe consiste à exprimer la variable $x_1(k)$ qui évalue la date du $k^{\text{ème}}$ franchissements de la transition t_1, en fonction du nombre du nombre de jetons initialement présents dans les places amont (p_1 et r_1). En effet, sur l'exemple représenté par la figure 1.5 pour franchir k fois la transition t_1, il est nécessaire de franchir au préalable :

- la transition t_4 : $k - m_{p_1}$ fois (soit ici $k - 1$ fois) ;
- la transition t_2 : $k - m_{r_1}$ fois (soit ici $k - 1$ fois) ;

Donc, t_1 est validée pour la $k^{\text{ème}}$ fois à la date maximale évaluée dans les expressions suivantes : $\{2 + x_2(k-1)$ et $1 + x_4(k-1)\}$.

En étendant ce raisonnement à toutes les transitions du réseau de Petri de l'exemple de la figure 1.5, nous obtenons le système d'équations suivant :

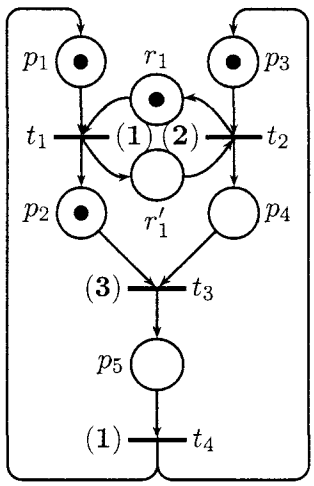


FIG. 1.5 – *Exemple de graphe d'événements T-temporisé*

$$\begin{cases} x_1(k) = \max\{2 + x_2(k-1); 1 + x_4(k-1)\} \\ x_2(k) = \max\{1 + x_1(k-1); 1 + x_4(k)\} \\ x_3(k) = \max\{1 + x_1(k-1); 2 + x_2(k)\} \\ x_4(k) = x_3(k) + 3 \end{cases} \tag{1.3}$$

Ce système possède la propriété de linéairité au sens du dioïde (max,+). En effet, chaque équation du système (1.3) s'exprime comme maximum ($\oplus$ = max) et de sommes ($\otimes$ = +) de variables caractérisant les franchissements des transitions du système.

(min,+) La problématique duale peut être considérée : « *déterminer le nombre maximal de franchissements n_i à la date t pour la tâche i dans un graphe d'événements temporisé* ». L'étude de l'évolution temporelle du graphe d'événements temporisé est sensiblement différente. On privilégie maintenant le décompte d'occurrences de mise à feu des transitions en fonction du temps.

La formule générique suivante permet l'abstraction d'un GET en un système d'équations également linéaires :

$$n_i(t) = \min_{t_j \in {}^{\bullet\bullet}t_i} \left\{ m_{{}^{\bullet}t_i \cap t_j{}^{\bullet}} + n_j(t - \tau_{ij}) \right\} \tag{1.4}$$

Avec comme convention $\forall t < 0, \forall i \in \mathbb{N}, n_i(t) = 0$. Pour l'exemple 1.6, nous obtenons le système d'équations suivant :

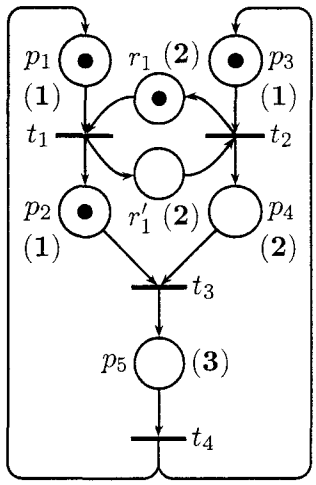


FIG. 1.6 – *Exemple de graphe d'événements P-temporisé*

$$\begin{cases} n_1(t) = \min\{1 + n_2(t-2); 1 + n_4(t-1)\} \\ n_2(t) = \min\{n_1(t-2); 1 + n_4(t-1)\} \\ n_3(t) = \min\{1 + n_1(t-1); n_2(t-2)\} \\ n_4(t) = n_3(t-3) \end{cases} \tag{1.5}$$

L'intérêt réside tant pour les équations (1.5) que l'équation (1.4) dans le caractère récurrent des systèmes présentés précédemment. De plus, l'utilisation d'un formalisme algébrique permet l'utilisation de techniques de résolutions, telles que l'expression des valeurs propres [CTCG$^+$] pour la recherche du comportement asymptotique, et de la fonction d'entrée (pour obtenir une sortie désirée) .... Cette théorie est développée pour les graphes d'événements temporisés simples.

Les méthodes exactes sont très intéressantes pour caractériser le comportement des GET. En particulier il apparaît dans les équations (1.5) la possibilité de tenir compte du marquage initial du graphe dont dépend le comportement d'exécution parallèle du modèle GET. Les seules opportunités d'ordonnancement qu'il est possible d'exprimer sur ce modèle concernent la recherche du marquage (nombre des marques et positionnement initial de ces marques) permettant une exécution parallèle et optimisée de ce modèle dynamique. Cette approche est particulièrement intéressante dans le cas de modèles bouclés à comportement cyclique selon l'exemple de la figure 1.6

Cependant les systèmes de production manufacturière ne sont que partiellement concernés par cette formalisation algébrique. De plus, ce type d'atelier flexible intègre très

souvent des assemblages complexes (modélisés par des arcs pondérés) ainsi que des partages de ressources. Cette méthode de calcul n'est donc pas applicable en l'état. Une extension existe pour les GET pondérés [CGQ98], par transformations des graphes d'événements pondérés en graphes d'événements simples [Jen92, Mun93] ainsi que pour le cas de politiques de routage lorsque des conflits sont présents dans le modèle [OCG94, CGQ98].

Nous tenterons donc de voir dans quelle mesure il est possible d'étendre ces résultats au cas des modèles comportant des assemblages de multiples composants et également des conflits d'accès aux ressources.

1.6.5 Les marquages partiels

Une autre technique intéressante en vue de résoudre un problème d'ordonnancement a été menée par Ahmer Benasser dans sa thèse [Ben00]. Elle consiste à rechercher les conditions de propagation d'un marquage initial hypothétique appelé marquage partiel garantissant l'accès à un marquage résultant. Ce problème d'accessibilité est ici résolu de façon originale selon une technique qui s'apparente à la propagation de contraintes mais en ne considérant que les marquages possibles sur le modèle support de type réseau de Petri [BY99, BY01].

Cette technique intègre différentes notions notamment le concept de marquage partiel. L'idée est ici d'associer à chaque place un couple contenant un vecteur indexé sur l'ensemble des places et une contrainte linéaire, liant les variables du vecteur. Le but est d'abstraire un ensemble de marquages atteignables après le franchissement de steps partiels. La notion de step s'entend ici comme un sous-ensemble de transitions simultanément franchissable.

Un step partiel est ainsi défini à l'aide d'un vecteur indexé sur l'ensemble des transitions et d'une contrainte linéaire. Tout comme le marquage partiel, le step partiel permet de représenter un ensemble de steps par instanciations des variables respectant la contrainte linéaire.

Bien sûr, la franchissabilité d'un step partiel à partir d'un marquage partiel se fait sous certaines conditions. Il est nécessaire que pour une valuation donnée, la contrainte linéaire associée au marquage partiel et au step partiel soient vérifiées, ainsi qu'une contrainte linéaire représentant la validité du step partiel vis à vis du marquage partiel (veiller à ne pas considérer une séquence contenant un marquage négatif).

Le but de cette méthode est de reporter la complexité de la résolution de la phase finale à la résolution des contraintes linéaires à l'aide d'un solveur de contraintes. En effet, le résultat est que l'utilisation des steps partiels capture toutes les séquences possibles entre deux marquages et un nombre d'étapes données. Dans ce sens tous les ordonnancements candidats de transitions sont évalués lors de la résolution du problème d'accessibilité évoqué.

Des applications ont été menées dans le cadre de l'ordonnancement des ressources, pour le **H**oist **S**cheduling **P**roblem et pour un problème critique de transport visant à optimiser l'ordonnancement de trains sur le nœud ferroviaire SNCF complexe de Gonesse [BY99].

1.7 Conclusion

Au cours de ce chapitre nous avons évoqué dans le contexte de la production flexible manufacturière les différents travaux relatifs à l'évaluation optimisée d'un ordonnancement cyclique. Compte tenu de la complexité de ces ordonnancements il apparaît que la majorité des recherches engagées sur ce sujet aboutissent à des heuristiques de plus en plus efficaces. En particulier les dernières approches de Ouajdi Korbaa poussent très loin les techniques de linéarisation avant d'aboutir à la mise en œuvre d'une heuristique d'ordonnancement qui considère à ce niveau un réseau de Petri relativement proche d'un GET. Les travaux d'Ahmer Benasser sont une autre façon d'aborder le problème en considérant d'une part le marquage comme un vecteur de paramètres et d'autre part la notion de step partiel qui tient compte a priori du parallélisme d'exécution des opérations.

D'un autre point de vue la modélisation mathématique basée sur l'algèbre des dioïdes est extrêmement séduisante mais se limite au cas des GET sans pondération. L'ensemble de ces considérations nous a conduit à rechercher dans un premier temps l'opportunité de prendre en compte les graphe d'événements pondéré par une approche exacte puis le cas des conflits d'accès aux ressources simples.

Première partie

Étude du comportement logique d'un réseau de Petri

Introduction

Dans un premier temps, nous aborderons le problème des indéterminismes dans les réseaux de Petri, plus particulièrement ceux liés aux partages de ressources. Dans certaines configurations, les ateliers flexibles possèdent des machines multi-tâches partagées entre différents usinages à réaliser. Il convient donc de séquencer celle-ci pour l'obtention d'une commande de l'atelier.

Dans les travaux antérieurs, ce problème est résolu en utilisant des politiques de placements ou des fonctions de routages [OCCG94]. Des fonctions de routages sont aussi utilisées dans le cadre d'utilisation de structures algébriques [Tou82, MM90, CGQ95b, CGQ98].

Notre but est de ne négliger aucun séquencement possible afin de garantir, après l'évaluation des performances, un ordonnancement final optimal. Nous sommes donc amenés à envisager une autre approche sous les hypothèses de fonctionnement suivante :

- d'une part, les nombres d'opérations que doit subir une pièce est connu et déterministe, seules les machines partagées entre plusieurs tâches induisent des conflits structurels.
- d'autre part, le nombre d'affectations pour chaque machine par cycle ont été déterminées dans une phase d'optimisation antérieure du flux [Kor98, Cam97, Ohl95]. Le modèle résultant des phases antérieures d'optimisation est un RdP d'un type particulier puisque relativement proche du modèle graphe d'événements pondéré. La pondération découle le plus souvent de la prise en compte de lots de composants ou de pièces ou encore de processus complexes d'assemblage. Outre cette pondération la seule autre non-linéarité résiduelle résulte des conflits d'accès aux ressources dont l'affectation n'est pas déterminée à ce niveau de l'étude. Ce modèle de production intègre donc l'assemblage ou le désassemblage complexe, ce qui n'était pas envisagé antérieurement par l'équipe SED du LAIL.

L'objet de cette première partie de notre étude concerne donc la transformation

d'un RdP vérifiant les hypothèses décrites précédemment en un modèle sans conflits structurels [TB02, TBG02b]. Sachant que le système de production est tenu à une production définie à l'avance, l'idée consiste à intégrer les hypothèses de production dans le modèle construit afin de lever l'indéterminisme. Cette transformation permettra d'obtenir un graphe d'événements pondéré associé à un système d'équations contraignant le marquage initial d'un sous-ensemble de places du GEP.

Le plan adopté pour cette première partie comporte quatre paragraphes :

En premier lieu, nous définirons la classe de réseaux de Petri concernés par cette étude.

En second lieu, nous décrirons un exemple de réseau de Petri. Ce modèle analyse différents types de partages de ressource.

Dans le troisième paragraphe, nous présenterons une étude de classification des différents types de partages de ressources. Dans cette perspective, nous étudierons les partages de ressources pour lesquels les transitions en conflits sont franchies une seule fois par cycle. Dans ce cas les arcs connectant les transitions à la place contenant les ressources, ne sont pas pondérés. Le premier point de cette étude concerne la résolution de l'indéterminisme structurel du réseau de Petri. L'étude des partages de ressources est décomposée en deux sous-paragraphes. Tout d'abord, nous considérerons le cas d'une seule machine, puis l'hypothèse de plusieurs machines partagées entre les différentes tâches à exécuter. Cette restriction induit la résolution d'un problème de complexité abordable pour permettre de bien comprendre le principe de la résolution présentée.

Le quatrième paragraphe est consacré à la généralisation de la problématique précédente. Nous supposons toujours qu'une ou plusieurs machines sont partagées entre différentes tâches, par contre ces opérations peuvent être exécutées plusieurs fois au cours d'un cycle. Nous verrons que cette extension peut également être résolue.

Le dernier paragraphe contient une méthode visant à la résolution de la dernière extension du problème précédent. L'objectif est de résoudre un conflit d'accès aux ressources pour lequel les tâches requièrent un ou plusieurs exemplaires de la ressource. Cette extension se caractérise par des pondération sur les arcs. La technique de résolution permettra l'obtention d'un graphe d'événements couplé à un système d'inéquations contraignant le marquage initial du réseau.

Enfin une conclusion terminera ce chapitre.

Chapitre 2

Résolution de partage de ressources

2.1 Définition du cadre de l'étude

Le but de ce paragraphe est de définir une classe potentielle d'application de la théorie exposée dans ce chapitre. La définition de cette classe de réseaux de Petri est faite de manière constructive.

Définition 2 *Nous appellerons graphe d'événements avec ressources (GER), un réseau de Petri* $R = (P \cup P_R, T, W)$ *tel que :*

1. $P \neq \varnothing$ *et* $T \neq \varnothing$
2. (P,T,W) *est un graphe d'événements composé d'une ou plusieurs composantes fortement connexe*
3. $\forall p \in P_R$, $|^\bullet p| = |p^\bullet|$

Remarque 1 *le graphe d'événements définit par* (P,T,W) *peut être non pondéré ou pondéré, ainsi que les arcs d'extrémité une des places l'ensemble* P_R. *À la différence de la définition donnée dans [TE97, FTE] où* (P,T,W) *est une machine à états.*

les places spéciales contenues dans l'ensemble P_R seront dénommées par r_i, pour signifier la présence d'un partage de ressources entre plusieurs tâches à réaliser. En effet, pour l'exemple représenté par la figure 2.1, nous obtenons les caractéristiques suivantes :

- $P_R = \{r_1, r_2\}$ dont les éléments correspondent aux différentes ressources partagées dans le système modèlisé

- $P = \{p_1,p_2,p_3,p_4,p_5,p_6,p_7\}$, $T = \{t_1,t_2,t_3,t_4,t_5,t_6\}$ est un graphe d'événements composé de deux composantes fortement connexe :
  - $\{p_1,p_2,p_3,p_4,p_5\}$, $\{t_1,t_2,t_3,t_4\}$
  - $\{p_6,p_7\}$, $\{t_5,t_6\}$
- $\forall p \in P_R$, $|{}^\bullet p| = |p^\bullet|$

Donc l'exemple proposé appartient à la classe de réseau de Petri définie précédemment.

Dans ce chapitre nous considérerons uniquement le champ d'application restreint par la définition suivante :

Définition 3 *Nous utiliserons les graphes d'événements avec ressources définis par un réseau de Petri $R = (P \cup P_R,T,W)$ tel que :*

1. $P \neq \varnothing$ *et* $T \neq \varnothing$
2. (P,T,W) *est un graphe d'événements composé d'une ou plusieurs composantes fortement connexe*
3. $\forall p \in P_R$, ${}^\bullet p = p^\bullet$

En effet, la classe d'application définie par la définition 3 permet de proposer des exemples illustratifs simples et représentatifs de la problématique abordé. De plus, nous verrons des exemples d'applications de cette méthode pour des exemples plus générique représentatifs de la définition 2.

2.1.1 Exemple illustratif

Nous proposons d'illustrer notre étude à partir de l'exemple décrit figure 2.1. L'intérêt est ici de pouvoir illustrer de manière pertinente la première difficulté rencontrée dans l'étude des réseaux de Petri à l'aide de structures algébriques de type dioïde. Il permet également de se faire une idée sur la classe d'applications potentielles de ce travail.

La chaîne de production modélisée figure 2.1 permet de fabriquer deux produits, A et B, à l'aide de deux types de ressources partagées (représentées par les places r_1 et r_2). La ressource r_1 est disponible en deux exemplaires et la ressource r_2 existe en un seul exemplaire. Ceci est représenté par le nombre des jetons dans les places r_1, r_2.

Le produit A est composé d'une pièce A_1 et de trois pièces A_2. Le premier composant A_1 est fabriqué par l'usinage de la pièce située en p_1, ce qui est symbolisé par le franchissement de la transition t_1. La fabrication du deuxième composant est effectué sur la

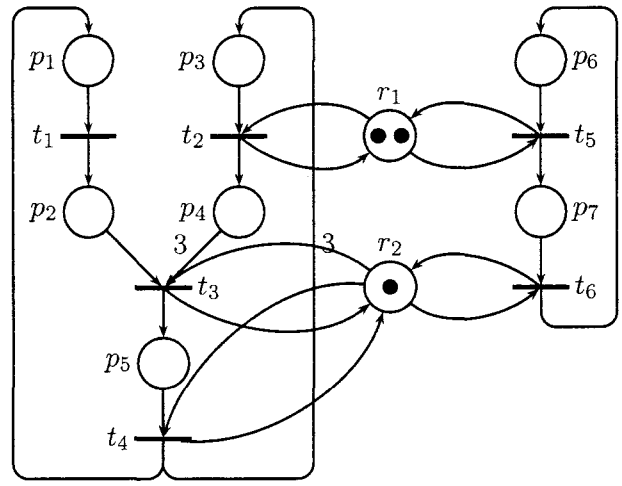


FIG. 2.1 – *Exemple d'illustration*

transition t_2 en utilisant une ressource r_1. La machine de type r_2 effectue alors l'assemblage d'une pièce A_1 et des trois pièces A_2, au niveau de la transition t_3. Enfin, le résultat de l'assemblage est contrôlé par la machine de type r_2 pour obtenir le produit final (transition t_4).

Le deuxième produit B est usiné selon une gamme linéaire à l'aide successivement des même machines r_1 et r_2. Afin de caractériser le comportement cyclique du système, le modèle est rebouclé à l'aide des arcs joignant respectivement la transition t_4 aux places p_1 et p_3, et la transition t_6 à la place p_6.

On décrira l'exemple d'une production pour laquelle on cherche à produire par cycle un produit fini A pour un produit fini B, dans ce cas la fréquence d'utilisation des transitions par cycle est d'un franchissement pour chaque transition terminale du réseau t_4 ou t_6 et en particulier pour les tâches nécessitant une machine partagée, représentées par les transitions t_5 ou encore t_3, t_4 et t_6, la transition t_2 doit être franchie trois fois par cycle. Le nombre de franchissements par cycle interviendra lors de la résolution.

Remarquons que le conflit induit par le partage des deux ressources r_1 peut paraître inutile, il permet cependant de limiter le nombre des franchissements simultanés de la transitions t_2. Afin de caractériser la production nous devons définir une politique de routage. Cette question a été évoquée dans l'article de Guy Cohen [CGQ95a], sans considérer le caractère discret du routage, mais par approximation sur les flux réels de matière. Au cours de travaux antérieurs [OCG94] a été évoqué l'idée du contrôle d'une production quantifiée

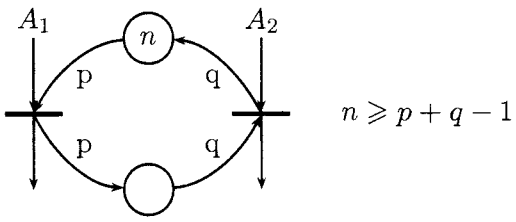


FIG. 2.2 – *Illustration d'une politique de routage*

par exemple de q produits A_1 et p produits A_2 par l'adjonction d'un graphe de contrôle de la synchronisation qui conduit au modèle décrit figure 2.2. Dans ce cas la pondération des arcs est inévitable. Pour notre exemple et afin de simplifier la présentation nous considérons le cas le plus simple d'une production d'un même nombre de pièce A et de pièce B. Nous allons donc décrire dans ce cas la technique de résolution des partages de ressources rencontrés pour notre exemple.

Considérons la figure 2.1 : il existe deux conflits induits par le partage des deux ressources r_1 et r_2. Notre objectif est d'étudier le comportement logique du réseau de Petri en utilisant le dioïde $(\min, +)$ ce qui nécessite le caractère déterministe du réseau de Petri étudié. Cependant, notre exemple comporte plusieurs conflits potentiels. Par exemple, les transitions t_2 et t_5 sont en conflit d'accès aux exemplaires de la ressource r_1. Les machines de type r_1 sont ainsi partagées entre les deux tâches caractérisées par les transitions t_2 et t_5.

2.2 Résolution de partage de ressources avec franchissement unitaire des transitions

Après avoir introduit l'exemple sur lequel nous allons appliquer les résultats à venir, nous aborderons la résolution effective du problème relevant des partages de ressources au sein de notre chaîne de production. Dans ce cas un ou plusieurs exemplaires de la ressource sont partagés entre plusieurs tâches exécutées une seule fois par cycle. La résolution de ce problème est impérative pour aborder la modélisation du RdP initial à l'aide d'un formalisme algébrique. En effet, le conflit d'accès aux ressources induit un système d'équations non-déterministe et une résolution beaucoup plus complexe.

Pour palier ce problème, nous utilisons une technique de transformation du réseau

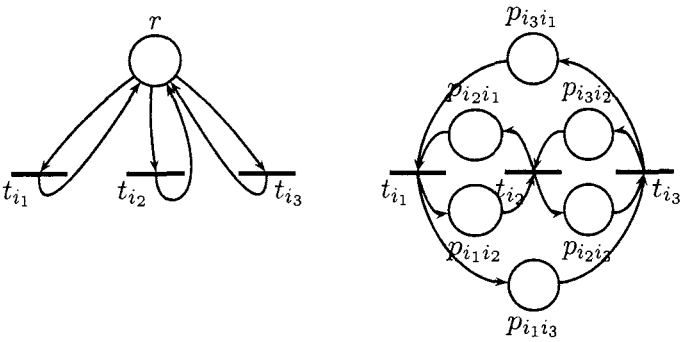


FIG. 2.3 – *Exemple de partage de ressources*

de Petri. Le but est de rendre le réseau de Petri déterministe sans perdre aucun séquencement possible. Pour ce faire, nous présentons une méthode de transformation du réseau de Petri, puis nous étudions les différents types de partages de ressources pour adjoindre au réseau un système d'équations contraignant le comportement du sous réseau constitué par les transitions en conflit. Le but de cette étude est de transformer le réseau de Petri intégrant des conflits d'accès aux ressources en un graphe d'événements dont le marquage est contraint par un système d'équations. Le système d'équations constitue une abstraction de tous les séquencements possibles des ressources partagées.

Dans cette partie nous ne considérons que les partages de ressources avec franchissement unitaire des transitions en conflit lors d'un cycle. Le but est d'aborder progressivement différentes classes de partages de ressources afin de simplifier l'étude de chacun des cas. Le premier thème abordé est celui de la transformation structurelle (paragraphe 2.2.1) du réseau de Petri pour obtenir un réseau déterministe. Les parties suivantes seront consacrées à la détermination du système d'équations associé au RdP transformé. Nous considérerons le cas d'une ressource simple (paragraphe 2.2.2) puis d'un nombre quelconque de ressources uniques (paragraphe 2.2.3).

2.2.1 Construction du graphe d'événements

Si l'on suppose que la fréquence de déclenchement des transitions par cycle est déterminée, le problème du séquencement des ressources consiste ici à déterminer l'ordre des n tâches (éventuellement dans le cas d'exécution parallèle) à réaliser : cela revient ici à définir un ordre total de déclenchement des transitions en conflit. En terme de réseau de Petri, il s'agira donc de déterminer une séquence de steps pour laquelle chaque transition

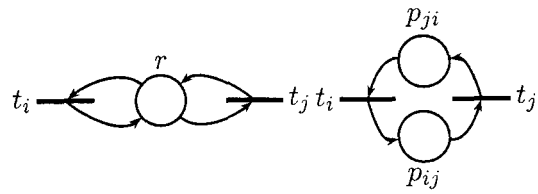


FIG. 2.4 – *Exemple de transformation*

apparaîtra une et une seule fois, la taille des steps étant limité par le nombre d'exemplaires disponibles de la machine.

L'idée consiste donc à construire un réseau de Petri déterministe, déduit du modèle initial pour lequel nous pouvons décrire, à l'aide d'un marquage initial spécifique, toutes les séquences potentielles de transitions franchissables lorsque ces transitions partagent m ressources identiques. Nous devons donc imposer un ordre partiel sur les transitions, c'est ainsi que nous introduisons deux places pour chaque couple de transitions, le marquage de ces places définira la relation d'ordre entre deux transitions $t_i \preceq t_j$ ou $t_j \preceq t_i$. Dans ce but, nous proposons la transformation illustrée par la figure 2.4 et utilisant la définition donnée ci dessous :

Définition 4 *Nous utilisons l'abréviation de TGES pour le réseau de Petri obtenu par transformation de chaque place r qui représente une ressource partagée, cette transformation est décrite ci-dessous :*

- *Nous éliminons la place initiale r associée au partage de ressources.*
- *$\forall i,j \in \mathbb{N}, 1 \leqslant i < j \leqslant n$, nous introduisons deux places représentées par p_{ij} et p_{ji} entre les deux transitions t_i et t_j (voir la figure 2.4).*

Nous obtenons à l'aide de cette transformation un graphe d'événements simple (ou pondéré) selon la nature du RdP initial simple (ou pondéré). Le premier intérêt d'une telle transformation est le déterminisme du graphe obtenu ; le second est le comportement (min ,+)-linéaire d'un tel système est dans le cas d'un RdP initial simple. Considérons l'exemple illustratif de la figure 2.5, cette transformation caractérise le partage d'une seule ressource entre quatre tâches, représentées par les transitions t_1, t_2, t_3 et t_4. Dans la partie suivante, nous verrons qu'il est possible de déterminer un marquage initial de ce TGES associé à l'un quelconque des ordres possibles de franchissement des transitions $\{t_1, t_2, t_3, t_4\}$ et réciproquement. Pour prouver ce résultat nous utilisons le vecteur d'occurrences représenté par $N(k) = (n_i(k))_{1 \leqslant i \leqslant n}$, nous obtenons ainsi pour l'exemple de la figure 2.5 l'équation suivante :

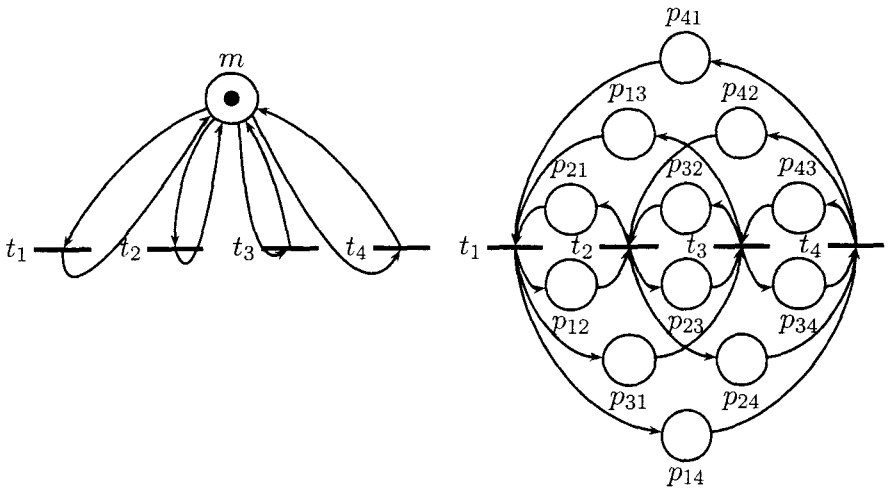


FIG. 2.5 – *Exemple de résolution du partage de ressources*

$$N(k+1) = B(M(0)) \otimes N(k) \text{ où } B(M(0)) = \begin{pmatrix} \infty & m_{p21} & m_{p31} & m_{p41} \\ m_{p12} & \infty & m_{p32} & m_{p42} \\ m_{p13} & m_{p23} & \infty & m_{p43} \\ m_{p14} & m_{p24} & m_{p34} & \infty \end{pmatrix}$$

La complexité en termes de construction de ce nouveau modèle pour n transitions et une seule place de partage de ressources sur le modèle initial peut facilement être évalué en évaluant le cardinal de Δ l'ensemble des éléments non diagonaux de $B(M(0))$.

$$\triangle = n^2 - n = n(n-1)$$

Le cardinal de Δ représente le nombre de places créées pour un partage de ressources simple entre n transitions.

On obtient ainsi un graphe d'événements dont il reste à déterminer le marquage initial. Quelles conditions doit alors vérifier le marquage initial pour qu'il corresponde à l'un quelconque des séquencements réalisables?

2.2.2 Le cas d'une seule machine

Nous étudierons tout d'abord le cas le plus simple d'une seule machine. Nous démontrerons l'existence d'une équivalence entre la résolution d'un partage de ressources à une machine entre plusieurs tâches et une recherche de marquage initial dans le TGES. Nous montrerons la possibilité d'une abstraction de l'ensemble des séquencements possibles

de la ressource (auxquels correspond un ensemble de marquages initiaux), en un système d'équations dont les variables sont les marquages initiaux des places mises en jeu. Le système d'équations traduit des contraintes sur le marquage initial, ainsi que des contraintes imposées par le système. La preuve se fera en deux étapes : Dans un premier temps, Pour un séquencement de la ressource il correspond un marquage initial spécifique du TGES. Puis dans un second temps, pour un marquage initial du TGES respectant certaines contraintes, il correspond un et un seul séquencement de la ressource.

Dans cette partie, nous considérerons le cas d'une machine partagée entre un ensemble de transitions $T' = \{t_1, \ldots, t_d\}$ connectées à la place r.

Nous supposons que toutes les tâches sont exécutées une seule fois en ne disposant que d'une seule machine. Séquencer consiste donc à définir un ordre strict sur les tâches en conflit. Un séquencement définit un ordre de franchissements des steps successifs. Chaque step est un sous-ensemble de transitions simultanément franchissables. Prenant en compte l'hypothèse d'une seule machine partagée entre plusieurs tâches (chaque tâche n'étant exécutée qu'une seule fois). nous pouvons énoncer une description particulière d'un séquencement des tâches.

Définition 5 *Pour un séquencement $\sigma = t_{i_1} \ldots t_{i_d}$, si t_i est franchie avant t_j alors cette condition sera représentée par $t_i \prec t_j$ $(1 \leqslant i \neq j \leqslant d)$.*

La loi $\prec$ permet la représentation des contraintes de précédence parmi deux transitions du sous-ensemble de transitions T'. De plus, elle possède la propriété ci-dessous :

Proposition 1 *La relation $\prec$ est une relation d'ordre strict sur l'ensemble des transitions T'.*

Preuve. On vérifie facilement que cette relation est une relation d'ordre strict. Vérifions que cette relation est transitive : en effet, supposons que, pour trois transitions t_i, t_j et t_k, nous savons que t_i est franchie avant t_j et que t_j est franchie avant t_k. Nous en déduisons que t_i est franchie avant t_k. Ce qui se traduit formellement par la formule suivante :

$$\forall t_i, t_j, t_k \in T', \text{ si } t_i \prec t_j \text{ et } t_j \prec t_k \text{ alors } t_i \prec t_k$$

Nous venons de démontrer que la relation $\prec$ est transitive.

Vérifions que la relation est anti-reflexive : montrons que l'on ne peut pas comparer élément à lui même, i.e. $\forall t \in T', \neg(t \prec t)$. Soit t une transition du TGES, aucune contrainte

de précédence ne peut être établie sur une même transition t. Donc la relation $\prec$ est anti-reflexive. Par suite, la relation $\prec$ est une relation d'ordre strict entre les transitions. ■

Puisque nous ne considérons pour l'instant que le cas d'une seule machine, la relation $\prec$ permet de définir l'ordre strict dans lequel les tâches sont effectuées sur la machine. Les transitions étant maintenant bien ordonnées, nous considérons en toute généralité une séquence de transitions de l'ensemble T' numérotées dans l'ordre naturel de franchissement. Nous obtenons ainsi : $t_1 \prec t_2 \prec \ldots \prec t_d$.

À l'aide de la relation $\prec$, nous pouvons construire un marquage initial du GES comme suit :

Définition 6 *Pour chaque séquencement, nous appelons « marquage initial associé », le marquage du TGES défini par :*

$$1 \leqslant i \neq j \leqslant d, \text{ si } t_i \prec t_j \text{ alors } p_{ji} \text{ contient un jeton et } p_{ij} \text{ ne contient aucun jeton.}$$

Cette définition exprime que la transition t_j ne peut être validée avant que la transition t_i ne soit franchie. À l'aide de cette définition, nous pouvons obtenir une relation générale que doivent vérifier les transitions du sous-ensemble T' des transitions en conflit.

Proposition 2 *Pour chaque séquencement, le marquage du graphe d'événements associé (définition 6) vérifie :*

$$\forall 1 \leqslant i \neq j \leqslant d,\ m_{p_{ij}} + m_{p_{ji}} = 1 \tag{2.1}$$

et induit le comportement caractérisé par le séquencement.

Preuve. Comme précédemment, sans perte de généralité, nous pouvons naturellement ordonner les transitions de l'ordre de franchissement (cet ordre suit l'ordre numérique). Numérotons les transitions dans l'ordre de franchissement de celles-ci; ainsi nous obtenons l'ordre des transitions données ci-dessous :

$$t_1 \prec t_2 \prec \ldots \prec t_d$$

Supposons que la transformation du réseau de Petri ait été réalisée en respectant les conditions de la définition 4. Nous nous intéressons maintenant au comportement du réseau de Petri obtenu. Pour cela nous évaluons les vecteurs d'occurrences successifs (somme des steps franchis). L'état initial est : $N(0) = (0, \ldots, 0)^T$ pour lequel les compteurs d'occurrences sont

initialisés à zéro. $N(k)$ est obtenu par recurrence à partir de $N(k-1)$ et donc à partir de $N(0)$ par l'équation suivante :

$$N(k) = B(M(0))^k \otimes N(0) \tag{2.2}$$

La relation (2.2) contient la matrice $B(M(0))$, matrice est carrée de dimension d, et l'opérateur $\otimes$ représentant l'opérateur matriciel $(\min, +)$. La matrice $B(M(0))$, de ce réseau de Petri, et ses puissances successives sont données ci-dessous :

$$B(M(0)) = \begin{pmatrix} \infty & 1 & \dots & 1 \\ 0 & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & 1 \\ 0 & \dots & 0 & \infty \end{pmatrix} ; B(M(0))^2 = \begin{pmatrix} 1 & 1 & \dots & 1 & 2 \\ 1 & 1 & \dots & 1 & 1 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \dots & 0 & 1 & 1 \end{pmatrix}$$

$$B(M(0))^3 = \begin{pmatrix} 1 & \dots & \dots & 1 & 2 & 2 \\ 1 & & & & 1 & 2 \\ 1 & & & & & 1 \\ 0 & \ddots & & & & \vdots \\ \vdots & \ddots & \ddots & & & \vdots \\ 0 & \dots & 0 & 1 & 1 & 1 \end{pmatrix}$$

Au rang d, nous obtenons :

$$B(M(0))^d = \begin{pmatrix} 1 & 2 & \dots & 2 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 2 \\ 1 & \dots & \dots & 1 \end{pmatrix}$$

À partir des équations (2.2), nous obtenons facilement les valeurs successives que prend le vecteur d'occurrences $N(k)$:

$$N(1) = (1,0,0,\dots,0)^T$$
$$N(2) = (1,1,0,\dots,0)^T$$
$$\dots$$
$$N(d) = (1,1,1,\dots,1)^T$$

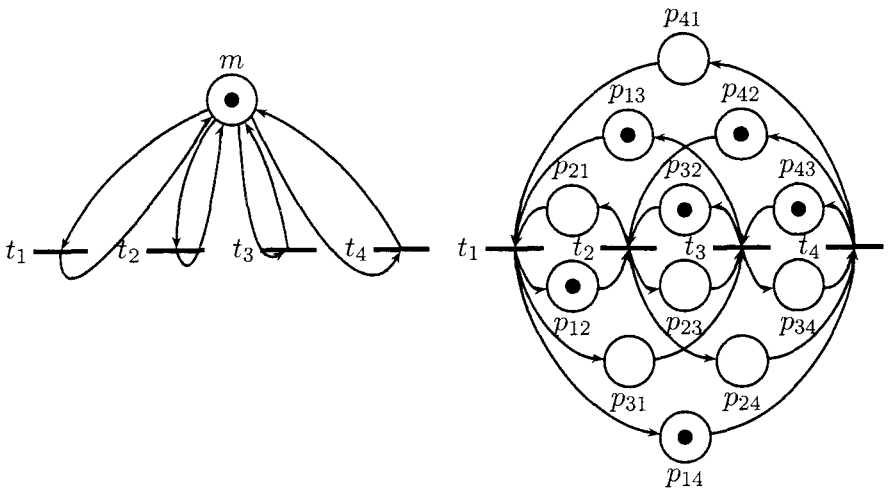


FIG. 2.6 – *Exemple de marquage intial*

Nous déduisons que chaque step successif contient une et une seule transition différente ($S(k) = N(k) - N(k-1)$). De plus, nous déduisons de cette dernière expression l'équation suivante :

$$\forall k \in [\![1,d]\!], S(k) = t_k$$

Ainsi le TGES satisfait le comportement $t_1 \prec t_2 \prec \ldots \prec t_d$ voulu en respectant la définition 6. Ce résultat est, en fait, indépendant de l'une quelconque des séquences initiales choisies t_1, $t_2, \ldots,$ t_d. Donc à chaque séquence de tir, définissant un ordre parmi les transitions, correspond un marquage initial du TGES. Pour conclure, le jeu d'équations, défini par l'équation générique (2.1), est satisfait de par la construction de TGES, grâce à la definition 6. ■

Par conséquent, nous venons de prouver que chaque séquencement possible est produit par un et un seul marquage initial spécifique du TGES solution du système d'équations défini précédemment. Le TGES est donc le modèle générique d'un séquencement séquentiel paramétrable par un marquage initial. Puisque nous considérons le cas du franchissement unitaire des transitions sur un cycle, Il est facile de déduire qu'il existe pour d transitions en conflit $(d-1)!$ permutations possibles, et autant de séquencements différents possibles. La proposition 2 induit qu'il existe au moins $(d-1)!$ marquages initiaux de notre sous réseau. Ainsi tous ces séquencements sont générés par un système d'équations pour lequel les solutions résultent des marquages initiaux du TGES dont la structure reste invariante.

Pour notre exemple, nous avons décrit sur la figure 2.6 le marquage initial associé à la séquence de tir $t_1 \prec t_2 \prec t_3 \prec t_4$. Nous vérifions aisément que l'équation (2.1) est

satisfaite.

Réciproquement, nous allons établir que pour un marquage initial du TGES respectant les contraintes définies précédemment il lui correspond une séquence de tir spécifique dans le réseau de Petri initial.

Proposition 3 *Pour chaque marquage initial du graphe d'événements associé vérifiant les contraintes, il correspond un séquencement de l'affectation de la ressource.*

Preuve. Supposons que l'ensemble des contraintes de la proposition 2 soient satisfaites. Alors pour tout circuit élémentaire de deux transitions t_i et t_j, les deux places p_{ij} et p_{ji} de ce circuit satisfont la condition (2.1) suivante :

$$m_{p_{ij}} + m_{p_{ji}} = 1$$

Nous en déduisons que l'une des deux places est marquée, c'est-à-dire p_{ij} ou p_{ji} contient un jeton. Supposons que p_{ji} contienne un jeton. Par conséquent pour ce circuit élémentaire, t_i est validée avant la transition t_j et pour que la transition t_j soit validée il doit y avoir un jeton dans la place en amont de la transition t_j (la place p_{ij}); c'est-à-dire il est nécessaire que la transition t_i doit être franchie. Donc un ordre strict vient d'être décrit entre ces deux transitions. Ces contraintes donnent l'ordre suivant entre ces deux transitions : $t_i \prec t_j$. La propriété est vérifiée pour l'ensemble des transitions, et les contraintes de précédence sont globalement vérifiées. Ainsi la définition d'une notion d'ordre, entre deux transitions de l'ensemble T', permet de construire un ordre strict sur l'ensemble T', ce qui induit un séquencement des affectations successive de la ressource. En effet, le résultat obtenu est l'ordre de franchissement des transitions en conflit. ■

Notons que dans le cas contraire, le marquage initial obtenu correspond à un blocage comme décrit ci après :

Remarque 2 *Considérons l'ensemble des solutions vérifiant le système de contraintes. Nous remarquons qu'il existe des marquages initiaux pour lesquels aucune transition n'est franchissable. En effet, considérer les contraintes de précédences de d'une paire de transitions ne permet évidemment pas d'assurer la vivacité de la solution globale. À titre d'exemple, considérons le marquage initial $M(0)$ décrit sur la figure 2.6 avec :*

$$m_{22} = 1,\ m_{32} = 1,\ m_{43} = 1,\ m_{14} = 1$$

Il vérifie les contraintes, mais il représente un blocage car une incohérence est présente dans la définition de l'ordre global entre les transitions de l'ensemble T', nous obtenons :

$$t_1 \prec t_2,\ t_2 \prec t_3,\ t_3 \prec t_4 \text{ et } t_4 \prec t_1$$

Par la suite, nous verrons que la satisfaction des « conditions de vivacité » permet de supprimer les solutions représentant un séquencement non vivant du système d'équations obtenu.

2.2.3 Le cas de plusieurs machines

Dans le paragraphe 2.2.2, nous avons exhibé une relation nécessaire que doit vérifier le marquage initial pour que le TGES représente tous les séquencements possibles dans le cas particulier où la ressource est unique. Par la suite, nous étudions le cas plus général de plusieurs exemplaires d'une même machine partagée entre plusieurs tâches.

Le problème consiste en un partage de α ($\alpha \in \mathbb{N}^*$) ressources identiques partagées entre d tâches, nous supposons de nouveau que d transitions doivent être franchies une et une seule fois par cycle. Le but est d'obtenir un séquencement de l'affectation de ces machines, ce qui revient à déterminer une séquence de steps dont la taille est inférieure ou égale au nombre des machines. Le parallélisme d'exécution induit est au plus égal à α. On dira que la taille d'un step est au plus égale à α.

Définition 7 *Un séquencement, dans un réseau contenant des ressources multiples partagées entre n tâches, est défini par une séquence de steps S_1,S_2,...,S_k. Chaque transition apparaît une et une seule fois dans la séquence et la taille maximale du step ne dépasse pas le nombre de machines identiques : α.*

Dans ce contexte nous considérons que deux transitions, au moins, peuvent être franchies simultanément. La relation d'ordre strict $\prec$, définie précédemment, ne convient plus. Par conséquent, il est nécessaire, de définir deux nouvelles relations :

Définition 8 *Nous définissons les notations suivantes :*

- $t_i \sim t_j$ *si* t_i *et* t_j *sont franchies lors du même step (franchisement en parallèle : $t_i \,//\, t_j$)*
- $t_i \preceq t_j$ *si* $t_i \sim t_j$ *ou* $t_i \prec t_j$

Proposition 4 $\sim$ *est une relation d'équivalence.*

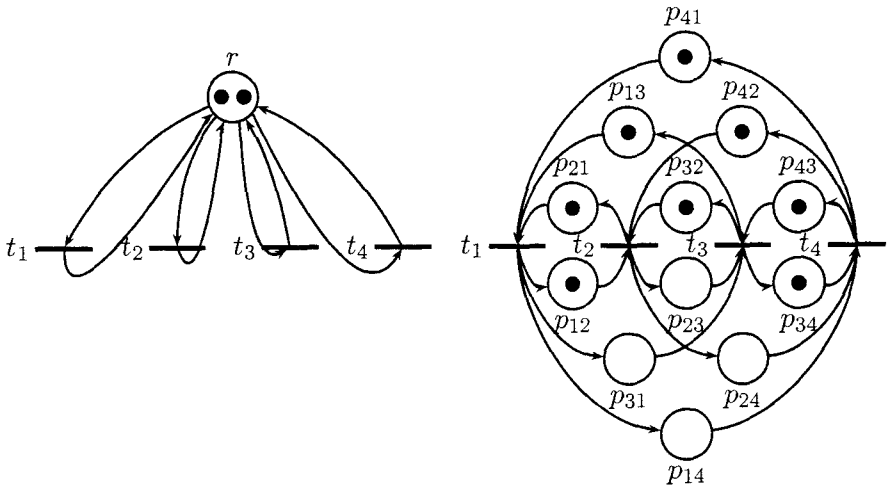


FIG. 2.7 – *Exemple de résolution de partage de deux ressources*

Preuve. Prouvons que la relation $\sim$ est bien une relation d'équivalence. Nous devons montrer que la relation est :

i) Reflexive : $\forall(t_i) \in T'$, $t_i \sim t_i \Leftrightarrow t_i \sim t_i$

ii) Symétrique : $\forall(t_i,t_j) \in T'^2$, $t_i \sim t_j \Leftrightarrow t_j \sim t_i$

iii) Transitive : $\forall(t_i,t_j,t_k) \in T'^3$ $t_i \sim t_j$ et $t_j \sim t_k \Leftrightarrow t_i \sim t_k$

i) La relation est reflexive car une transition t_i ne peut être franchie lors d'un seul step.

ii) Montrons maintenant que cette relation est symétrique. Le fait que deux transitions soient franchies lors du même step n'induit pas de relation d'ordre entre deux transitions d'un même step, puisqu'elles sont franchies en même temps. Ainsi pour deux transitions t_i et t_j de l'ensemble T', on a l'équivalence suivante : $t_i \sim t_j \Leftrightarrow t_j \sim t_i$.

iii) À l'aide du même argument que précédemment. Nous montrons aisément la transitivité de la relation $\sim$. En effet, si t_i est franchie en même temps que t_j et si t_j est franchie ne parallèle avec t_k, alors nous avons : $t_i \sim t_k$.

On conclut que la relation $\sim$ est une relation d'équivalence. ■

Ainsi dans le cas où $t_i \sim t_j$, nous dirons que les deux transitions appartiennent à la même classe d'équivalence. Le sous-ensemble de transitions représentant la classe d'équivalence d'une transition t_i sont les transitions franchissables simultanément avec t_i. Par suite, cette notion correspond à celle de step et la relation $\preceq$ définit une relation d'ordre sur l'ensemble des steps franchissables. Nous pouvons maintenant ordonner les classes d'équivalences (steps) suivant leurs ordres de franchissement.

Sur l'exemple représenté par la figure 2.7, nous avons présenté la résolution d'un

partage de deux ressources entre quatre transitions. Le séquencement retenu vérifie les relations suivantes :

$$\begin{array}{ll} t_1 \sim t_2 & t_1 \prec t_3 \\ t_3 \sim t_4 & t_1 \prec t_4 \\ & t_2 \prec t_3 \\ & t_2 \prec t_4 \end{array}$$

Nous en déduisons la séquence de steps associée suivante :

$$t_1 \,/\!/\, t_2 \prec t_3 \,/\!/\, t_4$$

Définition 9 *Un marquage d'un TGES M et un séquencement S_1, $S_2, \ldots, S_l$ sont associés si $M[S_1\ S_2 \ldots S_l\rangle M$.*

À titre d'exemple, si l'on considère la figure 2.7. Nous associons le marquage du TGES à le séquencement $S_1 = t_1 \,/\!/\, t_2$ puis $S_2 = t_3 \,/\!/\, t_4$. En effet, le marquage donné sur la figure de droite représente une solution du problème du séquencement pour le problème énoncé précédemment. Nous allons voir comment il est possible d'obtenir le marquage du TGES pour un séquencement quelconque.

La construction de la structure TGES est effectuée suivant la même méthode décrite dans la partie précédente. Cependant du fait qu'il existe α exemplaires de la ressource, il peut exister un sous-ensemble de transitions (au plus α) simultanément franchissables. Le lemme suivant donne une condition suffisante pour qu'un marquage du TGES induise un comportement où les steps ont pour chacun d'eux une taille inférieure ou égale au nombre de machines : α.

Lemme 1 *Si un marquage initial vérifie les conditions suivantes :*

$$\forall (t_{i_1}, \ldots, t_{i_{\alpha+1}}) \subset T^{\alpha+1}, \left(\sum_{j=1}^{\alpha} m_{p_{i_{j+1} i_j}} \right) + m_{p_{i_1 i_{\alpha+1}}} \leqslant \alpha \tag{2.3}$$

Alors le franchissement simultané de $\alpha + 1$ transitions dans le même step est impossible.

Preuve. Lorsque l'on considère un circuit élémentaire dans le TGES, nous obtenons un invariant de place défini par la somme des marquages des places qu'il contient. La

somme des marquages des places constituant le circuit est invariante quelles que soient les transitions franchies et ainsi le marquage obtenu après le franchissement d'un step valide vérifie l'inéquation (2.3). Par suite, nous pouvons réduire la preuve, en ne démontrant que la proposition suivante :

Si un marquage initial vérifie les conditions suivantes :

$$\forall (t_{i_1}, \ldots, t_{i_{\alpha+1}}) \subset T^{\alpha+1}, \left(\sum_{j=1}^{\alpha} m_{p_{i_{j+1} i_j}} \right) + m_{p_{i_1 i_{\alpha+1}}} \leqslant \alpha$$

Alors avec ce marquage, il y a, au plus, $\alpha + 1$ transitions validées.

Supposons que $\alpha + 1$ transitions soient franchissables dans le premier step, alors toutes les places en amont de chacune des transitions doivent posséder au moins un jeton. Considérons un circuit contenant uniquement les $\alpha + 1$ transitions citées précédemment, ce circuit possède donc $\alpha + 1$ places. Chacune des places étant en amont d'une des $\alpha + 1$ transitions, supposées validées, elles contiennent au moins un jeton. Par conséquent, nous venons de décrire un circuit de $\alpha + 1$ transitions contenant $\alpha + 1$ jetons, ce qui contredit l'hypothèse que traduit l'inégalité (2.3). ■

Sur l'exemple représenté figure 2.7, nous obtenons une série de contraintes qui dérivent de l'inéquation (2.3). En particulier, cela signifie que pour tout circuit élémentaire contenant deux transitions (soit trois places), il ne doit y avoir que deux jetons au total des marquages des trois places. Sachant que chaque place de ce sous réseau ne peut contenir qu'un seul jeton, nous pouvons reformuler l'équation (2.3) par : dans tout circuit élémentaire du TGES contenant trois places seules deux d'entre elles doivent être marquées d'un seul jeton.

Proposition 5 *À chaque séquencement possible est associé un marquage initial dans le TGES respectant les contraintes données par la relation (2.3).*

Preuve. Soit σ un séquencement représenté sous forme d'une séquence σ de steps : $\sigma = S_1, S_2, \ldots, S_k$. Le but est de construire un marquage M pour lequel nous ayons $M[\sigma\rangle M$. En fait, le marquage M engendre un régime cyclique ayant le comportement décrit par la séquence de steps σ.

La construction de ce marquage M est effectuée en deux étapes :

- Si $t_i \prec t_j$ alors nous plaçons un jeton dans la place p_{ji} en amont de la transition t_i et aucun jeton dans la place p_{ij} en aval de celle-ci.

- Si $t_i \sim t_j$ alors nous plaçons un jeton dans les deux places du circuit connectant les transitions t_i et t_j, ceci exprime le fait qu'il n'y ait pas de contraintes de précédence entre ces deux transitions.

On ordonne comme précédemment les transitions telles que $t_1 \preceq t_2 \preceq \ldots \preceq t_d$. Par conséquent, on obtient la matrice $B(M(0))$ triangulaire supérieure par blocs suivante:

$$B(M(0)) = \begin{pmatrix} A_1 & \mathrm{I} & \ldots & \mathrm{I} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathrm{I} \\ 0 & \ldots & 0 & A_l \end{pmatrix} \text{ où } A_i = \begin{pmatrix} \infty & 1 & \ldots & 1 \\ 1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 1 & \ldots & 1 & \infty \end{pmatrix} \text{ avec : } \dim(A_i)=\delta_i$$

I représente une matrice pour laquelle les cœfficients sont égaux à un et possède la taille ad hoc. A_i représente la matrice caractéristique du step S_i. En effet, S_i est un step contenant toutes les transitions simultanément franchies. ainsi, lorsque l'on considère un circuit contenant deux transitions d'un même step, les deux places intermédiaires sont marquées d'un jeton. C'est dans ce sens que tous les éléments non diagonaux de A_i sont deux à deux égaux à 1, ceci explique la forme particulière des matrices A_i. La dimension δ_i de A_i exprime donc le fait que lors du step S_i le nombre des transitions franchies simultanément est égal à δ_i.

Il reste maintenant à prouver que le marquage ainsi défini engendre le comportement logique spécifié. Calculons, dans ce sens, les puissances successives de la matrice $B(M(0))$. Nous obtenons les matrices suivantes:

$$B(M(0))^2 = \begin{pmatrix} I & I & \ldots & I & II \\ I & I & \ldots & I & I \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & I \\ 0 & \ldots & 0 & I & I \end{pmatrix} ; B(M(0))^3 = \begin{pmatrix} I & \ldots & \ldots & I & II & II \\ I & & & & I & II \\ I & & & & & I \\ 0 & \ddots & & & & \vdots \\ \vdots & \ddots & \ddots & & & \vdots \\ 0 & \ldots & 0 & I & I & I \end{pmatrix} ; \ldots ;$$

au rang k, nous obtenons :

$$B(M(0))^k = \begin{pmatrix} I & II & \dots & II \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & II \\ I & \dots & \dots & I \end{pmatrix}$$

Le symbole II représente une matrice dont les cœfficients sont tous égaux à 2 et de taille ad hoc. Nous pouvons alors facilement calculer la valeur du vecteur d'occurrence après le franchissement de chaque step, en utilisant la relation de récurrence :

$$N(k+1) = B(M(0)) \otimes N(k)$$

et nous obtenons donc l'expression du vecteur d'occurrence $N(k)$ comme suit :

$$N(k) = B(M(0))^k \otimes N(0)$$

$$\forall 1 \leqslant i \leqslant l, N(i) = (\mathbb{1}_1, \dots, \mathbb{1}_i, 0, \dots, 0)^T$$

$\mathbb{1}_i$ représente un vecteur pour lequel les cœfficients sont tous égaux à 1 avec la dimension ad hoc (la taille de $\mathbb{1}_i$ est égale à la dimension de la matrice carrée A_i). Nous déduisons facilement les steps franchis $S_i = N(i) - N(i-1)$ qui correspondent précisément aux sous-ensembles de transitions définis par les classes d'équivalences. Nous concluons que $M[S_1 S_2 \dots S_k\rangle M$. Comme précédemment, le résultat est prouvé indépendamment de tous les ordres de franchissements possibles.

Montrons maintenant que le marquage initial ainsi défini satisfait bien la relation (2.3) du lemme 1. Raisonnons par l'absurde et supposons l'existence d'un circuit contenant $(\alpha + 1)$ transitions $t_{i_1}, \dots, t_{i_{\alpha+1}}$, chaque transition t_{i_j} possède une unique place en entrée et en sortie (appelée respectivement $p_{i_j i_{j-1}}$ et $p_{i_{j+1} i_j}$), pour laquelle le marquage des places vérifie : $\left(\sum_{j=1}^{\alpha} m_{p_{i_{j+1} i_j}}\right) + m_{p_{i_1 i_{\alpha+1}}} \geqslant \alpha + 1$. Aussi dans ce circuit, nous ne considèrons que $\alpha + 1$ places contenant au plus un jeton. Ainsi, nous concluons que chaque place $p_{i_{j+1} i_j} = {}^\bullet t_{i_j} \cap t_{i_{j+1}}{}^\bullet$ considérée contient un jeton. Nous concluons, par la défintion de la relation entre les transitions, que :

$$t_{i_1} \preceq t_{i_2} \preceq \dots \preceq t_{i_\alpha} \preceq t_{i_1} \tag{2.4}$$

Nous allons prouver que ces transitions sont toutes équivalentes. Supposons, au contraire, que $t_{i_u} \prec t_{i_v}$ pour un $1 \leqslant u \neq v \leqslant \alpha$, nous déduisons de la relation (2.4), le résultat suivant :

$$t_{i_1} \preceq \ldots \preceq t_{i_u} \prec t_{i_v} \preceq \ldots \preceq t_{i_1}$$

Par conséquent, $t_{i_1} \prec t_{i_1}$, ce qui est absurde par la définition de l'ordre « $\prec$ ». Nous concluons que toutes les $\alpha + 1$ transitions sont équivalentes, c'est-à-dire franchies lors du même step, ce qui implique que le séquencement S_1, $S_2 \ldots S_l$ n'est pas valide. ■

Nous venons de prouver que pour chaque séquencement, il existe un marquage initial tel que les contraintes (2.3) soient satisfaites. Le paragraphe suivant traite de la réciproque :

Pour un marquage initial donné vérifiant les contraintes imposées, existe-t-il un séquencement associé?

Proposition 6 *Si M est un marquage initial du TGES vérifiant la relation (2.3) alors soit le marquage induit un blocage, soit il existe un séquencement cyclique σ tel que $M[\sigma > M$.*

Preuve. Soit M un marquage vérifiant la relation (2.3). Nous supposerons que le marquage M n'induit pas de blocage, c'est-à-dire que nous faisons l'hypothèse de l'existence d'une transition vivante, que l'on nommera t_i. Nous supposerons de plus l'existence séquence de steps telle que :

$$S_1 S_2 \ldots S_k \text{ telle que } M[S_1 S_2 \ldots S_k\rangle$$

Le franchissement de la transition t_i de ce circuit implique la consommation des jetons des places en amont de cette transition. Or :

$$\forall j \neq i, \{p_{ji}\} = {}^\bullet t_i \cap t_j{}^\bullet \tag{2.5}$$

Le franchissement de t_i redevient possible uniquement lorsque les places amont sont de nouveau marquées. L'équation (2.11) permet de dire que le tir de t_i n'est possible qu'après le franchissement de toutes les transitions du TGES. Ainsi l'indice k peut être choisi de telle façon que la séquence de steps $S = S_1 S_2 \ldots S_k$ soit la plus grande possible et ne contienne qu'une occurrence et une seule de chaque transition. La transition t_i est supposée vivante et celle-ci ne peut être franchie que si les autres ont été tirées. Donc deux cas se présentent, soit S contient toutes les transitions et la preuve est terminée. Soit S ne contient pas toutes les transitions, nous en déduisons l'existence d'une transition non franchissable,

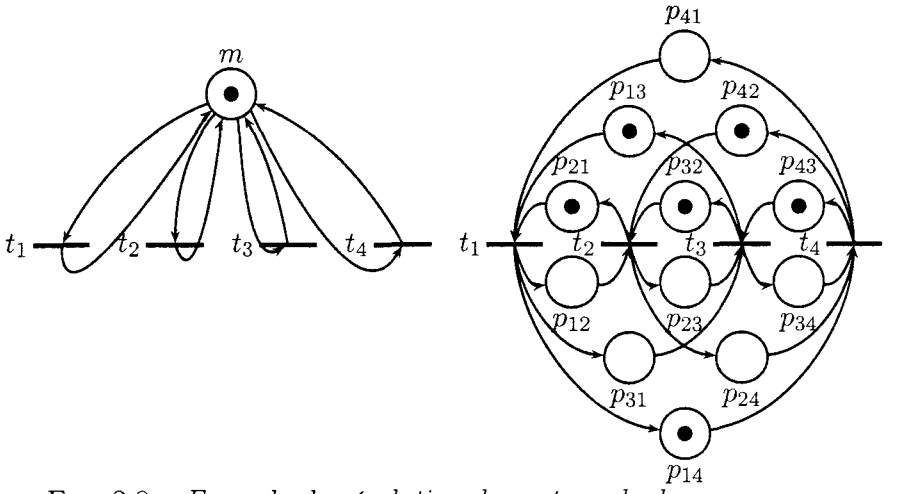


FIG. 2.8 – *Exemple de résolution de partage de deux ressources*

ce qui implique la non franchissabilité de la transition t_i à partir du marquage obtenu après le franchissement de la séquence S. Or compte tenu de la maximalité de la séquence S et de la vivacité de la transition t_i, nous concluons que ce cas est impossible. Par suite, S est une séquence de steps contenant toutes les transitions et la proposition 5 impose que le marquage initial vérifie une inégalité qui interdit le franchissement simultané de plus de α transitions. En conclusion, la séquence de steps $S = S_1 S_2 \dots S_k$ est un séquencement cyclique. ■

À ce stade de notre étude, nous sommes capable de transformer un partage de ressources pour obtenir un TGES auquel est adjoint un système d'équations contraignant le marquage initial de ce dernier. Par contre, nous illustrons la théorie ci-dessus sur le contre exemple 2.8, pour lequel nous donnons un marquage initial pour lequel toutes les transitions ne sont pas franchissables. En effet, nous constatons que les circuits élémentaires du TGES ne sont pas tous marqués. On conclut que le système d'équations possède de *fausses* solutions, dans le sens où ces *fausses* solutions conduisent à un blocage. Dans le paragraphe suivant, nous verrons qu'il est possible d'éliminer ces *fausses* solutions à priori grâce à une simple condition de vivacité du TGES.

2.2.4 Condition de vivacité

Dans la suite, le problème abordé est celui des conditions de vivacité dans le TGES. Puisque le TGES obtenu est un graphe d'événements simple, Il résulte de travaux antérieurs [CHEP71], la condition nécessaire et suffisante de vivacité suivante :

Théorème 1 *Pour qu'un graphe d'événements simple soit vivant il faut et il suffit qu'il existe au moins un jetons dans chaque circuit élémentaire du réseau.*

Dans l'exemple représenté par la figure 2.8, le circuit $t_1 \rightarrow t_3 \rightarrow t_4 \rightarrow t_1$ ne contient aucun jeton ($m_{p31} = m_{p34} = m_{p41} = 0$), ce qui confirme que notre TGES, muni du marquage représenté par la figure 2.8, n'est pas vivant.

Nous appliquons les conditions de vivacité à notre TGES pour obtenir un jeu d'équations supplémentaires. Le but est de contraindre le marquage du TGES pour éliminer les *fausses* solutions décrites précédemment. Il faut donc décrire tous les circuits élémentaires contenus dans le TGES. En effet, pour que le TGES soit vivant il faut et il suffit qu'il existe au moins un jeton dans chaque circuit (théorème 3). C'est-à-dire que la somme des marquages des places contenues dans chaque circuit considérée soit supérieure ou égale à 1.

De plus, un circuit élémentaire ne peut contenir plus de transitions que le nombre total de transitions du TGES, sous peine de répétition. Donc les inéquations ne peuvent contenir plus de quatre inconnues.

Nous obtenons un jeu d'inéquations supplémentaire, ces inéquations sont décrites ci-dessous pour notre exemple (figure 2.8 : 4 transitions et α machines) :

- Pour les circuits de taille 2 : $\forall 1 \leqslant i \neq j \leqslant \alpha,\ m_{ij} + m_{ji} \geqslant 1$
- Pour les circuits de taille 3 : $\forall 1 \leqslant i_1 \neq i_2 \neq i_3 \leqslant \alpha,\ m_{i_1 i_2} + m_{i_2 i_3} + m_{i_3 i_1} \geqslant 1$
- Pour les circuits de taille 4 : $\forall 1 \leqslant i_1 \neq i_2 \neq i_3 \neq i_4 \leqslant \alpha,\ m_{i_1 i_2} + m_{i_2 i_3} + m_{i_3 i_4} + m_{i_4 i_1} \geqslant 1$

Théorème 2 *Le système d'inéquations induit par la relation (2.3), augmenté du système d'inéquations induisant la vivacité (théorème 3), capture uniquement tous les séquencements cycliques possibles.*

Preuve. Soit S une séquence de step associées à un marquage M initial du TGES solution du système d'inéquations augmenté du système d'inéquations lié à la condition de vivacité. Aussi M est une solution particulière du système d'inéquations, donc la proposition 7 impose que S est un séquencement cyclique de la ressource ou conduit vers un blocage. De même, M est une solution particulière du système d'inéquation garantissant la vivacité du TGES. D'où S est un séquencement cyclique de la ressource.

Réciproquement, S est un séquencement cyclique de la ressource. Montrons que M constitue une solution du système d'inéquations augmenté du système d'inéquations. Par hypothèse S est un séquencement et induit un comportement périodique donc vivant. Aussi, nous en déduisons respectivement que M est une solution du système d'équations

car S est un séquencement, et M est une solution du système d'inéquations car S induit un comportement non bloquant. ■

En conclusion, considérons un réseau de Petri dont les indéterminismes ne proviennent que des partages de ressources, tel qu'il possède des contraintes de franchissements unitaires des transitions en conflits, sous réserve que les arcs en jeu ne soient pas pondérés. Le théorème 4 montre la possibilité de transformer le réseau de Petri en un graphe d'événements simple tout en préservant l'ensemble des comportements possibles. Nous allons voir par la suite que l'hypothèse des franchissement *unitaire* des transitions peut être relaxée.

2.3 Résolution du partage de ressources avec franchissement multiple des transitions

Dans la partie 2.2, nous avons abordé le problème des conflits d'accès aux ressources pour un franchissement unitaire des transitions. Considérons le cas où une ou plusieurs machines identiques sont partagées par plusieurs tâches, sans restriction sur le nombre de franchissements, à la différence des paragraphes 2.2.2 et 2.2.3.

2.3.1 Construction du graphe d'événements

L'idée consiste à se ramener au cas de l'étude précédente. C'est-à-dire transposer un problème de partage de ressources entre plusieurs transitions, dont plusieurs franchissements par cycle est permis, en un problème de partage de ressources entre transitions franchies une seule fois par cycle.

Dans ce but, nous proposons la procédure décrite ci-dessous :

- *première étape* : Chaque transition t est dupliquée autant de fois que t doit être franchie au cours d'un cycle. Nous obtenons un ensemble de transitions en conflit noté T'.
- *deuxième étape* : Nous effectuons la transformation décrite au paragraphe 2.2.1 sur l'ensemble de transition T'.
- *troisième étape* : Une recherche du marquage initial est effectuée, celle-ci est décrite précédemment (paragraphes 2.2.2 ou 2.2.3, suivant le nombre de machines en jeu).

Par conséquent, cette procédure permet de générer un TGES pour lequel chaque transition est franchie une seule fois. Nous retrouvons alors la problématique du paragraphe précédent, après avoir dupliqué certaines transitions du graphe initial.

2.3.2 Recherche du marquage initial

Notre approche de la recherche de marquage initial sur le TGES obtenu diffère sensiblement de la recherche de marquage initial menée lorsque nous faisions l'hypothèse de franchissement unique des transitions en conflit.

La méthode utilisée au sein du paragraphe 2.2.3 fonctionne pour le TGES obtenu, l'ensemble des séquencements obtenu est correct et cet ensemble recouvre l'ensemble des séquencements possibles. Par contre, les transitions dupliquées introduites précédemment représentent la même tâche. Il est donc inutile de considérer une notion d'ordre entre deux clones d'une même transition. Ainsi, nous convenons de définir un ordre par défaut des transitions dupliquées représentant l'ordre numérique affecté lors de leur création. Cette méthode vise à éliminer les séquencements redondants.

Nous présentons la méthode de recherche de marquage initial en intégrant la remarque précédente. Nous obtenons l'algorithme suivant :

- *Première étape* : Pour le sous-réseau du TGES formé des transitions dupliquées, une notion d'ordre parmi ces transitions ne présente aucun sens, puisque celles-ci représente la même tâche. Nous ordonnons ces transitions suivant l'ordre numérique qui leur est affecté. L'ordre précédent permet de définir le marquage des places intermédiaires entre deux clones d'une même transition. Le nombre de variables est ainsi réduit et la complexité de résolution du système d'équation global s'en trouve également réduit.
- *deuxième étape* : Nous considérons le TGES, il est composé à la fois de places dont le marquage est prédéterminé (première étape) et de places dont le marquage est inconnu. Nous résolvons ce problème avec le même algorithme exposé dans le paragraphe 2.2.2 ou le paragraphe 2.2.3 si le problème comporte respectivement une ou plusieurs ressources. La différence réside dans le fait que certaines variables utilisées lors de cette deuxième étape possèdent déjà une valeur établie lors de la première étape.

Le fait que le système d'inéquations capture tous les séquencements possibles de(s) ressource(s) repose sur le fait que les clones d'une transition t_i, de l'ensemble T', représentent une même tâche. En effet, considérons les séquencements qui diffèrent par permutations des clones d'une transition t_i (t_i^1, t_i^2 ...), les clones d'une transition t_i représentent une même tâche à exécuter. Par conséquent, une permutation des clones d'une transition t_i n'influe pas sur le séquencement final de(s) ressource(s). Par conséquent, le fait de fixer l'ordre de

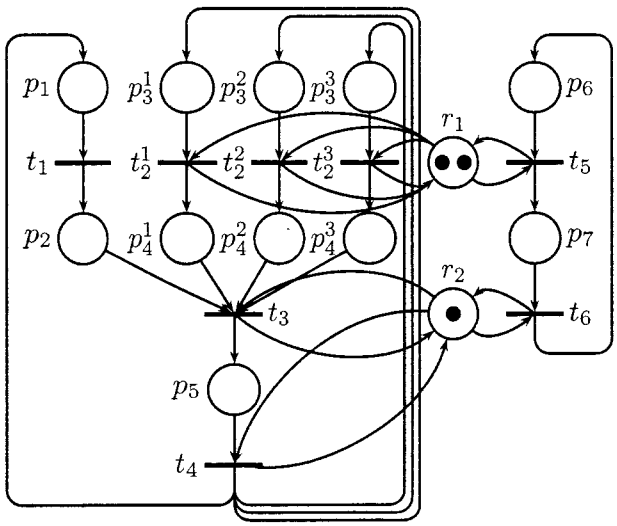


FIG. 2.9 – *Le réseau de Petri transformé*

franchissement des clones, ou de manière équivalente le marquage du sous réseau formé par les clones d'une transition t_i (première étape), n'est pas une restriction de l'ensemble des solutions. De plus, le fait de fixer le marquage d'un certain nombre de places du TGES permet de diminuer la complexité de résolution du système d'équations.

2.4 Application sur l'exemple d'illustration

Nous proposons maintenant d'appliquer cette théorie sur l'exemple illustratif 2.1. La ressource r_1 est une ressource partagées entre deux transitions t_2 et t_5, sachant que la transition t_2 est franchie trois fois par cycle et la transition t_5 une seule fois. Par conséquent la transitions t_2 est dupliqée trois fois. La ressource r_2 est partagées entre des transitions franchies une seule fois par cycle.

Le réseau de Petri est tout d'abord transformé selon la procédure décrite dans le paragraphe 2.2 et nous obtenons dans ce sens le RdP de la figure 2.9. Ce réseau de Petri possède la particularité que les ressources sont partagées entre différentes tâches exécutées une fois par cycle, notamment des clones d'une même tâche (transitions t_2^1, t_2^2, t_2^3).

Nous appliquons une nouvelle transformation qui vise à l'obtention d'un graphe d'événements simple (sans conflit). Le principe, décrit dans le paragraphe 2.2.1, permet de modifier la structure du réseau et nous obtenons le réseau de Petri décrit par la figure 2.10. Le marquage présent sur la figure 2.10 est fixé à l'avance, car il détermine l'ordre de fran-

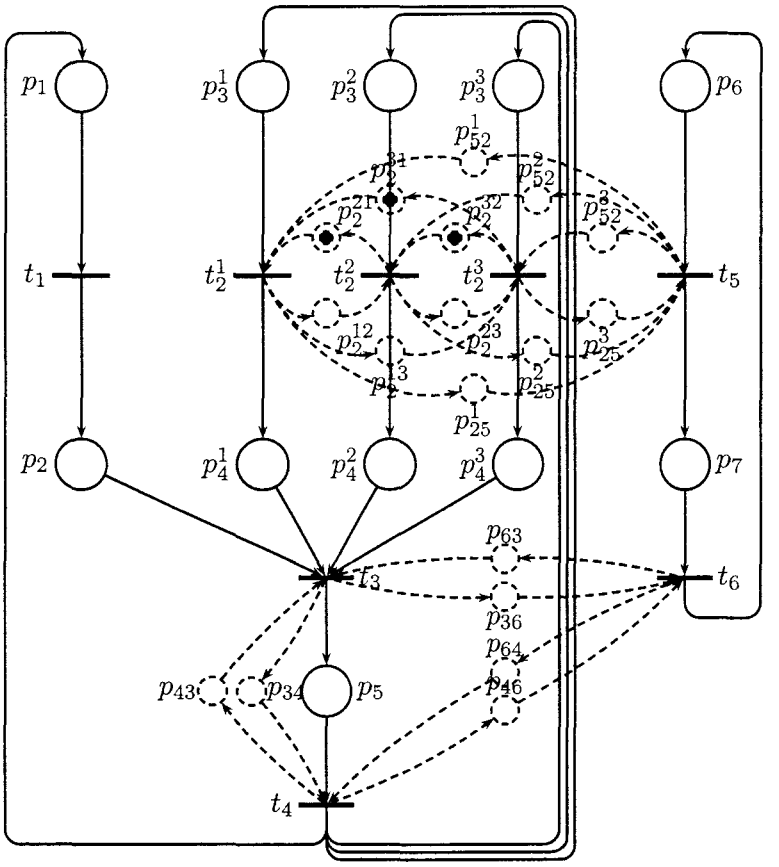


FIG. 2.10 – *Graphes d'événements pondéré résultant*

chissement des clones de la transition t_2. En effet, l'ordre de franchissement de ces clones étant aléatoire, il est préférable de fixer un ordre (numérique) pour ne pas augmenter la complexité de la résolution inutilement.

À ce graphe d'événements, il est adjoint deux systèmes d'équations contraignant le comportement des deux TGES associés aux ressources r_1 et r_2. Pour notre exemple, nous obtenons les systèmes suivants :

$$\text{Contraintes relatives à } r_2 \begin{cases} m_{p_{34}} + m_{p_{43}} \leqslant 1 \\ m_{p_{36}} + m_{p_{63}} \leqslant 1 \\ m_{p_{64}} + m_{p_{46}} \leqslant 1 \end{cases} \tag{2.6}$$

Par conséquent, le marquage initial du TGES associé à la ressource r_2 doit satisfaire les contraintes données par le système (2.6). Ce système d'inéquations représentent les contraintes de franchissements et les contraintes liées à la vivacité des solutions.

Pour le TGES associé à la ressource r_1, il y a deux exemplaires de la ressource. De plus, la ressource est partagées entre les transitions t_5 et t_2 (transitions t_2^1, t_2^2 et t_2^3). Le marquage initial doit vérifier le système d'équations suivant :

$$\text{Contraintes relatives à } r_1 \begin{cases} m_{p_2^{12}} + m_{p_2^{23}} + m_{p_2^{31}} \leqslant 2 \\ m_{p_2^{13}} + m_{p_2^{32}} + m_{p_2^{21}} \leqslant 2 \\ m_{p_2^{12}} + m_{p_{25}^2} + m_{p_{52}^1} \leqslant 2 \\ m_{p_{25}^1} + m_{p_{52}^2} + m_{p_2^{12}} \leqslant 2 \\ m_{p_2^{13}} + m_{p_{25}^3} + m_{p_{52}^1} \leqslant 2 \\ m_{p_{25}^1} + m_{p_{52}^3} + m_{p_2^{31}} \leqslant 2 \\ m_{p_2^{23}} + m_{p_{25}^3} + m_{p_{52}^2} \leqslant 2 \\ m_{p_{25}^2} + m_{p_{52}^3} + m_{p_2^{32}} \leqslant 2 \end{cases} \tag{2.7}$$

De plus, le marquage initial doit vérifier la condition de vivacité qui s'exprime par

les inéquations suivantes :

$$\text{Contraintes de vivacité} \begin{cases} m_{p_2^{21}} + m_{p_2^{12}} \geqslant 1 \\ m_{p_2^{32}} + m_{p_2^{23}} \geqslant 1 \\ m_{p_{52}^{3}} + m_{p_{25}^{3}} \geqslant 1 \\ m_{p_2^{31}} + m_{p_2^{13}} \geqslant 1 \\ m_{p_{52}^{2}} + m_{p_{25}^{2}} \geqslant 1 \\ m_{p_{52}^{1}} + m_{p_{25}^{1}} \geqslant 1 \\ m_{p_2^{12}} + m_{p_2^{23}} + m_{p_2^{31}} \geqslant 1 \\ m_{p_2^{12}} + m_{p_{25}^{2}} + m_{p_{52}^{1}} \geqslant 1 \\ m_{p_2^{13}} + m_{p_{25}^{3}} + m_{p_{52}^{1}} \geqslant 1 \\ m_{p_{25}^{1}} + m_{p_{52}^{2}} + m_{p_2^{21}} \geqslant 1 \\ m_{p_{25}^{1}} + m_{p_{52}^{3}} + m_{p_2^{31}} \geqslant 1 \\ m_{p_2^{13}} + m_{p_2^{32}} + m_{p_2^{21}} \geqslant 1 \\ m_{p_2^{23}} + m_{p_{25}^{3}} + m_{p_{52}^{2}} \geqslant 1 \\ m_{p_2^{25}} + m_{p_{52}^{3}} + m_{p_2^{32}} \geqslant 1 \\ m_{p_2^{12}} + m_{p_2^{23}} + m_{p_{25}^{3}} + m_{p_{52}^{1}} \geqslant 1 \\ m_{p_{25}^{1}} + m_{p_{52}^{3}} + m_{p_2^{32}} + m_{p_2^{21}} \geqslant 1 \end{cases} \tag{2.8}$$

De ce fait, l'ensemble des marquages solutions des systèmes d'inéquations (2.7) et d'inéquations (2.8) caractérise l'ensemble des séquencements admissibles pour la ressource r_1 et le système d'inéquations (2.6) caractérise l'ensemble des séquencements de la ressource r_2. En effet, chaque marquage solution du système d'inéquations (2.7) est un séquencement des ressources et un marquage solution du système d'inéquations (2.6) est vivant. Sachant que dans le cas de ressources simples le problème d'ordonnancement est NP-complet, il est intéressant de considérer la complexité de la résolution d'un tel système.

D'une part dans le pire des cas, c'est-à-dire lorsque l'on partage une ressource entre plusieurs tâches franchies une seule fois, il existe $(n-1)!$ séquencements possibles correspondant à tous les ordres possibles entre les transitions. Ce qui confirme le caractère non-polynômial de l'optimisation. L'avantage de l'approche proposée est que tous les ordres sont conservés et nous le verrons plus tard, qu'il ne sera pas nécessaire de tous les évaluer.

D'autre part, lorsque l'on partage plusieurs exemplaires d'une même ressource plu-

sieurs transitions sont franchies simultanément. Lorsque l'on franchit plusieurs transitions par step, la complexité se trouve fortement réduite selon le nombre des steps résultants. Supposons que l'on veuille un régime cyclique en d steps pour n transitions alors il reste $(d-1)!$ ordre possibles pour les steps considérés. Le parallélisme d'exécution tend donc à réduire sensiblement la complexité si $d << n$.

Évoquons pour terminer le cas pour lequel transitions sont franchies plusieurs fois par cycle (contraintes de gammes). Dans ce cas, les clones des transitions franchies plusieurs fois sont déjà séquencés. Par conséquent, la complexité de la résolution est également nettement inférieure à celle qui prendrait en considération la totalité des franchissements (nombre de transitions après "clonage").

L'intérêt d'une telle approche résulte donc de l'intégration d'un jeu de contraintes sur le marquage initial. Ainsi, tous les séquencements possibles sont réduits à un système d'équations et d'inéquations. Il est évident que l'ordonnancement optimal est conservé puisque tous les séquencements possibles sont abstrait par le système d'inéquations. Cependant en optimisant le parallélisme, c'est-à-dire en limitant la valeur de d devant n, la complexité est réduite très fortement même si cette dernière reste d'expression non polynomiale.

2.5 Résolution des partages de ressources avec pondération des arcs

À ce stade de notre étude, nous connaissons une méthode de transformation d'un partage de ressource dont chaque tâche en conflit ne requiert qu'un seul exemplaire de la ressource lors de son exécution. Notre but est d'étendre la résolution au cas des partages de ressources multiples entre différentes tâches où les arcs sont pondérés et les tirs des transitions non nécessairement unitaires. Dans ce sens, nous exhiberons un exemple illustratif, dérivé de l'exemple précédent, représenté par la figure 2.11. Cet exemple sera décrit au cours du premier paragraphe. Puis nous décrirons la méthode visant à l'obtention d'un graphe d'événements associé à un système d'inéquations dont l'ensemble des solutions contient les séquencements possibles des transitions en conflit.

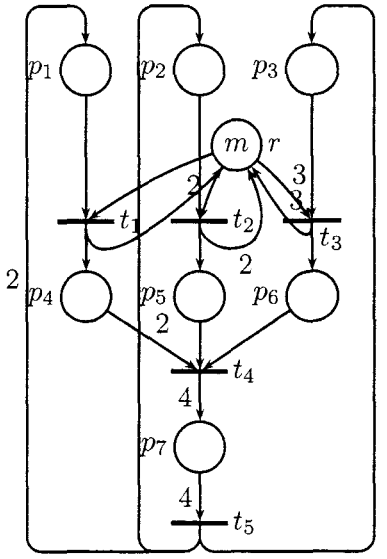


FIG. 2.11 – *Exemple d'illustration*

2.5.1 Exemple illustratif

À titre d'exemple d'illustration, considérons le cas de la gamme de production décrite figure 2.11. La chaîne de production permet de fabriquer un produit réalisé par un assemblage de trois types composants : deux de type A_1, un de chaque type A_2 et A_3.

Le premier composant A_1 est fabriqué par l'usinage de la pièce située en p_1 en utilisant un exemplaire de la ressource r, ce qui est symbolisé par le tir de la transition t_1. La fabrication du deuxième composant est effectuée sur la transition t_2 en utilisant deux exemplaires de la ressource r. Le dernier composant est usiné à l'aide de trois ressources de type r et symbolisé par le franchissement de la transition t_3. Un assemblage de deux pièces A_1, une pièce A_2 et une pièce A_3 est alors effectué au niveau de la transition t_4. Enfin, le résultat de l'assemblage est contrôlé; ce qui est symbolisé par le franchissement de la transition t_5 et l'on obtient le produit final. Le modèle est rebouclé virtuellement pour prendre en compte la production cyclique du système.

Aussi pour notre exemple les hypothèses pour un cycle de production sont les suivantes :

- Deux franchissements de la transition t_1
- Un franchissement de la transition t_2
- Un franchissement de la transition t_3

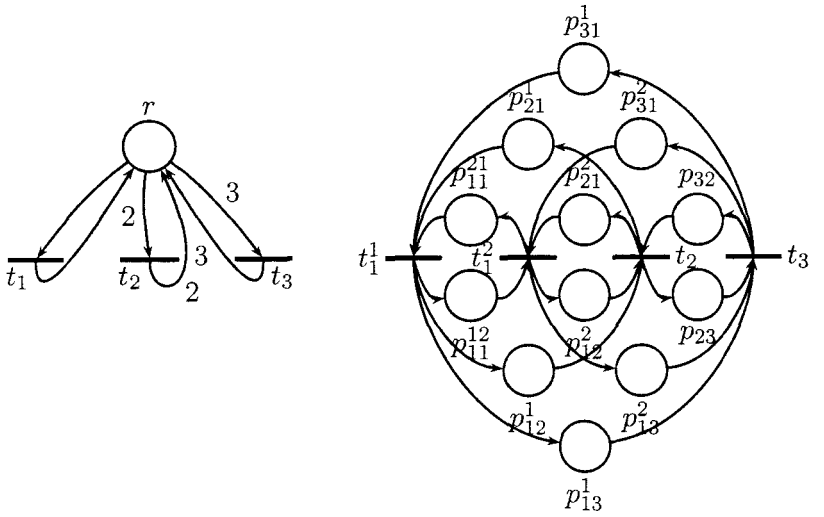


FIG. 2.12 – *Construction du TGES*

2.5.2 Résolution du partage de la ressource r

Construction du TGES

Nous utilisons la méthode de transformation exposé dans le paragraphe 2.3.1. Le TGES résultant est représenté par la figure 2.12.

On constate la présence de deux clone de la transition t_1, représentatif des deux franchissements de celle-ci durant un cycle. Ainsi que des places intermédiaires entre chaque paire de transitions dont le but est de caractériser les contraintes de précédence entre deux transitions.

Le TGES construit permet d'aborder l'abstraction des séquencements possibles en un système d'inéquations. Cette méthode est décrite par la suite.

Résolution des partages de ressources

Nous ciblons cette étude sur la transformation d'un conflit structurel engendré par un partage de ressources. Dans la partie précédente nous avons vu que la transformation d'un conflit structurel permet l'obtention d'un graphe d'événements. L'objet de ce paragraphe est d'abstraire l'ensemble des séquencements possibles des ressources. Chaque séquencement peut être caractérisé par un marquage initial du TGES obtenu par la méthode de construction décrite précédemment. Ce paragraphe est donc consacré à la recherche d'un système d'inéquations dont l'ensemble des solutions correspond de façon équivalente à celles

obtenues pour la recherche d'un ensemble de marquages initiaux. Chaque marquage initial engendre alors l'un quelconque des séquencements possibles des ressources.

Ce paragraphe est divisé en trois parties. Dans un premier temps, nous exposerons une méthode visant l'obtention d'un système d'inéquations dont un marquage initial solution correspond à un séquencement donné des ressources. Dans un second temps, étendons le système d'inéquations précédemment obtenu dans le but de garantir la vivacité d'un marquage initial. Puis pour conclure, nous prouverons que tous les séquencements possibles des ressources peuvent être obtenus à l'aide d'un marquage initial du TGES solution du système global d'inéquations ainsi généré.

Obtention d'un système d'inéquations

Du fait de la transformation évoquée au paragraphe 2, nous étudions le cas d'un partage de m exemplaires identiques d'une ressource entre d tâches représentées par un ensemble de transitions $T' = \{t_1, \ldots, t_d\}$ franchie une seule fois par cycle. Le but de ce paragraphe est d'exhiber un ensemble de relations que doit nécessairement vérifier un marquage initial pour engendrer un séquencement des ressources. Plus généralement, nous recherchons l'obtention d'un système d'inéquations tel que chaque marquage initial corresponde à un séquencement des ressources, solution du système. En faisant l'hypothèse d'un parallélisme d'exécution, nous déterminons en fait une séquence de steps réalisables du point de vue des hypothèses spécifiées sur le fonctionnement du TGES.

Dans ce sens, nous allons exhiber une méthode visant l'obtention d'un marquage initial engendrant l'un quelconque des séquencements spécifiés. Le marquage initial recherché doit donc intégrer le fait que m ressources sont partagées. Par suite, plusieurs transitions peuvent être franchies simultanément, leur nombre dépend du nombre de ressources nécessaires à l'exécution de ce step. Par conséquent, nous allons définir des ensembles de transitions dont la définition repose sur le nombre de ressources consommées dans le cas d'un franchissement en parallèle de toutes les transitions.

Définition 10 *On notera T_i^*, un sous-ensemble $\{t_{i_1}, \ldots, t_{i_{\alpha+1}}\}$ de transitions de T' tel que :*

$$(\forall j) : 1 \leqslant j \leqslant \alpha + 1, \sum_{k=1;k\neq j}^{\alpha+1} W(r,t_{i_k}) \leqslant m \text{ et } \sum_{k=1}^{\alpha+1} W(r,t_{i_k}) > m \tag{2.9}$$

Où $W(r,t_{i_k})$ représente la pondération de l'arc joignant la place r à la transition t_{i_k}.

La définition précédente permet de considérer des sous-ensembles T_i^* de transitions de l'ensemble des transitions du TGES. Les inéquations (2.9) permettent d'affirmer que α transitions de T_i^* peuvent être franchies mais que le franchissement simultané des $\alpha + 1$ transitions de T_i^* est impossible. L'idée est donc ici d'interdire le franchissement simultané de toutes les transitions contenues dans un ensemble de type T_i^*. ce que nous énonçons par le lemme suivant :

Lemme 2 *Soit M un marquage initial du TGES, s'il vérifie les conditions suivantes :*

$$\forall T_i^* = \{t_{i_1}, \ldots, t_{i_{\alpha+1}}\}, \left(\sum_{j=1}^{\alpha} m_{p_{i_{j+1} i_j}} \right) + m_{p_{i_1 i_{\alpha+1}}} \leqslant \alpha \tag{2.10}$$

Alors le franchissement simultané de toutes les transitions de l'ensemble T_i^* dans un même step est impossible.

Preuve. Soit un sous-ensemble de transitions de type $T_i^* = \{t_{i_1}, \ldots, t_{i_{\alpha+1}}\}$, ces transitions vérifient par hypothèses les inéquations (2.9). Donc le franchissement simultané de toutes les transitions de T_i^* ne peut se produire.

Dans ce sens, on parcourt les ensembles de type T_i^*, puis on forme l'ensemble des circuits de place sur la base des transitions contenues dans l'ensemble T_i^*. Considérons un circuit de ce type, le franchissement en parallèle de toutes les transitions qu'il contient exigerait un nombre de ressources supérieur à celles en présence (inéquation (2.9)). Par conséquent, il convient de ne pas valider simultanément toutes les transitions de l'ensemble T_i^*. Par contre tout sous ensemble strict de transitions peut être validé lors d'un step (inéquation (2.9)). De ce fait, une place au moins de ce circuit ne doit pas être pas marquée; ce qui implique que l'ensemble des transitions considérées ne peut être franchies lors du premier step.

Les places contenues dans les circuits précédemment décris forment des invariants de places. En effet, toute transition possède une place en amont et une place en aval. Par conséquent le franchissement des transitions ne modifie pas la somme des marquages des places. Donc l'ensemble des transitions considérées ne peut être franchies lors du fonctionnement du réseau de Petri. ■

À titre d'exemple, considérons pour le TGES représenté figure 2.11. Un sous-ensemble de transitions répondant aux critères de la définition 10 est donné par $T_1^* = \{t_1^1, t_1^2, t_2\}$. Nous obtenons les deux chemins représentés figure 2.13. D'après le lemme 2, nous en déduisons que dans chaque circuit une des places ne doit pas être marquée pour éviter le franchissement en parallèle des transitions : t_1^1,t_1^2 et t_2.

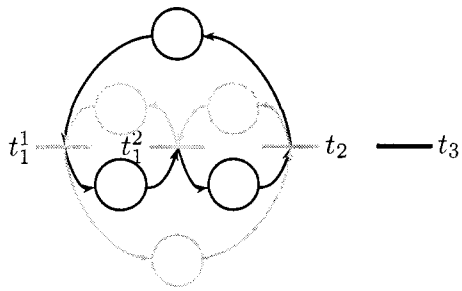


FIG. 2.13 – *Exemple de chemins*

Le lemme 2 précédent exprime une condition suffisante pour qu'un marquage initial du TGES induise un comportement où les steps consomment un nombre de ressources inférieur ou égal au nombre ressources disponibles : m. Ce résultat engendre donc un système d'inéquations du type de l'inéquation (2.10). Ce système d'inéquations contraint le marquage des places du TGES dans le but d'obtenir un marquage initial engendrant un séquencement de la ressource tout en validant les spécifications associées aux franchissements des transitions.

Intégration des conditions de vivacité

Notre objectif est maintenant d'associer au système d'inéquations obtenu à l'aide de l'inéquation générique (2.10) obtenu précédemment un système d'inéquations complémentaires garantissant la vivacité du modèle. Notre raisonnement se base sur la proposition suivante :

Proposition 7 *Si M est un marquage initial du TGES vérifiant les relations (2.10) alors soit le marquage induit un blocage, soit il existe un séquencement cyclique σ tel que $M[\sigma > M$.*

Preuve.

Soit M un marquage vérifiant la relation (2.10). Nous supposerons que le marquage M n'induit pas de blocage, c'est-à-dire que nous faisons l'hypothèse de l'existence d'une transition vivante, que l'on nommera t_i. Nous supposerons de plus l'existence séquence de steps telle que :

$$S_1 S_2 \dots S_k \text{ telle que } M[S_1 S_2 \dots S_k\rangle$$

Le franchissement de la transition t_i de ce circuit implique la consommation des jetons des places en amont de cette transition. Or :

$$\forall j \neq i, \{p_{ji}\} = {}^\bullet t_i \cap t_j{}^\bullet \tag{2.11}$$

Le franchissement de t_i redevient possible uniquement lorsque les places amont sont de nouveau marquées. L'équation (2.11) permet de dire que le tir de t_i n'est possible qu'après le franchissement de toutes les transitions du TGES. Ainsi l'indice k peut être choisi de telle façon que la séquence de steps $S = S_1 S_2 \dots S_k$ soit la plus grande possible et ne contienne qu'une occurrence et une seule de chaque transition. La transition t_i est supposée vivante et celle-ci ne peut être franchie que si les autres ont été tirées. Donc deux cas se présentent, soit S contient toutes les transitions et la preuve est terminée. Soit S ne contient pas toutes les transitions, nous en déduisons l'existence d'une transition non franchissable, ce qui implique la non franchissabilité de la transition t_i à partir du marquage obtenu après le franchissement de la séquence S. Or compte tenu de la maximalité de la séquence S et de la vivacité de la transition t_i, nous concluons que ce cas est impossible. Par suite, S est une séquence de steps contenant toutes les transitions et la proposition 7 impose que le marquage initial vérifie une inégalité qui interdit le franchissement simultané de plus de α transitions. En conclusion, la séquence de steps $S = S_1 S_2 \dots S_k$ est un séquencement cyclique. ■

L'idée principale utilisée dans cette partie consiste à filtrer les solutions qui correspondent à un blocage du TGES. La garantie de vivacité des solutions revient donc à éviter les blocages possibles afin que seul les séquencements cycliques possibles des ressources subsistent. En référence à des travaux de travaux antérieurs [CHEP71], rappelons la condition nécessaire et suffisante de vivacité suivante :

Théorème 3 *Pour qu'un graphe d'événements simple soit vivant il faut et il suffit qu'il existe au moins un jetons dans chaque circuit élémentaire du réseau.*

Il est alors nécessaire de décrire les circuits élémentaires du TGES pour garantir la vivacité d'un marquage à l'aide du théorème 3. C'est-à-dire que la somme des marquages des places contenues dans chaque circuit élémentaire doit être supérieure ou égale à un. De plus, un circuit élémentaire ne peut contenir plus de transitions que le nombre total de transitions du TGES, sous peine de répétition. Donc les inéquations ne peuvent contenir plus d'inconnues que d'éléments dans l'ensemble T', en effet une inéquation du type précédent contient les marquages de places intermédiaires entre deux transitions d'un chemin contenant

un sous ensemble de T' sans répétition. Les contraintes exposées permettent d'énoncer le théorème suivant :

Théorème 4 *Le système d'inéquations, augmenté du système induisant la vivacité, capture uniquement tous les séquencements cycliques possibles.*

Si l'on considère un réseau de Petri pondéré pour lequel les contraintes de franchissement des transitions ont été exprimés et dont les indéterminismes ne proviennent que des partages de ressources. Le théorème 4 permet d'affirmer qu'il est possible de le transformer en un graphe d'événements pondéré tout en préservant l'ensemble des comportements possibles du réseau de Petri.

Application

Appliquons la théorie exposée précédemment à l'exemple illustratif de la figure 2.11 introduit au début de cet article. Nous envisageons la cas de trois ressources ($m = 3$).

La première étape consiste en la duplication de la transition t_1 en deux transitions t_1^1 et t_1^2 dont la fréquence est d'un franchissement par cycle. Le réseau de Petri est transformé en supprimant le partage de ressources et en effectuant la construction du TGES, représenté en noir sur la figure 2.14. Puis nous résolvons le partage de ressources à l'aide de la méthode exposée dans le paragraphe 2.5.2.

Nous devons ajouter les systèmes d'inéquations liés au partages des ressources et aux conditions de vivacité dans le TGES. La première étape est de déterminer tous les sous-ensembles de de $T' = \{t_1^1, t_1^2, t_2, t_3\}$ respectant les conditions du lemme 10, nous obtenons :

$$\begin{cases} \{t_2,t_3\} \\ \{t_1^1,t_3\} \\ \{t_1^2,t_3\} \\ \{t_1^1,t_1^2,t_2\} \end{cases} \quad \begin{cases} m_{p_{32}} + m_{p_{23}} \leqslant 1 \\ m_{p_{31}^1} + m_{p_{13}^1} \leqslant 1 \\ m_{p_{31}^2} + m_{p_{13}^2} \leqslant 1 \\ m_{p_{11}^{21}} + m_{p_{12}^1} + m_{p_{21}^2} \leqslant 2 \\ m_{p_{11}^{12}} + m_{p_{12}^2} + m_{p_{21}^1} \leqslant 2 \end{cases} \tag{2.12}$$

Le théorème 2 permet, à partir des sous-ensembles de T' donnés dans le système (2.12), de donner un système garantissant la validité des solutions obtenues, c'est-à-dire que les marquages initiaux obtenus engendrent bien un séquencement des ressources. Ce qui se traduit par le système d'inéquations supplémentaire suivant garantissant la vivacité du séquence-

ment :

$$\left\{\begin{array}{l} m_{p_{11}^{21}} + m_{p_{11}^{12}} \geqslant 1 \\ m_{p_{21}^{2}} + m_{p_{12}^{2}} \geqslant 1 \\ m_{p_{32}} + m_{p_{23}} \geqslant 1 \end{array}\right. \qquad \left\{\begin{array}{l} m_{p_{21}^{1}} + m_{p_{12}^{1}} \geqslant 1 \\ m_{p_{31}^{2}} + m_{p_{13}^{2}} \geqslant 1 \\ m_{p_{31}^{1}} + m_{p_{13}^{1}} \geqslant 1 \end{array}\right.$$

$$\left\{\begin{array}{l} m_{p_{11}^{21}} + m_{p_{12}^{1}} + m_{p_{21}^{2}} \geqslant 1 \\ m_{p_{11}^{12}} + m_{p_{12}^{2}} + m_{p_{21}^{1}} \geqslant 1 \\ m_{p_{11}^{21}} + m_{p_{13}^{1}} + m_{p_{31}^{2}} \geqslant 1 \\ m_{p_{11}^{12}} + m_{p_{13}^{2}} + m_{p_{31}^{1}} \geqslant 1 \\ m_{p_{21}^{1}} + m_{p_{13}^{1}} + m_{p_{32}} \geqslant 1 \\ m_{p_{12}^{1}} + m_{p_{23}} + m_{p_{31}^{1}} \geqslant 1 \\ m_{p_{21}^{2}} + m_{p_{13}^{2}} + m_{p_{32}} \geqslant 1 \\ m_{p_{12}^{2}} + m_{p_{23}} + m_{p_{31}^{2}} \geqslant 1 \end{array}\right. \qquad \left\{\begin{array}{l} m_{p_{11}^{21}} + m_{p_{21}^{2}} + m_{p_{32}} + m_{p_{13}^{1}} \geqslant 1 \\ m_{p_{11}^{12}} + m_{p_{12}^{2}} + m_{p_{23}} + m_{p_{31}^{1}} \geqslant 1 \\ m_{p_{12}^{1}} + m_{p_{21}^{2}} + m_{p_{13}^{2}} + m_{p_{31}^{1}} \geqslant 1 \\ m_{p_{21}^{1}} + m_{p_{12}^{2}} + m_{p_{31}^{2}} + m_{p_{31}^{1}} \geqslant 1 \\ m_{p_{13}^{1}} + m_{p_{31}^{2}} + m_{p_{12}^{2}} + m_{p_{21}^{1}} \geqslant 1 \\ m_{p_{31}^{1}} + m_{p_{13}^{2}} + m_{p_{21}^{2}} + m_{p_{12}^{1}} \geqslant 1 \end{array}\right. \tag{2.13}$$

Cette transformation conduit à l'obtention du graphe d'événements pondéré représenté par la figure 2.14. Le comportement logique de ce réseau de Petri peut être étudié à l'aide du dioïde (min ,+) [TBG02a]. Nous obtenons alors le système d'équations (2.14) (min ,+)-linéaires suivant :

$$\left\{\begin{array}{l} n_1^1(k+1) = \min\left(n_1^2(k) + m_{p_{11}^{21}}, n_2(k) + m_{p_{21}^{1}}, n_3(k) + m_{p_{31}^{1}}, n_5(k) + m_{p_1^1}\right) \\ n_1^2(k+1) = \min\left(n_1^1(k) + m_{p_{11}^{12}}, n_2(k) + m_{p_{21}^{2}}, n_3(k) + m_{p_{31}^{2}}, n_5(k) + m_{p_1^2}\right) \\ n_2(k+1) = \min\left(n_1^1(k) + m_{p_{12}^{1}}, n_1^2(k) + m_{p_{12}^{2}}, n_3(k) + m_{p_{32}}, n_5(k) + m_{p_2}\right) \\ n_3(k+1) = \min\left(n_1^1(k) + m_{p_{13}^{1}}, n_1^2(k) + m_{p_{13}^{2}}, n_2(k) + m_{p_{23}}, n_5(k) + m_{p_3}\right) \\ n_4(k+1) = \min\left(n_1^1(k) + m_{p_4^1}, n_1^2(k) + m_{p_4^2}, n_2(k) + m_{p_5}, n_3(k) + m_{p_6}\right) \\ n_5(k+1) = \left\lfloor \dfrac{4 \times n_4(k) + m_{p_7}}{4} \right\rfloor = n_4(k) + \left\lfloor \dfrac{m_{p_7}}{4} \right\rfloor \end{array}\right. \tag{2.14}$$

Par conséquent, le réseau de Petri décrit par la figure 2.11, avec $m = 3$ exemplaires de la ressource r, est transformé en un graphe d'événements (figure 2.14) dont le marquage initial est contraint par les systèmes d'inéquations (2.12) et (2.13). De plus dans le cas particulier de notre exemple, le comportement logique du graphe d'événements pondéré obtenu peut être décrit à l'aide d'un système d'équations (min ,+)-linéaires (système (2.14)). Nous verrons au sein de la partie suivante qu'il est possible pour une classe de graphe d'événements pondéré de représenté le comportement logique à l'aide d'un système d'équations (min ,+)-linéaires.

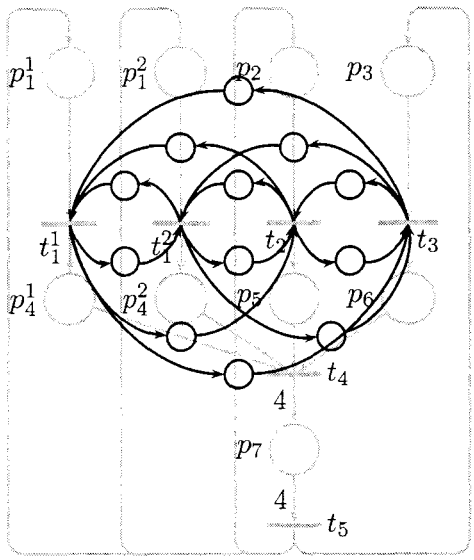


FIG. 2.14 – *Le graphe d'événements pondéré résultant*

2.6 Conclusions

Dans ce chapitre, la présentation des résultats s'est effectuée de manière progressive dans le but d'exposer clairement les difficultés rencontrées.

Dans un premier temps, le partage d'un exemplaire d'une ressource partagée entre des transitions franchies une fois par cycle. La résolution du partage de la ressource consiste en une transformation structurelle du réseau, dans le but d'obtenir un réseau déterministe. Les choix possibles sont abordés lors de la prise en compte des contraintes de précédences parmi les transitions séquences franchissements possibles, ce qui se traduit par un système d'équations et inéquations dont les solutions sont un marquage initial d'un sous-ensemble de places du graphe d'événements obtenu induisant un comportement correct.

Dans un second temps, nous prenons en compte les exemplaires multiples d'une ressource partagée entre plusieurs transitions en conflit, celles-ci sont franchies une seule fois par cycle. La méthode de résolution est sensiblement la même que précédemment, la prise en compte de plusieurs exemplaires de la ressource implique la prise en compte de contraintes plus complexe. Le contraintes sur le marquage du sous réseau de Petri ainsi construit se traduisent par un système d'inéquations. Aussi l'ensemble des séquencements possibles des transitions en conflits sont solutions d'un système d'inéquations

Puis, la prise en compte des franchissements multiples possibles des transitions en

conflit impose une modification de la technique de construction. En effet dans le but de retrouver la problématique précédemment étudiée, la construction du graphe d'événements est enrichie d'une duplication des transitions. Ainsi le nombre de clones d'une transition en conflit est égal au nombre de franchissement de celle-ci par cycle.

Enfin, l'extension finale de cette problématique concerne la pondération sur les arcs. Nous obtenons une méthode générique face à ce type de problème. Cette prise en compte abouti à la construction d'un graphe d'événements associé à un système d'inéquations caractérisant les séquencements possible des transitions initialemaent en conflit d'accès aux ressources.

Nous avons présenté dans ce chapitre une méthode de transformation structurelle d'un réseau de Petri contenant des partages de ressources, donc des conflits structurels. Notre but était de générer un graphe d'événements pondérés, le caractère déterministe du réseau permet alors une approche algébrique de l'étude du comportement logique du réseau de Petri obtenu. Le principe de notre démarche à été de prendre en compte le nombre de franchissements de chaque transition au cours d'un cycle et/ou le nombre d'exemplaire de la ressource en partage. Ces données permettent d'établir un système d'équations, qui induit un paramétrage par les marquages initiaux possibles du graphe d'événements pondérés. Par conséquent, notre réseau de Petri avec ses données initiales est transformé en un graphe d'événements pondéré avec un jeu d'équations régissant le marquage initial des nouvelles places crées.

L'abstraction des données initiales permet de considérer les séquencements potentiels comme un ensemble de solutions d'un système d'équations et d'inéquations. L'objectif est d'abstraire complètement le comportement logique du réseau de Petri avant toute évaluation des solutions. Cette approche permet de repousser l'évaluation des solutions après la prise en compte de tous les hypothèses faite sur le réseau de Petri et ainsi d'obtenir un système d'équations suffisamment contraint pour diminuer fortement la complexité de résolution.

Deuxième partie

Traduction algébrique du comportement logique d'un réseau de Petri

Introduction

Le type d'application que l'on considère ici concerne les systèmes de production de type manufacturière. Ce type de système doit prendre en compte les activités d'assemblages et l'utilisation de ressources partagées. Ces caractéristiques induisent, lors de la modélisation par réseau de Petri, des indéterminismes et des pondérations sur les arcs. De plus, ce type de système intègre des hypothèses d'exploitation; en effet, la demande de pièces est déterminée à l'avance ce qui permet d'en déduire le nombre de franchissement de chaque transition durant un cycle dans l'hypothèse d'un fonctionnement cyclique.

Au sein de la partie précédente, une transformation d'un partage de ressources simple ou multiple à permis de rendre déterministe un réseau de Petri comportant des conflits d'accès à des ressources partagées simples ou multiples, par intégration de contraintes de fonctionnement. Le modèle obtenu est un graphe d'événements pondéré. Le caractère déterministe du réseau permet d'aborder une étude du comportement logique. Aussi dans ce chapitre nous nous intéressons au comportement logique des GEP, nous proposons une formalisation du comportement logique d'un graphe d'événements pondéré en traduisant le comportement du graphe en un système d'équations dont les variables sont les marquages initiaux des places. En effet, la prise en compte d'un marquage initial inconnu permettra une recherche de marquage initial intégrant des propriétés spécifiques telles que la cyclicité. Pour introduire cette étude, nous utilisons un formalisme algébrique dédié inspiré de l'algèbre des dioïdes. Notre approche consistera à abstraire le comportement logique d'un graphe d'événements pondéré. Puis lorsque c'est possible, de valider le formalisme développé pour les graphes d'événements simples. L'intérêt est alors de pouvoir transformer le système d'équations en un système équivalent linéaire [Gau92b]. Dans ce sens, une propriété intéressante qui découle de la causalité concerne la linéarité du modèle. Notre objectif est de voir dans quelles mesures il est possible de proposer une étude formelle des GEP.

Pour illustrer notre théorie, nous utiliserons un exemple représenté par la figure 3.1, cet exemple sera décrit ultérieurement. Les hypothèses de production d'un système de production manufacturière induisent la connaissance du nombre de franchissements de chaque transition. Ajoutons que l'assemblage des pièces est effectué en quantité non nécessairement identiques, traduit par des pondérations sur les arcs du réseau de Petri. Nous allons donc envisager d'étudier le réseau de Petri à l'aide de structures algébriques dédiées.

Le chapitre est organisé en deux parties. Tout d'abord, nous introduisons la nouvelle structure algébrique proposée. Puis nous verrons les principes de bases de notre approche concernant les graphes d'événements pondérés et en particulier comment cette technique permet l'étude du comportement logique d'un tel graphe.

Chapitre 3

Linéarisation

Dans ce chapitre, nous chercherons à exprimer algébriquement le comportement d'un graphe d'événements pondéré consistant. Dans le contexte des applications concernant l'ordonnancement, l'étude d'un tel système est facilitée par une stratégie franchissement au plus tôt des transitions [Chr86]. En effet la stratégie déclenchement au plus tôt des transitions d'un GEP est optimale du point de vu de la performance de tels systèmes. Étant donné qu'un GEP ne comporte pas de conflits, l'évolution du système peut être représentée sous forme d'une séquence et non sous forme d'un arbre décisionnel: comme l'indique la figure 3.2. L'évolution d'un réseau de Petri est décrite par une séquence de marquages (représentés par $M(0)$, $M(1)$, ..., $M(k)$, ...) obtenue après franchissement d'une séquence de steps (représentés par $S(1)$, $S(2)$,..., $S(k)$,...). Le step $S(k)$ est donc le seul step comportant toutes les transitions (éventuellement en plusieurs exemplaires) franchissables à partir du marquage $M(k-1)$.

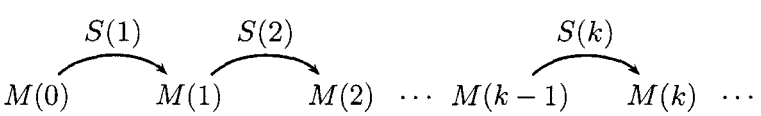


FIG. 3.2 – *Illustration du comportement*

Ce chapitre est organisé en quatre parties. La première partie consiste en un état de l'art, ce paragraphe permet de situer cette étude en regard des travaux existant et de mettre en évidence les difficultés liées à nos objectifs. Puis dans une deuxième partie nous décrirons

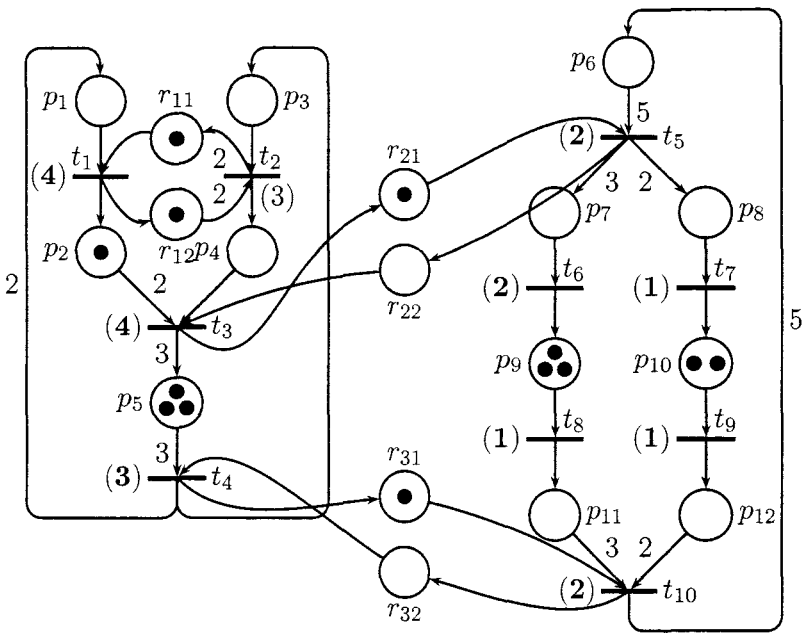


FIG. 3.1 – *Exemple de graphe d'événements pondéré*

l'exemple, représenté par la figure 3.1, introduit précédemment, puis nous introduirons une nouvelle structure algébrique. Nous verrons, en troisième partie comment le comportement logique d'un réseau de Petri peut être abstrait sous la forme d'un système d'équations non-linéaires. Enfin dans la dernière partie, nous verrons que pour certains graphes d'événements pondérés, il est possible de linéariser les systèmes d'équations obtenus.

3.1 État de l'art

Ce paragraphe contient des références de travaux existants dont le sujet est l'étude des GEP, afin d'une part, de situer la démarche et d'autre part, de mettre en évidence les principales difficultés de cette approche.

3.1.1 Une approche par fluidification

Une problématique consiste en l'étude du comportement temporel des réseau de Petri. Dans ce cas, on prend en considération les dates de franchissement des transitions. Cette thématique intègre de nombreux travaux [CDQV83, CMQV84, CMQV85, CDQV85, CMQV86, CMQV89, Gau94, Gau95b, LBH99]. Plus particulièrement dans une article [CGQ95a,

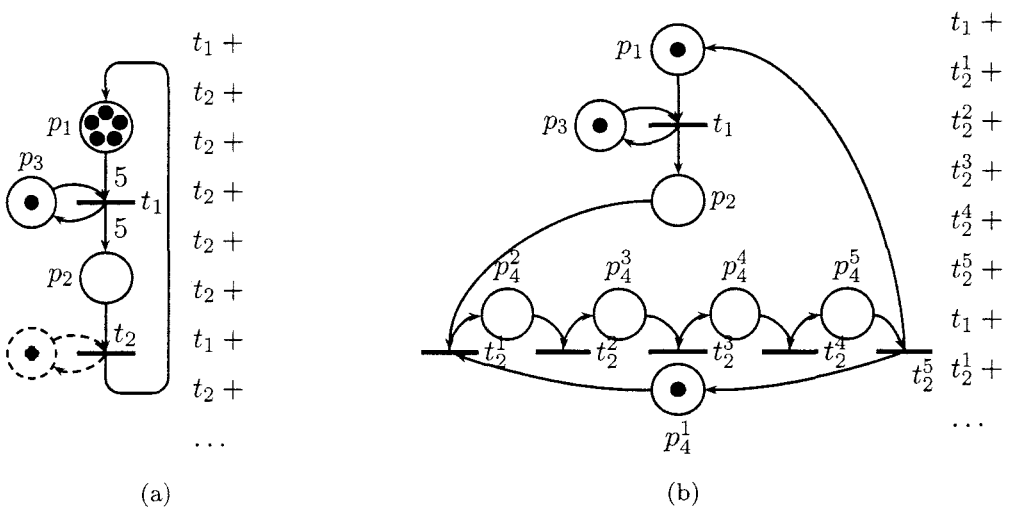


FIG. 3.3 – *Transformation d'un GEP en un GES*

CGQ98, RTS99], qui prend en compte le problème des pondération sur les arcs en transformant le modèle dans le contexte des réseaux de Petri continus. Les réseaux de Petri continus possèdent la particularité suivante: une transition est franchie lorsque tous les places en amont sont marquées. Ce qui implique que l'on autorise les franchissements rationnels des transitions.

Cette technique consiste à modifier la règle de franchissement des transitions pour étudier les GEP à l'aide d'une technique directement inspirée de l'étude des GES. Aussi une étude des performances du réseau est permise mais sans intégrer le caractère événementiel du franchissement des transitions, en effet celles-ci sont franchie de manière rationnelle.

3.1.2 La méthode d'Alix Munier

Une autre technique consiste à modifier la structure du GEP dans le but de supprimer les pondérations des arcs. L'approche proposée par [Mun93] consiste à transformer un GEP en un GES en ajoutant des transitions. Toutefois, il existe des différences notables avec l'approche que nous utiliserons, nous les exposerons sur un exemple.

Remarquons tout d'abord que cette approche ne permet pas le franchissement multiples d'une même transition : en d'autres termes, il existe une place implicite contenant un jeton et bouclée sur chaque transition (figure 3.3(a)). Si l'on considère les deux réseaux de Petri munis de deux marquages initiaux différents représentés par les figures 3.3(a) et 3.4(a),

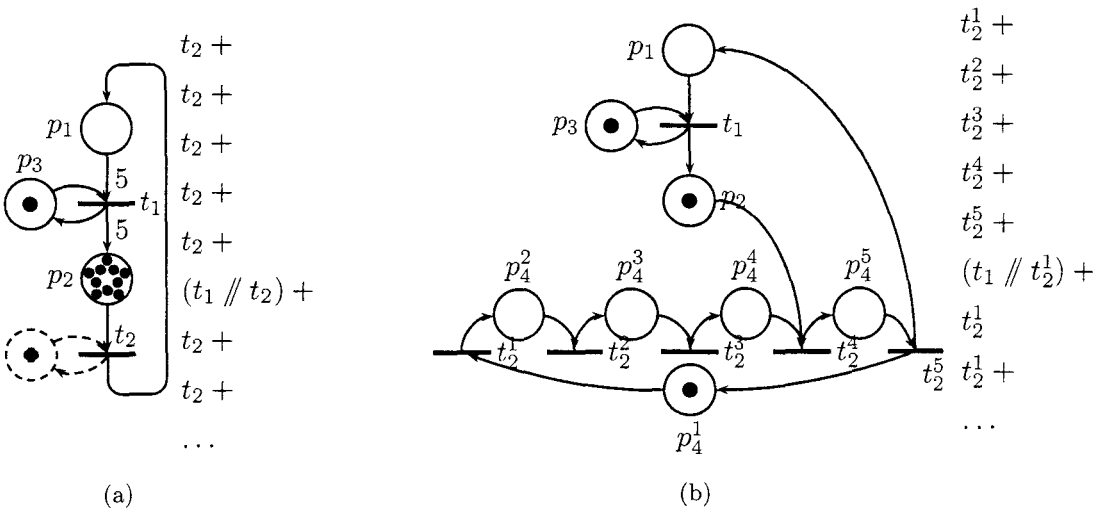


FIG. 3.4 – *Transformation d'un GEP en un GES*

la méthode d'A. Munier permet de générer les deux GES représentés par les figures 3.3(b) et 3.4(b). Ainsi, cette approche fournit un GES dont le marquage initial et **la structure** dépendent du marquage initial du GEP. Enfin une dernière caractéristique réside dans la manière de déduire le comportement du GEP à partir de celui du GES. Chaque transition du GEP à un ou plusieurs représentant dans le GES : chaque tir d'une transition du GEP est représenté par le tir d'un de ses représentants.

Nous verrons dans la suite de ce chapitre la transformation que nous proposons pour obtenir un GES conduit à une structure de modèle unique, paramétrée par son marquage initial et autorisant le déclenchement simultané des transitions.

3.1.3 Une transformation structurelle

Dans une autre étude concernant les graphes d'événements pondéré, effectuée par E. Teruel et *al.* [Jen92], une technique de transformation d'un graphe d'événements pondéré est élaborée, elle permet l'obtention d'un réseau de Petri simple. Ainsi cette technique de transformation permet l'obtention d'un réseau non pondéré mais le caractère déterministe n'est pas conservé. La transformation d'un graphe d'événements pondéré est illustré par la figure 3.5.

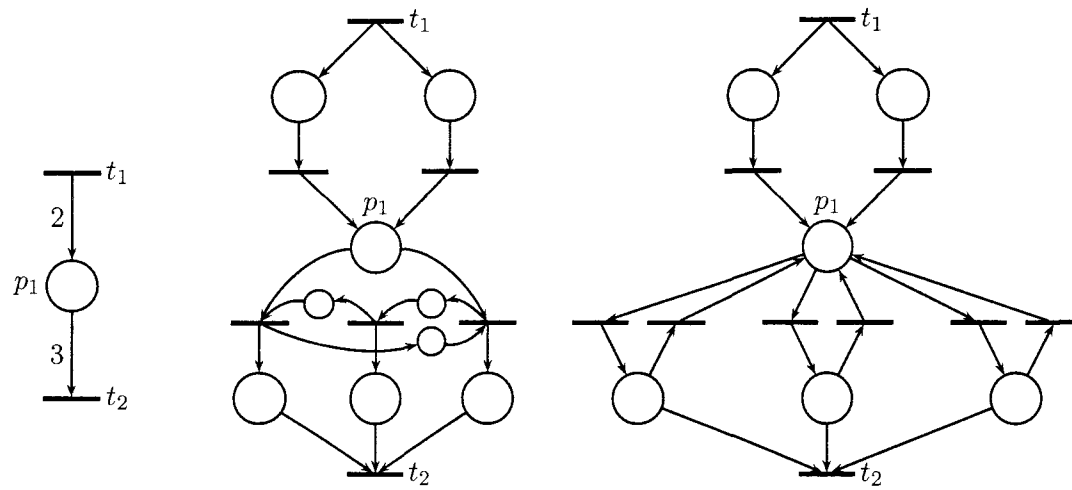


FIG. 3.5 – *Transformation d'un GEP en RdP non pondéré*

Deux transformations sont proposées d'un même graphe d'événements pondéré, On décompose les arcs pondérés joignant deux noeuds en :

- arc joignant une transition à une place : On introduit des transitions supplémentaires leur nombre correspondant au poids de l'arc d'origine, une place est crée en amont de chacune des transitions crées précédemment et chacune de ces places est connectée à la transition d'entrée.
- arc joignant une place à une transition : on relie la place aux transitions créés (leur nombre est égal au poids de l'arc). puis l'on créé une place aval pour chacune des transitions, reliées à leur tour à la transition d'origine.

Cette transformation illustre que le GEP peut être transformé en un réseau de Petri sans pondération. Par contre, le réseau de Petri obtenu ne possède plus le déterminisme initial.

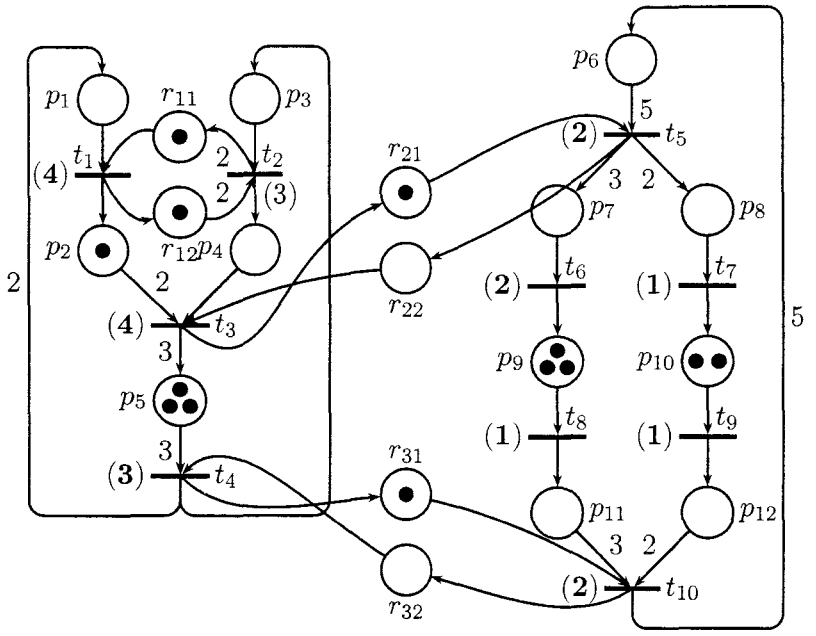


FIG. 3.6 – *Graphes d'événements pondéré*

3.2 Introduction

3.2.1 Description de l'exemple

Ce paragraphe est destiné à la description de l'exemple d'illustration 3.6 utilisé au cours de ce chapitre. Cet exemple est une variante de l'exemple du chapitre 2, les nouvelles spécificités introduites vont permettre d'illustrer les résultats de ce chapitre. Cet exemple provient du domaine de référence relevant de la production manufacturière, la représentation (figure 3.6) tient compte des techniques exposée dans le premier chapitre pour transformer le réseau de Petri en un GEP. Cet exemple est volontairement simple mais permet d'illustrer de manière pertinente la théorie exposée dans ce chapitre.

Le graphe d'événements pondéré représenté par la figure 3.6 modèlise une chaîne de production et permet toujours de produire deux types de produit A et B, à l'aide des trois types de ressources partagées (représentées par les places r_{11}, r_{12}, r_{21}, r_{22}, r_{31} et r_{32}). La ressource r_1 existe en deux exemplaires, la ressource r_2 existe en un seul exemplaire et la ressource r_3 en un exemplaire. Ceci est représenté par le nombre de jetons dans les places r_{11}, r_{12}, r_{21}, r_{22}, r_{31} et r_{32}.

Le produit A est composé de deux pièces A_1 et d'une pièce A_2. Le premier com-

posant A_1 est fabriqué par l'usinage de la pièce située en p_1, ce qui est symbolisé par le franchissement de la transition t_1 à l'aide d'une ressource de type r_1. La fabrication du deuxième composant est effectué sur la transition t_2 en utilisant les deux ressources de type r_1. La machine de type r_2 effectue alors l'assemblage de deux pièces A_1 et d'une pièce A_2, au niveau de la transition t_3. Enfin, le résultat de l'assemblage est controlé par la machine de type r_3 pour obtenir le produit final (transition t_4).

Le deuxième produit B résulte d'un traitement par lots successivement des machines r_2 et r_3.

Afin de caractériser le comportement cyclique du système, le modèle est rebouclé à l'aide des arcs joignant la transition t_4 aux places p_1 et p_3, et la transition t_{10} à la place p_6.

Nous avons défini une politique de routage pour cet exemple. Elle correspond à la production d'un même nombre de produit A pour un même nombre de produit B. Sachant que le produit A est composé de deux sous-produits A_1 et A_2. Le produit B est composé de cinq pièces traitées en deux lot de deux et trois pièces respectivement. Ces spécificités expliquent les pondérations sur les arcs et les partages de ressources.

Considérons la figure 3.6, celle-ci représente le réseau de Petri obtenu après la résolution des partages de ressources (chapitre 2). Le routage de la ressources r_1 est défini à l'avance, il caractérise les deux franchissements de la transition t_1 alternés avec un franchissements de la transition t_2. Deux conflits d'accès aux ressources, défini par les ressources r_2 et r_3, ont été résolu au chapitre précédent pour obtenir le graphe d'événements pondéré. Les équations résultantes de la technique de résolution des partages de ressources sont les suivantes :

$$\begin{cases} m_{r21} + m_{r22} = 1 \\ m_{r31} + m_{r32} = 1 \end{cases}$$

En utilisant la méthode de linéarisation des ressources nous aboutissons à un graphe d'événements pondéré déterministe mais dont nous allons voir qu'il correspond en général à un modèle algébrique formel non linéaire.

3.2.2 Mise en équation de l'exemple

À ce stade de notre étude, nous considérons un graphe d'événements pondéré avec éventuellement un système d'équations dont les variables sont les marquages initiaux des

places du réseau de Petri. Notre but est d'étudier le comportement logique du graphe d'événements obtenu pour en déduire un marquage initial répondant aux spécifications d'un fonctionnement périodique. Dans ce sens, nous allons abstraire le comportement logique de ce GEP en considérant des compteurs sur les transitions permettant de mémoriser le nombre d'occurrences de celles-ci.

Pour l'exemple d'illustration représenté par la figure 3.6 donné précédemment, nous obtenons le système d'équations suivant :

$$\begin{cases} n_1(k+1) = \min\left(2n_2(k) + m_{r_{11}}, 2n_4(k) + m_{p_1}\right) \\ n_2(k+1) = \min\left(\left\lfloor \frac{n_1(k) + m_{r_{12}}}{2} \right\rfloor, n_4(k) + m_{p_3}\right) \\ n_3(k+1) = \min\left(\left\lfloor \frac{n_1(k) + m_{p_2}}{2} \right\rfloor, n_2(k) + m_{p_4}, n_5(k) + m_{r_{22}}\right) \\ n_4(k+1) = \min\left(\left\lfloor \frac{3n_3(k) + m_{p_5}}{3} \right\rfloor, n_{10}(k) + m_{r_{32}}\right) \\ n_5(k+1) = \min\left(n_3(k) + m_{r_{21}}, \left\lfloor \frac{5n_{10}(k) + m_{p_6}}{5} \right\rfloor\right) \\ n_6(k+1) = 3n_5(k) + m_{p_7} \\ n_7(k+1) = 2n_5(k) + m_{p_8} \\ n_8(k+1) = n_6(k) + m_{p_9} \\ n_9(k+1) = n_7(k) + m_{p_{10}} \\ n_{10}(k+1) = \min\left(n_4(k) + m_{r_{31}}, n_8(k) + m_{p_{11}}, n_9(k) + m_{p_{12}}\right) \end{cases} \tag{3.1}$$

Ce système nous permet, à partir du marquage initial, de déterminer le comportement du GEP par une évaluation successives du vecteur d'occurrences. Nous constatons que ce système est proche du formalisme $(\min, +)$ développé pour l'évaluation de performance des graphes d'événements simple. Pourtant on remarque la présence de division entière dans le système d'équations (3.1), ce qui confère à ce système d'équations le caractère non-linéaire. Les valeurs du type : $\left\lfloor \frac{n_1(k) + m_{r_{12}}}{2} \right\rfloor$ expriment la non-linéarité apparente du réseau de Petri. Ces termes ne permettent pas la mise en équation sous forme d'un système d'équations récurrent à l'aide du dioïde $(\min, +)$. Dans le but d'obtenir une méthode de mise en équation, nous allons définir une nouvelle structure algébrique permettant, de façon systématique, de formaliser une telle mise en équation.

3.2.3 Une nouvelle structure algébrique

Nous venons d'étudier des équations particulières précédentes, elles ne permettent pas l'utilisation des algèbres dédiées telles que les dioïdes. Ceci nous conduit à définir une

nouvelle structure définie par la suite.

$(\overline{\mathbb{N}}, \min, \text{div})$, où $\overline{\mathbb{N}} = \mathbb{N} \cup \{+\infty\}$, min et div sont des lois définies ci après :

Définition 11 min *et* div *sont deux opérateurs binaires définis sur* $\overline{\mathbb{N}}$ *par:*

$$- \ \forall (a,b) \in \overline{\mathbb{N}}^2, \min(a,b) = \begin{cases} a & si\ a \leqslant b \\ b & si\ a > b \end{cases}$$

$$- \ \forall (a,b) \in \overline{\mathbb{N}}^2,\ a \text{ div } b = \begin{cases} \left\lfloor \frac{a}{b} \right\rfloor & si\ a \in \mathbb{N}\ et\ b \in \mathbb{N}^* \\ +\infty & si\ a \in \mathbb{N}\ et\ b = 0 \\ 0 & si\ a \in \mathbb{N}\ et\ b = +\infty \\ +\infty & si\ a = +\infty\ et\ b \in \overline{\mathbb{N}} \end{cases}$$

Propriété 1 *Les deux lois précédentes possèdent les propriétés suivantes:*

- $(\overline{\mathbb{N}}, \min)$ *est un monoïde commutatif dont l'élément neutre est* $+\infty$.
- $(\overline{\mathbb{N}}, \text{div})$ *est un ensemble avec une loi interne, pour laquelle* $+\infty$ *est un élément absorbant à gauche, et* 1 *est un élément neutre à droite.*

Ces propriétés peuvent facilement être déduites des définitions. Malheureusement, la loi div n'est ni associative, ni distributive. Ceci ne confère pas à la structure algébrique $(\overline{\mathbb{N}}, \min, \text{div})$ celle de dioïde [CGQ95a]. Dans ce chapitre, nous verrons cependant en quoi ce formalisme n'est pas en contradiction avec les travaux originaux sur les algèbres linéaires du type $(\min, +)$.

À partir de la structure algébrique précédente, on peut définir le produit matriciel : Soit $(m,n,p) \in \mathbb{N}^3$, $A = (a_{ij})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}$ et $B = (b_{ij})_{\substack{1 \leqslant i \leqslant n \\ 1 \leqslant j \leqslant p}}$ deux matrices à cœfficients entiers, $A \circ B$ est la matrice définie comme suit :

$$A \circ B = (\min_{1 \leqslant k \leqslant n} (b_{kj} \text{ div } a_{ik}))_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant p}}$$

3.3 Principe de base

Notre objectif est de formaliser le comportement logique d'un GEP. Dans ce sens, nous étudions les franchissements des transitions à l'aide du vecteur d'occurrences, nommément $N(k)$. Le vecteur d'occurrences représente la somme des steps franchis, concrétement chaque composante du vecteur est un compteur de franchissement d'une transition qui lui

est associée. Aussi, il nous permet de caractériser le comportement logique comme nous le verrons plus en détail par la suite.

Pour les GEP, nous utiliserons l'algèbre (min , div) introduite dans le paragraphe précédent. à l'aide de cette structure nous pouvons évaluer le step maximal $S(k)$ en fonction du marquage précédent $M(k-1)$ (paragraphe 3.3.1). Puis nous recherchons une équation récurrente sur le vecteur d'occurrences (paragraphe 3.3.2). Cette équation est en général non linéaire du fait de la pondération des arcs, mais elle rejoint le formalisme de l'algèbre (min ,+) pour les GES [LBH99].

Dans toute la suite, $PN = (P,T,W,M(0))$ est un GEP consistant sans transition source où $P = \{p_1,p_2,\ldots,p_m\}$ et $T = \{t_1,t_2,\ldots,t_n\}$. Le step $S(k)$ et le marquage $M(k)$ sont respectivement notés $S(k) = (s_i(k))_{1\leqslant i\leqslant n}$ et $M(k) = (m_i(k))_{1\leqslant i\leqslant m}$.

3.3.1 Équations récurrentes sur les marquages

Soit un GEP consistant, le caractère déterministe de ce graphe induit que la stratégie de tir au plus tôt est optimale [Chr86]. Ainsi, notre objectif est de calculer, connaissant un marquage $M(k)$, le nombre de franchissement maximal de toutes les transitions franchissables à partir du marquage initial du modèle considéré. Ce qui implique l'obtention d'un step maximal $S(k+1)$: plus grand step franchissable à partir du marquage $M(k)$.

Pour les graphes d'événements, la notion de step maximal est définie. Par la proposition suivante :

Proposition 8 *Pour un graphe d'événements, l'ensemble des steps franchissables contient un majorant.*

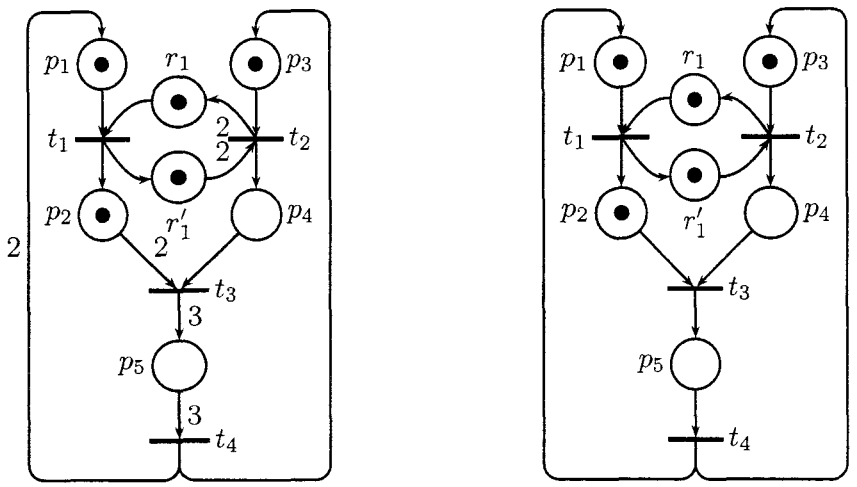
Preuve. La preuve consiste à démontrer que le calcul du nombre de franchissements d'une transition est indépendant du nombre de franchissements des autres transitions.

Considérons un graphe d'événements déterministe; par définition, les places ne sont connectées qu'à une seule transition :

$$|{}^\bullet p| = |p^\bullet| = 1$$

Le franchissement d'une transition t implique la consommation de jeton(s) dans la(les) place(s) en amont de t. Or, le fait que le réseau soit déterminisme implique que deux transitions quelconque du réseau ne possèdent pas de transitions amont communes :

$${}^\bullet t_i \cap {}^\bullet t_j = \varnothing \tag{3.2}$$



(a) Un graphe d'événements pondéré

(b) Un graphe d'événements simple

FIG. 3.7 – *Deux réseaux de Petri*

Par suite, le franchissement d'une transition t_i ne modifie pas le marquage des place en amont d'une autre transition t_j (propriété (3.2)). Les nombres de franchissement sont indépendant entre eux et le step, dont la valeur de chaque composante est définie par le nombre de franchissements maximal de la transition associée, constitue un majorant de l'ensemble des transitions. ■

Grace à la proposition précédente le calcul du nombre maximal de franchissements de chacune des transitions a un sens, chaque nombre de franchissements des transitions étant indépendant. L'ensemble des steps franchissables admet un plus grand élément. Comme nous appliquons la stratégie optimale de franchissement au plus tôt, nous allons étudier le calcul du step maximal par la suite.

Remarque 3 *considérons le sous réseau de notre exemple principal pondéré, représenté par la figure 3.7. Nous constatons que les places possèdent en aval une seule transition, propriété utilisées dans la preuve précédente. Par suite, vu la stratégie de tir au plus tôt, le franchissement du step maximal constitue la stratégie optimale. Aussi, nous obtenons le step t_1 pour l'exemple 3.7(a) et le step t_1 // t_2 pour l'exemple 3.7(b). Cet exemple permet d'appréhender les difficultés que nous rencontrerons par la suite lors du calcul des steps maximaux pour un graphe d'événements contenant des pondérations sur les arcs. En effet dans l'exemple représenté par la figure 3.7(a), malgré le jeton de la place r'_1 la transition t_2*

n'est pas franchissable.

La présence des poids sur les arcs complique le calcul de $S(k)$ et fait intervenir l'opérateur « div » défini au paragraphe précédent. Nous allons voir dans la proposition suivante une méthode pour calculer le step maximal étant donné un marquage $M(k-1)$.

Proposition 9 *L'ensemble des steps franchissables à partir du marquage $M(k-1)$ admet un plus grand élément $S(k)$ défini par :*

$$\forall j \in [\![1,n]\!],\ s_j(k) = \min_{p \in P} \{m_p(k-1)\ \text{div}\ W(p,t)\}\,. \tag{3.3}$$

Nous en déduisons la forme vectorielle suivante :

$$S(k) = \mathrm{Pre}^T \circ M(k-1). \tag{3.4}$$

Preuve. Soit $M = (m_p)_{p \in P}$ un marquage pour le réseau de Petri, et $S = (s_t)_{t \in T}$ un step. Avec la loi de franchissement donnée pour les steps, on déduit l'inégalité suivante :

$$M[S\rangle \Longleftrightarrow \forall p \in P,\ m_p \geqslant \sum_{t \in T} s_t \cdot W(p,t)$$

Or, un seul des termes de la somme $\sum_{t \in T} s_t.W(p,t)$ est non nul car une place d'un GEP ne possède qu'une seule transition d'entrée ($|{}^\bullet p| = 1$). On obtient donc :

$$M[S\rangle \Longleftrightarrow \forall p \in P, \forall t \in T,\ m_p \geqslant s_t \cdot W(p,t)$$

Si $p \notin {}^\bullet t$, alors $W(p,t) = 0$. D'où :

$$\begin{aligned} M[S\rangle &\Longleftrightarrow \forall t \in T, \forall p \in {}^\bullet t,\ m_p \geqslant s_t \cdot W(p,t) \\ &\Longleftrightarrow \forall t \in T, s_t \leqslant \min_{p \in {}^\bullet t} \left\{ \frac{m_p}{W(p,t)} \right\} \end{aligned} \tag{3.5}$$

Le plus grand entier respectant l'inégalité (3.5) est $\min_{p \in {}^\bullet t} \{m_p\ \text{div}\ W(p,t)\}$. Puisque le nombre s_t, pour $t \in T$, est un entier, on obtient la relation :

$$M[S\rangle \Longleftrightarrow \forall t \in T,\ s_t \leqslant \min_{p \in {}^\bullet t} \{m_p\ \text{div}\ W(p,t)\}$$

Or, si $p \notin {}^\bullet t$, alors $W(p,t) = 0$, d'où $m_p\ \text{div}\ W(p,t) = +\infty$. La relation précédente s'écrit donc :

$$M[S\rangle \Longleftrightarrow \forall t \in T,\ s_t \leqslant \min_{p \in T} \{m_p\ \text{div}\ W(p,t)\}$$

Ainsi, l'ensemble des steps franchissables à partir de M admet un unique plus grand élément S vérifiant l'équation (3.3). ■

Comme l'illustration 3.2 le montre, le franchissement d'un step $S(k)$ conduit à un nouveau marquage $M(k)$. L'équation d'état (A.2) d'un réseau de Petri permet alors d'obtenir une relation liant deux marquage successifs.

Corollaire 1 *On peut établir, pour deux marquages successifs $M(k)$ et $M(k+1)$, la récurrence suivante :*

$$M(k+1) = M(k) + C \cdot (\mathrm{Pre}^T \circ M(k)) \tag{3.6}$$

Où C est la matrice d'incidence associée au réseau de Petri

Remarque 4 *L'opérateur « $\circ$ » symbolise le formalisme* (min , div), *ne se prête pas au calcul analytique. En effet, l'équation (3.6) ne permet pas l'obtention de l'équation récurrente ci-dessous :*

$$M(k+1) = M(k) + (C \cdot \mathrm{Pre}^T) \circ M(k) \tag{3.7}$$

À titre d'exemple nous pouvons donner les valeur des matrices C, Pre *et $C \cdot \mathrm{Pre}^T$). Pour l'exemple 3.7(a), Nous obtenons le valeurs suivantes :*

$$C = \begin{pmatrix} -1 & 0 & 0 & 2 \\ 1 & 0 & -2 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 3 & -3 \\ -1 & 2 & 0 & 0 \\ 1 & -2 & 0 & 0 \end{pmatrix} \quad avec \ \mathrm{Pre} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 3 \\ 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \end{pmatrix}$$

Nous en déduisons la valeur de $C \cdot \mathrm{Pre}^T$ (produit au sens usuel) :

$$C \cdot \mathrm{Pre}^T = \begin{pmatrix} -1 & 0 & 0 & 0 & 6 & -1 & 0 \\ 1 & -4 & 0 & -2 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 3 & 0 & -2 \\ 0 & -2 & 1 & -1 & 0 & 0 & 2 \\ 0 & 6 & 0 & 3 & -9 & 0 & 0 \\ -1 & 0 & 2 & 0 & 0 & -1 & 4 \\ 1 & 0 & -2 & 0 & 0 & 1 & -4 \end{pmatrix}$$

Ce qui permettrait de pouvoir exprimer un marquage $M(k)$ quelconque en fonction du marquage initial sans pour autant devoir calculer les marquages intermédiaires. Malheureusement, les propriétés de l'opérateur div *ne permettent pas la factorisation opérée pour l'obtention de l'équation (3.7). Aussi celle-ci n'est pas égale à l'équation (3.6). La preuve de cette affirmation est réalisée à l'aide d'un contre exemple. En effet, les matrices C,* Pre *et $C \cdot$* Pre *de l'expression (3.7) (ci-dessus), pour un marquage $M = (1,0,\ldots 0)$ donnent les résultats suivants :*

- *Pour l'équation (3.6) :* 1
- *Pour l'équation (3.7) :* -1

Par conséquent, la recherche d'une formule donnant directement un marquage $M(k)$ en fonction du marquage initial revient implicitement à calculer les marquages intermédiare $M(1),M(2),\ldots,M(k-1)$.

Ce système d'équations déterministe s'avère donc très utile pour calculer par ordinateur à partir d'un état initial donné le comportement dynamique d'un GEP. Cependant il s'avère inutilisable en l'état pour rechercher un marquage initial $M(0)$ (parmi un ensemble de possibilités) qui conduirait à un comportement cyclique spécifié. Ce qui justifie pleinement l'utilisation d'heuristiques pour résoudre le type de problème correspondant à une recherche de marquage initial.

Lors de l'étude du comportement logique d'un graphe d'événements pondéré, nous sommes amenés à calculer les différents marquages qui constituent le régime cyclique du GEP. Aussi le calcul des transitions franchies à partir d'un marquage pour obtenir le successeur s'avère nécessaire, ce qui est fait dans la proposition 9. Puis le résultat du corollaire 1 permet de donner le marquage $M(k+1)$ obtenu après le franchissement du step maximal $S(k)$. En conclusion, nous sommes capable d'évaluer le comportement logique d'un graphe d'événements pondéré tel que le décrit la figure 3.2. Cependant l'équation (3.6) récurrente permet simplement d'évaluer le marquage, étape par étape. Malgré la difficulté que nous venons de soulever, les résultats décris dans ce paragraphe vont cependant nous permettre d'étudier le comportement logique de manière formelle en utilisant des compteurs sur les transitions, mémorisant le nombre de franchissements de celle-ci. Ce qui est l'objet du paragraphe suivant.

3.3.2 Équations récurrentes sur le nombre d'occurrences des transitions

Le nombre d'occurrences d'une transition correspond un compteur de franchissements successifs de celle-ci. Nous allons utiliser les nombres d'occurrences des transitions sous forme d'un vecteur, ce qui permettra de traduire le comportement logique du réseau sous forme d'une équation vectorielle. Dans le but d'exhiber les relations liant les vecteurs d'occurrences successifs. Pour les GES, nous avons rappelé dans [TBG01] le caractère linéaire des relations. La présence de poids non unitaires sur les arcs, nous conduit à utiliser la structure algébrique (min , div) définie dans le paragraphe précédent.

On définit le *vecteur d'occurrences* $N(k) = (n_i(k))_{1 \leqslant i \leqslant n}$ après le franchissement des steps $S(1)$, $S(2)$, ..., $S(k)$ par :

$$\forall i \in [\![1,n]\!], n_i(0) = 0 \text{ et } \forall k \geqslant 1, n_i(k) = \sum_{j=1}^{k} s_i(j)$$

Un premier résultat consiste à établir une équation de récurrence du premier ordre sur le vecteur d'occurrences $N(k)$:

Proposition 10 *Pour les GEP, le vecteur d'occurrences vérifie l'équation du premier ordre suivante :*

$$\forall k \geqslant 1, N(k) = N(k-1) + \mathrm{Pre}^T \circ [M(0) + C \cdot N(k-1)] \tag{3.8}$$

Preuve. Par définition de $N(k)$ et d'après l'équation (3.4), on obtient :

$$\begin{aligned} \forall k \geqslant 1, N(k) &= N(k-1) + S(k) \\ &= N(k-1) + \mathrm{Pre}^T \circ M(k-1) \end{aligned}$$

L'équation d'état se traduit par :

$$M(k-1) = M(0) + C \cdot (S(1) + S(2) + \ldots + S(k-1))$$

Or $S(1) + S(2) + \ldots + S(k-1) = N(k-1)$. Ce qui justifie l'équation suivante :

$$M(k-1) = M(0) + C \cdot N(k-1)$$

Ce qui prouve la validité de la relation (3.8). ■

L'équation précédente est une équation récurrente. Mais le problème rencontré est que, comme dans le paragraphe précédent, les propriétés de la loi div et donc de la loi « $\circ$ » ne permettent pas d'obtenir une formule simple pour évaluer un vecteur d'occurrences

d'indice quelconque en fonction de la valeur du vecteur initial. Une telle formule reviendrait à calculer les nombres d'occurrences intermédiaires dû à la non associativité de la loi div.

Cette équation traduit bien le caractère causal des GEP et du modèle associé, le modèle n'est malheureusement pas linéaire et ne permet pas une résolution de type recherche de point fixe [CTGG99] d'une itération autrement que par énumération de chacun des états de $N(k)$.

Par contre, le résultat précédent permet d'obtenir la forme des composantes $n_i(k)$ (associées à la transition t_i) du vecteur $N(k)$ en fonction de la valeur des composantes précédentes :

Théorème 5 *La composante $n_i(k)$ vérifie l'équation :*

$$\forall k \geqslant 1, n_i(k) = \min_{p \in {}^\bullet t_i} \left(\left\lfloor \frac{m_p(0) + W({}^\bullet p,p) n_{\bullet p}(k-1)}{W(p,t_i)} \right\rfloor \right) \tag{3.9}$$

Preuve. La preuve est donnée dans Soit $i \in [\![1,n]\!]$. De l'équation (3.8), on obtient pour le calcul de $n_i(k)$.(Rappel : $X \operatorname{div} 0 = \left\lfloor \frac{X}{0} \right\rfloor = +\infty$)

$$n_i(k) = n_i(k-1) + \min_{p \in P} \left\lfloor \frac{m_p(0) + \min\limits_{t \in T} \left((W(t,p) - W(p,t)) n_t(k-1) \right)}{W(p,t_i)} \right\rfloor$$

Si une transition $t \notin {}^\bullet p$, alors $W(t,p) = 0$ et si $t \notin p^\bullet$ alors $W(p,t) = 0$.

$$n_i(k) = n_i(k-1) + \min_{p \in P} \left\lfloor \frac{m_p(0) + W({}^\bullet p,p) n_{\bullet p}(k-1) - W(p,p^\bullet) n_{p^\bullet}(k-1)}{W(p,t_i)} \right\rfloor$$

Si $p \notin {}^\bullet t_i$, alors $W(p,t_i) = 0$.

$$\begin{aligned} n_i(k) &= n_i(k-1) + \min_{p \in {}^\bullet t_i} \left\lfloor \frac{m_p(0) + W({}^\bullet p,p) n_{\bullet p}(k-1) - W(p,t_i) n_{t_i}(k-1)}{W(p,t_i)} \right\rfloor \\ &= \min_{p \in {}^\bullet t_i} \left(\left\lfloor \frac{m_p(0) + W({}^\bullet p,p) n_{\bullet p}(k-1)}{W(p,t_i)} \right\rfloor \right) \end{aligned}$$

■

Cette formule permet d'évaluer concrètement le nombre de franchissements possible d'une transition t quelconque du réseau connaissant son marquage initial. À titre d'exemple, l'application du résultat donne pour la transition t_2 de la figure 3.7 :

- pour le réseau de Petri représenté par la figure 3.7(a). Nous obtenons :

$$n_2(k) = \min \left(\left\lfloor \frac{1 + n_1(k-1)}{2} \right\rfloor, 1 + n_4(k-1) \right) \tag{3.10}$$

– pour le réseau de Petri représenté par la figure 3.7(b). Nous obtenons :

$$n_2(k) = \min(1 + n_1(k-1), 1 + n_4(k-1)) \tag{3.11}$$

L'équation (3.8) traduit, en général, un comportement non linéaire : c'est-à-dire qu'on ne peut le réduire à un jeu d'équations (min ,+)-linéaires. Ce qui est le cas pour le graphe d'événements pondéré représenté par la figure 3.7(a), l'équation (3.10) ne s'exprime pas dans le formalisme (min ,+) dû à la présence d'une division euclidienne contrairement à l'équation (3.11). Cependant, pour une certaine classe de GEP, il sera possible de linéariser le système d'équations associé.

3.4 Analyse du comportement logique d'un GEP

Nous allons aborder l'étude de la linéarisation de systèmes d'équations. Le vecteur d'occurrences des transitions, introduit précédemment, permet de formaliser le comportement logique d'un graphe d'événements pondéré. La difficulté rencontrée consiste en un calcul complexe des composantes du vecteur d'occurrences. En effet, les équations donnant la valeur des composantes font intervenir une division euclidienne.

Dans ce paragraphe, nous proposons tout d'abord d'exhiber le système d'équations associé au comportement d'un graphe d'événements pondéré ainsi que les difficultés liées à cette étude. Puis nous dégagerons une méthode de linéarisation pour une large classe de GEP, qu'on appellera *graphes d'événements pondérés linéarisables* (GEPL), dont on donnera les spécificités par la suite. Le but est de donner une méthode de linéarisation des équations caractérisant le comportement du GEP, afin de rejoindre les études faites pour les graphes d'événements simples.

Ce paragraphe comporte trois parties. la première expose le système d'équations caractérisant le comportement de l'exemple représenté par la figure 3.8 et la difficulté liée à l'étude d'un tel système. Puis nous proposerons, au sein de la deuxième partie, un algorithme de linéarisation. L'algorithme permet, pour la classe des GEPL, d'obtenir un système d'équations (min ,+)-linéaires. La troisième partie est réservée à l'application des résultats sur l'exemple principal de cette thèse.

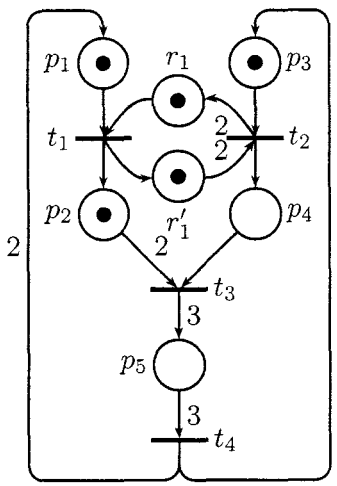


FIG. 3.8 – *Exemple illustratif*

3.4.1 Équations du réseau

Dans ce paragraphe, Nous considérons le GEP représenté par la figure 3.8. Il représente une partie du GEP global représenté par la figure 3.1, le but est de bien comprendre le fonctionnement de la mise en équation. Dans ce sens, nous allons mettre en équation le comportement de ce GEP. Aussi nous verrons : d'une part, l'utilité de la structure algébrique mise en jeu dans le paragraphe précédent; et d'autre part, les difficultés rencontrées lors de cette étude.

À l'aide des résultats précédent (d'après l'équation (3.9)), nous obtenons le système d'équations suivant :

$$\begin{cases} n_1(k) = \min\left(m_{p_1}(0) + 2n_4(k-1), m_{r_{11}}(0) + 2n_2(k-1)\right) \\ n_2(k) = \min\left(m_{p_3}(0) + n_4(k-1), \left\lfloor \dfrac{m_{r_{12}}(0) + n_1(k-1)}{2} \right\rfloor\right) \\ n_3(k) = \min\left(\left\lfloor \dfrac{m_{p_2}(0) + n_1(k-1)}{2} \right\rfloor, m_{p_4}(0) + n_2(k-1)\right) \\ n_4(k) = \left\lfloor \dfrac{3n_3(k-1) + m_{p_5}(0)}{3} \right\rfloor \end{cases} \tag{3.12}$$

Pour obtenir un système d'équations caractérisant en toute généralité le comportement de notre exemple, nous avons abstrait le marquage initial. Pour le cas particulier du marquage donné, nous obtenons :

$$\begin{cases} n_1(k) = \min\left(1 + 2n_4(k-1), 1 + 2n_2(k-1)\right) \\ n_2(k) = \min\left(1 + n_4(k-1), \left\lfloor \frac{1 + n_1(k-1)}{2} \right\rfloor\right) \\ n_3(k) = \min\left(\left\lfloor \frac{1 + n_1(k-1)}{2} \right\rfloor, n_2(k-1)\right) \\ n_4(k) = n_3(k-1) \end{cases} \tag{3.13}$$

Le système ainsi obtenu ne peut engendrer une équation matricielle récurrente linéaire en raison de la présence de parties entières rendant les équations non linéaires (la linéarité s'entend ici au sens de l'algèbre $(\min, +)$). Notre exemple est à la limite du comportement linéaire, puisque son fonctionnement est très simple à décrire pour un marquage donné. À contrario, la description algébrique semble complexe mais nous allons voir que pour une classe particulière de graphes d'événements pondérés une linéarisation du système d'équations est possible.

3.4.2 Description formelle de l'algorithme de linéarisation GEP/GES

En raison du caractère non linéaire des équations d'état récurrente (3.13), nous proposons de rechercher les conditions pour lesquelles une linéarisation des GEP est possible tout en restant indépendante du marquage initial pour garder un caractère généraliste de la méthode. Cette contrainte d'indépendance est forte, car elle permet en cas de linéarisation d'exhiber une technique originale et efficace de recherche d'un marquage initial optimal. De plus, elle se différencie de la méthode d'Alix Munier (3.1.2) (cf. paragraphe 3.1.2) ainsi que nous le verrons en fin de ce chapitre.

Algorithme de linéarisation

Considérons le GEP consistant de la figure 3.8 candidat à un séquencement cyclique. Le choix de la cyclicité consiste en la décomposition d'une production de moyenne ou grande série en une production répétitive de plusieurs quantité plus petites. Le séquencement d'un nombre limité de tâches dans un cycle permet de faire chuter très fortement la complexité du problème. L'optimisation est alors réalisé sur un cycle de production et non sur la production totale, ceci ne permet pas de garantir l'optimalité de la solution obtenue.

Le fait que notre GEP soit consistant implique la présence d'un invariant total de transitions (semi-flot sur transitions). Un invariant total de transitions est un vecteur dont les composantes ne sont pas forcément toutes unitaires. Pour les GES, les composantes d'un

invariant sont toutes unitaires, cela signifie que chaque transition est franchie exactement une fois par cycle, pour revenir au marquage initial. Un autre exemple concerne le fait que chaque transition d'un GEP peut être tirée au moins une fois par cycle (peut-être plus) pour revenir au marquage initial et recommencer un nouveau cycle. L'idée est donc de *normaliser* l'invariant de transitions d'un GEP de manière à ce que les transitions aient « virtuellement » le même nombre d'occurrences par cycle. Afin d'illustrer la théorie exposée ci-après, nous appliquerons les résultats sur le graphe d'événements pondéré représenté par la figure 3.8 afin de bien comprendre les concepts mis en jeu.

Nous notons l'invariant de transition $\theta = \begin{pmatrix} \theta_1 & \theta_2 & \dots & \theta_n \end{pmatrix}^T$. Cet invariant vérifie l'expression $C \cdot \theta = 0$. Donc on en déduit l'équation ([Mun93]) suivante :

$$\begin{aligned} &\forall p \in P, W(^\bullet p,p) \cdot \theta_{\bullet p} - W(p,p^\bullet) \cdot \theta_{p^\bullet} = 0 \\ &\forall p \in P, \frac{\theta_{\bullet p}}{W(p,p^\bullet)} = \frac{\theta_{p^\bullet}}{W(^\bullet p,p)} \end{aligned} \tag{3.14}$$

On pose ppcm le plus petit commun multiple des entiers θ_i. Par conséquent, pour tout $i \in [\![1,n]\!]$, il existe un entier λ_i tel que $\theta_i \cdot \lambda_i = \text{ppcm}$. On introduit le vecteur d'état $X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \end{pmatrix}^T$ défini par $x_i = \lambda_i \cdot n_i$, pour tout $i \in [\![1,n]\!]$.

Définition 12 *Si* $(\theta_i)_{1 \leqslant i \leqslant n}$ *représente l'invariant de transitions, on appelle* vecteur d'occurrences normalisé *le vecteur* $X = (x_i)_{1 \leqslant i \leqslant n}$ *défini à partir du vecteur d'occurrences* $N = (n_i)_{1 \leqslant i \leqslant n}$, *selon les relations suivantes :*

$$x_i = \lambda_i \cdot n_i \text{ où } \lambda_i = \frac{\text{ppcm}}{\theta_i} \text{ avec } \text{ppcm} = \underset{1 \leqslant i \leqslant n}{\text{ppcm}}(\theta_i) \tag{3.15}$$

exemple : l'invariant de transitions du GEP représenté par la figure 3.8 est : (2,1,1,1). cela signifie que, lors d'un cycle, après le franchissement de : $2t_1$, t_2, t_3, t_4 nous revenons au marquage initial. Dasn ce cas, il vient ppcm = 2, $\lambda_1 = 2$, $\lambda_3 = 1$, $\lambda_4 = 1$. Nous obtenons ainsi le vecteur d'occurrences normalisé suivant :

$$\begin{aligned} x_1(k) &= 2n_1(k) \\ x_2(k) &= n_2(k) \\ x_3(k) &= n_3(k) \\ x_4(k) &= n_4(k) \end{aligned}$$

Rappelons que l'évolution du vecteur d'occurrences d'un GEP est régie par le système d'équations (3.9) mais il ne possède pas la propriété de (min ,+)-linéarité. On se

propose de voir comment ce système est transformé pa l'utilisation du vecteur d'occurrences normalisé.

Lemme 3 *Pour tout entier $k \in \mathbb{N}$, les composantes x_i du vecteur d'occurrences normalisé peuvent s'écrire sous la forme*

$$x_i(k) = \min_{t_j \in {}^{\bullet\bullet}t_i} \left\{ \lambda_i \left\lfloor \frac{x_j(k-1) + \gamma_{ij}}{\lambda_i} \right\rfloor \right\} \tag{3.16}$$

où les $(\gamma_{ij})_{t_j \in {}^{\bullet\bullet}t_i}$ sont des constantes entières dépendantes du marquage initial.

Preuve.

On note p_{ji} la place, si elle existe, qui se situe entre les transitions t_j et t_i. Si on utilise les notations de la définition 12, on a alors d'après le théorème 3 :

$$x_i(k) = \min_{t_j \in {}^{\bullet\bullet}t_j} \left\{ \lambda_i \left\lfloor \frac{W(t_j,p_{ji}) \dfrac{\theta_j}{\text{ppcm}} x_j(k-1) + m_{p_{ji}}(0)}{W(p_{ji},t_i)} \right\rfloor \right\} \text{ car } n_j(k) = \frac{\theta_j}{\text{ppcm}} x_j(k)$$

$$= \min_{t_j \in {}^{\bullet\bullet}t_i} \left\{ \lambda_i \left\lfloor \frac{x_j(k-1) + \dfrac{\text{ppcm}}{\theta_j W(t_j,p_{ji})} m_{p_{ji}}(0)}{\text{ppcm} \dfrac{W(p_{ji},t_i)}{\theta_j W(t_j,p_{ji})}} \right\rfloor \right\}$$

Or nous déduisons de l'équation (3.14) et de l'équation (3.15) les résultats suivants :

$$\frac{W(p_{ji},t_i)}{\theta_j W(t_j,p_{ji})} = \frac{1}{\theta_i} \text{ et } \text{ppcm} \frac{1}{\theta_i} = \lambda_i$$

Donc $x_i(k) = \min_{t_j \in {}^{\bullet\bullet}t_i} \left\{ \lambda_i \left\lfloor \dfrac{x_j(k-1) + \gamma_{ij}}{\lambda_i} \right\rfloor \right\}$ avec $\gamma_{ij} = \dfrac{\lambda_j}{W(t_j,p_{ji})} m_{p_{ji}}(0)$. ■

Même si certaines des équations du système ainsi obtenu peuvent être linéaires (par exemple, si $\lambda_i = 1$), ce n'est pas le cas en général. Le principe de la linéarisation, décrit par l'organigramme de la figure 3.9 consiste à intégrer progressivement toute expression non linéaire dans le vecteur d'état. Aussi, nous commençons par l'introduction d'un nouveau vecteur d'état Y contenant les composantes du vecteur X ($y_i = x_i$ pour $i \in [\![1,n]\!]$). Le vecteur Y vérifie le système d'équations (3.16). Si le système contient l'expression non linéaire du type $f(y_i(k-1))$, nous introduisons dans le vecteur d'état une nouvelle composante, que l'on note $y_{n+1}(k)$, définie par $y_{n+1}(k) = f(y_i(k)) - f(y_i(0))$. La constante $f(0)$ est soustraite

de l'expression non linéaire pour garantir à l'état initial $y_{n+1}(0) = 0$. Cette nouvelle composante permet de linéariser l'expression qui était initialement non linéaire puisqu'elle s'écrira $f(y_i(k-1)) = y_{n+1}(k-1) + f(0)$. Il s'agit maintenant d'exprimer la nouvelle composante $y_{n+1}(k)$ en fonction des autres composantes du vecteur $Y(k-1)$. Ce procédé est réitéré tant qu'il subsiste des expressions non-linéaires. Une question se pose ici est donc de savoir si cette procédure se termine en un nombre fini d'itérations?

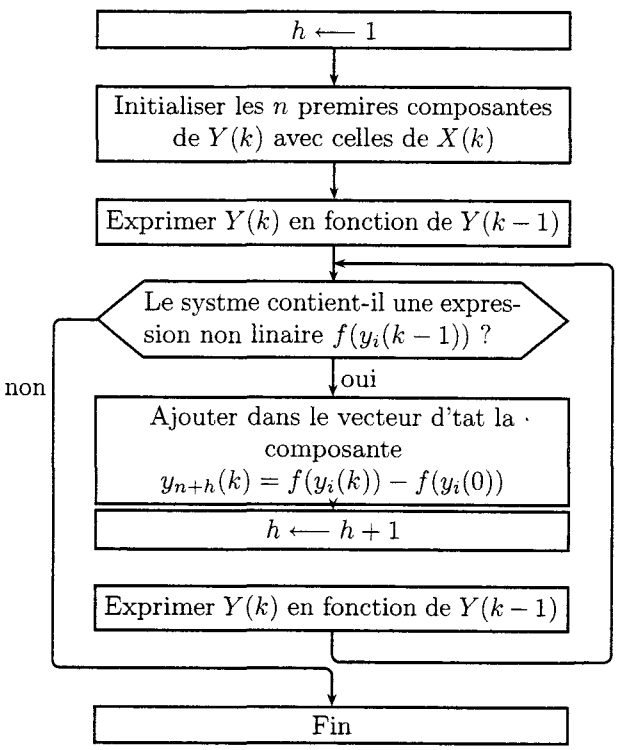


FIG. 3.9 – *Procédure de linéarisation*

Application sur un exemple simple

Appliquons le processus de normalisation à l'exemple 3.8, la première étape consiste en la normalisation du système :

$$\begin{cases} x_1(k) = \min\left(m_{p_1}(0) + x_4(k-1), m_{r_1}(0) + x_2(k-1)\right) \\ x_2(k) = \min\left(2m_{p_3}(0) + x_4(k-1), 2\left\lfloor \dfrac{m_{r_1'}(0) + x_1(k-1)}{2} \right\rfloor\right) \\ x_3(k) = \min\left(2m_{p_4}(0) + x_2(k-1), 2\left\lfloor \dfrac{m_{p_2}(0) + x_1(k-1)}{2} \right\rfloor\right) \\ x_4(k) = 2\left\lfloor \dfrac{m_{p_5}(0)}{3} \right\rfloor + x_3(k-1) \end{cases} \tag{3.17}$$

L'expression $2\left\lfloor \dfrac{m_{r_1'}(0) + x_1(k-1)}{2} \right\rfloor$ rend la deuxième équation de ce système non linéaire. Le principe de linéarisation consiste à intégrer cette expression dans le vecteur d'état. Aussi, nous utiliserons un nouveau vecteur d'état $Y(k)$ contenant les composantes du vecteur $X(k)$ (pour $i \in [\![1,4]\!], y_i(k) = x_i(k)$) ainsi que la composante non linéaire

$$y_5(k) = 2\left\lfloor \frac{m_{r_1'}(0) + x_1(k)}{2} \right\rfloor - 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor$$

La constante $2\left\lfloor \dfrac{m_{r_1'}(0)}{2} \right\rfloor$ permet d'avoir à l'état initial $y_5(0) = 0$, ceci permet une initialisation correcte du vecteur d'occurrences tel que : $Y(k) = 0$ et ainsi les nombres de franchissements des transitions réelles ou virtuelles sont initialisés à 0. Par ailleurs, la seconde équation du système (3.17) s'écrit :

$$y_2(k) = \min\left(2m_{p_3}(0) + y_4(k-1), y_5(k-1) + 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor\right)$$

De la même façon nous ajoutons la composante suivante :

$$y_6(k) = 2\left\lfloor \frac{m_{p_2}(0) + x_1(k)}{2} \right\rfloor - 2\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor$$

Ces composantes sont intégrées dans le vecteur d'état $Y(k)$.

Il s'agit maintenant d'exprimer les composantes $y_5(k)$ et $y_6(k)$ en fonction de celles de $Y(k-1)$: si l'on obtient des expressions non linéaires alors celles-ci devront également être intégrées dans le vecteur d'état. En notant :

$$\alpha = 2\left\lfloor \frac{m_{r_1}(0) + m_{r_1'}(0)}{2} \right\rfloor - 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor$$
$$\beta = 2\left\lfloor \frac{m_{p_1}(0) + m_{r_1'}(0)}{2} \right\rfloor - 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor$$
$$\gamma = 2\left\lfloor \frac{m_{r_1}(0) + m_{p_2}(0)}{2} \right\rfloor - 2\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor$$
$$\delta = 2\left\lfloor \frac{m_{p_1}(0) + m_{p_2}(0)}{2} \right\rfloor - 2\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor$$

Nous obtenons :

$$\begin{cases} y_1(k) = \min\left(m_{p_1}(0) + y_4(k-1), m_{r_1}(0) + y_2(k-1)\right) \\ y_2(k) = \min\left(2m_{p_3}(0) + y_4(k-1), y_5(k-1) + 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor\right) \\ y_3(k) = \min\left(2m_{p_4}(0) + y_2(k-1), y_6(k-1) + 2\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor\right) \\ y_4(k) = y_3(k-1) + 2\left\lfloor \frac{m_{p_5}(0)}{3} \right\rfloor \\ y_5(k) = \min\left(\alpha + y_2(k-1), \beta + y_4(k-1)\right) \\ y_6(k) = \min\left(\gamma + y_2(k-1), \delta + y_4(k-1)\right) \end{cases} \tag{3.18}$$

Il vient finalement, le système $(\min, +)$-linéaire de dimension 6 augmenté par rapport au système (3.17) (de dimension 4) et comportant deux composantes non linéaires de x_1. Le système précédent peut être décrit sous forme matricielle (l'opérateur $\otimes$ dénote le produit matriciel au sens de $(\min, +)$), comme suit :

$Y(k) = B(M(0)) \otimes Y(k-1)$ où $B(M(0))$ est décrite ci-dessous :

$$\begin{pmatrix} +\infty & m_{r_1}(0) & +\infty & m_{p_1}(0) & +\infty & +\infty \\ +\infty & +\infty & +\infty & 2m_{p_3}(0) & 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor & +\infty \\ +\infty & 2m_{p_4}(0) & +\infty & +\infty & +\infty & 2\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor \\ +\infty & +\infty & 2\left\lfloor \frac{m_{p_5}(0)}{3} \right\rfloor & +\infty & +\infty & +\infty \\ +\infty & \alpha & +\infty & \beta & +\infty & +\infty \\ +\infty & \gamma & +\infty & \delta & +\infty & +\infty \end{pmatrix}$$

Le processus de linéarisation vient de se terminer en un seul pas par l'obtention d'un système d'équations $(\min, +)$-linéaires. Nous allons maintenant aborder la question délicate

des conditions d'arrêt de l'algorithme. Dans le paragraphe suivant, nous verrons qu'effectivement le processus se termine sous une condition qui constituera la limite d'application de ce résultat.

Terminaison de l'algorithme

L'organigramme de la figure 3.9 présente une procédure de linéarisation des équations qui régissent le comportement d'un GEP. Le but de ce paragraphe est de montrer que pour une certaine classe de GEP, la procédure est en fait un algorithme, c'est-à-dire que la boucle contenue dans la procédure s'arrête pour conduire à un système d'équations fini dont la caractéristique principale est d'être $(\min, +)$-linéaire.

Rappelons que la non linéarité des expressions provient de la division euclidienne. Cependant, certaines transitions particulières qu'on appellera *transitions de normalisation* n'induisent pas de non linéarité.

Définition 13 *Une* ***transition de normalisation*** *est une transition pour laquelle le cœfficient qui lui est associé dans l'invariant de transitions vaut* 1.

Pour l'exemple représenté par la figure 3.8, nous constatons la présence de trois transitions de normalisation, nommément : t_2, t_3 et t_4.

La définition précédente nous permet d'énoncer une condition suffisante pour que la procédure décrite figure 3.9 s'arrête :

Théorème 6 *La procédure de linéarisation s'arrête si tout circuit du GEP contient au moins une transition de normalisation. Cette classe de GEP sera appelé* graphes d'événements pondérés linéarisables *(GEPL).*

Preuve. Soit λ et γ deux entiers. On note $f_{\lambda,\gamma}$ la fonction définie par :

$$\forall x \in \mathbb{N}, f_{\lambda,\gamma}(x) = \lambda \left\lfloor \frac{\gamma + x}{\lambda} \right\rfloor$$

Commençons par démontrer qu'après chaque itération, le système d'équations, qui donnent $Y(k)$ en fonction de $Y(k-1)$, fait apparaître des expressions de la forme :

$$f_{\lambda_{i_1},\gamma_{j_1}} \circ f_{\lambda_{i_2},\gamma_{j_2}} \cdots \circ f_{\lambda_{i_l},\gamma_{j_l}}(y_r(k-1))$$

tel que :

i) il existe un entier i_{l+1} tel que $y_r(k-1)$ soit un multiple de $\lambda_{i_{l+1}}$;

ii) $t_{i_{l+1}} \in {}^{\bullet\bullet}t_{i_l}$, $t_{i_l} \in {}^{\bullet\bullet}t_{i_{l-1}}$, ..., $t_{i_2} \in {}^{\bullet\bullet}t_{i_1}$;

iii) les transitions $t_{i_2}, t_{i_3}, \ldots, t_{i_l}$ ne sont pas des transitions de normalisation

Cela se démontre facilement par récurrence sur le nombre d'itérations.

Cette proposition est vraie à la première itération d'après le lemme 3.

Supposons que ce soit le cas jusqu'à l'itération courante.

Montrons que ce résultat est vrai pour l'itération suivante. Si le système ne contient pas d'expressions non linéaires, alors le résultat est vrai. Sinon, d'après l'hypothèse de récurrence, il existe une expression non-linéaire qui s'écrit :

$$f(y_r(k-1)) = \overset{l}{\underset{q=1}{\circ}} \left(f_{\lambda_{i_q},\gamma_{j_q}}\right)(y_r(k-1))$$

où y_r est un multiple de $\lambda_{i_{l+1}}$.

La procédure de linéarisation consiste à ajouter dans le vecteur d'état la composante, que l'on note $y_s(k)$, définie par $y_s(k) = f(y_r(k)) - f(0)$. Cette composante s'écrit également

$$f_{\lambda_{i_1},\gamma'_{j_1}} \circ \left(\overset{l}{\underset{q=2}{\circ}} \left(f_{\lambda_{i_q},\gamma_{j_q}}\right)\right)(y_r(k))$$

avec $\gamma'_{j_1} = \gamma_{j_1} + f(0)$ car $f(0)$ est un multiple de λ_{i_1}.

D'après l'hypothèse de récurrence, $y_r(k)$ ne contient que des expressions de la forme $\overset{h}{\underset{q=l+1}{\circ}} \left(f_{\lambda_{i_q},\gamma_{j_q}}\right)(y_h(k-1))$ où $y_h(k-1)$ est un multiple de $\lambda_{i_{h+1}}$. La composante $y_s(k)$ ne contient donc que des expressions de la forme $\overset{h}{\underset{q=1}{\circ}} \left(f_{\lambda_{i_q},\gamma_{j_q}}\right)(y_h(k-1))$. Cette expression vérifie donc i) et ii) d'après l'hypothèse de récurrence. Pour démontrer iii), d'après l'hypothèse de récurrence, il reste à prouver que $t_{i_{l+1}}$ n'est pas une transition de normalisation. Si c'était le cas alors $\lambda_{i_{l+1}}$ serait un multiple des λ_{i_q}. On aurait alors $f_{\lambda_{i_l},\gamma_{j_l}}(y_r(k-1)) = \lambda_{i_l}\left\lfloor \dfrac{\gamma_{j_l}}{\lambda_{i_l}} \right\rfloor + y_r(k-1)$ car $\dfrac{y_r(k-1)}{\lambda_{i_l}}$ est un entier et peut être être sorti de la partie entière. Si l'on pose $\gamma'_{j_{l-1}} = \gamma_{j_{l-1}} + \lambda_{i_l}\left\lfloor \dfrac{\gamma_{j_l}}{\lambda_{i_l}} \right\rfloor$, on a alors :

$$f_{\lambda_{i_{l-1}},\gamma_{j_{l-1}}} \circ f_{\lambda_{i_l},\gamma_{j_l}}(y_r(k-1)) = f_{\lambda_{i_{l-1}},\gamma'_{j_{l-1}}}(y_r(k-1))$$

Ainsi, de proche en proche, l'expression $\overset{l}{\underset{q=1}{\circ}} \left(f_{\lambda_{i_q},\gamma_{j_q}}\right)(y_r(k-1))$ serait linéaire, ce qui contredit les hypothèses. Finalement iii) est vérifié.

On se propose maintenant de démontrer que la procédure de linéarisation termine. Si ce n'était pas le cas, alors il existerait une expression $\overset{l}{\underset{q=1}{\circ}} \left(f_{\lambda_{i_q},\gamma_{j_q}}\right)(y_r(k))$ dont la taille

(l) serait aussi grande que l'on veut et vérifiant ii) et iii). Cela signifierait qu'il existerait dans le réseau un chemin, dont la taille n'est pas bornée, ne contenant pas de transition de normalisation. Étant donné qu'il n'existe qu'un nombre fini de transitions, le chemin en question peut être supposé suffisament long pour contenir toutes les transitions du réseau. Or, par hypothèse, tous les circuits contiennent une transition de normalisation. On conclut que la procédure de linéarisation est un algorithme pour la classe des GEPL. ■

Le théorème 6 précédent permet de définir une nouvelle classe de réseau de Petri. La particularité de cette classe est que le GEP peut être abstrait en un système d'équations linéaires caractérisant son comportement. Par contre, le théorème 6 impose une condition suffisante de linéarisation. Par conséquent, tous les réseau de Petri dont le comportement peut s'abstraire en un système d'équations linéaires n'est pas entièrement caractérisé. L'extension de cette classe de réseau de Petri fait partie des perspectives du travail présenté.

3.5 Résolution et champ d'application

Considérons la figure 3.1. On commence par rechercher l'invariant total de transitions θ. Cet invariant vérifie $\theta = \begin{pmatrix} 2 & 1 & 1 & 1 & 1 & 3 & 2 & 3 & 2 & 1 \end{pmatrix}$. On peut facilement vérifier que tous les circuits du réseau possèdent une transition de normalisation. L'exemple proposé appartient donc bien à la classe des GEPL et nous utiliserons la technique de linéarisation du système d'équations sur cet exemple à titre d'illustration. On considère donc le vecteur d'occurrences normalisé suivant $X = \left(x_i\right)^T_{1 \leqslant i \leqslant 10}$ défini par $x_1 = 3n_1$, $x_6 = 2n_6$, $x_7 = 3n_7$, $x_8 = 2n_8$, $x_9 = 3n_9$, pour toutes les autres composantes $x_i = 6n_i$.

Le vecteur d'occurrences normalisé satisfait le système suivant :

$$
\begin{cases}
x_1(k) = \min\left(3m_{p_1}(0) + x_4(k-1), 3m_{r_{11}}(0) + x_2(k-1)\right) \\
x_2(k) = \min\left(6m_{p_3}(0) + x_4(k-1), 6\left\lfloor \dfrac{3m_{r_{12}}(0) + x_1(k-1)}{6} \right\rfloor\right) \\
x_3(k) = \min\left(6\left\lfloor \dfrac{3m_{p_2}(0) + x_1(k-1)}{6} \right\rfloor, 6m_{p_4}(0) + x_2(k-1),\right. \\
\qquad \left. 6m_{r_{22}}(0) + x_5(k-1)\right) \\
x_4(k) = \min\left(6\left\lfloor \dfrac{m_{p_5}(0)}{3} \right\rfloor + x_3(k-1), 6m_{r_{32}}(0) + x_{10}(k-1)\right) \\
x_5(k) = \min\left(6m_{r_{21}}(0) + x_3(k-1), 6\left\lfloor \dfrac{m_{p_6}(0)}{5} \right\rfloor + x_{10}(k-1)\right) \\
x_6(k) = 2m_{p_7}(0) + x_5(k-1) \\
x_7(k) = 3m_{p_8}(0) + x_5(k-1) \\
x_8(k) = 2m_{p_9}(0) + x_6(k-1) \\
x_9(k) = 3m_{p_{10}}(0) + x_7(k-1) \\
x_{10}(k) = \min\left(6m_{r_{31}}(0) + x_4(k-1), 6\left\lfloor \dfrac{2m_{p_{11}}(0) + x_8(k-1)}{6} \right\rfloor,\right. \\
\qquad \left. 6\left\lfloor \dfrac{3m_{p_{12}}(0) + x_9(k-1)}{6} \right\rfloor\right)
\end{cases}
\tag{3.19}
$$

Parmi ces équations, la deuxième, la troisième et la dernière contiennent des expressions non linéaires. Pour linéariser ces équations, nous introduisons le vecteur d'état Y dont les dix premières composantes correspondent à celles du vecteur d'occurrences normalisé et nous ajoutons dans le vecteur d'état Y ces expressions non linéaires. Par exemple, il apparaît dans la deuxième équation l'expression non linéaire $6\left\lfloor \dfrac{3m_{r_{12}}(0) + y_1(k-1)}{6} \right\rfloor$. Nous ajoutons donc, dans le vecteur d'état Y, la composante $y_{11}(k)$ définie par :

$$
y_{11}(k) = 6\left\lfloor \frac{3m_{r_{12}}(0) + y_1(k)}{6} \right\rfloor - 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor
$$

La deuxième équation s'écrit alors :

$$
y_2(k) = \min\left(6m_{p_3}(0) + y_4(k-1), y_{11}(k-1) + 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor\right)
$$

Il reste alors à exprimer $y_{11}(k)$ en fonction des composantes de $Y(k-1)$. Pour cela, on exprime $y_1(k)$ en fonction de $Y(k-1)$ et on utilise le fait que la somme, la division, la partie entière et la multiplication sont distributives par apport à l'opérateur div. Il vient alors :

$$
\begin{aligned}
y_{11}(k) &= 6\left\lfloor \frac{3m_{r_{12}}(0) + \min\left(3m_{p_1}(0) + y_4(k-1), 3m_{r_{11}}(0) + y_2(k-1)\right)}{6} \right\rfloor \\
&\quad - 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor \\
&= \min\left(6\left\lfloor \frac{3m_{r_{12}}(0) + 3m_{p_1}(0)}{6} + \frac{y_4(k-1)}{6} \right\rfloor, \right. \\
&\quad \left. 6\left\lfloor \frac{3m_{r_{12}}(0) + 3m_{r_{11}}(0)}{6} + \frac{y_2(k-1)}{6} \right\rfloor\right) - 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor
\end{aligned}
$$

Or $y_2(k-1)$ et $y_4(k-1)$ sont, par définition, des multiples de 6. Les entiers $\frac{y_2(k-1)}{6}$ et $\frac{y_4(k-1)}{6}$ peuvent donc être sortis de la partie entière. D'où finalement la forme linéaire suivante :

$$
\begin{aligned}
y_{11}(k) = \min\Bigg(&6\left\lfloor \frac{m_{r_{12}}(0) + m_{p_1}(0)}{2} \right\rfloor + y_4(k-1), \\
&6\left\lfloor \frac{m_{r_{12}}(0) + m_{r_{11}}(0)}{2} \right\rfloor + y_2(k-1)\Bigg) - 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor
\end{aligned}
$$

L'expression non linéaire qui apparaît dans la troisième équation du système (4.12) se linéarise en introduisant la composante y_{12} définie par :

$$
y_{12}(k) = 6\left\lfloor \frac{3m_{p_2}(0) + x_1(k)}{6} \right\rfloor - 6\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor
$$

On obtient alors :

$$
\begin{aligned}
y_3(k) = \min\Bigg(&6\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor + y_{12}(k-1), 6m_{p_4}(0) + y_2(k-1), \\
&6m_{r_{22}}(0) + y_5(k-1)\Bigg) \\
y_{12}(k) = \min\Bigg(&6\left\lfloor \frac{m_{p_2}(0) + m_{p_1}(0)}{2} \right\rfloor + y_4(k-1), 6\left\lfloor \frac{m_{p_2}(0) + m_{r_{11}}(0)}{2} \right\rfloor \\
&+ y_2(k-1)\Bigg) - 6\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor
\end{aligned}
$$

Pour linéariser la dernière équation du système (4.12), on introduit les composantes $y_{13}(k)$ et $y_{14}(k)$ définies par :

$$
\begin{aligned}
y_{13}(k) &= 6\left\lfloor \frac{2m_{p_{11}}(0) + x_8(k)}{6} \right\rfloor - 6\left\lfloor \frac{m_{p_{11}}(0)}{3} \right\rfloor \\
y_{14}(k) &= 6\left\lfloor \frac{3m_{p_{12}}(0) + x_9(k)}{6} \right\rfloor - 6\left\lfloor \frac{m_{p_{12}}(0)}{2} \right\rfloor
\end{aligned}
$$

On a alors :

$$\begin{aligned}
y_{10}(k) &= \min\left(6m_{r_{31}}(0) + y_4(k-1), y_{13}(k-1) + 6y_{10}(k),\right. \\
&\qquad \left. y_{14}(k-1) + 6\left\lfloor \frac{m_{p_{12}}(0)}{2} \right\rfloor\right) \\
y_{13}(k) &= 6\left\lfloor \frac{2m_{p_{11}}(0) + 2m_{p_9}(0) + x_6(k-1)}{6} \right\rfloor - 6\left\lfloor \frac{m_{p_{11}}(0)}{3} \right\rfloor \\
y_{14}(k) &= 6\left\lfloor \frac{3m_{p_{12}}(0) + 3m_{p_{10}}(0) + x_7(k-1)}{6} \right\rfloor - 6\left\lfloor \frac{m_{p_{12}}(0)}{2} \right\rfloor
\end{aligned}$$

D'autres expressions non linéaires apparaissent. On introduit alors les composantes définies par :

$$\begin{aligned}
y_{15}(k) &= 6\left\lfloor \frac{2m_{p_{11}}(0) + 2m_{p_9}(0) + x_6(k)}{6} \right\rfloor - 6\left\lfloor \frac{m_{p_{11}}(0) + m_{p_9}(0)}{3} \right\rfloor \\
y_{16}(k) &= 6\left\lfloor \frac{3m_{p_{12}}(0) + 3m_{p_{10}}(0) + x_7(k)}{6} \right\rfloor - 6\left\lfloor \frac{m_{p_{12}}(0) + m_{p_{10}}(0)}{2} \right\rfloor
\end{aligned}$$

Finalement, nous obtenons :

$$\begin{aligned}
y_{13}(k) &= y_{15}(k-1) + 6\left\lfloor \frac{m_{p_{11}}(0) + m_{p_9}(0)}{3} \right\rfloor - 6\left\lfloor \frac{m_{p_{11}}(0)}{3} \right\rfloor \\
y_{14}(k) &= y_{16}(k-1) + 6\left\lfloor \frac{m_{p_{12}}(0) + m_{p_{10}}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{p_{12}}(0)}{2} \right\rfloor \\
y_{15}(k) &= y_5(k-1) + 6\left\lfloor \frac{m_{p_{11}}(0) + m_{p_9}(0) + m_{p_7}(0)}{3} \right\rfloor - 6\left\lfloor \frac{m_{p_{11}}(0) + m_{p_9}(0)}{3} \right\rfloor \\
y_{16}(k) &= y_5(k-1) + 6\left\lfloor \frac{m_{p_{12}}(0) + m_{p_{10}}(0) + m_{p_8}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{p_{12}}(0) + m_{p_{10}}(0)}{2} \right\rfloor
\end{aligned}$$

Nous obtenons finalement, le système $(\min, +)$-linéaire de dimension 16 augmenté par rapport au système (3.12) (de dimension 10 comportant quatre composantes non-linéaires). Le système précédent peut alors être décrit sous forme matricielle (l'opérateur $\otimes$ dénote le produit matriciel au sens de $(\min, +)$), comme suit :

$$Y(k) = B(M(0)) \otimes Y(k-1) \tag{3.20}$$

où la matrice $B(M(0))$ est représentée figure 4.8.

$$
\begin{pmatrix}
+\infty & 3m_{r_{11}}(0) & +\infty & 3m_{p_1}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & +\infty & 6m_{p_3}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & 6\left\lfloor\frac{m_{r_{12}}(0)}{2}\right\rfloor & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & 6m_{p_4}(0) & +\infty & +\infty & 6m_{r_{22}}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & 6\left\lfloor\frac{m_{p_2}(0)}{2}\right\rfloor & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & 6\left\lfloor\frac{m_{p_5}(0)}{3}\right\rfloor & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & 6m_{r_{32}}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & 6m_{r_{21}}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & 6\left\lfloor\frac{m_{p_6}(0)}{5}\right\rfloor & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & +\infty & +\infty & 2m_{p_7}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & +\infty & +\infty & 3m_{p_8}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & +\infty & +\infty & +\infty & 2m_{p_9}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & +\infty & +\infty & +\infty & +\infty & 3m_{p_{10}}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & +\infty & 6m_{r_{31}}(0) & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & 6\left\lfloor\frac{m_{p_{11}}(0)}{3}\right\rfloor & 6\left\lfloor\frac{m_{p_{12}}(0)}{2}\right\rfloor & +\infty & +\infty \\
+\infty & \alpha & +\infty & \beta & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & \gamma & +\infty & \delta & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty \\
+\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & \eta & +\infty \\
+\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & \lambda \\
+\infty & +\infty & +\infty & +\infty & \mu & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & \infty \\
+\infty & +\infty & +\infty & +\infty & \nu & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & \infty
\end{pmatrix}
$$

$$
\begin{aligned}
\alpha &= 6\left\lfloor\frac{m_{r_{11}}(0)+m_{r_{12}}(0)}{2}\right\rfloor - 6\left\lfloor\frac{m_{r_{12}}(0)}{2}\right\rfloor \\
\gamma &= 6\left\lfloor\frac{m_{r_{11}}(0)+m_{p_2}(0)}{2}\right\rfloor - 6\left\lfloor\frac{m_{p_2}(0)}{2}\right\rfloor \\
\eta &= 6\left\lfloor\frac{m_{p_{11}}(0)+m_{p_9}(0)}{3}\right\rfloor - 6\left\lfloor\frac{m_{p_{11}}(0)}{3}\right\rfloor \\
\mu &= 6\left\lfloor\frac{m_{p_{11}}(0)+m_{p_9}(0)+m_{p_7}(0)}{3}\right\rfloor - 6\left\lfloor\frac{m_{p_{11}}(0)+m_{p_9}(0)}{3}\right\rfloor
\end{aligned}
\qquad
\begin{aligned}
\beta &= 6\left\lfloor\frac{m_{p_1}(0)+m_{r_{12}}(0)}{2}\right\rfloor - 6\left\lfloor\frac{m_{r_{12}}(0)}{2}\right\rfloor \\
\delta &= 6\left\lfloor\frac{m_{p_1}(0)+m_{p_2}(0)}{2}\right\rfloor - 6\left\lfloor\frac{m_{p_2}(0)}{2}\right\rfloor \\
\lambda &= 6\left\lfloor\frac{m_{p_{12}}(0)+m_{p_{10}}(0)}{2}\right\rfloor - 6\left\lfloor\frac{m_{p_{12}}(0)}{2}\right\rfloor \\
\nu &= 6\left\lfloor\frac{m_{p_{12}}(0)+m_{p_{10}}(0)+m_{p_8}(0)}{2}\right\rfloor - 6\left\lfloor\frac{m_{p_{12}}(0)+m_{p_{10}}(0)}{2}\right\rfloor
\end{aligned}
$$

FIG. 3.10 – *Matrice B(M(0))*

3.6 Conclusion et perspectives

Dans ce chapitre, nous avons exposé une méthode visant à l'étude du comportement logique d'un GEP. Cette étude est motivée par l'obtention des marquages initiaux engendrant un comportement spécifique du réseau de Petri sous-jacent (séquencement des ressources et un régime cyclique).

La méthode consiste à abstraire le comportement du GEP considéré en un système d'équations, dont les variables mémorisent les franchissements des transitions. En utilisant une structure algébrique inspirée des dioïdes nommément $(\min, \mathrm{div})$ et malgré la présence des divisions euclidiennes caractérisant la présence de pondération dans le réseau de Petri. Nous avons montré la pertinence de ce formalisme pour une étude du comportement logique et que pour une large classe de GEP nommée ici GEPL, il est possible de linéariser le système d'équations en un système $(\min, +)$-linéaire.

Cette technique permet de considérer l'ensemble des séquencements possibles au sein du GEP, ainsi que l'obtention d'une abstraction du comportement sous forme d'une équation récurrente. Celle-ci nous permettra de déduire les marquage initiaux correspondant au comportement cyclique en un nombre de steps spécifiés.

Les perspectives de ces travaux consistent à expliciter une condition nécessaire et suffisante pour qu'un GEP soit linéarisable et ainsi pouvoir définir la classe des GEPL dans leur globalité.

Troisième partie

Applications

Chapitre 4

Applications

4.1 Introduction

Lors des études menées, dans ce mémoire, au sein des chapitres 2 et 3, nous avons constaté que :

Dans un premier temps, un réseau de Petri, dont les indéterminismes sont liés aux partages de ressources, peut être représenté à l'aide d'un graphe d'événements pondéré auquel est adjoint un système d'équations régissant le marquage d'un sous-ensemble de places. Le réseau de Petri obtenu est déterministe et l'ensemble des scenarii possibles est caractérisé par l'ensemble des solutions du système d'équations associé.

Dans un second temps, nous avons démontré la possibilité de réaliser une abstraction du comportement logique en un système d'équations. À priori, celui-ci est non-linéaire, mais une technique de linéarisation permet d'exhiber un système d'équations $(\min,+)$-linéaires. L'ensemble des solutions du système caractérise un ensemble de marquages initiaux et ainsi à une solution est lui associé un des comportements logiques possible.

L'objet de ce chapitre est de présenter quelques applications de cette théorie en vue d'en illustrer les possibilités d'application. Dans ce sens, la première partie est consacrée à une application résultant de la linéarisation du système d'équations. Le système $(\min,+)$-linéaire peut être traduit en un graphe d'événements simple. Au sein de la seconde partie, nous exhiberons des exemples d'applications tirés de la production manufacturière. Puis une application à l'évaluation de performances sera proposée.

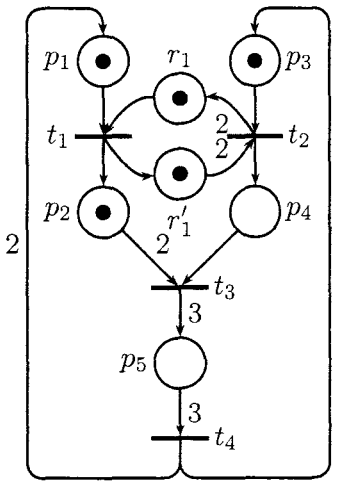


FIG. 4.1 – *Exemple de graphes d'événements*

4.2 GES équivalent

Nous proposons l'étude de l'exemple illustratif représenté par la figure 4.1. Au sein du chapitre 3, une nouvelle structure algébrique (min , div) a permis d'exhiber le système d'équations (4.1) dont les solutions capturent le comportement logique du graphe d'événements sous jacent. À l'aide d'une technique de linéarisation, nous rejoignons le formalisme (min ,+) développé pour les graphes d'événements simple.

Pour l'exemple illustratif représenté par la figure 4.1, le système suivant abstrait le comportement logique en utilisant le formalisme (min , div) :

$$\begin{cases} n_1(k) = \min\left(m_{p_1}(0) + 2n_4(k-1), m_{r_1}(0) + 2n_2(k-1)\right) \\ n_2(k) = \min\left(m_{p_3}(0) + n_4(k-1), \left\lfloor \dfrac{m_{r'_1}(0) + n_1(k-1)}{2} \right\rfloor\right) \\ n_3(k) = \min\left(\left\lfloor \dfrac{m_{p_2}(0) + n_1(k-1)}{2} \right\rfloor, m_{p_4}(0) + n_2(k-1)\right) \\ n_4(k) = \left\lfloor \dfrac{m_{p_5}(0)}{3} \right\rfloor + n_3(k-1) \end{cases} \tag{4.1}$$

Le processus de linéarisation, développée au chapitre précédent, a permis l'obtention d'un système (min ,+)-linéaire associé au système d'équations (4.1) par l'ajout de deux composantes supplémentaires ($y_5(k)$ et $y_6(k)$) ci-dessous :

$$\begin{cases} y_1(k) = \min\left(m_{p_1}(0) + y_4(k-1), m_{r_1}(0) + y_2(k-1)\right) \\ y_2(k) = \min\left(2m_{p_3}(0) + y_4(k-1), y_5(k-1) + 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor\right) \\ y_3(k) = \min\left(2m_{p_4}(0) + y_2(k-1), y_6(k-1) + 2\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor\right) \\ y_4(k) = y_3(k-1) + 2\left\lfloor \frac{m_{p_5}(0)}{3} \right\rfloor \\ y_5(k) = \min\left(\alpha + y_2(k-1), \beta + y_4(k-1)\right) \\ y_6(k) = \min\left(\gamma + y_2(k-1), \delta + y_4(k-1)\right) \end{cases} \tag{4.2}$$

Où les constantes α, β, γ et δ sont définies comme suit :

$$\begin{aligned} \alpha &= 2\left\lfloor \frac{m_{r_1}(0) + m_{r_1'}(0)}{2} \right\rfloor - 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor \\ \beta &= 2\left\lfloor \frac{m_{p_1}(0) + m_{r_1'}(0)}{2} \right\rfloor - 2\left\lfloor \frac{m_{r_1'}(0)}{2} \right\rfloor \\ \gamma &= 2\left\lfloor \frac{m_{r_1}(0) + m_{p_2}(0)}{2} \right\rfloor - 2\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor \\ \delta &= 2\left\lfloor \frac{m_{p_1}(0) + m_{p_2}(0)}{2} \right\rfloor - 2\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor \end{aligned}$$

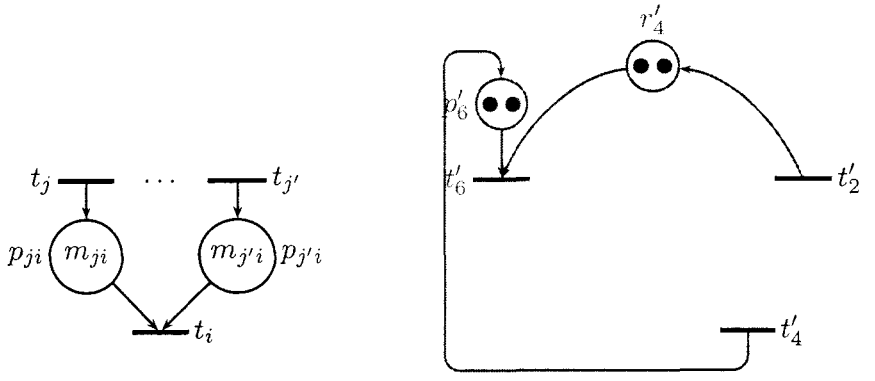
4.2.1 Technique de construction

Tout d'abord, étudions la forme générique des équations (min ,+)-linéaires données ci-dessous :

$$x_i(k) = \min_{j \in J_i} \{x_j(k-1) + m_{ji}\} \tag{4.3}$$

J_i représente l'ensemble des indices associé au compteur $x_i(k)$ et m_{ji} la constante associée au compteur x_j. Les entités J_i et m_{ji} peuvent être interprétée respectivement comme l'ensemble des indices des transitions en amont de t_i et m_{ji} comme le marquage de la place intermédiaire entre la transition t_i et t_j.

La construction consiste en la traduction du système d'équations (min ,+)-linéaires en un graphe d'événements simple associé. Une équation générique d'un tel système est donnée par l'équation (4.3). En utilisant le fait que les variables $x_j(k-1)$ de l'équation (4.3) caractérisent les transitions en amont de t_i et m_{ji} le marquage de l'unique place, du graphe



(a) modèle générique

(b) Application à la transition t_6 de l'exemple

FIG. 4.2 – *Construction*

d'événements, intermédiaire entre les transitions t_i et t_j. Pour reconstruire le graphe d'événements simple associé, on procède comme suit :

Pour chaque terme $x_j(k-1)+m_{ji}$, on crée la transition t_j en amont de la transition t_i, ainsi qu'une place intermédiaire contenant m_{ji} jetons. Pour l'équation (4.3) générique, la technique de construction est représentée par la figure 4.2(a).

À titre d'exemple, nous considérons la composante suivante :

$$y_6(k) = \min(\gamma + y_2(k-1), \delta + y_4(k-1))$$

Pour rejoindre le formalisme de l'équation (4.3), nous obtenons $J_i = \{2,4\}$. Par conséquent, les transitions t_2 et t_4 sont construites ainsi que deux places intermédiaires r'_4 et p'_6 dont le marquage initial est : $m_{26} = 2$ et $m_{46} = 2$. Cette construction permet l'obtention du réseau de Petri représenté par la figure 4.2(b).

Cette technique est appliquée au système d'équations (4.2) $(\min, +)$-linéaires. Nous obtenons le graphe d'événements simple associé représenté par la figure 4.3.

Le graphe d'événements pondéré avec les contraintes de fonctionnement initiales et le graphe d'événements simple possèdent le même système d'équations $(\min, +)$-linéaires associé. Après l'application de l'algorithme de linéarisation pour le GEP et dû à la construction précédente pour le GES. Nous en déduisons que les deux réseaux de Petri possèdent le même comportement. Aussi le graphe d'événements simple obtenu peut être considéré comme le représentant de tous les réseaux de Petri dont l'abstraction du comportement logique et les

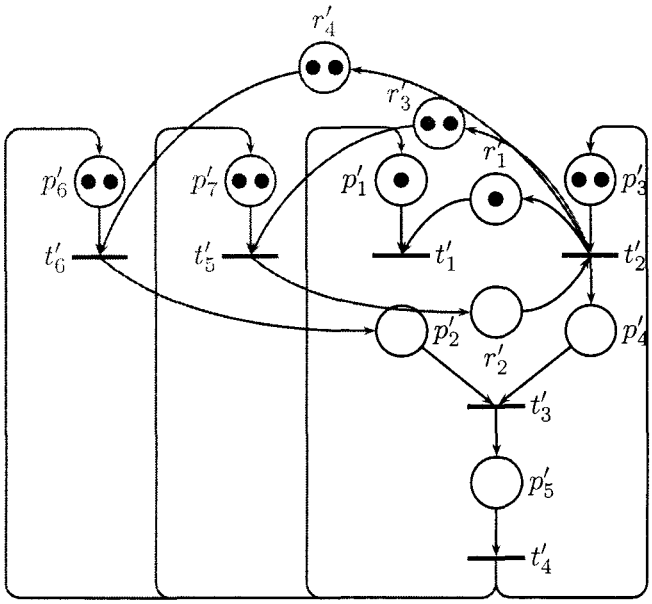


FIG. 4.3 – *Graphe d'événements linéarisé*

contraintes de fonctionnement conduisent au même système d'équations $(\min, +)$-linéaires donné par le système (4.2) après application du processus de linéarisation.

4.3 Exemples illustratifs

Dans le but d'exhiber la classe d'applications potentielles de la théorie exposée dans ce mémoire, nous nous intéressons à un exemple issu de la thèse d'O. Fournier [Fou02]. Pour décrire cet exemple, le tableau (4.3) représente la description des tâches à réaliser pour l'obtention des produits de sortie. Chaque colonne donne une information spécifique sur le modèle représenté par la figure 4.4, nous définissons :

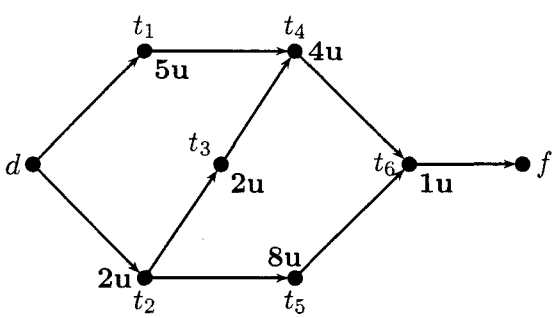


FIG. 4.4 – *Exemple illustratif*

Descriptif des tâches			
Nature	Nom	Durée	Ressource
Transformation	T_1	5	R_1
Désassemblage	T_2	2	R_2
Transformation	T_3	2	R_2
Assemblage	T_4	4	R_1
Transformation	T_5	8	R_3
Assemblage	T_6	1	R_2

- La colonne *Nature* désigne le type de l'opération à effectuer
- La colonne *Nom* l'identifiant de la tâche sur la figure 4.4
- La colonne *Durée* le temps requit pour l'exécution de l'opération
- La colonne *Ressource* désigne la ressource nécessaire à l'exécution de l'opération

Cette étude à pour but de déduire tous les séquencements possibles des ressources en présence pour éviter les blocages potentiels du système de production. Le système de production décrit précédemment peut-être représenté à l'aide d'un réseau de Petri, le réseau obtenu est représenté par la figure 4.5. Les ressources R_1, R_2 et R_3 induisent des conflits d'accès aux ressources qu'il convient de résoudre pour envisager l'obtention des séquencements possibles.

Dans ce sens, nous utilisons la méthode développée dans le chapitre 2 pour transformer le réseau de Petri de la figure 4.5 en un graphe d'événements associé à un système d'équations contraignant le marquage initial du graphe. Le hypothèses de production de ce modèle indiquent un seul franchissement pour chaque transition par cycle de production.

Le fait que le réseau de Petri décrit par la figure 4.5 soit non pondéré implique que la transformation va engendrer un graphe d'événements simple. Cette transformation consiste à effectuer les étapes suivantes :

- Suppression des places contenant les ressources (places R_1, R_2)
- Introduction d'une paire de places p_{ij} et p_{ji} joignant respectivement une paire de transitions t_i à t_j et t_j à t_i en conflit d'accès à la même ressource.

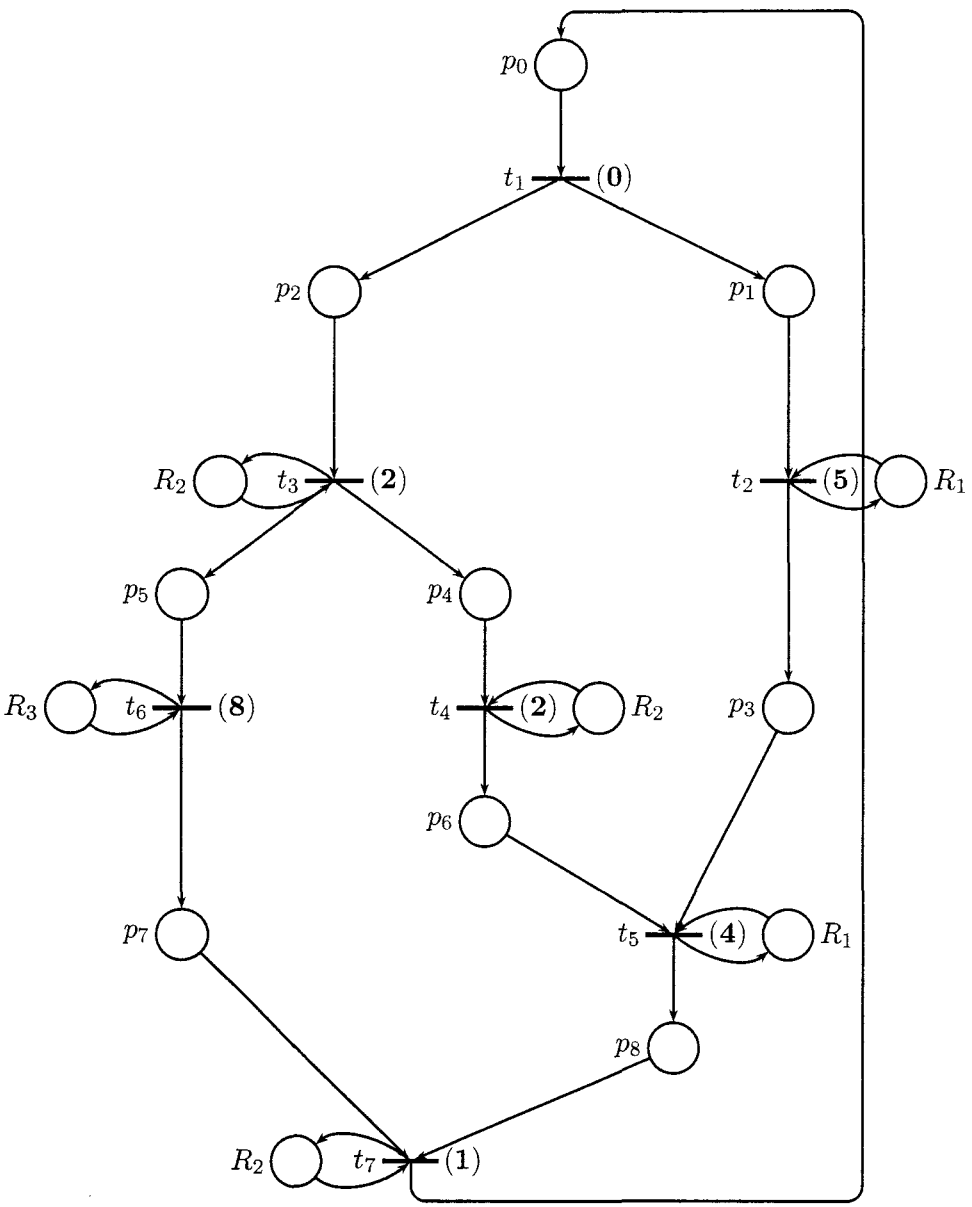


FIG. 4.5 – *Exemple illustratif*

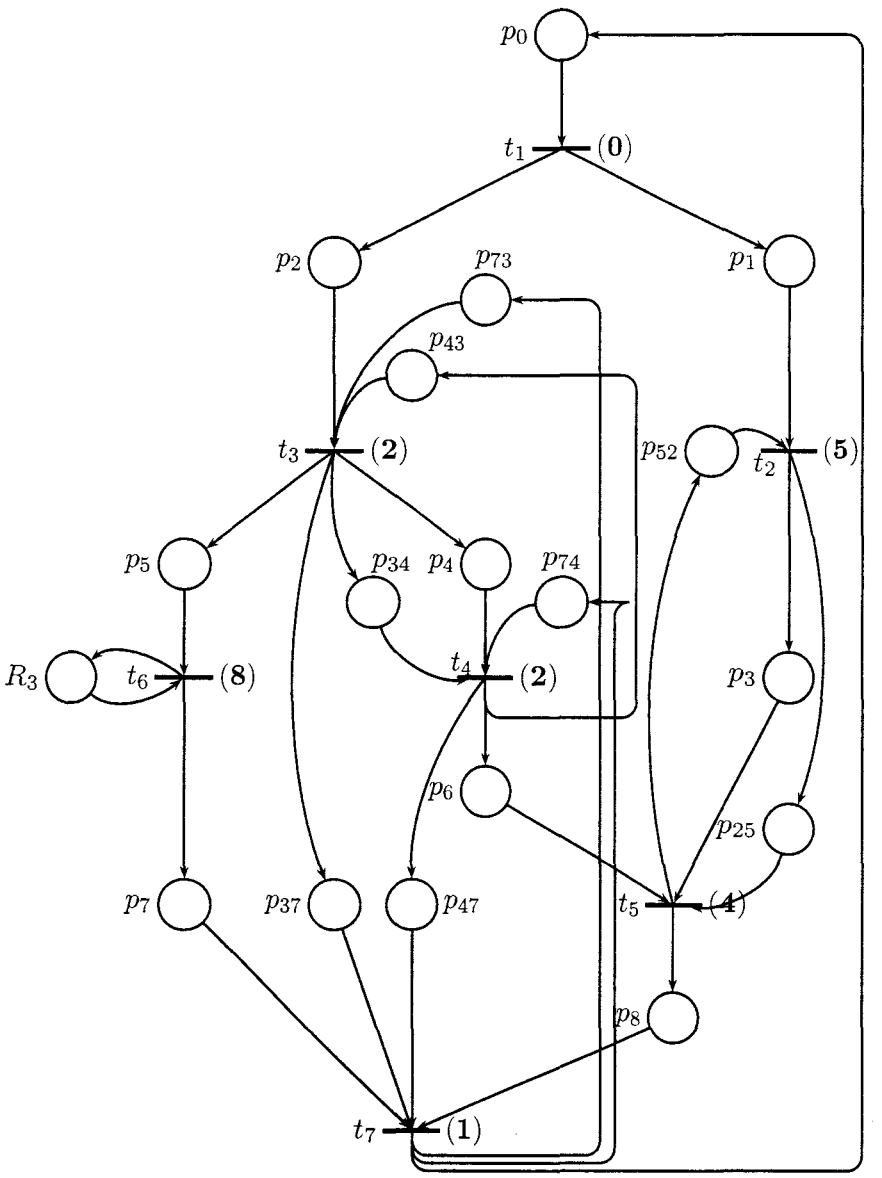


FIG. 4.6 – *Résolution des partages de ressources*

De plus, le marquage des places introduites sera contraint par un systèmes d'équations caractérisant l'ensemble des séquencements possibles des ressources partagées. Le résultat de cette transformation est décrit par la figure 4.6 auquel est adjoint un ensemble de contraintes sur le marquage des places ajoutées. Cet ensemble de contraintes est écrit sous forme de systèmes d'équations (cas d'une seule machine) traduisant les contraintes de précédence entre les transitions et la vivacité du marquage obtenu. Ces systèmes sont décrit ci-dessous :

$$\text{Contraintes relatives à la ressource } R_1 : m_{p_{52}} + m_{p_{25}} = 1 \tag{4.4}$$

$$\text{Contraintes relatives à la ressource } R_2 \begin{cases} m_{p_{43}} + m_{p_{34}} = 1 \\ m_{p_{37}} + m_{p_{73}} = 1 \\ m_{p_{47}} + m_{p_{74}} = 1 \end{cases} \tag{4.5}$$

$$\text{Contraintes relatives à la ressource } R_3 : m_{R_3} = 1 \tag{4.6}$$

La ressource R_3 n'induit aucune transformation structurelle car elle n'est pas source d'indéterminisme dans le réseau de Petri initial. Il est maintenant possible de faire une étude du comportement logique du graphe d'événements simple en utilisant le dioïde $(\min, +)$ pour abstraire les nombres d'occurrences de chaque transition en un système d'équations $(\min, +)$-linéaires auquel seront adjoint les systèmes d'équations (4.4), (4.5) et (4.6).

Par conséquent, nous allons décrire la relation récurrente que vérifie les composantes du vecteur d'occurrences $N(k+1)$, c'est-à-dire après le franchissement du $(k+1)^{ème}$ step, en fonction des composantes du vecteur $N(k)$. Cette abstraction du comportement est effectuée sur la base d'un marquage générique $M = (m_p)_{p \in P}$. Nous obtenons le système d'équations suivant :

$$\begin{cases} n_1(k+1) = n_7(k) + m_{p_0} \\ n_2(k+1) = \min\{n_1(k) + m_{p_1}, n_5(k) + m_{p_{52}}\} \\ n_3(k+1) = \min\{n_1(k) + m_{p_2}, n_4(k) + m_{p_{43}}, n_7(k) + m_{p_{73}}\} \\ n_4(k+1) = \min\{n_3(k) + m_{p_4}, n_3(k) + m_{p_{34}}, n_7(k) + m_{p_{74}}\} \\ n_5(k+1) = \min\{n_2(k) + m_{p_3}, n_2(k) + m_{p_{25}}, n_4(k) + m_{p_6}\} \\ n_6(k+1) = \min\{n_3(k) + m_{p_5}, n_6(k) + m_{R_3}\} \\ n_7(k+1) = \min\{n_3(k) + m_{p_{37}}, n_4(k) + m_{p_{47}}, n_5(k) + m_{p_8}, n_6(k) + m_{p_7}\} \end{cases} \tag{4.7}$$

Le système d'équations (4.7) permet d'évaluer le comportement logique du réseau de Petri en fonction du marquage initial, c'est-à-dire pour un marquage initial donné les nombre d'occurrences successifs sont solution du système (4.7). De plus, une équation récurrente matricielle sur le nombre de steps franchis peut être considérée. Ce qui se traduit par l'équation suivante :

$$N(k+1) = B(M(0)) \otimes N(k) \tag{4.8}$$

Où la matrice $B(M(0))$ est la matrice caractéristique du système d'équations (4.7) et l'opérateur $\otimes$ représente le formalisme $(\min, +)$ donnée ci-dessous :

$$\begin{pmatrix} +\infty & +\infty & +\infty & +\infty & +\infty & +\infty & m_{p_0} \\ m_{p_1} & +\infty & +\infty & +\infty & m_{p_{52}} & +\infty & +\infty \\ m_{p_2} & +\infty & +\infty & m_{p_{43}} & +\infty & +\infty & m_{p_{73}} \\ +\infty & +\infty & \min\{m_{p_4}, m_{p_{34}}\} & +\infty & +\infty & +\infty & m_{p_{74}} \\ +\infty & \min\{m_{p_3}, m_{p_{25}}\} & +\infty & m_{p_6} & +\infty & +\infty & +\infty \\ +\infty & +\infty & m_{p_5} & +\infty & +\infty & m_{R_3} & +\infty \\ +\infty & +\infty & m_{p_{37}} & m_{p_{47}} & m_{p_8} & m_{p_7} & +\infty \end{pmatrix}$$

Ce résultat permet de passer à une phase de résolution dans le but de trouver les séquencements possibles. Pour effectuer cette résolution, l'équation récurrente (4.8) est utilisée lors de la résolution avec une inconnue k correspondant au nombre de steps conduisant à un régime cyclique. Nous en déduisons l'équation suivante :

$$B(M(0))^k \otimes \varepsilon = \mathbb{1} \tag{4.9}$$

Il reste à déterminer une solution de ces systèmes. Pour illustrer notre propos, nous allons exhiber un marquage initial conduisant à un régime cyclique en trois steps. Pour effectuer cette résolution, nous devons calculer la matrice $B(M(0))$ à la puissance trois au sens de la loi $\otimes$. Ce qui correspond à l'équation (4.9) pour le cas $k = 3$ à laquelle est adjoint les systèmes d'équations (4.4), (4.5) et (4.6). Après résolution de ce système, une des solutions possibles est définie par le marquage initial défini par les équations ci-dessous :

- $m_{p_0} = m_{p_3} = m_{p_4} = m_{p_7} = m_{p_{25}} = m_{p_{73}} = m_{p_{34}} = m_{p_{74}} = m_{R_3} = 1$
- $m_{p_1} = m_{p_2} = m_{p_6} = m_{p_5} = m_{p_8} = m_{p_{52}} = m_{p_{43}} = m_{p_{37}} = m_{p_{47}} = 0$

Cette solution induit un comportement cyclique en une séquence de trois steps décrite ci-dessous :

1. $t_1 \,/\!/\, t_4$
2. $t_3 \,/\!/\, t_5$
3. $t_2 \,/\!/\, t_6 \,/\!/\, t_7$

Nous venons d'exposer une application dans le domaine des système de production manufacturière. Le modèle est représenté à l'aide du formalisme réseau de Petri, puis le réseaux de Petri est transformé en un graphe d'événements simple dont le marquage d'un sous-ensemble de places est contraint par un système d'équations additionnel (formé des systèmes (4.4), (4.5) et (4.6)). Par conséquent, une étude du comportement logique est possible en utilisant un formalisme dédié : le dioïde $(\min, +)$ dont l'utilité est d'abstraire le comportement du graphe d'événements simple en un système d'équations $(\min, +)$-linéaires. Lors du calcul final des séquencements possibles, il reste à introduire le nombre d'étapes nécessaire pour un fonctionnement cyclique du modèle.

4.4 Évaluation de performance

L'application exposée dans ce paragraphe à pour but d'évaluer les performances d'un atelier de production manufacturière dont une modélisation préalable à été exécutée à l'aide des réseaux de Petri. L'objectif est de calculer les séquencements possibles avec le respect d'un ensemble de contraintes décrite par la suite. Puis de calculer le temps de cycle obtenu pour chaque solution.

Dans ce sens cette approche se décompose en plusieurs étapes. Tout d'abord, l'abstraction du comportement logique du modèle en un système d'équations utilisant le formalisme développé au chapitre 3. Puis une linéarisation du système obtenu est effectuée, le but est l'obtention d'un système (min ,+)-linéaire pour ensuite permettre une résolution du système obtenu. Pour enfin effectuer une évaluation des performances des solutions obtenues en utilisant le formalisme (max ,+) [Gau95a].

4.4.1 Équations du réseau

En considérant le graphe d'événements représenté par la figure 4.7, nous déterminons les équations relatives aux occurrences des franchissements des transitions à l'aide de la structure algébrique introduite précédemment. Il vient le système d'équations suivant :

$$
\begin{cases}
n_1(k) = \min\left(m_{p_1}(0) + 2n_4(k-1), m_{r_{11}}(0) + 2n_2(k-1)\right) \\
n_2(k) = \min\left(m_{p_3}(0) + n_4(k-1), \left\lfloor \dfrac{m_{r_{12}}(0) + n_1(k-1)}{2} \right\rfloor\right) \\
n_3(k) = \min\left(\left\lfloor \dfrac{m_{p_2}(0) + n_1(k-1)}{2} \right\rfloor, m_{p_4}(0) + n_2(k-1), \right. \\
\qquad\qquad \left. m_{r_{22}}(0) + n_5(k-1)\right) \\
n_4(k) = \min\left(\left\lfloor \dfrac{m_{p_5}(0)}{3} \right\rfloor + n_3(k-1), m_{r_{32}}(0) + n_{10}(k-1)\right) \\
n_5(k) = \min\left(m_{r_{21}}(0) + n_3(k-1), \left\lfloor \dfrac{m_{p_6}(0)}{5} \right\rfloor + n_{10}(k-1)\right) \\
n_6(k) = m_{p_7}(0) + 3n_5(k-1) \\
n_7(k) = m_{p_8}(0) + 2n_5(k-1) \\
n_8(k) = m_{p_9}(0) + n_6(k-1) \\
n_9(k) = m_{p_{10}}(0) + n_7(k-1) \\
n_{10}(k) = \min\left(m_{r_{31}}(0) + n_4(k-1), \left\lfloor \dfrac{m_{p_{11}}(0) + n_8(k-1)}{3} \right\rfloor, \right. \\
\qquad\qquad \left. \left\lfloor \dfrac{m_{p_{12}}(0) + n_9(k-1)}{2} \right\rfloor\right)
\end{cases}
\tag{4.10}
$$

Comme nous l'avons constaté précédemment, le système ainsi obtenu ne peut engendrer une équation récurrente (min ,+)-linéaire en raison de la présence de parties entières rendant les équations non linéaires.

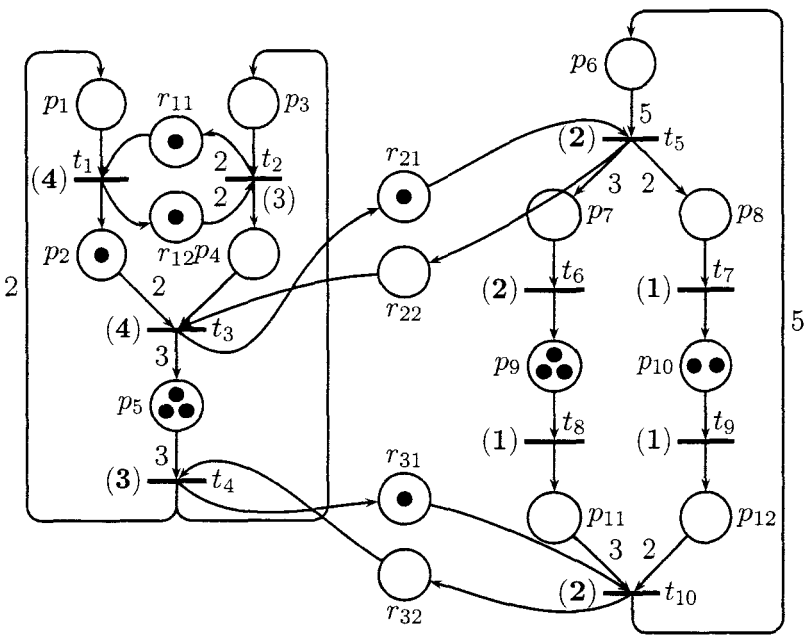


FIG. 4.7 – *Graphes d'événements pondéré*

4.4.2 Linéarisation du système

Description de l'exemple

La chaîne de production modélisée par la figure 4.7 permet de fabriquer deux produits, A et B, à l'aide de trois types de ressources partagées (représentées par les places r_{11}, r_{12}, r_{21} r_{22}, r_{31} et r_{32}). La ressource r_1 existe en deux exemplaires, les deux autres en un exemplaire.

Le produit A est composé de deux pièces A_1 et d'une pièce A_2. Le premier composant est fabriqué par une machine de type r_1 (transition t_1). La fabrication du deuxième composant nécessite les deux exemplaires de la ressource r_1. La machine de type r_2 effectue alors l'assemblage de deux pièces A_1 et d'une pièce A_2 (transition t_3). Enfin, le résultat de l'assemblage est usiné sur la machine de type r_3 (transition t_4) pour obtenir le produit final. Afin de tenir compte de la durée des tâches, le réseau de Petri est temporisé au niveau des transitions : ainsi, par exemple la durée de l'opération associée à la transition t_1 est de quatre unités de temps.

Le deuxième produit B est traité par lot de cinq pièces. La première opération est effectuée sur un lot par la machine de type r_2 (transition t_5). Ensuite le lot est désolidarisé

pour former un ensemble de trois pièces (place p_7) et un autre de deux pièces (place p_8). Chacun de ces ensembles est traité séparément (transitions t_6 et t_8 d'une part et transitions t_7 et t_9 d'autre part). Un lot de cinq pièces est de reconstitué pour être usiné sur la machine de type r_3 (transition t_{10}).

Afin de caractériser le comportement cyclique du système, le modèle est rebouclé (arcs joignant la transition t_4 aux places p_1 et p_3, et la transition t_{10} à la place p_6).

Abstraction du modèle

Considérons la figure 4.7. On commence par rechercher l'invariant total de transitions θ. Cet invariant vérifie $\theta^T = \begin{pmatrix} 2 & 1 & 1 & 1 & 1 & 3 & 2 & 3 & 2 & 1 \end{pmatrix}^T$. Tous les circuits du réseau possèdent une transition de normalisation. L'exemple proposé appartient donc bien à la classe des GEPL et nous utiliserons la technique de linéarisation sur cet exemple à titre d'illustration. On considère donc le vecteur d'état suivant $X = \left(x_i\right)^T_{1 \leqslant i \leqslant 10}$ défini par $x_1 = 3n_1$, $x_6 = 2n_6$, $x_7 = 3n_7$, $x_8 = 2n_8$, $x_9 = 3n_9$, pour toutes les autres composantes $x_i = 6n_i$.

Le vecteur d'occurrences normalisé satisfait le système suivant :

$$\begin{cases} x_1(k) = \min\left(3m_{p_1}(0) + x_4(k-1), 3m_{r_{11}}(0) + x_2(k-1)\right) \\ x_2(k) = \min\left(6m_{p_3}(0) + x_4(k-1), 6\left\lfloor \dfrac{3m_{r_{12}}(0) + x_1(k-1)}{6} \right\rfloor\right) \\ x_3(k) = \min\left(6\left\lfloor \dfrac{3m_{p_2}(0) + x_1(k-1)}{6} \right\rfloor, 6m_{p_4}(0) + x_2(k-1),\right. \\ \qquad\qquad \left. 6m_{r_{22}}(0) + x_5(k-1)\right) \\ x_4(k) = \min\left(6\left\lfloor \dfrac{m_{p_5}(0)}{3} \right\rfloor + x_3(k-1), 6m_{r_{32}}(0) + x_{10}(k-1)\right) \\ x_5(k) = \min\left(6m_{r_{21}}(0) + x_3(k-1), 6\left\lfloor \dfrac{m_{p_6}(0)}{5} \right\rfloor + x_{10}(k-1)\right) \\ x_6(k) = 2m_{p_7}(0) + x_5(k-1) \\ x_7(k) = 3m_{p_8}(0) + x_5(k-1) \\ x_8(k) = 2m_{p_9}(0) + x_6(k-1) \\ x_9(k) = 3m_{p_{10}}(0) + x_7(k-1) \\ x_{10}(k) = \min\left(6m_{r_{31}}(0) + x_4(k-1), 6\left\lfloor \dfrac{2m_{p_{11}}(0) + x_8(k-1)}{6} \right\rfloor,\right. \\ \qquad\qquad \left. 6\left\lfloor \dfrac{3m_{p_{12}}(0) + x_9(k-1)}{6} \right\rfloor\right) \end{cases} \tag{4.11}$$

Parmi ces équations, la deuxième, la troisième et la dernière contiennent des ex-

pressions non linéaires. Pour linéariser ces équations, nous introduisons le vecteur d'état Y dont les dix premières composantes correspondent à celles du vecteur d'occurrences normalisé et nous ajoutons dans le vecteur d'état Y ces expressions non linéaires. Par exemple, il apparaît dans la deuxième équation l'expression non linéaire $6\left\lfloor \frac{3m_{r_{12}}(0)+y_1(k-1)}{6} \right\rfloor$. Nous ajoutons donc, dans le vecteur d'état Y, la composante $y_{11}(k)$ définie par :

$$y_{11}(k) = 6\left\lfloor \frac{3m_{r_{12}}(0)+y_1(k)}{6} \right\rfloor - 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor$$

La deuxième équation s'écrit alors :

$$y_2(k) = \min\left(6m_{p_3}(0)+y_4(k-1), y_{11}(k-1)+6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor\right)$$

Il reste alors à exprimer $y_{11}(k)$ en fonction des composantes de $Y(k-1)$. Pour cela, on exprime $y_1(k)$ en fonction de $Y(k-1)$ et on utilise le fait que la somme, la division, la partie entière et la multiplication sont distributives par apport à l'opérateur div. Il vient alors :

$$\begin{aligned} y_{11}(k) &= 6\left\lfloor \frac{3m_{r_{12}}(0)+\min\left(3m_{p_1}(0)+y_4(k-1), 3m_{r_{11}}(0)+y_2(k-1)\right)}{6} \right\rfloor \\ &\quad - 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor \\ &= \min\left(6\left\lfloor \frac{3m_{r_{12}}(0)+3m_{p_1}(0)}{6} + \frac{y_4(k-1)}{6} \right\rfloor, \right. \\ &\quad \left. 6\left\lfloor \frac{3m_{r_{12}}(0)+3m_{r_{11}}(0)}{6} + \frac{y_2(k-1)}{6} \right\rfloor\right) - 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor \end{aligned}$$

Or $y_2(k-1)$ et $y_4(k-1)$ sont, par définition, des multiples de 6. Les entiers $\frac{y_2(k-1)}{6}$ et $\frac{y_4(k-1)}{6}$ peuvent donc être sortis de la partie entière.

D'où finalement la forme linéaire suivante :

$$\begin{aligned} y_{11}(k) = \min\left(6\left\lfloor \frac{m_{r_{12}}(0)+m_{p_1}(0)}{2} \right\rfloor + y_4(k-1), 6\left\lfloor \frac{m_{r_{12}}(0)+m_{r_{11}}(0)}{2} \right\rfloor + y_2(k-1)\right) \\ -6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor \end{aligned}$$

L'expression non linéaire qui apparaît dans la troisième équation du système (4.12) se linéarise en introduisant la composante y_{12} définie par :

$$y_{12}(k) = 6\left\lfloor \frac{3m_{p_2}(0)+x_1(k)}{6} \right\rfloor - 6\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor$$

On obtient alors :

$$y_3(k) = \min\left(6\left\lfloor\frac{m_{p_2}(0)}{2}\right\rfloor + y_{12}(k-1), 6m_{p_4}(0) + y_2(k-1), 6m_{r_{22}}(0) + y_5(k-1)\right)$$
$$y_{12}(k) = \min\left(6\left\lfloor\frac{m_{p_2}(0) + m_{p_1}(0)}{2}\right\rfloor + y_4(k-1), 6\left\lfloor\frac{m_{p_2}(0) + m_{r_{11}}(0)}{2}\right\rfloor + y_2(k-1)\right) - 6\left\lfloor\frac{m_{p_2}(0)}{2}\right\rfloor$$

Pour linéariser la dernière équation du système (4.12), on introduit les composantes $y_{13}(k)$ et $y_{14}(k)$ définies par :

$$y_{13}(k) = 6\left\lfloor\frac{2m_{p_{11}}(0) + x_8(k)}{6}\right\rfloor - 6\left\lfloor\frac{m_{p_{11}}(0)}{3}\right\rfloor$$
$$y_{14}(k) = 6\left\lfloor\frac{3m_{p_{12}}(0) + x_9(k)}{6}\right\rfloor - 6\left\lfloor\frac{m_{p_{12}}(0)}{2}\right\rfloor$$

On a alors :

$$y_{10}(k) = \min\left(6m_{r_{31}}(0) + y_4(k-1), y_{13}(k-1) + 6y_{10}(k), y_{14}(k-1) + 6\left\lfloor\frac{m_{p_{12}}(0)}{2}\right\rfloor\right)$$
$$y_{13}(k) = 6\left\lfloor\frac{2m_{p_{11}}(0) + 2m_{p_9}(0) + x_6(k-1)}{6}\right\rfloor - 6\left\lfloor\frac{m_{p_{11}}(0)}{3}\right\rfloor$$
$$y_{14}(k) = 6\left\lfloor\frac{3m_{p_{12}}(0) + 3m_{p_{10}}(0) + x_7(k-1)}{6}\right\rfloor - 6\left\lfloor\frac{m_{p_{12}}(0)}{2}\right\rfloor$$

D'autres expressions non linéaires apparaissent. On introduit alors les composantes définies par :

$$y_{15}(k) = 6\left\lfloor\frac{2m_{p_{11}}(0) + 2m_{p_9}(0) + x_6(k)}{6}\right\rfloor - 6\left\lfloor\frac{m_{p_{11}}(0) + m_{p_9}(0)}{3}\right\rfloor$$
$$y_{16}(k) = 6\left\lfloor\frac{3m_{p_{12}}(0) + 3m_{p_{10}}(0) + x_7(k)}{6}\right\rfloor - 6\left\lfloor\frac{m_{p_{12}}(0) + m_{p_{10}}(0)}{2}\right\rfloor$$

Finalement, nous obtenons le système d'équations (min ,+)-linéaires suivant :

$$
\begin{cases}
y_1(k+1) = \min\left(3m_{p_1}(0) + y_4(k), 3m_{r_{11}}(0) + y_2(k)\right) \\
y_2(k+1) = \min\left(6m_{p_3}(0) + y_4(k), y_{11}(k) + 6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor\right) \\
y_3(k+1) = \min\left(6\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor + y_{12}(k), 6m_{p_4}(0) + y_2(k), 6m_{r_{22}}(0) + y_5(k)\right) \\
y_4(k+1) = \min\left(2\left\lfloor \frac{m_{p_5}(0)}{3} \right\rfloor + y_3(k), 2m_{r_{32}}(0) + y_{10}(k)\right) \\
y_5(k+1) = \min\left(2\left\lfloor \frac{m_{p_6}(0)}{5} \right\rfloor + y_{10}(k), 2m_{r_{21}}(0) + y_3(k)\right) \\
y_6(k+1) = 2\left\lfloor \frac{m_{p_7}(0)}{5} \right\rfloor + y_5(k) \\
y_7(k+1) = 2\left\lfloor \frac{m_{p_7}(0)}{5} \right\rfloor + y_5(k) \\
y_8(k+1) = 2m_{p_8}(0) + y_6(k) \\
y_9(k+1) = 2m_{p_9}(0) + y_7(k) \\
y_{10}(k+1) = \min\left(6m_{r_{31}}(0) + y_4(k), y_{13}(k) + 6y_{10}(k), y_{14}(k) + 6\left\lfloor \frac{m_{p_{12}}(0)}{2} \right\rfloor\right) \\
y_{11}(k+1) = \min\left(6\left\lfloor \frac{m_{r_{12}}(0) + m_{p_1}(0)}{2} \right\rfloor + y_4(k), 6\left\lfloor \frac{m_{r_{12}}(0) + m_{r_{11}}(0)}{2} \right\rfloor + y_2(k)\right) \\
\qquad\qquad -6\left\lfloor \frac{m_{r_{12}}(0)}{2} \right\rfloor \\
y_{12}(k+1) = \min\left(6\left\lfloor \frac{m_{p_2}(0) + m_{p_1}(0)}{2} \right\rfloor + y_4(k), 6\left\lfloor \frac{m_{p_2}(0) + m_{r_{11}}(0)}{2} \right\rfloor + y_2(k)\right) \\
\qquad\qquad -6\left\lfloor \frac{m_{p_2}(0)}{2} \right\rfloor \\
y_{13}(k+1) = y_{15}(k) + 6\left\lfloor \frac{m_{p_{11}}(0) + m_{p_9}(0)}{3} \right\rfloor - 6\left\lfloor \frac{m_{p_{11}}(0)}{3} \right\rfloor \\
y_{14}(k+1) = y_{16}(k) + 6\left\lfloor \frac{m_{p_{12}}(0) + m_{p_{10}}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{p_{12}}(0)}{2} \right\rfloor \\
y_{15}(k+1) = y_5(k) + 6\left\lfloor \frac{m_{p_{11}}(0) + m_{p_9}(0) + m_{p_7}(0)}{3} \right\rfloor - 6\left\lfloor \frac{m_{p_{11}}(0) + m_{p_9}(0)}{3} \right\rfloor \\
y_{16}(k+1) = y_5(k) + 6\left\lfloor \frac{m_{p_{12}}(0) + m_{p_{10}}(0) + m_{p_8}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{p_{12}}(0) + m_{p_{10}}(0)}{2} \right\rfloor
\end{cases}
\tag{4.12}
$$

Nous obtenons finalement, le système $(\min,+)$-linéaire de dimension 16 augmenté par rapport au système (3.12) (de dimension 10 comportant quatre composantes non-linéaires). Le système précédent peut alors être décrit sous forme matricielle (l'opérateur $\otimes$ dénote le produit matriciel au sens de $(\min,+)$), comme suit :

$$
Y(k) = B(M(0)) \otimes Y(k-1) \tag{4.13}
$$

où la matrice $B(M(0))$ est représentée figure 4.8.

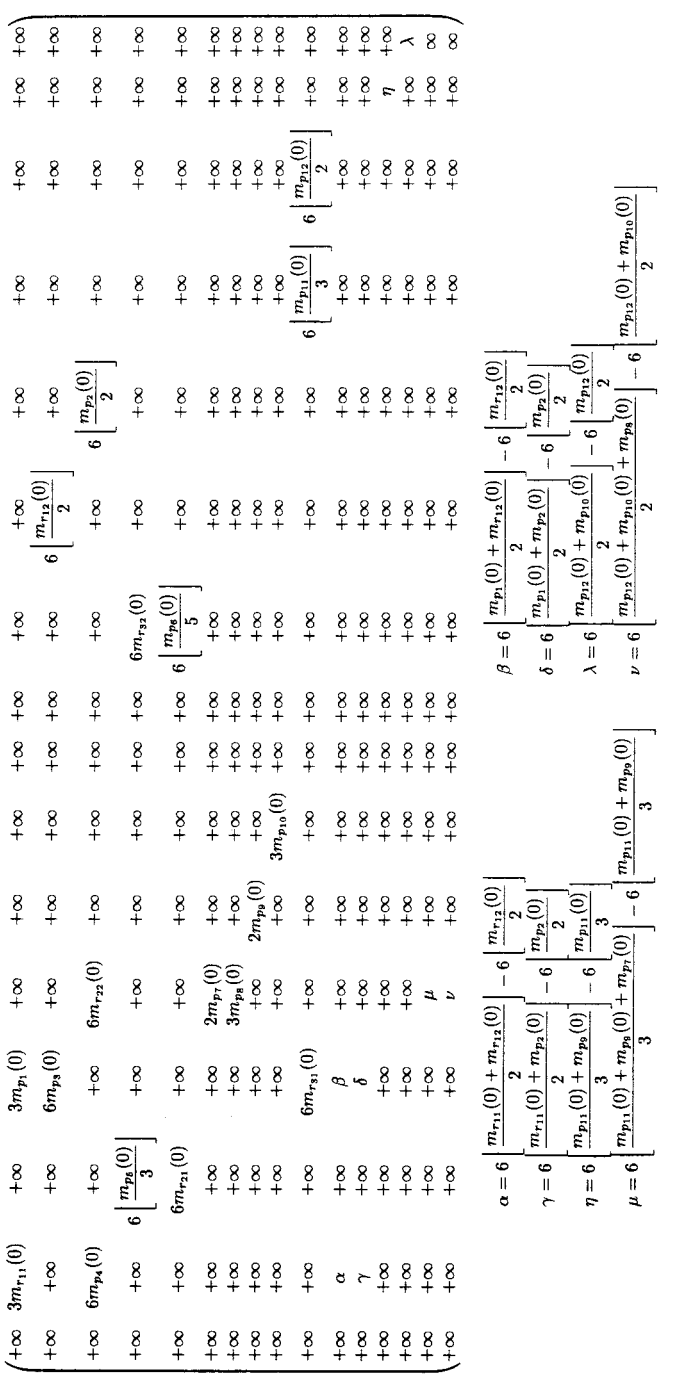
Équivalence comportementale

L'équation (4.13) est $(\min,+)$-linéaire. On peut alors facilement exhiber le GES dont le $k^{\text{ième}}$ vecteur d'occurrences correspond à $Y(k)$. Ainsi, pour le marquage initial décrit figure 3.1, on obtient alors le graphe d'événements simple représenté par la figure 4.9. Les composantes finies de la matrice $B(M(0))$ représentent le nombre de jetons présents dans une place située entre deux transitions ; pour notre exemple, le cœfficient α (ligne 11 colonne 2) traduit le fait que la place située entre les transitions t'_{11} et t'_2 contient α jeton(s), et un cœfficient infini indique qu'aucune place ne lie les transitions. Ainsi, la structure du réseau (places, transitions et arcs) est donnée par les composantes finies de la matrice $B(M(0))$. Or la finitude des composantes ne dépend que du marquage initial : si l'on considère deux marquages initiaux différents, les graphes d'événements simples correspondants ne différeront que par leur marquage initial, leur structure générale reste alors la même.

Il s'agit maintenant d'établir le lien entre le GEP et le GES résultant de la linéarisation. Ce lien est donné par la relation qui lie le vecteur d'occurrences au vecteur d'occurrences normalisé. Par exemple, nous avons, pour l'exemple considéré, $y_1 = 3n_1$: un tir de la transition t_1 (figure 3.1) correspond à trois tir de la transition t'_1 (figure 4.9). Ainsi, le comportement du GEP se déduit très facilement de celui du GES projeté sur les transitions $t'_1, t'_2, \ldots, t'_{10}$.

Vu l'invariant de transitions $\begin{pmatrix} 2 & 1 & 1 & 1 & 1 & 3 & 2 & 3 & 2 & 1 \end{pmatrix}^T$, le franchissement d'un cycle dans le GEP se traduit dans le GES par une augmentation de six unités des composantes du vecteur d'occurrences. En d'autres termes, chacune des transitions du GES est franchie exactement six fois lors d'un cycle.

Finalement, le comportement non linéaire du système initial du réseau pondéré est réduit au comportement linéaire d'un graphe d'événements simple dont la dimension a été augmentée par l'ajout de six composantes. Cette approche est similaire à celle proposée par Alix Munier [Mun93], mais elle se distingue par deux différences notables : d'une part, on autorise le franchissement multiple des transitions, d'autre part la structure du GES obtenu est indépendante du marquage initial.



$$\alpha = 6\left\lfloor \frac{m_{r11}(0)+m_{r12}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{r12}(0)}{2} \right\rfloor \qquad \beta = 6\left\lfloor \frac{m_{p1}(0)+m_{r12}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{r12}(0)}{2} \right\rfloor$$

$$\gamma = 6\left\lfloor \frac{m_{r11}(0)+m_{p2}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{p2}(0)}{2} \right\rfloor \qquad \delta = 6\left\lfloor \frac{m_{p1}(0)+m_{p2}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{p2}(0)}{2} \right\rfloor$$

$$\eta = 6\left\lfloor \frac{m_{p11}(0)+m_{p9}(0)}{3} \right\rfloor - 6\left\lfloor \frac{m_{p11}(0)}{3} \right\rfloor \qquad \lambda = 6\left\lfloor \frac{m_{p12}(0)+m_{p10}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{p12}(0)}{2} \right\rfloor$$

$$\mu = 6\left\lfloor \frac{m_{p11}(0)+m_{p8}(0)+m_{p7}(0)}{3} \right\rfloor - 6\left\lfloor \frac{m_{p11}(0)+m_{p9}(0)}{3} \right\rfloor \qquad \nu = 6\left\lfloor \frac{m_{p12}(0)+m_{p10}(0)+m_{p8}(0)}{2} \right\rfloor - 6\left\lfloor \frac{m_{p12}(0)+m_{p10}(0)}{2} \right\rfloor$$

FIG. 4.8 – *Matrice B(M(0))*

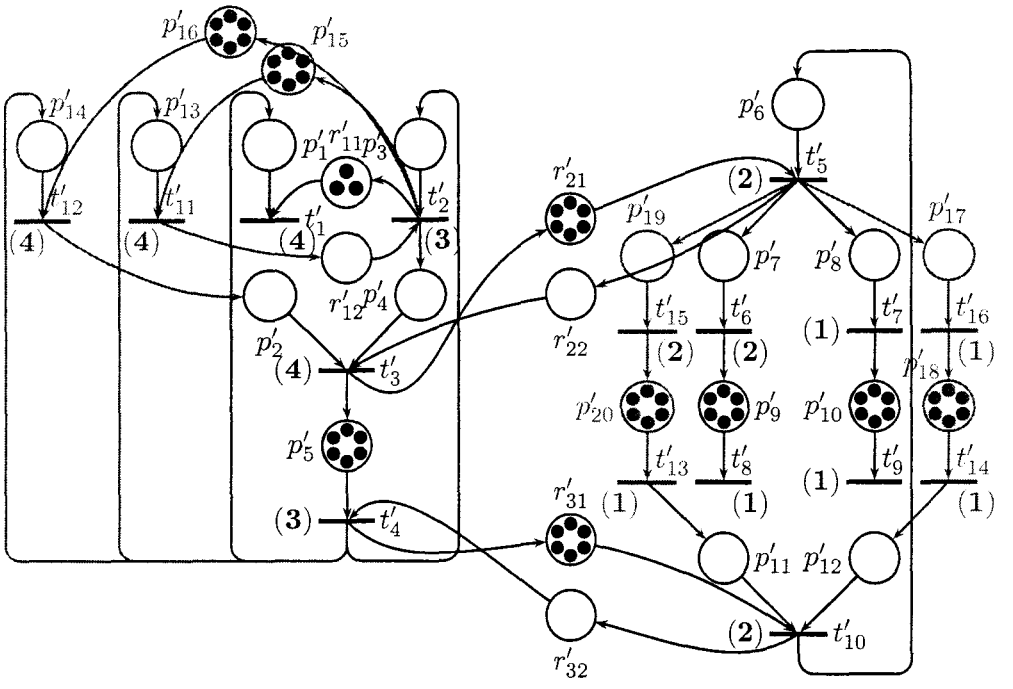


FIG. 4.9 – *Graphe d'événements linéarisé*

Formulation du problème

Dans l'équation (4.13), linéaire au sens de (min ,+), il apparaît bien un paramétrage direct de l'opérateur linéaire B par le marquage initial du graphe $M(0)$. Il est donc possible grâce à cette écriture de rechercher un marquage initial qui conduit à accéder à un comportement donné. Le cas particulier d'un régime cyclique en p steps est particulièrement intéressant. En effet, nous avons vu qu'après chaque cycle, les composantes du vecteur occurrences sont augmentées de six unités. La résolution de ce problème revient donc à rechercher le marquage initial qui satisfait l'équation $Y(p) = Y(0) + 6 \cdot \mathbb{1}$. Sachant que $Y(p) = (B(M(0)))^p \otimes Y(0)$, on obtient l'équation suivante :

$$(B(M(0)))^p \otimes \varepsilon = 6 \cdot \mathbb{1}$$

où $Y(0)$ est le vecteur nul et $\mathbb{1}$ le vecteur dont chaque composante est unitaire. Dans la suite, nous allons rechercher, à titre d'exemple, un marquage initial pour un fonctionnement périodique du réseau en deux steps. Ce choix est l'un des plus simples, d'autres valeurs de p peuvent évidemment être envisagées (Plus la valeur p est grande, plus le nombre d'en-cours

dans le système sera petit). On en vient finalement à résoudre l'équation suivante :

$$(B(M(0)))^2 \otimes \varepsilon = 6 \cdot \mathbb{1} \tag{4.14}$$

Celle-ci nous donne un système de seize équations à dix-huit inconnues, auquel il faut ajouter un jeu d'équations traduisant les contraintes de capacité sur les machines présentes dans l'atelier. Étant donné que nous disposons de deux machines de type r_1, une machine de type r_2 et r_3, nous cherchons à résoudre le système suivant :

$$\begin{cases} (B(M_0))^2 \otimes \varepsilon = 6 \cdot \mathbb{1} \\ m_{r_{11}} + m_{r_{12}} = 2 \\ m_{r_{21}} + m_{r_{22}} = 1 \\ m_{r_{31}} + m_{r_{32}} = 1 \end{cases}$$

Solutions du problème

Ces équations sont ensuite implantées dans le solveur de contraintes de *PrologIV*, sur un ordinateur possédant un processeur Céléron cadencé à 333 Mhz avec 64 Mo de RAM, le résultat est obtenu en neuf secondes.

Pour notre exemple, la résolution de ce système donne un ensemble de solutions dont chacune d'elle représente un marquage initial du réseau de la figure 2.1. Nous désirons des solutions qui minimisent les en-cours, c'est-à-dire le nombre de jetons dans le réseau. Nous ne nous intéressons donc qu'aux solutions *minimales* (M est une solution minimale s'il n'existe pas de solution M' vérifiant $M' \preceq M$ et $M' \neq M$). Il vient alors les solutions minimales candidates suivantes (elles sont notées à l'aide du vecteur caractéristique suivant $(m_{p_1}, m_{p_2}, \ldots, m_{p_{12}}, m_{r_{11}}, m_{r_{12}}, m_{r_{21}}, m_{r_{22}}, m_{r_{31}}, m_{r_{32}})$) :

- Δ_1 = (2,0,1,1,3,5,0,0,3,2,3,2,2,0,1,0,1,0)
- Δ'_1 = (0,2,1,1,3,5,3,2,0,0,3,2,0,2,0,1,0,1)
- Δ_2 = (2,0,0,1,3,5,0,0,3,2,0,0,2,0,1,0,0,1)
- Δ'_2 = (2,2,1,1,0,5,3,2,0,0,3,2,0,2,0,1,1,0)
- Δ_3 = (2,2,0,1,3,5,3,2,0,0,3,2,2,0,0,1,0,1)
- Δ'_3 = (2,2,1,0,3,5,0,0,3,2,3,2,0,2,1,0,1,0)
- Δ_4 = (2,2,1,1,0,0,3,2,0,0,3,2,2,0,0,1,1,0)

Marquage initial	Cycle		
Δ_1	$(2t_1 /\!/ t_5 /\!/ 3t_8 /\!/ 2t_9 /\!/ t_{10})$	+	$(t_2 /\!/ t_3 /\!/ t_4 /\!/ 3t_6 /\!/ 2t_7)$
Δ'_1	$(t_2 /\!/ t_3 /\!/ t_4 /\!/ 3t_6 /\!/ 2t_7)$	+	$(2t_1 /\!/ t_5 /\!/ 3t_8 /\!/ 2t_9 /\!/ t_{10})$
Δ_2	$(2t_1 /\!/ t_4 /\!/ t_5 /\!/ 3t_8 /\!/ 2t_9)$	+	$(t_2 /\!/ t_3 /\!/ 3t_6 /\!/ 2t_7 /\!/ t_{10})$
Δ'_2	$(t_2 /\!/ t_3 /\!/ 3t_6 /\!/ 2t_7 /\!/ t_{10})$	+	$(2t_1 /\!/ t_4 /\!/ t_5 /\!/ 3t_8 /\!/ 2t_9)$
Δ_3	$(2t_1 /\!/ t_3 /\!/ t_4 /\!/ 3t_6 /\!/ 2t_7)$	+	$(t_2 /\!/ t_5 /\!/ 3t_8 /\!/ 2t_9 /\!/ t_{10})$
Δ'_3	$(t_2 /\!/ t_5 /\!/ 3t_8 /\!/ 2t_9 /\!/ t_{10})$	+	$(2t_1 /\!/ t_3 /\!/ t_4 /\!/ 3t_6 /\!/ 2t_7)$
Δ_4	$(2t_1 /\!/ t_3 /\!/ 3t_6 /\!/ 2t_7 /\!/ t_{10})$	+	$(t_2 /\!/ t_4 /\!/ t_5 /\!/ 3t_8 /\!/ 2t_9)$
Δ'_4	$(t_2 /\!/ t_4 /\!/ t_5 /\!/ 3t_8 /\!/ 2t_9)$	+	$(2t_1 /\!/ t_3 /\!/ 3t_6 /\!/ 2t_7 /\!/ t_{10})$

TAB. 4.1 – *Régime cyclique des solutions*

- $\Delta'_4 = (0,2,1,0,3,5,0,0,3,2,0,0,0,2,1,0,0,1)$

La linéarisation du modèle permet d'obtenir, d'une part, une mise en équation linéaire du comportement logique du GEP et, d'autre part, d'obtenir des marquages initiaux vérifiant une contrainte de performance (ici deux steps). Nous pouvons maintenant à l'aide de ces marquages initiaux déduire le comportement logique du réseau. Pour chacune de ces solutions, nous obtenons un régime cyclique dont la période est constituée de deux steps. Le tableau 4.1 donne le régime cyclique résultant de chacun des huit marquages initiaux. Ces solutions sont ensuite évaluées afin de déterminer celles qui conduisent à la meilleure performance temporelle.

Remarque 5 *Les solutions sont couplées, c'est à dire que la deuxième solution associée résulte de la première après franchissement d'un step. Elles possèdent donc le même comportement cyclique avec un déphasage d'un step.*

Performance temporelle des solutions

Après avoir obtenu toutes les marquages minimaux en deux steps, nous nous intéressons à la durée du cycle pour chacune des solutions. La temporisation déterministe est effectuée sur les transitions comme on peut le voir sur la figure 4.7 ainsi que sur la figure 4.9 où les temporisations additionnelles sont affectées en fonction des variables introduites. Dans notre exemple, les transitions t'_1, t'_2,...,t'_{10} ont la même temporisation que les transitions t_1, t_2,...,t_{10}. Chaque transition ajoutée est un représentant d'une des dix premières transitions :

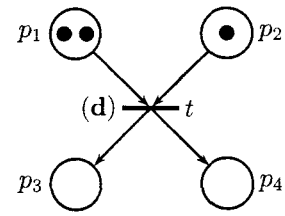


FIG. 4.10 – *Illustration de la temporisation*

elle reçoit donc la même temporisation que la transition représentée. Ainsi, par exemple, les transitions t'_{11} et t'_{12} en tant que représentants de la transition t_1 ont une temporisation de quatre unités de temps.

Pour l'évaluation de la performance temporelle, nous utiliserons un vecteur dateur $D = (D_p)_{p \in P}$ qui associe à chaque place une date représentant la date d'arrivée du dernier jeton dans la place. Lorsque les transitions sont tirées, les dates associées aux places évoluent, en tenant compte de la synchronisation et de la durée du tir. Le franchissement d'un step à partir d'un marquage où les places sont datées par le vecteur dateur $D = (D_p)_{p \in P}$ produit un marquage de vecteur dateur $D' = (D'_p)_{p \in P}$ défini par :

$$\forall p \in P, D'_p = \begin{cases} \max(D_p, \max\limits_{p' \in {}^{\bullet\bullet}p}(D_{p'} + d_{{}^{\bullet}p})) & \text{si la transition } {}^{\bullet}p \text{ est franchie} \\ D_p & \text{sinon} \end{cases}$$

où $d_{{}^{\bullet}p}$ est la durée associée à la transition en amont de la place p.

Ainsi, si l'on considère l'exemple de la figure 4.10, nous obtenons pour la place p_3

$$D'_3 = \begin{cases} \max(D_3, D_1 + d, D_2 + d) & \text{si la transition t est franchie} \\ D_3 & \text{sinon} \end{cases}$$

Appliquons maintenant cette technique de calcul à l'évaluation temporelle de performance de chacune des huit solutions qui ont permis l'ordonnancement des transitions (tableau 4.1). Considérons les huit marquages minimaux qui produisent un fonctionnement en deux steps. Le calcul du vecteur dateur obtenu après un cycle, nous donne très facilement la performance temporelle. Les solutions obtenues par les huit marquages sont notées $(D_1, D_2, \ldots, D_{12}, D_{r_{11}}, D_{r_{12}}, D_{r_{21}}, D_{r_{22}}, D_{r_{31}}, D_{r_{32}})$ où D_i désigne la temporisation affecté à la place p_i. On obtient :

- Pour Δ_1, (5,_,5,7,8,2,_,_,4,3,1,1,7,_,8,_,5,_), soit un cycle en 8 unités de temps.

- Pour Δ'_1, (_,7,3,3,4,5,6,6,_,_,3,2,_,7,_,6,_,6), soit un cycle en 7 unités de temps.
- Pour Δ_2, (3,_,_,7,8,5,_,_,3,5,_,_,6,_,8,_,_,5), soit un cycle en 8 unités de temps.
- Pour Δ'_2, (7,7,7,3,_,2,6,6,_,_,3,2,_,7,_,6,7,_), soit un cycle en 7 unités de temps.
- Pour Δ_3, (3,4,_,7,4,5,6,6,_,_,3,2,7,_,_,6,_,5), soit un cycle en 7 unités de temps.
- Pour Δ'_3, (5,7,5,_,7,2,_,_,4,3,1,1,_,7,7,_,5,_), soit un cycle en 7 unités de temps.
- Pour Δ_4, (7,4,7,7,_,_,6,6,_,_,3,2,7,_,_,6,7,_), soit un cycle en 7 unités de temps.
- Pour Δ'_4, (_,7,3,_,7,5,_,_,4,3,_,_,_,7,7,_,_,5), soit un cycle en 7 unités de temps.

Ainsi, la meilleure performance temporelle (temps de cycle de 7 unités de temps) est obtenue pour les marquage initiaux Δ'_1, Δ'_2, Δ_3, Δ'_3, Δ_4 et Δ'_4.

4.5 Conclusion

Ce chapitre a permis d'illustrer les principales applications de la théorie exposée dans ce mémoire.

L'intérêt de la méthode de linéarisation présentée aux chapitre 2 et 3 conduit à une modélisation finale à partir d'un graphe d'événements simple dont la structure est indépendante du marquage initial considéré. Nous avons montré dans ce chapitre qu'une variation du marquage du modèle initial se traduit par une variation du marquage du graphe d'événements simple final. En outre, cette modélisation permet une évaluation des performances du modèle considéré à l'aide des dioïdes existants : structures algébrique dédiées à cette étude pour les graphes d'événements non pondérés $(\max,+)$ et $(\min,+)$.

Dans ce sens, un premier exemple illustratif a permis d'aborder un cas d'étude concret issu des problèmes liés à la production manufacturière. Après avoir obtenu un système d'équations linéaires caractérisant le comportement du réseau, un système d'équations et d'inéquations associé contraignant le marquage d'un sous-ensemble de places du graphe d'événements simple est alors exhibé. Finalement sur un horizon donné, il est possible de déterminer tous les séquencements des opérations permettant d'obtenir un comportement cyclique en un nombre fini de step.

Pour conclure ce chapitre une évaluation des performances d'un modèle réseau de Petri est proposée. En utilisant la technique de linéarisation, nous montrons sur cet exemple qu'il est possible de calculer tous les séquencements possibles sur un horizon défini décrivant un fonctionnement cyclique, en trois steps. Sur la base de ces solutions nous

réalisons l'évaluation des performances temporelles de cet exemple. Nous obtenons ainsi une solution réalisant le temps de production minimal pour un fonctionnement cyclique en trois steps.

Quatrième partie

Conclusion générale

Conclusion et perspectives

En conclusion de ce mémoire, nous dresserons le bilan des apports originaux de cette thèse, pour ensuite présenter les perspectives de recherche que nous proposons concernant la détermination d'un ordonnancement cyclique pour une classe de problèmes contenant les **S**ystèmes **F**lexibles de **P**roduction **M**anufacturière. Les objectifs d'optimisation de critères retenus dans ce mémoire concernent à la fois le temps total de production et l'en-cours utilisé dans l'atelier de production.

4.6 Conclusion

Dans ce mémoire, nous avons présenté l'étude du comportement logique d'une classe de réseaux de Petri en utilisant une approche algébrique. La problématique considérée est connexe aux thématiques existantes. D'une part les travaux de Harald Ohl, Hervé Camus et Ouajdi Korbaa dont l'objectif est l'ordonnancement cyclique et l'évaluation des performances au sein des ateliers de production et d'autre part, les travaux de Pascal Yim et Ahmer Benasser orienté vers une abstraction logique de réseaux de Petri par propagation de contraintes. Cependant, les indéterminismes et les pondérations rendent la mise en équation complexe et par conséquent, les systèmes non linéaires obtenus sont difficilement utilisables en l'état. Par conséquent, pour obtenir un modèle linéaire, nous décomposons l'étude de cette classe de problèmes en deux étapes.

Tout d'abord, le premier problème abordé dans cette thèse concerne les conflits d'accès aux ressources. Ainsi au chapitre 2, l'étude des partages de ressources est envisagée dans un ordre croissant de difficulté pour proposer finalement une méthode générique de résolution ces conflits. L'idée consiste à intégrer les spécification d'une production particulière au sein du modèle étudié dans le but d'obtenir un graphe d'événements dont le déterminisme est la clé initiale d'une mise en équation à caractère linéaire. En effet, le nombre d'exécu-

tions d'une tâche est déterminé au cours d'une phase préliminaire d'optimisation du flux ; il est alors nécessaire de calculer tous les séquencements potentiels conduisant à un régime cyclique. Dans cette perspective, les différents partages de ressources sont étudiés afin d'élaborer une méthode de transformation dans l'objectif d'aboutir à un graphe d'événements dont le marquage d'un sous-ensemble de places est contraint par un système d'inéquations en général, d'équations dans les cas les plus simples . Par conséquent, la technique de linéarisation des partages de ressources exposée dans ce chapitre permet de transformer un réseau de Petri dont les indéterminismes sont liés aux partages de ressources en un graphe d'événements, éventuellement pondéré, dont le marquage d'un sous-ensemble de places est contraint.

Dans ce sens, au cours du chapitre 3, nous avons considéré la mise en équation du comportement logique des graphes d'événements pondéré. Nous avons mis en évidence le caractère non-linéaire des équations obtenues en raison des pondérations des arcs du graphe d'événements (obtenu après la première phase de transformation). Nous avons alors proposé une méthode de linéarisation relative à une classe de graphe d'événements pondéré recouvrant une large part des systèmes de production en fonctionnement cyclique. Le but est d'aboutir in fine à une mise en équation de ces graphes d'événements à l'aide d'un système linéaire au sens du dioïde $(\min, +)$. Dans ce sens, une structure algébrique $(\min, \mathrm{div})$ à été défini afin de mettre en équation le comportement logique d'un graphe d'événements pondéré. Nous avons alors démontré un théorème énonçant les conditions de linéarisation des GEP. À l'aide d'un algorithme de linéarisation, nous avons donné une première méthode de recherche du système d'équations linéaires associé.

Au cours du dernier chapitre de ce mémoire, nous avons illustré les principaux résultats à l'aide d'exemples concrets. un premier exemple tiré des systèmes de production manufacturière est étudié, afin d'illustrer le formalisme précédemment présenté, ce premier exemple permet de mettre en évidence l'opportunité de construire un graphe d'événements simple associé aux équations obtenues aprés linéarisation. Le second exemple d'illustration tiré de la thése d'Olivier Fournier permet d'illustrer sur un cas d'étude concret la technique de linéarisation proposée au chapitre 2. Enfin une troisième exemple reprenant la base des cas traités au cours des différents chapitres illustre à titre de synthèse les principaux résultats de ce mémoire en allant jusqu'à l'analyse des performances temporelles.

4.7 Perspectives

Les perspectives de ces travaux de recherches peuvent être décomposées en quatre classes.

En premier lieu, les perspectives à court terme concernent la prise en compte des contraintes temporelles. En effet, l'évaluation des performances se fait à posteriori sur les solutions obtenues à partir du système final d'équations et d'inéquations. Or l'objectif est de filtrer les séquencements possibles suivant une borne temporelle. L'idée serait donc d'intégrer un système d'équations supplémentaires dont les solutions caractériseraient les performances temporelles des séquencements, directement, dans le système d'équations caractérisant les séquencements admissibles. Ainsi en spécifiant dés le départ l'horizon de recherche ainsi qu'une borne temporelle, les solutions obtenues caractériseraient un séquencement des opérations possibles en garantissant de surcroît que la performance temporelle reste inférieure à la borne initialement spécifiée.

Cette approche possède deux intérêts majeurs. D'une part, elle permettrait de réduire l'ensemble des solutions possibles ; en effet, nous obtenons un séquencement admissible contraint à une performance majorée par la borne initialement définie. Elle est donc de nature à réduire la complexité de résolution du système d'équations ainsi obtenu. D'autre part, une intégration des contraintes temporelles permettrait d'aborder le problème de l'ordonnancement des tâches, puisque l'obtention d'un séquencement admissible et des dates de déclenchement des opérations permettrait de construire un ordonnancement notamment au niveau des ressources partagées.

En second lieu, il serait intéressant d'étudier spécifiquement la résolution des systèmes d'équations linéaires obtenus. En effet, nous obtenons un système d'un type particulier (algèbre $(\min, +)$) dont les variables sont entières : ce sont les marquages initiaux des places. La résolution est actuellement effectuée à l'aide d'un solveur de contraintes (Prolog *IV* ou Cplex). La mise en place de techniques de résolutions ou des techniques du type recherche de valeurs propres ou recherche de point fixe pourraient conduire à de meilleures performances.

En troisième lieu, il serait intéressant d'envisager une extension du problème pour considérer l'analyse des performances des architectures de transport en production manufacturière. En effet, la prise en compte du transport est essentielle car elle doit être adaptée à la production désirée.

Enfin une quatrième piste de recherche pourrait concerner l'optimisation des per-

formances d'un système déterministe par analogie à l'optimisation du temps de cycle d'un système de production, ou encore de la prise en compte du problème de réduction des en-cours. Un domaine d'application de ce type de problème pourrait concerner la gestion parallèle de tâches dans la réalisation de services répétitifs. Un autre domaine possible est l'organisation et la gestion de systèmes de transports de biens ou de personnes.

Annexes

Annexe A

Les réseaux de Petri

A.1 Définitions et notations

Le modèle réseaux de Petri est connu comme un très bon outil, à la fois graphique et mathématique, de modélisation des systèmes à événements discrets, en particulier les systèmes de production flexible manufacturière (PFM). Dans le but de déduire le comportement de certains problèmes de PFM nous ferons une étude structurelle d'un tel modèle. Ainsi dans un premier temps nous définirons les réseaux de Petri, pour ensuite décrire leur comportement. Nous terminerons avec la définition d'une sous-classe de réseaux de Petri nommés *graphe d'événements* que nous étudierons par la suite.

A.2 Les réseaux de Petri

Nous commençons par définir la notion de *graphe de Petri* [Mur89, Val99, Dia01]. Un graphe de Petri est un graphe orienté biparti. Avec deux types de nœuds appelés *places* et *transitions*, les places seront représentées graphiquement par des cercles et les transitions par des rectangles ou des traits. Les arcs orientés connectent une place à une transition ou une transition à une place, les arcs sont étiquetés par des pondérations le poids unitaire étant omis. Plus formellement, un *graphe de Petri* est un triplet (P,T,W), où :

- $P = \{p_i | 1 \leqslant i \leqslant |P|\}$ et $T = \{t_i | 1 \leqslant j \leqslant |T|\}$ sont des ensembles, non vides et disjoints.

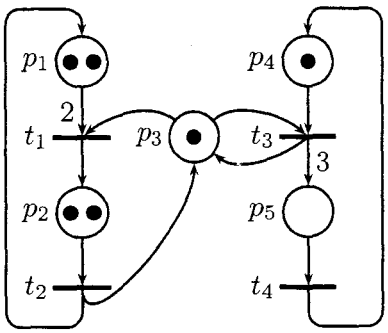


FIG. A.1 – *Exemple de réseau de Petri*

- W : $(P \times T) \cup (T \times P) \longrightarrow \mathbb{N}$ est une application de $(P \times T) \cup (T \times P)$ dans l'ensemble des entiers naturels $\mathbb{N}$

les éléments des ensembles P et T sont respectivement appelés places et transitions. L'application W représente les arcs et leurs pondération : si p est une place et t une transition alors $W(p,t)$ et $W(t,p)$ représentent le poids de l'arc joignant respectivement p à t et t à p (un poids nul signifie que les deux nœuds ne sont pas reliés).

Par exemple, la figure A.1 représente un réseau de Petri comportant cinq places et quatre transitions. Deux des arcs possédent une pondération non unitaire, comme par exemple celui joignant la place p_1 à la transition t_1.

Le graphe décrit précédemment décrit la structure du système de production, l'état du système modélisé peut être décrit en mettant des jetons (appelés également marques) dans les places. Ainsi nous introduisons la notion de *marquage*, que l'on notera M. Un *marquage* est une application de l'ensemble P des places dans l'ensemble des entiers naturels qui à toute place $p \in P$ associe un entier que l'on note m_p et qui représente le nombre de jetons présents dans la place p.

Un *réseau de Petri* $(P,T,W,M(0))$ est un graphe de Petri (P,T,W) muni d'un marquage $M(0)$ appelé *marquage initial*.

À titre d'exemple considérons le réseau de Petri représenté par la figure A.1. Il est composé de cinq places $P = \{p_1, \ldots, p_5\}$, ainsi que de quatre transitions $T = \{t_1, \ldots, t_4\}$. Les nœuds de ce réseau de Petri sont connectés par des arcs orientés pondérés. Si l'on considère par exemple la place p_1 et le transition t_1, nous avons $W(p_1,t_1) = 2$: il existe un arc de poids 2 joignant p_1 à t_1. Par contre, il n'existe pas d'arc joignant la transition t_1 à la place

p_1, on en déduit que $W(t_1,p_1) = 0$. Enfin la place p_1 contient deux jetons, donc $m_{p_1} = 2$.

Les ensembles des transitions en amont et en aval d'une place p seront respectivement notés ${}^\bullet p$ et $p^\bullet$. De même, ${}^\bullet t$ et $t^\bullet$ représentent respectivement les ensembles des places en amont et en aval d'une transition t. Ces ensembles sont définis par :

- $\forall p \in P, {}^\bullet p = \{t \in T | W(t,p) > 0\}$ et $p^\bullet = \{t \in T | W(p,t) > 0\}$
- $\forall t \in T, {}^\bullet t = \{p \in P | W(p,t) > 0\}$ et $t^\bullet = \{p \in P | W(t,p) > 0\}$

Dans notre exemple (figure A.1), l'ensemble des places en amont de la transition t_1 est constitué des places p_3 et p_1, c'est-à-dire $\{p_1,p_3\}$ l'ensemble des transitions en amont de la place p_3 est $\{t_2,t_3\}$.

A.3 Comportement d'un réseau de Petri

Maintenant que la modélisation d'un système à événements discret est définie, nous portons notre intérêt sur le comportement de celui-ci. La dynamique d'un système se traduit, dans cette représentation, par l'évolution du marquage du réseau de Petri obtenu : ceci ce fait par *franchissement* (ou *tir*) de transitions.

Une transition $t \in T$ sera dite *franchissable* (ou *sensibilisée* ou encore *validée*) à partir d'un marquage $M = (m_p)_{p \in P}$, si toutes les places en amont de la transition t possèdent un nombre de jetons suffisant, déterminé par le poids des arcs. Ce qui se traduit par la condition suivante :

$$\forall p \in P,\ m_p \geqslant W(p,t) \tag{A.1}$$

Par exemple pour le réseau de Petri représenté par la figure A.1, la transition t_4 n'est pas validée alors que chacune des transitions t_1, t_2, et t_3 est franchissable. Le franchissement d'une transition t se traduit par une consommation de jetons dans les places en amont de la transition t et une création de jetons dans les places en aval de cette même transition. Le nombre de jetons consommés ou créés dans une place p est égal au poids de l'arc joignant la place p et la transition t. La présence d'un nombre de jetons suffisant dans les places amont de la transition t étant induite par la vérification de la condition A.1, le franchissement de la transition t induit la consommation des jetons des places en amont de la transitions et la production de jetons dans les places en aval de la transition t. Ceci se traduit par l'equation suivante :

$$\forall p \in P,\ m'_p = m_p - W(p,t) + W(t,p)$$

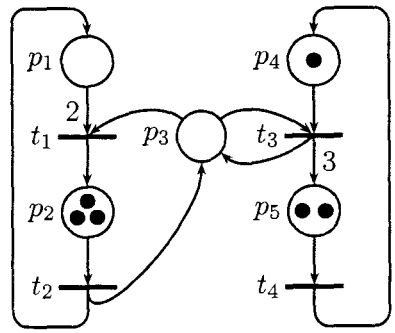


FIG. A.2 - *Franchissement de la transition* t_1

Dans notre exemple (figure A.1), si on franchit uniquement la transition t_1 on consomme alors deux jetons dans la place p_1 et un jeton dans la place p_3. Puis on ajoute un jeton dans la place p_2. On obtient alors un nouveau marquage représenté par la figure A.2.

On introduit maintenant la notion de *step* [JK91]caractérisant le tir en parallèle de transitions simultanément franchissables. Ainsi, un step est un multi-ensemble de transitions : un ensemble pour lequel les transitions peuvent avoir plusieurs occurrences. Un step sera représenté par une expression symbolique. Par exemple, $S = t_1 \,/\!/\, 2t_4$ est un step contenant une occurrence de t_1 et deux occurrences de la transition t_4. Pour un step S, le symbole s_t représentera le nombre d'occurrences de la transition t. Un step est franchissable à partir d'un marquage si toutes les transitions du step sont simultanément franchissables, c'est-à-dire s'il est possible d'effectuer toutes les consommations de jetons requis. Plus formellement, un step S est franchissable à partir d'un marquage $M = (m_p)_{p \in P}$ si et seulement si :

$$\forall p \in P,\ m_p \geqslant \sum_{t \in T} s_t W(p,t)$$

Le franchissement d'un step S conduit au marquage $M' = (m'_p)_{p \in P}$, défini par :

$$\forall p \in P,\ m'_p = m_p - \sum_{t \in T} s_t W(p,t) + \sum_{t \in T} s_t W(t,p).$$

Le marquage ainsi produit correspond à celui obtenu après franchissement (éventuellement multiple) des transitions qui composent le step. Á titre d'exemple(figure A.1), lorsque l'on considère les transitions t_1 et t_3, ces deux transitions sont toutes deux franchissables mais pas en paralléle.

Un réseau de Petri est associé à une représentation matricielle. Dans la suite on note : $P = \{p_1,p_2,\ldots,p_{|P|}\}$ et $T = \{t_1,t_2,\ldots,t_{|T|}\}$. Le réseau est alors entièrement déterminé par ses matrices caractéristiques : la matrice des *pré-conditions* $Pre = (W(p_i,t_j))_{\substack{1\leqslant i\leqslant |P| \\ 1\leqslant j\leqslant |T|}}$, la matrice des *post-conditions* $Post = (W(t_j,p_i))_{\substack{1\leqslant i\leqslant |P| \\ 1\leqslant j\leqslant |T|}}$. La matrice d'incidence C est définie par $C = Post - Pre$.

On représentera le marquage $M = (m_{p_i})_{1\leqslant i\leqslant |P|}$ et le step $S = (s_{t_j})_{1\leqslant j\leqslant |T|}$ par les vecteurs colonnes $M = \begin{pmatrix} m_1 & m_2 & \ldots & m_{|P|} \end{pmatrix}^T$ et $S = \begin{pmatrix} s_1 & s_2 & \ldots & s_{|T|} \end{pmatrix}^T$ où $\forall i \in \{1,2,\ldots,|P|\}$, $m_i = m_{p_i}$ et $\forall j \in \{1,2,\ldots,|T|\}$, $s_j = s_{t_j}$. Le comportement du réseau de Petri est alors régi par l'*équation fondamentale* (A.2), appelée également *équation d'état*. Cette équation lie le marquage M' produit par le franchissement d'un step S à partir d'un marquage M :

$$M' = M + C \cdot S \tag{A.2}$$

A.4 Les graphes d'événements

Nous définissons maintenant deux sous-classe de réseaux de Petri que nous serons amenés à considérer par la suite : *les graphes d'événements simples* et *pondérés*.

- les *graphes d'événements simples* (GES) pour lesquels chaque place possède une seule transition entrante et une seule transition sortante, et dont le poids des arcs est unitaire :

$$\forall t \in T, \forall p \in P, \begin{cases} |{}^\bullet p| = |p^\bullet| = 1 \\ W(t,p) \in \{0,1\} \\ W(p,t) \in \{0,1\} \end{cases}$$

- les *graphes d'événements pondérés* (GEP), vus comme une extension des graphes d'événements simples, pour lesquels les arcs peuvent être pondérés :

$$\forall t \in T, \forall p \in P, \begin{cases} |{}^\bullet p| = |p^\bullet| = 1 \\ W(t,p) \in \mathbb{N} \\ W(p,t) \in \mathbb{N} \end{cases}$$

À l'évidence, les GEP sont plus généraux que les GES. Nous allons considérer plus particulièrement les réseaux de Petri consistants et conservatifs : ceux pour lesquels il existe

un *T-semiflot* (c'est-à-dire un vecteur X non nul et positif tel que $CX = 0$) et un *P-semiflot* (un vecteur Y non nul et positif tel que $Y^T C = 0$). Le franchissement d'une séquence de transitions correspondant au T-semiflot ne modifie pas le marquage.

Étant donné que chaque place d'un GEP ne possède qu'une transition en amont et une transition en aval, par abus de langage, on identifiera ${}^\bullet p$ (respectivement $p^\bullet$) à la transition en amont (respectivement en aval) de la place p : ainsi, si ${}^\bullet p = \{t_i\}$ (respectivement $p^\bullet = \{t_i\}$), alors ${}^\bullet p$ (respectivement $p^\bullet$) désignera la transition t_i.

Le caractère déterministe de ces réseaux, i.e. le fait que chaque place possède une et une seule transition en aval, évite un conflit entre deux transitions. Le marquage initial du graphe d'événements suffit pour déterminer le comportement logique du système. Le fonctionnement déterministe permet une approche mathématiques en utilisant l'algèbre des dioïdes.

A.5 Les graphes d'événements temporisés

Lorsque l'on évalue la performance d'un système de production, nous étudions les temps de franchissements des transitions. donc deux cas se présente :

- La temporisation est affectée aux places. Un jeton, arrivant dans une place, reste indisponible durant un délai correspondant à la date affectée à la place. Á titre d'exemple, considérons le réseau de Petri représenté par la figure ??, un jeton arrivant à la date t dans la place p_1 ne pourra être consommé qu'à partir de la date $t + \tau_1$.
- Temporisation des transitions, le jeton est consommé par la transition et la création dans les places aval est effectuée après la période associé à la place.(système (max ,+))

Bibliographie

[Ama94] S. AMAR. « *Systèmes automatisés et flexibles de production manufacturière : méthode de conception du système de coordination par prototypage orienté objet de la partie procédé* ». thèse de doctorat, Université de Lille 1, 1994.

[Aus94] C. AUSFELDER. « *Contribution à la conception d'un système de conduite pour les systèmes flexibles de production manufacturière : modélisationet validation de la commande* ». thèse de doctorat, Université de Lille 1, 1994.

[BCOQ92] F. BACCELLI, G. COHEN, G.J. OLSDER, et J.P. QUADRAT. *Synchronization and Linearity.* Wiley, 1992.

[BEN82] R. BELLMANN, A. O. ESOGBUE, et I. NABESHIMA. *Mathematical Aspects of Scheduling and Applications.* Pergamon Press, 1982.

[Ben00] A. BENASSER. « *Accessibilité dans les réseaux de Petri: une approche basée sur la programmation par contraintes* ». thèse de doctorat n° 2695, Université de Lille 1, 20 janvier 2000.

[Boi91] S. BOIS. « *Intégration de la gestion des modes de marche dans le pilotage d'un système automatisé de production* ». thèse de doctorat, Université de Lille 1, 1991.

[Bon87] R. BONETTO. *Les Ateliers Flexibles de Production.* Hermès, 1987.

[Bou93] J.-P. BOUREY. « Méthode de conception de la commande de systèmes flexibles de production manufacturière ». habilitation à diriger les recherches, Université de Lille 1, 1993.

[BY99] A. BENASSER et P. YIM. « Railway Traffic Planning with Petri Nets and Constraint Programming ». *APII-JESA, Modelling of Reactive Systems*, 33(8–9):959–975, novembre 1999.

[BY01] A. BENASSER et P. YIM. « A Logical Abstraction for Autonomous Timed Petri

Net ». Dans *International Conference on Industrial Engineering and Production Management*, Université Laval, Quebec, Canada, 20–23 août 2001.

[Cam97] H. CAMUS. « *Conduite de systèmes flexibles de production manufacturière par composition de régimes permanents cycliques : modélisation et évaluation de performances à l'aide des réseaux de Petri* ». thèse de doctorat n° 1975, Université de Lille 1, 18 mars 1997.

[Cas96] E. CASTELAIN. « Contribution à la conduite des systèmes de production manufacturière ». habilitation à diriger les recherches, Université de Lille 1, 1996.

[CC82] J. CARLIER et P. CHRÉTIENNE. « Un domaine très ouvert : les problèmes d'ordonnancement ». *R.A.I.R.O. Recherche opérationnelle*, 16(3):175–217, août 1982.

[CC88a] J. CARLIER et P. CHRETIENNE. *Problème D'ordonnancement : Modélisation / Compléxité / Algorithnmes.* Éditions Masson, Paris, France, 1988.

[CC88b] J. CARLIER et P. CHRETIENNE. « Timed Petri Net Schedules. ». *Lecture Notes in Computer Science: Advances in Petri Nets 1988*, 340:62–84, 1988. NewsletterInfo: 32.

[CDQV83] G. COHEN, D. DUBOIS, J.-P. QUADRAT, et M. VIOT. « Analyse du comportement périodique de systèmes de production par la théorie des dioïdes ». Rapport Technique 191, INRIA, Rocquencourt, février 1983. in french.

[CDQV85] G. COHEN, D. DUBOIS, J.-P. QUADRAT, et M. VIOT. « A Linear-System-Theoric View of Discrete-Event Processes and Its Use for Performance Evaluation in Manufacturing ». *IEEE Transactions on Automatic Control*, 30(3):210–220, mars 1985.

[Cer88] A. CERNAULT. *La Simulation Des Sytèmes de Production.* Cepaudes Editions, 1988.

[CG79] R.-A. CUNINGHAME-GREEN. *Algebraic Realization of Discrete Dynamical Systems.* Numéro 166. Lecture notes in Economics and mathematical Systems, Berlin, springer-verlag édition, 1979.

[CGQ95a] G. COHEN, S. GAUBERT, et J.-P. QUADRAT. « Algebraic System Analysis of Timed Petri Nets ». Dans J. GUNAWARDENA, éditeur, *Idempotency*, Publications of the Newton Institute. Cambridge University Press, mars 1995.

[CGQ95b] G. COHEN, S. GAUBERT, et J.P. QUADRAT. « Asymptotic Throughput of Conti-

nuous Timed Petri Nets ». Dans *Proceedings of the 34th Conference on Decision and Control*, New Orleans, Dec 1995.

[CGQ98] G. COHEN, S. GAUBERT, et J.-P. QUADRAT. « Timed Event Graphs with Multipliers and Homogeneous Min-Plus Systems ». *IEEE Transactions on Automatic Control*, 43(9):1296–1302, septembre 1998.

[CHEP71] F. COMMONER, A. W. HOLT, S. EVEN, et A. PNUELI. « Marked directed graphs ». *Journal of Computer and System Sciences*, 5(5):511–523, 1971.

[Chr86] P. CHRETIENNE. « Timed Petri Nets: A Solution to the Minimum-Time-Reachability Problem between two States of a Timed-Event Graph. ». *J. Syst. Software*, 6(1-2):95–101, 1986. NewsletterInfo: 25.

[CMQV84] G. COHEN, P. MOLLER, J.P. QUADRAT, et M. VIOT. « Linear system theory for discrete-event systems ». Dans *23rd IEEE Conf. on Decision and Control*, Las Vegas, Nevada, 1984.

[CMQV85] G. COHEN, P. MOLLER, J.P. QUADRAT, et M. VIOT. « Une théorie linéaire des systèmes à événements discrets ». Rapport de recherche 362, INRIA, Le Chesnay, France, 1985.

[CMQV86] G. COHEN, P. MOLLER, J.P. QUADRAT, et M. VIOT. « Dating and counting events in discrete event systems ». Dans *25th IEEE Conf. on Decision and Control*, Athens, Greece, 1986.

[CMQV89] G. COHEN, P. MOLLER, J.-P. QUADRAT, et M. VIOT. « Algebraic Tools for the Performance Evaluation of Discrete Event Systems ». *IEEE Proceedings*, 77(1):39–57, janvier 1989.

[Cra94] E. CRAYE. « Contribution au contrôle / commande de système de production manufacturière ». habilitation à diriger les recherches, Université de Lille 1, 1994.

[Cru91] D. CRUETTE. « *Méthodologie de conception des systèmes complexes à événements discrets : application à la conception et à la validation de la commande de cellules flexibles de production dans l'industrie manufacturière* ». thèse de doctorat, Université de Lille 1, 1991.

[CTCG+] J. COCHET-TERRASSON, G. COHEN, S. GAUBERT, M. MCGETTRICK, et J.-P QUADRAT. « Numerical computation of spectral elements in max-plus algebra ». In Proc. IFAC Conf. on Syst. Structure and Control (1998).

[CTGG99] J. COCHET-TERRASSON, S. GAUBERT, et J. GUNAWARDENA. « A constructive fixed point theorem for min-max functions ». *Dynamics and Stability of Systems, 14(4)*, 1999.

[DHP+93] F. DICESARE, G. HARHALAKIS, J.-M. PRITH, M. SILVA, et F. B. VERNADAT. *Practice of Petri Nets in Manufacturing.* Chaoman & Hall, 1993.

[Dia01] M. DIAZ. *Les réseaux de Petri.* Édition Hermès, 2001.

[EK93] S. EL KHATTABI. « *Intégration de la surveillance de bas niveau dans la conception des systèmes à événements discrets : application aux systèmes de production flexibles* ». thèse de doctorat, Université de Lille 1, 1993.

[ER82] J. ERSCHLER, D. , Levêque, et F. ROUBELLAT. « Periodic Loading of Flexible Manufacturing Systems ». Dans *IFIP Congress, APMS,*, pages 327–339, France, 1982.

[Fou02] O. FOURNIER. « *Conception de la commande d'un système automatisé de production : apport des graphes et de l'ordonnancement cyclique* ». thèse de doctorat, Université de la réunion, mars 2002.

[FTE] J. M. Colom F. TRICAS, F. García-Vallés et J. EZPELETA. « New Methods for Deadlock Prevention and Avoidance in Concurrent Systems ». Dans *Jornadas de Concurrencia 2000, Cuenca, Spain, June, 14-16, 2000*, pages 97–110.

[Gau92a] S. GAUBERT. *Introduction aux systèmes dynamiques à événements discrets.* Polycopié de cours donné à l'ENSTA, 1992.

[Gau92b] S. GAUBERT. « *Théorie des systèmes linéaires dans les dioïdes* ». Thèse, École des Mines de Paris, July 1992.

[Gau94] S. GAUBERT. « On Semigroups of Matrices in the (max ,+) Algebra ». Rapport de recherche 2172, INRIA, janvier 1994.

[Gau95a] S. GAUBERT. « Performance Evaluation of (max,+) Automata ». *IEEE Trans. on Automatic Control*, 40(12), Dec 1995.

[Gau95b] S. GAUBERT. « Resource Optimization and (Min,+) Spectral Theory ». *IEEE Transactions on Automatic Control*, 40(11):1931–1934, novembre 1995.

[GBK88] J.-C. GENTINA, J.-P. BOUREY, et M. KASPUTA. « Colored Adaptive Structured Petri Nets - a Tool for the Automatic Synthesis of Hierarchical Control of Flexible Manufacturing Systems ». *Computer Integrated Manufacturing Systems*, 1(1):39–47, février 1988.

[GM84] M. GONDRAN et M. MINOUX. *Graphs and Algorithms*. John Wiley and Sons, New York, 1984.

[Gun94] J. GUNAWARDENA. « Min-max functions ». *Discrete Event Dynamic Systems*, 4:377–406, 1994.

[Ham91] S. HAMMADI. « *Une méthode d'ordonnancement minimisant le temps d'attente et le transit dans les systèmes de production flexible de type Job-Shop* ». thèse de doctorat, Université de Lille 1, 1991.

[HP89] H. P. HILLION et J. P PROTH. « Performance Evaluation of Job-Shop Systems Using Timed Event Graphs ». *IEEE Transactions on Automatic Control*, 34(1):3–9, janvier 1989.

[Huv94] B. HUVENOIT. « *De la conception à l'implémentation de la commande modulaire et hiérarchisée de systèmes flexibles de production manufactuière* ». thèse de doctorat, Université de Lille 1, 1994.

[Jen92] K. JENSEN, éditeur. *On Weignted T-Systems*, Sheffield, UK, June 1992. Lecture Notes in Computer Science, Springer-Verlag.

[JK91] R. JANICKI et M. KOUTNY. « Optimal Simulations, Nets and Reachability Graphs ». Dans G. ROZENBERG, éditeur, *Advances in Petri Nets 1991*, volume 524 de *Lecture Notes in Computer Science*, pages 205–226. Springer-Verlag, 1991.

[Kas88] M. KASPUTA. « *Génération Assistée D'un Graphe Fonctionnel Destiné À L'élaboration Structurée Du Modèle de la Partie Commande Pour Les Cellules de Production Flexibles Dans L'industrie Manufacturière* ». thèse de doctorat, Université de Lille 1, 1988.

[Ker96] L. KERMAD. « *Contribution à la supervision et à la gestion des modes et des configurations des systèmes flexibles de production manufacturière* ». thèse de doctorat, Université de Lille 1, 1996.

[Kor98] O. KORBAA. « *Commande cylclique des systèmes flexibles de production manufacturière à l'aide des réseaux de Petri: de la planification à l'ordonnancement des régimes transitoires* ». thèse de doctorat, Université de Lille 1, 09 juillet 1998.

[LBH99] S. LAHAYE, J.L. BOIMOND, et L. HARDOUIN. « Timed Event Graphs with

Variable Resources: Asymptotic Behavior, Representation in (min ,+) Algebra ». *Journal Européen des Systèmes Automatisés*, 33(8-9):1015–1032, novembre 1999.

[Lee02] Jong Kun LEE. « *Une méthode d'analyse d'ordonnancement des systèmes flexibles de production manufacturière utilisant le dépliage des réseaux de Petri* ». Thèse de doctorat, Université de Lille 1, 2002.

[MM90] J. MESEGUER et U. MONTANARI. « Petri Nets are Monoids. ». Dans USA: SRI International TECHNICAL REPORT MENLO PARK, CA, éditeur, *Information and Computation, Vol. 88, No. 2*, pages 105–155. Springer-Verlag, octobre 1990.

[Mun93] A. MUNIER. « Régime asymptotique optimal d'un graphe d'événements temporisé généralisé. Application à un problème d'assemblage ». *RAIRO-APII*, 27:487–513, 1993.

[Mur89] T. MURATA. « Petri Nets : Properties, Analysis and Applications ». *Proceedings of the IEEE*, 77(4):541–580, avril 1989.

[OCCG94] H. OHL, H. CAMUS, E. CASTELAIN, et J.-C. GENTINA. « A Heuristic Algorithm for the Computation of Cyclic Schedules and the Necessary WIP to Obtain Optimal Cycle Time ». Dans *Rensselaer's Fourth International Conference on Computer Integrated Manufacturing and Automation Technology, CIMAT'94*, volume 1, pages 339–344, New York (U.S.A.), octobre 1994.

[OCCG95] H. OHL, H. CAMUS, E. CASTELAIN, et J.-C. GENTINA. « Petri Net Modeling of Ratio-Driven Flaxible Manufacturing Systems and Implications on the WIP for Cyclic Schedules ». *IEEE conference on Systems*, 4(5):3081–3086, octobre 1995.

[OCG94] H. OHL, E. CASTELAIN, et J.-C. GENTINA. « Synchrony Theory Applied to Control Problems in Flexible Manufacturing Systems ». Dans *IEEE Conference on Systems, Man and Cybernetics*, volume 2, pages 1689–1694, San Antonio, Texas, U.S.A., 1994.

[Ohl95] H. OHL. « *Fonctionnement répétitifs de systèmes flexibles de production manufacturière : analyse et optimisation des performances à l'aide de réseaux de Petri* ». thèse de doctorat, Université de Lille 1, 1995.

[Pet62] C. A. PETRI. « *Kommunikation mit Automaten* ». PhD thesis, Institut fur Instrumentelle Mathematik, Bonn, 1962. PhD Thèse.

[Plu90] M. PLUS. « Linear systems in (max,+)-algebra ». Dans *Proceedings of the 29th Conference on Decision and Control*, Honolulu, Dec. 1990.

[PX95] J.-M. PROTH et X. XIE. *Les réseaux de Petri pour la conception et la gestion des systèmes de production.* Édition Masson, 1995.

[RTS99] L. RECALDE, E. TERUEL, et M. SILVA. « Autonomous Continuous P/T Systems. ». Dans DONATELLI, SUSANNA et KLEIJN, JETTY, éditeurs, *Lecture Notes in Computer Science: Application and Theory of Petri Nets 1999, 20th International Conference, ICATPN'99, Williamsburg, Virginia, USA*, volume 1630, pages 107–126. Springer-Verlag, juin 1999.

[ST96a] M. SILVA et E. TERUEL. « Petri Net for the Design and Operation of Manufacturing Systems ». *CIMAT'96*, 1:330–343, mai 1996.

[ST96b] M. SILVA et E. TERUEL. « A Systems Theory Perspective of Discrete Event Dynamic Systems: The Petri Net Paradigm ». *CESA'96*, pages 1–12, juillet 1996.

[SU89a] P. SERAFINI et W. UKOVICH. « A Mathematical Model for Periodic Scheduling Problems ». *Society for Industrial and Apllied Mathematics*, 2(4):pp. 550–581, novembre 1989.

[SU89b] P. SERAFINI et W. UKOVICH. « A Mathematical Model for Periodic Scheduling Problems ». *SIAM J. Disc*, 2(4):550–581, novembre 1989.

[Taw95] R. TAWEGOUM. « *Contrôle temps réel de déroulement des opérations dans les systèmes de productions flexibles* ». thèse de doctorat, Université de Lille 1 - École Centrale de Lille, 1995.

[TB02] B. TROUILLET et A. BENASSER. « Cyclic Scheduling Problems with Assemblies: An Approach Based to the Search of an Initial Marking in a Marked Graph ». *IEEE Systems, Man and Cybernetics*, octobre 2002.

[TBG01] B. TROUILLET, A. BENASSER, et J.-C. GENTINA. « Sur la modélisation du comportement dynamique des graphes d'événements pondérés ». Dans G. JUANOLE et R. VALETTE, éditeurs, *Modélisation des systèmes réactifs (MSR'2001)*, pages 447–462, Toulouse, France, 17–19 octobre 2001. Hermes.

[TBG02a] B. TROUILLET, A. BENASSER, et J.-C. GENTINA. « On the Modelling of the Dynamical Behavior of Weighted T-Systems ». *APII - JESA*, 36(7):931 – 944, 2002.

[TBG02b] B. TROUILLET, A. BENASSER, et J.-C. GENTINA. « Transforamtion of Cyclic Scheduling Problem of a Large Class of FMS into the Search of an Optimized Initial Marking of a Linearizable Weighted T-System ». Dans 6th international Workshop on DISCRETE EVENT SYSTEMS (WODES'2002), éditeur, *Silva, M. and Giua, A. and Colom, J.M.*, pages 83–90, Zaragoza, Spain, 2–4 octobre 2002. IEEE computer Society.

[TE97] Fernando TRICAS et Joaquín EZPELETA. « A partial approach to the problem of deadlocks in processes with resources. ». Dans FARWER, B., MOLDT, D., et STEHR, M.-O., éditeurs, *Report FBI-HH-B-205/97: Proceedings of the Workshop on Petri Nets in System Engineering (PNSE'97), Hamburg, September 25-26, 1997*, pages 135–150. Universit¨at Hamburg, septembre 1997.

[Tog92] A. TOGUYENI. « *Surveillance et diagnostic en ligne dans les ateliers flexibles de l'industrie manufacturière* ». thèse de doctorat, Université de Lille 1, 1992.

[Tou82] J. M. TOUDIC. « Algorithmes d'algèbre linéaire pour l'analyse structurelle des réseaux de Petri ». Dans *Revue Technique Thomson-CSF*, volume 14(1), pages 131–155, mars 1982.

[Val94] C. VALENTIN. « Modeling and Analysis Methods for a Class of Hybrid Dynamic Systems ». *Symposium Automatisation des processus mixtes : Les systèmes dynamiques hybrides*, pages 221–226, novembre 1994.

[Val99] R. VALETTE. *Les réseaux de Petri.* Polycopié de cours donné au LAAS-CNRS Toulouse, 1999.

[Zim81] ZIMMERMANN. *Linear Conbinatorial Optimization in Ordered Algebraic Structures.* North-Holland, Amsterdam, 1981.