Développement d'une méthode structurelle de commande par supervision des systèmes à événements discrets modélisés par les réseaux de Petri

Mohaman Gonza

UNIVERSITE DE NGAOUNDERE

ECOLE NATIONALE SUPERIEURE DES SCIENCES AGRO-INDUSTRIELLES

THE UNIVERSITY OF NGAOUNDERE

NATONAL SCHOOL OF AGRO-INDUSTRIALS SCIENCES

DEPARTEMENT DE GENIE ELECTRIQUE, ENERGETIQUES ET AUTOMATIQUE

DEPARTMENT OF ELECTRICAL ENGINEERING, ENERGETICS AND AUTOMATIC

UNITE DE FORMATION DOCTORALE PHYSIQUE APPLIQUEE ET INGENIERIE (UFD-PAI)

Développement d'une méthode structurelle de commande par supervision des systèmes à événements discrets modélisés par les réseaux de Petri

THÈSE

Présentée en vue de l'obtention du diplôme de **DOCTORAT/Ph.D.**

Parcours/Spécialité :
Génie électrique et Automatique industrielle

Par
GONZA Mohaman

Master en Informatique, Electronique et Automatique (IEA)

Matricule : 06M008EN

Thèse soutenue publiquement le 09 janvier 2019 devant le Jury composé de :

M. BEDA TIBI	Professeur	Université de N'Gaoundéré	Président
M. ALLA HASSANE Lotfi	Professeur	Université de Grenoble, Alpes (France)	Directeur
M. BITJOKA Laurent	Professeur	Université de N'Gaoundéré	Directeur
M. TIEUDJO Daniel	Maître de Conférences	Université de N'Gaoundéré	Rapporteur
M. BOUETOU BOUETOU T.	Professeur	Université de Yaoundé I	Rapporteur
M. NOUREDDINE ZERHOUNI	Professeur	ENSMM de Besançon (France)	Rapporteur
M. HAMAN - DJALLO	Maître de Conférences	Université de N'Gaoundéré	Membre

Année 2018

Je dédie cette thèse

A l'âme de mon feu Père et à ma très chère mère

Les mots ne sauraient exprimer tout l'amour et la reconnaissance pour vos prières et vos bénédictions… !

A ma chère épouse et nos deux enfants

Merci d'être toujours à mes côtés. Ton amour dévoué et tendre donne du sens à notre vie de famille

Remerciements

Tout d'abord, « je veux remercier le Dieu saint de tout mon cœur ! Sans oublier un seul de ses bienfaits »

Je remercie chaleureusement mes directeurs de thèse :

- M. Laurent BITJOKA, Professeur à l'Université de Ngaoundéré,
- M. Hassane ALLA, Professeur à l'Université de Grenoble Alpes

Vous êtes pour moi plus que des directeurs de thèse, aussi bien pour vos qualités scientifiques, sociales et humaines. Vous avez fait toujours preuve de patience et de confiance. Merci pour tout !

Je remercie vivement les rapporteurs pour l'honneur d'avoir accepté de porter un jugement sur ce travail de thèse. Vos commentaires précis et judicieux ont amélioré la qualité de ma thèse.

J'exprime ma profonde ma reconnaissance aux imminents membres du jury, qui ont accepté de juger et d'examiner de manière constructive mon travail.

J'aimerais remercier les membres des jurys des journées doctoriales et de pré-soutenance, pour les conseils et les orientations apportés à ce travail.

J'exprime ma plus profonde et sincère gratitude à mes frères et à mes sœurs, ainsi qu'à mes plus proches parents. Que ce mémoire soit un témoignage de la fraternité, surtout à toi NGWAKEN.

J'adresse mes plus vifs remerciements à Monseigneur Samuel KLEDA, Archevêque métropolitain de Douala. Pour les largesses que vous m'avez accordé bien qu'étant employé dans votre structure MACACOS.

J'adresse également mes sincères remerciements à M. Francis HAPPI et M. Mohamadou SALISSOU respectivement DG et DGA de Beta Consult ; à M. Abdouramane SOULBANKAI, DG de SCPR. En travaillant dans vos structures à temps partiel, j'ai pu bénéficier de vos soutiens.

J'exprime ma reconnaissance à toutes les personnes qui m'ont aidé et soutenu au quotidien, à l'instar du Dr Boukar OUSMAN, Mamoudou OUSOUMANOU …

Je n'oublierai pas les amis, dont le soutien moral et affectif, a été de tous les instants.

Résumé

Plusieurs méthodes formelles de commande par supervision des systèmes à événements discrets (SED) basées sur les réseaux de Petri (RdP) ont été développées. Cependant, elles sont partiellement structurelles à cause de la nécessité de construire le graphe de marquage, qui est un automate, pour déterminer les états interdits par la spécification. Certaines de ces méthodes sont limitées à une classe particulière de RdP. Notre travail s'est focalisé en particulier sur le RdP synchronisé (RdPS), qui offre une meilleure structure pour définir les langages et les automates. Parmi les méthodes basées sur les RdP, la méthode des invariants (simple et efficace) est la plus utilisée bien qu'elle ne garantisse pas en général une solution de contrôle optimale. Pour pallier ce problème, nous avons développé une méthode complètement structurelle, sans construction du graphe de marquage, pour les RdPS ordinaires ou généralisés. Elle consiste à déterminer de manière structurelle les contraintes admissibles pour la méthode des invariants. Elle est basée sur la condition de contrôlabilité structurelle des états du RdPS du SED en boucle fermée, obtenu par produit synchrone des RdPS du procédé et de la spécification. En outre, nous avons prouvé que la condition de contrôlabilité structurelle, définie par la condition de marquage des places d'entrée d'une transition synchrone incontrôlable, est équivalente à la condition de contrôlabilité définie par les langages du RdPS. Cette condition définit l'équation d'un hyperplan séparateur qui assure la séparation des états admissibles des états interdits dans le SED. Les contraintes dérivées de d'hyperplan séparateur sont admissibles et peuvent être simplifiées systématiquement. Elles permettent de faire un lien biunivoque entre la théorie de commande par supervision de Ramadge et Wonham et la méthode des invariants pour le calcul d'un contrôleur optimal au sent maximal permissif. Même si pour certaines structures très complexes (à grande échelle), la solution maximale permissive ne peut être garantie directement, notre approche a donné le contrôleur optimal dans les études de cas réels considérés.

Mots clés : Automates à états finis, Commande par supervision, Contrôlabilité, Hyperplan Séparateur, Réseaux de Petri, Systèmes à Evénements Discrets.

Abstract

Several formal methods of supervisory control of discrete events systems (DES), based on Petri nets (PN) have been developed. However, they are partially structural because of the need to construct the marking graph, which is an automaton, to determine the states forbidden by the specification. Some of those methods are limited to a particular class of Petri nets.

Our work focused on the labeled Petri nets (LPN), which offers a better structure for defining the languages and automata. Among the based methods based on Petri nets, the invariant place method (simple and efficient) is the most used, although it does not guarantee in the general an optimal solution.

In order to overcome this problem, we have developed a completely structural method, without construction of the marking graph for ordinary labeled Petri nets. It consists in determining in a way structural, admissible constraints for the invariant method place. It is based on the structural condition of controllability of closed loop SED LPN states, got by synchronous product of the process and specification LPN. In addition, we have proved that the structural condition of controllability, defined by the marking condition of entrance places of an uncontrollable synchronous transition, is equivalent to the condition of controllability defined by the languages of the LPN. This condition defines the equation of a separator hyperplan which ensures separation of the admissible states from the forbidden states in the SED. Constraints derived from hyperplane separator are admissible and can be simplified systematically. They provide a one-to-one link between the theory of supervisory control of Ramadge and Wonham and the calculation of maximum permissive controller by the invariant place method. Although for some complex structures (great scale), the maximum permissive solution cannot be guarantee directly; our approach gave the optimal controller in the real case studies considered.

Keys words: Controllability, Discrete Event Systems, Finite State Automata Hyperplan Separator, Petri Net, Supervisory Control

Table des matières

Table des figures

Liste des sigles et symboles

- **SED** : système à évènements discrets
- ***E* alphabet** : ensemble fini non vide de symboles
- **σ mot (une séquence)** : suite de symboles de l'alphabet *E*.
- **(*)** Opération de *fermeture de Kleene* (mots obtenus par concaténation)
- **Projection $P_2(\sigma)$,** est la suite obtenue de σ en gardant les symboles qui appartiennent à E_2
- || Opération de *produit synchrone*
- $\mathcal{A} = (Q, E, \delta, q_0, Q_m)$ Un automate (à états finis) déterministe
- $L(\mathcal{A})$: langage généré par *A*
- $R = \langle P, T, W^-, W^+, M_0 \rangle$ réseaux de Petri (RdP)
- $W^-(\bullet, t_j)$ (resp. $W^+(\bullet, t_j)$) est la fonction d'incidence avant (resp. arrière)
- $(\bullet, t_j)$ poids d'un arc : nombre minimum de marques ou jetons requis dans la place p_i
- $L(R)$: langage généré par *R*
- $M_k = [M_k(p_0)\ M_k(p_1)\ \ldots\ M_k(p_n)]^T$ vecteur de marquage du RdP : Etat courant du SED.
- $\mathcal{A}(R, M_0)$: Espace d'état accessible du RdP
- $\mathcal{G} = \{M, E, \delta, M_0\}$: Graphe de marquage du RdP
- $N = \langle R, E, f \rangle$, où $R = \langle P, T, W^-, W^+, M_0 \rangle$: RdP synchronisé (RdPS)
- $C = (V, E, \xi, v_0, 2^{Ec}, \theta)$: modèle automate du contrôleur
- $E_{uc} \cap L(P) \subseteq \overline{K}$: condition de contrôlabilité définit sur les langages
- E_{uc} : ensemble d'événements incontrôlables
- $Sup\,C(K)$: Langage suprême contrôlable
- $\mathcal{M}_I$: Ensemble des états interdits
- $\mathcal{M}_A$: Ensemble des états admissibles ou autorisés
- $\mathcal{M}(l, b)$**:** Contraintes linéaires entre les marquages des places du RdP dites GMEC
- **GMEC :** General Mutual Exclusion Constraint
- $N = N_p \,||\, N_s$ RdPS du SED en boucle fermée
- $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$: Condition de contrôlabilité structurelle

Introduction

La synthèse de commande par supervision des systèmes industriels est une activité complexe à cause de la philosophie de l'automatisation et des exigences en sûreté de fonctionnement. La synthèse se fait généralement de manière expérimentale par une stratégie essais-erreurs, qui est souvent basée sur la compétence et l'expérience du concepteur. Par conséquent, les détails nécessaires pour modéliser et commander ces systèmes ne permettent pas d'imposer le respect des spécifications. En théorie, les systèmes industriels (manufacturier, robotique, etc …) sont considérés comme des systèmes à événements discrets (SED), dont les variables d'état évoluent spontanément d'un état à l'autre, en fonction de l'occurrence d'événements instantanés (Wonham & Kai, 2017).

La modélisation des SED est basée sur trois concepts : fonction, structure et comportement. Pour comprendre ces concepts nous devons distinguer deux notions : l'état et la structure du SED. Un état rend la structure observable et fonctionnelle, puisque la structure du SED est représentée par ses états. La structure du SED est la manière dont le comportement du SED est réalisé (Wang et *al.,* 2013). Un comportement présente une relation agrégée avec un ensemble d'états, permettant de décrire le SED en termes de ces états admissibles ou interdits. Notoirement, le concept de fonction est l'utilité du comportement, perçu par l'utilisateur (Wang et *al.,* 2013). Le comportement dynamique renvoie à la relation d'une variable d'état avec une évolution de type temporel ou événementiel. La motivation de notre travail concerne les systèmes dynamiques de type événementiel, que nous distinguons des systèmes dynamiques de type temporel (Komenda et *al.,* 2009). Ainsi, la séquence des événements constitue le langage qui caractérise le comportement du SED, modélisable avec les outils tels que : les automates et les langages, les réseaux de Petri (RdP), les files d'attente, etc. (Cassandras & Lafortune, 2008). Le besoin de méthodes formelles pour la synthèse des SED a donné naissance à la théorie de commande par supervision (Ramadge & Wonham, 1983 ; Wonham, 2015) basée sur les automates à états finis et les langages. En principe, le comportement du système est donné par un langage, qui peut contenir des séquences d'événements inadmissibles pour certaines spécifications de fonctionnement qui lui sont imposées. Logiquement, les séquences inadmissibles conduisent le système vers certains états qui ne sont pas souhaitables, puisqu'ils violent les spécifications. Par conséquent,

l'accessibilité à ces états indésirables doit être empêchée par un contrôleur (superviseur), qui évolue en boucle fermée.

La synthèse du contrôleur consiste à déterminer l'ensemble des séquences admissibles pour le procédé qui appartiennent aussi à la spécification (Ramadge et *al*., 1989 ; Wonham, 2005). Le contrôleur intervient sur l'évolution du procédé en interdisant l'occurrence de certains événements contrôlables E_c du SED. L'existence des événements incontrôlables E_{uc} que le contrôleur ne peut pas empêcher pose un sérieux problème de contrôlabilité (Gaudin & Marchand, 2004). Ainsi, le contrôleur synthétisé est optimal au sens maximal permissif si et seulement si la condition de contrôlabilité, définit sur les langages, est vérifiée. La synthèse d'un tel contrôleur à partir des langages du SED est difficile (Kumar, 1991). Face à la difficulté de calculer le contrôleur à partir des langages formels, l'algorithme de Kumar (1991) fournit une méthode efficiente pour déterminer les états interdits dans l'espace d'état du SED, tout en vérifiant la condition de contrôlabilité à partir des modèles automates. L'inconvénient majeur est que le contrôleur est donné sous forme d'automate dont la taille peut être inexploitable, à cause de l'explosion combinatoire du nombre d'états inhérente (Yin & Lafortune, 2017 ; Yoo & Lafortune, 2002). Le nombre d'états à considérer ainsi que le manque de structure dans les modèles automates, limite la possibilité d'implantation de cette approche pour des SED réels.

Pour tirer avantage de la structure des SED, les formalismes comme les réseaux de Petri (RdP) sont réputés pour exprimer, plus facilement, les comportements dynamiques et les spécifications. Les RdP permettent une description compacte du SED, sans énumération de l'espace d'état entier (Yin &Lafortune, 2017). De plus, les RdP ont une grande similitude avec les automates et plusieurs avantages (Zhou & DiCesare, 1993 ; David & Alla, 2010) puisqu'ils génèrent les langages plus expressifs (Basile, 2006 ; Giua & Seatzu, 2007).
Des nombreux travaux (Seatzu et *al.*, 2012) ont porté sur le développement des méthodes basées sur les RdP, avec des résultats et des concepts très intéressants. Malheureusement, elles sont partiellement structurelles à cause de la nécessité de construire le graphe de marquage accessible, qui est sujet à l'explosion combinatoire des états. Par exemple, la méthode basée sur la théorie des régions (Badouel *et al.*, 1995; Ghaffari *et al*., 2003) qui prend en compte le concept de contrôlabilité est basée sur la détermination des états admissibles par construction du graphe de marquage. La synthèse du contrôleur est réalisée

par résolution d'un nombre important d'inégalités. La théorie de Holloway et Krogh (1990) exploite les graphes d'événements cyclique et contrôlé, qui ne permettent pas de modéliser les situations de conflits rencontrées dans les procédés. L'approche basée sur les RdP synchronisés et colorés (Godon, 1996), quant à elle permet de prendre en compte les spécifications comportementales. Cependant, la complexité de l'algorithme pour déterminer les états interdits peut conduire à l'énumération de tous les états. Une autre approche, basée sur la théorie d'invariant de places (Yamalidou et *al.*, 1996 ; Moody & Antsaklis, 1998), est simple et efficace pour synthétiser un contrôleur maximal permissif, si l'ensemble adéquat des contraintes admissibles lui est fourni (Kattan, 2004 ; Dideban, 2007; Vasiliu & Alla, 2011). Cependant, les spécifications sont définies par les états interdits qu'il faut obtenir par construction du graphe de marquage (Dideban 2005; Vasiliu *et al.*, 2009). Dans ces différentes méthodes, le graphe de marquage accessible du RdP est analysé pour identifier les états interdits et les blocages.

Notre hypothèse de travail est qu'il est possible d'établir un lien biunivoque entre la théorie de commande par supervision et la méthode des invariants de place, si la condition de contrôlabilité est définie à partir de la structure du RdP de commande du SED. Cette condition de contrôlabilité sera appelée condition de contrôlabilité structurelle. Notre objectif principal est de proposer une méthode intrinsèquement structurelle pour la synthèse des contrôleurs basés sur les RdP, applicable dans le contexte industriel, sans construction du graphe de marquage accessible. Notre méthode doit prendre en compte le problème de contrôlabilité et pallier aux difficultés rencontrées dans la synthèse d'un contrôleur maximal permissif pour des RdP ordinaires et généralisés. Il s'agit de la détermination des contraintes admissibles pour la méthode des invariants de place.

Notre contribution va consister à proposer une méthode qui exploite le produit synchrone des RdP pour déterminer la condition de contrôlabilité structurelle ; et qui garantit l'existence d'une solution de contrôle maximal permissif par la séparabilité des états du SED par un hyperplan séparateur. Il nous a paru nécessaire de commencer par une présentation des outils de modélisation des SED. C'est l'objet du premier chapitre où nous évoquons quelques notions relatives aux SED et aux langages formels. Ensuite, nous présentons les outils les plus usuels dans la commande par supervision des SED tels que les automates et les Réseaux de Petri et, les propriétés qui peuvent nous aider dans la suite de notre travail.

Le chapitre deux se concentre uniquement sur les détails relatifs à la théorie de commande par supervision des SED proposée par Ramadge et Wonham (1983), en rappelant tout d'abord les principes cette théorie. Dans le troisième chapitre, nous montrons en premier lieu, qu'il est possible d'avoir un contrôleur maximal permissif lorsque le SED est modélisé par les RdP. En second lieu, nous décrivons les méthodes les plus représentatives de synthèse de contrôleurs basées sur le RdP et leurs limites vis-à-vis de la théorie originale. Par ailleurs, la technique de simplification des contraintes basée sur les invariants est rappelé pour les besoins d'optimisation structurelle des contrôleurs. A la fin du chapitre, l'objectif de thèse est explicité pour relever la nécessite d'une méthode structurelle applicable aux SED réels. Au quatrième chapitre nous décrivons la méthode complètement structurelle de synthèse de contrôleurs basée sur une condition de contrôlabilité structurelle obtenue à partir des propriétés des RdP synchronisés et du produit synchrone des modèles du procédé et de la spécification. La preuve d'une équivalence entre la condition de contrôlabilité structurelle et la condition de contrôlabilité définie sur les langages est donnée, et une technique simplifiée de détermination des contraintes admissibles est présentée. Nous montrons comment la méthode synthèse des contrôleurs par les invariants de place est systématique et optimale dans la totalité des études de cas considérés. Enfin, nous terminons par une conclusion et les perspectives de ce travail.

Chapitre I

Modélisation des systèmes a événements discrets

Ce chapitre présente les notions et les outils de modélisation des systèmes à événements discrets (SED). Apres une présentation des propretés clés et des opérations de compostions nécessaires à dans la synthèse de la commande par supervision, une comparaison entre les deux outils les plus usuels est effectuée pour justifier le choix du réseau de Petri synchronisé dans ce travail.

Chapitre I

Modélisation des systèmes à événements discrets

I.1. Les systèmes à événement discrets

L'objectif de la théorie de commande par supervision est de développer un formalisme général pour modéliser, synthétiser et commander des systèmes dynamiques (Raisch, 2000 ; Wonham & Kai, 2017). Un système dynamique est un système dans lequel la sortie dépend généralement des valeurs passées et de l'entrée. En effet, l'état d'un tel système à un instant *t* doit décrire son comportement à cet instant. Dans la théorie de commande par supervision, le terme état a une signification beaucoup plus précise et constitue la pierre angulaire du processus de modélisation et des nombreuses techniques de synthèse de contrôleurs (Cassandras & Lafortune, 2008 ; Seatzu et *al*., 2012). L'espace d'état d'un système dynamique est l'ensemble de toutes les valeurs possibles que l'état peut prendre. Lorsque l'espace d'état du système est décrit naturellement par un ensemble discret et que les transitions d'état ne sont observées qu'à des instants discrets. Nous parlons d'un « système à événements discret » où les transitions d'états sont associées aux événements (Murata, 1989 ; Wonham, 2011).

Définition I.1.1 (Système à évènements discrets) Un système à évènements discrets est un système dynamique évoluant suite à l'occurrence d'événements, et dont l'espace des états est discret (Seatzu et *al*., 2012). La succession des événements constitue une trajectoire d'états qui évolue conformément à l'occurrence, à intervalles irréguliers généralement inconnus, des évènements physiques qui déterminent une transition d'état.

❑

Un système à événements discrets (SED) satisfait donc les deux propriétés suivantes :

- Espaces d'états et d'évènements discrets,
- Transition d'état piloté par des événements.

Le mot discret ne signifie ni « temps discret», ni «état discret» mais réfère au fait que la dynamique est régi par des événements dont les instants d'occurrence n'ont pas une importance fondamentale ; seul compte réellement l'ordre de l'occurrence de ces événements. Typiquement, les systèmes manufacturiers, les systèmes de transport, les systèmes de communication etc. dont les comportements sont basés sur l'occurrence d'événements asynchrones dans le temps, sont des SED (Yoo & Lafortune, 2002). L'état d'un tel système peut avoir des valeurs logiques ou symboliques, qui changent en réponse aux évènements. Ils peuvent également être décrits en termes logiques ou symboliques. Un événement peut être identifié avec une action spécifique prise (par exemple, appuyer sur un bouton). Il peut être considéré comme un événement spontané dicté par la nature (par exemple, une machine en panne), ou cela peut être le résultat de conditions satisfaites (par exemple, le niveau de fluide dans un réservoir). Dans notre travail, nous utiliserons le symbole *e* pour désigner un événement. Nous supposons que le comportement d'un SED est décrit en termes de séquences d'événements de la forme $e_1\ e_2 \cdot \cdot \cdot e_n$. En considérant que nous pouvons définir un ensemble d'événements *E* dont les éléments sont tous ces événements. Evidemment, *E* est un ensemble discret.

Exemple I.1

On considère un robot qui charge des pièces mécaniques sur un convoyeur. Le robot peut être dans trois états : en arrêt, en marche (chargement) et en panne (pièce mal positionnée). Les évènements qui régissent son évolution sont : *a* (saisir une pièce), *b* (pièces correctement positionnée), *c* (pièce mal positionnée) et *d* (pièces repositionnées). La figure I.1 représente une évolution possible du SED.

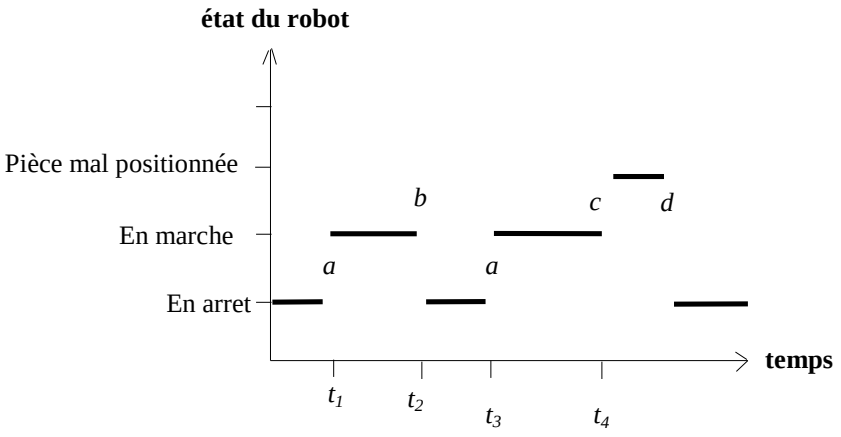


Figure I.1 – Chronogramme de l'évolution de l'état du robot

Dans l'état initial le robot est supposé être en arrêt. Au moment t_1 une pièce est saisie, l'événement *a* (début du chargement de pièce sur le convoyeur) se produit, et la machine évolue dans l'état de marche. Il y a deux évolutions possibles, à partir de cet état. Soit le chargement est effectué sans problème et le robot charge la pièce mécanique sur le convoyeur simultanément à l'occurrence de l'événement *b* (pièces correctement positionnée), soit le robot tombe en panne (pièce mal positionnée), fait signalé par l'occurrence de l'événement *c* (pièce mal positionnée). Depuis l'état de panne, le repositionnement des pièces par l'événement *d* ramène le robot à son état initial. Le robot mène à terme le premier cycle de chargement à l'instant t_2 (événement *b*), commence un nouveau cycle à l'instant t_3 (événement *a*), tombe en panne à l'instant t_4 (événement *c*) et est réparée à l'instant t_5 (événement *d*).

▲

Dans le cas où l'instant d'occurrence des événements est pertinent, l'évolution d'un SED peut être décrite par un ensemble de couples (*e, t*) où « *e* » représente un événement et « *t* » représente la date d'occurrence de cet événement. Dans notre exemple, l'évolution de l'état du robot peut être définie par les couples de couples (*e, t*) où « *e* » représente un événement et « *t* » représente la date d'occurrence de cet événement. Ainsi, l'ensemble ordonné de couples (a, t_1), (b, t_2), (a, t_3), (c, t_4), (d, t_5), etc…, constitue une séquence décrite à un niveau temporel. Dans un SED logique, le modèle ne précise pas les dates d'occurrences des évènements. Donc, une hypothèse simplificatrice est de ne considérer que l'ordre dans lequel les évènements se produisent, par exemple *abac . . . etc.*

En considérant l'évolution de l'état d'un SED, cette simplification est justifiée lorsque seule importe la succession d'états et les événements associés provoquant les transitions d'état. On ne s'occupe pas de la question de savoir quand le système entre dans un état particulier ou combien de temps le système reste à cet état. C'est un modèle simple sur lequel de nombreuses techniques formelles ont été développées. Il permet d'étudier les propriétés dynamiques indépendamment des dates : identifier les séquences admissibles d'opérations et vérifier l'absence de blocage, etc.

Nous nous appuierons sur cette compréhension de base afin d'étudier des modèles formels pour résoudre les problèmes d'analyse et de synthèse de commande par supervision. Dans ce cadre, les automates et les réseaux de Petri (RdP) sont les deux outils de modélisation les plus communs des systèmes à évènements discrets. Ces formalismes ont en commun le fait

l'ensemble des évènements constitue un alphabet et la séquence des évènements est un mot, qui permet de construire les langages du SED en utilisant une structure de transition d'état (Hopcroft & Ullman, 1979 ; Papadimitriou, 1986). Ces formalismes diffèrent par la façon dont ils représentent l'information d'état. Ils sont également susceptibles de diverses opérations de composition, ce qui permet de construire le modèle global d'un SED à partir de modèles élémentaires des composants du SED. Cela rend les automates et les réseaux de Petri pratiques pour la construction de modèles. Par ailleurs, ces formalismes de modélisation peuvent être non temporisés, temporisés (RdP Temporisé, Algèbre Max-Plus, ...) ou stochastiques (chaînes de Markov, files d'attentes,...), selon l'intérêt du niveau de modélisation. Dans ce travail, seul l'aspect non temporisé sera présenté et étudié.

I.2. Les langages formels

Les langages sont les outils de base de la théorie de commande par supervision des SED. Ce formalisme s'appuie sur des opérations sur les événements. On peut considérer l'ensemble des événements E comme un «alphabet» et des séquences (finies) des événements comme des «mots». Ainsi, un langage représente l'ensemble de tous les événements ordonnés qui pourraient se produire dans le SED. Pour en savoir plus sur théorie des langages formels, se référer à (Hopcroft & Ullman, 1979; Hopcroft et *al*., 2007 ; Cassandras & Lafortune, 2008).

I.2.1. Les alphabets et les mots

Définition I.2.1. (Alphabet) Un alphabet E est un ensemble fini non vide de symboles. Le nombre de symboles qu'il contient est appelé sa cardinalité et dénoté par $|E|$.

❑

Considérons l'alphabet de l'exemple I.1 : $E = \{a, b, c, d\}$. Il a une cardinalité $|E| = 4$. Il est composé des symboles a, b, c et d.

Définition I.2.2. (Mot) Un mot (une séquence) σ défini sur un alphabet E est une suite de symboles de E. Le nombre de symboles qui compose la suite est appelé longueur du mot (séquence) σ et dénoté par $|\sigma|$.

❑

Considérons les alphabets de l'exemple I.1, le mot $\sigma = abac$ défini sur E a longueur $|\sigma| = 4$. On dénote par $|\sigma|_e$ le nombre de fois que le symbole e est présent dans σ. Le mot vide, de longueur 0, est noté ε et est défini sur n'importe quel alphabet.

Définition I.2.3. (Ensemble de mots) On dénote E^* l'ensemble de tous les mots d'une longueur quelconque que l'on peut définir sur un alphabet E, par :

$$E^* = \bigcup_{i \geq 0} E^i \tag{I.1}$$

Où ∪ représente le symbole d'union, l'opération (*) étant la *fermeture de Kleene* (Giua, 2013).

❑

Etant donné l'alphabet $E = \{a, b, c, d\}$ pour déterminer l'ensemble E^* on peut énumérer tous les mots sur E, inclus le mot vide, dans un ordre croissant. Nous obtenons alors :

$E^* = \{\varepsilon, a, b, c, d, ab, ac, ad, bc, bd, abc, \ldots\}$

Nous remarquerons que, bien que E ait une cardinalité finie, l'ensemble E^* a toujours une cardinalité infinie.

Définition I.2.4. (Préfixe et suffixe) Si le mot $\sigma \in E^*$ peut s'écrire $\sigma = uvz$ où $u, v, z \in E^*$, alors le mot u est appelé préfixe de σ, le mot v est appelé sous-chaine de σ et le mot z est appelé suffixe de σ.

❑

Définition I.2.5. (Langage) Un langage L défini sur un alphabet E est l'ensemble fini des mots formés à partir des événements dans E. C'est-à-dire, tout sous-ensemble de mots ou séquences de E^*. Sa cardinalité, à savoir le nombre de mots qu'il contient, est notée $|L|$.

❑

Par exemple, soit l'alphabet $E = \{a, b, c, d\}$, on considère les langages suivants :

- $L_1 = \{\varepsilon, a\}$, constitué de 2 mots seulement
- $L_2 = \{a, ab, aba, acd\}$, constitué de 4 mots.

Un langage peut être décrit explicitement en énumérant tous les mots ou en utilisant une notation ensembliste.

Remarques I.1. Nous appelons langage infini, tout langage comportant un nombre infini de mots. $L = E^*$ est un langage infini. Nous pouvons définir le *préfixe-clôture* d'un langage L,

comme le langage contenant tous les préfixes des mots ou séquences de *L*. Nous noterons $\bar{L}$, le préfixe-clôture du langage *L*.

$\bar{L} = \{\sigma_1 \in E^* | \exists\, \sigma_2 \in E^*, \sigma_1\sigma_2 \in L\}$

❑

I.2.1. Les opérations sur les langages

Les opérations binaires usuelles, telles que l'union et l'intersection peuvent être appliquées aux langages.

Définition I.2.6. (Intersection et union) Soit $L_1 \subseteq E_1^*$ et $L_2 \subseteq E_2^*$ deux langages. Les ensembles $\bar{E} = E_1 \cap E_2$ et $E = E_1 \cup E_2$ sont l'intersection et l'union de leurs alphabets respectivement. On définit les langages suivants :

- L'intersection de L_1et L_2, notée $L_1 \cap L_2 = \{\sigma \in \bar{E}^* | \sigma \in L_1, \sigma \in L_2\}$
- L'union de L_1 et L_2 , notée $L_1 \cup L_2 = \{\sigma \in E^* | \sigma \in L_1 \text{ ou } \sigma \in L_2\}$

❑

Exemple I.2

$L_1 = \{\varepsilon, a\}$ et $L_2 = \{a, ab, aba, acd\}$, donc $L_1 \cap L_2 = \{a\}$ et $L_1 \cup L_2 = \{\varepsilon, a, ab, aba, acd\}$

▲

L'opération de concaténation, défini sur les mots, peut être redéfinie en tant qu'opération sur les langages.

Définition I.2.7. (Concaténation) Étant donnés deux langages L_1, $L_2 \subseteq E^*$ on définit la concaténation de L_1 et L_2 comme le langage

$L_1L_2 = \{\sigma_1\sigma_2 \in E^* | \sigma_1 \in L_1, \sigma_2 \in L_2\}$

Il comprend tous les mots qui sont la concaténation d'une séquence en L_1, avec une séquence en L_2

❑

Exemple I.3

$L_1 = \{\varepsilon, a\}$ et $L_2 = \{a, ab, aba, acd\}$, donc $L_1 L_2 = \{a\}$ et $L_1 \cup L_2 = \{a, ab, aba, acd, aa, aab, aaba, aacd\}$

▲

Une opération unaire sur les langages est l'étoile de Kleene.

Définition I.2.8. (Etoile de Kleene) Étant donné un langage $L \subseteq E^*$, son étoile de Kleene (parfois appelée fermeture de Kleene) est le langage

$$L^* = \bigcup_{k=0}^{\infty} L^K \tag{I.2}$$

Il est constitué de tous les mots obtenus par concaténation de mots de L en un nombre quelconque de fois, y compris zéro.

❑

Un autre type d'opération fréquemment utilisé sur les langages est la projection naturelle désignée par la lettre P; un indice i est généralement ajouté pour spécifier les ensembles E_i pour des raisons de clarté lorsqu'il s'agit de plusieurs ensembles. Supposons E_1 et E_2 deux alphabets (pas nécessairement disjoints) fixés avec $E_2 \subseteq E_1$.

Commençons par définir la projection naturelle sur les mots par :

- $P_i: (E_1 \cup E_2)^* \rightarrow E_i^*$, $i = 1,2$

et l'application inverse par :

- $P_i^{-1}: E_i^* \rightarrow (E_1 \cup E_2)^*$, $i = 1,2$

Où, $P_i(\varepsilon) := \varepsilon$ et $P_i(\sigma) := \begin{cases} \sigma \text{ si } \sigma \in E_i, \\ \varepsilon \text{ si } \sigma \notin E_i, \end{cases}$

$P_i(\mu\sigma) := P_i(\mu)\, P_i(\sigma)$ *pour* $\mu \in (E_1 \cup E_2)^*$, $\sigma \in E_1 \cup E_2$,

$P_i^{-1}(\lambda) := \{\mu \in (E_1 \cup E_2)^* \mid P_i(\mu) = \lambda\}$

L'opération de projection prend un mot formé à partir du plus grand ensemble d'événements E_1 et efface les événements qui n'appartiennent pas à l'ensemble d'événements le plus petit E_2.

Définition I.2.9. (Projection) Soit le mot $\sigma \in E_1^*$ et un alphabet $E_2 \subseteq E_1$. La projection de σ sur E_2, notée $P_2(\sigma)$, est la suite obtenue de σ en gardant les symboles qui appartiennent à E_2 et effaçant les autres symboles.

❑

La projection P et son inverse P^{-1} sont étendues aux langages en les appliquant simplement à tous les mots du langage.

Pour $L \subset (E_1 \cup E_2)^*$ et $L_i \subset E_i$,

- $P_i(L) := \{\sigma \in E_i^* : (\exists \mu \in L), P_i(\sigma) := \mu\}$,

- $P_i^{-1}(L) := \{\mu \in (E_1 \cup E_2)^* : (\exists \sigma \in L), P_i(\mu) := \sigma\}$

La dernière opération que nous considérons, appelée produit synchrone, joue, comme nous le verrons, un rôle important lors de la description du comportement d'un SED composé de plusieurs sous-systèmes. Le produit synchrone est généralement défini sur des langages et nécessite des opérations de projection et de projection inverse (Seatzu et *al*., 2012).

Définition I.2.10. (Produit synchrone) Soient deux langages $L_1 \subseteq E_1^*$ et $L_2 \subseteq E_2^*$ soit $E = E_1 \cup E_2$ l'union de leurs alphabets. Le produit synchrone de L_1 et L_2 est le langage

$$L_1 || L_2 = \{\sigma \in E^* | P_1(\sigma) := \sigma_1 \in L_1, P_2(\sigma) := \sigma_2 \in L_2\},$$

Il est composé de toutes les séquences (mots) sur E dont la projection sur E_1 est une séquence σ_1 de L_1 et la projection sur E_2 est une séquence σ_2 de L_2

❑

En particulier, si les alphabets des deux langages L_1 et L_2 sont les mêmes, c'est-à-dire $E_1 = E_2 = E$, alors pour chaque $\sigma \in E^*$; nous avons $P_1(\sigma) := P_2(\sigma) := \sigma$ et l'opération de produit synchrone est équivalente à l'intersection des langages.

$$L_1 || L_2 = \{\sigma \in E^* | P_1(s) := \sigma \in L_1, P_2(\sigma) := \sigma \in L_2\} = L_1 \cap L_2 \quad \text{(I.3)}$$

L'opération de produit synchrone est associative et commutative et par conséquent, elle peut naturellement être étendue à plus de deux langages.

I.2.3. Langages réguliers

Les langages réguliers sont d'un intérêt tout particulier pour la modélisation des SED. Ils peuvent être représentés de façon concise par des expressions régulières. Une expression régulière sur un alphabet E est une expression dont les opérandes sont des symboles de E, et dont les opérateurs sont pris dans l'ensemble {+, •, *}. Les opérateurs +, • et * sont interprétés respectivement comme les opérateurs d'union, de concaténation et de *fermeture de Kleene* sur des langages.

Définition I.2.11. (Langage régulier) Nous appelons langage régulier, tout langage qui peut être défini par une expression régulière.

❑

Les expressions régulières $E_1 = a.b^*$et $E_2 = (a.b)^*$ définissent des langages différents : L_1 = {*ε, a, b, ab, abb, abbb...*} et, respectivement, L_2 = {*ε, ab, abab, ababab, …*}. Le *théorème de Kleene* (Hopcroft et *al*., 1979) établit l'équivalence entre les langages réguliers et les langages modélisables par des automates finis : il existe toujours un automate qui génère un langage régulier donné et, à l'inverse, le langage d'un automate s'exprime toujours par une expression régulière.

I.3. Les automates à états finis

Les automates à états finis sont un modèle capable de représenter les langages selon des règles bien définies et qui est utilisé à juste titre pour décrire les SED. Il peut être vu comme une machine à états (Raisch, 2000 ; Wonham , 2008) représentée par un ensemble d'états liés par des transitions associées à des événements et ayant des entrées et des sorties discrètes. Un automate est dit à états finis si le nombre d'états est borné. Nous parlons d'automate à états infinis lorsque l'ensemble d'états est infini. Si le prochain état de l'automate peut être déterminé sans équivoque à partir de son état courant et du symbole d'entrée, alors l'automate est appelé déterministe. Autrement, l'automate est dit non-déterministe. Dans cette section, un modèle particulier, appelé automate fini déterministe, est présenté. Un automate à états finis et déterministe (AFD) peut être défini formellement comme suit (Hopcroft et *al*., 2007 ; Cassandras & Lafortune, 2008)

Définition I.3.1. (Automates à états finis) Un automate (à états finis) déterministe est un 5-tuplet $\mathcal{A} = (Q, E, \delta, q_0, Q_m)$ où :

– Q est un ensemble fini d'états ;

– E est un alphabet ;

– $\delta : Q \times E \rightarrow Q$ est une fonction de transition d'état ;

– $q_0 \in Q$ est un état initial ;

– $Q_m \subseteq Q$ est un ensemble d'états finaux (ou états marqués).

❑

Un automate ne comportant pas de sorties est appelé accepteur : $\mathcal{A} = (Q, E, \delta, q_0)$. Un tel modèle ne comporte pas d'état final. Il faut noter que dans un automate, l'alphabet représente l'ensemble des évènements tandis que la fonction de transition spécifie la dynamique de l'automate. $q_k = \delta(q_i , e)$ exprime le fait l'occurrence de l'évènement e lorsque l'état actuel de

l'automate est q_i , conduit à l'état q_k. Ainsi, un automate peut être décrit par son graphe de transitions d'états, dans lequel chaque état correspond à un nœud et est représenté par un cercle : en particulier, l'état initial est représenté par un cercle avec une flèche d'entrée, et un état final par un double cercle. Si $q_k = \delta(q_i , e)$, il y aura une flèche dirigée à partir du nœud q_i au nœud q_k , étiquetée avec le symbole e pour représenter la transition de q_i à q_k , et cette flèche est appelée transition.

Considérons le SED de l'exemple I.1 dans lequel l'ensemble des évènements est E= {a, b, c, d}. Son graphe de transition est donné à la figure I.2, où les nœuds représentent les états et les arcs étiquetés représentent les transitions entre états. Les arcs du graphe fournissent une représentation graphique de la fonction de transition de l'automate,

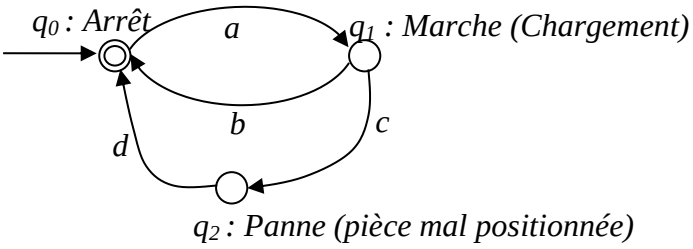


Figure I.2 – Graphe de transition d'états de l'exemple I.1

Ce graphe fournit une description de la dynamique d'un automate : L'ensemble des nœuds est l'ensemble d'états de l'automate, Q ={q_0, q_1, q_2}. L'état initial q_0 est indiqué par une flèche entrante. Les états finaux sont représentés par des doubles cercles, ainsi : Q_m = {q_0}. Les étiquettes des transitions sont les éléments de l'ensemble d'évènements (alphabet) E de l'automate. La fonction de transition δ est une fonction partielle, à savoir, pour certains $q_k \in Q$ et certains $e \in E$, il peut ne pas exister un évènement e en sortie de l'état q_k . Notons, cependant, que l'on ne peut pas avoir plusieurs transitions avec la même étiquette en sortie d'un état q_k .

Remarques I.2. La fonction de transition d'état δ peut etre étendue pour associer un état d'arrivée q_k, à tout état de départ q_i et à toute séquence σ_j ; $\delta : Q \times E^* \rightarrow Q$, $q_k = \delta(q_i , \sigma_j)$.

❑

I.3.1. Langages engendrés par les automates

La relation entre les langages et les automates est réalisée en inspectant le diagramme de transition d'état d'un automate. Considérons tous les chemins orientés qui peuvent être suivis dans le diagramme de transition d'état, en commençant à l'état initial. Considérons parmi eux tous les chemins qui se terminent dans un état marqué. Ceci conduit aux notions de langages générés et marqués par un automate.

Définition I.3.2. (Langages générés et marqués)

Le langage généré par $\mathcal{A} = (Q, E, \delta, q_0, Q_m)$ est

- $L(\mathcal{A}) := \{\sigma \in E^* : \delta(q_0, \sigma) \text{ est défini}\}$

Le langage marqué par est

- $L_m(\mathcal{A}) := \{\sigma \in L(\mathcal{A}) : \delta(q_0, \sigma) \in Q_m\}$

❑

Le langage $L(\mathcal{A})$ est un langage régulier qui représente l'ensemble de toutes les séquences qui permettent de rejoindre un état quelconque de l'automate à partir de son état initial. $L(\mathcal{A})$ est préfixé par définition, puisque un chemin n'est possible que si tous ses préfixes sont également possibles. Si δ est une fonction totale sur son domaine, alors nécessairement $L(\mathcal{A}) = E^*$. Le langage $L_m(\mathcal{A})$, est le sous-ensemble de $L(\mathcal{A})$ constitué uniquement de séquences σ correspondent à des chemins qui se terminent à un état marqué dans le diagramme de transition d'état. Puisque tous les états de Q ne sont pas nécessairement marqués, le langage $L_m(\mathcal{A})$ n'est pas nécessairement préfixé. Une séquence σ est la concaténation des étiquettes d'événements des transitions composant un chemin. Par conséquent, une séquence σ est dans $L(\mathcal{A})$ si et seulement si elle correspond à un chemin admissible dans le diagramme de transition d'état. Une séquence σ est acceptée (ou reconnue) par $\mathcal{A}$ si, en franchissant les transitions d'état impliquées par σ depuis l'état initial de l'automate, un état final $q_k \in Q_m$ est atteint. L'ensemble des mots acceptés par un automate constitue le langage accepté par l'automate. C'est pourquoi, le langage marqué est également appelé le langage accepté (ou reconnu) par l'automate.

Remarques I.3. La classe des langages acceptés (ou reconnus) par les automates finis construits sur un alphabet E est égale à la classe des langages réguliers sur E.

❑

Définition I.3.3. (État accessible et co-accessible) Soit l'automate fini déterministe $\mathcal{A} = (Q, E, \delta, q_0, Q_m)$. On appelle état accessible de $\mathcal{A}$ tout état $q_k \in Q$ qui peut être atteint par l'automate depuis son état initial et en franchissant les transitions d'état impliquées par une séquence $\sigma \in E^*$. Un état $q_k \in Q$ est co-accessible, s'il y a un chemin dans le diagramme de transition d'état tel qu'à partir de cet état l'automate peut atteindre un état marqué. Par extension, l'automate $\mathcal{A}$ est dit accessible si tous ses états sont accessibles et coaccessible si tous ses états sont coaccessibles.

❑

Un automate peut atteindre un état q_k tel qu'aucun évènement ne peut être exécuté. Alors, le système est bloqué parce qu'il est entré dans un état de blocage (Ezpeleta et *al.*, 1995 ; Lautenbach & Ridder, 1996), sans avoir terminé sa tâche. Un autre problème à considérer c'est quand il y a un ensemble d'états non marqués dans $\mathcal{A}$ qui sont accessibles l'un à l'autre mais sans qu'aucune transition ne sorte de cet ensemble. Nous avons un problème de vivacité et le système est bloqué lorsqu'il entre dans cet ensemble L'évidence et l'importance de ces problèmes dans les SED nous amène à formuler cette définition.

Définition I.3.4. (Blocage) L'ensemble d'états accessibles de $\mathcal{A}$ est $Q_a = \{q_i \in Q \mid \exists\ e \in E^*, \delta(q_0, e) = q_i\}$; $\mathcal{A}$ est accessible si $Q_a = Q$. L'ensemble d'états co-accessibles est $Q_{ca} = \{ q_i \in Q \mid \exists\ e \in E^*, \delta(q_i, e) \in Q_m\}$; $\mathcal{A}$ est accessible si $Q_{ca} = Q$. Nous disons que l'automate $\mathcal{A}$ est non bloquant si chaque état accessible est co-accessible, c'est-à-dire $Q_a = Q_{ca}$.
En termes de langage, L'automate $\mathcal{A}$ est bloquant si $\overline{L}_m(\mathcal{A}) \subset L(\mathcal{A})$. En particulier $\mathcal{A}$ est non bloquant quand $\overline{L}_m(\mathcal{A}) = L(\mathcal{A})$.

❑

Ainsi, si un automate est bloquant, cela signifie qu'un blocage peut se produire. D'où la nécessité de garantir par le contrôle que $\overline{L}_m(\mathcal{A}) = L(\mathcal{A})$. C'est l'objet de la théorie de commande par supervision présentée au chapitre II.

I.3.2. Composition des automates

La modélisation d'un SED par des automates a besoin d'opérations qui permettent de modifier de façon appropriée le diagramme de transition d'état en fonction, par exemple, d'une opération de langage que l'on souhaite effectuer. Ce paragraphe présente deux opérations sur

les automates: le produit, noté x, et la composition parallèle, notée ||. La composition parallèle est souvent appelée produit synchrone et le produit est parfois appelé produit totalement synchrone (Fabre et *al*., 2005 ; Zhou et *al*., 2008). Ces opérations permettent de combiner plusieurs automates, de sorte que le modèle global du SED puisse être construit à partir du comportement conjoint d'un ensemble d'automates élémentaires fonctionnant simultanément. Ces opérations sont effectuées sur un type spécial d'automate ne comportant pas de sorties, appelé accepteur : $\mathcal{A} = (Q, E, \delta, q_0)$. Le mot *automate* sera utilisé en lieu et place d'accepteur.

Soit deux automates $\mathcal{A}_1$ et $\mathcal{A}_2$ définis respectivement par : $\mathcal{A}_1 = (Q_1 , E_1 , \delta_1 , q_{01})$ et $\mathcal{A}_2 = (Q_2 , E_2 , \delta_2 , q_{02})$. Les deux types d'interconnexion des composants $\mathcal{A}_1$ et $\mathcal{A}_2$ du SED avec les ensembles d'événements E_1 et E_2 sont $\mathcal{A}_1 \times \mathcal{A}_2$ et $\mathcal{A}_1 || \mathcal{A}_2$. La différence majeure entre les deux opérations se rapporte à la façon dont les événements privés, qui n'appartiennent pas à $E_1 \cap E_2$, sont gérés.

a) Produit synchrone de deux automates

Le produit *synchrone* est effectué quand les alphabets des langages associés aux automates considérés ont au moins un événement en commun.

Définition I.3.5. (Produit synchrone) Le produit synchrone entre $\mathcal{A}_1$ et $\mathcal{A}_2$ est l'automate (l'accepteur) $\mathcal{A} = \mathcal{A}_1 || \mathcal{A}_2 = (Q, E, \delta, q_0)$ où

- $Q = Q_1 \times Q_2$ est l'ensemble des états,
- $E = E_1 \cup E_2$ représente l'alphabet,
- $q_0 = (q_{01}, q_{02})$ est l'état initial,
- $\delta : Q \times E$

Pour tout $q = (q_1, q_2) \in Q$ et pour tout événement $e \in E$, la fonction de transition δ est définie de la façon suivante :

- Pour $e \notin E_1 \cap E_2$:

$$\delta((q_1, q_2), e) = (q_1', q_2) \text{ si } \delta_1(q_1, e) = q_1' \text{ et } e \in E_1$$

$$\delta((q_1, q_2), e) = (q_1, q_2') \text{ si } \delta_2(q_2, e) = q_2' \text{ et } e \in E_2$$

- Pour $e \in E_1 \cap E_2$:

$$\delta((q_1, q_2), e) = (q_1', q_2') \text{ si } \delta_1(q_1, e) = q_1' \text{ et } \delta_2(q_2, e) = q_2'$$

❑

Dans le produit synchrone, un événement commun appartenant à $E_1 \cap E_2$, ne doit se produire de manière synchrone que si les deux automates l'exécutent simultanément. Les événements privés de $(E_2 \backslash E_1) \cup (E_1 \backslash E_2)$, peuvent se produire autant que possible.

Exemple I.4. (Giua, 2017)

Considérons un atelier de production composé d'un robot et d'une zone de stockage qui contient deux places disponibles (on dit que sa capacité est 2). Le robot choisit des objets dans un convoyeur qui est toujours plein (évènement *a*) et les dépose dans la zone de stockage (évènement *b*). Les objets peuvent ensuite être retirés (évènement *c*) de la zone de stockage. La figure I.3 représente la disposition de cette structure. L'Automate avec alphabet $E_1 = \{a, b\}$ dans la figure I.4(a) modélise le robot : q_{10} désigne l'état robot inoccupé, q_{11} désigne l'état robot en fonctionnement. L'automate avec alphabet $E_2 = \{b, c\}$ dans la figure 1.4 (b) modélise la zone de stockage : chaque état q_{2k} désigne le nombre k d'objets qu'elle contient (pour $k = 0, 1, 2$). Le modèle du convoyeur n'est pas pris en compte parce qu'il est toujours plein. Si le robot est inoccupé, il peut prendre un objet, et cela indépendamment de l'état de la zone de stockage. Pareil, si la zone de stockage n'est pas vide, un objet peut en être retiré et cela indépendamment de l'état du robot. Ceci est le résultat du fait que *a* est un évènement privatif de $\mathcal{A}_1$, et que *c* est un évènement privatif de $\mathcal{A}_2$. A l'inverse, le dépôt d'un objet dans la zone de stockage peut seulement avoir lieu si le robot a choisi un objet (état q_{11}) et si la zone de stockage n'est pas pleine (état q_{20} ou q_{21}).

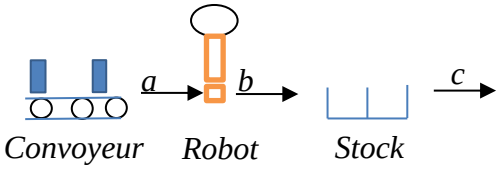
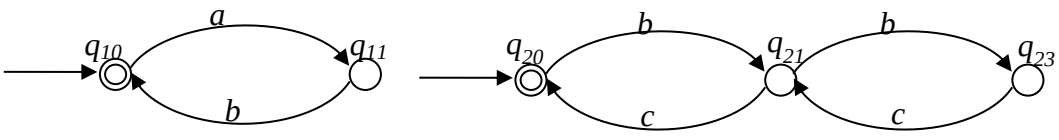


Figure I.3 – Disposition de l'atelier de production



(**a**) (**b**)

Figure I.4 –. (a) Modèle du robot. (b) Modèle de la zone de stockage.

Dans l'automate $\mathcal{A} = \mathcal{A}_1 || \mathcal{A}_2$, un état q_{ij} correspond à un couple (q_i, q_j) où q_i est un état de $\mathcal{A}_1$ et q_j est un état de $\mathcal{A}_2$. Ce modèle permet d'exprimer la conjonction des contraintes de fonctionnement définies par $\mathcal{A}_1$ et par $\mathcal{A}_2$. Il est construit sur un alphabet qui est la réunion des alphabets de $\mathcal{A}_1$ et $\mathcal{A}_2$, c'est-à-dire : $\{a, b, c\}$, où b est un évènement qui est en commun dans les alphabets de $\mathcal{A}_1$ et $\mathcal{A}_2$.

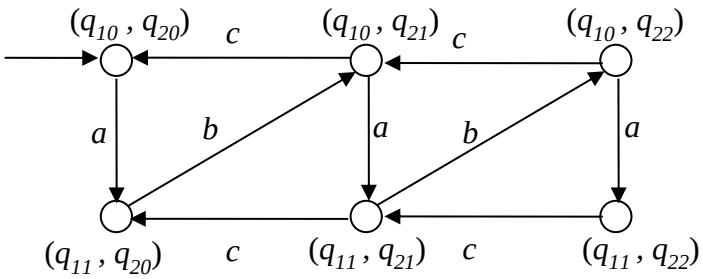


Figure I.5 –. Modèle automate complet (synchrone) de l'atelier de production

▲

Si $E_1=E_2=E$, alors le produit synchrone $\mathcal{A}_1||\mathcal{A}_2$ se réduit au produit totalement synchrone $\mathcal{A}_1 \times \mathcal{A}_2$, puisque tous les événements doivent être synchronisés. Le produit totalement synchrone ou composition par produit est restrictif car il ne permet que des transitions sur des événements communs. La relation de composition de $\mathcal{A}_1$ et $\mathcal{A}_2$ se réduit à :

- $(\delta_1 \text{ x } \delta_2)((q_1, q_2), e) = (q_1', q_2') =$ si $\delta_1(q_1, e) = q_1'$ et $\delta_2(q_2, e) = q_2'$ pour $e \in E$ et $(q_1, q_2) \in (Q_1, Q_2)$
- *Et non définie sinon*

Dans la commande par supervision, ce type de produit complètement synchrone se rencontre lors du fonctionnement en boucle fermée du procédé avec son contrôleur (Gaudin & Marchand, 2004).

b) Produit asynchrone

Si $E_1 \cap E_2 = \emptyset$ et $E = E_1 \cup E_2$, alors il n'y a pas d'événements synchronisés et $\mathcal{A}_1 || \mathcal{A}_2$ est le comportement simultané de $\mathcal{A}_1$ et $\mathcal{A}_2$. Nous parlons alors de produit asynchrone, est réalisée quand les alphabets des langages associés aux automates considérés n'ont aucun événement en commun.

Définition I.3.6. (Produit asynchrone) Le produit asynchrone de $\mathcal{A}_1$ et $\mathcal{A}_2$, est défini par le 5-uplet : $(Q_1 \times Q_2, E_1 \cup E_2, \delta_1 \times \delta_2, q_{01} \times q_{02})$, avec

- $(\delta_1 \text{ x } \delta_2)((q_1, q_2), e) = (q_1', q_2)$ si $\delta_1 (q_1, e) = q_1'$ pour $e \in E_1$
- $(\delta_1 \text{ x } \delta_2)((q_1, q_2), e) = (q_1, q_2')$ si $\delta_2 (q_2, e) = q_2'$ pour $e \in E_2$

❑

Le produit asynchrone n'est qu'un cas particulier du produit synchrone.

Exemple I.5.

Considérons le système de fabrication manufacturier composé de deux machines identiques M1 et M2 (figure I.6). Les deux machines travaillent de façon indépendante, puisent des pièces brutes en amont et rejettent des pièces usinées en aval.

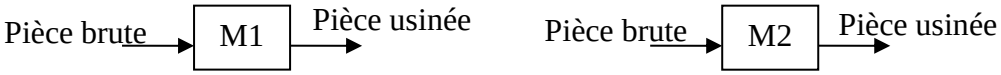


Figure I.6 – Système manufacturier classique (Ramadge et Wonham, 1989)

Nous pouvons associer à chaque machine de ce SED un automate (accepteur) décrivant son comportement, avec ses événements associés.

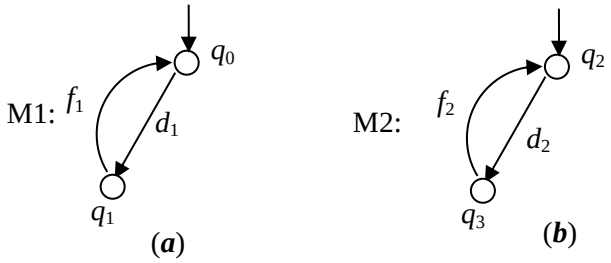


Figure I.7 – Modèles automates des machines M1 et M2

La machine M2 possède un fonctionnement similaire à celui de la machine M1. Notons respectivement E_1 et E_2, les alphabets des machines M1 et M2. Nous avons : $E_1 = \{d_1, f_1\}$ et $E_2 = \{d_2, f_2\}$. Les événements d_1 et d_2 respectivement, modélisent le début de cycle, qui est simultané avec la prise de pièce en amont. Les événements f_1 et f_2 respectivement, modélisent la fin de cycle qui est simultanée avec le dépôt d'une pièce en aval. Le fonctionnement du

système manufacturier est défini sur un alphabet $E = E_1 \cup E_2$ et le modèle automate global peut être obtenu en effectuant le produit asynchrone des modèles M1 et M2 (figure I.8).

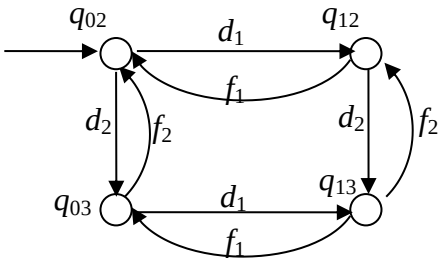


Figure I.8 – Produit asynchrone des modèles M1 et M2 de l'exemple …

Dans l'automate global, un état noté q_{ij}, est un couple (q_i , q_j), où q_i est un état de M1 et q_j est un état de M2.

▲

I.3.3. Langages du produit synchrone des automates

Soient $\mathcal{A}_1$ et $\mathcal{A}_2$ deux automates. Pour caractériser les langages résultants du produit synchrone $\mathcal{A}_1||\mathcal{A}_2$ en fonction des langages de $L(\mathcal{A}_1)$ et $L(\mathcal{A}_2)$, considérons la projection inverse de langages introduite dans la Section I.2. En considérant que le plus grand ensemble d'événements est $E_1 \cup E_2$, nous avons :

$$L(\mathcal{A}_1||\mathcal{A}_2) = P_1^{-1}[L(A_1)] \cap P_2^{-1}[L(A_2)]$$

$$L_m(\mathcal{A}_1||\mathcal{A}_2) = P_1^{-1}[L_m(A_1)] \cap P_2^{-1}[L_m(A_2)] \quad \text{(I.4)}$$

Dans le cas particulier où les alphabets E_1 et E_2 sont identiques, une séquence est reconnue par A si, et seulement si, elle est reconnue à la fois par $\mathcal{A}_1$ et par $\mathcal{A}_2$:

$$L(A_1||A_2) = L(A_1) \cap L(A_2)$$

$$L_m(A_1||A_2) = L_m(A_1) \cap L_m(A_2) \quad \text{(I.5)}$$

Les preuves formelles de ces résultats peuvent être trouvées dans (Hopcroft & Ullman, 1979). Le produit synchrone des automates occupe une place essentielle dans la théorie de commande par supervision des SED: il y a produit synchrone du procédé avec les spécifications (Wonham, 2011).

La modélisation de SED réels à l'aide d'automates est généralement difficile à mettre en œuvre du fait de l'explosion combinatoire du nombre d'états inhérente aux modèles à

automates (Vasiliu, 2012). Le paragraphe suivant présente les réseaux de Petri qui évitent cet inconvénient et conduit à des modèles compacts concis pour des systèmes de grande taille.

I.4. Les réseaux de Petri

Les réseaux de Petri (RdP) constitue une alternative aux automates dans le sens où ils représentent explicitement la fonction de transition du SED (David & Alla, 2010). Ce formalisme s'adapte parfaitement à la description des SED avec l'avantage d'être un modèle structurel plus compact, qui manipule les événements selon certaines règles. Il peut représenter une plus grande classe de langages (Hopcroft et *al*., 2007). La littérature (Giua & Seatzu, 2007 ; David & Alla, 2010) présente plusieurs classes de RdP dont l'analyse structurelle inclut l'analyse de l'accessibilité de manière similaire aux automates. La classe de RdP qui nous intéresse dans cette thèse est le RdP synchronisé où à chaque transition est associée un événement. Les concepts de produit synchrone, de graphe de marquages accessibles et de langages d'un RdP présentés dans cette section sont importants pour la synthèse de commande par supervision par les RdP.

I.4.1. Notions fondamentales

Un réseau de Petri (RdP) est un graphe orienté comportant un ensemble fini de places (symbolisées par des cercles) et un ensemble fini de transitions (symbolisées par des traits), avec des arcs orientés qui assurent la liaison d'une place vers une transition ou d'une transition vers une place. Les transitions, les places et les arcs reliant celles-ci définissent les composants de base de la structure graphique d'un RdP. L'ensemble des places permet de représenter l'état du système et l'ensemble des transitions sont associées à des événements dont l'occurrence provoque le changement d'état du SED (Ferrier & Boimond, 2013). Un exemple de RdP est illustré à la figure I.9. C'est le modèle du robot qui charge des pièces mécaniques sur un convoyeur présenté dans l'exemple I.1.

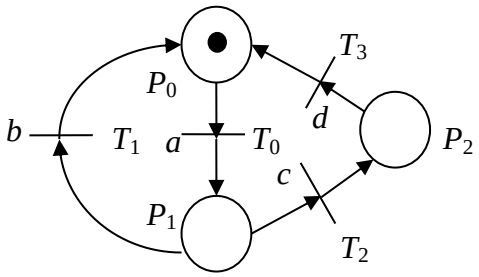


Figure I.9 – Modèle RdP du robot de l'exemple I.1.

L'ensemble des places $P = \{P_0, P_1, P_2\}$, où P_0 représente l'état d'arrêt, P_1 l'état de marche (chargement), et P_2 l'état de panne (pièce mal positionnée). L'ensemble des transitions $T = \{T_0, T_1, T_2, T_3\}$ correspond, respectivement, aux changements d'état associés aux événements *a* (début du chargement de pièce), *b* (pièces correctement positionnée), *c* (pièce mal positionnée) et *d* (repositionnement des pièces).

Définition I.4.1. (Réseau de Petri) Un réseau de Petri est un 4-uplet $R = \langle P, T, W^-, W^+ \rangle$ où :

$P = \{p_1, p_2, \ldots, p_n\}$ est un ensemble fini et non vide de places,
$T = \{t_1, t_2, \ldots, t_m\}$ est un ensemble fini de transitions,
W^-(resp. W^+) est la fonction d'incidence avant (resp. arrière) de domaine $P \times T$ et de co-domaine $\mathbb{N}$.

❑

Définition I.4.2. (RdP marqué) Un RdP marqué est un 5-uplet $R = \langle P, T, W^-, W^+, M_0 \rangle$, où :

$\langle P, T, W^-, W^+ \rangle$ est un RdP et $M_0 \in \mathbb{N}^m$ est le marquage initial de R définissant ainsi l'état initial du RdP . $W^-(\bullet, t_j)$ (resp. $W^+(\bullet, t_j)$) est la fonction d'incidence avant (resp. arrière) de domaine $P \times T$ et de co-domaine $\mathbb{N}$. Habituellement, $\mathbb{N}$ dénote l'ensemble des entiers non négatifs.

❑

La notation $(\bullet, t_j)$ définit le nombre minimum de marques ou jetons requis dans la place p_i pour valider la transition t_j. Les places et les transitions sont connectées par les arcs orientés $(\bullet, t_j)$. Le nombre de jetons qu'un arc singulier peut transférer définit son poids.

Définition I.4.3. (Poids) Le poids d'un arc est définit de la manière suivante : si $(\bullet, t_j) = r$, où $r > 1$ est un entier, un arc orienté à partir de la place p_i vers la transition t_j est dessiné avec le

poids *r*. Si $r = 1$, un arc non pondéré est dessiné et s'il arrive que $r = 0$, alors aucun arc n'est dessiné. ❑

Les fonctions d'incidence avant et arrière sont des matrices qui peuvent être synthétisées en une seule, à savoir, la matrice d'incidence**.**

Définition I.4.4. (Matrice d'incidence) La matrice d'incidence d'un réseau de Petri est la matrice entière *W* définie par :

$$\forall(\bullet, t_j) \in P \times T,\ W(\bullet, t_j) = W^{+}(\bullet, t_j) - W^{-}(\bullet, t_j). \qquad \text{(I.6)}$$

❑

Dans cette définition, $W^{-}(\bullet, t_j)$ est la précondition associée à la transition t_j et la place p_i ; elle définit le nombre minimal de jetons ou marques dans la place nécessaire au franchissement de la transition *t* et retirées de p_i en cas de franchissement. De même, $W^{+}(\bullet, t_j)$ est la postcondition associée à la transition t_j et la place P_i ; elle définit le nombre de marques apportées à p_i par le franchissement de t_j. L'évolution d'un SED modélisé par RdP est caractérisée par le franchissement des transitions. Cependant, pour qu'une transition puisse se produire, il faut que toutes les conditions nécessaires au franchissement soient satisfaites. Les informations concernant ces conditions sont contenues dans les places d'entrée de la transition et sont indiqués par des marques ou jetons pour signaler l'état de chaque ressource à un moment donné, par exemple : machine libre, stock vide, convoyeur en panne, etc.

L'état d'un RdP est donné par un nombre de jetons dans chaque place du réseau, c'est-à-dire le marquage des places. Le RdP est caractérisé par un marquage initial des places et la règle de franchissement des transitions.

Définition I.4.5. (Marquage) Un marquage M_k d'un RdP, $R = \langle P, T, W^{-}, W^{+} \rangle$ est une fonction de l'ensemble des places *P* vers les entiers non négatifs $\mathbb{N}$, $M_k : P \rightarrow \mathbb{N}$, qui définit le nombre de marques $M_k(p_i)$ présentes dans chaque place $p_i \in P$ du RdP.

❑

Le marquage peut aussi etre défini comme un *n*-vecteur de marquage du RdP :

$M_k = [M_k(p_0)\ M_k(p_1)\ \ldots\ M_k(p_n)]^{T}$, où $n = |P|$ et $M_k(p_i) \in \mathbb{N}$, $i=0, \ldots, n$.

Le vecteur M_k donne pour chaque place p_i le nombre de jetons contenus dans la place. Le nombre de jetons dans la place p_i est $M_k(p_i) \in \mathbb{N}$, $i = 0, \ldots, n$.

L'effort de modélisation d'un SED repose toujours sur le concept d'état. Nous identifions le vecteur de marquage $M_k = [M_k(p_0)\ M_k(p_1)\ ...\ M_k(p_n)]^T$ du RdP à l'état courant du SED.

Par exemple, pour le RdP de la figure I.9, nous remarquons qu'il y a un seul jeton, dans la place P_0 — le robot est à l'arrêt. L'état du système est alors : $M_0 = [1\ 0\ 0]^T$.

Par le franchissement d'une transition, les jetons appartenant aux places qui sont connectées par les arcs avec cette transition peuvent être transféré vers d'autres places. Une transition t_j peut être franchie dans le marquage M_k si $M_k \geq W^-(\bullet, t_j)$. Cette relation entre transitions et marquages est dénotée par $M_k[t_j\rangle$. Si t_j peut être franchie dans M_k , t_j est dite franchissable au marquage M_k pour obtenir le marquage M_{k+1} alors

$$M_{k+1} = M_k + W(\bullet, t_j) \tag{I.7}$$

où $W(\bullet, t_j) = W^+(\bullet, t_j) - W^-(\bullet, t_j)$ est la matrice d'incidence du RdP

Ainsi, le comportement dynamique du RdP peut être définit par des séquences de franchissement de telles transitions et la relation de franchissement est étendue aux séquences de transitions $s = t_1 t_2 \ldots t_m \in T^*$ by $M_k[s\rangle$. (*dénote la fermeture Kleene (Dorsaf et *al*., 2012))
Un état M_k du RdP est accessible à partir de l'état initial M_0, si il y a une séquence franchissable $s = t_1 t_2 \ldots t_m$, de vecteur caractéristique de dimension m tel que

$$M_k = M_0 + W(\bullet, t_j)\ \bar{s} \tag{I.8}$$

Chaque composante j de $\bar{s}$ correspond au nombre des franchissements de la transition t_j dans la séquence s. En général, si une séquence de franchissement s est réalisable à partir d'un état M_i , le marquage atteint, M_k, est déterminé par l'équation d'état suivante :

$$M_k = M_i + W(\bullet, t_j)\ \bar{s} \tag{I.9}$$

Attention : $M_k = M_i + W\bar{s}$ n'implique pas que s soit une séquence de franchissement pour M_i

A partir de cette équation, il est possible d'obtenir un ensemble fini ou infini d'état (marquage) à partir de l'état initial, car elle fournit une condition nécessaire (mais pas suffisante) d'accessibilité. L'ensemble d'accessibilité du RdP est l'ensemble de tous les marquages accessibles à partir du marquage initial M_0 par le franchissement des séquences de transition s. Il permet de représenter le comportement du SED par des structures telles que le

graphe des marquages et les langages.

I.4.2. Graphe de marquages

Lorsqu'un RdP est marqué, le marquage correspond à l'état du SED, et son évolution correspond au franchissement des transitions qui peuvent se produire lorsque certaines conditions sont vérifiées. En pratique on étudie le comportement d'un RdP à partir de l'état initial M_0 ou par les langages formels.

Définition I.4.6. (Espace d'état accessible) Étant donné un RdP R et son état initial M_0, un état M_k est accessible s'il existe une séquence de franchissement s tel que $M_0[s\rangle$. L'espace d'états accessibles, est défini comme l'ensemble des marquages accessibles à partir de M_0 par le franchissement de séquences des transitions. Il est noté $\mathcal{A}(R, M_0)$ et nous avons :

$$\mathcal{A}(R, M_0) = \{M_k \in \mathbb{N}^n \mid (\exists\, s \in T^*),\ (M_0[s\rangle M_k)\} \tag{I.10}$$

Exemple I.6.

Dans le cas du RdP du SED de la figure I.8. il est facile de vérifier que $\mathcal{A}(R, M_0) = \{[1\ 0\ 0]^T, [0\ 1\ 0]^T, [0\ 0\ 1]^T\}$.

▲

L'espace d'états accessibles peut être représenté sous forme de graphe d'accessibilité, ayant les marquages (états) accessibles correspondant aux nœuds et les transitions aux arcs. Si l'ensemble d'accessibilité est infini, alors évidemment le graphe d'accessibilité est infini. Dans un tel cas, un algorithme (Karp & Miller, 1969 ; Karp et *al*., 1972) permet de construire un graphe fini, appelé le graphe de couverture, où chaque arc correspond encore à une transition, tandis que chaque nœud correspond soit à un seul marquage accessible, soit il représente un jeu de marques accessibles. Si l'ensemble d'accessibilité est fini (cas des RdP bornés), il est possible d'énumérer de manière systématique l'espace d'accessibilité au moyen du graphe de marquages accessibles (Paola et *al*., 2013 ; Rezig et *al*., 2016). Ici, chaque nœud correspond à un marquage accessible, et chaque arc correspond à une transition.

Définition I.4.7 (Graphe de marquage) Un graphe de marquage est un 4-uplet $\mathcal{G} = \{\mathcal{M}, E, \delta, M_0\}$, où :

- $\mathcal{M}$ est l'ensemble (fini) des marquages ou états,
- E est l'ensemble des événements associés aux transitions,
- $\delta : \mathcal{M} \times E \rightarrow \mathcal{M}$ est la fonction de transition d'état : $\delta(M_i , t_j) = M_k$, et
- $M_0 \in \mathcal{M}$ est l'état initial.

❑

Le graphe de marquage explore toutes les évolutions possibles du SED, en examinant à chaque pas la liste complète des transitions franchissables. Cependant, il peut être de taille très importante et son analyse formelle est complexe.

Exemple I.7

Considérons le RdP du robot qui charge des pièces mécaniques sur un convoyeur. Sa dynamique est la suivante : A partir du marquage initial $M_0 = [1, 0, 0]^T$ la transition T_0 peut être franchie. Si la transition T_0 est franchie, on atteint le marquage $M_1 = [0, 1, 0]^T$. Dans ce cas les transitions franchissables seront T_1 et T_2. Quand la transition T_1 est franchie, nous retournons vers le marquage initial M_0 à nouveau. Quand la transition T_2 est franchie, on atteint le marquage $M_2 = [0, 0, 1]^T$. Dans ce cas seule la transition T_3 est franchissable. Par cette méthode on peut déterminer tous les états possibles à partir d'état initial M_0. La figure ci-dessous donne le graphe de marquages du RdP de la figure I.10

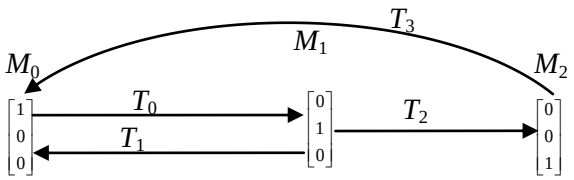


Figure I.10 – Graphe de marquage du RdP de la figure I.9

▲

Comme le graphe de marquages du RdP d'un SED est l'automate de ce denier, alors il est possible de définir les langages d'un RdP de manière analogue.

I.4.3. Langages des réseaux de Petri

Dans la section précédente, nous avons mis en évidence l'importance de caractériser un SED par son espace d'états accessibles. Mais, la puissance de modélisation d'un SED est également liée aux séquences d'événements qu'il peut générer, qui dans le cadre du réseau de Petri correspond aux séquences de transitions franchissables. Comme nous l'avons déjà souligné, une séquence de transitions est un mot, et un ensemble de mots est un langage. Dans cette section, nous nous intéressons aux langages définis par les RdP. En particulier, nous définissons d'abord les langages générés et acceptés, ensuite le RdP synchronisé.

Définition I.4.8. (Langage généré) Le langage généré par un RdP, $R = \langle P, T, W^-, W^+, M_0 \rangle$ est l'ensemble des séquences qui sont franchissables à partir de l'état initial M_0, c'est-à-dire,

$$L(R) = \{s \in T^* \mid M_0[s\rangle\}$$

❑

Le langage généré par R est ainsi un langage préfixe-clos. Ce langage inclut le mot vide ε, car pour tout $M_k \in \mathbb{N}^n$, $M_k[\varepsilon\rangle M_k$

Définition I.4.9. (Langage marqué). Considérons le RdP, $R = \langle P, T, W^-, W^+, M_0 \rangle$, qui reconnait le langage $L(R)$. Soit F un ensemble d'états finaux qui correspondent à la réalisation d'une certaine tâche. Le langage accepté par R, noté $L_m(R)$, est l'ensemble des séquences franchissables à partir de l'état initial M_0 et qui conduit à un état $M_k \in F$, c'est-à-dire,

$L_m(R) = \{s \in T^* \mid \exists\ (M_k \in F);\ M_0[s\rangle M_k\} \Rightarrow L_m(R) \subseteq L(R)$

❑

Jusqu'à présent, nous avons précisé que le comportement dynamique du RdP est tributaire des transitions. Cette hypothèse reliée aux RdP ordinaires peut être étendue aux RdP où les transitions sont étiquetées par des événements, telle que la puissance de la modélisation du SED soit strictement reliée aux séquences d'événements qu'il peut générer. Ainsi, le comportement dynamique du RdP peut être représenté par l'ensemble des séquences franchissables à partir de l'état initial. Pour cela, nous avons besoin de spécifier exactement quel événement correspond à chaque transition. D'où l'intérêt d'utiliser le RdP synchronisé qui offre la possibilité de conditionner le franchissement d'une transition à l'occurrence d'un événement (Pocci *et al.*, 2011) et l'avantage de générer des langages plus expressives (Hopcroft *et al.*, 2007).

Définition I.4.10. (RdP synchronisé) Un RdP synchronisé (RdPS) est un 3-tuplet $N = \langle R, E, f \rangle$, où $R = \langle P, T, W^-, W^+, M_0 \rangle$ est un RdP, E est un ensemble fini d'événements incluant l'événement toujours occurent ε et $f : T \rightarrow E$, une fonction d'étiquetage. Cette fonction associe à chaque transition $t_j \in T$ un événement $e_j \in E$, tel que $e_j := f(t_j)$. f est étendu par homomorphisme de T^* vers E^* de manière canonique : $f(\varepsilon) := \varepsilon$ et $f(\sigma t_j) := f(\sigma)f(t_j)$ pour $\sigma \in T^*$ et $t_j \in T$. Si $f(\varepsilon) \neq \varepsilon$ pour tout $t_j \in T$ la fonction d'étiquettage f est appelée ε - libre.

❑

Dans les RdPS, il y a le poids des arcs, cependant les transitions sont étiquetées par des événements. Une transition validée peut ou ne peut pas être franchie dépendant de l'occurrence de l'événement actuel ou non. Mais, une fois validée, la transition a le potentiel d'être franchie. La définition ci-dessus permet de référencer des transitions différentes par un même événement et de considéré au lieu des séquences de transitions, les séquences d'événements. Dans ce sens, l'ensemble des séquences franchissables est considéré comme un langage généré. En considérant seulement certains états qui correspondent à la réalisation de certaines tâches comme états finaux, nous obtenons le langage marqué ou accepté du RdPS (Komenda et *al*., 2008). Ceci nous conduit à définir les langages du RdPS de manière similaire aux automates, c'est-à-dire à partir des événements (Murata, 1989).

Définition I.4.11. (Langage d'un RdPS) Le langage généré par le RdPS $N = \langle R, E, f \rangle$, définit sur l'alphabet E, est l'ensemble des séquences d'événements qui se produisent à partir de l'état initial M_0, tel que

$L(N) = \{\sigma \in E^* \mid \exists\, s \in T^*, \sigma = f(s) \text{ et } M_0[\sigma\rangle\}$

❑

Définition I.4.12. Considérons le RdP synchronisé $N = \langle R, E, f \rangle$, qui reconnait le langage $L(N)$. Soit F un ensemble de d'états finaux (ou acceptés). Le langage accepté par N, noté $L_m(N)$, est l'ensemble des séquences qui se produises à partir de l'état initial M_0 et qui conduit à un état $M_k \in F$, c'est-à-dire,

$L_m(N) = \{\sigma \in E^* \mid \exists\, (M_k \in F);\ M_0[\sigma\rangle M_k\} \Rightarrow L_m(N) \subseteq L(N)$

❑

Exemple I.8

Considérons à nouveau le RdP de la Figure I.9. Supposons que $f(T_0) = a$, $f(T_1)$ =b, $f(T_2) = c$, $f(T_3) = d$, alors $L(N)$ = [a(b+cd)]*. De plus, si $F = \{[1\ 0\ 0]^T\}$, le langage marqué est $L_m(N)$ = [a(cd+b)]*.

▲

I.4.4. Produit synchrone des RdP

Le produit synchrone ou synchronisation des RdP est une opération plus facile que celui des automates. C'est une opération structurelle, elle consiste à fusionner les transitions synchronisées sur le même événement. Sans perte de généralité, nous supposons que chaque événement est associée à au plus une transition dans les RdPS élémentaires du SED afin d'établir la définition suivante

Définition I.4.13. (Produit synchrone des RdPS) Le produit synchrone de deux RdP synchronisé $N_1 = \{R_1, E_1, f_1\}$, $R_1 = (P_1, T_1, W^-_1, W^+_1, M_{10})$ définit sur E_1 et $N_2 = \{R_2, E_2, f_2\}$, $R_2 = (P_2, T_2, W^-_2, W^+_2, M_{20})$ définit sur E_2 , est un nouveau $N = N_1 || N_2 = \{R, E, f\}$, $R_= (P, T, W^-, W^+, M_0)$, définit sur l'alphabet $E = E_1 \cup E_2$, tel que :

$P := P_1 \cup P_2$.

$T := T_1 \cup T_2 - T_{12}$, $T_{12} := \{ t_i \in T_1 \mid \exists\, t_j \in T_2 / f_1(t_i) = f_2(t_j)\}$

$W^-(\bullet, t_j)$: = $\{W^-_1(\bullet, t_j)$ si $p_i \in P_1$ ou $W^-_2(\bullet, t_j)$ si $p_i \in P_2\}$

$W^+(\bullet, t_j)$: = $\{W^+_1(\bullet, t_j)$ si $p_i \in P_1$ ou $W^+_2(\bullet, t_j)$ si $p_i \in P_2\}$

$f := T \rightarrow E$ *est défini pour tout* $t_j \in T$ *par*

$$f(t_j) = \begin{cases} f_1(t_j) & si\ \boldsymbol{t}_j \in T_1,\ f_1(t_j) \in E_1 \\ f_2(t_j) & si\ \boldsymbol{t}_j \in T_2,\ f_2(t_j) \in E_2 \\ f_1(t_j) = f_2(t_j) & si\ \boldsymbol{t}_j \in T_{12},\ f(t_j) \in E_1 \cap E_2 \end{cases}$$

$M_0(p_i) = \{M_{10}(p_i)$ si $p_i \in P_1$ ou $M_{20}(p_i)$ *si* $p_i \in P_2\}$

❒

Exemple I.9 (Dideban, 2007)

Considérons un système de production composé de deux machines couplées décrit dans la figure I.11.(a). Ici les événements f_1 et t_2 sont partagés entre les RdPS et leur produit synchrone est présenté à la figure I.11.(b). Le RdPS global est obtenu simplement par

synchronisation structurelle des RdPS des machines en fusionnant les transitions ayant les mêmes étiquettes.

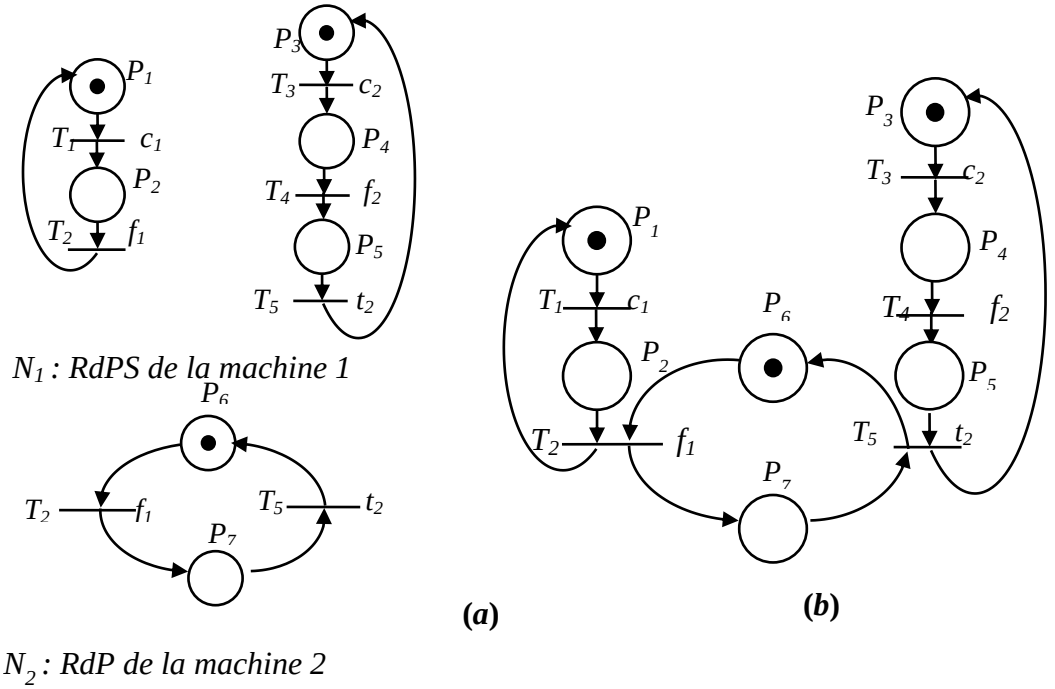


Figure I.11 – Les modèles RdPS des deux machines (a) et leur produit synchrone (b)

▲

Le produit synchrone est une opération formelle qui permet de décrire des caractéristiques intéressantes des systèmes concurrents.

I.4.5. Langages du produit synchrone des RdP

Rappelons que le produit synchrone est généralement défini sur les langages et nécessite les opérations de projection et de projection inverse (Seatzu *et al*., 2012) que nous avons déjà présentées à la section II. Nous notons maintenant $\sigma \in E^*$ comme une séquence générée par $N_1||N_2$, les projections de σ sur les alphabets E_1 et E_2 seront les séquences $\sigma_1 \in E_1^*$ et $\sigma_2 \in E_2^*$.

Soient $\sigma_1 \in L(N_1)$ et $\sigma_2 \in L(N_2)$ les langages générés par N_1 et N_2. Soit $L(N_1||N_2) \subseteq (E_1 \cup E_2)^*$ le langage généré par $N_1||N_2$. Pour toute séquence $\sigma \in L(N_1||N_2)$,

- $P_1(\sigma) := \sigma_1 \in L(N_1)$ et $P_2(\sigma) := \sigma_2 \in L(N_2)$, *si* $E_1 \cap E_2 \neq \emptyset$
- $P_1(\sigma) \cap P_2(\sigma) = \sigma \in L(N_1||N_2)$, *si* $E_1 = E_2$

Puisque $E_1 \cup E_2 = E$, nous pouvons écrire

$$L(N_1||N_2) = \{\sigma \in E^* | P_1(\sigma) := \sigma_1 \in L(N_1), P_2(\sigma) := \sigma_2 \in L(N_2)\} \quad \text{(I.11)}$$

Si les alphabets des deux langages $L(N_1)$ and $L(N_2)$ sont les mêmes, c'est-à-dire $E_1 = E_2 = E$, alors pour chaque $\sigma \in E^*$ nous avons $P_1(\sigma) = P_2(\sigma) := \sigma$ et ainsi

$$L(N_1||N_2) = \{\sigma \in E^* | P_1(\sigma) := \sigma \in L(N_1), P_2(\sigma) := \sigma \in L(N_2)\}$$
$$= L(N_1) \cap L(N_2) \quad \text{(I.12)}$$

I.4.6. Propriétés structurelles d'analyse des Réseaux de Petri

Notre première tâche, en modélisant un SED, est d'utiliser des modèles appropriés, qui décrivent adéquatement son comportement en fournissant un cadre pour effectue son analyse. La modélisation des SED par les RdP offre des propriétés intéressantes pour l'analyse de ce dernier. Dans le cadre de notre travail, les propriétés intéressantes sont de deux ordres :

- *les propriétés comportementales* qui permettent l'analyse du problème d'accessibilité de base, qui consiste à établir si un état donné est accessible à partir de l'état initial ou si cet état n'est pas bloquant.
- *les propriétés structurelles* qui permettent la démonstration de plusieurs propriétés indépendamment de l'état initial.

Dans les sections précédentes, les propriétés comportementales ont été partiellement décrites, mais les définitions formelles de ces propriétés telles que l'accessibilité, la bornitude, la vivacité, le blocage etc., sont bien développées dans (Cassandras & Lafortune, 1999)

a) Les propriétés comportementales

Problème d'accessibilité : Nous pouvons définir l'ensemble des états accessibles d'un RdP synchronisé par :

$$\mathcal{A}(N, M_0) = \{M_k \in \mathbb{N}^m \mid (\exists\, \sigma \in E^*), (M_0[\sigma\rangle M_k)\} \quad \text{(I.13)}$$

Dans ce cadre un problème d'accessibilité consiste soit à se demander si un état particulier fait partie de $\mathcal{A}(N, M_0)$ (c'est-à-dire s'il est accessible).

Définition I.4.14 (Problème d'accessibilité). Soit un RdP synchronisé, $N = \langle R, E, f \rangle$ et un état initial M_0. Le problème d'accessibilité pour l'état M_k consiste à se demander si ($\exists\, \sigma \in E^*$), ($M_0\, [\sigma\rangle M_k$).

❒

Il n'existe pas de condition nécessaire et suffisante donnant une réponse générale au problème d'accessibilité; il existe par contre une condition nécessaire.

Théoreme I.4.1. Si M_k est un état accessible pour le RdP synchronisé, N de matrice d'incidence W, alors $M_k = M_i + W\bar{s}$, de variable $\bar{s}$ admet une solution dans $\boldsymbol{N}^{m}$ (m étant le nombre de transition du RdP). (Si $\bar{s}$ existe, c'est l'image commutative d'une séquence de franchissement permettant de passer de M_0 à M_k).

❒

Bornitude

Définition I.4.15. (Bornitude) Une place $P_i \in P$ d'un RdP synchronisé N est dite b-bornée avec $b \geq 1$ si et seulement si : $\forall M_k \in$ A (N; M_0), $M_k(P_i) \leq b$

- Un RdP synchronisé N est appelé borné si toutes ses places sont bornées.
- Un RdP synchronisé est appelé binaire (ou sauf) pour un état initial M_0 donné si toutes ses places sont 1-bornées (contiennent au plus un jeton).

❒

Il existe encore un autre moyen qui permet de démontrer que des places d'un RdP synchronisé sont bornées, il fait intervenir notion d'invariants du RdP considéré.

Vivacité et blocage

Définition I.4.16 (Vivacité) Une transition t_j d'un RdP synchronisé N est vivante si elle peut toujours être franchie à partir d'un état accessible quelconque

$\forall M_k \in \mathcal{A}(N; M_0), \exists\, M_{k+1} \in \mathcal{A}(N; M_0)$ et $M_{k+1}\, [t_j\rangle M_k$

Généralement, nous dirons qu'un RdP est vivant si toutes ses transitions sont vivantes.

❒

Définition I.4.17. (Blocage) Un état accessible M_k d'un RdP synchronisé N est une situation de blocage si pour toute transition $t_j \in T$, t_j n'est pas franchissable pour M_k. Un RdP est sans

blocage si aucun de ses marquages atteignables n'est une situation de blocage. Un blocage est un état ou marquage dans lequel aucune transition n'est franchissable.

❒

b) Les invariants structurels

Les invariants font partie des propriétés structurelles des RdP, c'est-à-dire des propriétés indépendantes de tout état ou marquage initial mais, qui dépendent uniquement de la structure du RdP. Les invariants sont des moyens importants d'analyse des RdP puisqu'ils permettent d'étudier la structure du RdP indépendamment de tout comportement (Lautenbach, 1987). Les invariants de marquage correspondent à des ensembles de places dont le nombre pondéré de jetons reste constant quel que soit le marquage ou l'état accessible. Ils sont représentés par des vecteurs X de dimension n, où n est le nombre de places du RdP; calculés en trouvant les solutions de l'équation suivante :

$$X^T.W = 0 \qquad \text{(I.14)}$$

Où W est la matrice d'incidence n x m du RdP

Le vecteur X solution de $X^T.\ W = 0$ est appelé P- invariant ou P- semi – flot. L'invariant de marquage déduit de tout P-semi-flot impose la loi d'invariant à l'espace d'états accessibles du RdP : en fait, si un état M_k est accessible à partir de l'état initial M_0 , nous avons :

$$X^T.M_k = X^T.M_0 = \text{Constante} \qquad \text{(I.15)}$$

Où : M_0 est l'état initial du RdP, et M_k représente tout état accessible

Cette relation, nécessaire mais pas suffisante pour l'accessibilité, signifie que la somme pondérée des jetons dans les places de l'invariant reste constante pour tous les états accessibles, et cette somme est déterminée par l'état initial du RdP. Cette propriété d'invariants est principalement utilisé dans la synthèse de contrôleur des SED et il y a des algorithmes efficaces pour trouver tous les invariants d'un RdP (David & Alla. 2010).

Par exemple considérons le RdP du système de production composé de deux machines couplées présenté dans la figure I.11. Dans ce modèle, nous avons 3 invariants minimaux de marquage pour l'ensemble des places $P = \{P_1, P_2, P_3, P_3, P_4, P_5, P_6, P_7\}$:

$$M_k(P_1) + M_k(P_2) = 1 \ ; \ M_k(P_3) + M_k(P_4) + M_k(P_5) = 1 \text{ et } M_k(P_6) + M_k(P_7) = 1,$$

En faisant la somme de ces 3 invariants, on peut construire un invariant qui contient toutes les places du RdP global, à savoir :

$$M_k(P_1) + M_k(P_2) + M_k(P_3) + M_k(P_4) + M_k(P_5) + M_k(P_6) + M_k(P_7) = 3$$

Dans le cas général, ce type de RdP est dit conservatif, car l'invariant de marquage qui contient toutes les places du RdP.

I.5. Comparaison RdP et automates à états finis

Concernant les formalismes de modélisation des SED, les automates et les RdP ont une grande similitude car ils représentent tous deux la structure de transition du SED. Par rapport aux automates, les RdP ont plusieurs avantages (Zhou & DiCesare 1993, David & Alla 2010) puisqu'ils tiennent explicitement compte du parallélisme, de la synchronisation et du conflit. Ils génèrent un langage plus expressif incluant des langages réguliers (Lafortune & Yoo, 1990 ; Giua & Seatzu, 2007). La structure des RdP est de taille plus petite que celle d'un automate modélisant le même système. Les états sont représentés par les marquages donnant ainsi une description compacte du SED, sans énumération de l'espace d'état. De plus, le produit synchrone (composition parallèle) qui permet de modéliser le comportement global du SED provoque l'explosion combinatoire du nombre d'états. C'est-à-dire, la croissance exponentielle de l'espace d'états ($Q = Q_1 \times Q_2$). Dans le cas des RdP, le produit synchrone est caractérisée par une croissance linéaire ou combinatoire du nombre de places ($P = P_1 \cup P_2$). Il nécessite donc un effort de calcul réduit par rapport à celui demandé aux automates car, le RdP composé est obtenu en fusionnant simplement les transitions avec la même étiquette. La croissance du nombre de transitions du RdP composé est, elle aussi, plus faible que celle l'automate composé. (Yin & Lafortune, 2017). Par conséquent, cette propriété fait du RdP un excellent moyen pour la modélisation de manière intuitive et compacte des systèmes complexes. Par un passage systématique du RdP vers l'automate, il est possible de profiter totalement des propriétés des deux modèles. En effet, l'analyse du langage dans la section précédente montre que les RdP ont une grande similitude avec les automates (Murata, 1989).
Rappelons que dans les RdP l'état est un vecteur d'entiers non-négatif. Cela permet également de décrire des spécifications logiques sous une forme numérique. Par exemple, si nous voulons imposer au système de production composé de deux machines (figure I.11) la

spécification suivante : la première machine ne devrait jamais fonctionner si la seconde machine est inactive. Nous devons imposer la contrainte $M_k(P_2) + M_k(P_3) \leq 1$.

La possibilité offerte par les RdP de décrire l'espace d'état d'un SED, avec un ensemble de vecteurs entiers et les langages qu'ils génèrent, ont une implication forte dans l'analyse structurelle et la commande par supervision des SED. Il est judicieux d'utiliser des formalismes de modélisation qui nous permettront de représenter les langages d'une manière qui met en évidence les propriétés structurelles sur le comportement du SED et qui est facile à manipuler. Dans ce domaine de la recherche, plusieurs approches formelles bien fondées ont été développées (Moody & Antsaklis, 1998 ; Iordache et *al*., 2001; Paola et *al*., 2013)

I.6. Conclusion

Dans ce premier chapitre nous avons présenté un aperçu des notions et des propriétés essentielles des SED que nous allons utiliser dans la suite de notre travail. Les systèmes industriels de production considérés comme SED peuvent être modélisés par un langage, et si ce dernier est régulier, il est possible de construire un automate et/ou un RdP dont le langage décrit précisément le comportement du SED. Du point de vue de la modélisation, ces deux outils de base ont des implications relatives au phénomène d'explosion combinatoire du nombre d'états. Ceci est inhérent au produit synchrone de plusieurs sous-modèles d'un système complexe. Les RdP, avec leurs structures de modélisation très riches et leurs modèles concis, sont les plus adaptés à la description des SED. De plus, lorsque cela est nécessaire ils permettent un passage systématique vers le modèle automate du SED, par construction du l'espace d'états accessibles. Cela permet de profiter totalement des propriétés des deux modèles. Toutes ces notions représentent la base qui nous permettra, d'un côté, de présenter la théorie de commande par supervision des SED et de l'autre côté de poursuivre notre travail sur la synthèse structurelle des contrôleurs discrets basée sur les RdP.

Chapitre II

Synthèse de commande par supervision basée sur les automates et langages

Le chapitre se concentre uniquement sur les détails relatifs à la théorie de commande par supervision des SED initiée proposée par Ramadge et Wonham (1983) en rappelant tout d'abord les principes de base de cette théorie. Nous mettrons l'accent sur le concept clé de la contrôlabilité et deux types de spécifications, qui caractérise la synthèse des contrôleurs. La présentation de l'algorithme de Kumar nous permettra de définir les états interdits dans un SED.

Chapitre II

Synthèse de commande par supervision basée sur les automates et les langages

II.1. Théorie de commande par supervision

II.1.1. Principe de base

Le besoin de méthodes formelles et d'outils performants capables de traiter des problèmes de supervision des SED a donné naissance à la théorie de la commande par supervision (Ramadge & Wonham, 1986, ; Ramadge & Wonham, 1987 ; Radmage & Wonham, 1989). La théorie de commande par supervision initiée par Ramadge et Wonham repose sur l'utilisation d'outils tels que les automates et les langages pour la synthèse de contrôleurs quelle que soit la complexité du SED (Wonham, 2011). Dans cette approche, le procédé *P* est considéré comme un SED, générant spontanément des événements, et évoluant en boucle fermée avec le contrôleur *C*. Celui-ci a la tâche d'orienter l'évolution du SED dans l'objectif de restreindre son comportement au fonctionnement désiré. Le schéma de supervision est présenté dans la figure II.1.

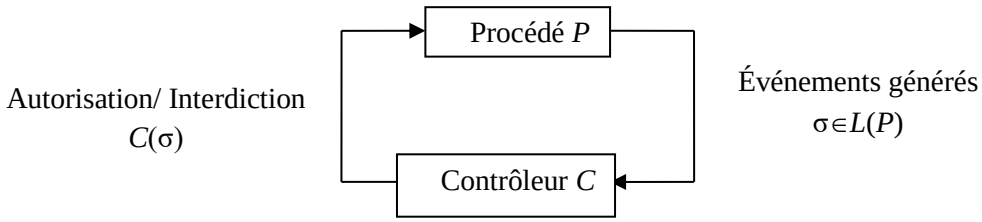


Figure II.1 – Principe de commande par supervision

Dans ce schéma, un procédé est couplé à un contrôleur ou superviseur. Les entrées du contrôleur sont les sorties du procédé, et vice versa. Le contrôleur peut modifier le fonctionnement du procédé en autorisant ou en interdisant l'occurrence d'événements. En pratique, certains événements générés par un procédé ne peuvent pas être interdits. En effet, si

l'on considère l'exemple de la presse typographique de la figure II.2, il paraît naturel que la défaillance d'impression de la presse ne puisse pas être interdite. Un tel événement sera appelé *événement incontrôlable*. Au contraire, on appellera *événement contrôlable*, tout événement qui peut être interdit à n'importe quel moment.

Définition II.1. (Evénements) De manière générale, L'alphabet des évènements E d'un procédé est partitionné comme suit :

$$E = E_c \cup E_{uc} \text{ (avec } E_c \cap E_{uc} = \emptyset)$$

Où E_c est l'ensemble des évènements contrôlables et E_{uc} est l'ensemble des évènements incontrôlables.

❑

L'idée est que lorsque le procédé est prêt à exécuter un évènement contrôlable, le contrôleur peut désactiver l'évènement, notamment en empêcher l'occurrence. En revanche, le contrôleur n'a aucun moyen d'empêcher l'occurrence d'un évènement incontrôlable.

Exemple II.1 (Guia, 2017)

Considérons l'automate de la figure II.2, où nous avons pris la convention de représenter par un arc barré, toute transition associée à un événement contrôlable. Dans ce cas, l'alphabet $E = \{a, b, c, d, e, f\}$ est partitionné en $E_c = \{a, b, d, f\}$ et $E_{uc} = \{c, e\}$.

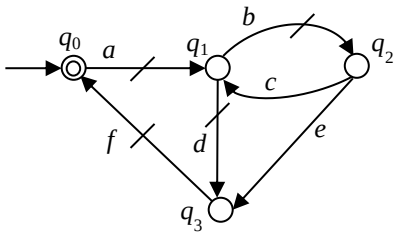


Figure II.2 – Un procédé de presse typographique modélisé par un automate

L'automate de la figure II.2 représente une presse typographique qui est initialement à repos (état initial q_0). Une fois que la forme imprimante a été encré (évènement a) la presse est prête à imprimer (état q_1). De cet état, l'opérateur peut commencer (évènement b) une opération d'impression (état q_2) ou peut mettre (évènement d) la presse hors ligne (état q_3) pour nettoyer ou changer la forme imprimante. Lorsque la presse a démarré l'impression (état q_2) l'opération peut terminer avec succès (évènement c) et la presse revient à l'état q_1, ou une erreur d'impression peut se produire (évènement e) et dans ce cas, la presse va de façon

autonome hors ligne. L'exécution d'une maintenance (évènement *f*) ramène la machine à l'état de repos.

▲

Dans cet automate l'état final coïncide avec l'état initial, pour indiquer que, à la fin d'une ou de plusieurs impressions, la machine devrait revenir à l'état de repos. La notion d'évènements contrôlables et incontrôlables a une claire interprétation. A titre d'exemple, l'évènement *a,* désigne le fait que la presse est encrée : cette opération est effectuée par l'opérateur de la presse et il est raisonnable de supposer que cet évènement peut être désactivé par un contrôleur si nécessaire. Au contraire, *e* représente une défaillance d'impression : évidemment, cela est un évènement indésirable, mais le contrôleur n'a aucun moyen pour l'empêcher. De la même manière, on suppose que l'évènement *c,* est lui aussi incontrôlable, car une fois que la presse a commencé à imprimer, aucune action ne peut l'empêcher de terminer l'opération. L'objectif de la commande par supervision est de concevoir un contrôleur *C* afin de le coupler en boucle fermée avec le procédé *P* pour assurer le fonctionnement désiré.

II.1.2. Concept de la commande par supervision

Dans la théorie de la commande par supervision, un procédé *P* est un système à contrôler. Son comportement est décrit par les séquences d'évènements (à savoir, le langage) qu'il peut générer sur un alphabet *E*. En particulier, deux langages peuvent être associés à un procédé :

- Le langage $L \subseteq E^*$clos par préfixe, composé par l'ensemble des séquences d'évènements que le système peut générer ;
- Le langage marqué $L_m \subseteq L$, composée par les séquences d'évènements générés par le système qui correspond à la réalisation de certaines tâches.

Il est naturel d'utiliser un automate à états finis pour modéliser un procédé. Le comportement de ce dernier peut s'avérer ne pas être entièrement satisfaisant dans le sens où il ne respecte pas certaines s*pécifications,* données par le cahier des charges. Il est donc nécessaire de restreindre par le biais d'un contrôleur, ce comportement. Etant donné un procédé et un ensemble de spécifications logiques, l'objectif de la théorie de commande par supervision est de synthétiser un contrôleur de sorte que le comportement du procédé couplé au contrôleur respecte les spécifications.

Considérons l'automate $P = (Q, E, \delta, q_0, Q_m)$ tel que $L(P) = L$ et $L_m(P) = L_m$; le rôle de la supervision consiste à interdire le franchissement de certaines transitions et à autoriser le franchissement d'autres transitions, par l'intermédiaire de l'action du contrôleur (Kattan, 2004). Ainsi, le procédé couplé au contrôleur peut être perçu comme un système qui reçoit en entrée une liste d'événement interdits Φ et en sortie donne l'ensemble des événements possibles dans chaque état moins l'ensemble des événements interdits dans cet état.

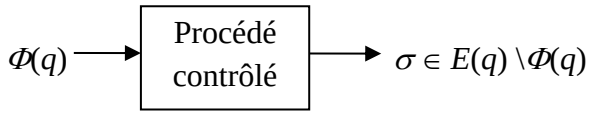


Figure II.3 – Procédé contrôlé

Depuis un état q, le procédé contrôlé C/P évolue de façon spontanée en produisant un événement $\sigma \in E(q) \setminus \Phi$, où $\sigma \in E(q) \setminus \Phi$ dénote l'ensemble des événements qui appartiennent à l'ensemble $E(q)$ et qui n'appartiennent pas à l'ensemble Φ. Conformément à la figure II.1, un événement peut être généré par le procédé contrôlé s'il peut être généré par le procédé P en isolation et s'il est autorisé (non interdit) par le contrôleur C. Par extension, une séquence d'événements σ est possible dans le fonctionnement en boucle fermée, si elle est possible dans le procédé en isolation ($\sigma \in L(P)$), et si elle est autorisée par le contrôleur ($\sigma \in L(C)$).

II.1.3 Définition d'un contrôleur

Soulignons que la fonction de transition de l'automate P peut être commandée par C dans le sens où les événements contrôlables de P peuvent être autorisés ou interdits par C.

Définition II.1.2. (Contrôleur) Un contrôleur C contrôlant un procédé P peut être représenté par une fonction $C : L(P) \to \Gamma$ avec $\Gamma = \{\gamma \in 2^E \mid E_{uc} \subseteq \gamma\}$, $\Gamma \subseteq 2^E$ représente l'ensemble des lois de commande, qui génère une séquence d'événements admissibles

$$\xi_0 = C(\varepsilon), \xi_1 = C(e_1), \xi_2 = C(e_1 e_2), \cdots$$

en réponse à la séquence d'évènements $\sigma = e_1 e_2 \cdots \in L(P)$ générée par le procédé.

❑

Il est à noter que dans la définition, on admet que, pour toute séquence σ générée par le procédé P, l'entrée de commande $C(\sigma)$ déclenchée par le contrôleur est admissible après σ.

Ainsi, pour toute séquence $\sigma \in L(P)$, $C(\sigma) \cap \Gamma(\delta(q_0, \sigma))$ est l'ensemble des événements autorisés par le contrôleur *C* et que *P* peut exécuter à partir de son état courant $\delta(q_0, \sigma)$.

Un contrôleur peut être perçu comme une machine à états déterministe qui évolue conformément à l'occurrence d'un événement de *E* généré par le procédé *P*. On peut remarquer qu'entre deux occurrences successives d'événements la sortie du contrôleur reste inchangée. Ainsi, on peut représenter un contrôleur par un modèle tel que la sortie ne dépend que de l'état, c'est-à-dire par une machine de Moore (Dideban, 2007). Nous pouvons enfin donner une définition formelle d'un contrôleur.

Définition II.1.3. (Contrôleur) Le contrôleur *C* peut être défini par le 6-uplet $C = (V, E, \xi, v_0, 2^{Ec}, \theta)$ où *V* est un ensemble fini d'états; *E* est l'alphabet d'entrée; $\xi : V \times E \rightarrow V$ est la fonction de transition d'états; v_0 est l'état initial; 2^{Ec} est l'alphabet de sortie; et $\theta : V \rightarrow 2^{Ec}$ est la fonction d'affectation de sortie.

❑

Pour chaque état *v*, le contrôleur *C* fournit en sortie une liste d'événements interdits $\Phi = \theta(v)$. Ainsi, chaque sortie de *C* est un élément de 2^{Ec}, où 2^{Ec} est l'ensemble de tous les sous-ensembles de E_c. On appelle langage de supervision et on note $L(C)$ le langage associé au contrôleur *C*. Le langage $L(C)$ est l'ensemble des séquences d'événements autorisés par *C*. Formellement, $L(C)$ peut être défini de la façon suivante :

Définition II.1.4. (Langage d'un contrôleur) Le langage $L(C)$ associé à l'automate $C = (V, E, \xi, v_0, 2^{Ec}, \theta)$ est défini de façon récursive par le plus petit langage sur E^* satisfaisant :

- $\varepsilon \in L(C)$
- si $\mu \in L(C)$ alors $\mu\sigma \in L(C)$ pour $\sigma \in E$ si $\sigma \notin \theta(\xi(v_0, \mu))$.

❑

Le procédé contrôlé *C*/*P* est un SED évoluant en boucle fermée constitué d'un procédé *P* couplé au contrôleur *C* dont on peut déterminer son langage généré ainsi que son langage marqué. Ces deux langages sont des sous ensemble de $L(P)$ et $L_m(P)$ contenant les séquences possible en présence de *C*. Le langage $L(C/P)$ représente alors le fonctionnement en boucle fermée du SED. Le langage $L(C/P)$ est simplement défini par :

$$L(C/P) = L(P) \cap L(C) \qquad \text{(II.1)}$$

Définition II.1.5. (Langage procédé contrôlé) Le langage généré par le procédé contrôlé en boucle fermée, *C*/*P*, est défini comme suit :

- $\varepsilon \in L(C/P)$
- $[(\sigma \in L(C/P)) \text{ et } (\sigma\mu \in L(P)) \text{ et } (\mu \in C(\sigma))] \Leftrightarrow [\sigma\mu \in L(C/P)]$.

Le langage marqué par *C*/*P* est défini comme suit :

$L_m(C/P) := L(C/P) \cap L_m(P)$

❑

Le langage marqué revêt une importance dans la mesure où il caractérise les séquences qui permettent d'accomplir des taches tout en respectant les contraintes de supervision. On dira que le contrôleur est non-bloquant pour *P* si

$$L_m(C/P) \subseteq \overline{L}_m(C/P) \subseteq L(C/P) \subseteq L(P) \qquad \text{(II.2)}$$

Le modèle automate reconnaissant *L*(*C*/*P*) est obtenu en effectuant le produit synchrone de *P* et de *C*. nous appellerons *fonctionnement en boucle fermée*, le fonctionnement du procédé couplé à son contrôleur ou superviseur. La théorie de commande par supervision des SED fournit de nombreux concepts et résultats théoriques. Néanmoins, seuls les concepts fondamentaux nécessaires dans la suite de notre étude seront présentés. Nous nous focaliserons sur le concept de contrôlabilité (Cassandras & Lafortune, 2008) qui est nécessaire, ainsi que sur la synthèse des contrôleurs par automates (Kumar, 1991).

II.2. Condition de contrôlabilité et existence d'un contrôleur

II.2.1. Condition de contrôlabilité

Dans un SED réels, tous les événements ne sont pas contrôlables. Il existe des événements incontrôlables dont le contrôleur ne peut interdire. De fait, tous les contrôleurs ne sont pas admissibles. Nous souhaitons que le fonctionnement du procédé couplé au contrôleur respecte les spécifications et soit le plus permissif possible. Etant donné un procédé *P* et une spécification de fonctionnement *S*, on souhaite synthétiser un contrôleur *C* de façon à ce que le système en boucle fermée *C*/*P*, respecte la spécification, ce langage est noté *K*. C'est-à-dire, qu'on doit chercher le langage $L(P) \cap L(S)$ qui correspond à l'ensemble des séquences qui peuvent être générées par le procédé et qui sont tolérées par la spécification. La

synthèse d'un tel contrôleur tel que $L(C/P) = K$ est basée sur la condition de contrôlabilité introduite par Ramadge et Wonham pour caractériser les langages (Ramadge et *al*., 1987)

Théorème II.2.1. (Condition de contrôlabilité) Considérons un SED $P = (Q, E, \delta, q_0, Q_m)$ où $E_{uc} \subseteq E$ est l'ensemble des événements. Soit $K \subseteq L(P)$, où $K \neq \emptyset$. Alors il existe un contrôleur C tel que $L(C/P) = \bar{K}$ si et seulement si

$\bar{K} E_{\text{uc}} \cap L(P) \subseteq \bar{K}$

Cette condition sur K est appelé condition de contrôlabilité

❑

La contrôlabilité est un concept clé dans la théorie de commande par supervision ayant des implications sur l'existence du contrôleur (Ramadge & Wonham, 1989). Nous énonçons une définition générale de ce concept.

Définition II.2.1. (Contrôlabilité) Soient K et $L(P) = \bar{L}(P)$ les langages sur l'ensemble d'événements E. Soit E_{uc} un sous ensemble designé de E. K est contrôlable par rapport à $L(P)$ et E_{uc} si

. $\bar{K} E_{\text{uc}} \cap L(P) \subseteq \bar{K}$

Où $\bar{K}$ représente le préfixe-clôture du langage de spécification et $L(P)$ le langage du procédé

❑

Par cette définition, la contrôlabilité est une propriété de préfixe-clôture des langages. Ainsi, K est contrôlable si et seulement si $\bar{K}$ est contrôlable. Autrement dit, si K est contrôlable, alors pour tout $\sigma \in \bar{K}$, pour tout $\mu \in E_{uc}$, $\sigma\mu \in L(P) \Rightarrow \sigma\mu \in \bar{K}$

II.2.2. Condition d'existence d'un contrôleur

Soient $L(P)$ le langage du procédé et K un comportement admissible inclus dans $L(P)$. La condition d'existence d'un contrôleur qui assure le comportement admissible du SED en boucle fermée est intimement lié au concept de contrôlabilité des langages, c'est-à-dire la possibilité d'imposer que le langage généré par le procédé couplé au contrôleur, $L(C/P)$, appartienne à un certain langage de spécification Si le comportement admissible est un langage contrôlable, un contrôleur existe. De même, il est connu qu'un contrôleur existe lorsque, le langage du procédé et le langage de spécification sont tous les deux réguliers (Wonham, 2003). Par contre, si le comportement admissible n'est pas contrôlable, nous

devons restreindre davantage le comportement du système au sous-langage contrôlable pour lequel un contrôleur existe (Wonham, 2005). Ainsi, pour un langage K préfixe-clos, inclus dans le langage $L(P)$ du procédé, il existe un contrôleur C tel que $L(C/P) = K$ si et seulement si K est contrôlable par rapport à $L(P)$ (Radmadge & Wonham, 1989). Si le langage K n'est pas contrôlable par rapport au langage du procédé, il n'existe pas de contrôleur C tel que $L(C/P) = K$. Dans ce cas, il est nécessaire de trouver une solution de contrôle plus restrictive et la synthèse d'un contrôleur requiert alors la détermination du plus grand sous ensemble de K qui soit contrôlable, c'est-à-dire le langage suprême contrôlable, noté $Sup\mathrm{C}(K)$. Évidemment le langage $L(C/P) = Sup\mathrm{C}(\mathrm{K})$ est inclus dans K et décrit le contrôleur le plus permissif possible qui assure le respect de la spécification pour le SED en boucle fermée. Puisque, $Sup\mathrm{C}(K)$ correspond au plus grand sous-langage contrôlable de K relativement à E_{uc} et $L(P)$ (Brandin & Wonham, 1994), il existe un contrôleur C tel que

$$L(C/P) = Sup\mathrm{C}(K) \qquad \text{(II. 3)}$$

Dans ce sens, $L(C/P)$ correspond au plus grand comportement de P qui vérifie la spécification K tout en respectant la condition de contrôlabilité. Un tel contrôleur sera alors qualifié de maximal permissif. Il existe dans la littérature plusieurs méthodes pour le calcul d'un langage suprême contrôlable :

- L'algorithme proposé par Radmadge et Wonham (1989) est basé sur un calcul itératif.
- L'algorithme proposé par Kumar (1991), non itératif, est basé sur la détermination d'états défendus dans le modèle automate du fonctionnement désiré, par une simple exploration des espaces d'états du SED.

Remarque II.1. Si le fonctionnement désiré K est préfixe-clos, alors son langage suprême contrôlable $Sup\mathrm{C}(K)$ est aussi préfixe-clos**.** Dans le cas où les langages du procédé et de spécification sont réguliers, le plus grand langage contrôlable est régulier (Ushio, 1990)

❑

Dans (Ezpeleta *et al*., 1995) un nouveau problème de commande par supervision est étudié: le Blocage. Ici, un concept est utilisé pour déterminer un contrôleur qui peut également permettre le blocage afin d'atteindre un plus grand comportement. La contrôlabilité et le non blocage des SED nécessitent des conditions supplémentaires (Giua & DiCesare, 1994) qui ne seront pas abordées dans ce travail.

II.3. Synthèse du contrôleur

II.3.1. Les spécifications de contrôle

Nous savons que la théorie de commande par supervision consiste à restreindre le comportement d'un procédé P par le biais d'un contrôleur C de manière à ce que le procédé ainsi contrôlé respecte un ensemble de spécifications (objectifs de contrôle) K que le procédé initial ne vérifiait pas : $L(C) \cap L(P) \subseteq K$. Initialement, le procédé, la spécification (objectif) ainsi que le contrôleur étaient modélisés par des langages (réguliers). À partir de ces langages (en particulier K et $L(P)$, le contrôleur C doit être calculé de manière à assurer des comportements du procédé qui appartient aussi aux spécifications. Dans le cas général, pour déterminer un contrôleur permettant le respect d'un cahier des charges donné, il faut d'abord définir un modèle de spécification qui traduit fidèlement les contraintes imposées. Une spécification décrit le comportement souhaité d'un procédé contrôlé. Nous considérons ici deux types de spécifications.

Définition II.3.1. (Spécification d'état) Étant donné un procédé dont l'espace d'état est Q, une spécification d'état consiste en un sous-ensemble $Q_A \subseteq Q$ qu'on appelle ensemble des états admissibles.

❑

Une telle spécification est représentée sur la figure II.4. Les états se trouvant dans l'ensemble $Q_I = Q \backslash Q_A$ sont appelés états interdits.

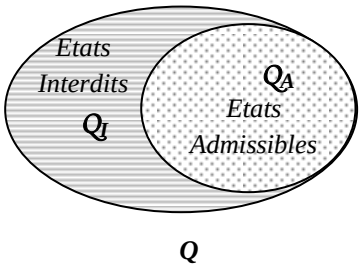


Figure II.4 – Une spécification d'états pour un procédé avec un espace d'états Q.

Définition II.3.2. (Spécification de langage) Étant donné un procédé dont le langage clos est $L(P) \subseteq E^*$, une spécification du langage consiste en un langage $K \subseteq E^*$ qu'on appelle ensemble des séquences admissibles.

❑

Une telle spécification est représentée sur la figure II.5. Les séquences du langage $L(P) \cap K$, qui sont générés par le procédé et sont également admissibles, sont appelées séquences désirés. Les séquences du langage $L(P) \setminus K$, qui sont générées par le procédé, mais ne sont pas admissibles, sont appelés séquences interdites.

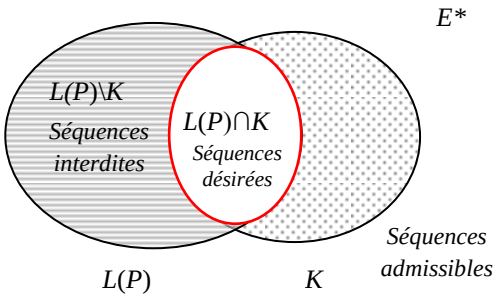


Figure II.5 – Une spécification de langage K pour un procédé avec langage clos $L(P) \subseteq E^*$

Dans notre travail, nous allons discuter de la manière dont il est possible de synthétiser un contrôleur capable de faire respecter ces deux types de spécifications.

II.3.2. Synthèse d'un contrôleur pour les spécifications d'états

Définissons d'abord le problème de commande par supervision pour une spécification d'états

Définition II.3.3. (Problème de commande par supervision pour une spécification d'états) Considérons un procédé P avec l'ensemble des états Q et soit une spécification qui consiste en un ensemble d'états admissibles $Q_A \subseteq Q$. Trouver un contrôleur maximalement permissif C tel que le procédé contrôlé n'atteindra jamais un état interdit dans $Q_I = Q \setminus Q_A$.

❑

Remarque II.2. Un autre type d'états interdits correspond aux états de blocage. Pour notre approche cette différence n'est pas importante.

❑

Pour éviter que les états interdits dans $Q_I = Q \setminus Q_A$ soient atteints, il est nécessaire d'interdire l'occurrence de tous les évènements menant d'un état admissible $q_i \in Q_A$ à un état interdit $q_k \in Q_I$. Cependant, il peut arriver qu'à partir d'un état admissible, il existe une séquence d'événements incontrôlables (à savoir, une séquence de transitions incontrôlables) qui donne un état interdit et une telle séquence ne peut pas être interdite par un contrôleur.

Définition II.3.4. (États faiblement interdits) Étant donné un procédé $P = (Q, E, \delta, q_0, Q_m)$ et un ensemble d'états admissibles Q_A, nous définissons l'ensemble des états faiblement interdits

$$Q_F = \{q_i \in Q_A \mid \exists \sigma \in E_{uc} \; ; \; \delta(q_i, \sigma) = q_k \in Q_I\}$$

contenant tous les états du procédé à partir desquels un état interdit est accessible par une séquence qui ne contient que des évènements incontrôlables.

❑

L'algorithme de Kumar (Kumar, 1991), permet de résoudre ce problème de commande par supervision pour une spécification d'états car, il est simplement basé sur la détermination des états interdits et les états faiblement interdits du modèle automate du fonctionnement désiré. Ainsi, en présence d'évènements incontrôlables, le contrôleur doit veiller à ce que le procédé ne parvienne pas à un état interdit ou faiblement interdit. Le contrôleur est dit maximalement permissif s'il empêche seulement d'atteindre des états interdits ou faiblement interdits. Ces quelques notions importantes nous permettent de présenter directement l'algorithme de Kumar

❖ **Algorithme de Kumar**

Soit $P = (Q, E, \delta, q_0)$ le modèle accepteur du procédé, et $S = (V, E, \xi, v_0)$ celui de la spécification que nous souhaitons lui imposer. L'algorithme de Kumar permet de vérifier la contrôlabilité du langage de spécification $L(S)$. Si le langage de spécification $L(S)$ est contrôlable, l'automate P/S décrivant le procédé contrôlé est le produit synchrone des automates P et S. Dans le cas où le langage $L(S)$ n'est pas contrôlable cet algorithme permet de synthétiser un modèle automate du langage suprême contrôlable du fonctionnement désiré, c'est-à-dire, $Sup\mathrm{C}(K)$. Ceci est illustré à la figure II.6.

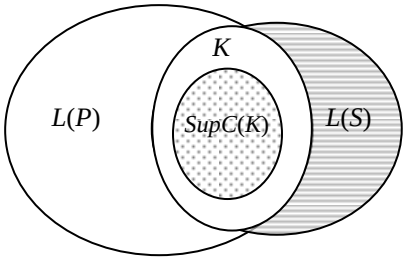


Figure II.6 – Langage de spécification, fonctionnement désiré et langage suprême contrôlable d'un fonctionnement désiré

Intuitivement, la détermination de *Sup*C(*K*), revient à interdire par la supervision les événements contrôlables qui précèdent, de "plus près", les états sources d'incontrôlabilité. La supervision permet ainsi d'assurer le fonctionnement spécifié de façon optimale. Nous pouvons maintenant présenter cet algorithme qui calcule l'automate *D*'qui accepte le langage suprême contrôlable du fonctionnement désiré pour le SED en boucle fermée *C*/*P* = *P*||*S*.

L'algorithme de Kumar est basé sur les 4 étapes suivantes :

Etape 1. On construit le produit synchrone *D* de *P* et de *S*, c'est-à-dire, *D* = *P*||*S*.

Le langage *L*(*D*) sera noté *K*.

Etape 2. On détermine l'ensemble des états interdits, Q_I

Définition II.3.5. (État interdit) On appelle état interdit, tout état (q_i, v_j) de *D* tel qu'il existe un événement incontrôlable $\sigma \in E_{uc}$ où $\delta(q_i, e)$ est définie dans *P* et $\xi(v_j, e)$ n'est pas définie dans *S*. Dans ce contexte, l'ensemble des états interdits Q_I sera défini comme ci-dessous :

$$Q_I = \{(q_i , v_j) \mid \exists\ e \in E_{uc} \text{ avec } \delta(q_i, e) \text{ défini, et } \xi(v_j, e) \text{ non défini } \}$$

❑

Etape 3. On détermine l'ensemble des états faiblement défendus, Q_F

Définition II.3.6. (État faiblement interdit) On appelle état faiblement défendu, tout état (q_i, v_j) de *D* qui n'est pas un état défendu et tel qu'il existe une séquence événements incontrôlables $\sigma \in E_{uc}$ qui conduit à un état interdit (q_i', v_j') de *D*. Dans ce contexte, l'ensemble des états faiblement interdits Q_F sera défini comme ci-dessous:

$$Q_F = \{(q_i, v_j) \mid (q_i, v_j) \in Q_A,\ (q_i', v_j') \in Q_I \text{ ou } (q_i', v_j') \in Q_F \text{ et } \sigma \in E_{uc} ; (q_i, v_j) \xrightarrow{\sigma} (q_i', v_j')\}$$

Où Q_A dénote l'ensemble des états possibles et autorisés par la spécification.

❑

Etape 4. On supprime de *D* l'ensemble des états interdits ainsi que l'ensemble des états faiblement interdits (ainsi que les transitions associées à ces états).

Nous supprimons de *D* l'ensemble des états non accessibles, c'est-à-dire, tout état (q_i, v_j) tel qu'il n'existe pas de chemin permettant de rejoindre (q_i, v_j) depuis l'état initial.

On appelle *D'* le modèle accepteur du fonctionnement désiré ainsi obtenu.

A partir des ensembles Q_A, Q_F et Q_I, on peut construire deux autres ensembles d'états qui nous seront très utiles dans la suite de ce travail :

- **L'ensemble des états interdits frontière Q_B :** Cet ensemble correspond aux états interdits ou faiblement interdits atteignables par occurrence d'événements contrôlables. Formellement, cet ensemble est défini ci-dessous.

Définition II.3.7. (L'ensemble des états interdits frontière) Soit Q_A l'ensemble des états admissibles ou autorisés par la spécification, Q_I l'ensemble des états interdits et Q_F faiblement interdits. L'ensemble des états interdits frontière Q_B est l'ensemble

$$Q_B = \{(q_i, v_j) \mid (q_i, v_j) \in \mathcal{M}_A,\ (q_i', v_j') \in \mathcal{M}_I \text{ ou } (q_i', v_j') \in \mathcal{M}_F \text{ et } \sigma \in E_c,\ (q_i, v_j) \xrightarrow{\sigma} (q_i', v_j')\}$$

❑

- **L'ensemble des états autorisés critiques, Q_{AC} :** Cet ensemble correspond aux états à partir desquels l'occurrence d'événements contrôlables mène à un état frontière.

Définition II.3.8. (L'ensemble des états autorisés critiques) L'ensemble des états autorisés critiques $\mathcal{M}_{AC}$ sera défini comme ci-dessous :

$$Q_{AC} = \{(q_i, v_j) \mid (q_i, v_j) \in \mathcal{M}_A,\ (q_i', v_j') \in \mathcal{M}_B \text{ et } \sigma \in E_c,\ (q_i, v_j) \xrightarrow{\sigma} (q_i', v_j')\}$$

❑

L'ensemble des états interdits frontières Q_B et l'ensemble des états autorisés critiques Q_{AC} sont essentiels dans la synthèse de contrôleurs par les RdP. En effet, par l'interdiction de franchissement des événements contrôlables d'états autorisés critiques vers les états interdits frontières, on empêche l'accessibilité de tous les états interdits. Tous les ensembles d'états de l'espace d'états d'un SED sont illustrés dans la figure II.7.

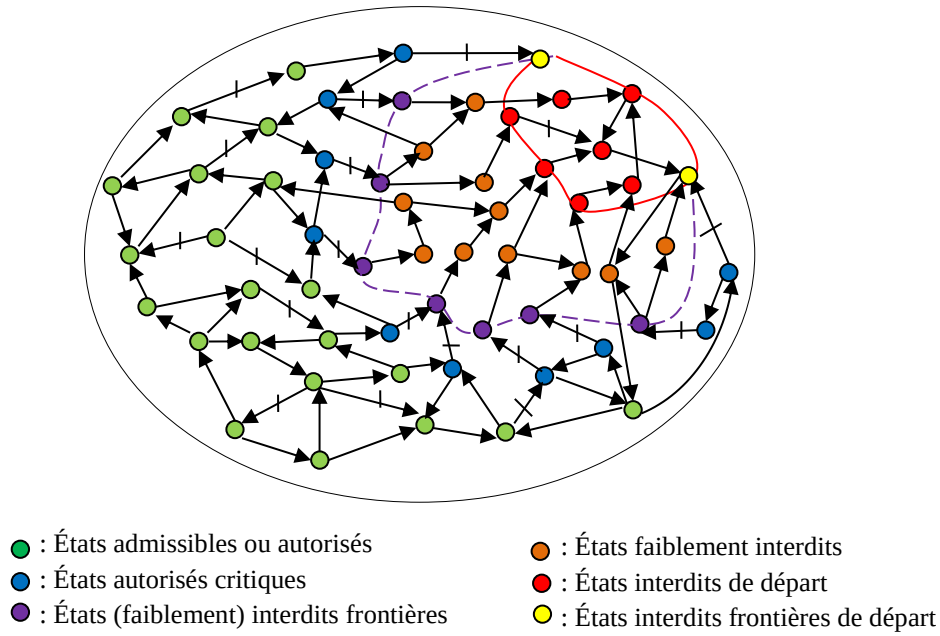


Figure II.7 – Ensembles d'états dans le modèle de fonctionnement du SED en boucle fermée

Exemple II.2.

Considérons un atelier de fabrication dont la disposition est illustrée à la figure II.8.

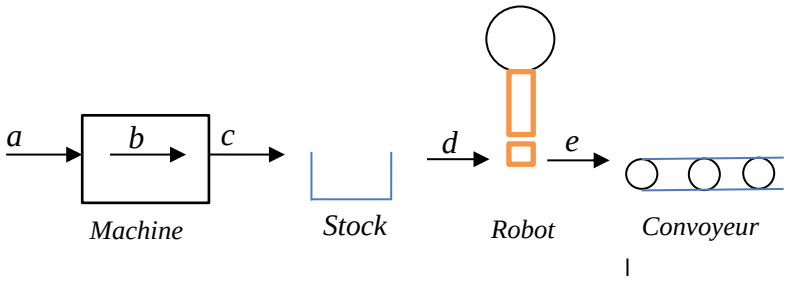
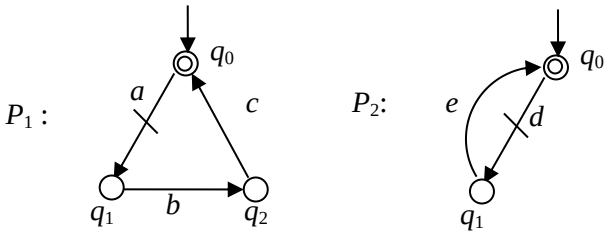


Figure II.8 – Disposition d'un atelier de fabrication

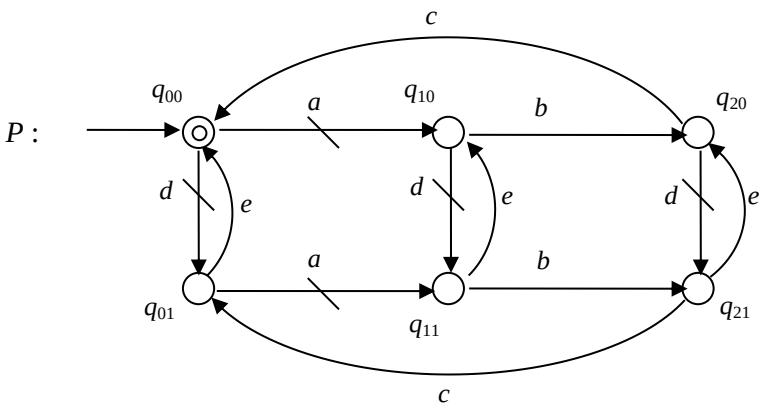
Les deux principales composantes de cet atelier sont la machine et le robot. La machine est chargée avec une pièce (évènement *a*) et après l'avoir usinée (évènement *b*) la délivre en sortie dans un stock (évènement *c*). Le robot prend une pièce du stock (évènement *d*) et la déplace (évènement *e*) sur un convoyeur qui l'enlève.

Modélisation du procédé : Chaque composant de cet atelier peut être modélisé par un automate (accepteur). Dans la figure II.9 (a), nous avons l'automate P_1 sur l'alphabet $E_1 = \{a, b, c\}$ qui décrit la machine, tandis que l'automate P_2 sur l'alphabet $E_2 = \{d, e\}$ décrit le robot. Le fonctionnement de l'atelier fabrication est décrit par l'automate $P = P_1 \| P_2$ sur l'alphabet E

= $E_1 \cup E_2$ obtenu par produit synchrone des deux composants. Ces automates représentés à la figure II.9 (b).



(*a*)



(*b*)

Figure II. 9 – (a) Les modèles de la machine P_1 et du Robot P_2 et (b) le modèle $P = P_1 || P_2$ du procédé global

Supposons que l'ensemble des évènements contrôlables est $E_c = \{a, d\}$ alors que l'ensemble des évènements incontrôlables est $E_{uc} = \{b, c, e\}$.

Modélisation de la Spécification : Le fonctionnement de notre atelier de fabrication doit respecter la spécification suivante. D'abord, le robot doit démarrer une opération de déplacement (évènement *d*) seulement après que la machine ait produit une pièce (évènement *c*). En outre, étant donné que le stock de capacité est limitée à 1, la machine devrait mettre une nouvelle pièce dans le stock seulement après que le robot ait pris la précédente. Le stock est supposé vide dans son état initial. Ceci est représenté par l'automate de spécification *S* dans la figure II.10.

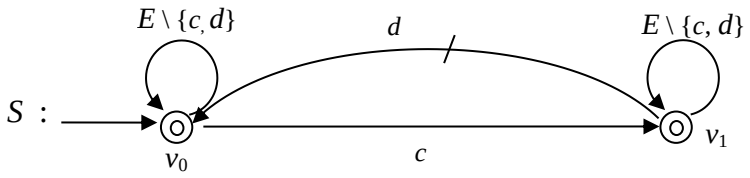


Figure II. 10. Modèle de la spécification pour l'atelier de fabrication

Lorsque le stock est vide, l'occurrence de l'événement contrôlable *d* est interdite (début du cycle du robot). Sur l'occurrence de l'événement *c* (fin du cycle de la machine et dépôt d'une pièce dans le stock), l'automate *S* change d'état et passe dans l'état v_1. Dans cet état, l'occurrence de l'événement *c* est interdite (fin du cycle fin du cycle de la machine). Nous montrons maintenant comment il est possible de vérifier, si l'automate donné *S* est admissible pour le procédé *P*, par application de l'algorithme de Kumar. La procédure nécessite de construire l'automate produit synchrone $D = P||S$ qui génère l'ensemble des états possibles du système. Notons que par construction chaque état de cet automate *D* est un couple (q_{ij}, v_k) où $q_{ij} \in Q$ est un état de *P* et $v_k \in V$ est un état de *S*. La figure suivante donne l'automate $D = P||S$ et l'ensemble des états interdits du procédé contrôlé.

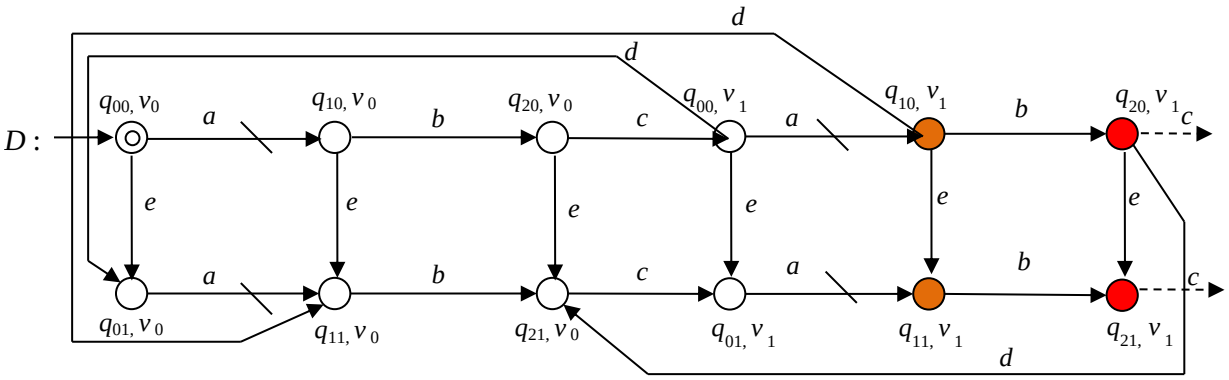


Figure II. 11 – Le Modèle automate $D = P||S$ du procédé contrôlé avec des états interdits

Nous pouvons observer ce qui suit.

- l'évènement incontrôlable *b* ne provoque pas des états incontrôlables dans *D*.
- l'évènement incontrôlable *c* provoque des états interdits dans *D*. En fait, il existe deux états $q_{20, v1}$ et $q_{21, v1}$ qui sont donc source d'incontrôlabilité

Dans la figure II.11, ces deux états interdits sont marqués en rouge foncé. En outre, nous avons aussi deux états faiblement interdits $q_{10,\ v\ 1}$ et $q_{11,\ v1}$ (marqué en rouge clair) à partir desquels une séquence σ = *b* d'évènements incontrôlables conduit à un état interdit.

- l'évènement incontrôlable *e* ne provoque pas des états interdits dans *D*.

Par conséquence, nous pouvons conclure que l'automate du SED vis-à-vis de la spécification *S* n'est pas admissible car, il contient des états interdits

– $Q_I = \{q_{20,\ v\ 1}\ ,\ q_{21,\ v\ 1}\}$.

L'ensemble d'états faiblement interdits est

– $Q_F = \{q_{10,\ v\ 1}\ ,\ q_{11,\ v\ 1}\ ,\ q_{20,\ v\ 1}\ ,\ q_{21,\ v\ 1}\}$.

Nous pouvons énoncer le résultat suivant.

Proposition II.3.1. Un automate de spécification *S* est admissible par rapport au procédé *P* si et seulement si l'automate composé *D* = *P*||*S* n'a pas d'état interdit.

❑

Comme l'automate du SED vis-à-vis de la spécification n'est pas admissible, il est nécessaire de restreindre davantage le comportement du procédé pour assurer qu'aucun état interdit ne sera atteint. Ceci peut se faire en retirant tous ces états interdits suivant l'étape 4 de l'algorithme de Kumar. Nous obtenons ainsi un automate final (figure II.12) qui est constitué des états admissibles et non faiblement interdits. Le modèle automate final caractérise le langage suprême contrôlable du fonctionnement désiré *supC*(*K*), *K*=*L*(*D*).

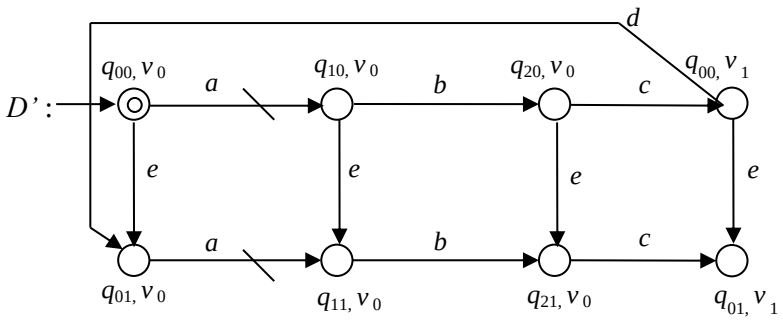


Figure II. 12 – Modèle final de procédé contrôlé sans états interdits (contrôleur)

Nous observons que le contrôleur construit par l'Algorithme de Kumar est :

– Admissible, car, il désactive uniquement les évènements incontrôlables ;

- correct, puisque, le système en boucle fermée génère seulement des états autorisés ;
- maximalement permissif, dans la mesure où il empêche le système de générer des états faiblement interdits.

Le modèle final de procédé contrôlé sans états interdits coïncide avec le contrôleur *C*. En fait, comme *C* affine *P*, nous avons $L(C) \subseteq L(P)$ et $L_m(C) \subseteq L_m(P)$. Ainsi, le système en boucle fermée *C*/*P* a un langage clos $L(C/P) = L(P) \cap L(C) = L(C)$ et langage marqué $L_m(C/P) = L_m(P) \cap L_m(C) = L_m(C)$.

▲

Remarque II.2. L'algorithme de Kumar fonctionne comme suit. D'abord, il vérifie si l'automate du SED en boucle fermée vis-à-vis de la spécification *S* est admissible. Dans ce cas, il peut être utilisé en tant que contrôleur, à savoir, $S = C$ et $C/P = P||S = D$. Si *D* a des états interdits et faiblement interdits, il permet d'éviter de les atteindre en retirant tous ces états. Nous avons le contrôleur *C* tel que $C/P = D'$. Un tel contrôleur est appelé un contrôleur monolithique (Wonham, 2011).

❑

II.4. Remarques sur l'approche initiée par Ramage & Wonham

La commande par supervision doit garantir le respect du cahier des charges (spécification) désiré quelle que soit la complexité du SED. L'approche classique initiée par Ramadge et wonham (1989) est confrontée au problème d'explosion combinatoire d'états, qui rend impossible la synthèse de contrôleurs pour des systèmes de taille raisonnable. Si le modèle du procédé comporte *n* états et le modèle de la spécification a *m* états, alors l'algorithme de Kumar permet de synthétiser un contrôleur comportant *n.m* états. Il apparaît clairement que maîtriser la taille des contrôleurs est un objectif crucial pour l'implantation de la théorie de commande par supervision de Ramadge et Wonham. Différents travaux d'extension de cette théorie, proposent des méthodes pour réduire la taille des modèles obtenus par la remise en cause du point de vue centralisé au profit d'un point de vue hiérarchique, modulaire sous observation partielle ou décentralisée (Lin & Wonham, 1994; Lin et *al*., 1990). Cependant, il reste encore beaucoup de difficultés de modélisation et d'implantation. D'autres extensions se penchent sur les possibilités de modélisation plus générales, telles que les systèmes hybrides (Sreenivas & Krogh, 1992 ; Uzam & Wonham, 2006), les SED à structure vectorielle (Li &

Wonham, 1994), et les réseaux de Petri, sur lesquels portent nos travaux. En effet, les réseaux de Petri offrent une représentation compacte des systèmes ce qui permet de réduire cette complexité et peut potentiellement être considérées comme un moyen efficace de la synthèse des contrôleurs, conformément à la théorie Ramadge et Wonham, puisqu'ils génèrent des langages.

Chapitre III

Synthèse de commande par supervision basée sur les réseaux de Petri

La théorie de commande par supervision basée sur les automates à états finis permet de réaliser la synthèse du contrôleur maximal. L'inconvénient majeur est l'explosion combinatoire du nombre d'états qui rend difficile la synthèse et l'implantation quand il s'agit de systèmes complexes. Dans ce chapitre, nous montrons les limites des travaux d'extension de cette théorie portant sur la modélisation par réseaux de Petri (RdP) d'une part ; d'autre part, celles des méthodes les plus connues pour la synthèse de contrôleur basée sur les RdP. Pour ce faire, nous clarifions les concepts fondamentaux relatifs à ces méthodes.

Chapitre III

Synthèse de commande par supervision basée sur les réseaux de Petri

III.1. L'usage du RdP dans la théorie de commande par supervision

Les automates à états finis ont fourni un cadre formellement général pour l'établissement des propriétés fondamentales pour la commande par supervision des SED. Mais, le manque de structure dans les modèles automates rend ces modèles inadaptés pour décrire des systèmes complexes. Et le phénomène d'explosion combinatoire du nombre d'états inhérent, est un frein majeur à l'implantation de la théorie de commande par supervision. Comme nous l'avons souligné au chapitre 1, les RdP sont un palliatif approprié à cause de leur puissance de représentation graphique et de leurs propriétés structurelles. Leur usage dans la synthèse de commande par supervision est plus récent et divers (Holloway & Krogh, 1990 ; Uzam & Wonham, 2006). En tant que générateurs de langages (Lafortune & Yoo, 1990; Cassandra et Lafortune, 2008), ils représentent un bon compromis entre la modélisation structurelle et la synthèse de contrôleur maximal permissif pour des spécifications d'états ou de langages (Kumar & Holloway, 1996).

Définition III.1.1. (**Contrôleur maximal permissif**) Un contrôleur est maximal permissif si et seulement si, tout état admissible du SED est accessible, et tout état interdit ne l'est pas sous supervision. Un tel contrôleur est dit optimal.

❑

Les états d'un SED sont donnés par le modèle global de fonctionnement en boucle fermée obtenu par l'opération de produit synchrone. Pour ce qui est des RdP, le produit synchrone est une synchronisation structurelle entre les RdP du procédé et celui des spécifications, qui permet d'avoir un modèle de fonctionnement en boucle fermée de taille réduite. Cependant, un problème de contrôlabilité peut apparaître lorsque la synchronisation est réalisée par des

transitions associées aux événements incontrôlables. Il est admis que, l'existence des transitions incontrôlables conduit souvent à l'existence des états interdits, qui sont systématiquement déterminés par l'algorithme de Kumar (Kumar, 1991).

Considérons, l'exemple I.5 du système manufacturier classique présenté au chapitre 1 auquel nous imposons une spécification de stock.

Spécification *:* Le fonctionnement du système manufacturier doit respecter la présence d'un stock de capacité limitée à 1, situé entre les 2 machines (figure III.1.). Nous supposons donc à présent que les machines travaillent en série. La présence du stock entre M1 et M2 impose que la machine M2 ne peut commencer à travailler que si elle peut prendre une pièce dans le stock, c'est-à-dire, si le stock est plein. Et la machine M1 ne peut déposer une pièce dans le stock que si celui-ci est vide. Le stock est supposé vide dans son état initial.

Figure III.1 – Le système manufacturier sous la spécification de stock

Les RdP du procédé et de la spécification sont présentés à la figure III.2 suivante

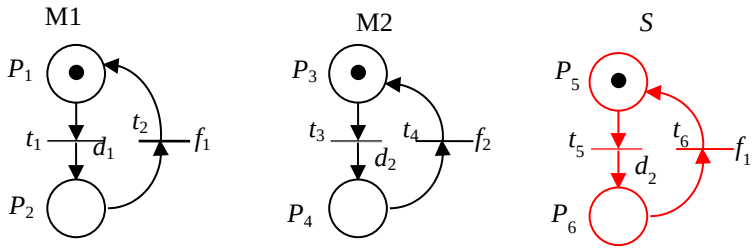


Figure III.2 – RdP du système manufacturier et de la spécification de stock

Le produit synchrone des RdP du procédé (M1 et M2) et de la spécification donne le RdP du fonctionnement du SED en boucle fermée (figure III.3).

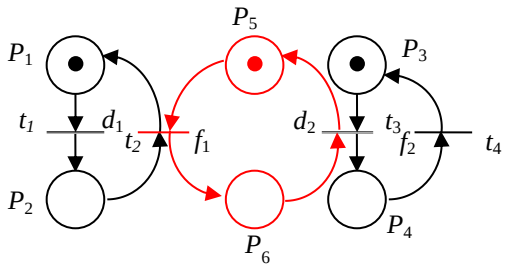


Figure III.3 – Modèle RdP du système manufacturier classique en boucle fermée

Les événements d_1 et d_2 sont contrôlables et évènements f_1 et f_2 sont incontrôlables. Les événements communs au modèle du procédé et à la spécification sont f_1 et d_2. Dans la dynamique du RdP du système en boucle fermée, il y aura potentiellement des états interdits à cause de la synchronisation via l'événement incontrôlable f_1.

▲

Dans le fonctionnement en boucle fermée, une transition est franchissable si elle est à la fois validée par rapport au procédé et à la spécification. Les états interdits se produisent lorsque les places en amont d'une transition incontrôlable appartenant au procédé sont marquées et les places appartenant à la spécification ne le sont pas. Considérons le RdP du système en boucle fermée, représenté dans la Figure III. 3, dont le graphe de marquages, est équivalent à un automate, donné dans la Figure III. 4. Un état est interdit lorsque la place P_2 en amont de la transition t_2 est marquée et simultanément, la place P_5 en amont de la transition t_2 n'est pas marquée. Un tel état est accessible à partir de l'état initial M_0 via des séquences de franchissement de transitions.

Nous pouvons à partir du graphe de marquages, calculer formellement l'ensemble des états interdits en appliquant l'algorithme de Kumar (Kumar et *al.*, 1996), présenté dans la section II.3.2 du chapitre 2. Soient T_c et T_{uc} les ensembles des transitions associés aux événements contrôlables et, respectivement, incontrôlables. Soient N_P le modèle RdP synchronisé du procédé à contrôler, N_S celui de la spécification à lui imposer, et $N = N_P||N_S = \langle R, E, f\rangle$ où $R = \langle P, T, W^-, W^+, M_0\rangle$, le modèle RdP synchronisé du système en boucle fermée correspondant. L'algorithme de Kumar peut être définit dans le contexte des RdP, à savoir :

Pas 1. Construire le graphe de marquage de $N = N_P||N_S$

Pas 2. Déterminer de l'ensemble d'états interdits, $\mathcal{M}_I$.

Définition III.1.2. (État interdit) On appelle état interdit, tout marquage M_k de $N = N_p||N_s$ état (q_i, v_j) de D tel qu'il existe une transition incontrôlable $t_j \in T_{uc}$ où $M_k[t_j\rangle$ est défini dans N_p et $M_k\,[t_j\rangle$ n'est pas définie dans N_s . L'ensemble des états interdits $\mathcal{M}_I$ sera défini comme ci-dessous :

$$\mathcal{M}_I = \{M_k \mid \exists\ t_j \in T_{uc} \text{ avec } M_k\,[t_j\rangle \text{ défini dans } N_p, \text{ et non défini dans } N_s\ \}$$

❑

Pas 3. Détermination de l'ensemble d'états faiblement interdits, $\mathcal{M}_F$.

Définition III.1.3. (État faiblement interdit) Étant donné le graphe de marquages de $N = N_p||N_s$ et un ensemble d'états admissibles $\mathcal{M}_A$, nous définissons l'ensemble d'états faiblement interdits

$$\mathcal{M}_F = \{\ M_k \in \mathcal{M}_A \mid \exists\ t_j \in T_{uc}\ ;\ M_k[t_j\rangle \in \mathcal{M}_I\}$$

contenant tous les états du procédé à partir desquels un état interdit est accessible par une séquence de franchissement des transitions incontrôlables.

❑

L'ensemble des états admissibles de $N = N_p||N_s$ est donné par $\mathcal{M}_A = \mathcal{A}(N, M_0)\backslash\{\ \mathcal{M}_I \cup \mathcal{M}_F\}$

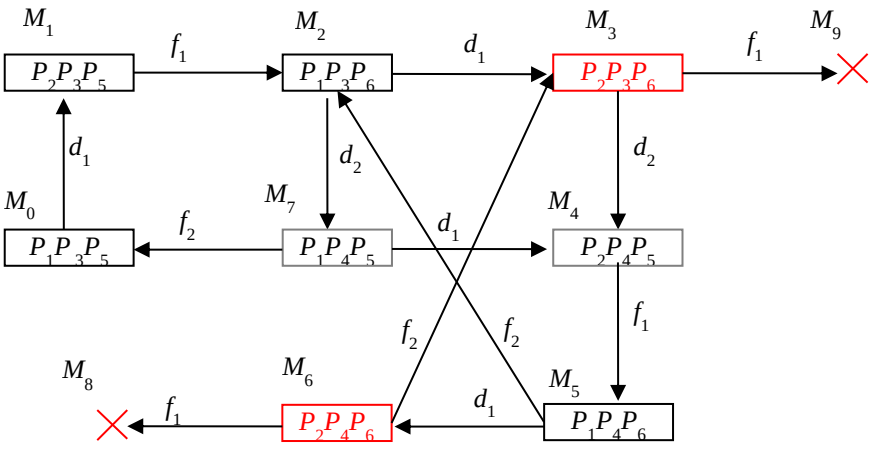


Figure III.4 – Graphe de marquages du RdP système manufacturier classique avec la spécification de stock

- l'ensemble des états interdits est $\{M_8, M_9\}$;
- l'ensemble des états faiblement interdits est $\{M_3, M_6\}$;
- l'ensemble des états admissibles est $\{M_0, M_1, M_2, M_4, M_5, M_7\}$.

Pour ne pas atteindre l'état M_3, il faut interdire la transition d_1 à partir de l'état M_2. De même, pour ne pas atteindre l'état M_6, il faut interdire la transition d_1 à partir de l'état M_5.

▲

La synthèse du contrôleur optimal consiste à supprimer l'ensemble des états interdits et faiblement interdits du graphe de marquages (Pas 4 de l'algorithme). Cela fait qu'il rend quasiment impossible le retour au RdP de départ. Pour pallier à cet inconvénient, les états interdits sont transformés en contraintes linéaires dans plusieurs méthodes de synthèse de commande basées sur les RdP (Giua et *al*., 1992 ; Boel et *al*., 1995 ; Luo. & Nonami, 2011).

III.2. Contraintes linéaires associées aux états interdits dans le RdP

Dans la synthèse de commande par supervision, les spécifications sont souvent données comme un ensemble d'états interdits en présence de transitions associées aux événements incontrôlables. Leur modélisation par RdP est plus facile que par automates à états finis ; bien qu'il soit évident d'avoir des restrictions pour certaines classes de RdP comme les graphes d'événements cycliques et saufs (Holloway & Krogh, 1990), qui ne modélisent pas les spécifications comportementales (situations de conflit). En général, les spécifications d'états interdits sont représentées par des contraintes linéaires entre les marquages des places du RdP (Kattan, 2004), dites GMEC (General Mutual Exclusion Constraint). Une GMEC est une condition qui limite la somme pondérée de jetons dans un ensemble de places. Il est possible d'interdire l'ensemble des états interdits par ces contraintes linéaires. En particulier si l'ensemble des états admissibles est exprimé par un ensemble de GMEC et si tous les évènements sont contrôlables, alors une solution à base de RdP est maximale permissive (Dideban & Alla, 2008).

Définition III.2.1. (GMEC) Soit R, un RdP marqué avec l'ensemble de places P. Une GMEC (l, b) définit un ensemble des états admissibles :

$$\mathcal{M}(l, b) = \{M_k \in \mathcal{A}(R, M_0) \mid l^T.M_k \leq b\}$$

Où $l \in \mathbb{Z}^n$ est le vecteur (colonne) des poids de la contrainte dans $\mathbb{Z}$, ensemble des entiers, $\mathcal{A}(R, M_0)$ est l'espace d'états accessibles du RdP, et $b \in \mathbb{Z}^n$ est la borne de la contrainte.

❑

Si le vecteur de pondération est binaire $l \in \{0,1\}^n$, alors la GMEC est non-pondérée.

Cependant, il est possible de trouver un ensemble des GMEC non-pondérés, équivalent pour toute GMEC pondérée qui décrit le fonctionnement désiré du système. Une étude détaillée est présentée dans (Guia, 2003). Généralement, une spécification comprend un ensemble des contraintes :

$$\mathcal{M}(l, b_v) = \bigcap_{i=1}^{r} M(l_i, b_i)$$

$$= \{M_k \in A(R, M_0) \mid L^T.M_k \leq b_v\}, \qquad \text{(III-1)}$$

Où $L = [l_1 \quad l_2 \ \ldots \ l_r]$, $b_v = [b_1 \ b_2 \ \ldots \ b_r]^T$ et $\mathcal{M}(l, b_v)$ représente l'ensemble des états admissibles par l'ensemble des contraintes

Lorsque l'espace d'états accessibles, $\mathcal{A}(R, M_0)$ est égal à $\mathcal{M}(l, b)$, la contrainte $\mathcal{M}(l, b)$ est redondante vis-à-vis de $\mathcal{A}(R, M_0)$. On peut ainsi trouver deux ensembles (l_1, b_{v1}) et (l_2, b_{v2}) de contraintes qui sont équivalentes par rapport au RdP si :

$$\mathcal{A}(R, M_0) \cap \mathcal{M}(l_1, b_{v1}) = \mathcal{A}(R, M_0) \cap \mathcal{M}(l_2, b_{v2}) \qquad \text{(III-2)}$$

Pour des RdP saufs et conservatifs, il n'est pas toujours possible de décrire l'ensemble des états interdits par des contraintes linéaires sauf (Giua et *al.*, 1992).

Définition III.2.2. (RdP Conservatif) Un RdP est dit conservatif, s'il existe un vecteur de pondération des places X à composantes strictement positives tel que, pour tout état initial M_0 et tout état M_k accessible à partir de l'état M_0 :

$$X^T.M_k = X^T.M_0$$

❑

Théorème III.2.1. Soit $R = \langle P, T, W^-, W^+, M_0 \rangle$ un RdP sauf et conservatif et soit $\mathcal{M}_I$ un ensemble des états interdits. Il y a un ensemble des contraintes GMEC (l, b) tel que :

$$\mathcal{A}(R, M_0) \backslash \mathcal{M}_I = \mathcal{A}(R, M_0) \cap \mathcal{M}(l, b)$$

❑

La démonstration formelle se trouve dans (Giua et *al.*, 1992) mais, il peut être présenté intuitivement par un exemple. Considérons le graphe de marquage de la figure III.4. Par la spécification de stock nous avons les états interdits, au mieux faiblement interdits $M_3 = P_2P_3P_6$ et $M_6 = P_2P_4P_6\}$. Si nous prenons le premier état interdit M_3, la seule possibilité pour

que les places P_2, P_3 et P_6 soient marquées en même temps est définie par l'état $P_2P_3P_6$. Cet état est facilement interdit par la contrainte $M_3(P_2) + M_3(P_3) + M_3(P_5) \leq 2$, où $M_3(P_2)$, $M_3(P_3)$ et $M_3(P_5)$ présentent les nombres des marques dans les places P_2, P_3 et P_6. Nous pouvons interdire chacun des états interdits par une contrainte linéaire.

$P_2P_3P_6$ interdit $\Leftrightarrow M_3(P_2) + M_3(P_3) + M_3(P_5) \leq 2$

$P_2P_4P_6$ interdit $\Leftrightarrow M_6(P_2) + M_6(P_3) + M_6(P_5) \leq 2$

Maintenant, supposons que les RdP du SED sont saufs et soit M_k un état interdit dont le support est $Support\ (M_k) = \{P_1\ P_2\ P_3\ \ldots\ P_n\}$ et correspond à l'ensemble des places marquées qui le constituent.

Définition III.2.3. (***Support* (M_k)**) La fonction *Support* (M_k) d'un vecteur d'état $M_k \in \{0,1\}^N$ est telle que : *Support* (M_k) $= \{P_1\ P_2\ P_3\ \ldots\ P_n\}$, ensemble des places marquées dans le vecteur d'état M_k

❑

Par exemple, le support du vecteur $M_0 = [1, 0, 1, 0, 0, 1, 0]^T$ est : $Support(M_0) = \{P_1, P_3, P_6\}$

Dans le cas général, l'état M_k peut être interdit par la contrainte équivalente suivante

$$\sum_{i=1}^{n} M_k(P_i) \leq n-1 \qquad \text{(III.3)}$$

Où n est le nombre des places marquées dans l'état M_k , et $M_k(P_i)$ est le marquage booléen des place P_i dans l'état M_k.

Dans le cas où le RdP du procédé et celui des spécifications sont tous deux généralisés, les contraintes linéaires GMEC pondérées ($l \notin \{0, 1\}^n$) peuvent être construites à partir des états interdits (Kattan, 2004) de la manière suivante :

$$\sum_{i=1}^{n} l_i^T . M_k(P_i) \leq b \qquad \text{(III.4)}$$

Où : $M_k(P_i)$ est le marquage de la place P_i, l_i et b sont les nombres entiers tels que : l_i représente le poids du marquage d'une place dans la contrainte et b la borne.

III.2.1. Simplification des contraintes linéaires

Dans les systèmes réels, il y a un grand nombre d'états interdits, donc un grand nombre de contraintes linéaires. Selon Dideban, (2007), il est possible de simplifier les contraintes par une méthode systématique qui, permet d'arriver toujours au résultat le plus simple possible. Par exemple, pour les RdP conservatifs une simplification des contraintes est possible en utilisant les équations déduites des invariants à l'aide des propriétés basés sur les invariants ci-après.

Propriété III.2.1. Soit $\mathcal{M}_I = \{(P_1P_k \ldots P_i), (P_2P_k \ldots P_i)...,(P_r P_k \ldots P_i)\}$ un sous-ensemble des états interdits quelconque d'un graphe de marquage et $m_1 + m_2 + ...m_r = 1$ un invariant de marquage du RdP . Les r contraintes linéaires déduites de l'ensemble des états interdits sont équivalentes à une seule contrainte :

$$m_1 + m_k + \ldots + m_i \leq n\text{-}1$$

$$m_2 + m_k + \ldots + m_i \leq n\text{-}1$$

$$\ldots \qquad \xleftarrow{(m_1+m_2+\ldots m_r=1)} \qquad m_k + \ldots + m_i \leq n\text{-}2$$

$$m_r + m_k + \ldots + m_i \leq n\text{-}1$$

Où n est le nombre de places marquées. (III.5)

❑

Il est possible d'utiliser les inégalités pour générer de nouvelles simplifications

Propriété III.2.2 Soit $M_I = \{(P_1P_i \ldots P_j, k)\ (P_2P_i \ldots P_j, k),...,(P_r P_i \ldots P_j, k)\}$ un sous-ensemble de contraintes quelconque (états interdits et borne correspondante) et $m_1 + m_2 + ...m_r \leq 1$. Les r contraintes linéaires sont équivalentes à une seule contrainte comme ci-dessous :

$$m_1 + m_i + \ldots + m_j \leq k$$

$$m_2 + m_i + \ldots + m_j \leq k$$

$$\ldots \qquad \xleftarrow{(m_1+m_2+\ldots m_r \leq 1)} \qquad (m_1 + m_2 + \ldots + m_r) + m_i + \ldots + m_j \leq k$$

$$m_r + m_i + \ldots + m_j \leq k \qquad \text{(III.6)}$$

Dans le graphe de marquages de la figure III. 4 présenté, il y a deux états interdits $P_2P_3P_6$ et $P_2P_4P_6$ dans lesquels les places P_2 et P_6 ne peuvent pas être marquées en même temps. Les

deux états peuvent être interdits par la contrainte $M(P_2) + M(P_6) \leq 1$.

Cette simplification est faite de manière manuelle. Nous comprenons qu'en diminuant le nombre de contraintes, on simplifie considérablement la synthèse des contrôleurs. C'est pourquoi, le travail que nous proposons au chapitre 4, nous pourrons effectuer la simplification des contraintes linéaires de manière systématique si cela est nécessaire.

III.2.2. Contrainte linéaire et problème de blocage

Dans le graphe de marquages d'un RdP, nous pouvons retrouver en plus des états admissibles et les états interdits imposées par la spécification, les états correspondants aux problèmes de vivacité et de blocage. Le RdP est dit bloqué si aucune transition n'est activée. Ceci est une condition inadmissible dans les SED réelles. Ainsi, le blocage est un problème majeur à résoudre lors de la synthèse d'un contrôleur. Une procédure intéressante de prévention des blocages est la notion structurelle de siphons (Giua & DiCesare , 1994 ; Wu & Zhou, 2004) qui consiste à ajouter au RdP du procédé des places supplémentaires afin d'éviter d'atteindre des siphons vides (Iordache et *al.*, 2001 ; Wu & Zhou, 2004). Le problème de blocage apparaît lorsqu'un siphon devient vide.

Définition III.2.4. Un siphon est un ensemble de places, $P_s \subseteq P$, tel que, pour tout $p_i \in P$, l'ensemble des transitions d'entrée de p_i, noté

$${}^{\bullet}p_i = \{t_j \in T \mid (t_j, p_i) \in W^{+}(\bullet, t_j)\},$$

est inclus dans son ensemble des transitions de sortie, noté

$$p_i^{\bullet} = \{t_j \in T \mid (p_i, t_j) \in W^{-}(\bullet, t_j)\},$$

c'est-à-dire, ${}^{\bullet}p_i \subseteq p_i^{\bullet}$.

Le blocage apparaît lorsque le siphon devient vide.

❑

Nous pouvons utiliser les siphons pour faire face aux situations où les états interdits sont causés par des problèmes de vivacité et blocage (Ezpeleta et *al.*, 1995). Dans ce cas, la condition de contrôle peut être décrite par une contrainte linéaire. De plus, si les spécifications à imposer au procédé sont données en termes d'états interdits, il est évident que nous puissions exploiter les propriétés structurelles des RdP pour résoudre le problème de blocage. C'est-à-dire, changer le paradigme de commande par supervision du point de vue du langage au point de vue de l'état. C'est ce qui a donné lieu au développement des méthodes de

synthèse de contrôleurs basées sur les RdP. Nous présentons schématiquement quelques-unes des méthodes les plus connues (Giua & DiCesare, 1994)

III.3. Synthèse de contrôleur par la méthode des invariants de place

Les spécifications imposées au procédé à contrôler étant données par des contraintes linéaires, il est possible que le comportement dynamique du procédé à contrôler viole certaines contraintes. Dans ce cas, un contrôleur doit être synthétisé. Les invariants de place peuvent être utilisés pour calculer de manière simple le contrôleur en boucle fermée avec un système modélisé par un RdP de fonctionnement en boucle fermée (Kumar & Holloway, 1996; Gaudin, 2004). L'idée consiste à construire un invariant de marquage pour chaque contrainte, de telle façon que le controleur couplé RdP de fonctionnement en boucle fermée puisse garantir que l'ensemble d'états interdits ne sera pas atteint. Le rôle du contrôleur est de surveiller le système et de prendre des actions correctives pour éliminer tout comportement indésirable. Nous allons détaillér cette approche, en insistant particulièrement sur le problème de l'optimalité de la solution obtenue, parce qu'elle sera utilisée dans notre travail .

III.3.1. Description de la méthode

Considérons un SED modélisé par RdP qui a besoin de supervision. Soient $W_P \in \mathbb{Z}^{n \times m}$ avec n places et m transitions, la matrice d'incidence du procédé R_P; et $W_C \in \mathbb{Z}^{nc \times m}$ constitué des transitions du procédé R_P et d'un ensemble séparé de places, la matrice d'incidence du controleur C. Le controleur couplé au procédé constitue le SED contrôlé dont la matrice d'incidence est $W \in \mathbb{Z}^{(n + nc) \times m}$. Chaque place utilisée pour contrôler le procédé ajoute une ligne à la matrice d'incidence W du SED controlé. Donc, W est composé de deux matrices à savoir la matrice du procédé original W_P et la matrice d'incidence du contrôleur, W_C .

$$W = \begin{bmatrix} W_P \\ W_C \end{bmatrix}, \; M_0 = \begin{bmatrix} M_{P0} \\ M_{C0} \end{bmatrix} \tag{III.7}$$

Les arcs connectant les places du contrôleur au RdP du procédé à contrôler sont calculés par l'équation d'invariant de place $X^T.W = 0$; où les inconnus sont les éléments de la nouvelle ligne de W et le vecteur P-semi-flot X est l'invariant de place désiré, donné par $X^T = [l_1\ l_2\ \ldots\ l_n, 1]$. L'objectif du contrôleur est de forcer le procédé à respecter les contraintes de la forme

$$\sum_{i=1}^{n} l_i . M_k(P_i) \leq b, \ \forall M_k \in \mathcal{A}(R, M_0) \quad \text{(III.8)}$$

Où $M_k(P_i) \in \mathbb{Z}^n$, $M_k(P_i) \geq 0$ est le vecteur de marquage du procédé, $l_i \in \mathbb{Z}^n$

De manière générale toutes ces contraintes peuvent être groupées et écrites dans une forme matricielle comme

$$L^T . M_k \leq b_v \quad \text{(III.9)}$$

Où $L = [l_1\ l_2\ \ldots\ l_n] \in \mathbb{Z}^{nc\ x\ n}$ represente la matrice de pondération des contraintes , $b_v \in \mathbb{Z}^{nc}$ est le vecteur des bornes.

La matrice L permet de calculer de manière algébrique la matrice d'incidence du contrôleur et son état initial comme le souligne le théorème III.3.1. Nous pouvons incorporer les spécifications en ajoutant à l'inégalité des contraintes un ensemble de variables élémentaires positives et entières $M(P_c) = b_v - L^T.M_k$. Ces variables représentent les nouvelles places P_C qui contiennent les marquages supplémentaires nécessaires pour satisfaire l'égalité. L'invariant de marquage qui assure le respect des contraintes dans R_P devient

$$L^T . M_k + M(P_C) = b_v \Leftrightarrow \left[L^T\ I \right] M_k = b_v \quad \text{(III.10)}$$

Où $M(P_C) \in \mathbb{Z}^{nc}$, $M(P_C) \geq 0$ est l'ensemble des marquages du controleur et I est la matrice identité de dimension n_c

III.3.2. Calcul du contrôleur

Le controleur est calculé en observant que l'introduction des variables élementaire impose un ensemble d'invariants de place sur le procédé controlé. Il se compose de places reliées aux transitions du procédé à controler, R_P par des arcs calculés à l'aide du concept d'invariants de marquage de place. Comme le vecteur de pondération $X^T = [L^T, I]$ caractérise l'invariant de place de R_P, alors $X^T W = 0 \Leftrightarrow [L^T, I].[W_P\ W_C]^T = 0$. Ce qui permet d'écrire

$$L^T W_P + W_C = 0 \Rightarrow W_C = - L^T W_P \quad \text{(III.11)}$$

L'état initial du contrôleur M_{C0} est calculé à partir de l'invariant de marquage et de l'état initial du procédé M_{P0}

$$X^T.M_0 = b_v$$

$$L^T M_{P0} + M_{C0} = b_v \Rightarrow M_{C0} = - L^T M_{P0} \qquad \text{(III.12)}$$

Théorème III.3.1. Synthèse de controleur basé sur l'invariant : si $b - L^T M_{P0} \geq 0$ alors le RdP du contrôleur est defini par $W_C \in \mathbb{Z}^{nc \times m}$ avec l'état initial $M_{C0} \in \mathbb{Z}^{nc}$ tel que : $W_C = - LW_P$ et $M_{C0} = b_v - L^T M_{P0}$

❑

Si l'inegalité $b - L^T M_{P0} \geq 0$ n'est pas vraie, alors les contraintes ne peuvent pas etre imposées par le contrôleur parce que les conditions initiales du procédé se trouvent hors du cadre autorisé par les contraintes. Selon les règles d'evolution du RdP, le contrôleur pourra desactiver les transitions seulement quand le franchissement indiqué causerait la negaticité d'au moins une de ces places.

Exemple III.1.

Considérons à nouveau l'exemple classique du système à deux machines, M1 et M2, qui travaillent en tandem. Dans ce scénario nous supposons que la nouvelle spécification defini la capacité du stockage limité à 1, avec des implications suivantes :

- la machine M2 ne peut commencer l'usinage que si le stock est plein,
- la machine M1 ne peut déposer une pièce dans le stock que si le stock est vide.

Le RdP du système en boucle fermée modifié, est présenté dans la figure III.5.

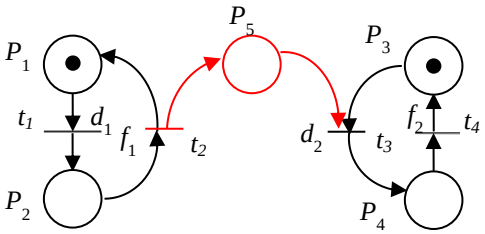


Figure III.5 –Modèle RdP du système en boucle fermée avec la nouvelle specification

Considérons le RdP de la figure III.5, et supposons que l'objectif de la nouvelle spécification est de limiter le marquage de la place P_5 à 1. Cette spécification peut être représentée par la contrainte suivante :

$M(P_5) \leq 1$, avec $b = 1$ et $L^T = [0\ \ 0\ \ 0\ \ 0\ \ 1]$

La matrice d'incidence et le marquage initial du système en boucle fermée sont déterminés à partir du modèle RdP :

$$W_P = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 1 & -1 & 0 \end{bmatrix} \text{ et } M_0 = [1\ 0\ 1\ 0\ 0]^T$$

A partir de la contrainte $L^T = [0 \quad 0 \quad 0 \quad 0 \quad 1]$, nous pouvons calculer par la méthode des invariants de place la matrice incidence du contrôleur et l'état initial de la place de contrôle

$$W_C = - L^T W_P = [0\ -1\ 1\ 0]$$

$$M_{C0} = 1 - 0 = 1$$

Le RdP final du SED contrôlé est présenté dans la figure III.6.

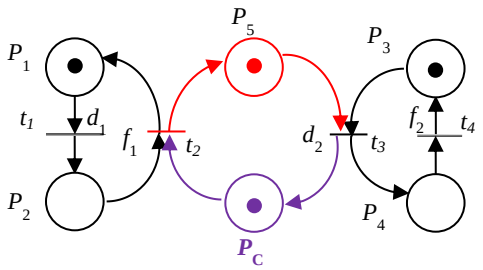


Figure III.6 – Modèle RdP contrôlé du système classique avec la nouvelle spécification

▲

Le contrôleur (place de contrôle) connecté par synchronisation structurelle au RdP du procédé a pour rôle de le surveiller et de prendre des actions correctives pour éliminer tout comportement indésirable. En effet, si la place de contrôle P_C n'est pas marquée, alors la transition t_2 doit être bloquée. Malheureusement, le contrôleur ne peut pas interdire le franchissement de t_2 puisqu'elle est associée à l'événement incontrôlable f_1. Par conséquent, le contrôleur obtenu de cette manière ne peut garantir le respect de la nouvelle spécification. Il est évident que la solution est non contrôlable et le contrôleur est extrêmement sous-optimal (figure III.7). Pour aboutir a une solution controlable, il faudrait que les événements impliqués dans l'évolution du procédé soient contrôlables. Or, dans un procédé réel l'existence des événements incontrôlables est indéniable. Dans ce contexte, il est possible de garantir à priori un fonctionnement conforme aux spécifications en utilisant l'algorithme de Kumar en conjonction avec la méthode des invariants de place

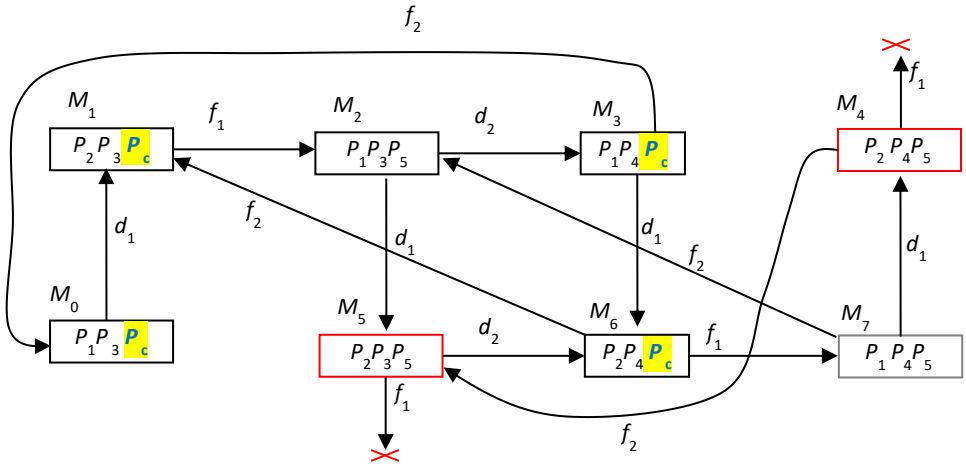


Figure III.7 – Graphe de marquage du RdP contrôlé de la figure III.6

L'exploration du graphe de marquage de la figure III.7 et l' identification des états interdits nous permettent d'obtenir la solution de contrôle optimal.

L' ensemble d'états faiblement interdits est: $\{P_2P_3P_5, P_2P_4P_5\}$

Le controleur doit empecher le système à violer la contrainte $M(P_5) \leq 1$. Il suffit d'associer à ces états faiblement interdits les contranites linéaires équivalentes, nous avons :

$$P_2P_3P_5 \text{ interdit} \Leftrightarrow M_5(P_2) + M_5(P_3) + M_5(P_5) \leq 2$$

$$P_2P_4P_5 \text{ interdit} \Leftrightarrow M_4(P_2) + M_4(P_4) + M_4(P_5) \leq 2$$

La simplification donne :

$$M_5(P_2) + M_5(P_3) + M_5(P_5) \leq 2$$

$$M_4(P_2) + M_4(P_4) + M_4(P_5) \leq 2 \xleftrightarrow{(M(P_4)+M(P_3)=1)} M(P_2) + M(P_5) \leq 1$$

Le vecteur contrainte est : $L^T = [0\ 1\ 0\ 0\ 1]$

Le contrôleur est défini par : $W_C = -L^T W_P = [-1\ 0\ 1\ 0]$ et $Mco = 1 - 0 = 1$

Nous obtenons le RdP contrôlé de la figure III.8 dont le graphe de marquage est representé

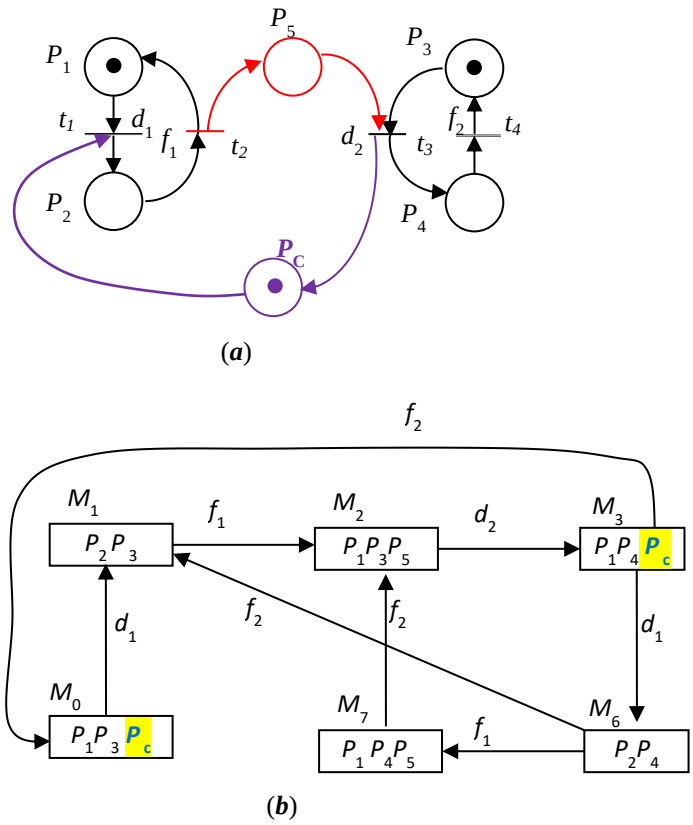


Figure III.8. – a) Modèle RdP sous contrôle optimal et b) Graphe de marquage admissible

Le contrôleur obtenu ici est contrôlable et garantit un contrôle maximal permissif tel que $M(P_5) \leq 1$ (le contenu du stock est toujours plus petit que 1). Si la place de contrôle P_c n'est pas marquée, alors la transition t_1 associée à l'événement contrôlable d_1 est bloquée. Ceci est juste un cas favorable, en général, cela ne se passe pas aussi bien en présence de l'incontrôlabilité : il est possible que des états interdits supplémentaires soient générés lors de la synchronisation entre les RdP du procédé à contrôler et du contrôleur.

▲

La méthode des invariants est très puissante car elle permet de synthétiser facilement et de manière structurelle un contrôleur à partir des contraintes qui peuvent etre simplifiées (Kattan 2004). Le principal inconvenient est qu'elle ne donne pas en général une solution optimale. Par conséquent, trouver de manière systématique le contrôleur maximal permissif, necessite la construction du graphe de marquage et la détermination des contraintes adéquates, qui prennent en compte les (éventuels) états faiblements interdits et les situations de blocage.

III.3.3. Limites du contrôleur vis-à-vis des transitions incontrôlables

Le RdP du contrôleur synthétisé par l'approche développée ici est donné par : $W_C = - LW_P$

Soit W_I la sous-matrice d'incidence représentant la partie incontrôlable. Elle contient les colonnes de W_P qui correspondent aux transitions incontrôlables (Basile, 2006). Soit LW_I la sous-matrice de LW_P qui correspond aux transitions incontrôlables du procédé. Le contrôleur viole les contraintes si LW_I contient au moins un élément strictement positif, c'est-à-dire, s'il y a des arcs partant de la place de contrôle vers une transition incontrôlable. Si LW_I contient des valeurs positives, les contraintes ne sont pas satisfaites. C'est juste une condition nécessaire, mais pas suffisante, qui correspond à une source potentielle d'incontrôlabilité. Dans notre cas, où l'événement f_1 est incontrôlable, le contrôleur n'est pas capable d'interdire l'occurrence de l'événement, ni par conséquent le franchissement de la transition. Pour résoudre ce problème (sans passer par le graphe de marquage comme nous l'avons fait), une méthode intuitive consiste à remonter les branches jusqu'à trouver une transition contrôlable qui soit en aval de place de contrôle (Yamalidou et *al*. 1996). Donc, il s'agit de remonter jusqu'à la transition associée à l'évenement d_1 qui est contrôlable. Nous pouvons aisément comprendre que cette méthode n'est pas toujours favorable. En réalité, si l'évenement d_1 est supposé incontrôlable, nous sommes amenés à remonter encore et encore la branche du RdP. Ce qui peut provoquer l'interdiction de certains états admissibles ou un blocus. De fait, cette technique pas formelle et nous pouvons perdre l'optimalité. Pour résoudre ce problème il est préférable de modifier les contraintes L tel que, les nouvelles contraintes prennent en compte l'incontrôlabilité selon la démarche proposée par (Moody & Antsaklis, 2000). Le principal inconvénient est qu'elle ne donne pas en général un contrôleur maximalement permissif.

III.4. Synthèse de contrôleur par la théorie des régions

La synthèse de contrôleur par la théorie des régions (Ghaffari et *al*., 2003), basée sur le graphe de marquages accessibles, représente une méthode optimale pour assurer le fonctionnement désiré du SED. En effet, le contrôleur obtenu est le plus permissif possible , pour les spécifications de type états interdits avec intégration de la contrainte de vivacité et des transitions incontrôlables (Basile, 2006). Il est constitué des places de contrôle à ajouter au RdP du SED pour restreindre son graphe de marquages accessibles à l'ensemble d'états admissibles $\mathcal{M}_A$.

Soit M_0 l'état initial du RdP à contrôler, et soit $\mathcal{G}$ son graphe de marquages accessibles , pour déterminer l'ensemble des états admissibles $\mathcal{M}_A$, on suit les étapes suivantes :

- étape 1 : Détermination de l'ensemble des états interdits.
- étape 2 : Génération du graphe des marquages partiels.
- étape 3 : Détermination de l'ensemble des états dangereux.
- étape 4 : Recherche de l'ensemble des états admissibles.
- étape 5 : Recherche du comportement admissible du procédé.

Le graphe obtenu correspond au graphe de marquages admissibles, dénoté $\mathcal{G}_A$, qui traduit le comportement maximal permissif du SED. Les contraintes d'états interdits et de vivacité sont satisfaites par tous les états $\mathcal{G}_A$, et aucun état ne mène de façon incontrôlable à l'extérieur de $\mathcal{G}_A$. La commande par supervision doit restreindre le comportement du SED en inhibant toute transition contrôlable qui mène à un état bloquant ou interdit pour respecter les contraintes données.

III.4.1. Méthodologie de synthèse du contrôleur

Considérons le RdP du SED en boucle fermée augmenté d'une place de contrôle P_c, chaque état graphe de marquage admissible $\mathcal{G}_A$ doit rester accessible. Ceci implique que P_c doit vérifier la *condition d'accessibilté*, c'est-à-dire :

$$M(P_c) = M_0(P_c) + W(P_{c,}.)\vec{\Gamma}_{M_i} \geq 0 \quad \forall M_i \in \mathcal{G}_A \tag{III.13}$$

Où $\vec{\Gamma}_{M_i}$ est un chemin non orienté de G_A joignant M_0 et M_i

D'autre part, P_c doit satisfaire les équations de cycle correspondant aux cycles de $\mathcal{G}_A$, donc l'application de l'équation (III.13) le long d'un cycle non orienté quelconque γ conduit à la relation suivante, appelée *équation de cycle* :

$$\sum_{T_j \in T} W(P_c, T_j).\vec{\gamma}[t_j] = 0 \quad \forall \gamma \in \mathcal{S}_A \tag{III.14}$$

Où $\vec{\gamma}[t_j]$ designe la somme algébrique de toutes les occurrences de t_j dans γ et $\mathcal{S}_A$ est l'ensemble des cycles de base du graphe $\mathcal{G}_A$.

Quant aux événements à séparer pour obtenir G_A à partir du graphe de marquages initial, seules les transitions menant d'un état admissible à un état interdit doivent être inhibées.
Par conséquent, chaque nouvelle place P_C doit interdire une transition contrôlable qui mène à un état interdit. Cette interdiction se fait par la solution de l'équation ci-dessous, *appelée condition de séparation d'évènement* :

$$M_0(P_c) + W(P_c, \bullet)\vec{\Gamma}_{M_i} + W(P_c, t_j) < 0 \qquad \text{(III.15)}$$

Ainsi, pour chaque transition contrôlable qui mène à un état interdit, on résout le système composé des équations (III.13), (III.14) et (III.15) pour déterminer une nouvelle place de contrôle P_c. Le problème de commande par supervision peut être résolu de façon optimale par l'ajout d'un ensemble de places de contrôle au modèle du système en question si et seulement s'il existe une solution $W(P_c, \bullet)$, $M_0(P_c)$ vérifiant les équations précédentes pour chaque transition contrôlable qui mène à un état interdit.

Exemple III.2 :

Reprenons notre exemple de système manufacturier classique où les machines ainsi que la spécification sont modélisées par des RdP respectivement. Dans le fonctionnement en boucle fermée du procédé et de la spécification (figure III.3), une transition est franchissable si elle est validée par rapport au procédé et validée par rapport à la spécification. Les transitions communes au modèle du procédé et à la spécification sont f_1 , incontrôlable et d_2 contrôlable. L'exploration du graphe de marquage correspondant (figure III.4) permet de distinguer :

- l'ensemble des états dangereux est $\{M_3 , M_6\}$.
- l'ensemble des états admissibles est $\{M_0 , M_1 , M_2 , M_0 , M_5 , M_7\}$.

Pour ne pas atteindre le marquage M_3, il faut interdire la transition d_1 à partir de l'état M_2, ainsi que pour ne pas atteindre l'état M_6, il faut interdire la transition d_1 à partir de l'état M_5. L'ensemble des états admissibles permet d'obtenir le graphe de marquage admissible G_A suivant :

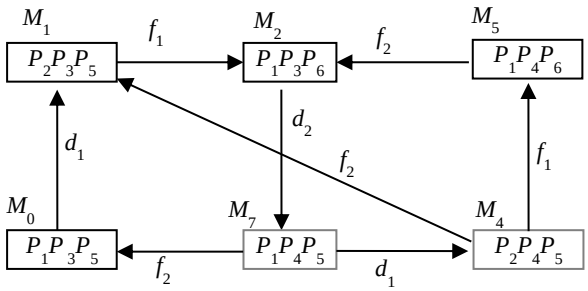


Figure III.9 – Graphe de marquage admissible $\mathcal{G}_A$

Pour la premiere transition à interdire à partir de M_2 on cherche la solution ($W(P_{c1}, \bullet)$, $M_0(P_{c1})$) :

- $M_3(P_{c1}) = M_0(P_{c1}) + W(P_{c1}, d_1) + W(P_{c1}, f_1) + W(P_{c1}, d_1) < 0$
- $M_0(P_{c1}) + 2W(P_{c1}, d_1) + W(P_{c1}, f_1) < 0$

La resolution du système permet d'obtenir la place de contrôle P_{C1} telle que :

- $W(P_{c1}, \bullet) = (-1, 0, 1, 0)$ et $M_0(P_{c1}) = 1$

Pour la deuxième transition à interdir à partir de M_5 , nous avons :

- $M_6(P_{c2}) = M_0(P_{c2}) + 2W(P_{c2}, d_1) + 2W(P_{c2}, f_1) + W(P_{c2}, f_2) + (P_{c2}, f_1) < 0$

La résolution du système permet d'obtenir la place de contrôle P_{C1} telle que :

- $W(P_{c2}, \bullet) = (-1, 0, 1, 0)$ et $M_0(P_{c2}) = 1$

Nous supprimons la place P_{c2} parce qu'elle est redondante et nous representons le modèle de RdP du SED controlé.

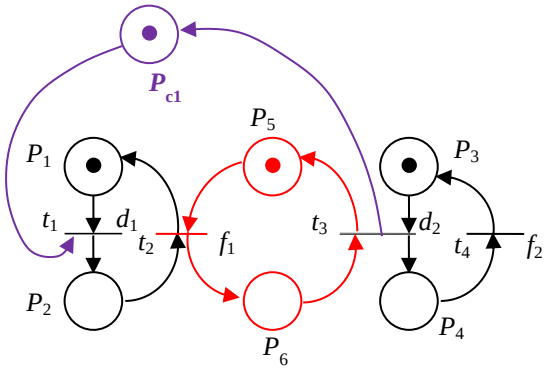


Figure III.10 – Le modèle de RdP contrôlé du système manufacturier classique

A l'origine, la théorie des régions a été développée pour la synthèse des RdP purs.

Définition 3.4.1. (**RdP pur**) Un RdP $R = (P, T, W^-, W^+)$ est pur lorsque $\forall P_i \in P, \forall t_j \in T, W^-(\bullet, t_j) \times W^+(\bullet, t_j) = 0$. C'est-à-dire qu'il n'existe pas de transition ayant une place d'entrée qui est aussi place de sortie.

❑

Cette théorie, permet de synthétiser le contrôleur à partir de deux algorithmes :

- le premier donne le graphe de comportement admissible G_A ,
- le deuxième (*condition d'accessibilité*, *équation de cycle*, *condition de séparation*) permet d'obtenir les places de contrôles à imposer au système pour respecter les contraintes données.

Le contrôleur obtenu est maximal permissif mais, le nombre d'inéquations en nombres entiers engendrées est important et il n'y a pas de méthodes efficaces pour résoudre de tels systèmes (Kattan, 2004).

III.5. Synthèse de contrôleur par retour d'état

Dans les méthodes de synthèse de contrôleurs par les invariants et par la théorie des régions, le contrôleur a été déterminé par l'ajout de places de contrôle, qui modifient la structure du RdP du SED en boucle fermée. Cependant, il est possible d'utiliser une autre méthode qui, au lieu d'ajouter des places de contrôle, définit une commande qui peut autoriser ou interdire le franchissement de chaque transition contrôlable. Cette méthode, appelée synthèse de contrôleur par retour d'état (Holloway & Krogh, 1990) est toujours optimale quand il s'agit de contrôler les SED modélisés par des graphes d'événements cycliques et saufs. La résolution du problème d'états interdits est basée sur la structure du graphe d'événements cycliques et saufs, qui constitue une classe particulière des RdP, où il n'y a pas de possibilité de conflit, de choix, ni de cumul.

III.5.1. Principe du retour d'état

La méthode stipule qu'une transition t_j peut être franchie seulement si elle est validée (c'est-à-dire, il y a des jetons dans toutes les places d'entrée de t_j) et s'il y a des jetons fictifs dans les places de contrôle. Le contrôle d'une transition peut être vu comme un commutateur binaire associé à la transition, c'est-à-dire qu'il laisse ou ne laisse pas passer les jetons des ports d'entrée aux ports de sortie (figure III.11).

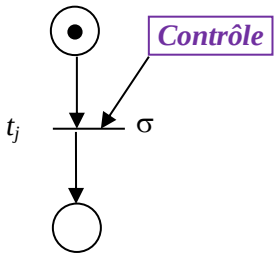


Figure III.11 – Transition contrôlée par retour d'état

L'objectif de la commande par supervision étant d'empêcher des états interdits, par le contrôle des transitions contrôlables à partir de l'ensemble des états autorisés critiques donnant accès aux états interdits, nous reprenons l'idée de la section II.3.2 du chapitre 2 pour l'appliquer dans le cadre des RdP.

Définition III.5.1. (Etat admissible) Soit $\mathcal{M}_A$ l'ensemble des états admissibles ou au autorisés et $\mathcal{M}_F$ l'ensemble des états faiblement interdits. L'ensemble des états admissibles critiques $\mathcal{M}_{AC}$ pour la transition contrôlable $t_j \in T_C$ est défini comme suit :

$$\mathcal{M}_{AC,\, t_j} = \{M_k \in \mathcal{M}_A \mid M_k\,[t_j\rangle \in \mathcal{M}_F\}$$

❑

L'exploration du graphe de marquage de notre exemple classique permet d'identifier l'ensemble des états admissibles critiques comme présenté dans la figure III.12.

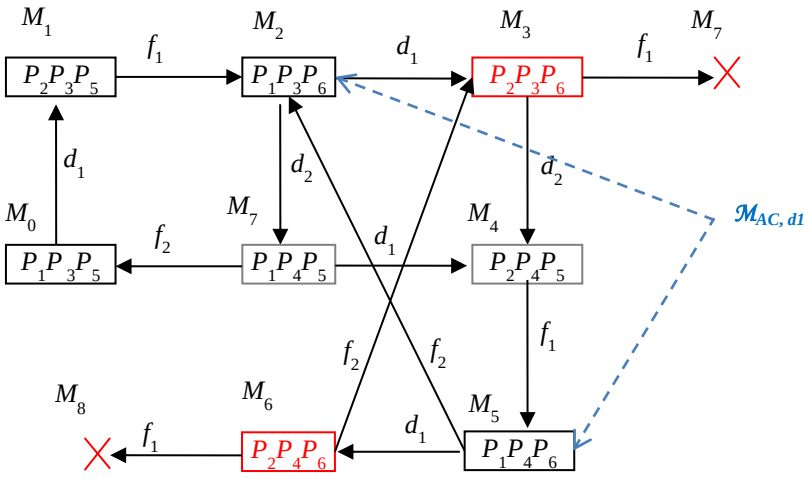


Figure III.12 – Identification des états admissibles critiques

Pour la transition contrôlable t_1 (associée à l'événement d_1) nous avons :

$$\mathcal{M}_{AC,\, d1} = \{P_1P_3P_6\,,\, P_1P_4P_6\}$$

Ainsi, pour cette transition il faut créer un contrôle U_{t1} , tel que le franchissement de la transition t_1 dans l'ensemble des états critiques $\mathcal{M}_{AC,\,d1}$ soit interdit.

III.5.2. Contrôle des transitions

La condition de franchissement de t_1 exige que cette transition ne soit franchie à partir d'aucun marquage de cet ensemble (Dideban et Alla, 2006). Cette condition peut être calculée comme ci-dessous :

$$\textit{Condition de franchissement } t_1 = \left(M(P_1)\,M(P_3)\,M(P_6) + M(P_1)\,M(P_4)\,M(P_6)\right)' \qquad \text{(III.16)}$$

Où le symbole « ' » signifie le complément logique.

Nous pouvons interdire le franchissement de la transition seulement à partir de l'ensemble des états critiques. Cette condition sera d'autant plus compliquée que le nombre d'états critiques pour chaque transition augmente.

Soit $\mathcal{M}_{AC,\,t_j}$ l'ensemble des états admissibles critiques pour la transition t_j.. Le contrôle $Ut_j(M_k) \rightarrow \{0,1\}$ est défini comme ci-dessous :

$$Ut_j\left(M_k\right) = \begin{cases} 0 & M_k \in \mathcal{M}_{AC}, t_j \\ 1 & si\ non \end{cases} \qquad \text{(III.17)}$$

En considérant des RdP saufs (Dideban, 2004), on peut calculer le contrôle $Ut_j(M_k)$ à l'aide de l'expression logique con_M_k :

$$con\_M_k = M_k(P_1)\,M_k(P_2)\ \ldots\ m_{jr}\ \ldots\ M_k(P_n)\,m_{jn} = \bigcap_{i=1}^{n} M_k(P_i) \qquad \text{(III.18)}$$

La variable Booléenne con_M_k est égale à 1 lorsque l'état actuel du système est l'état M_k. Le complémentaire de cette variable pour la transition t_j , donne le contrôle $Ut_j(M_k)$ pour cette transition. Ainsi, par l'utilisation du contrôle $Ut_1(M_k)$, on peut interdire le franchissement de t_1 à partir de l'ensemble $\mathcal{M}_{AC,\,d1} = \{P_1P_3P_6\ ,\ P_1P_4P_6\}$. La structure du RdP contrôlé par cette méthode est présentée à la figure III.13.

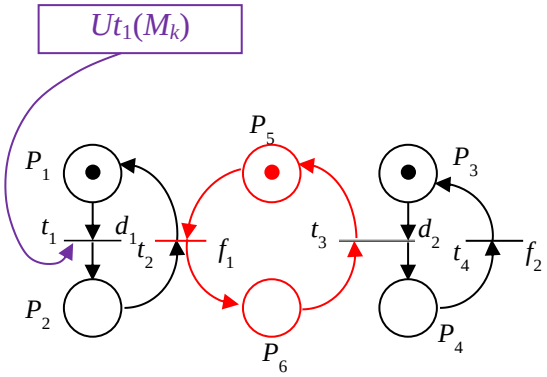


Figure III.13 – Contrôle empêchant l'accessibilité des états interdits

Le RdP du SED en boucle fermée est contrôlé par l'inhibition de l'événement contrôlable d_1 , défini en munissant la transition t_1 par la place de contrôle et non spécifiquement.

Remarque III.5.1. Cette méthode synthétise un contrôleur maximal permissif qui garantit qu'aucun état interdit ne sera jamais atteint. De nombreuses approches utilisent cette méthode pour empêcher des états interdits, en général sous forme de CGEM (Giua et *al*., 1992).

❑

Le problème principal de cette méthode est qu'en général le calcul du contrôle est complexe et long en temps réel. Pour tout RdP sauf nous pouvons construire le graphe de marquages pour déterminer les états critiques (Dideban & Alla, 2006). Mais, dans le cas des graphes d'événements, la synthèse des contrôleurs est exécutée, sans avoir besoin de générer ou analyser de manière exhaustive le graphe de marquage (Holloway &Krogh, 1990), en utilisant la structure du RdP de celui-ci. Cependant, elle n'est pas applicable en général.

Une extension (Holloway et *al*., 1996) a été ultérieurement apportée à la méthode de contrôle par retour d'état afin d'accroitre son applicabilité à une classe très large de RdP, y compris les RdP non-saufs. Le problème majeur de cette approche est la complexité des expressions logiques associées aux transitions de contrôle, qui reste souvent très grande en dépit de toutes les techniques de simplification développées (Dideban & Alla, 2006 ; Vasiliu et *al*., 2009). Par ailleurs, une approche hybride de construction de contrôleur, applicable sur tous les types de RdP, qui permet de calculer les conditions de franchissement des transitions contrôlables a été proposée dans (Uzam & Wonham, 2006). Mais, la taille du RdP augmente considérablement. Enfin, les restrictions apportées à la structure des RdP font que ceux-ci

ne peuvent par exemple pas modéliser les conflits, de même il ne doit y avoir qu'un seul jeton par cycle dans le RdP. Cela limite considérablement la portée d'une telle approche.

III.6. Problème de synthèse de contrôleurs par RdP et objectif de la thèse.

III.6.1. Problème de synthèse de contrôleurs par RdP

Les RdP sont des formalismes qui ont permis de réaliser de nombreux travaux avec des résultats intéressants et des méthodes adaptées, selon les concepts, à la synthèse des contrôleurs. Mais, aucun résultat général n'a encore été développé jusqu'à lors. La méthode de synthèse de contrôleur par retour d'état (Krogh & Holloway, 1990) est limitative du point de vue de la structure de la classe du RdP utilisée. La théorie des régions est très peu applicable pour des systèmes complexes à cause de la complexité du calcul des places de contrôle et de la nécessite d'énumérer tous les états admissibles du graphe de marquage. Un travail visant à simplifier la théorie des régions par réduction du nombre d'équations du système linéaire a été proposé dans (Rezig, 2016). Il est basé sur des concepts de coupes minimales et de marquages canoniques. Ce qui a permis de développer d'une nouvelle approche sans générer le graphe de marquage où, les spécifications de contrôle sont basées sur des séquences de transitions et non pas sur des états. Pour des spécifications basées sur les états, la méthode de synthèse de contrôleurs par les invariants est remarquable. C'est une méthode fondamentalement prometteuse, même si elle ne garantit pas une solution optimale dans le cas général et surtout, si l'incontrôlabilité des événements est considérée. Si le contrôleur est calculé de manière classique, alors des inclusions (totales ou partielles) de support entre les états interdits et les états autorisés peuvent entraîner l'interdiction d'un grand nombre de comportements désirables du SED. La méthode des invariants permet de déterminer les places de contrôle à ajouter, afin de satisfaire les spécifications d'états souvent exprimées sous forme de contraintes linéaires de type GEMC. Si le nombre de contraintes linéaires est très grand les méthodes de simplification des contraintes peuvent être utilisées (Dideban, 2007). Afin de garantir l'optimalité de la méthode en présence d'événements incontrôlables, l'approche de Vasiliu (2012) nécessite la construction du graphe de marquage pour identifier les ensembles d'états admissibles. Nous pouvons constater que le problème classique de l'explosion combinatoire du nombre d'états se retrouve dans le graphe de marquage, qui est un automate. Ce qui pose encore des difficultés d'implantation (Uzam , 2010). Généralement certains auteurs les auteurs utilise les RdP saufs (Iordache et *al*., 2001 ;

Dideban & Alla, 2008; Iordache et *al.*, 2013). Malgré tout, la méthode des invariants reste une méthode simple et efficace, capable de synthétiser un contrôleur maximal et permissif lorsque l'ensemble des contraintes adéquates lui est fourni (Vasiliu, 2012). Nous pouvons qualifier ces méthodes comme étant partiellement structurelles parce qu'elles nécessitent la génération et l'exploration du graphe de marquage qui peut être fastidieuse pour les SED complexe.

III.6.2. Objectif de la thèse

Notre objectif principal est de trouver une approche complètement structurelle qui permette de déterminer à partir de la structure des RdP du SED en boucle fermée, sans génération du graphe de marquages, les contraintes linéaires adéquates pour la synthèse d'un contrôleur maximal permissif par la méthode des invariants. La solution de contrôle que nous souhaiterions obtenir devra prendre en compte l'existence d'évènements incontrôlables, qui ne peuvent pas être inhibés par le contrôleur : ce qui serait très avantageux pour des SED réels comme les systèmes de production par exemple. C'est pourquoi, nous projetons d'établir de manière analogique, la condition de contrôlabilité à partir de l'analyse de la synchronisation structurelle des RdP du procédé et des spécifications via les transitions associées aux événements incontrôlables. La condition de contrôlabilité jouera un rôle important dans la commande par supervision car, le contrôleur observe l'évolution du procédé et peut à tout moment interdire l'exécution des événements contrôlables. Nous traduirons la condition de contrôlabilté en des contraintes admissibles sur les marquages des places du RdP ; ce qui nous permettra de faire un lien biunivoque entre la théorie de la commande par supervision et la synthèse de contrôleurs basée sur les invariants de place. Sous l'hypothèse que la méthode des invariants est adaptés aux RdP ordinaires et/ou généralisés, nous estimons qu'une telle approche, donnera lieu à de fortes applications pratiques.

Chapitre IV

Synthèse structurelle de contrôleur par lien biunivoque entre la théorie de commande par supervision et la méthode des invariants

Dans ce chapitre, nous visons à établir un lien structurel et biunivoque entre la théorie de commande par supervision et la méthode des invariants de place ; afin de disposer, dans le cadre des RdP, d'une méthode synthèse de contrôleurs complètement structurelle et optimale en présence d'incontrôlabilité. Cette méthode ne nécessite pas la construction du graphe de marquage accessible. Elle se situe dans la phase de construction du RdP du SED en boucle fermée par produit synchrone, de manière à permettre la détermination des contraintes admissibles pour la méthode des invariants.

Chapitre IV

Synthèse structurelle de contrôleur par lien biunivoque entre la théorie de commande par supervision et la méthode des invariants

IV.1. Introduction

L'utilisation des RdP dans le cadre de la théorie de commande par supervision montre qu'à partir du graphe de marquage, une solution de contrôle maximal permissif peut être obtenue. En effet, lorsque le langage généré par le graphe de marquage est contrôlable, le RdP du fonctionnement du SED en boucle fermée constitue le contrôleur. Mais, s'il n'est pas contrôlable, il est nécessaire de déterminer et supprimer les états interdits et bloquants (Li & Zhou, 2009). Or, la structure du contrôleur est monolithique (Guia, 2013). Ce qui limite la possibilité d'implantation de la solution, car la suppression des états interdits (afin d'obtenir un graphe de marquage admissible et non bloquant) n'est pas adaptée aux RdP. D'où, la nécessité de synthétiser un contrôleur qui modifie la structure du RdP du SED en boucle fermée, pour garantir le respect des spécifications en empêchant l'accessibilité aux états interdits. Malheureusement, dans les méthodes représentatives de synthèse de contrôleur par les RdP (David & Alla, 2010), le graphe de marquage est construit pour analyser et déterminer l'ensemble des états interdits. En comparant ces méthodes à la théorie de commande par supervision, nous constatons qu'elles sont partiellement structurelles et n'abordent pas le problème de contrôlabilité comme dans (Kumar & Holloway, 1996). Cependant, la méthode des invariants de place est la plus simple et efficace pour obtenir une solution optimale en présence d'incontrôlabilité, si les contraintes linaires sont admissibles (Vasiliu & Alla, 2011). C'est pourquoi, nous devons déterminer ces contraintes admissibles à partir de la structure du RdP du SED en boucle fermée, et établir un lien biunivoque entre la théorie de commande par supervision et la méthode des invariants de place.

IV.2. Construction du RdP de fonctionnement du SED en boucle fermée

IV.2.1. Produit synchrone des RdP de commande par supervision

Un SED réel est souvent constitué de sous-systèmes, en interaction plus ou moins forte. Le modèle global du SED est obtenu par le produit synchrone des modèles de ces sous-systèmes (Takai & Ushio, 2003). Le produit synchrone est le premier pas de l'algorithme de Kumar (1991) en ce qui concerne la théorie commande par supervision basée sur les automates. Mais, c'est une opération particulièrement adaptée aux RdP, car elle se réduit simplement à la synchronisation structurelle (voir chapitre 1). Par conséquent, nous pouvons à partir des RdP du procédé et de la spécification, construire le RdP de fonctionnement du SED en boucle fermée, dont la complexité dépend uniquement de la structure du RdP.

La puissance de modélisation d'un SED étant strictement reliée aux séquences d'événements qu'il peut générer, la classe des RdP synchronisés (RdPS) est parfaitement adaptée.

Nous rappelons qu'un RdPS est un 3-uplet $N = \langle R, E, f\rangle$ où $R = \langle P, T, W, M_0\rangle$ est un RdP, E un alphabet et $f : T \rightarrow E$ est une fonction d'étiquetage qui associe un événement de E à une transition de T. Sans perte de généralité, nous supposons que chaque événement est associé à une transition t_j au plus.

Dans le cadre de notre travail, un procédé à contrôler sera décrit par le RdPS, $N_p = \langle R_p, E_p, f_p\rangle$ avec $R_p = \langle P_p, T_p, W_p, M_{p0}\rangle$, tandis qu'une spécification de son fonctionnement sera décrite par le RdPS, $N_s = \langle R_s, E_s, f_s\rangle$ avec $R_s = \langle P_s, T_s, W_s, M_{s0}\rangle$. Le modèle RdPS du SED en boucle fermée N, dont l'espace des états est noté $\mathcal{A}(N, M_0)$, est donné par le produit synchrone des modèles N_p et N_s , d'espaces d'états $\mathcal{A}(N_p, M_{p0})$ et $\mathcal{A}(N_s, M_{s0})$ respectivement.

Définition IV.2.1. (Produit synchrone) Le produit synchrone des RdPS $N_p = \langle R_p, E_p, f_p\rangle$ avec $R_p = \langle P_p, T_p, W_p, M_{p0}\rangle$ et $N_s = \langle R_s, E_s, f_s\rangle$ avec $R_s = \langle P_s, T_s, W_s, M_{s0}\rangle$ est un autre RdPS , $N = N_p \,||N_s = \langle R, E, f\rangle$ défini sur l'alphabet $E = E_p \cup E_s$, avec $R = \langle P, T, W, M_0\rangle$ tel que :

- $P := P_p \cup P_s$;
- $T := \{(t_p, t_s) \in T_p \text{ x } T_s \mid f_p(t_p) = f_s(t_s), f(t_p, t_s) := f_p(t_p) = f_s(t_s)\}$;
- $W := \{(\bullet, (t_p, t_s)) \in P \text{ x } T \mid (\bullet, t_p) \in W_p \text{ ou } (\bullet, t_s) \in W_s\} \cup \{(\bullet, (t_p, t_s)) \in P \text{ x } T \mid (t_p, \bullet) \in W_p \text{ ou } (t_s, \bullet) \in W_s\}$
- $M_0(p_i) = \{M_{p0}(p_i) \text{ si } p_i \in P_p \text{ ou } M_{s0}(p_i) \text{ si } p_i \in P_s\}$

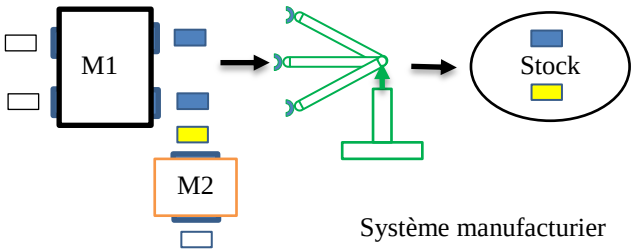
❐

La dynamique du RdPS du SED en boucle fermée, $N = N_p||N_s$, est appelée fonctionnement désirée du SED en boucle fermée.

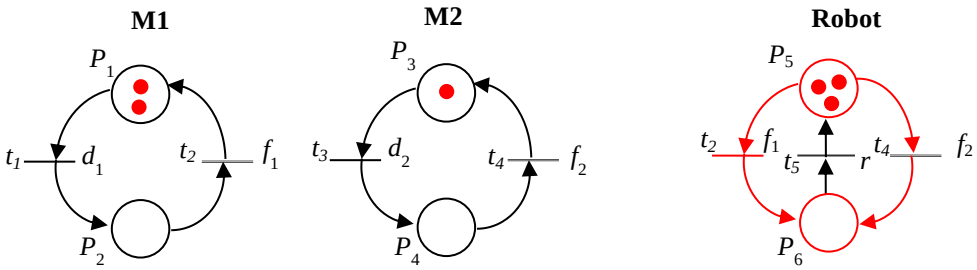
Remarque IV.1. : Il est fréquent de considérer un procédé composé de m RdPS, N_{P1} , ..., N_{Pm} qui travaillent simultanément. Le RdPS global du procédé est $N_p = N_{p1}||\ldots..||N_{pm}$ et $E_p = E_{p1} \cup \ldots \cup E_{pm}$.

Exemple IV.1. (Vasiliu, 2012)

Soit un système manufacturier constitué de deux machines, M1 et M2, qui travaillent en parallèle pour produire deux variétés du même type de pièces. Nous supposons que le fonctionnement des machines est idéal (elles ne peuvent pas tomber en panne). La machine M2 a une capacité d'usinage d'une seule pièce à la fois, alors que la capacité d'usinage de la machine M1 est de deux pièces à la fois. Les pièces fabriquées par les deux machines sont recueillies dans un stock commun. Des robots assurent le transfert des pièces vers le stock.
La ***spécification*** du système stipule qu'il y a trois robots dédiés au transport des pièces entre les machines et le stock. Les débuts des opérations de fabrication (événements d_i) sont les seuls événements contrôlables du système. Les fins d'usinage (événements f_i) et le retour de chaque robot à l'état «disponible» (événement r) sont incontrôlables.



Système manufacturier



RdPS du Procédé, N_p RdPS Spécification, N_s

Figure IV.1 – Modèles du système manufacturier traité dans (Vasiliu, 2012)

Le produit synchrone des RdPS du procédé N_p constitué des deux machines (M1 et M2) et N_s celui de la spécification donnée par les 3 robots donne le RdPS du SED en boucle fermée (figure 4.2) défini sur l'alphabet $E = \{d_1, f_1, d_2, f_2, r\}$.

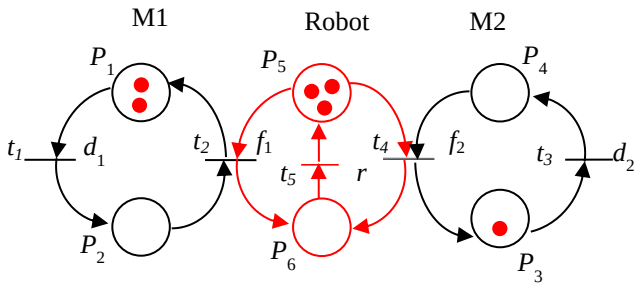


Figure IV.2 – Modèle RdPS du SED en boucle fermée obtenu par produit synchrone

▲

L'exemple nous montre que le produit synchrone réalise la synchronisation structurelle. Une paire de transitions (t_p , t_s) associée à événement commun $E_p \cap E_s$ dans les RdPS du procédé N_p et de la spécification N_s distincts, est remplacée avec une seule transition $t_j = (t_p , t_s)$, $t_j \in T$ dans $N = N_p || N_s$. Nous disons que N_p et N_s sont synchronisés sur l'ensemble d'événements communs $E_p \cap E_s$, tandis qu'entre deux synchronisations successives, les RdPS N_p et N_s exécutent les événements privés appartenant à $E \setminus (E_p \cap E_s)$. En particulier, la transition $t_j = (t_p , t_s)$ est dite transition synchrone. Elle a comme places d'"entrée" l'union respective de celles de la paire de transition (t_p , t_s), que nous désignons par : ${}^\bullet t_j = \{{}^\bullet t_p \cup {}^\bullet t_s\}$, avec

- ${}^\bullet t_p = \{p_i \in P_p \mid W(\bullet, t_j) > 0\}$, l'ensemble des places d'entrée de la transition t_j appartenant à N_p
- ${}^\bullet t_s = \{p_i \in P_s \mid W(\bullet, t_j) > 0\}$, l'ensemble des places d'entrée de la transition t_j appartenant à N_s

Nous tenons à rappeler que $W(\bullet, t_j)$ définit le nombre minimum de marques requis dans la place p_i pour valider la transition t_j. Par ailleurs, si nous considérons les langages générés par les RdPS, nous pouvons facilement (voir section I.4.5, Chapitre 1) montrer que le langage généré par le RdPS du SED en boucle fermée, satisfait $L(N_p || N_s) = L(N_p) \cap L(N_s)$ lorsque N_p et N_s sont déterministes et définis sur l'alphabet E (Hopcroft *et al.*, 2007).

Définition IV.2.2. (RdPS déterministe) Un RdPS est déterministe si pour tout $M_k \in \mathcal{A}(N, M_0)$ et pour des paires de transitions distinctes (t_p , t_s) telles que $M_k [t_p \rangle$ et $M_k [t_s \rangle$, il maintient $f_p(t_p) \neq f_s(t_s)$. Et pour tous les états accessibles, il n'y a pas de conflits entre les transitions qui partagent le même événement.

❒

IV.2.2. Condition de franchissement des transitions synchrones du RdPS

Le franchissement d'une transition t_j entraine les modifications du marquage d'un état et permet d'analyser la dynamique du RdPS du SED en boucle fermée.

a) Franchissement d'une transition dans un RdPS

Une transition t_j du RdPS est validée par un état $M_k \in \mathbb{N}^n$ si $M_k \geq W(\bullet, t_j)$. Si p_i est une place d'entrée de la transition t_j alors $W(\bullet, t_j)$ est le poids de l'arc (p_i, t_j). Soit $M_k(p_i)$ le marquage d'une place d'entrée p_i. La condition sur le marquage de la place d'entrée p_i est donnée par :

$$M_k(p_i) \geq W(\bullet, t_j) \qquad \text{(IV.1)}$$

De manière formelle, une transition t_j est dite validée quand la condition sur le marquage de la place d'entrée est satisfaite. C'est-à-dire,

$$\text{si } \langle M_k(p_i) \geq W(\bullet, t_j) \rangle \text{ alors } \langle t_j \text{ est validée} \rangle$$

La condition de franchissement de la transition t_j peut être représentée par une matrice nx1, définie par $u_j = W(\bullet, t_j)$. Dans le cas général, nous désignons $\mathcal{U} = (u_j / j = 1, 2, \ldots m)$, la matrice de condition de franchissement des transitions. Précisons que dans le cadre RdPS, chaque transition est associée un événement. Par conséquent, une transition t_j validée n'est pas forcement franchissable. Elle deviendra franchissable quand l'événement associé à la transition se produit.

b) Franchissement d'une transition synchrone dans un RdPS

- **Cas où $Card(^\bullet t_j) = 2$**

Considérons le RdPS du système manufacturier en boucle fermée donné à la figure 4.1. Nous avons deux transitions synchrones t_2 et t_4 avec $^\bullet t_2 = \{P_2 , P_5\}$ et $^\bullet t_4 = \{P_5 , P_4\}$. Examinons le cas de la transition synchrone t_2 associée à l'événement f_1, la matrice de franchissement de

transition $u_2 = [0\ 1\ 0\ 0\ 1\ 0]^T$, donne la condition sur les marquages des places d'entrée P_2 et P_5 pour que cette transition soit validée :

$$\text{si } \langle M_k(P_2) \geq 1 \rangle \ \& \ \langle M_k(P_5) \geq 1 \rangle \ \text{ alors } \langle t_2 \text{ est validée} \rangle$$

La transition t_2 validée ne peut être franchie que si l'événement f_1 se produit. Le même raisonnement peut être effectué avec la transition t_4 associée à l'événement f_2.

- **Cas où $Card(^\bullet t_j) > 2$**

Dans le cas où nombre n de places d'entrée de la transition t_j est $n \geq 2$, la conjonction des marquages de ces places détermine la condition de franchissement de la transition :

$$\text{si } \langle M_k(p_1) \geq W^-(p_1, t_j) \rangle \ \& \ \langle M_k(p_2) \geq W^-(p_2, t_j) \rangle \ \& \ \ldots \ \langle M_k(p_n) \geq W^-(p_n, t_j) \rangle \ \text{ alors } \langle t_j \text{ est validée} \rangle$$

La transition synchrone t_j ne sera franchie que si l'événement associé se produit.

Remarque IV.2. Dans le contexte particulier de la commande par supervision, un tel franchissement peut être inadmissible si l'événement associé à la transition synchrone t_j est incontrôlable, alors que la condition sur les marquages des places viole une spécification.

IV.2.3. Conséquence du franchissement des transitions synchrones

Considérons le RdPS du comportement désiré $N = N_P || N_S$. La condition de franchissement d'une transition synchrone t_j suppose que les deux RdPS, N_p et N_s soient dans des états où la transition t_j est validée. Elle sera franchie dès que l'événement qui lui est associé se produit. Or, dans un SED réel, certains événements sont contrôlables (E_c) alors que les autres sont incontrôlables (E_c) et, donc ne peuvent donc être empêchés. Ce qui signifie que, l'ensemble des évènements E du RdPS $N = N_P || N_S$ peut être partitionné comme suit

$$E = E_c \cup E_{uc} \text{ (avec } E_c \cap E_{uc} = \emptyset) \qquad \text{(IV.2)}$$

Nous savons que $f : T \rightarrow E$; nous supposons que deux transitions $N = N_p || N_s$ ne sont pas associées au même événement. L'ensemble des transitions T peut être partitionné comme :

$$T = T_c \cup T_{uc} \qquad \text{(IV.3)}$$

Où T_c est l'ensemble des transitions contrôlables et T_{uc} l'ensemble des transitions incontrôlables.

Si l'événement associé à la transition synchrone t_j est incontrôlable, alors il n'est pas possible d'empêcher son franchissement lorsque les conditions sur les marquages des places d'entrée ne respectent pas la spécification donnée. La liberté d'une transition synchrone incontrolable à etre franchie est limitée uniquement par la structure et l'état du RdPS du SED en boucle fermée. Par exemple, la transition est bloquée par insuffisance ou manque de jetons dans les places d'entrée du RdPS de la spécification N_s alors qu'elle est validée par le marquage des places d'entrée du RdPS du procédé N_p. Par conséquent, $N = N_p||N_s$ est susceptible de générer des comportements inadmissibles, à savoir :

- la génération d'états interdits supplémentaires,
- le problème de contrôlabilité de ces états

Considérons par similitude avec les propriétés d'automates que la notion de contrôlabilité est liée aux événements E. Alors, le procédé décrit N_p sur l'alphabet E_p a comme langage clos $L(N_P)$ et langage marqué $L_m(N_P)$. S'il est non bloquant alors, $\overline{L_m}(N_p) = L(N_p)$. De même, la spécification représentée par le RdPS non bloquant N_S définit un langage clos et un langage marqué tel que : $\overline{L_m}(N_s) = L(N_s)$. Une telle spécification définit un ensemble de séquences admissibles dans le langage du fonctionnement désiré $L(N) = L(N_p) \cap L(N_s)$. Ainsi, le problème de contrôlabilité revient à déterminer l'admissibilité d'un certain ensemble d'états accessibles à partir de l'état initial dans $N = N_p||N_S$. Par conséquent, le problème de commande par supervision peut se réduire à un problème de détermination structurelle des états interdits source d'incontrôlabilité dans $N = N_p||N_S$, et de synthèse d'un contrôleur qui doit empêcher d'atteindre un tel état. Les états interdits source d'incontrôlabilité sont des états à partir desquels l'événement synchrone incontrôlable se produit dans le RdPS du procédé N_p.

IV.3. Contrôlabilité du RdPS du SED en boucle fermée

Une fois que le RdPS du SED en boucle fermée est construit, nous avons besoin de vérifier s'il est non bloquant et contrôlable. La synchronisation structurelle via une transition incontrôlable est une source potentielle d'incontrôlabilité, qui peut conduire $N = N_p||N_S$ à être bloqué ou incontrôlable. La vérification de la contrôlabilité peut être faite par construction du graphe de marquage (Seatzu et *al.*, 2012) ou par l'analyse structurelle de la condition de franchissement des transitions synchrones incontrôlables. Pour mettre en évidence le

problème de contrôlabilité (Giua & DiCesare, 1994), nous considérons le système manufacturier de l'exemple IV.1.

IV.3.1. Vérification de la contrôlabilité à partir du graphe de marquage

Considérons le RdPS du système manufacturier en boucle fermée donné à la figure IV.2 défini sur $E = E_c \cup E_{uc}$, avec $E_c = \{d_1, d_2\}$ et $E_{uc} = \{f_1, f_2, r\}$. La figure IV.3 correspond au graphe de marquage accessible, où l'état du SED est symbolisé par une association entre son support et son marquage (Définition III.2.3, chapitre 3)

Notation IV.3.1. Soit $P = \{p_1, p_2, ..., p_n\}$ l'ensemble des places d'un RdPS, $N = N_P || N_S$. Soit $M_k = [M_k(p_1), M_k(p_2), \ ... \ , M_k(p_n)]^T$, $M_k(p_i) \in \mathbb{N}$, un état quelconque de N. L'état M_k peut être représenté comme suit : $M_k = \{p_i^{M(Pi)} \mid i = 1, n \text{ avec } M_k(p_i) \geq 1\}$.

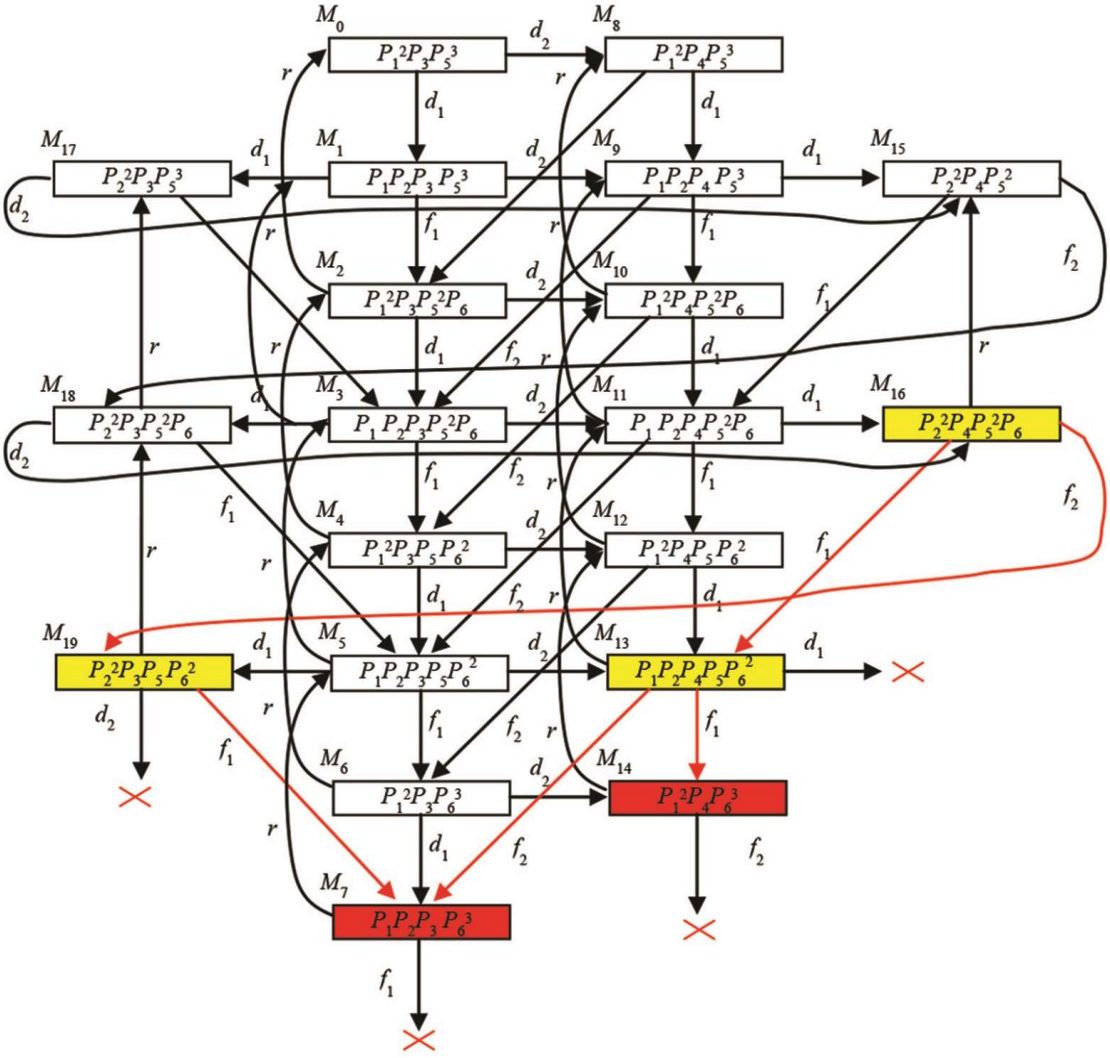


Figure IV.3 – Graphe de marquage du RdPS de l'exemple IV.1

Une exploration du graphe de marquage permet d'identifier les états interdits par la spécification, à savoir :

- $M_7 = P_1P_2P_3\ P_6^3$, la machine M1 est en cours d'usinage, alors qu'il n'y a pas de robot disponible pour prendre en charge la pièce usinée.
- $M_{14} = P_1^2P_4P_6^3$, la machine M2 est en cours d'usinage, alors qu'il n'y a pas de robot disponible pour prendre en charge la pièce usinée.

Puisque, la fin de l'opération de fabrication, ainsi que le retour du robot, sont des événements incontrôlables, il est alors impossible de savoir s'il y aura un robot disponible pour prendre en charge la pièce usinée au moment où elle est éjectée par la machine.

Les états M_7 et M_{14} peuvent être représentés par les marquages de forme $M_k = (M_{p,k}, M_{s,k})$ où $M_{p,k} \in \mathcal{A}(N_p, M_{p0})$ et $M_{s,k} \in \mathcal{A}(N_s, M_{s0})$. Nous avons :

- $M_7 = [1\ 1\ 1\ 0\ 0\ 3]^T$ où $M_{p,7} = [1\ 1\ 1\ 0]^T$ et $M_{s,7} = [0\ 3]^T$
- $M_{14} = [2\ 0\ 0\ 1\ 0\ 3]^T$ où $M_{s,14} = [2\ 0\ 0\ 1]^T$ et $M_{s,14} = [0\ 3]^T$

Pour ces deux états et pour les transitions synchrones t_2 et t_4 associées aux événements incontrôlables f_1 et f_2 :

- $M_{p,7}\ [f_1\rangle$ est autorisé , car t_2 est validée par $M_7(P_2) = 1$ alors que, $M_{s,7}[f_1\rangle$ n'est pas défini, puisque t_2 est bloquée par $M_7\ (P_5) = 0$
- $M_{P,14}[f_1\rangle$ (respectivement $M_{p,14}[f_2\rangle$) est autorisé, t_4 est validé par $M_{14}(P_4) = 1$ alors que, $M_{s,14}[f_1\rangle$ (respectivement $M_{s,14}[f_2\rangle$) n'est pas défini, car t_4 est bloquée par $M_{14}\ (P_5) = 0$.

Définition IV.3.1. (États interdits par la spécification). Formellement les états interdits par la spécification dans le RdPS du SED en boucle fermée, $N=N_P||N_S$, sont définis par :

$\{M_k = (M_{p,k}, M_{s,k}) \in \mathcal{A}(N, M_0) \mid \exists\ t_j = (t_p, t_s) \in T_{uc}$ tel que, $M_{p,k}[t_p\rangle$ est autorisée et $M_{s,k}[t_s\rangle$ n'est pas défini$\}$.

❒

En raison de la synchronisation structurelle via les transitions associées aux événements incontrôlables communs f_1 et f_2, les états sources potentielles d'incontrôlabilité sont évidents. Ces états sont identifiables lorsqu'on essaie de remonter les arcs du graphe de marquage à partir des états interdits par la spécification (Kumar, 1991).

Pour cet exemple, nous trouvons trois états, qui peuvent conduire le système vers les états interdits par la spécification ($M_7 = P_1P_2P_3\ P_6^3$ et $M_{14} = P_1^2P_4P_6^3$) sous l'occurrence des événements incontrôlables commun f_1 et f_2, à savoir :

- $M_{19} = P_2^2P_3P_5\ P_6^2$, la machine M1 est en train de travailler sur deux pièces, alors qu'un seul robot est disponible.
- $M_{13} = P_1P_2P_4P_5P_6^{\ 2}$, une pièce est en cours de fabrication sur chaque machine, alors qu'il y a un seul robot disponible
- $M_{16} = P_2^2P_4P_5^2\ P_6$, trois pièces sont en cours d'usinage, alors que juste deux robots sont disponibles

Pour ces états sources d'incontrôlabilité nous avons :

- $M_{19}\ [f_1\rangle$ n'est pas autorisée dans $N=N_p||N_s$ car, $M_{19}(P_2) + M_{19}(P_4) = 2$ alors que $M_{19}(P_5) =1$. Mais, $M_{p,19}\ [f_1\rangle$ est autorisée, car t_2 est validée par $M_{19}\ (P_2) =2$
- $M_{13}\ [f_1\rangle$ (respectivement $M_{13}\ [f_2\rangle$) n'est pas autorisée dans $N=N_P||N_S$ car, $M_{13}(P_2) + M_{13}(P_4) = 2$ alors que $M_{13}(P_5)=1$. Mais, $M_{p,13}\ [f_1\rangle$ (respectivement $M_{13}\ [f_2\rangle$) est autorisé, car t_2 est validée par $M_{13}\ (P_2)=1$ et t_4 est validée par $M_{13}(P_4) =1$
- $M_{16}\ [f_1\rangle$ (respectivement $M_{16}[f_2\rangle$) n'est pas autorisée dans $N=N_p||N_s$ car, $M_{16}(P_2) + M_{16}(P_4) = 3$ alors que $M_{16}(P_5) =2$. Mais, $M_{p,16}[f_1\rangle$ (respectivement $M_{16}[f_2\rangle$) est autorisé, car t_2 est validée par $M_{16}\ (P_2) =2$ et t_4 est validée par $M_{16}(P_4) =1$

Par conséquent, un état M_k est incontrôlable s'il existe une transition incontrôlable qui n'est pas validée dans $N=N_P||N_S$, tandis qu'elle est validée dans l'état $M_{p,K}$ du procédé N_p isolé. Nous considérons qu'un tel état est un état interdit par la synchronisation incontrolable.

Définition IV.3.2. (Etats interdits par la synchronisation incontrôlable) Un état interdit par la synchronisation incontrôlable dénote un état M_k à partir duquel une transition synchrone incontrôlable est validée dans le fonctionnement du procédé N_p et non validée dans le fonctionnement désiré du SED en boucle fermée, $N=N_p||N_s$, à savoir :

$\{M_k = (M_{p,k}\ ,\ M_{s,k}) \in \mathcal{A}(N,\ M_0) \mid \exists\ t_j = (t_p,\ t_s) \in T_{uc}$ tel que, $M_k\ [t_j\ \rangle$ n'est pas validée et $M_{p,k}\ [t_j\rangle$ est validé$\}$.

❒

L'analyse du graphe de marquage montre qu'à partir des ces trois états interdits (M_{19}, M_{13}, M_{16}), il n'y a aucun moyen d'empêcher l'accessibilité aux états interdits par la specfication (M_7, M_{14}). En réalité, tous ces états interdits identifiés par construction du graphe de marquage sont générés par les synchronisations incontrolables et violent la spécification N_s, qui est définie par le nombre de robots disponibles. En considérant pour chaque état, les places d'entrées des transitions synchrones t_2 et t_4, nous pouvons remarquer que $M_k(P_5) < M_k(P_2) + M_k(P_4)$, comme ci-dessous :

- M_7 : $M_7(P_5) = 0$ et $M_7(P_2) + M_7(P_4) = 1$
- M_{14} : $M_{14}(P_5) = 0$ et $M_{14}(P_2) + M_{14}(P_4) = 1$
- M_{19} : $M_{19}(P_5) = 1$ et $M_{19}(P_2) + M_{19}(P_4) = 2$
- M_{13} : $M_{13}(P_5) = 1$ et $M_{13}(P_2) + M_{13}(P_4) = 2$
- M_{16} : $M_{16}(P_5) = 2$ et $M_{16}(P_2) + M_{16}(P_4) = 3$

Ainsi, pour éviter d'atteindre ces états interdits identifiés, il est nécessaire d'avoir $M_k(P_5) \geq M_k(P_2) + M_k(P_4)$ pour chaque état accessible. Cette condition garantira la contrôlabilité du RdPS du SED en boucle fermée, $N=N_P||N_S$.

IV.3.2. Vérification de la contrôlabilité à partir du RdPS du SED en boucle fermée

A partir du RdPS du SED en boucle fermée, $N=N_p||N_s$, le problème de commande par supervision consiste à synthétiser un contrôleur qui restreint l'espace d'état $\mathcal{A}(N, M_0)$ à l'ensemble des états admissibles $\mathcal{M}_A$, en évitant d'atteindre l'ensemble des états interdits $\mathcal{M}_I$.

Prenons en compte la condition sur le marquage des places entrées ${}^\bullet t_j$ en concordance avec la condition de franchissement des transitions. Pour tout état $M_k = (M_{p,k}, M_{s,k}) \in \mathcal{A}(N, M_0)$, la transition synchrone incontrôlable $t_j = (t_p, t_s) \in T_{uc}$ est franchissable si

$$\begin{cases} M_{p,k}({}^\bullet t_j) \geq W_p^-(\bullet, t_j) \\ M_{s,k}({}^\bullet t_j) \geq W_s^-(\bullet, t_j) \end{cases} \quad \text{(IV.3)}$$

Où nous considérons $M_{p,k}({}^\bullet t_j)$ comme le marquage des places d'entrée de t_j, qui appartiennent au procédé N_p et $M_{s,k}({}^\bullet t_j)$ comme le marquage des places d'entrée de t_j, qui appartiennent à la spécification N_s.

L'ensemble des états interdits par la spécification $\mathcal{M}_I$ est caractérisé par des états, tel que : $t_j \in T_{uc}$ est autorisé par $M_{p,k}$ dans N_p mais, non autorisé par $M_{s,k}$ dans N_s. C'est-à-dire,

$$\{\forall t_j = (t_p, t_s) \in T_{uc}, \exists M_k = (M_{p,k}, M_{s,k}) \text{ tel que } \begin{cases} M_{p,k}(^\bullet t_j) \geq W_p^-(\bullet, t_j) \\ M_{s,k}(^\bullet t_j) < W_s^-(\bullet, t_j) \end{cases} \} \quad \text{(IV.4)}$$

Le RdPS du SED en boucle fermée, $N=N_p||N_s$, est contrôlable si et seulement s'il n'est pas possible d'atteindre un état interdit $M_k \in \mathcal{M}_I$, à partir des états sources d'incontrôlabilité. L'ensemble des états sources d'incontrôlabilité correspond aux états faiblement interdits, tel que : t_j est autorisée par $M_{p,k}$ dans N_p mais, non autorisée par M_k dans $N=N_p||N_s$. En d'autres termes, $N=N_p||N_s$ est contrôlable si et seulement si la condition suivante est satisfaite

$$\{\forall t_j = (t_p, t_s) \in T_{uc}, \exists M_k = (M_{p,k}, M_{s,k}) \text{ tel que } \begin{cases} M_{p,k}(^\bullet t_j) \geq W_p^-(\bullet, t_j) \\ M_{s,k}(^\bullet t_j) < M_{p,k}(^\bullet t_j) \end{cases} \} \quad \text{(IV.5)}$$

Sinon, $N=N_p||N_s$ est incontrôlable.

Vérifions ces définitions vis-à-vis des états interdits identifiés dans le graphe marquage du RdPS du SED en boucle fermée de l'exemple IV.1 (figure IV.2).

- Pour la transition synchrone incontrôlable t_2 isolé, $^\bullet t_2 = \{P_2, P_5\}$. Les états à interdire sont donnés par : $\begin{cases} M_{p,k}(P_2) \geq W_p^-(\bullet, t_2) \\ M_{p,k}(P_5) < M_{s,k}(P_2) \end{cases}$
  - $M_7 := M_{s,7}(P_5) = 0, M_{p,7}(P_2) = 1$
  - $M_{19} := M_{s,19}(P_5) = 0, M_{p,19}(P_2) = 2$
- Pour la transition synchrone incontrôlable t_4 isolé, $^\bullet t_4 = \{P_4, P_5\}$. les états à interdire sont donnés par : $\begin{cases} M_{p,k}(P_4) \geq W_p^-(\bullet, t_4) \\ M_{p,k}(P_5) < M_{s,k}(P_4) \end{cases}$
  - $M_{14} := M_{s,14}(P_5) = 0, M_{p,14}(P_2) = 1$
- Pour les transitions synchrones incontrôlables t_2 et t_4 simultané, $^\bullet t_2 \cup {}^\bullet t_4 = \{P_2, P_4, P_5\}$. les états à interdire sont donnés par : $\begin{cases} M_{p,k}(P_2) \geq W_p^-(\bullet, t_2) \text{ et } M_{p,k}(P_4) \geq W_p^-(\bullet, t_4) \\ M_{p,k}(P_5) < M_{s,k}(P_2) + M_{s,k}(P_4) \end{cases}$
  - $M_{13} := M_{s,13}(P_5) = 1, M_{p,13}(P_2) + M_{p,13}(P_4) = 2$ et
  - $M_{16} := M_{s,13}(P_5) = 1, M_{p,13}(P_2) + M_{p,13}(P_4) = 3$

❐

Nous constatons que le RdPS du SED en boucle fermée de l'exemple IV.1 n'est pas contrôlable comme le montre son graphe de marquage (figure IV.3) qui contient toute l'information nécessaire pour la synthèse de contrôleur maximal permissif (Vasiliu, 2012 ; Lacerda, 2013).

Proposition IV.1. Soit $N = N_p || N_s$, le RdPS du SED en boucle fermée. N est contrôlable si et seulement si, pour toute transition synchrone incontrôlable $t_j \in T_{uc}$,

$$\begin{cases} M_{p,k}(^\bullet t_j) \geq W_p^-(\bullet, t_j) \\ M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j) \end{cases} \quad \text{(IV.6)}$$

❐

Preuve :

Cette preuve est évidente puisque, la condition $M_{s,k}(^\bullet t_j) < M_{p,k}(^\bullet t_j)$ définit les états interdits y compris les états interdits frontières qui sont sources d'incontrôlabilité dans le RdPS du SED en boucle fermée. Par conséquent, tout état $M_k \in \mathcal{A}(N, M_0)$ qui ne satisfait pas la condition $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$ est explicitement admissible et contrôlable.

❐

Remarque IV.2. Nous avons pris un exemple simple où les transitions incontrôlables ont juste deux places d'entrées, une pour le RdPS du procédé N_p et une pour le RdPS de la spécification N_s. Cette hypothèse sera considérée dans la suite pour nous concentrer sur la contribution principale. Toutefois la généralisation à une structure quelconque ne pose aucun problème conceptuel.

IV.4. Condition de contrôlabilité structurelle

La contrôlabilité est un concept clef de la théorie de commande par supervision ayant des implications sur l'existence du contrôleur (Cassandras et Lafortune, 2010). Elle a été définie par les langages formels (Wonham, 2017) et constitue la base de synthèse de contrôleur maximal permissif en présence des événements incontrôlables, E_{uc}.

Définition IV.4.1. **(Contrôlabilité)** Etant donné $K = L(N_p) \cap L(N_s)$, le langage défini par la spécification du SED en boucle fermée, $N = N_p || N_s$ et $L(N_p)$ le langage du procédé. K est dit contrôlable par rapport au langage $L(N_p)$ si :

$$\overline{K}\, E_{uc} \cap L(N_P) \subseteq \overline{K},$$

Où $\overline{K}$ est la préfixe-clôture de K.

❒

Rappelons que la synthèse de contrôleurs par RdP aborde rarement le problème de la contrôlabilité (Kumar & Hollaway, 1996) dans le cas général. En fait, la condition de contrôlabilité classique (Wonham, 2003) doit être adaptée au RdP.

Considérons par exemple le RdPS du système manufacturier de l'exemple IV.1, la place P_5 indique la disponibilité des robots. Il n'est pas utile d'examiner si le marquage d'une telle place peut être obtenu à partir d'une séquence quelconque ou à partir d'un état initial. Ce qui signifie que la condition de contrôlabilité peut être redéfinie dans le cadre exclusif des RdPS si nous considérons le fait que le langage $\overline{K}$ caractérise l'ensemble d'états admissibles $\mathcal{M}_A$.

Il semble donc naturel d'introduire une notion différente de contrôlabilité, plus adaptée à la structure du RdPS du SED en boucle fermée, pour une spécification donnée. C'est-à-dire, une condition de contrôlabilité structurelle définissant un ensemble d'états admissibles $\mathcal{M}_A$ généré par des séquences admissibles dans le procédé N_p.

Dans le comportement dynamique du RdPS $N = N_p || N_s$, la relation $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$ est une condition nécessaire pour la contrôlabilité des états M_k, lorsque $M_{p,k}(^\bullet t_j) \geq W_p^-(\bullet, t_j)$. L'équation $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$ fournit simplement une condition de franchissement de la transition synchrone incontrôlable (Gonza et *al.*, 2019).

Définition IV.4.2. (Condition de contrôlabilité structurelle) Considérons le RdPS d'un SED en boucle fermée, $N = N_p || N_s$ et la transition synchrone incontrôlable, $t_j \in T_{uc}$. Soit $\mathcal{M}_A \subseteq \mathcal{A}(N, M_0)$, l'ensemble d'états admissibles du SED. Alors, la condition

$$M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j) \qquad \text{(IV.7)}$$

est appelée condition de contrôlabilité structurelle

❒

Cette définition spécifie le rôle du contrôleur qui, consiste à désactiver les transitions contrôlables pour éviter les états interdits générés par les synchronisations incontrôlables et à s'assurer que le SED en boucle fermée, $N = N_p || N_s$, ne génère que des états dans $\mathcal{M}_A$.

Propriété IV.4.1. La condition de contrôlabilité structurelle $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$ définie dans le cadre des RdPS est équivalente à la condition de contrôlabilité classique, $\overline{K}\, E_{\text{uc}} \cap L(N_p) \subseteq \overline{K}$ basée sur les langages. C'est-à-dire,

$$M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j) \Leftrightarrow \overline{K}\, E_{uc} \cap L(N_P) \subseteq \overline{K}$$

❒

Preuve.

Considérons le RdPS du SED en boucle fermée $N = N_p || N_s$, où N_s est une spécification sur les états $M_k \in \mathcal{A}(N, M_0)$

- $L(N) = L(N_p) || L(N_s)$, est l'ensemble des séquences qui permet d'atteindre tout état de l'espace d'état, $\mathcal{A}(N, M_0)$ à partir l'état initial M_0. Cet ensemble est préfixe-clos.
- $L(N) = \varepsilon$, correspond à l'état initial M_0 par définition

Soit $K = L(N_p) \cap L(N_s)$ avec $K \subseteq L(N)$ le langage de la spécification du SED

- K, est l'ensemble des séquences qui permet d'atteindre tout état dans l'espace d'état pertinent, $\mathcal{M}_A \cup \mathcal{M}_B$
- $\overline{K}$, est ensemble des séquences admissibles qui permet d'atteindre tout état admissible $M_k \in \mathcal{M}_A$ dans le fonctionnement du SED en boucle fermée.

1. $\Rightarrow$: Nous savons que, $\overline{K}\, E_{\text{uc}} \cap L(N_p) \subseteq \overline{K}$

Considérons une séquence admissible $s \in \overline{K}$ (c'est-à-dire, $M_0[s\rangle \in \mathcal{M}_A$), suivie par un événement incontrôlable $\mu \in E_{uc}$, associée à la transition synchrone $t_j = (t_p, t_s)$, telle que $s\mu \in L(N)$. Alors, les places d'entrée $^\bullet t_p$ dans N_p sont marquées, $M_{p,k}(^\bullet t_j) \geq W_p^-(\bullet, t_j)$. Puisque, $s\mu \in \overline{K}$ alors, la transition synchrone incontrôlable t_j est aussi autorisé dans $N = N_p || N_s$, et le marquage des places d'entrée $^\bullet t_s$ dans N_s vérifie nécessairement $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$.

2. $\Leftarrow$: Nous savons que, $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$

Considérons un état admissible $M_k \in \mathcal{M}_A$, accessible à partir de M_0 après une séquence admissible (c'est-à-dire, $s \in \overline{K}$), suivit par un état $M_k[\mu\rangle$ après occurrence de $\mu \in E_{uc}$, si $s\mu \notin L(N_P)$, l'événement incontrôlable ne peut pas être interdit dans le procédé alors

$s\mu \in \bar{K} E_{uc} \cap L(N_p)$. Comme pour cette état nous avons : $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$, alors la transition incontrôlable t_j est autorisée dans $N = N_p||N_s$ et la séquence $s\mu \in \bar{K}$.

❒

IV.4. Etats admissibles et interdits par la condition de contrôlabilité structurelle

A partir de la structure du RdPS du SED en boucle fermée, nous pouvons identifier sans construire le graphe de marquage, les états interdits et le non-blocage (Holloway et *al.*, 2004 ; Uzam, 2010; Iordache et *al.*, 2013). En effet, une synchronisation incontrôlable peut générer une situation anormale dans laquelle le procédé N_p a atteint un état à partir duquel la transition synchrone incontrôlable $t_j \in T_{uc}$ peut être franchie, mais n'est pas admissible dans $N=N_p||N_s$. Un tel état, source d'incontrôlabilité, est inclus dans l'ensemble des états interdits $\mathcal{M}_I$. Les états interdits peuvent être déterminés par la condition de contrôlabilité structurelle, qui est en concordance avec la condition de franchissement des transitions synchrones incontrôlables. En réalité, chaque fois qu'un tel état (qui doit être interdit) est atteint, si l'événement incontrôlable associé à la transition synchrone se produit, le RdPS, $N=N_p||N_s$ évolue vers les états interdits par la spécification. Puisqu'il n'y a pas moyen d'empêcher le franchissement d'une transition incontrôlable, ces états doivent aussi être évités ou interdits.

Définition IV.3.3. (Etats interdits) Soit $\mathcal{A}(N, M_0)$ l'espace d'état accessible de $N = N_P||N_S$, avec N_S la spécification de type état. L'ensemble des états interdits $\mathcal{M}_I$ (y compris les états sources d'incontrolabilités qui sont des états interdits frontières $\mathcal{M}_B$) est celui pour lequel la condition de contrôlabilité structurelle $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$ est violée. Il est donné par :

$$\mathcal{M}_I = \{M_k \in \mathcal{A}(N, M_0) , \exists\ t_j = (t_p, t_s) \in T_{uc} ,\ M_{s,k}(^\bullet t_j) < M_{p,k}(^\bullet t_j) \} \quad \text{(IV.8)}$$

❒

Cette nouvelle définition est une amélioration de celle utilisée dans des travaux antérieurs (Kumar & Holloway, 1996; Giua, 2013) qui caractérise correctement l'incontrôlabilité pour une spécification donnée. De manière correspondante, nous définissons l'ensemble des états admissible.

Définition IV.3.3. (Etats admissibles) Soient $N = N_p||N_s$, le RdPS du SED en boucle fermée et $\mathcal{A}(N, M_0)$ l'espace d'état accessible. L'ensemble des états admissibles, $\mathcal{M}_A$ est celui pour lequel la condition $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$ est vérifiée. Il est donné par :

$$\mathcal{M}_A = \{M_k \in \mathcal{A}(N, M_0)\ ,\ \exists\ t_j = (t_p, t_s) \in T_{uc}\ ,\ M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)\ \} \quad \text{(IV.9)}$$

❒

L'ensemble $\mathcal{M}_A$ est contrôlable en présence des transitions incontrôlables $t_j \in T_{uc}$ et vis à vis de $\mathcal{A}(N, M_0)$, si $\mathcal{M}_A \cap \mathcal{A}(N, M_0) \subseteq \mathcal{M}_A$. Par conséquent, tous les états accessibles sont admissibles sous la condition de contrôlabilité structurelle $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$.

Considérons le graphe de marquage (espace d'état pertinent) de l'exemple IV.1 traité dans la section 2 (figure IV.3), la condition de contrôlabilité structurelle définie par les deux transitions synchrones donne l'ensemble d'états admissible

$$\mathcal{M}_A = \{M_0, M_1, M_2, M_3, M_4, M_5, M_6, M_8, M_9, M_{10}, M_{11}, M_{12}, M_{15}, M_{17}, M_{18}\}$$

Cette mise au point montre que la condition de contrôlabilité structurelle est nécessaire pour l'admissibilité des états et l'identification des états interdits et de blocage.

Exemple IV.2

Considérons un système de production (Achour, 2005) constitué de deux ressources R_1 et R_2 fabriquant deux types de produits A et B. La spécification est définie par la disponibilité des ressources R_1 et R_2. Le modèle RdPS du système en boucle fermée, N est donné à la figure 4.6.a. L'ensemble des transitions contrôlables est $T_c = \{t_1, t_4\}$. Donc, le contrôleur pourra contrôler uniquement le déclenchement (d_i,) du processus de fabrication. Il ne peut qu'observer I 'arrivée (a_i) et la sortie (s_i) des pièces. L'ensemble des transitions incontrôlables est $T_{uc} = \{t_2, t_3, t_5, t_6\}$. Le graphe de marquage $\mathcal{G}(N, M_0)$ est donné à la figure 4.6.b

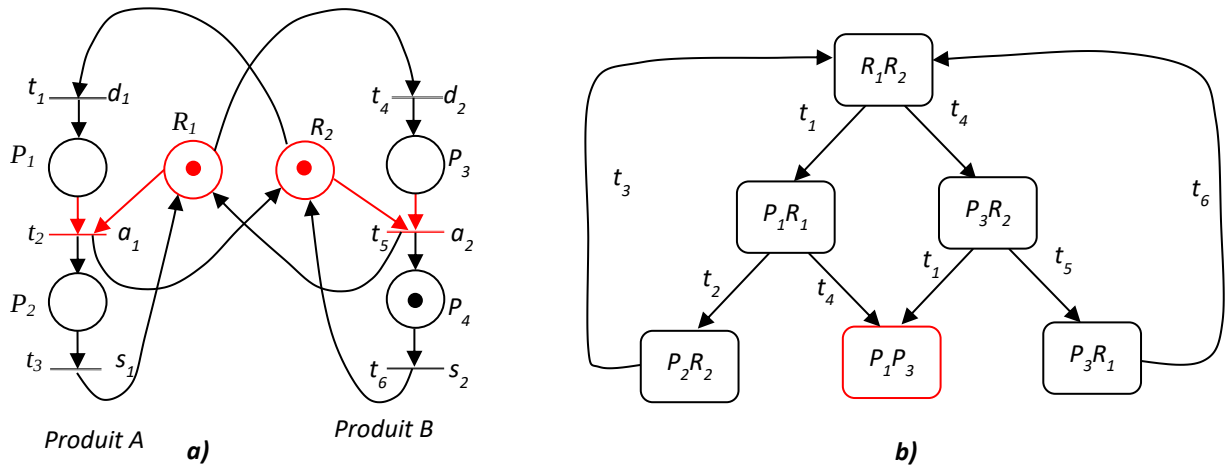


Figure IV.4 – Modèles du système de production : *a*) RdPS du système en boucle fermée, *b*) graphe de marquage $\mathcal{G}(N, M_0)$

Les transitions synchrones incontrôlables sont $\{t_2, t_5\} \in T_{uc}$, avec ${}^\bullet t_2 = \{P_1, R_1\}$, ${}^\bullet t_5 = \{P_3, R_2\}$

$\mathcal{M}_A = \{R_1R_2, P_1R_1, P_2R_2, P_3R_2, P_3R_1\}$; avec

- $\mathcal{M}_{A,\,t2} = \{R_1R, P_1R_1, P_2R_2, P_3R_2, P_3R_1\}$; $M(R_1) \geq M(P_1)$
- $\mathcal{M}_{A,\,t5} = \{R_1R_2, P_1R_1, P_2R_2, P_3R_2, P_3R_1\}$; $M(R_2) \geq M(P_3)$

$\mathcal{M}_I = \{P_1P_3\}$

▲

Au vu de tout ce qui précède, la condition de contrôlabilité structurelle peut être traduite en contrainte admissible sur les marquages, de façon à constituer un lien biunivoque entre la théorie de la commande par supervision et la méthode par les invariants de place. Selon le paradigme de la théorie de commande par supervision, le RdPS d'un SED en boucle fermée, $N = N_p||N_s$, génère un langage $L(N) = L(N_p) \cap (N_s)$. Le problème de contrôle consiste à synthétiser un contrôleur qui restreint le fonctionnement de $N = N_p||N_s$ à $\mathcal{M}_A \cap \mathcal{A}(N, M_0)$, tout en évitant les états interdits générés par les synchronisations incontrôlables.

IV.5. Contraintes admissibles et condition de contrôlabilité structurelle

L'ensemble des états admissibles $\mathcal{M}_A$ représente le comportement maximal permissif et peut être prédit à l'avance par la condition de contrôlabilité structurelle ; d'où son utilité pour la

détermination des contraintes admissibles pour la méthode des invariants de place (Huang et al., 1996 ; Tacconi et *al.*, 1996)

IV.5.1. Contraintes admissibles

Ici, nous sommes préoccupés par la détermination des contraintes linéaires (de type GMEC) qui vérifient la condition de contrôlabilité structurelle pour chaque transition synchrone incontrôlable $t_j = (t_p, t_s) \in T_{uc}$ du RdPS du SED en boucle fermée. Nous considérons l'ensemble des places d'entrée ${}^\bullet t_j$, qui appartiennent au procédé N_p et l'ensemble de places d'entrée ${}^\bullet t_j$, qui appartiennent à la spécification N_s. Si l'un de ces deux ensembles est borné alors il existe une contrainte de type GMEC qui est admissible (Giua et *al.*, 1992).

Définition IV.5.1. (Contrainte admissible) Etant donné un RdPS du SED en boucle fermée avec un état initial M_0. Pour chaque transition synchrone incontrôlable $t_j = (t_p, t_s) \in T_{uc}$, une contrainte admissible satisfait deux conditions.

- (1) $M_{s,0}({}^\bullet t_j) \geq M_{p,0}({}^\bullet t_j)$
- (2) Pour tout état M_k accessible à partir M_0 via une séquence d'états admissible M_i où $M_{s,i}({}^\bullet t_j) \geq M_{p,i}({}^\bullet t_j)$, pour i=1, n ; $M_{s,k}({}^\bullet t_j) \geq M_{p,k}({}^\bullet t_j)$

Si une contrainte ne satisfait pas les deux conditions alors elle est inadmissible

❐

Exemple IV. 3

Considerons un système magasin/consommateur. Le magasin vend un seul type de produits entreposés dans un stock de capacité limitée : son état est caractérisé par le nombre de produits en stock P_1 et par le nombre d'emplacements libres dans le stock P_2. Le nombre de produits disponibles dans le magasin augmente quand le magasin est livré t_1 (*l*) et diminue quand le consommateur effectue un achat t_2 (*a*). Le consommateur a deux états : soit il consomme P_3, soit il va faire ses courses P_4. t_3 (*f*) symbolise la fin de consommation.

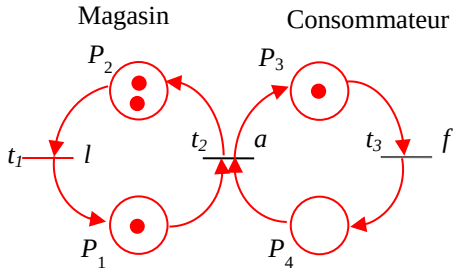


Figure IV.5 - RdPS du système magasin/consommateur en boucle fermée

Le RdPS de la figure IV.5 contient 2 transitions incontrolables t_1 et t_2. Le marquage des places P_1, P_2 et P_3 ne peut pas être contrôlé à partir du franchissement de ces transitions.

❖ **La contrainte $M(P_1) + M(P_3) \leq 2$ est inadmissible**

L'état initial du système $M_0 = [1\ 2\ 1\ 0]^T$ satisfait la contrainte, et la condition de controlabilité structurelle est $M_k\ (P_1) \geq M_k\ (P_4)$. Le franchissement incontrolable t_1 pourra conduire à l'état $M_1 = [2\ 1\ 1\ 0]^T$ qui viole $M_k\ (P_1) + M_k\ (P_3) \leq 2$ alors que $M_k\ (P_1) \geq M_k\ (P_4)$. La contrainte empêche le franchissement de t_1 à partir de l'état initial et interdit un état admissible. Par ailleurs, le franchissement de la sequence t_3t_1 pourra conduire à l'état $M_3 = [2\ 1\ 0\ 1]^T$ (accessible à partir M_0 via l'état admissible $M_2 = [1\ 2\ 0\ 1]^T$ qui viole $M_k\ (P_1) + M_k\ (P_3) \leq 2$, alors que $M_k\ (P_1) \geq M_k\ (P_4)$.

Cette contrainte n'est pas admissible car elle a échoué à la condition (2) de la défintion IV.4.1 Ce problème peut etre généralement causé par la relation de couverture entre états interdits et états admissibles (Alla et *al.*, 2000) .

❖ **La contrainte $M_k\ (P_2) + M_k\ (P_3) \leq 3$ est admissible**

Soit $M_k = [0\ 3\ 0\ 1]^T$, l'état interdit par $M_k(P_1) \geq M_k(P_4)$. Si nous calculons la contrainte associée à cet état en limitant la somme des jetons dans les places concernées, nous obtenons : $M_k(P_2) + M_k(P_3) \leq 3$. Cette contrainte est admissible selon la definition, pour tout état accessible qui satisfait $M_k\ (P_1) \geq M_k\ (P_4)$. Ainsi, la contrainte $M_k\ (P_2) + M_k\ (P_3) \leq 3$ satisfait aussi $M_k\ (P_1) + M_k\ (P_3) \leq 2$ et peut être imposée par un contrôleur calculé en utilisant la méthode des invariants de place.

Malheureusement, le controleur calculé de cette manière necessite souvent la construction du graphe de marquage (Gonza & Bitjoka, 2018). De plus, nous n'avons aucune garantie que le

controleur soit maximal permissif surtout pour des contraintes GMEC-non-pondérés où le support des états est unique.

Considérons l'état interdit $M_7 = P_1P_2P_3\ P_6^3$ de l'exemple IV.1. La contrainte associée à cet état est : $M(P_1) + M(P_2) + M(P_3) + M(P_6) \leq 5$. Cette contrainte n'est pas admissible, car bien qu'elle interdit l'état $M_7 = P_1P_2P_3\ P_6^3$, elle interdit aussi un état admissible : $M_6 = P_1^2P_3\ P_6^3 \in \mathcal{M}_A$ à cause de l'idée de l'unicité du support.

▲

La restriction du fonctionnement désiré du SED en boucle fermée par une contrainte peut s'avérer très dangereuse pour un système plus complexe, où elle peut entraîner l'interdiction d'un très grand nombre d'états admissible (Vasiliu, 2012). Par conséquent nous allons proposer une technique de détermination des contraintes admissibles qui assure une relation bijective entre l'ensemble des états admissibles et la condition de controlabilité structurelle.

IV.5.2. De la condition de la contrôlabilité structurelle à la contrainte admissible

Le problème de la commande par supervision peut être défini par l'ensemble d'états admissibles $\mathcal{M}_A$ ou par l'ensemble des états interdits $\mathcal{M}_I$ (Rezig, 2016). Soient $M_k \in \mathcal{M}_B$ un état interdit et $\left\{ p_i^{M_k(pi)} \mid i = 1, n, \text{avec } M_k(p_i) \geq 1 \right\}$ toutes les places marquées correspondantes. A partir des états interdits les contraintes linéaires peuvent être construites (Kattan, 2004; Toma et al, 2003). La contrainte lineaire pour cet état est donné par l'équation.

$$L^T.M_k \leq b_v \qquad \text{(IV.10)}$$

Où $L^T = [l_1\ \ l_2\ \ldots\ l_n] \in \mathbb{Z}^{nc\ x\ n}$ représente la matrice de pondération des contraintes, $b_v \in \mathbb{Z}^{nc}$ est le vecteur des bornes.

Par exemple, l'état interdit $M_7 = P_1P_2P_3\ P_6^3$ peut etre interdite par la contrainte $M_7(P_1) + M_7(P_2) + M_7(P_3) + M_7(P_6) \leq 5$, où $L^T = [1\ 1\ 1\ 0\ 0\ 1]$ et $b_v = 5$

▲

Il est possible d'utiliser la méthode d'invariants de place pour imposer ce type de contrainte (Achour, 2005). Ces contraintes de la forme d'inegalité ($L^T.M_k \leq b_v$) sont utiles pour representer une large gamme de problèmes d'états interdits (Moody et Antsaklis, 1998),

même lorsque l'état interdit par la spécification n'apparait pas directement parmi les états du SED. Cependant, le contrôle des SED n'est pas seulement utilisé pour prévenir les états interdits mais, aussi pour imposer le SED à realiser un fonctionnement désiré ou un langage désiré (Li & Wonham, 1994, Giua & DiCesare, 1994).

Nous avons vu au chapitre 3 que deux autres ensembles d'états peuvent être construits à partir de $\mathcal{M}_A$ et $\mathcal{M}_I$: l'ensemble des états interdits frontières $\mathcal{M}_B$ et l'ensemble des états admissibles critiques $\mathcal{M}_{AC}$. L'ensemble des états admissibles critiques $\mathcal{M}_{AC}$, correspond aux états admissibles à partir desquels l'occurrence d'événements contrôlables mène vers un état interdit. Et, la condition de controlabilité structurelle permet aussi de définir $\mathcal{M}_{AC}$, à savoir :

Définition IV.5.2. (Etats admissibles critiques) Soit $t_j = (t_p, t_s) \in T_{uc}$, l'ensemble des états admissibles critiques, $\mathcal{M}_{AC}$, correspond aux états admissibles dont les marquages des places d'entrée sont égaux ; et à partir desquels un état interdit frontière peut etre atteint par le franchissement d'une transition controlable.

$$\mathcal{M}_{AC} = \{\mathcal{M}_k \in \mathcal{M}_A \mid \exists\ t_j = (t_p, t_s) \in T_{uc}\ ;\ M_{s,k}(^\bullet t_j) = M_{p,k}(^\bullet t_j)\ \}$$

❐

Par l'interdiction de franchissement des transitions contrôlables à partir des états admissibles critiques $\mathcal{M}_{AC}$, nous empêchons l'accessibilité de tout état interdit dans $\mathcal{M}_I$. Par conséquent, le franchissement d'une transition incontrôlable ne conduira pas à état qui viole $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$. Si cette condition est imposée à tout état M_k alors, $N = N_P || N_S$ doit quitter l'état M_k par franchissement d'une transition sans violer la contrainte écrite sous forme d'inégalité de type $L^T . M_k \leq b_v$, à savoir :

$$M_{p,k}(^\bullet t_j) - M_{s,k}(^\bullet t_j) \leq 0 \ où \quad L^T = \left[0, .., l_p, 0, .., l_s, 0 \ldots, 0\right] \ et \ b_v = 0 \qquad \text{(IV.11)}$$

Il est prouvé dans la littérature que : si l'ensemble des états admissibles $\mathcal{M}_A$ est exprimé par un ensemble de contraintes linéaires et si $\mathcal{M}_A$ est contrôlable, alors une solution de contrôle existe et elle est maximale permissive. Par conséquent, la condition $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$ defini sur le RdPS du SED en boucle fermée, $N = N_p || N_s$, respecte les spécifications et correspond au langage admissible $L_a(N)$ contenant seulement les séquences admissibles. Toutefois, Il peut arriver qu'un état de $N = N_p || N_s$ viole la condition de contrôlabilité, d'où la

nécessité d'un contrôleur qui permet d'obtenir $\mathcal{M}_A$, en inhibant les transitions contrôlables à partir de $\mathcal{M}_{AC}$ pour empecher les états interdits. Il s'agit d'effectuer une séparation de l'espace d'états accessibles $\mathcal{A}(N, M_0)$ en ensembles disjoints d'états admissibles $\mathcal{M}_A$ et interdits $\mathcal{M}_I$ incluant les états interdits frontières, $\mathcal{M}_B$. Avantageusement, la contrainte $M_{p,k}(^\bullet t_j) - M_{s,k}(^\bullet t_j) \leq 0$ définit l'équation de l'hyperplan séparateur qui correspond à la condition structurelle de séparation des états, à savoir

$$M_{p,k}(^\bullet t_j) - M_{s,k}(^\bullet t_j) = 0 \qquad \text{(IV.12)}$$

L'hyperplan séparateur constitue la frontière assurant la partition de l'espace d'états accessibles $\mathcal{A}(N, M_0)$. Le contrôleur implémentant un tel hyperplan garantit l'admissibilité de l'état initial et accessibilité à tout état admissible : $M_0 \in \mathcal{M}_A$ et $M_0\,[s\rangle \in \mathcal{M}_A$.

Définition IV.5.3. (Hyperplan séparateur) Soit n le nombre de places du RdPS $N = N_P || N_S$ et soit $L^T \in \mathbb{Z}^n$ le vecteur de contrainte ; et $t_j = (t_p, t_s) \in T_{uc}$, la transition synchrone incontrôlable. l'hyperplan séparateur $\mathcal{H}\left(L, t_j\right)$ est caractérisé par des états $M_k \in \mathcal{M}_A$ tel que $M_k\,[tj\rangle \in \mathcal{M}_I$,

$$\mathcal{H}\left(L, t_j\right) = \{L^T \in \mathbb{Z}^n \mid M_{p,k}(^\bullet t_j) - M_{s,k}(^\bullet t_j) = 0\} \qquad \text{(IV.13)}$$

Où $L^T = \left[0, \ldots, l_p, 0, .., l_s, 0 \ldots, 0\right]$ *avec* $l_s = 1, l_p = -1$ et tout autre coefficient est nul

❐

Exemple IV.3

Considérons le graphe de marquages (figure IV.6) de l'exemple classique de commande par supervision (voir chapitre 3). Vis-à-vis de la transition synchrone t_2 associée à l'événement incontrôlable f_1, nous avons :

$\mathcal{H}\left(L, t_2\right) = \left\{ \left[0\,1\,0\,0\,-1\,0\right] ; M_{p,k}\left(P_2\right) - M_{s,k}\left(P_5\right) = 0 \right\}$ et $\mathcal{M}_{AC} = \{P_1P_3P_6\,,\, P_1P_4P_6\,\}$

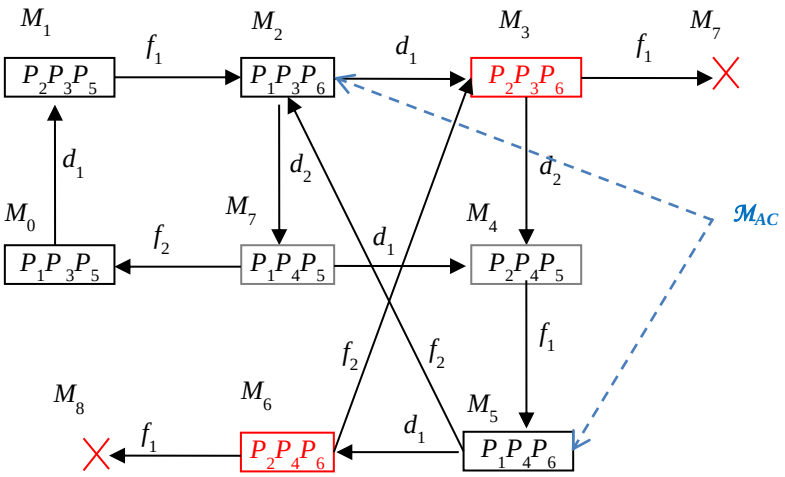


Figure IV.6 - Graphe de marquages de l'exemple classique de commande par supervision

▲

Les contraintes déduites de l'équation d'hyperplan séparateur peuvent être systématiquement simplifiées par la propriété d'invariant partiel (Dideban, 2007) puisqu'il est possible d'avoir plusieurs transitions synchrones incontrôlables partage les mêmes places d'entrée.

Considérons le graphe de marquages de l'exemple IV.1. Nous avons, vis-à-vis des deux transitions t_2 et t_4, les hyperplans séparateurs:

$$\mathcal{H}(L,t_2)=\left\{\left[0\,1\,0\,0\,-1\,0\right];M_{p,k}(P_2)-M_{s,k}(P_5)=0\right\}$$

$$\mathcal{H}(L,t_4)=\left\{\left[0\,0\,0\,1\,-1\,0\right];M_{p,k}(P_4)-M_{s,k}(P_5)=0\right\}$$

Simplification :

Puisqu'il n'est pas possible que P_5 ne soit pas marqué et en même temps que P_2 et P_4 soient marquées, c'est-à-dire :

$$M_{s,k}(P_5)\leq 0 \; et \; M_{p,k}(P_2)+M_{p,k}(P_4)+M_{s,k}(P_5)>0 \Rightarrow M_{p,k}(P_2)+M_{p,k}(P_4)\leq 0$$

Nous pouvons écrire ce qui suit,

$$\begin{cases} M_{p,k}(p_2)-M_{s,,k}(p_5)=0 \\ M_{p,k}(p_4)-M_{s,,k}(p_5)=0 \end{cases} \xleftarrow{M_{p,k}(p_2)+M_{p,k}(p_4)\leq 0} M_{p,k}(p_2)+M_{p,k}(p_4)-M_{s,,k}(p_5)=0$$

Finalement nous avons simplement

$$\mathcal{H}(L,t_{2,4})=\left\{\left[0\,1\,0\,1\,-1\,0\right],M_{p,k}(P_2)+M_{p,k}(P_4)-M_{s,k}(P_5)=0\right\}$$

L'ensemble $\mathcal{M}_{AC} = \{P_1P_2P_4P_5^2P_6, P_1P_2P_3P_5P_6^2, P_1^2P_3P_6^3, P_1^2P_4P_5P_6^2\}$

▲

A partir de tout état admissible critique, nous pouvons interdire le franchissement d'une transition contrôlable lorsque qu'elle est validée par la condition de franchissement. Nous savons que la méthode synthèse de contrôleurs par retour d'état permet effectivement d'interdire le franchissement d'une telle transition. Malheureusement, elle est d'autant plus compliquée que le nombre d'états critiques est important. En ce qui concerne la théorie de regions, (Ramadge et Wonham, 1987), le problème peut etre résoulu de façon optimale, au sens maximal permissif, par l'ajout d'une nouvelle place de contrôle P_C au RdPS du SED en boucle fermée (Toma, 2005). Malheureusement, cette méthode est complexe et les contraintes sont exprimées sous forme de marquages interdits. Or, nos contraintes sont exprimés les sous forme de contraintes linéaires de type GMEC, plus concises que les marquages interdits. Notre seule alternative (nous l'avons vu dans le chapitre 3) est la méthode des invariants de place où les spécifications sont des contraintes linéaires de type GMEC. Bien qu'à l'origine la méthode des invariants ne se penche pas sur le problème de contrôlabilité, la condition de controlabilité structurelle s'y penche (Proprieté IV.4.1) et permet de déterminer les contraintes admissibles pour garantir l'optimatlité, même en présence d'incontrolabilité. Il est admis que les contraintes de type GMEC ne sont pas générales comme celles definies les langages (Kumar et Holloway, 1996). Mais il est intéressant de les utiliser à cause de la facilité avec laquelle elles peuvent etre imposées aux RdPS.

IV.6. Synthèse structurelle de contrôleur maximal permissif par les invariants de place

Notre objectif est de synthétiser un contrôleur par une méthode structurelle, qui prend en compte les états interdits générés par les synchronisations incontrôlables et les couvertures entre les états. Un tel contrôleur pourra garantir que le SED n'entrera pas dans un état interdit. En effet, nous disposons dès le départ, des RdPS du procédé N_p et celui de la spécification N_s. Le produit synchrone nous permet de construire le RdPS du SED en boucle fermée, $N=N_p||N_s$.

Si $\mathcal{A}(N, M_0) \subseteq \mathcal{M}_A$, alors $N=N_p||N_s$ est contrôlable et il peut sembler naturel d'utiliser N_S, comme contrôleur approprié pour assurer que seuls les états admissibles sont générés. Mais, il n'y a aucune garantie que N_S autorise tous les états admissibles: il peut tenter de bloquer le franchissement d'une transition incontrôlable.

Dans le cas où $N=N_p||N_s$ est incontrôlable, un contrôleur maximal permissif, dénoté *C*, doit être déterminé à partir de la condition de controlabilité structurelle et à l'aide de la méthode des invariants de place.

Définition IV.6.1 (Contrôleur est maximal permissif) Un contrôleur est maximal permissif si tous les états admissibles sont accessibles sous contrôle et tout franchissement de transition, causant l'évolution du SED à partir d'un état admissible vers un état interdit par la spécification, n'est pas autorisé.

❒

IV.6.1. Synthèse de contrôleur via la condition de contrôlabilité structurelle

La contrainte admissible définie par la condition de contrôlabilité structurelle permet de calculer le contrôleur maximal permissif sans qu'il ne soit nécessaire d'analyser son graphe de marquage, qui peut être très important.

Les contraintes $M_{p,k}(^\bullet t_j) - M_{s,k}(^\bullet t_j) \le 0$ définies par la condition de contrôlabilité structurelle sont des inégalités linéaires qui caractérisent tous les états admissibles de l'espace d'états $\mathcal{A}(N, M_0)$. Pour tout état accessible une telle contrainte satisfait l'invariant de marquage des places (Yamalidou et *al*., 1996). A cet effet, nous pouvons calculer pour chaque contrainte, la place de contrôle qui doit être connectée avec la structure du RdPS, $N = N_p||N_s$. Les arcs connectant les places de contrôle à la structure de $N = N_p||N_s$ sont définis par l'équation

$$\forall M_k \in \mathcal{A}(N, M_0),\ x^T.M_k = x^T.M_0 \qquad \text{(IV.14)}$$

Où le vecteur x est le P-semi-flow désiré de dimension, Card[$P_p \cup P_s$*] = n*

Il est nécessaire d'ajouter à la contrainte d'inégalité $M_{p,k}(^\bullet t_j) - M_{s,k}(^\bullet t_j) \le 0$, une quantité non négative, qui représente le marquage des places de contrôle : $M_C = M(P_{c,i}),\ P_{c,i} \in P_C$, tel que nous puissions avoir

$$M_C = -l^T M_k\ ;\quad où\ l^T = [0, \ldots, l_p, 0, \ldots, l_s, 0 \ldots, 0] \qquad \text{(IV.15)}$$

Ainsi, l'invariant de marquage des places qui assure le respect de la condition de contrôlabilité structurelle $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$ dans $N=N_p||N_s$ est donné par :

$$l^T M_k + M\left(P_{c,i}\right) = 0 \Leftrightarrow \left[l^T\ 1\right].M_k = 0 \qquad \text{(IV.16)}$$

Si le vecteur de pondération $x^T = [l^T\ 1]$, alors $x^T \cdot W = 0 \Leftrightarrow \left[l^T\ 1\right].\left[W_N\ W_C\right]^T = 0.$

Nous pouvons calculer :

- la matrice d'incidence du contrôleur : $W_C = -l^T.W_N$
- le marquage initial du contrôleur : $M_{C0} = -l^T.M_0$

De façon générale, la matrice d'incidence du contrôleur W_C et son état initial M_{C0} sont donnés par les équations matricielles suivantes :

$$\begin{aligned} &W_C = -L^T.W_N \\ &M_{C0} = -L^T.M_0 \\ &\text{où } L = \left[l_1\ l_2\ \ldots l_r\right] \in \mathbb{Z}^{n\ x\ Card[P_c]}\text{, est la matrice de pondération des contraintes} \end{aligned} \qquad \text{(IV.17)}$$

Le contrôleur C calculé doit être connecté au RdPS $N = N_P||N_S$ par synchronisation structurelle (Giva, 1996). Le comportement de $C||N$ sera exactement le comportement désiré à savoir, $L(C)\cap L(N) = \overline{K}$ et $\overline{K}\,E_{uc}\cap L(N_P) \subseteq \overline{K}$

Nous implémentons cette stratégie en calculant le contrôleur du système de l'exemple IV.1 ;
La condition de contrôlabilité obtenue est : $M_{s,k}\left(P_5\right) \geq M_{p,k}\left(P_2\right) + M_{p,k}\left(P_4\right)$

L'hyperplan séparateur est : $M_{p,k}\left(P_2\right) + M_{p,k}\left(P_4\right) - M_{s,k}\left(P_5\right) = 0$ et $L^T = [0\ 1\ 0\ 1\ -1\ 0]$

Nous avons :

$$W_C = -[0\ 1\ 0\ 1\ -1\ 0]\begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & -1 & 0 & -1 & 1 \\ 0 & 1 & 0 & 1 & -1 \end{bmatrix} = [-1\ \ 0\ \ -1\ \ 0\ \ 1]$$

$M_{c0} = [3]$

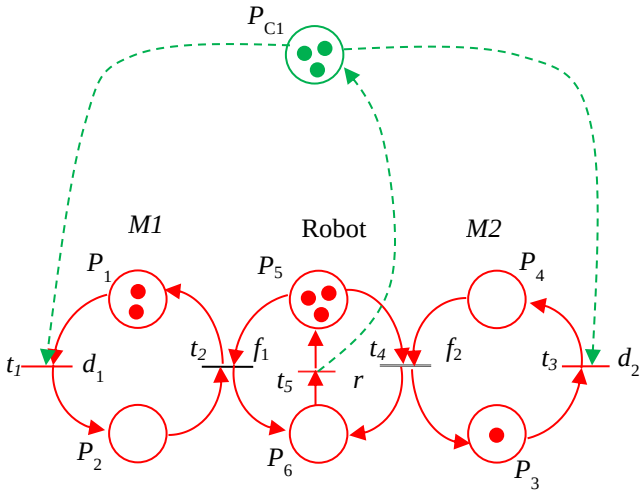


Figure IV.7 – RdPS de l'exemple IV.1 Contrôlé

▲

Pour aboutir à un tel résultat, Vasiliu (2012) a dû recourir à un algorithme de minimisation du nombre des contraintes afin de déterminer le nombre minimal de contraintes à conserver. Son approche est complexe et nécessite, en plus, la construction du graphe de marquage.

Notre démarche ne nécessite pas la construction du graphe de marquages. À partir de l'équation d'hyperplan séparateur définie par la condition de contrôlabilité structurelle, nous déduisons les contraintes admissibles pour la méthode des invariants de place. Par ailleurs, l'existence d'un contrôleur maximal permissif n'est pas garantie dans la méthode de Vasiliu (2012), alors ce problème est résolu dans notre approche par l'équation d'hyperplan séparateur qui prouve formellement que, si une solution de contrôle existe, elle est déterminée par la condition de contrôlabilité structurelle. C'est-à-dire l'admissibilité de l'état initial et la séparabilité des ensembles d'états admissibles et interdits.

IV.7. Application de méthode structurelle à quelques exemples classiques

Nous pouvons maintenant calculer les contrôleurs pour différents exemples classiques et explicites afin d'évaluer la pertinence d'une telle démarche.

IV.7.1. Calcul des contrôleurs pour des SED classiques

Exemple IV.4. Système classique de la théorie de commande par supervision

Reprenons l'exemple I.5 du système classique de la théorie de commande par supervision étudié au chapitre 3.

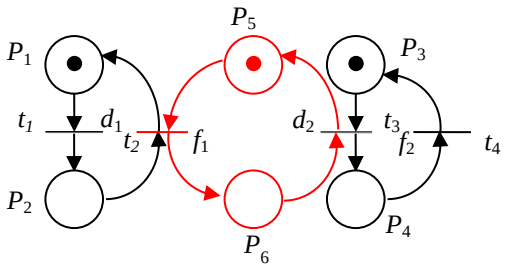


Figure IV.8 – RdPS du système manufacturier classique en boucle fermée

La condition de contrôlabilité obtenue est : $M_{s,k}(P_5) \geq M_{p,k}(P_2)$

L'hyperplan séparateur est : $M_{p,k}(P_2) - M_{s,k}(P_5) = 0$ et L^T= [0 1 0 0 -1 0]

Nous avons :

W_C = [–1 0 1 0]

$M_{c0} = 1$

En connectant le contrôleur au RdPS du système en boucle fermée, nous obtenons le même résultat que celui donné par la théorie des régions qui, se veut optimal.

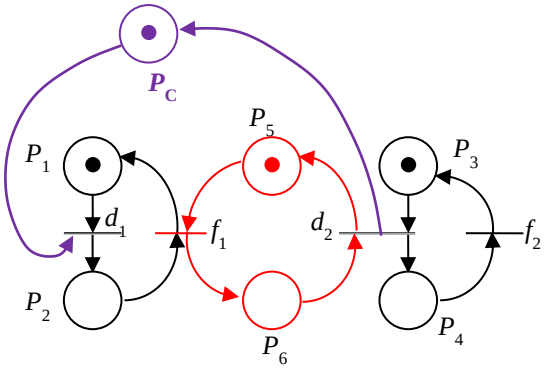


Figure IV.9– RdPS contrôlé du système manufacturier classique

L'optimalité du contrôleur repose sur la conservativité du RdPS du système en boucle fermée. C'est-à-dire aucun état interdit ne couvre un état admissible. Supposons que la spécification ne permet que le traitement d'une seule pièce, figure ci-dessous

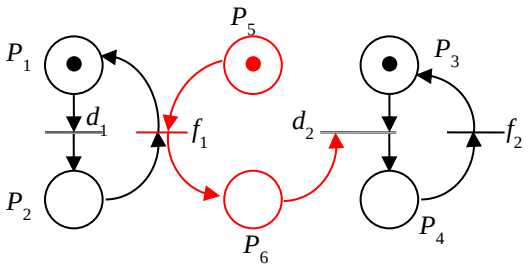


Figure IV.10 – RdPS du système en boucle fermé avec le traitement d'une seule pièce.

Cette figure montre qu'il est possible d'avoir des relations de couverture entre les états. Mais, le calcul du controleur est efficace et ne demande pas de modifier les modèles initiaux.

La condition de contrôlabilité obtenue est : $M_{s,k}(P_5) \geq M_{p,k}(P_2)$

L'hyperplan séparateur est : $M_{p,k}(P_2) - M_{s,k}(P_5) = 0$ et L^T= [0 1 0 0 -1 0]

$$W_C = -[0\ 1\ 0\ 0\ -1\ 0] \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \end{bmatrix} = [-1 \quad o \quad o \quad o]$$

$M_{c0} = 1$

La solution de contrôle obtenue optimale conforme à la théorie de commande par supervision.

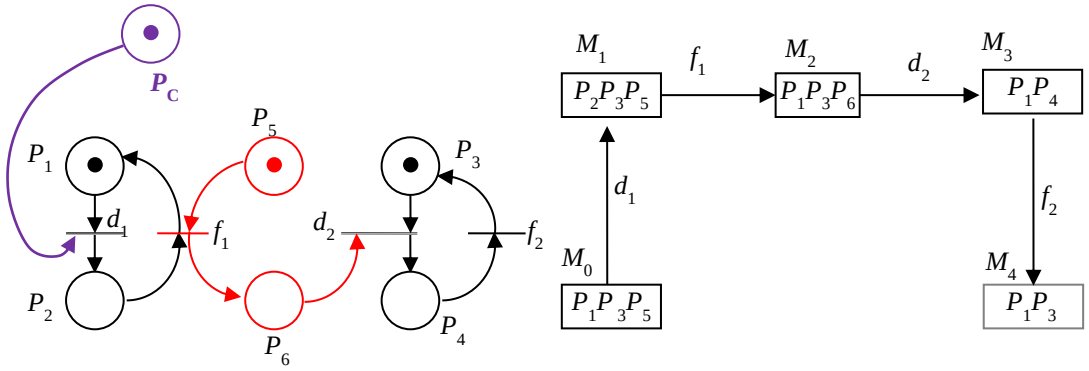


Figure IV.10 – RdPS contrôlé du système produisant une seule pièce et son graphe de marquage

▲

Exemple IV.5. Système de production constitué de deux ressources R_1 et R_2

Considérons le système de production constitué de deux ressources R_1 et R_2 fabriquant deux types de produits A et B donnée dans la figure IV. 4. Nous avons :

La condition de contrôlabilité obtenue est :

$$M_{s,k}(R_1) \geq M_{p,k}(P_1)$$
$$M_{s,k}(R_2) \geq M_{p,k}(P_3)$$

L'hyperplan séparateur est :

$$M_{p,k}(P_1) - M_{s,k}(R_1) = 0 \qquad \text{et} \quad L^{T} = \begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix}$$
$$M_{p,k}(P_3) - M_{s,k}(R_2) = 0$$

Nous pouvons calculer le contrôleur C.

$$W_C = -\begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & -1 & 1 & -1 & 1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 & -1 & 1 & 0 \\ -1 & 1 & 0 & -1 & 0 & 1 \end{bmatrix}$$

$$M_{c0} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

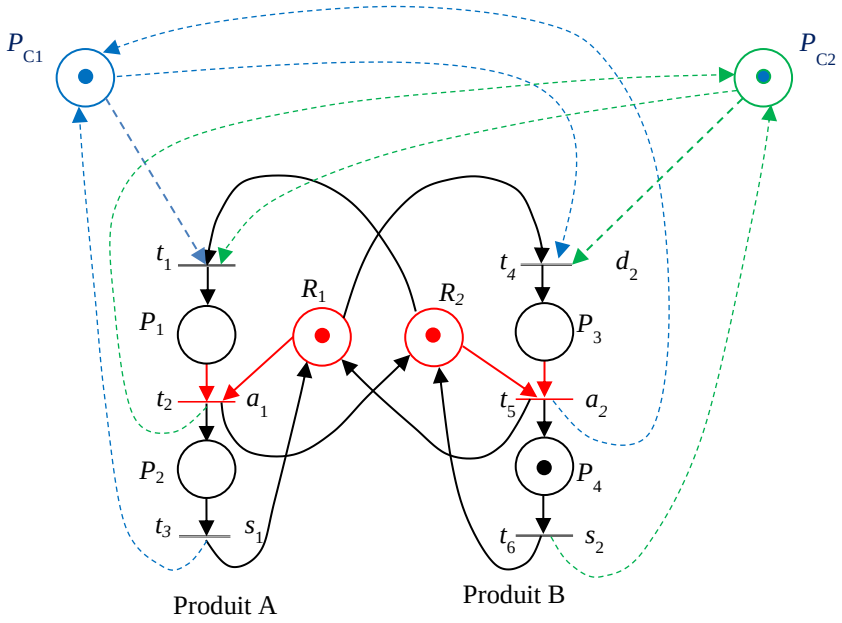


Figure IV.11 – RdPS du système de production contrôlé

▲

Exemple IV.6. (Dideban, 2007)

Considérons le système composé de deux machines et d'un robot (Figure IV.12). Chaque machine opère sur une seule pièce brute à la fois. Lorsque la machine a fini son travail (événement incontrôlable ftm_i), le robot décharge la machine et lorsqu'il termine le

déchargement de la machine (événement incontrôlable fdm_i), il transfère la pièce vers un stock. Après la fin du transfert (événement incontrôlable ftr_i), le robot revient à son état initial. Seul le début de tâche sur chaque machine (événement c_1 et c_2) est contrôlable. Les spécifications sont imposées par le robot.

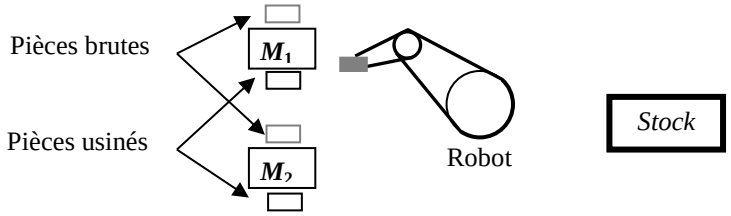


Figure IV.12. Système manufacturier composé de deux machines et d'un robot

Le modèle RdPS du système en boucle fermée est présenté dans la figure IV.13.

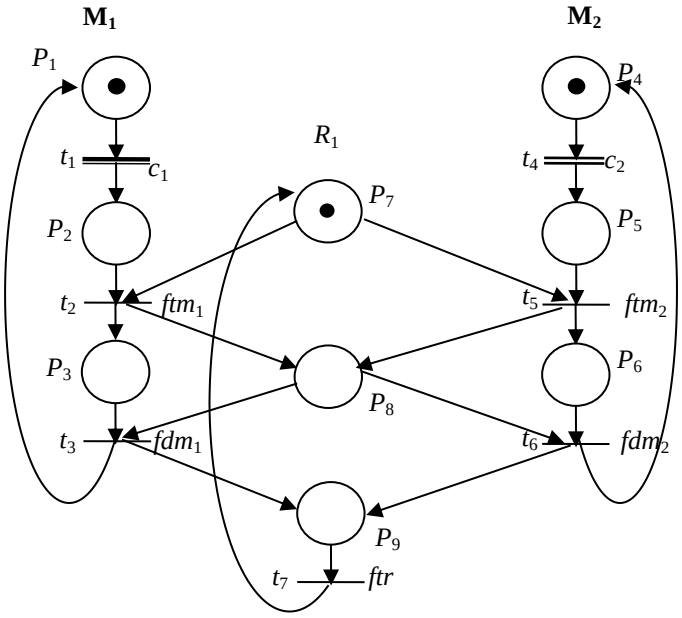


Figure. IV.13. RdPS du système manufacturier en boucle fermée

Les transitions synchrones incontrôlables sont : ftm_1, ftm_2, fdm_1 et fdm_2

La condition de contrôlabilité structurelle :

$$\begin{cases} M_{s,k}(P_7) \geq M_{p,k}(P_2) + M_{p,k}(P_5) \\ M_{s,k}(P_8) \geq M_{p,k}(P_3) + M_{p,k}(P_6) \end{cases}$$

L'hyperplan séparateur et vecteur contrainte :

$$\begin{cases} M_{p,k}(P_2)+M_{p,k}(P_5)-M_{s,k}(P_7)=0 \\ M_{p,k}(P_3)+M_{p,k}(P_6)-M_{s,k}(P_8)=0 \end{cases} \Leftrightarrow L^T = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & -1 & 0 \end{bmatrix}$$

Le calcul du contrôleur C par les invariants de marquage de place.

$$W_C = -\begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & -1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 1 & -1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & -1 \end{bmatrix}$$

$$= \begin{bmatrix} -1 & 0 & 0 & -1 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{C0} = 1$$

Le modèle RdP final est représenté dans la figure IV.14.

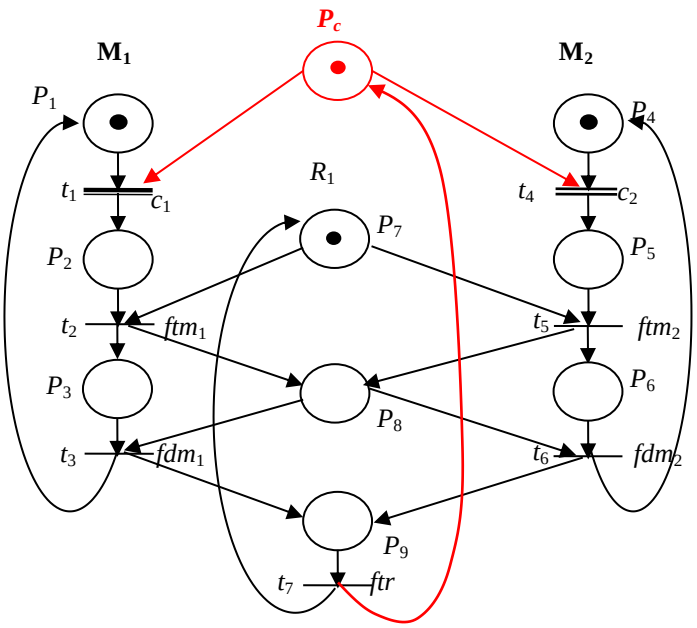


Figure IV. 14. Modèle RdPS du système manufacturier contrôlé

Le résultat obtenu est remarquable du point de vue structurel et l'ensemble des états interdits est identique à celui calculé formellement par l'algorithme de Kumar (1991), à savoir :

$\mathcal{M}_I = \{P_2P_5P_7, P_3P_5P_8, P_1P_5P_9, P_2P_6P_8, P_2P_4P_9, P_2P_5P_9\}$

De plus, ces états interdits sont dûs uniquement à la synchronisation avec les événements incontrôlables ftm_1 et ftm_2. Ce qui permet de comprendre que notre méthode tient compte de la simplification des contraintes par l'invariant partiel effectué par Dideban (2007) qui considéré que les places P7 et P8 ne peuvent être marqués en même temps.

▲

Dans ces exemples jugés explicites pour nous, les contrôleurs obtenus sont optimaux au sens maximal permissif, et l'ensemble des états accessibles sous contrôle $\mathcal{M}_C$ correspond à l'ensemble d'états admissibles $\mathcal{M}_A$; définit par la condition de contrôlabilité structurelle. Mais, cette solution doit être mise en évidence dans le cas où les RdPS sont non-borné, puisqu'il est difficile d'obtenir un contrôleur monolithique qui soit contrôlable pour des RdPS non-borné (Giua et DiCesare, 1994) ; et que les arcs partants du contrôleur (places de contrôle) peuvent être connectés aux transitions incontrôlables du RdPS du SED en boucle fermée.

IV.7.2. Limites du contrôleur en terme de synchronisation incontrôlables

Un contrôleur maximal permissif observe l'évolution du SED et peut à tout moment interdire le franchissement des transitions contrôlables de $N = N_p||N_s$. Mais, il lui est impossible d'influencer les transitions incontrôlables. En effet, la place de contrôle agit seulement pour bloquer la transition lorsque le franchissement des transitions, cause $M(P_C) < 0$. C'est-à-dire lorsque le franchissement de la transition viole directement la condition de contrôlabilité structurelle, $M_{s,k}(^\bullet t_j) \geq M_{p,k}(^\bullet t_j)$.

La synchronisation incontrôlable entre le contrôleur C et le RdPS $N = N_p||N_s$ peut générer à nouveau des états interdits qui violent les contraintes admissibles. Ceci arrive lorsqu'au moins un élément de la matrice W_C, lié à une transition incontrôlable est strictement positive (Moody & Antsaklis, 1998). Pour résoudre ce problème, une méthode intuitive consiste à remonter les branches jusqu'à trouver une transition contrôlable qui soit en aval de place de contrôle (Yamalidou et al., 1996). Une autre méthode consiste à modifier les contraintes admissibles qui doivent être apportées pour s'assurer qu'aucune place de contrôle ne soit une place d'entrée d'une transition incontrôlable (Moody & Antsaklis, 2000). Mais, ces techniques ne sont pas optimales (Giua and Seatzu, 2007), car un état admissible peut être interdit ou un état interdit peut être autorisé. Toutefois, quelques auteurs ont proposé des

algorithmes qui permettent de trouver le contrôleur optimal lorsqu'il y a synchronisation avec les transitions incontrôlables (Basile, 2006 ; Basile et *al*., 2007)

IV.8. Conclusion

Nous avons présenté dans ce chapitre une méthode intrinsèquement structurelle pour la synthèse des contrôleurs RdP pour résoudre les problèmes de commande par supervision, lorsque les spécifications sont de type états interdits. Cette méthode fourni structurellement les contraintes admissibles en présence de transitions incontrôlables facilite la procédure de synthèse de contrôleur par les invariants de place. Le contrôleur calculé est un ensemble de places de contrôle à intégrer au modèle RdPS du SED en boucle fermée, obtenu par produit synchrone. En outre, les états interdits générés par les synchronisations incontrôlables entre le procédé et la spécification peuvent à présent être prédits à l'avance par la condition contrôlabilité structurelle. Pareillement, les états admissibles peuvent à être prédits à l'avance. La méthode peut être appliquée à des SED réels (systèmes industriels) les avantages suivants:

- Le comportement du SED contrôlé est non-bloquant, puisque d'états admissibles $\mathcal{M}_{\mathbf{A}}$ est accessible et co-accessible et ne contredit pas les spécifications d'états interdits,
- Le comportement du SED contrôlé est maximal permissif, car le contrôleur assure qu'aucun état interdit ne sera atteint par interdiction du franchissement des transitions contrôlables qui mènent vers les états interdits.

Conclusion générale

Conclusion générale

Le travail proposé dans cette thèse est une contribution qui s'inscrit dans le cadre formel de la commande par supervision des systèmes à événements discrets (SED) tant au niveau de la contrôlabilité qu'au niveau de la synthèse de contrôleurs maximal permissif. La théorie de commande par supervision des SED introduite par Ramadge et Wonham (1983) utilise les langages et les automates à états finis pour synthétiser un contrôleur optimal au sens maximal permissif. Mais, son principal inconvénient est l'explosion combinatoire du nombre d'états lorsqu'on a affaire à des systèmes réels. Pour résoudre ce problème, des nombreux auteurs ont proposés des méthodes utilisant les réseaux de Petri (RdP) comme outil de modélisation. Mais, les méthodes les plus simples ne sont pas optimales, et les plus complexes sont difficiles à mettre en œuvre. Par ailleurs, ces méthodes nécessitent souvent la construction du graphe des marquages accessibles, qui peut être très grand pour des systèmes complexes. Néanmoins, la méthode basée sur les invariants de place des RdP est remarquable et fondamentale de synthèse de contrôleurs, car elle exploite les propriétés structurelles.

Dans notre travail nous avons considéré les RdP synchronisés (RdPS), qui offrent une meilleure structure pour définir les langages et les automates, avec des événements contrôlables et incontrôlables. Les modèles nécessaires à la commande par supervision sont le RdPS du procédé et RdPS de la spécification. La construction RdPS du SED en boucle fermée est effectuée synchronisation structurelle (produit synchrone) desdits modèles. A ce niveau, nous avons été confrontés au problème de synchronisation via les transitions incontrôlables pour garantir la contrôlabilité du RdPS du SED en boucle fermée et l'admissibilité des états du SED. Pour éviter la construction du graphe de marquages, nous avons choisi d'analyser de manière structurelle, l'admissibilité et la contrôlabilité des états du SED. Cette analyse nous a permis de définir la condition sur les marquages des places amont lors du franchissement des transitions synchrones incontrôlables. Ainsi, nous avons établir la condition de contrôlabilité structurelle, qui permet de déterminer les états admissibles et les états interdits sources d'incontrôlabilités. Formellement, ces deux ensembles d'états sont nécessaires et suffisants pour le calcul du contrôleur maximal permissif. C'est pourquoi nous avons prouvé que la condition de contrôlabilité structurelle est équivalente la condition de contrôlabilité définie sur les langages.

A partir de la condition de contrôlabilité structurelle nous avons déduit l'équation d'hyperplan séparateur, qui assure la séparation des états du graphe de marquages accessibles en états admissibles et états interdits. Il s'en suit naturellement que l'hyperplan caractérise les états admissibles critiques à partir desquels il est possible d'atteindre les états interdits via le franchissement d'une transition contrôlables.

Au regard de ce qui précède, la condition de contrôlabilité structurelle a été traduit sous forme de contraintes linéaires de type GMEC. Ces contraintes sont admissibles pour le calcul des contrôleurs maximal permissif par la méthode des invariants de place. Ceci constitue un lien biunivoque entre la théorie de commande par supervision et la méthode des invariants de place. Par conséquent, le rôle contrôleur obtenu sera d'interdire le franchissement d'une transition contrôlable lorsqu'un état admissible critique est atteint dans le comportement dynamique du RdPS du SED en boucle fermé.

Des exemples d'application explicites et classiques nous ont permis de faire une évaluation critique de notre approche vis-à-vis des autres travaux. Il ressort que notre approche intrinsèquement structurelle et efficace pour une implantation dans un SED réel ; puisqu'il ne nécessite pas la construction du graphe de marquages. De plus, le contrôleur obtenu est optimal et les arcs sortant du modèle du contrôleur sont connectés à une transition contrôlable, dont le franchissement peut être interdit à tout moment dans l'évolution du procédé. Ce qui justifie que les résultats soient compatibles avec la méthodologie de synthèse de contrôleurs par conjonction de la théorie de commande par supervision et la méthode des invariants, présenté au chapitre 3.

Les perspectives de ce travail consistent à vérifier si pour certaines SED très complexes (à grande échelle), la solution maximale permissive ne peut être obtenue directement et d'étendre l'applicabilité de notre approche à d'autres classes des RdP tels que : les RdP temporisés et les RdP hybrides. En effet, le RdPS proposé est essentiellement discret par nature et peut être limité lorsque le nombre d'états augmente. D'où l'intérêt du RdP hybride (RdPH), qui contient au moins une place (ou transition) discrète et au moins une place (ou transition) continue, pour généraliser notre approche. Une autre perspective de notre travail est de coupler notre approche au diagnostic des SED.

Références bibliographiques

Références bibliographiques

Achour, Z., Rezg, N. and Xie, X. (2004). Supervisory controller of Petri Nets under Partial Observation. *in* Proc. IFAC WODES04: 7th Workshop on Discrete Event Systems (Reims, France).

Alla, H., David, R., Di Mascolo, M. et J-L. Ferrier (2000). *Analyse et commande des systèmes à événements discrets*. Editions Hermès Science

Badouel, E., Darondeau, P. and Bernardinello, L. (1995). Polynomial algorithms for the synthesis of bounded nets. *in* "Proccedings Caap 95". Lecture Notes in Computer Science 915. Springer, pp. 364–378.

Basile, F. C. (2006). Suboptimal supervisory control of Petri nets in presence of uncontrollable transitions via monitor places. *Automatica*, 42(6), 995-1004.

Basile, F., Carbone, C. and Chiacchio, P. (2007). Feedback control logic for backward conflict free choice nets. *IEEE Trans. On Automatic Control* 52, 387–400

Boel, R., Ben-Naoum, L. and Breusegem, V. V. (1995). On forbidden state problems for a class of controlled Petri nets. *IEEE Transactions on Automatic Control*, 40(10), 1717-1731.

Charbonnier, F., Alla, H. and David, R.(1999). The Supervised control of discrete-Event dynamic Systems. *IEEE Transactions on control systems technology*, 7(2)

Cassandras, C.G. and Lafortune, S. (2008). *Introduction to Discrete Event Systems*. 2nd Edition, Springer - Verlag.

Dideban, A. and Alla, H. (2006). Solving the problem of forbidden states by feedback control logical synthesis. *in* "Proc. IEEE 32nd Annual Conference on Industrial Electronics (IECON 2006)". Paris, 348–353.

David, R. and Alla, H. (2010). *Discrete, Continuous, and Hybrid Petri Nets*. Springer – Verlag

Di Cesare, F., Harhalakis, G., Proth, J.M., Silva, M. and Vernadat, F.B (1993). Petri Net Synthesis for Discrete Event Control of Manufacturing Systems. Kluwer

Dideban, A. (2007). Synthèse de contrôleurs discrets par simplification de contraintes et de conditions. Thèse de doctorat , Université Joseph Fourier.

Dideban, A. and Alla, H. (2008). Reduction of constraints for controller synthesis based on safe Petri nets. *Automatica* 44(7), 1697–1706.

Dorsaf, E-B., Haddad, S. and Hennicker, R. (2012). Refinement and asynchronous composition of modal Petri Nets. *Lecture Notes in Computer Science LNCS 690.* K. Jensen, S. Donatelli, and J. Kleijn (Eds.), 96–120. Springer

Ezpeleta, J., Colom, J. and Martinez, J. (1995). A Petri net based deadlock prevention policy for flexible manufacturing systems. *IEEE Trans. Robot. Autom*. 11(2), 173–184.

Fabre, E., Benveniste, A., Haar, S. and Jard, C. (2005) Distributed monitoring of concurrent and asynchronous Systems. *Journal of Discrete Event Systems* 15(1), 33–84

Ferrier J-L., Boimond J-L. (2013). Systèmes dynamiques à événement discrets. Du modèle à la commande. Séminaire dans le cadre des JDA'99. hal.archives-ouvertes.fr/hal-00844691

Gaudin, B., Marchand, H. (2004). Supervisory control of product and hierarchical discrete event systems. *European Journal of Control* , 10(2), 131–145

Ghaffari, A., Rezg, N. and Xie, X. (2003). Design of a Live and Maximally Permissive Petri net Controller using the Theory of Regions. *IEEE Trans. Robot. Autom*, 19(1), 137–141.

Giua, A., DiCesare, F. and Silva, M. (1992). Generalized mutual exclusion constraints on nets with uncontrollable transitions. *in* "Proc. IEEE International Conference on Systems, Man, and Cybernetics". Chicago (IL), 2, 974–979.

Giua, A., and DiCesare, F. (1994). Blocking and Controllability of Petri Nets in Supervisory Control. *IEEE Transactions on Automatic Control*, 39(4), 818–823.

Giua, A. and Seatzu, C. (2007). A systems theory view of Petri nets. *in* C. Bonivento, L. Marconi, C. Rossi & A. Isidori, eds. "*Advances in Control Theory and Applications. Lecture Notes in Control and Information Sciences.* Springer, 353, 99–127.

Giua, A. (2013). Supervisory control of Petri nets with language specifications. *in* C. Seatzu, M. Silva, and J. van Schuppen (eds.), Control of discrete-event systems. Springer , 43(3), 235–255.

Giua, A. (2017). Automates et commande supervisée. Note de cours. Aix-Marseille Université, France

Godon, A. (1996). *Contribution à la commande des systèmes à événements discrets par réseaux de Petri.* Thèse de doctorat en Automatique Industrielle. Université d'Angers.

Gonza, M., Alla, H., Bitjoka, L. (2019).''Structural Method for Supervisory Control of Discrete Event within the framework of Petri Nets: a One-to-one Link between the Supervisory Control Theory and the Place Invariant Method", *International Journal of Systems Science: operations & logistics*, TSYB-1567860, https://www.tandfonline.com/doi.org/10.1080/ /23302674.2019.1567860

Gonza, M. and Bitjoka, L. (2018) "Supervisory Control of Systems Modeled by Petri Net from Adequate Admissible Constraints", *International Journal of Science and Research (IJSR*), Volume 7 Issue 1

Holloway, L. and Krogh, H. (1990). Synthesis of feedback control logic for a class of controlled Petri nets. *IEEE Trans. Automatic control*, 35(5), 514-523

Holloway, L., Guan, X. & Zhang, L. (1996). A generalization of state avoidance policies for controlled Petri nets. *IEEE Trans. Autom. Control* , 41(6), 804–816.

Hopcroft, J.E. and Ullman, J.D (1979). *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading

Hopcroft, J., Motwani, R. and Ullman, J. (2007). *Introduction to Automata Theory, Languages and Computation*. 3rd Edition, Prentice Hall

Iordache, M., Moody, J. and Antsaklis, P. (2001), A method for the synthesis of liveness enforcing supervisors in Petri nets. *in* 'Proc. IEEE American Control Conference (ACC 2001). Arlington (VA), 6, 4943–4948.

Iordache, M., Wu, P., Zhu, F., and Antsaklis, P. (2013). Efficient design of Petri-net supervisors with disjunctive specifications. *In* Proc. IEEE Int. Conf. on Automation Science and Engineering, Coming Soon. Madison, USA.

Komenda, J., van Schuppen, J.H., Gaudin, B., Marchand, H. (2008). Supervisory control of modular systems with global specification languages. Automatica 44(4), 1127–1134

Komenda, J. and van Schuppen, J.H. (2007). Conditions structurelles dans le contrôle modulaire des systèmes à évenements discrets concurrents. *In*: Proc. Modélisation des Systèmes Reactifs. Ecole Normale Superior de Lyon, Hermes, Lavoisier

Karp, R. M. and Miller. R. E. (1969). Parallel program schemata. *Journal of Computer and System Sciences*. 3(2), 147–195.

Karp, R. M., Miller. R. E., and Arnold, L.(1972). Rapid identification of repeated patterns in strings, trees and arrays. *In* Proceedings of the 4th Annual ACM, Symposium on Theory of Computing. Rosenberg, 125-136.

Kattan, B. (2004). Synthèse structurelle d'un contrôleur basée sur le Grafcet . Thèse de doctorat, UJF Grenoble, France.

Kumar, R. (1991). Supervisory Synthesis Techniques for Discrete Event Dynamical Systems. Thesis for the degree of Doctor of Philosophy, University of Texas

Kumar, R. and Holloway, L. (1996). Supervisory control of deterministic Petri nets with regular specification languages. *IEEE Trans. Autom. Control* 41(2), 245–249

Lafortune, S. and Yoo, H. (1990). Some Results on Petri net Languages. *IEEE Trans. On Automatic Control*. AC-35(4), 482–485

Lautenbach, K., and Ridder. H. (1996). The linear algebra of deadlock avoidance - a Petri net approach. *Technical Report 25-96*. Institut fUr Informatik, Universitat Koblenz-Landau

Li, Y. and Wonham, W. (1994). Control of vector discrete-event systems. II. Controller synthesis. *IEEE Trans. Autom. Control* 39(3), 512–531.

Li, Z. and Zhou, M. (2009). Deadlock Resolution in Automated Manufacturing Systems: A Novel Petri Net Approach. *Springer*

Lin, F. and Wonham, W. M. (1990). Decentralized control and coordination of discret event systems with partial observation. *IEEE Transactions of Automatic Control*, 35(12):1330-1337.

Luo, J. and Nonami, K. (2011). Approach for transforming linear constraints on Petri nets. IEEE Trans. On Automatic Control, 56, 2751–2765.

Luo, J. and Zhou, M. C (2017). Petri-net Controller Synthesis for Partially Controllable and Observable Discrete Event Systems. *IEEE Trans. on Automatic Control*, 62(3), pp. 1301 – 1313

Moody, J. and Antsaklis, P. (1998). Supervisory Control of Discrete Event Systems Using Petri Nets. *Kluwer Academic Publishers*, Boston.

Moody, J. and Antsaklis, P. (2000). Petri net supervisors for DES with uncontrollable and unobservable transitions. *IEEETrans. on Automatic Control*, 45, 462–476.

Murata, T. (1989). Petri nets: properties, analysis and applications. *Proceedings IEEE*, 77(4):541-580

Paola, C. Maria., Giua, A. and Seatzu. C.(2013). Structural Analysis of Petri Nets. *In* C. Seatzu et al. (Eds.): Control of Discrete-Event Systems, LNCIS. Springer-Verlag, 433, 213–233

Papadimitriou, C. (1986). The Theory of Database Concurrency Control. *Computer Science Press*, Rockville

Pocci, M., Demongodin, I., Giambiasi, N. and Giua, A. (2011). Synchronizing sequences on not strongly connected Petri nets. *In Symposium On Theory of Modelling and Simulation (DEVS/TMS'11)*, Boston, USA

Raisch, J. (2000). Discrete abstractions of continuous systems–an Input/Output point of view. *Mathematical and Computer Modelling of Dynamical Systems*. Special Issue on Discrete Event Models of Continuous Systems. 6, 6–29

Ramadge, P.J. (1983). Control and Supervision of Discrete Event Processes. Ph.D. Thesis, Dept. Electrical Engineering, Univ. Toronto.

Ramadge, P.J. and Wonham, W.M. (1983). Supervisory control of a class of discrete event processes. *Lecture Notes in Computer Science (LNCIS)*, Springer-Verlag, 63, 477-498.

Ramadge, P.J. and Wonham, W.M. (1986). Modular Supervisory Control of Discrete-Event Systems. Proc. 7th Int. Conf. Analysis and Optimization of Systems. 202–214

Ramadge, P.J. and Wonham, W.M. (1987). Supervisory Control of a Class of Discrete-Event Processes. *SIAM Jour. Control and Optimization*, 25(1), 206–230,

Ramadge, P. and Wonham, W. (1989). The control of discrete event systems. *In* Proc. *IEEE : Special Issue on Dynamics of Discrete Event Systems*. 77 (1). 81–98

Rezig, S. (2016). *Approches Canoniques pour la Synthèse des Contrôleurs Réseaux de Petri.* Thèse Doctorat PhD, Université de Lorraine

Rezig, S. Achour, Z. and Rezg, N. (2016). Control Synthesis Based On Theory of Regions with Minimal Reachability Graph Knowledge. *In 8th IFAC Conference on Manufacturing Modelling, Management and Control (MIM)*

Seatzu , C., Silva, M. J. and van Schuppen , H. (2012). Control of discrete-event systems, Automata and Petri net Perspectives. Lecture Notes in Control and Information Science, *Springer*, 433

Tajer, A., Philippot, A., Gellot, F. and Carré-Ménétrier, V. (2004). Démarche Formelle de synthèse d'une commande sûre à partir d'une spécification Grafcet. CIFA 2004, Douz, 22-24

Takai, S. and Ushio, T. (2003). Supervisor synthesis for a class of concurrent discrete event systems. *Proceedings of the 42nd IEEE Conference on Decision and Control*, IEEE, 2686-2691.

Ushio, T. (1990.). On The Existence of Finite State Supervisors in Discrete-Event Systems. *Proc. 29th IEEE Int. Conf. Decision and Control* . Hawaii, 2857–2860

Uzam, M. and Wonham, W. (2006). A hybrid approach to supervisory control of discrete event systems coupling RW supervisors to Petri nets. *Int. J. of Adv. Manuf.Technol*. 28(7 - 8), 747 – 760.

Uzam, M. (2010). On suboptimal supervisory control of Petri nets in the presence of uncontrollable transitions via monitor places. *Int. J. of Advanced Manufacturing Technology*, 47,567–579.

Vasiliu A., Dideban A., Alla, H. (2009). Control synthesis for manufacturing systems using non-safe Petri nets. *J. of Control Eng. and Applied Informatics*, 11(2), 43 –50.

Vasiliu, A. and Alla, H. (2011). Control optimality for ordinary Petri nets. *In* 18th IFAC World Congress. Milano, 18 (1), 3599 – 3604.

Vasiliu, A-I (2012). Synthèse de contrôleurs des systèmes à événements discrets basée sur les réseaux de Pétri. Thèse PhD, UJF Grenoble- France. HAL archives ouvertes

Wang, J.W., Wang d, H.F., Ding e, J.L., Furuta , K., Kanno , T., Ip, W.H. and Zhang, W.J. (2013). On domain modelling of the service system with its application to enterprise information systems. Taylor & Francis, http://www.tandfonline.com/loi/teis20

Wonham, W. M. (2003). Notes on control of discrete-event systems. *Technical report, no ECE 1636F/1637S, Department of ECE*, University of Toronto

Wonham W. M. (2005). Supervisory Control of Discrete-Event Systems, *Research report*. http://www.control.toronto.edu/people/profs/wonham/wonham.html/

Wonham, W.M. (2008). Supervisory Control of Discrete-Event Systems. University of Toronto

Wonham, W. (2011). Supervisory control of discrete event systems, *Technical report*, University of Toronto, Dept. of Electrical and Computer Engineering. http://www.control.utoronto.ca/~wonham/

Wonham, W.M. (2015). Supervisory Control of Discrete-Event Systems. *In* J. Baillieul, T. Samad (Eds.), Encyclopedia of Systems and Control. Springer.

Wonham, W. M. and Kai, C. (2017). Supervisory control of discrete-event systems. *Technical report, DOI: 0.13140/RG.2.2.14314.52169, Department of ECE*, University of Toronto.

Wu, L. & Zhou, M. (2004). Elementary siphons of Petri nets and their application to deadlock prevention in flexible manufacturing systems. *IEEE Trans. Syst.,* Man, Cybern. A , 34(1), 38–51.

Yamalidou, K., Moody, J., Lemmon, M. & Antsaklis, P. (1996). Feedback control of Petri nets based on place invariants. *Automatica* 32(1), 15–28.

Yin and S. Lafortune (2017). On the decidability and complexity of diagnosability for labeled Petri Nets. *IEEE Transactions on Automatic Control.*

Yoo, T.-S. and Lafortune, S. (2002). A general architecture for decentralized supervisory control of discrete-event systems. *Discrete Event Dynamics Systems*, 12, 335–377

Zhou, M. C. and DiCesare, F. (1993). Petri Net Synthesis for Discrete Event Control of Manufacturing Systems. *Kluwer Academic Publishing, Norwell, MA.K. Yamalidou and J. C. Kantor*. "Modeling and optimal control of discrete-event chemical processes using Petri nets. Comput. Chem. Eng., 15(7), 503–519

Zhou, C., Kumar, R. and Sreenivas, R.S (2008). Decentralized modular diagnosis of concurrent discrete event systems. *In*: 9th Workshop on Discrete Event Systems, Goteborg, Sweden