Contributions à l'analyse formelle et au diagnostic à partir de réseaux de Petri colorés avec l'accessibilité arrière

Mohamed Bouali

Université de Technologie de Compiègne **Département de Génie Informatique**

Contributions à l'analyse formelle et au diagnostic à partir de réseaux de Petri colorés avec l'accessibilité arrière

THÈSE

présentée et soutenue publiquement le 21 décembre 2009

pour l'obtention du

Doctorat de l'Université de Technologie de Compiègne – UTC

Option Technologie de l'Information et des Systèmes – TIS

par

Mohamed BOUALI

Composition du jury

Président :	Philippe BONNIFAIT	Professeur des Universités, Université de Technologie de Compiègne
Rapporteurs :	Jean-François AUBRY	Professeur des Universités, Institut National Polytechnique de Lorraine, Nancy
	Kamel BARKAOUI	Professeur des Universités, Conservatoire National des Arts et Métiers, Paris
Examinateurs :	Jean-Marc THIRIET	Professeur des Universités, Université Joseph Fourier, Grenoble
	Walter SCHÖN	Professeur des Universités, Université de Technologie de Compiègne (directeur de thèse)
	Pavol BARGER	Maître de Conférences, Université de Technologie de Compiègne (directeur de thèse)

HEUristiques et DIAgnostic des SYstèmes Complexes — UMR 6599

Je dédie ce travail à :
Mes parents,
Mon petit frère.

Remerciements

Je tiens à remercier, en premier lieu, mes directeurs de thèse : Messieurs Pavol BARGER et Walter SCHÖN. Ils m'ont invité à une grande aventure qu'ils ont conduit d'une main de maître en instaurant un climat d'amitié et un environnement fertile au travail intellectuel. Ce travail est aussi le leur.

Un grand merci aux rapporteurs, Messieurs Kamel BARKAOUI et Jean-François AUBRY qui ont accepté de plancher sur ce travail. Ils ont su tirer sa qualité vers le haut à travers leurs remarques et leurs conseils avisés.

Messieurs Jean-Marc THIRIET et Philippe BONNIFAIT ont bien voulu examiner cette thèse. Je les remercie beaucoup pour cet effort qu'ils m'ont consacré.

Ma thèse s'est déroulée au sein du laboratoire HeuDiaSyC qui est une unité de recherche mixte UTC/CNRS. À ce titre, Je remercie chaleureusement toutes les instances et les personnes qui ont contribué de près ou de loin à l'élaboration de ce travail.

Une pensée particulière va vers ma famille dont l'éloignement est très pesant mais tout aussi motivant. Leur sacrifices m'ont permis d'arriver là où je suis et leur fierté est un carburant infini d'avancement et de dépassement de soi.

Une mention spéciale pour mes amis avec qui j'ai partagé mes joies et mes peines et qui ont su être présents et trouver les mots dans les moments les plus difficiles.

Un grand merci pour mes collègues sapeur pompiers du Centre de Secours Principal de Compiègne qui ont donné un visage et un sens à mon engagement citoyen en dehors du cadre de ma thèse.

Table des matières

Table des figures

Introduction générale

Depuis quelques années, la part des systèmes embarqués dans le marché de l'informatique et de l'électronique ne cesse de grandir. Ces systèmes omniprésents, se retrouvent dans toutes les tâches et toutes les activités comme les communications, les transports ou encore les tâches ménagères. Les exemples de ces systèmes ne manquent pas. Ceci va du téléphone mobile à la voiture "intelligente" et de l'aspirateur autonome au système de guidage des missiles. Si on prend l'exemple des téléphones mobiles, un constat rapide s'impose de lui même : leur complexité a explosé en quelques années. En effet, ce qui n'était au début qu'un appareil lourd et encombrant dont le seul but était de communiquer tout en gardant sa mobilité a évolué en une plaquette agréable et légère incluant une centrale multimédia, un agenda électronique, des programmes de bureautique, des terminaux pour différents réseaux (wifi, bluetooth, infra rouge, 3G), une console de jeux, etc. Cette profusion de fonctionnalités est rendue nécessaire par la concurrence acharnée que se livrent les fabricants. Il ne suffit pas de proposer une nouvelle fonctionnalité, mais il faut le proposer le premier. Ce phénomène pose donc deux contraintes fortes dans la conception de ces systèmes : 1) les délais de mise sur le marché doivent être les plus courts possibles ; et 2) le produit proposé doit contenir la même chose que le concurrent sinon plus. Tout ceci obéit à des contraintes strictes de tarif et de sûreté de fonctionnement.

La notion de *système embarqué* reste un concept général dont la définition n'et pas aisée à cerner. D'un point de vue linguistique, le terme *"système"* se réfère à un assemblage d'éléments formant un ensemble cohérent à but commun. Le terme *"embarqué"* se réfère au fait que le dispositif matériel embarque (contient) le dispositif logiciel[1]. Les méthodes et outils de conception des systèmes embarqués évoluent en parallèle avec l'évolution des systèmes pour pouvoir prendre en compte les exigences grandissantes. Elles doivent conjuguer avec une complexité du matériel, un enfouissement efficace d'un gros volume de logiciel et des délais de conception réduits. Le système ainsi conçu doit satisfaire en plus des exigences strictes de sûreté de fonctionnement. En effet, même pour les produits d'utilisation courante, les exigences en sûreté de fonctionnement sont très rigoureuses.

[1] Voir le chapitre 1 pour une définition détaillée des *systèmes embarqués*

L'exemple de téléphone mobile reste illustratif : une application de divertissement ne doit pas interférer avec un appel en cours ; ce dernier ne doit pas émettre trop d'ondes électromagnétiques et doit consommer le minimum d'énergie ; la batterie qui doit avoir la plus grande autonomie possible ne doit pas surchauffer ; l'ensemble doit polluer le moins possible ; et tout ceci n'est qu'une vue très synthétique des contraintes.

Dans ce contexte économique, l'un des défis de la recherche actuellement est de développer des techniques pour prendre en compte les exigences de sûreté de fonctionnement dans les systèmes embarqués. Ces techniques sont à intégrer directement dans les processus de conception. Le besoin est réel et immédiat car les méthodes classiques dans ce domaine (arbres de défaillances, diagramme de fiabilité, ...) atteignent vite leurs limites face aux nouvelles exigences. La modélisation basée sur les systèmes à événements discrets apparaît comme un moyen efficace pour atteindre cet objectif. En effet, les modèles sont par définition des abstractions des systèmes et des processus. Ils décrivent le contenu du système, la hiérarchisation des composants et l'ordre précis des événements dans le processus. D'un autre côté, la plupart des modèles se prêtent bien à l'analyse, et comme ils sont établis très tôt dans le processus de conception, ils sont utilisés pour vérifier l'adéquation des spécifications avec la structure et le comportement du système.

À partir de ce constat, un certain nombre de travaux ([MS07] et [Por93], [Kha03] pour ne citer que quelques uns) ont exploité les systèmes à événements discrets, et notamment les réseaux de Petri, pour effectuer des analyses efficaces de diagnostic et de sûreté de fonctionnement sur différents types de systèmes embarqués. L'intérêt n'est pas uniquement d'analyser le système mais aussi de le faire dans un cadre formel grâce à des outils mathématiques comme la logique linéaire. Nous avons entrepris, dans ce travail, d'étendre cette approche et de la généraliser. Le choix du modèle s'est porté sur les réseaux de Petri colorés (RdPC). Ce sont des extensions des réseaux de Petri ordinaires intégrant une sémantique de différentiation des jetons. Ce choix est motivé par la grande expressivité du modèle, ses fondements mathématiques solides permettant l'analyse formelle, la facilité de modélisation des différents aspects des systèmes embarqués (tels que les traitements parallèles et la communication) et l'expertise existante au sein de l'équipe de recherche. Le modèle RdPC se prête bien à une analyse dite *avant*, c'est-à-dire, en connaissant les causes on détermine les conséquences. Cet état de fait pose un certain nombre de défis dans le domaine des systèmes embarqués. D'abord, l'extension de l'expressivité amenée par les RdPC, et notamment les expressions des arcs, complique considérablement l'analyse du modèle comparativement aux RdP ordinaires. D'autre part, le diagnostic des défaillances se trouve difficile car, en diagnostic, les conséquences sont connues (un état défaillant est généralement signalé par une alarme sonore, visuelle, déviation par rapport à un profil, ...) et on

souhaite en connaître les causes. Comme, dans le domaine des systèmes embarqués, les défaillances sont des événements très rares, l'analyse avant s'en trouve inapplicable.

Ce travail exploite une vision duale à l'analyse avant appelée analyse par accessibilité arrière. Elle est plus adaptée au diagnostic et a pour objectif d'offrir un outil d'aide à la conception de systèmes embarqués sûrs de fonctionnement. Le principe est de transformer (inverser) le modèle pour baser l'analyse sur les conséquences (souvent, les défaillances). Cette inversion s'effectue grâce à des transformations structurelles. Les analyses sont ensuite conduites sur les modèles inversés en définissant des mécanismes complémentaires comme l'enrichissement de marquage généralisé aux cas RdPC. Les aspects algorithmiques sont consolidés par deux études théoriques : la première pour l'aspect local (algorithmes d'inversion) et la seconde pour l'aspect global (conduite d'analyse par accessibilité arrière)

Les deux premiers chapitres de ce manuscrit fournissent un contexte global à ce travail. Le chapitre 1 reprend des généralités sur les systèmes embarqués comme les définitions, le cycle de vie ainsi qu'une revue des tendances actuelles de R&D. Le second introduit les problématiques de la sûreté de fonctionnement (SdF) telles que les besoins en SdF et la difficulté de conception sûre de fonctionnement. Le chapitre 3 reprend la modélisation des systèmes embarqués. Il débute par une présentation de l'outil de modélisation choisi, à savoir, les RdPC. Vient ensuite une présentation synthétique des différentes méthodes d'analyse : empirique, formelle, analyse directe par modèle. Comme ce travail est axé sur l'inversion des modèles RdPC, un récapitulatif de l'état de l'art dans ce domaine est détaillé.

Les contributions de cette thèse (algorithmique et théorique) sont contenues dans les chapitres 4 et 5. Le premier détaille la démarche algorithmique en présentant les transformations permettant l'inversion des modèles ainsi que quelques problèmes inhérents à l'inversion des modèles. Ceci est suivi par les mécanismes complémentaires nécessaires à la mise en œuvre d'une analyse par accessibilité arrière et donne un échantillon de pistes et d'idées non menées à terme par défaut de temps. L'inversion de modèle est étudiée dans le chapitre 5 d'un point de vue théorique sur deux aspects. Le premier concerne les algorithmes d'inversion (vue locale) prouvés grâce à l'algèbre linéaire. Le second concerne la réalisation de l'analyse par accessibilité arrière (vue globale) prouvée grâce à la logique linéaire.

Les chapitres 6 et 7 appliquent la méthode d'analyse proposée à des exemples illustratifs. Ces exemples sont académiques dans le chapitre 6. Ils sont inspirés de la littérature et donnent un aperçu du domaine et de la manière d'application de l'accessibilité arrière pour le diagnostic d'un système modélisé. Le cas applicatif du chapitre 7 est tirée d'un système industriel : le système de freinage d'un tramway. Les analyses par accessibilité arrière sont appliquées d'une manière incrémentale pour aboutir, à la fin, à une étude avec un profil de vitesse.

Chapitre 1

Systèmes embarqués

Sommaire

Les systèmes embarqués sont présents dans les activités quotidiennes de chacun. Ils apportent du confort et contribuent à l'amélioration de la qualité de vie. Leur évolution a conduit à une augmentation croissante du nombre et de la complexités des fonctionnalités qu'ils doivent réaliser ce qui s'est traduit par une complexité croissante des architectures matérielles et logicielles de ces systèmes. Ceci a induit une évolution continue des méthodes et outils de conception. Le but de ces méthodes/outils n'est pas seulement de concevoir les architectures adéquates et sûres de fonctionnement, mais de le faire en minimisant les temps de conception ainsi que les coûts associés.

Le but de ce chapitre est de présenter brièvement les systèmes embarqués d'un point de vue conception. Pour cela, les systèmes embarqués sont définis en synthétisant différentes définitions de la littérature puis en mettant l'accent sur les

architectures et les nombreuses caractéristiques qu'ils doivent satisfaire. Ensuite, les différents modèles de cycle de vie des systèmes embarqués sont passés en revue. Ces modèles évoluent parallèlement aux processus de conception et à la complexité des systèmes. Ils traduisent des besoins de nouvelles méthodes de conception et d'augmentation du niveau d'exigences. Quelques unes des tendances actuelles de la recherche et du développement sont présentées. Enfin, le chapitre se termine avec une sélection de certains problèmes d'actualité dans le domaine des systèmes embarqués.

1.1 Introduction aux systèmes embarqués

1.1.1 Définition d'un système embarqué

Définir un système embarqué n'est pas chose aisée tellement ils sont répandus et variés. D'ailleurs, la littérature regorge de définitions sur le sujet. Certaines mettent l'accent sur le caractère fonctionnel, d'autres sur le caractère structurel. D'autres encore, définissent les systèmes embarqués en mettant en exergue les différences avec les PC (*Personal Computer*). C'est pourquoi, une synthèse de ces définitions est proposée dans la suite.

Selon [Hea97], un système embarqué est un système basé sur microprocesseur construit pour gérer une fonction ou une série de fonctions et qui n'est pas conçu pour être programmé par l'utilisateur final comme c'est le cas dans un PC. En effet, l'utilisateur peut faire des choix concernant la fonctionnalité mais ne peut pas changer la fonctionnalité. Un système embarqué est donc conçu pour exécuter une tâche particulière bien que paramétrable.

Selon [HS07], les systèmes embarqués sont des composants qui intègrent du logiciel et du matériel. Ils se caractérisent par une interaction continue avec leur environnement physique. Ils trouvent leur application dans de nombreux domaines comme les transports, les télécommunications, la distribution d'énergie ainsi que les produits électriques et électroniques.

Selon [Bar07],un système embarqué est un système informatique à but précis conçu pour accomplir une ou quelques fonctions dédiées, souvent avec des contraintes de calcul temps réel. Il est couramment embarqué comme une partie d'un appareil complet incluant les parties mécaniques et hardware.

Selon [Sad07], un système embarqué est un système électronique, piloté par un logiciel, qui est complètement intégré au système qu'il contrôle. Un système embarqué peut aussi être défini comme un système électronique soumis à diverses contraintes. Il combine généralement diverses technologies qui relèvent des domaines de la mécanique, de l'hydraulique, de la thermique, de l'électronique et des technologies de l'information.

Dans ces définitions, quelques points communs se dégagent autour de la notion de *système embarqué*. C'est un dispositif contraint (par la taille, le temps de réaction, l'environnement d'exploitation, ...), dédié à une (ou des) fonctionnalité(s) précise(s) et intégrant le logiciel qui le commande au sein de la partie matérielle.

Historiquement, le développement des systèmes embarqués fut très rapide. Il aura fallu moins de vingt ans pour passer du premier *vrai* système embarqué[2] au premier système d'exploitation commercial dédié aux systèmes embarqués[3], et moins de trente ans pour que le domaine de recherche étudiant les systèmes embarqués soit doté de sa propre revue[4] et de sa propre conférence organisée en 1989 à San Francisco (USA).

1.1.2 Architecture d'un système embarqué

Il existe autant d'architectures différentes que de définitions de systèmes embarqués. Deux familles d'architectures se distinguent dans la littérature : en blocs ou en couches. Les architectures en bloc considèrent le système embarqué comme une collection de parties programmables entourées de circuits dédiés (ASIC : *Application Specific Integrated Circuit*) et d'autres composants standards (ASSP : *Application Specific Standard Parts*) qui interagissent avec l'environnement à travers des capteurs et des actionneurs [Zur06]. La collection peut être un ensemble de puces sur une carte ou un ensemble de modules dans un circuit intégré. Les éléments programmables sont principalement les microprocesseurs et les DSP (*Digital Signal Processor*). Cette architecture, dont un exemple est illustré dans Fig.1.1, donne une vue structurelle du système. Dans cet exemple, l'architecture est centrée autour d'un CPU (*Central Processing Unit*) recevant en entrée des données des capteurs numérisées par un CAN (*Convertisseur Analogique Numérique*) et délivrant en sortie des commandes aux actionneurs via un CNA (*Convertisseur Numérique Analogique*). En interne, différents sous systèmes coopèrent pour accomplir la fonctionnalité assignée au système : le logiciel, l'interface utilisateur, les processeurs auxiliaires, etc.

L'architecture en couches est une abstraction du système qui ne montre donc pas les détails d'implémentation ou de conception des circuits. Cette représentation met l'accent sur les niveaux de communication et de circulation d'information à la manière des protocoles réseaux [Noe05]. Elle peut être enrichie et détaillée selon les besoins spécifiques avec l'incorporation, dans le schéma, de normes par

[2]Le premier système qui peut être considéré comme embarqué est le système de guidage utilisé dans les missions Apollo, développé par Charles Stark Draper au laboratoire d'instrumentation du MIT.

[3]En 1980, la société Hunter&Ready réalisa le premier système d'exploitation commercial pour systèmes embarqués nommé VRTX (*Versatile Real-Time Executive*).

[4]Première parution en 1988 de la revue *Embedded Systems Programming*

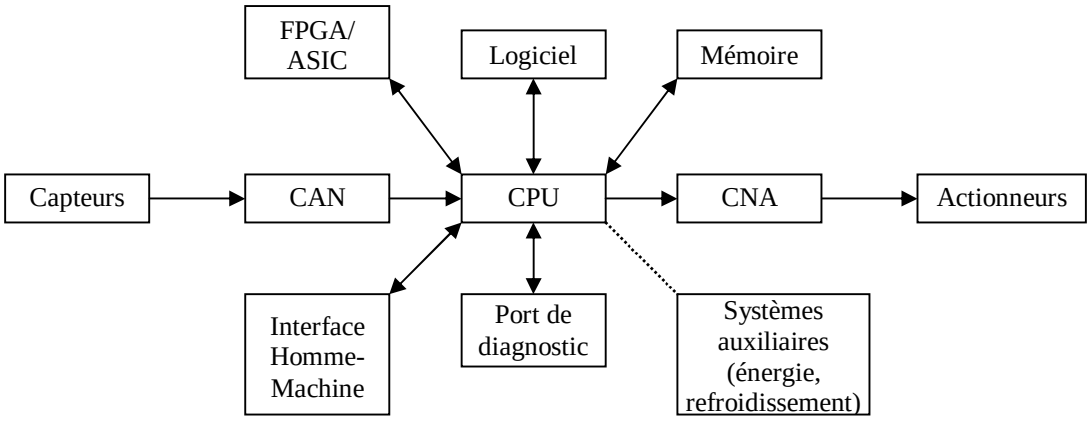


FIG. 1.1 – Exemple d'architecture en blocs d'un système embarqué

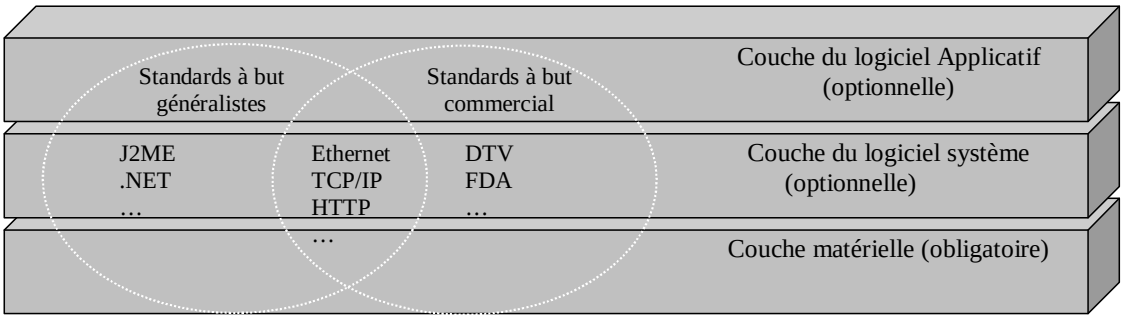


FIG. 1.2 – Exemple d'architecture en couches d'un système embarqué

exemple. Dans l'illustration Fig.1.2, le système est composé d'une couche obligatoire (la couche matérielle) et de deux couches optionnelles (logiciels système et application). Le schéma est enrichi avec différents protocoles, tels que les protocoles réseaux (HTTP, TCP/IP), nécessaires au fonctionnement du système qu'il représente.

1.1.3 Caractéristiques des systèmes embarqués

Les systèmes embarqués sont des systèmes multi-contraints. Ils doivent répondre à des contraintes souvent combinées selon le besoin, l'utilisation et l'environnement. Les exigences à satisfaire sont non seulement des exigences fonctionnelles concernant l'exactitude du calcul mais également des exigences extra-fonctionnelles (appelées aussi non-fonctionnelles). Ces dernières portent sur l'utilisation optimale des ressources (temps de calcul, mémoire, énergie) ainsi que sur l'autonomie, la réactivité et la robustesse du système. [HS07].

L'importance et le développement des aspects liés aux exigences a abouti à l'émergence de *l'ingénierie des exigences*. Elle est définie dans [SS97] comme une discipline dont le but est de découvrir et de documenter un ensemble d'exigences. Le terme *ingénierie* indique l'utilisation de techniques répétables et systématiques pour s'assurer que les exigences d'un système sont complètes, consistantes et pertinentes. Les exigences se divisent en deux catégories : exigences fonctionnelles et exigences extra-fonctionnelles. Les exigences fonctionnelles sont définies comme les fonctions qu'un système ou un composant doit accomplir[5]. Elles peuvent être documentées de plusieurs manières. Les plus communes sont des descriptions textuelles et des *use cases*. Ces derniers sont décrits par du texte, des listes énumérées, des diagrammes, etc. Les exigences extra-fonctionnelles sont toutes autres exigences que les exigences fonctionnelles. Plusieurs taxonomies existent pour classifier les exigences extra-fonctionnelles comme par exemple ISO9126[6], IEEE Std 830[7] et la taxonomie VOLERE [RR06].

Criticité : Le degré de criticité d'un système ou d'une fonction dépend des conséquences des déviations par rapport à un comportement nominal. Ces conséquences peuvent concerner les dangers encourus par les personnes et les biens, la capacité d'accomplissement des missions ou encore la rentabilité économique.

Réactivité : Les systèmes embarqués doivent interagir avec leur environnement. Leur temps de réaction doit être en rapport avec les événements sur lesquels ils agissent. C'est ce qui est appelé *temps réel*. L'expression *temps réel* ne signifie pas une réponse rapide mais une réponse dans des délais bornés ou raisonnables. Une réponse donnée trop tard équivaut à une réponse erronée. On peut distinguer deux types de systèmes temps réel : système temps réel dur (doit respecter des limites temporelles dans le pire cas d'exécution) ; système temps réel mou (doit respecter des limites temporelles pour une partie plus au moins importante de ses exécutions).

Autonomie : Cette caractéristique est essentielle pour les systèmes embarqués qui doivent remplir leur mission pendant de longues périodes sans intervention humaine, particulièrement quand l'intervention humaine est impossible ou trop lente comme dans une application spatiale, dans un environnement radioactif ou dans des zones de catastrophes naturelles.

[5]Définition tirée de *IEEE standard glossary of software engineering terminologie*. IEEE Std 610.12-1990

[6]ISO 9126 : Software engineering. 2000

[7]*Ieee recommended practice for software requirements specification*, 1998. Révision de IEEE Std 830-1998.

Sûreté de fonctionnement : L'environnement des systèmes embarqués est souvent hostile pour des raisons physiques (comme les chocs, les variations de température ou la radioactivité) ou humaines (comme les malveillances ou les erreurs de manipulation). C'est pour cela qu'un ensemble de moyens est imposé pour la gestion des risques. Cet ensemble de moyen est communément appelé sûreté de fonctionnement (voir chapitre 2). D'où la connexion des problématiques de la SdF aux systèmes embarqués.

Consommation d'énergie : La forte intégration des unités de calcul dans les systèmes embarqués impose la prise en compte de la consommation d'énergie, principalement pour deux raisons. D'abord, les sources d'énergie sont souvent limitées mais doivent assurer l'alimentation du système pendant une longue période (exemples : téléphones mobiles, réseaux de capteurs). La seconde raison est le problème de la dissipation d'énergie : une forte dissipation entraînant une augmentation de température, ce qui pourrait remettre en cause le fonctionnement du système embarqué.

Encombrement physique : Les systèmes embarqués possèdent des caractéristiques physiques (dimensions, poids) qu'on cherche souvent à minimiser pour des raisons fonctionnelles et/ou commerciales. Cet objectif est possible grâce à la miniaturisation des unités de calcul, du développement de matériaux légers (ex. fibres de carbone), d'une meilleure conception et intégration des parties mécaniques.

Adéquation Algorithme Architecture : Le logiciel embarqué est souvent soumis aux fortes contraintes de coût, de consommation d'énergie, d'encombrement physique et de sûreté de fonctionnement. Il doit fonctionner de manière optimale avec le matériel qu'il utilise afin de réaliser au mieux la tâche requise tout en limitant les ressources utilisées. La conception d'un logiciel embarqué est un compromis entre un logiciel dédié (donc optimal) mais coûteux et un logiciel générique moins coûteux mais moins efficace.

Coût : Ce critère est l'un des plus importants. Il désigne la valeur financière allouée qui conditionne le cycle de vie du système embarqué tout entier. Il influe sur les choix de conception (matériel, logiciel, méthodes, outils), de mise en œuvre, de commercialisation, de maintenance, d'amélioration et de fin de vie.

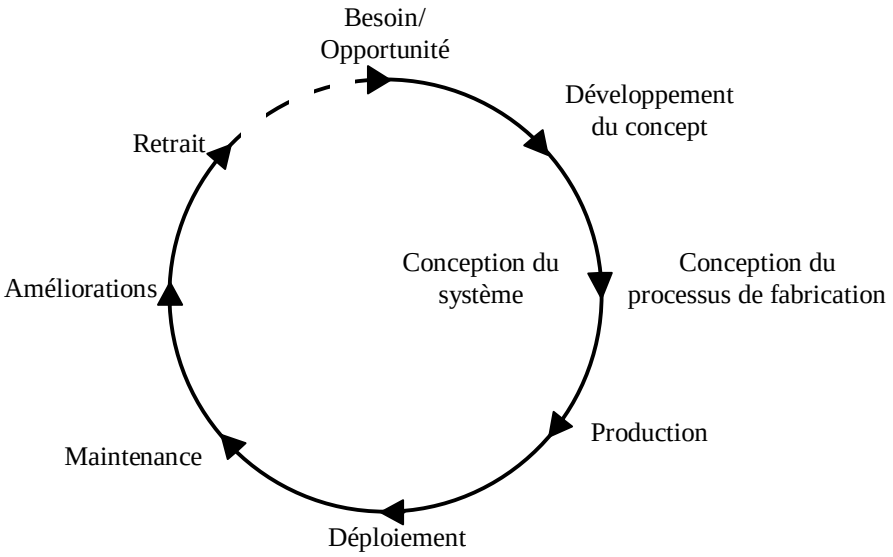


FIG. 1.3 – Le cycle de vie générique d'un système embarqué

1.2 Cycle de vie d'un système embarqué

Le *cycle de vie* désigne toutes les étapes du développement d'un système, de sa conception à sa disparition (voir Fig.1.3). L'objectif d'un tel découpage est de permettre de définir des jalons intermédiaires permettant la validation du développement du système, i.e. la conformité du système avec les spécifications exprimées, et la vérification du processus de développement, i.e. l'adéquation des méthodes mises en œuvre. L'origine de ce découpage provient du constat que les erreurs ont un coût d'autant plus élevé qu'elles sont détectées tardivement dans le processus de réalisation. Le cycle de vie tente de détecter les erreurs au plus tôt et ainsi de maîtriser la qualité, les délais de réalisation et les coûts associés. Différents formalismes de cycle de vie existent. Ce qui suit présente les plus utilisés : Modèle en cascade, cycle en V et cycle en spirale.

1.2.1 Modèle en cascade

Le modèle en cascade [Roy70] est hérité de l'industrie du bâtiment et travaux publics (BTP). Ce modèle repose sur les deux hypothèses suivantes : 1) la toiture ne peut pas être construite avant les fondations et 2) les conséquences d'une modification en amont du cycle ont un impact majeur sur les coûts en aval. Les phases traditionnelles de développement sont effectuées les unes après les autres, avec un retour possible sur les précédentes, voire au tout début du cycle (voir Fig.1.4).

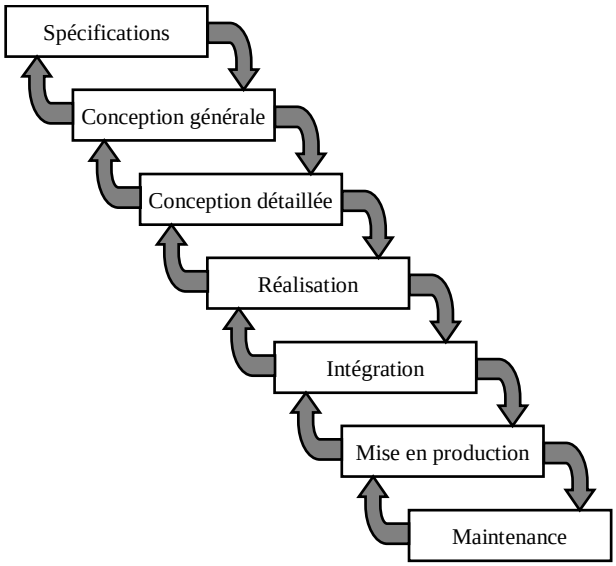


FIG. 1.4 – Les phases du Modèle en cascade

1.2.2 Cycle en V

Le modèle du cycle en V [Rip84] a été imaginé pour pallier le problème de réactivité du modèle en cascade. Ce modèle est une amélioration du modèle en cascade qui permet en cas d'anomalie, de limiter un retour aux étapes précédentes. Il contient deux parties : montante et descendante (voir Fig.1.5). Les phases de la partie montante, doivent renvoyer de l'information sur les phases en vis-à-vis lorsque des défauts sont détectés afin d'améliorer la conception. De plus le cycle en V met en évidence la nécessité d'anticiper et de préparer dans les étapes descendantes les besoins des futures étapes montantes : ainsi les besoins des tests de validation sont définis lors des spécifications, les besoins des tests unitaires sont définis lors de la conception, etc. Le cycle en V est devenu un standard de l'industrie depuis les années 1980.

1.2.3 Cycle en spirale

Le développement reprend les différentes étapes du cycle en V. Par la mise en œuvre de versions successives, le cycle recommence en proposant un produit de plus en plus complet (voir Fig.1.6). Le cycle en spirale [Boe88] met plus l'accent sur la gestion des risques que le cycle en V. En effet, le début de chaque itération comprend une phase d'analyse des risques. Ceci est rendu nécessaire par le fait que, lors d'un développement cyclique, il y a plus de risques de devoir défaire à l'itération N ce qu'on a fait à l'itération $N - X$.

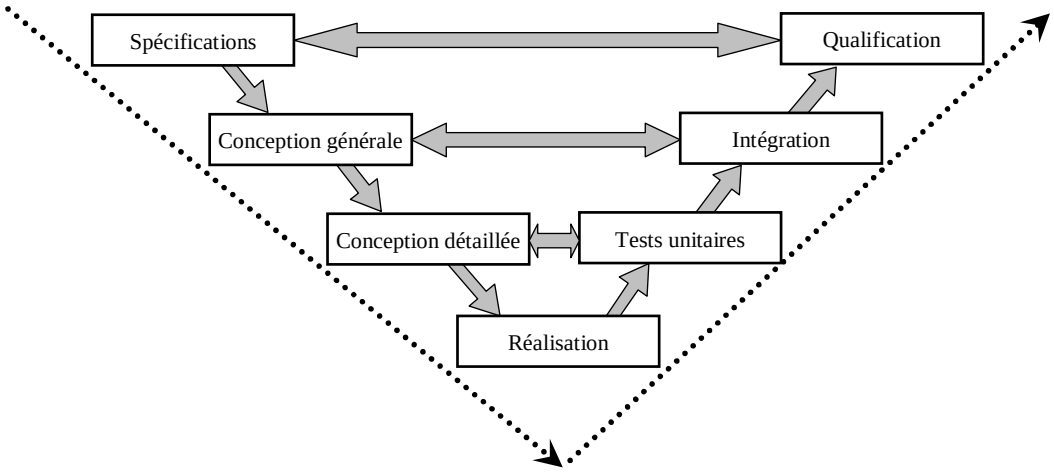


FIG. 1.5 – Les phases du cycle en V

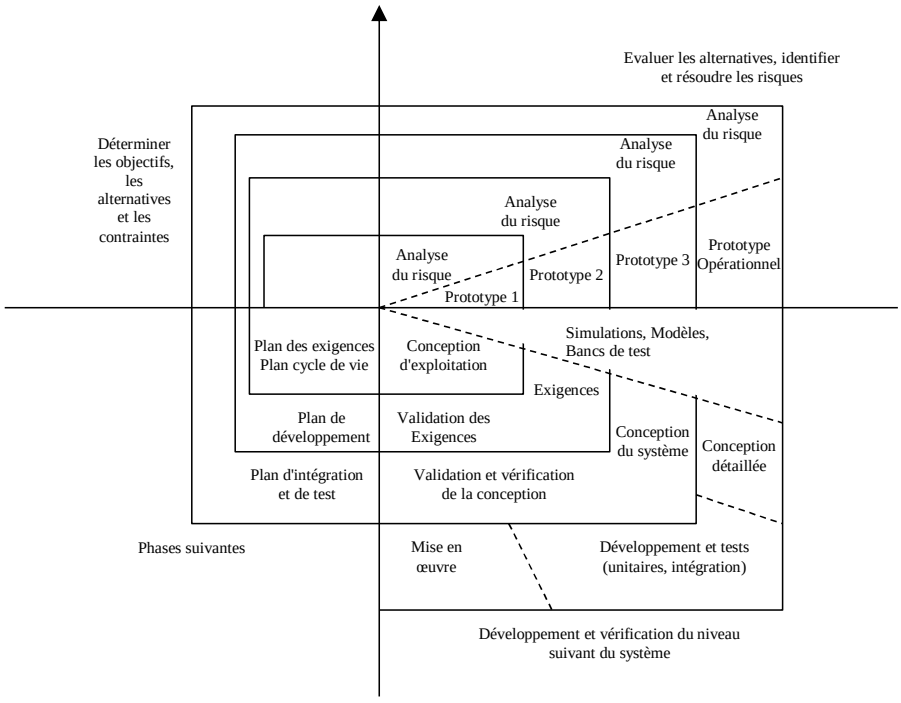


FIG. 1.6 – Les phases du cycle en spirale [Boe88]

1.3 Tendances dans les R&D

1.3.1 Conception matérielle

Dans le passé, les systèmes embarqués ont été conçus avec des approches ad hoc basées sur l'expérience acquise sur des produits similaires. Les processus de conception requièrent souvent beaucoup d'itérations car le système n'est pas spécifié d'une manière rigoureuse et non-ambiguë. Cette méthode a montré ses limites, spécialement en termes de vérification et d'efficacité avec la complexification considérable qu'ont connue les systèmes embarqués. La conception passait par une séquence de deux étapes : *abstraction* et *clusterisation*. L'abstraction est la description d'un objet avec un modèle où certains détails de bas niveaux sont ignorés (ex. l'expression booléenne dans une porte logique). La clusterisation est la connexion d'un ensemble de modèles de même niveau d'abstraction pour avoir de nouveaux objets qui montrent de nouvelles propriétés. Celles-ci n'apparaissent pas dans les modèles de base pris séparément. Grâce à l'application successive de ces deux étapes (*abstraction/clusterisation*), la conception est partie des dessins d'agencement, aux schémas de transistors, aux réseaux de portes logiques, aux descriptions RTL (*Register Transfert Level*) [Zur06].

La notion de *plateforme* est une clé pour l'utilisation de l'abstraction et clusterisation. Une plateforme est un modèle abstrait unique qui cache les détails de différentes implémentations possibles. La plateforme, comme par exemple une famille de microprocesseurs, périphériques et bus, permet aux développeurs de concevoir le plus haut niveau (généralement appelé application) sans se soucier des détails bas niveaux (ex : processeur pipeline). En même temps, ce mode de travail permet aux développeurs de plateformes de partager leurs conceptions et de réduire les coûts et les temps de développement.

Nous assistons de nos jours à l'apparition de nouvelles abstractions de plus haut niveau comme réponse à l'augmentation de la complexité des circuits intégrés. Les objets peuvent être des descriptions fonctionnelles de procédés complexes ou des spécifications de plateformes hardware complètes. Ils font appel à des modèles formels de haut niveau qui peuvent être utilisés pour effectuer une validation rapide au plus tôt du système final même avec un ensemble de détails réduit.

1.3.2 Conception fonctionnelle

D'une manière générale, cette étape détermine la distribution des fonctionnalités sur l'architecture physique. En d'autres termes, il faut déterminer quelles fonctions sont exécutées sur quelle partie de la plateforme. Cette étape requiert une attention particulière des compromis entre la complexité algorithmique, la flexibilité fonctionnelle et les coûts de mise en œuvre. Des outils d'aide à la conception

existent pour faciliter ces étapes. Ils sont subdivisés en deux familles. La première est représentée par exemple par Simulink[8], MATRIXx[9], Ascet-SD[10] [LBBZ97], SPD[11], SCADE[12] et SystemStudio[13]. Elle inclut des éditeurs de hiérarchies de blocs et des bibliothèques que le concepteur utilise pour composer des traitements de signaux orientés données et des systèmes de commandes embarqués. Les bibliothèques contiennent de simples blocs (additionneurs, multiplexeurs) et des blocs plus complexes (filtres, etc.). La deuxième famille contient, par exemple Tau[14], StateMate[15] [Val92], Esterel Studio[16], StateFlow[17]. Elle est orientée commande dans les systèmes embarqués. Dans ce cas, l'attention se focalise sur la réponse du système à son environnement et les entrées utilisateur, au lieu de se focaliser sur les calculs numériques.

1.3.3 Les profils UML

UML (*Unified Modeling Language*) [BRJ00], tel que standardisé par OMG[18] (*Object Management Group*) est une classe à lui tout seul. À l'origine, UML se focalisait sur des logiciels à but généraliste (pour entreprises par exemple) et non sur des logiciels embarqués temps réel. La spécification UML 2 [RJB04] peut être étendu par des profils pour la prise en charge de la conception des systèmes embarqués et/ou temps réel. Nous aborderons deux d'entre eux : SysML et MARTE.

Le profil SysML

SysML[19] est un langage de modélisation graphique développé par l'OMG, INCOSE[20] (*International Council on Systems Engineering*) et recherchant la conformité avec l'AP233[21]. Son objectif est de permettre à des concepteurs de disciplines variées de collaborer autour d'un modèle commun pour définir un système. Ce modèle doit donc regrouper les spécifications, les contraintes, et les paramètres de l'ensemble du système à partir de multiple vues de ce système. Ce besoin de modèle commun est issu du constat que la conception de système donne souvent lieu

[8] http ://www.mathworks.com/products/simulink/
[9] http ://www.ni.com/matrixx/
[10] http ://www.etas.com/en/products/ascet_software_products.php
[11] http ://www.coware.com/products/
[12] http ://www.esterel-technologies.com
[13] http ://www.synopsys.com/TOOLS/SLD/DIGITALSIGNALPROCESSING/Pages/SystemStudio.aspx
[14] www.telelogic.com/Products/tau/
[15] http ://www.telelogic.com/products/statemate/
[16] http ://www.esterel-technologies.com/
[17] www.mathworks.com/products/stateflow/
[18] http ://www.omg.org/
[19] http ://www.omgsysml.org/
[20] http ://www.incose.org/
[21] http ://www.ap233.org/

à une accumulation de documentations qui doivent toutes être croisées et mises à jour pour maintenir la cohérence et respecter les spécifications du système. SysML permet de définir les blocs qui deviendront des parties mécaniques, électroniques, informatiques, etc.

Le profil MARTE

MARTE (*Modeling and Analysis of Real-Time and Embedded Systems*)[22] est un profil qui ajoute à UML des capacités de développement basées sur des modèles de systèmes temps réel et de systèmes embarqués. MARTE fournit un support pour les étapes de spécification, conception et vérification/validation. Il est appelé à remplacer le profil SPT (*Schedulability, Performance and Time*). Conceptuellement, MARTE est structuré autour de deux axes : le premier modélise les caractéristiques des systèmes temps réel et systèmes embarqués ; le second étiquète les modèles pour une analyse des propriétés du système. Ces deux parties majeures sont associées à d'autres modules tels que la description du temps et l'utilisation concurrente des ressources.

1.3.4 Ingénierie système

L'Ingénierie Système (IS) a été définie pour prendre en compte la complexité et les contraintes croissantes des système, ainsi que les évolutions des besoins au cours de leur développement. Selon la norme IEEE 1220 [IEE99], l'Ingénierie Système est une approche interdisciplinaire et concerne la capacité de réussir la réalisation des systèmes. Pour cela, elle propose de spécifier et ordonnancer les activités à mettre en œuvre lors du développement d'un nouveau système [Dav09]. Le processus global de l'IS est composé de deux processus : le processus de management et le processus technique. Le rôle du processus de management est de suivre la tenue des délais et coûts du projet ainsi que d'organiser la coordination entre les équipes et les parties prenantes. Le processus technique est appliqué pour spécifier, concevoir et valider le système développé.

Plusieurs standards d'IS ont été définis pour apporter des atouts tels que la compétitivité des études, leur contractualisation, ou encore la maîtrise de la complexité du système. Les trois standards les plus utilisés actuellement sont : EIA 632 [EIA98], IEEE 1220 [**?**] et ISO/IEC 15288 [ISO03]. [Dav09] rapporte la différentiation des trois standards selon leurs propos :

- L'ISO/IEC 15288 établit un référentiel commun pour décrire le cycle de vie des systèmes.
- L'EIA 632 propose un ensemble intégré de processus aidant le développeur dans la conception d'un système.

[22] http ://www.omgmarte.org/

– L'IEEE 1220 fournit un standard focalisé sur le développement du système.

1.4 Activité de vérification et de validation

Le développement perpétuel des systèmes embarqués pose des défis intéressants du point de vue recherche et incite à la production et l'innovation scientifiques. Ces défis peuvent être classés en fonction du domaine particulier dans lequel ils sont posés : architectures du matériel, architectures du logiciel, conception conjointe et méthodes formelles. Nous nous concentrons dans ce contexte sur les aspects vérification et validation (V&V) qui sont des point pivots dans la conception des systèmes embarqués.

La vérification et la validation d'un système forment un processus qui s'étale tout au long du développement. Le terme vérification et validation englobe toutes les activités permettant de s'assurer que le logiciel correspond bien à son cahier des charges d'une part et que le cahier des charges répond bien aux besoins de l'utilisateur d'autre part. L'activité de V&V est donc présente tout au long du cycle de vie.

Concrètement, on ne cherche pas vraiment à distinguer entre les activités liées à la vérification de celles liées à la validation. Leur but commun est d'atteindre au final un système fiable. Néanmoins, les distinctions suivantes peuvent être établies : la vérification s'assure que le système est conforme à sa spécification. Elle répond donc à la question construisons-nous correctement le produit ? La validation s'assure que le système implémenté correspond aux attentes du futur utilisateur. Elle permet de démontrer donc que les exigences initiales ont bien été prises en compte. La validation répond donc à la question : construisons-nous le bon produit ?

Pour satisfaire les objectifs de la V&V, il faut avoir recours à des techniques d'analyse et de vérification statiques et dynamiques. Les techniques statiques sont consacrées à l'analyse des différentes représentations du système, comme le cahier des besoins, les modèles et documents de conception ou les programmes. Ces techniques sont appliquées à tous les niveaux du processus de développement. Les techniques dynamiques quant à elles imposent l'exécution du système pour observer son comportement. L'une des techniques de V&V, à laquelle nous nous intéressons dans le cadre de ce travail, est l'analyse. Deux types d'analyses peuvent être distingués : des analyses de cohérence des exigences et des analyses pour la vérification de la satisfaction des exigences

⋆ Analyse de la cohérence des exigences : la majorité des travaux sur ce sujet proposent la traduction des exigences dans un langage formel ou de contraintes ayant une sémantique bien définie.

⋆ Analyse de satisfaction des exigences : elle consiste en un ensemble d'analyses qui peuvent être réalisées sur un modèle afin de vérifier qu'il satisfasse

certaines propriétés ou exigences. Ces analyses se basent souvent sur des techniques telles que la preuve formelle et le model checking. Ces analyses nécessitent souvent d'avoir le modèle à vérifier sous un formalisme qui dépend de la technique d'analyse à appliquer (système états transitions, langage formel tel que B ou Z, ...). De la même façon les propriétés à vérifier doivent elles aussi être exprimées sous un formalisme adapté au modèle à vérifier.

Chapitre 2

Sûreté de fonctionnement : Concepts et méthodes

Sommaire

Comme vu dans le chapitre 1, et aussi selon [Sif09], les systèmes embarqués doivent répondre à des exigences d'encombrement, de consommation d'énergie, mais surtout de sûreté de fonctionnement. En effet, de par leur champ d'utilisation croissant, la confiance placée dans ces systèmes est, a priori, également croissante. Selon le cadre d'utilisation, l'accent est mis sur une (ou des) propriété(s) bien

spécifique(s), comme par exemple, le temps de réponse, la capacité à se protéger contre les malveillances ou l'aptitude à éviter les défaillances potentiellement catastrophiques pour l'environnement du système. L'intérêt de la sûreté de fonctionnement (souvent notée SdF) est le fait qu'elle rassemble ces exigences dans un cadre conceptuel unique incluant des propriétés telles que la fiabilité, la disponibilité, la sécurité et l'intégrité [LAB+96].

Ce chapitre revient sur le concept de sûreté de fonctionnement. Pour l'élaborer, nous nous sommes largement imprégnés des définitions de [LAB+96] et [Vil88]. Les définitions de base sont accompagnées des principales démarches et méthodes de la SdF et d'une rapide revue des dates clés de son histoire illustrant l'évolution des besoins et des techniques dans ce domaine. Un dernier point est consacré aux difficultés liées à la conception des systèmes sûrs de fonctionnement.

2.1 Introduction à sûreté de fonctionnement

2.1.1 Définitions

La sûreté de fonctionnement, au sens large, est la science des défaillances. Elle inclut leur connaissance, leur évaluation, leur prévision, leur mesure et leur maîtrise. Au sens strict, c'est l'aptitude d'une entité à satisfaire à une ou plusieurs fonctions requises dans des conditions données [Vil88], [CEI98].

Selon [LAB+96], la SdF est la propriété d'un système permettant à ses utilisateurs de placer une confiance justifiée dans le service délivré. Cette définition est reprise par le groupe de travail *IFIP 10.4 Working Group on Dependable Computing and Fault Tolerance*.

Selon l'étude [CM01], la sûreté de fonctionnement est un ensemble de moyens (démarches, méthodes, outils, vocabulaire) imposé par la *maîtrise des risques*. Elle est définie dans le souci de prendre des décisions optimales et justifiées face à des événements tels que les erreurs et les défaillances qui imposent le recours aux techniques d'identification, d'évaluation ou de réduction de risques.

2.1.2 Attributs, moyens et entraves de la sûreté de fonctionnement

La discipline de la sûreté de fonctionnement couvre un large spectre d'activités. Celles-ci sont regroupées par [LAB+96] en trois classes : *attributs*, *moyens* et *entraves*.

Les attributs de la SdF sont communément notés FDMSS (fiabilité, disponibilité, maintenabilité, sécurité-innocuité, sécurité-confidentialité). Cette notation est la traduction de la notation anglaise RAMSS (*Reliability*, *Availability*, *Main-*

tainability, *Safety*, *Security*). Les attributs de la SdF permettent d'exprimer les propriétés attendues des systèmes et d'apprécier la qualité du service délivré. Les attributs de la SdF sont définis comme suit :

La fiabilité : Aptitude d'une entité à accomplir les fonctions requises, dans des conditions données, pendant une durée donnée. Elle est généralement mesurée par la probabilité qu'une entité accomplisse une fonction requise, dans les conditions données, pendant l'intervalle de temps $[0, t]$.

La disponibilité : Aptitude d'une entité à être en état d'accomplir les fonctions requises dans des conditions données et à un instant donné. Elle est généralement mesurée par la probabilité qu'une entité soit en état d'accomplir une fonction requise dans des conditions données et à un instant t donné.

La maintenabilité : Aptitude d'une entité aux réparations et aux évolutions. Elle est généralement mesurée par la probabilité que la maintenance d'une entité accomplie dans des conditions données, avec des procédures et des moyens prescrits, soit achevée au temps t, sachant que l'entité est défaillante à l'instant $t = 0$.

La sécurité-innocuité : Aptitude à la non occurrence de conséquences catastrophiques pour l'environnement.

La sécurité-confidentialité : L'aptitude à la non occurrence de divulgation non autorisées d'informations conduit à la *confidentialité*. L'aptitude à la non occurrence d'altérations inappropriées d'informations conduit à l'*intégrité*. L'association de la confidentialité, de l'intégrité et de la disponibilité vis-à-vis des actions autorisées conduit à la *sécurité-confidentialité*.

Les entraves à la SdF sont les circonstances indésirables de la non sûreté de fonctionnement (la confiance ne peut plus être placée dans le service rendu). Les entraves sont au nombre de trois : fautes, erreurs et défaillances.

Défaillance : Elle survient lorsque le système dévie de son fonctionnement nominal.

Erreur : La partie de l'état du système susceptible d'entraîner une défaillance.

Faute : La cause supposée ou adjugée d'une erreur.

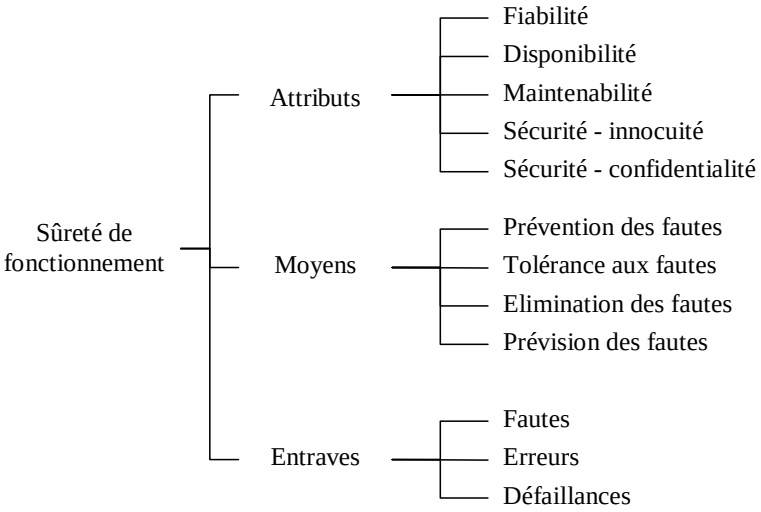


FIG. 2.1 – Arbre de la sûreté de fonctionnement

Les moyens de la SdF sont des techniques permettant de fournir au système l'aptitude à délivrer un service conforme à son fonctionnement nominal et de donner une confiance dans cette aptitude. Les moyens sont au nombre de quatre : prévention des fautes, tolérance aux fautes, élimination des fautes et prévision des fautes.

L'arbre de Fig.2.1 résume le découpage en classes de la discipline de la sûreté de fonctionnement.

2.2 Démarches et méthodes d'une approche SdF

Les démarches et méthodes de la SdF sont diverses et variées. Elles peuvent être classifiées selon deux critères. Le premier est selon le couple cause/conséquence : inductive (causes vers conséquences) et déductive (conséquences vers causes). Le second est selon le type du résultat fourni : qualitatif ou quantitatif. La littérature rapporte des démarches et méthodes comme l'analyse fonctionnelle, les allocations de sûreté de fonctionnement, l'AMDE (Analyse des Modes de Défaillances et de leurs Effets), l'AMDEC (Analyse des Modes de Défaillances et de leurs Effets et de leurs Criticités), les blocs diagramme de fiabilité, les calculs prévisionnels de fiabilité, les démonstrations de fiabilité opérationnelle, les études de maintenabilité, les calculs de disponibilité, les réseaux de Petri (sur lesquels se concentre cette étude), les graphes de Markov, les Méthodes des Espaces d'Etat, l'APR (Analyse Préliminaire des Risques), les arbres de défaillances, le traitement de retour d'expérience, etc. Ce qui suit détaille les méthodes et démarches les plus souvent citées.

2.2.1 Retour d'expérience

Le principe du retour d'expérience est d'améliorer la connaissance du système par l'observation, le recueil, l'analyse, le traitement des informations relatives au fonctionnement réel du système, à son environnement, à son impact sur son environnement [Ver01]. Améliorer sa connaissance, c'est peut être la compléter, l'enrichir, la rectifier, mais c'est certainement réduire les incertitudes et augmenter la confiance sur la connaissance que l'on a du système.

Le retour d'expérience est considéré généralement à tort, comme consistant uniquement à remplir des bases de données avec les caractéristiques des événements redoutés qui se produisent au fur et à mesure de leurs survenues. En fait, une démarche de retour d'expérience utile est orientée vers la réduction des incertitudes influentes dans l'analyse des risques. Elle joue un rôle à l'occasion de décisions dans lesquelles des analyses de risques sont prises en compte. En effet, l'analyse de risques s'appuie sur la connaissance du système, de son environnement, de son utilisation, etc. Il existe souvent de grandes incertitudes dans cette connaissance. Le retour d'expérience a pour but principal de réduire ces incertitudes [CM01].

Un retour d'expérience peut être organisé autour de deux approches : une approche à dominante qualitative et une approche à dominante quantitative. L'approche à dominante qualitative recueille et analyse chacune des occurrences des événements significatifs. Le retour d'expérience a alors pour fonction première de s'assurer que ces événements se sont réalisés selon les scénarios prévus ou alors de repérer un scénario (possible puisque réalisé) qui n'était pas imaginé ou que l'on croyait avoir rendu impossible.

L'approche à dominante quantitative recueille et comptabilise les événements ou les conditions dont la fréquence prévisionnelle a été utilisée dans les évaluations de sûreté de fonctionnement. L'accumulation de ces statistiques fournit des données de fiabilité issues directement de la réalité ; leur précision croît avec le temps et le nombre d'observations et elles permettent de réviser les données utilisées dans les évaluations. À l'aide de techniques mathématiques, le meilleur parti de ces données de retour d'expérience peut être tiré afin d'avoir le meilleur couple précision/confiance sur les données de fiabilité qui servent aux évaluations.

2.2.2 Analyse préliminaire de risques

L'analyse préliminaire de risques (APR) a pour but de produire une liste d'événements redoutés qui doivent être étudiés. Elle cherche : 1) à faire un tour aussi complet que possible des événements redoutés (fautes, erreurs, défaillances) ; et 2) à évaluer la gravité et les conséquences liées aux situations dangereuses et aux accidents potentiels. Une APR se fonde sur l'identification des situations que peut

Fonction (composant)	Mode de défaillance	Cause	Conséquences (effets)	Moyens de détection	Probabilité (P)	Gravité (G)	Criticité (P x G)	Actions

FIG. 2.2 – Structure d'un tableau AMDEC

connaître le système pour isoler les situations dangereuses. Elle à l'avantage de permettre un examen rapide des situations dangereuses, d'être économique en terme de temps passé et de ne pas nécessiter un niveau de description détaillé du système. Par contre, elle ne permet pas de décrire finement les enchaînements qui conduisent à un événement redouté. C'est pour cela que la démarche APR est généralement une première étape nécessitant la réalisation d'études complémentaires de sûreté de fonctionnement telles que l'AMDEC ou la méthode des arbres de défaillances utiles à la détermination des causes des événements indésirables décelés lors de cette analyse préliminaire.

2.2.3 Analyse des modes de défaillance, de leurs effets et de leurs criticités

L'AMDEC (Analyse des modes de défaillance, de leurs effets et de leurs criticités) est une extension de l'AMDE (Analyse des modes de défaillance, de leurs effets). Elle est extrêmement répandue dans tout ce qui a trait à la sécurité ou la disponibilité. Elle se présente sous forme d'un tableau résumant toutes les informations collectées (voir Fig.2.2). Son principe se résume dans les étapes suivantes :

- ⋆ Décomposer le système en éléments connus. Le niveau de détail de la décomposition se détermine sur le niveau auquel on peut associer des modes de défaillance et des fréquences si possible.
- ⋆ Associer à chaque élément ses modes de défaillance. Chaque mode de défaillance est un comportement autre que le fonctionnement nominal.
- ⋆ Identifier les effets sur le système de chaque mode de défaillance de chaque élément.
- ⋆ Associer à chacun des modes de défaillance de chaque élément sa criticité. Cette étape distingue l'AMDEC de l'AMDE.

Malgré sa popularité et son utilisation dans de nombreux domaines, l'AMDEC possède des inconvénients. Le plus grand est son incapacité à prendre en compte des défaillances multiples (systèmes redondants), dynamiques ou dépendantes [Fau04]. L'AMDEC considère uniquement les défaillances simples, c'est pourquoi elle est souvent associée à la méthode des arbres de défaillances qui est plus adaptée à l'analyse des systèmes redondants et donc aux défaillances multiples.

2.2.4 Arbres de défaillances

Un arbre de défaillances ou arbre de causes (Fig.2.3) modélise l'ensemble des combinaisons d'événements et de conditions qui peuvent aboutir à l'événement au sommet de l'arbre. Il utilise tous les connecteurs logiques ET, OU inclusif et exclusif ainsi que leurs combinaisons. Comme les liens sont des fonctions logiques, la probabilité de l'événement *sommet* se déduit, par des expressions booléennes, des probabilités des événements *feuilles*.

La construction d'un arbre de défaillances est une démarche déductive qui consiste à sélectionner un événement dont les scénarios qui y aboutissent doivent être représentés. Puis, en utilisant des connecteurs logiques, les événements (ou combinaisons d'événements) et les conditions produisant le sommet sont représentés. Puis, cette opération est réitérée (sur chaque sous arbre). La démarche, sur chaque sous arbre s'arrête dès qu'on arrive à des événements élémentaires. L'arbre de défaillances permet ainsi d'établir la liste des combinaisons d'événements et de conditions suffisantes pour provoquer l'événement étudié. Une telle combinaison est appelée une *coupe*. L'intérêt est porté en priorité sur les coupes minimales : la suppression d'un seul élément d'une coupe minimale le rend insuffisant pour provoquer l'événement étudié. En d'autres termes, une coupe minimale est un ensemble nécessaire et suffisant d'événements et de conditions pour provoquer l'événement étudié.

Bien que l'arbre de défaillances soit une méthode efficace de représentation synthétique des scénarios conduisant à un événement redouté, il présente des limites. L'une de ces limites est que l'ordre d'occurrence des événements menant vers l'état redouté n'est pas pris en compte. Dans cette méthode, il est également difficile de représenter des aspects temporels.

2.2.5 Arbres d'événements

À l'inverse de l'arbre de défaillances, l'analyse par arbre d'événements est une démarche inductive. À partir d'un événement (une défaillance) d'origine, cette analyse permet d'estimer la dérive du système en envisageant de manière systématique le fonctionnement ou la défaillance des dispositifs de détection, d'alarme, de prévention, de protection, etc. La construction de l'arbre consiste alors à partir de l'événement origine et à envisager soit le bon fonctionnement soit la défaillance de la première fonction de sécurité. L'événement initiateur est représenté schématiquement par un trait horizontal. Le moment où doit survenir la première fonction de sécurité est représenté par un nœud. La branche supérieure correspond généralement au succès de la fonction de sécurité, la branche inférieure à la défaillance de cette fonction. Le schéma (Fig.2.4) qui se lit souvent de gauche (une entrée unique : l'événement origine) vers la droite (les événements conséquences).

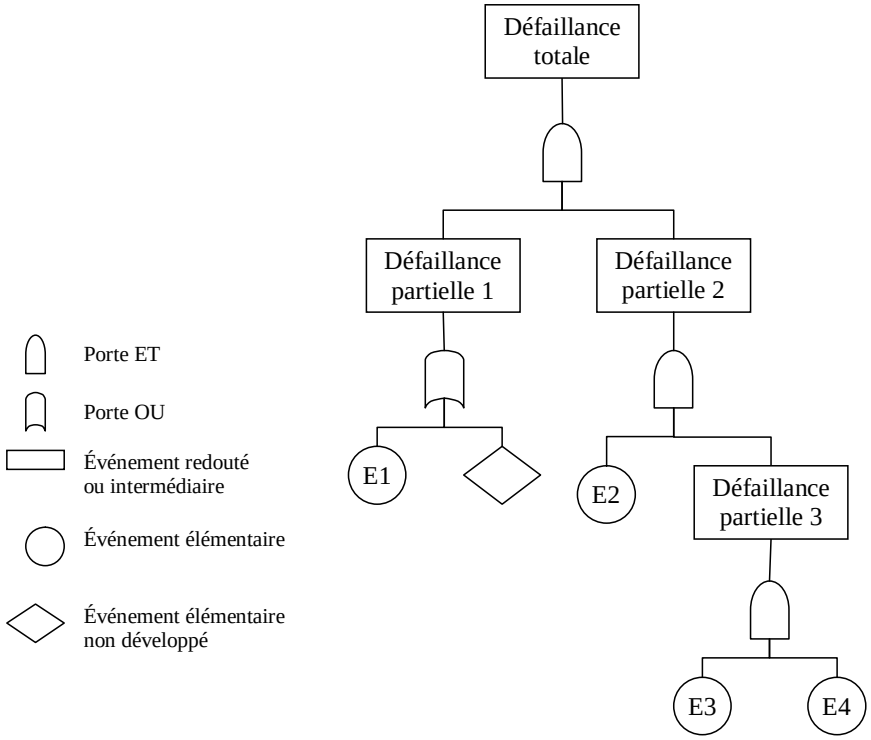


FIG. 2.3 – Exemple de structure d'un arbre de défaillance

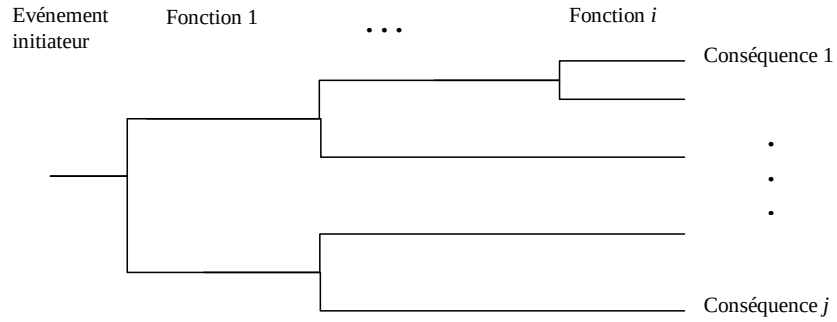


FIG. 2.4 – Exemple de structure d'un arbre d'événements

En progressant, le schéma présente des bifurcations et le nombre de branches croit au fur et à mesure de la progression vers les conséquences. Une probabilité peut être associée aux événements si celle-ci est connue.

2.2.6 Diagramme de succès

Le diagramme de succès est aussi appelé diagramme de fiabilité (*Reliability Block Diagram*). L'analyse par diagramme de succès considère pour un système : une entrée, une sortie et le fonctionnement. Des blocs représentent généralement des composants, des sous systèmes ou des fonctions. La modélisation consiste à chercher les liens entre ces blocs. Les principes suivants sont appliqués : 1) les blocs qui représentent des composants dont la défaillance entraîne la défaillance du système sont placés en série. Fig.2.5.a montre un exemple de diagramme série où le système est défaillant si $c1$ est défaillant ou $c2$ est défaillant. 2) Les blocs qui représentent des composants dont la défaillance n'entraîne la défaillance du système qu'en combinaison avec d'autres blocs sont placés en parallèle. Fig.2.5.b montre un exemple de diagramme parallèle où la défaillance du système est due à celle de $c1$ et $c2$.

Les diagrammes de succès sont principalement utilisés pour calculer la fiabilité de systèmes (ou de sous systèmes) modélisés. Parmi les inconvénients de cette méthodes, ceux déjà cités des arbres de défaillances. Ils présentent également des limites quant à la répétition des événements (pas d'événements répétés). De plus, tous les chemins entre l'entrée et la sortie ne considèrent que l'accomplissement d'une fonction avec l'hypothèse que les composants ne possèdent que deux états.

2.3 Evolution des besoins en SdF

L'importance de la SdF et la multitude des besoins auxquels elle répond est démontré par le grand nombre de normes qui existent dans le domaine. On peut

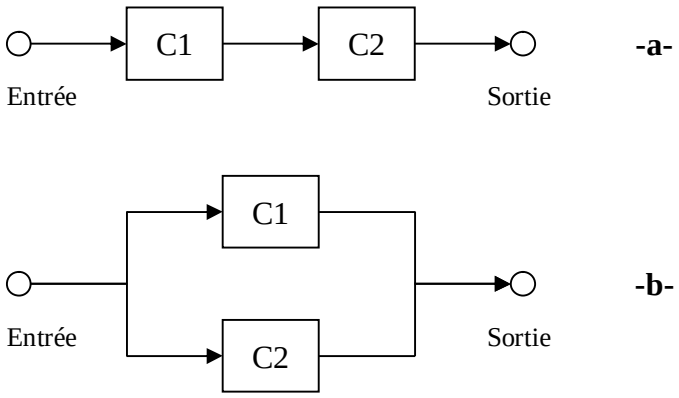


FIG. 2.5 – Exemple de diagrammes de succès

citer par exemple les normes IEC 61078 :2006 (les blocs diagramme de fiabilité) et IEC 60812 :2006 (Analyse des Modes de Défaillances et de leurs Effets). La notion même de la SdF a évolué et évolue toujours avec les besoins grandissants des industries. Cette évolution est rapportée dans [Vil88] et [CGCL04] dont voici quelques jalons historiques.

Les premières visions de la SdF d'un système reposait sur la fiabilité de son maillon (ou sous système) le plus faible. Or, dans un système, ce n'était pas systématiquement le maillon le plus faible qui se rompait en premier. Dans la fin du 19*ème* et début du 20*ème* siècle, des problèmes de fiabilité se sont posés dans les technologies innovantes de l'époque comme dans les trains transaméricains (roulements à billes et freins) et l'énergie hydroélectrique (transformateurs, lignes de tension, interconnexions de lignes) qui ont engendré des problèmes dramatiques. L'absence d'étude de sûreté de fonctionnement coûta au métro parisien ses 84 morts en 1903 dans un incendie. Les rames du métro étaient en bois et toute l'intensité électrique de l'alimentation des moteurs passait sous le plancher de toutes les voitures facilitant ainsi les incendies en cas de court-circuit.

Dans les années 1930, les transports aériens commençaient a collecter des informations statistiques sur les moteurs et les accidents des appareils. Les premiers objectifs quantifies l'ont été par le capitaine A.F. Pugsley de l'armée canadienne, entre 1939 et 1942, avec un taux de défaillance évalué à $10^{-5}/h$ pour les avions, dont $10^{-7}/h$ pour leur structure *« en considérant toutes les causes de pannes susceptibles d'entraîner un accident »*.

Les années 1940 voient le formidable essor des techniques de fiabilité grâce à des travaux comme ceux du scientifique allemand W. Von Braun qui a mis au point les missiles V1 et ceux d'Eric Pieruschka qui a donné la formule de calcul de la fiabilité d'une chaîne : la probabilité de survie d'une chaîne à une date t arbitraire est le produit des probabilités de survie de chacun de ses composants

à cette date, sous l'hypothèse que ces composants sont indépendants les uns des autres. Aux États-Unis, à cette époque, les nouvelles techniques permettent de gagner un facteur 4 sur la durée de vie des moteurs de traction des locomotives.

Les années 1950 assistent à l'avènement du concept de maintenance : \$1 en équipement génère \$2 en maintenance. À cette époque, les premières directives en électronique voient le jour, avec des spécifications d'essais de vieillissement accéléré. L'avènement du nucléaire entraîne les premières études sur la fiabilité humaine. En France, c'est le Centre national d'Etudes sur les Télécommunications qui commence ses travaux sur un recueil de données de fiabilité électronique.

Dans les années 1960, on assiste aux premiers pas des analyses relatives aux défaillances de composants et les premières vraies exigences de sûreté de fonctionnement. Ces analyses font appel à des démarches comme les arbres de causes et les analyses préliminaires des risques. Dans un souci d'harmonisation et de standardisation, la Commission électrotechnique internationale crée le Comité technique 56 "*Dependability*" (sûreté de fonctionnement) en octobre 1965. Les produits de ce groupe deviendront des normes internationales dès 1976.

Dans les années 1970-80, les résultats des premiers travaux sur la fiabilité du logiciel sont publiés. En 1979, la catastrophe nucléaire de TMI (*Three Miles Island*) a promu d'une manière inattendue les outils de sûreté de fonctionnement puisque le scénario qui a mené à la catastrophe était quasiment décrit dans un rapport précédent (le rapport Rasmussen qui présente une évaluation du risque nucléaire). Puis, ce sont les industries pétrochimiques qui procèdent à leurs premières études de risques, avant que les techniques de sûreté de fonctionnement ne soient diffusées dans la chimie, le ferroviaire, l'automobile, le traitement et l'épuration d'eau, etc.

Aujourd'hui, la réglementation et les certifications qu'elle impose, ont un double effet : le développement de l'utilisation des outils de sûreté de fonctionnement et une certaine idée de la couverture des risques (on évoque de plus en plus les responsabilités dans les accidents en délaissant peu à peu la notion de risque). En parallèle, la compétition que se livrent les grands groupes continue, ce qui les force à disposer de la meilleure productivité possible, et donc à réduire les arrêts de production et à maximiser la disponibilité de leurs équipements. Enfin, la sécurité des biens et des personnes n'a jamais été aussi importante qu'aujourd'hui aux yeux de l'opinion publique. La preuve est donnée par les actions vigoureuses autour de la notion de malveillance, la pression médiatique et écologique autour des accidents notables (plate-forme Piper Alpha, accident chimique de Bophal et d'AZF, catastrophe aérienne de la TWA ou naufrage de l'ERIKA) est telle qu'elle entraîne des conséquences très lourdes pour l'entreprise.

2.4 Conception sûre de fonctionnement

La conception de systèmes sûrs de fonctionnement se heurte à des défis liées à plusieurs facteurs comme la taille des systèmes, leur complexité ou encore la grande variété de systèmes et des exigences à prendre en considération. La prise en compte, à la fois, des exigences fonctionnelles et extra-fonctionnelles est un exemple des problèmes posés. D'autres exemples (les défaillances dépendantes et de cause commune, les facteurs humains et des domaines particuliers) sont apportés par [Vil88] dans le cadre d'utilisation de méthodes spécifiques pour la SdF. [LAB+96] propose une méthode de conception de systèmes sûrs de fonctionnement basée sur les moyens de la SdF.

2.4.1 Méthode de conception sûre de fonctionnement

Selon [LAB+96], un système sûr de fonctionnement est celui qui combine les moyens de la sûreté de fonctionnement : prévention des fautes, tolérance aux fautes, élimination des fautes et prévision des fautes ; de préférence, à chaque étape du processus de développement. L'interaction entre ces moyens est schématisée comme suit : malgré l'utilisation de méthodes de prévention de fautes, des erreurs (résultant des fautes) apparaissent. Ceci impose un diagnostic pour déterminer la (les) faute(s) cause(s) de l'erreur. Une fois trouvées les fautes causes sont éliminées (élimination de fautes). Celle élimination n'étant pas parfaite, conjuguée à l'imperfection des composants embarqués existants (matériels et logiciels), rend nécessaire la prévision des fautes. Le système dans lequel survient une faute doit souvent continuer à rendre un service d'où la tolérance aux fautes. Celle-ci est basée sur des méthodes de construction d'où l'élimination des fautes, prévision des fautes et ainsi de suite.

2.4.2 Cas d'utilisation de méthodes spécifiques pour la SdF

Les défaillances dépendantes et de cause communes

Deux défaillances E_1 et E_2 sont dites dépendantes si et seulement si

$$P(E_1.E_2) = P(E_1).P(E_2/E_1) \neq P(E_1).P(E_2)$$

En d'autres termes, les occurrences des deux défaillances ne peuvent pas être caractérisées l'une indépendamment de l'autre. La cause commune caractérise un événement (ou une série d'événements) dont l'effet est de produire plus qu'une défaillance. La problématique dans ce domaine est traitée dans des travaux anciens comme [FRH+75] et [Apo76] et plus récemment dans le domaine des systèmes embarqués comme [SK07] et [MSS09].

Le domaine des facteurs humains

L'interaction de l'humain avec un système embarqué dure pendant tout le cycle de vie de ce dernier (de la conception au retrait). Les erreurs introduites dans les systèmes embarqués par l'humain se situent à toutes les étapes : erreurs de conception, de programmation, de fabrication, d'exploitation, etc. Le facteur humain apporte un certain nombre de caractéristiques difficiles à prendre en compte dans une démarche de sûreté de fonctionnement telles que la variabilité, le besoin d'information, la capacité de prévision ou le stress.

2.5 Sûreté de fonctionnement et réseaux de Petri

L'utilisation des Réseaux de Petri (RdP) dans la sûreté de fonctionnement apporte un complément aux méthodes classiques quand celles-ci atteignent leurs limites dans des cas comme la "*fiabilité dynamique*" par exemple. Beaucoup de travaux s'intéressent à la relation entre la discipline de la SdF et la modélisation basées sur RdP avec toutes les extensions.

Les RdP se distinguent des autres modélisations de la sûreté de fonctionnement par leur utilisation à travers tout le processus de développement. L'avantage est de garder le même formalisme pour représenter le couple structure/comportement du système (analyse fonctionnelle) et le comportement défaillant (comme le représenterait un arbre de défaillance ou un graphe de Markov). Cette souplesse est augmentée par la double représentation des RdP : mathématique et graphique [MBT02].

Pour [LE98], [ME00] et [MEJP98], les approches adoptées pour étudier la SdF en général et la fiabilité en particulier reposent sur trois étapes principales. 1) la construction du modèle de spécification (conception du système), suivie 2) d'une étude qualitative sur la fiabilité qui met en avant les causes possibles des défaillances et les interconnexions entres ces sources. La dernière étape 3) est l'étude quantitative qui a pour objectif de quantifier les fréquences et les probabilités d'occurrence des défaillances. [Hei95] propose une modélisation en RdP pour l'étape 1 et une méthode combinant différentes analyses des RdP pour les deux autres étapes.

Certains travaux proposent des méthodes de traduction de formalismes divers en RdP pour une étude de sûreté de fonctionnement. Ainsi [KO01] traduit les arbres de causes en RdP, [MT95] traduit les arbres de défaillances dans leur équivalent en réseaux de Petri stochastiques généralisés (notés GSPN pour *Generalized Stochastic Petri Net*) et SRN (*Stochastic Reward Nets*) et [Rug07] traduit

le langage de description AADL (*Architecture Analysis and Design Language*)[23] en réseaux de Petri stochastiques.

Les applications pratiques des RdP pour une étude de SdF utilisent le plus souvent des extensions des RdP. [dSC04] utilise les réseaux de Petri stochastiques pour évaluer la SdF des réseaux Profibus-DP[24] combinant des méthodes analytiques, numériques et simulations en fonction de la difficulté d'analyse des modèles.

L'utilisation des GSPN dans [Hav93] est double. D'abord, ils servent à générer automatiquement les modèles Markoviens sous-jacents. Ensuite, l'espace d'état généré est utilisé pour des études de performances moyennant des heuristiques de réduction de la taille d'espace d'état. Ces heuristiques distinguent deux cas d'étude : modèles inversibles et modèles non inversibles.

[BBD03] présente une large synthèse sur l'utilisation des formalismes RdP pour les études SdF. L'objectif étant l'analyse quantitative basée sur une approche probabiliste, [BBD03] développe l'utilisation des modèles GSPN et leur contrepartie colorée : les réseaux de Petri bien-formés stochastiques (notés SWN pour *Stochastic Well-formed Nets*). L'utilisation des réseaux de Petri de haut niveau possède deux avantages principaux. D'abord, le fait de disposer d'un langage de description de haut niveau (avec tous les avantages et facilités que cela confère). Ensuite, la possibilité de réutilisation des méthodes et outils disponibles pour des formalismes de plus bas niveau.

Dans la classe des Réseaux de Petri Colorés (RdPC), [SS03] utilise ce formalisme pour ces nombreux avantages comme la facilité de visualisation et le prototypage rapide. Mais il pointe aussi les inconvénients de ce formalisme comme l'explosion combinatoire du graphe d'occurrence et les difficultés des calculs des invariants. D'où la proposition d'association des RdP au système de vérification de prototypes (PVS pour *Prototype verification System*) pour les preuves mathématiques.

L'intérêt d'utiliser les RdPC pour [ZBH96] dans un système de production est le fait de pas dupliquer les modèles RdP (comme les RdP stochastiques par exemple) dans le cas de machines produisant plusieurs produits. Il suffirait alors de concevoir le processus de production pour chaque produit individuellement et de fusionner à la fin les modèles "*non colorés*" pour obtenir un modèle RdPC. Tandis que pour [Bar03], qui utilise les RdPC pour évaluer des critères de sûreté de fonctionnement pour des systèmes distribués, l'intérêt (outre la représentation polyvalente statique/dynamique et la reconnaissance) réside dans le besoin d'exprimer des valeurs numériques (vitesse, température) et des données plus complexes (trames de communications). Dans des travaux comme [BSB09b] ou [BSB09a]

[23] AADL (SAE-AS5506) est standardisé par SAE (*Society of Automotive Engineers*) en novembre 2004

[24] Profibus est un bus de terrain crée par Siemens. Utilisé dans l'industrie, il relie les parties capteurs/actionneurs aux automates (contrôle commande)

l'intérêt des RdPC est accru par la facilité de modélisation et de prise en compte des aspects humains dans les analyse de SdF.

2.6 Conclusion

La sûreté de fonctionnement (SdF) est un concept pivot dans le cycle de vie des systèmes embarqués. Ce chapitre tente de fournir des bases de cette large discipline à travers quelques facettes comme les définitions de base, les démarches et l'évolution des besoins en SdF. Certaines facettes, et notamment les démarches, font ressortir l'importance de la modélisation. En effet, la modélisation est une notion centrale dans la SdF dans le sens où on y a souvent recourt pour exprimer les démarches/méthodes.

L'une des conclusions qui ressortent est l'imperfection des méthodes prises de manière isolées et donc la nécessite d'associer les démarches entre elles pour exprimer le plus d'attributs, d'entraves et de moyens possibles. L'expressivité des modèles n'étant pas identiques, nous avons opté pour une modélisation expressives : les réseaux de Petri. C'est pour cela que l'utilisation de ces modèles dans la SdF est détaillée à part.

Dans la suite, l'accent sera mis sur les aspects RdP de manière générale et RdPC de manière plus particulière. Le potentiel de ces modélisations en terme de SdF sera mis en avant ainsi que les limites actuelles dans ce domaine. Ceci aboutira, après un échantillon bibliographique, au choix d'une démarche particulière d'analyse des modèles RdPC qui débouchera sur une proposition de méthode d'analyse dite par accessibilité arrière.

Chapitre 3

Réseaux de Petri et la modélisation

Sommaire

Comme vu dans les chapitres 1 et 2, le recours aux modèles est primordial pour représenter l'architecture du système embarqué, sa conception, son fonctionnement ou encore les exigences auxquelles il doit répondre. On parle alors de modélisation. Avant de traiter ce sujet en profondeur, définissons d'abord ce terme.

La modélisation est la démarche consistant à substituer à un processus étudié un système issu d'un univers différent mais qui présente des analogies avec ce processus [Mal02]. Les natures des modèles générés sont multiples : textuel, graphique et/ou mathématique. Le modèle généré est une abstraction souvent simplifiée d'un système complexe. D'ailleurs, selon [Mal02], la modélisation est fructueuse à chaque fois que le système substitutif (le modèle généré) est plus facile à étudier que le système de base.

La modélisation permet de visualiser le système, de le simuler, d'analyser son comportement ou encore de tester les conséquences d'un événement. Elle se substitue ainsi à l'expérimentation quand celle-ci est difficile ou coûteuse. Elle permet donc des gains en temps, en coût et en complexité de développement et de maintenance. L'expérimentation reste néanmoins indispensable pour confirmer l'exactitude ou la proximité du modèle abstrait vis-à-vis du système représenté.

Le développement des outils de calcul a largement contribué à l'essor de la modélisation. On parle souvent de langage de modélisation. C'est une construction linguistique (textuelle ou graphique) pouvant exprimer une connaissance sur un système dans une structure définie par un ensemble de règles consistantes. Les règles servent à interpréter le sens de composantes du modèle. L'un des exemples les plus connus est, entre autres, la modélisation UML (*Unified Modeling Language*) applicable pour le développement logiciel

Ce travail s'intéresse à la modélisation basée sur les réseaux de Petri (RdP), et plus particulièrement, les réseaux de Petri colorés (RdPC). Ce choix est motivé par un ensemble de facteurs. D'abord, ces modèles sont graphiques et permettent de représenter le système d'une manière intuitive. Ils renseignent, dans un modèle unique, sur les deux aspects du système représenté : statique (grâce à la structure même du modèle) et dynamique (grâce à l'évolution des jetons dans la structure et éventuellement l'évolution de leurs valeurs). En outre, les RdP sont recommandés comme des méthodes formelles notamment dans le domaine de la sûreté de fonctionnement grâce au socle théorique (mathématique) sous-jacent au modèle graphique. En terme de puissance de modélisation, les RdP possèdent un très grande expressivité. Ils peuvent exprimer des aspects aussi variés que les communications, les contraintes temporelles et les lois de commande. Des outils comme *CPN Tools* permettent d'offrir une aide à la modélisation, la simulation et l'analyse des RdP. Mais, les problématiques et défis scientifiques, notamment dans le volet analyse des RdP, restent ouverts et offrent des opportunités intéressantes d'innovation[25].

Ce chapitre met l'accent sur les techniques d'analyse des RdP et RdPC et notamment les analyses structurelles basées sur l'accessibilité arrière et l'inversion

[25]Voir la section 2.5 pour une revue de l'utilisation des RdP dans les problématiques de la sûreté de fonctionnement

de modèles. Pour cela, les définitions de base sont données suivies des différents types d'analyses applicables aux RdP/RdPC. Ensuite vient un état de l'art des travaux s'intéressant à l'inversion des modèles. Les méthodes y sont exposées, illustrées et discutées.

3.1 Réseau de Petri Coloré

3.1.1 Définition d'un Réseau de Petri

Un réseau de Petri (abrégé RdP), appelé aussi réseau de Petri ordinaire ou encore réseau de Petri place/transition est introduit par [Pet62]. Il est défini par un quadruplet $(P, T, Pre, Post)$ où :

- P est un ensemble fini de *places*,
- T est un ensemble fini de *transitions* $T \cap P = \emptyset$,
- Pre est l'application (matrice) d'*incidence avant* : $Pre \subseteq (P \times T)$,
- $Post$ est l'application (matrice) d'*incidence arrière* : $Post \subseteq (T \times P)$.

Graphiquement, un RdP est un graphe biparti[26] orienté où les places sont représentées par des cercles (ou des ellipses) et les transitions par des traits (ou des rectangles). Les arcs orientés vont soit des places vers les transitions soit l'inverse (jamais d'arcs entre deux places ou deux transitions).

Les différents états d'un RdP correspondent à différentes distributions de *jetons* dans les places. Chacune de ces distributions est appelée *marquage* et est représentée par un *vecteur de marquage*. Le tir d'une transition t fait évoluer un marquage m_i vers un marquage m_{i+1}. Le tir de t est possible si et seulement si le marquage m_i est supérieur ou égal aux préconditions de t (ou en amont de t). On écrit dans ce cas $m_i \geq Pre_t$. Le marquage m_{i+1}, résultant du tir de t est donné par la formule : $m_{i+1} = m_i - Pre_t + Post_t$. Cette formule signifie, dans le cas des RdP ordinaires, que le tir de t supprime un jeton dans chaque place précondition de t et ajoute un jeton à chaque place postcondition de t (ou en aval de t).

3.1.2 Définition d'un Réseau de Petri Coloré

La notion de couleur dans un RdP apporte une sémantique de différentiation des jetons. Dans un réseau de Petri coloré (noté RdPC), les jetons possèdent une valeur typée. Ils ne sont plus anonymes comme dans les RdP ordinaires. Cette sémantique amène une grande expressivité et compacité au modèle RdPC. Les jetons peuvent changer de valeur lors de tirs de transitions grâce aux expressions associées aux arcs. Les tirs de transitions peuvent être conditionnés par les valeurs de jetons grâce aux gardes.

[26] biparti (ou bipartite) : constitué de deux types d'éléments. Dans le cas des RdP ce sont les places et les transitions.

Formellement, un Réseau de Petri Coloré non hiérarchique [JR91] est un produit cartésien $CPN = (\Sigma, P, T, A, N, C, G, E, I)$ défini par :

- ⋆ Σ l'ensemble fini de types non vides, appelés *ensembles de couleurs*,
- ⋆ P l'ensemble fini des *places*,
- ⋆ T l'ensemble fini des *transitions*,
- ⋆ A l'ensemble fini d'*arcs* tel que $P \cap T = P \cap A = T \cap A = \emptyset$,
- ⋆ N la fonction des nœuds. Elle est définie de A vers $P \times T \cup T \times P$,
- ⋆ C la *fonction de couleurs* de P vers Σ,
- ⋆ G la fonction des *gardes*. Elle est définie de T vers les expressions telles que :

$$\forall t \in T : [Type(G(t)) = B \wedge Type(Var(G(t))) \subseteq \Sigma],$$

- ⋆ E fonction d'*expression des arcs*. Elle est définie de A vers les expressions telle que :

$$\forall a \in A : [Type(E(a)) = C(p(a))_{MS} \wedge Type(Var(E(a))) \subseteq \Sigma]$$

où $p(a)$ est une place de $N(a)$,

- ⋆ I la fonction d'*initialisation*. Elle est définie de P vers les expressions fermées telles que :

$$\forall p \in P : [Type(I(p)) = C(p)_{MS}].$$

3.1.3 Modèles temporisés

La notion du temps est souvent citée dans les systèmes embarqués. On parle souvent de systèmes temps réel. Un tel système est défini comme *"celui dont la justesse des calculs ne dépend pas uniquement de la justesse de la logique de calcul mais aussi du temps dans lequel le résultat est produit. Si les contraintes temporelles du système ne sont pas réunies, une défaillance du système survient"* [Sta88]. À travers cette définition, l'importance du traitement du temps apparaît clairement. Il est nécessaire de relier les résultats au temps dans lequel ils ont survenu. Les RdP peuvent inclure les aspects temporels en associant un événement à l'instant dans lequel ils survient. Cette fonction peut être associée à une transition, une place ou un arc.

Il existe deux principale familles d'extensions temporelles des RdP [CR03] : les réseaux de Petri temporisés introduits par [Ram74], [JLL77] et les réseaux de Petri temporels introduits par [Mer74]. La différence entre les deux est que dans le cas des réseaux de Petri temporisés, une temporisation représente la durée minimale de tir d'une transition ou le temps de séjour minimum d'un jeton dans une place (ou durée exacte avec une règle de fonctionnement au plus tôt). Dans le cas des réseaux de Petri temporels, l'expression temporelle est donnée sous la forme d'un intervalle. Plusieurs sous classes et travaux existent dans ce domaine tels que

[KDCD96] (RdP P-temporels en sémantique forte), [Han93] (RdP A-temporels en sémantique faible). Des extensions de plus haut niveau ont été définies telles que les Réseaux de Petri Colorés Temporisés (*Timed Colored Petri Net* [JK09], [CJMK01]) et les réseaux de Petri colorés temporisés par intervalle (*Interval Timed Colored Petri Nets* [vdA93]).

Il existe une manière non déterministe de spécifier l'aspect temporel d'un RdP. Le temps associé à un élément du RdP est un délai décrit par une variable aléatoire (avec une distribution définie). Cette classe de RdP est appelée RdP stochastiques (SPN pour *Stochastic Petri Nets* [Sym78], [Mol82]). Le terme SPN désigne la classe la plus générale des RdP modélisant des délais stochastiques. Elle inclut différentes extensions comme les RdP stochastiques généralisés (GSPN pour *Generalized Stochastic Petri Nets* [MBC$^+$95]) qui introduisent des transitions immédiates et les RdP stochastiques bien formés (SWN pour *Stochastic Well-Formed Nets* [CDFH93]) qui introduisent la "couleur".

Bien sûr, dans un contexte plus large que les RdP, la notion de temps est incluse dans d'autres modèles. L'un des plus populaires sont les automates temporisés (*Timed Automata* [AD94]). Certains travaux, comme [CR03], proposent des traductions des modèles RdP vers les automates afin de profiter des techniques d'analyse existantes plus développées pour ces modèles. D'un point de vue implémentation, les aspects temporels ont permis la conception de différents outils mettant en œuvre les modèles temporisés. Citons comme exemple *UPPAAL* [ABB$^+$00], *CMC* [LL98], *Phaver* [Fre05] et *CPN Tools*.

3.2 Logiciel utilisé

Une étude comparative de logiciels de modélisation en RdPC se trouve dans [Bar03]. Elle est basée sur des critères imposés (comme le travail avec des données structurées) et des critères de plus-values (comme la disponibilité, le coût et la modularité). Cette étude compare principalement *Design/CPN*, *GreatSPN* et *Miss-RdP*. Pour les présents travaux, et en ce qui concerne la modélisation, le choix s'est porté sur le *CPN Tools* (qui a remplacé *Design/CPN*). La suite de cette section présentera donc le logiciel *CPN Tools*.

Le logiciel de modélisation *CPN Tools*[27], développé à l'université d'Aarhus au Danemark, édite, simule et analyse les réseaux de Petri colorés. Il se compose de plusieurs modules offrant ainsi différentes fonctionnalités. Il permet de modéliser graphiquement (dessiner) un RdPC dans une page de l'interface utilisateur. Le modèle comprend la structure statique du RdPC (places, transitions, arcs) et la partie dynamique (les jetons valués). Un simulateur intégré gère à la fois des RdPC temporisés et non temporisés. Il peut générer et analyser un espace d'états

[27] http ://wiki.daimi.au.dk/cpntools

partiel ou total et un rapport d'espace d'états peut contenir des informations sur les propriétés du RdPC telles que la vivacité ou le caractère borné. Le RdPC est stocké sur disque sous forme d'un document XML. Les moteurs de calcul et les traitements sont écrits en CPN ML qui est une variante du langage de programmation ML [D.U97]. Le *CPN Tools* est utilisé dans différents contextes : académiques comme dans [JKW07] et industriels comme dans [FTJ07]

Une place de RdPC dans *CPN Tools* est une ellipse à laquelle sont associés un nom, une couleur et éventuellement un marquage initial. Les jetons d'une place doivent appartenir à l'ensemble de couleurs de celle-ci. Une transition de RdPC dans *CPN Tools* est un rectangle possédant quatre attributs facultatifs : le nom, l'expression de la garde (placée entre crochets []), le code (programme en langage ML) et la temporisation (commence toujours par le caractère "@"). Un arc relie toujours une place et une transition. Son seul attribut, l'inscription (ou l'expression), peut contenir soit une ou plusieurs valeurs, soit une variable, soit une fonction.

3.3 Approches d'analyse des RdP et RdPC

Comme ce chapitre se concentre sur la modélisation basée sur RdP, les approches d'analyse qui vont suivre sont restreintes au point de vue de cette modélisation. Dans un contexte plus global, ces méthodes sont très générales et restent utilisables dans un cadre bien plus large que les RdP.

Dans cette section, trois types d'approches sont présentés : simulation de Monte Carlo, analyses formelles et analyses structurelles.

3.3.1 Simulation de Monte Carlo

Définitions

La dénomination *simulation de Monte Carlo*, formalisée par Metropolis et Ulam [MU49], fait référence aux jeux de hasard très prisés dans la région de Monte Carlo à Monaco. Cette simulation regroupe une famille de méthodes numériques utilisant des tirages de nombres aléatoires. Le but est d'évaluer de manière itérative un modèle de système (souvent complexe). La simulation de Monte Carlo peut être utilisée dans des domaines très variés comme l'économie, la physique nucléaire, la régulation des flux ou le calcul des intégrales.

Avantages

La simulation de Monte Carlo est un ensemble de méthodes constituant un outil mathématique très général, dont le champ d'application est très vaste. Pour

chaque application, ou suivant la nature du problème envisagé, la méthode de Monte Carlo employée possède ses propres caractéristiques. Le seul point commun entre elles est l'utilisation de nombres aléatoires pour décrire le caractère stochastique des phénomènes ou pour résoudre des problèmes plus complexes qui ne peuvent pas être traités analytiquement de manière efficace.

Inconvénients

Les résultats obtenus par la simulation de Monte Carlo sont exacts au sens statistique ; i.e. ils présentent une certaine incertitude qui diminue avec l'augmentation du nombre d'itérations et donc la durée de la simulation. Pour minimiser les temps de calcul et converger rapidement vers la solution, différentes méthodes existent telles que la réduction de la variance [PG80] et la simulation par sauts [Mon98]. De plus, les résultats obtenus dans les simulations peuvent être altérés ou biaisés par la qualité du générateur de nombres aléatoires (pseudo-aléatoires) d'où l'importance du choix de ce générateur.

Simulation de Monte Carlo et SdF

[MEJP98] a introduit une analogie entre les algorithmes de simulation de Monte Carlo dans la SdF et la théorie du transport. Cette analogie permet de transférer plusieurs outils de la théorie du transport vers la SdF tels que les estimateurs efficaces qui font correspondre un état de dysfonctionnement d'un système avec une absorption d'énergie.

[GBU03] propose d'injecter les connaissances fournies par l'étude qualitative de la SdF, sous forme d'un terme a priori, dans la formule à simuler, pour quantifier par simulation le risque de dysfonctionnement. Ce terme a priori provient d'une formalisation d'un (ou des) modèle d'étude qualitative tels que le diagramme de fiabilité et les arbres des causes et donc exploitant les sources connues de défaillances.

Une autre amélioration de la simulation de Monte Carlo, dans le cadre de la SdF et de l'étude de fiabilité, est proposée dans [ME00]. Elle consiste à utiliser une simulation biaisée faisant apparaître les événements rares. Le principe est de faire apparaître des termes dans la fonctionnelle à estimer et de biaiser le résultat.

3.3.2 Approches par analyse formelle

Les méthodes formelles sont des techniques basées sur des modèles mathématiques. Elles permettent de spécifier, développer et vérifier le système sous-jacent. Les méthodes formelles peuvent inclure des langages graphiques, parmi ceux-ci, les machines à état et les RdP [Gab06]. La construction d'un graphe de marquages atteignables s'approche des méthodes formelles pour l'analyse comportementale

des RdP. Ce faisant, certaines propriétés, telles que la vivacité et l'équité, peuvent être vérifiées. Ces méthodes trouvent leurs utilisations aussi bien dans un cadre industriel [HB99] que dans un cadre plus théorique [LM04].

Une autre technique d'analyse flexible et automatique, en relation avec les méthodes formelles est le *model checking* [CGP99]. Elle permet de vérifier si le comportement d'un système (modélisé) est conforme à une spécification donnée à l'aide d'une logique. Plusieurs types de logiques existent dont LTL (*Linear-time Temporal Logic*)[Lam80] et CTL (*Computation Tree Logic*)[EH82].

Les approches de validation temps réel visent à intégrer les caractéristiques aléatoires de l'évolution des systèmes temps réel avec la robustesse des approches formelles. Dans ce raisonnement, l'état du système est souvent décrit par une formule logique composée par un ou plusieurs prédicats. Un exemple de formalisme exprimant les propriétés temps réel est le formalisme CSL (*Countinuous Stochastic Logic*) utilisé dans [HSEG02] et [OTSW04]. Ce formalisme intègre des formules telles que : $P_{\geq\theta}(c)$: c est vrai avec une probabilité θ, et $c_1 U^{\leq t} c_2$: c_1 jusqu'à c_2 en un temps inférieur à t (t fini). Le formalisme CSL définit aussi une sémantique donnant une valeur de vérité aux formules et les règles d'inférence pour construire une démonstration à partir de ces formules. Le but est de valider le fait qu'un état du système soit sain ou pas et de démontrer la succession d'événements menant à cet état et les moments où ils se sont produits tout en minimisant l'erreur sur le résultat.

Avantages

Les méthodes formelles, de par leurs fondements mathématiques, fournissent des résultats exacts. Les algorithmes mettant en œuvre les approches formelles explorent l'ensemble des états atteignables pour fournir une décision quant à la propriété recherchée. De plus, l'expression des propriétés et besoins se fait d'une manière précise grâce à des formalismes et des logiques non ambiguës.

Inconvénients

L'un des grands problèmes des approches formelles est l'explosion combinatoire liée au nombre d'états que peut avoir le système. L'exploration de tous les états s'avère souvent fastidieuse voire impossible. D'un autre coté, l'analyse formelle des systèmes temps réel est stochastique à deux sens [HSEG02]. D'abord, les propriétés en entrée peuvent être probabilistes (obtenues statistiquement) et les résultats en sortie aussi peuvent être probabilistes (avec des erreurs bornées). Les grammaires formelles, visant à vérifier les propriétés des systèmes temps réels, incluent des termes, des formules et/ou des opérateurs temporels dont l'estimation de la valeur de vérité est bornée par une certaine erreur.

3.3.3 Approches par analyse structurelle

Les méthodes basées sur énumération d'états permettent de couvrir tout l'espace d'états atteignables, mais souffrent du problème d'explosion combinatoire. Pour l'éviter, une des solutions possibles est le passage par l'analyse structurelle. Cette analyse exploite la structure même du RdP pour en extraire les informations et les propriétés. Certains travaux proposent d'aller encore plus loin dans la démarche de l'analyse structurelle en utilisant non pas le RdP original mais une transformation de celui-ci. Le but principal est de faciliter l'analyse.

Dans le cas des RdPC, plusieurs types de transformations peuvent être définies. La transformation triviale est le dépliage du RdPC qui consiste à donner l'équivalent en RdP ordinaire d'un RdPC. La taille du RdP résultat augmente parfois exponentiellement par rapport à celle du RdPC d'origine. Pour réduire la taille du RdPC à étudier, [Had89] définit certaines règles et mécanismes. Cette réduction préserve les propriétés dynamiques du RdPC telles que la vivacité. Elle utilise plusieurs méthodes comme la post-agglomération, la pré-agglomération et l'élimination des places implicites qui permettent de "*supprimer*" des éléments du RdPC tout en gardant ses propriétés. Le but est de pouvoir effectuer une analyse plus efficace et plus rapide sur un RdPC *réduit* ayant les mêmes propriétés que RdPC initial.

Dans le cadre des RdP ordinaires, [Kha03] s'intéresse à l'accessibilité entre deux marquages vues de deux manières duales : l'accessibilité avant et l'accessibilité arrière. Le premier cas consiste, à partir d'un marquage initial (M_0), à construire les états successeurs pas à pas, jusqu'à atteindre le marquage final (M_f). Donc, considérant M_0 comme l'état présent, M_f est considéré comme l'état futur. Dans l'accessibilité arrière, M_f constitue l'état présent et M_0 est vu comme un état passé. Le but est de construire, à partir de l'état présent, les états prédécesseurs jusqu'à atteindre l'état passé (source du marquage présent). Cette étude montre que l'accessibilité avant (de M_0 vers M_f) n'est pas forcément identique à l'accessibilité arrière (de M_f vers M_0), c'est-à-dire que dans les deux cas, les marquages et les séquences explorées ne sont pas identiques. D'une manière générale, l'accessibilité avant permet de mettre en évidence les états qui mènent au marquage final et ceux qui mènent aux marquages autres que le marquage final. Réciproquement, l'accessibilité arrière identifie les états qui mènent d'un marquage initial à un marquage final, et aussi ceux qui mènent vers le marquage final sans passer par le marquage initial. Il est à noter que l'intersection des ensembles d'états produits par ces deux approches n'est pas vide.

La méthode choisie pour effectuer cette analyse est le RdP inverse. Considérant un RdP R tel que $R = (P, T, Pre, Post)$, le RdP inverse R^{-1} est défini par $R^{-1} = (P', T', Pre', Post')$ [Kha03]où :

$\star$ $P' = P$,

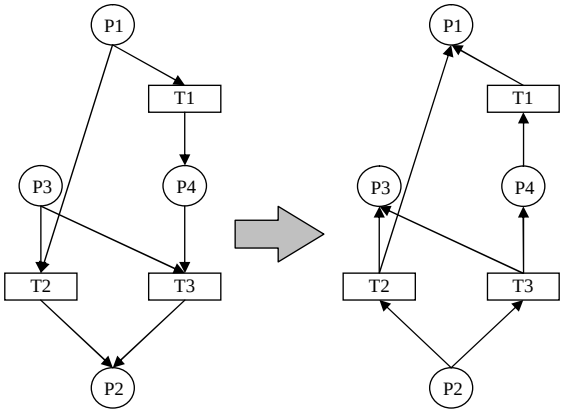


FIG. 3.1 – Exemple d'inversion de RdP

- $\star$ $T' = T$,
- $\star$ Les applications Pre' et Post' sont définies telles que $\forall p \in P, \forall t \in T \begin{cases} pre'(p,t) = post(p,t) \\ post'(p,t) = pre(p,t) \end{cases}$.

D'une manière informelle, l'inversion d'un RdP place/transition revient à l'inversion du sens des arcs qui le composent, ou encore, au remplacement de Pre par $Post$ et vice versa. Fig.3.1 illustre un exemple concret.

Pour vérifier formellement cet aspect, il suffit de travailler sur les matrices d'incidence (Pre et $Post$). Un RdP, étant un graphe bipartite, possède des matrices d'incidence avant et arrière correspondant aux relations places/transition. Pour exprimer la relation en accessibilité avant (à un pas) entre les places, il suffit de multiplier $Pre_{|P|\text{x}|T|}$ (|P| lignes et |T| colonnes) par $Post^t_{|T|\text{x}|P|}$ (la transposée de Post à |T| lignes et |P| colonnes).

Dans cet exemple :

$$Pre = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}, Post = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, Pre.Post^T = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Pour exprimer la relation arrière entre les places, il suffit de transposer la matrice $Pre.Post^T$. D'un autre coté, on a la relation $(Pre.Post^T)^T = (Post^T)^T.Pre^T = Post.Pre^T$. On voit bien que pour exprimer la relation inverse dans le RdP, la même formule est utilisée moyennant une permutation entre Pre et $Post$. Ceci montre la pertinence de la construction du RdP ordinaire inversé.

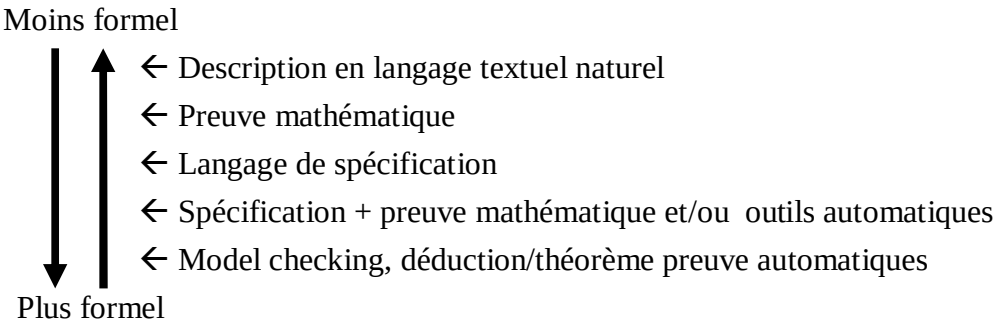


FIG. 3.2 – Spectre de formalisation

3.3.4 Spectre de formalisation

La relation entre le domaine formel et le domaine informel réside en deux points [Gab06] : la description d'un problème pratique dans une représentation formelle et l'interprétation des résultats (issus du calcul des représentation formelles) sous forme d'une solution pratique. L'un des points les plus importants est de trouver une manière compréhensive et convenable pour définir et décrire le problème[28] afin de faciliter la recherche de la solution.

Toujours selon [Gab06], le niveau formel peut être varié d'une application à l'autre et d'un domaine à l'autre en se basant sur les exigences et le niveau de détail des spécifications disponibles. Fig.3.2 montre les différents niveaux du spectre de formalisation. Dans cette figure, le langage de spécification est utilisé comme un ensemble de formules d'un langage formel décrivant le système étudié.

3.4 Problématique d'inversion de modèles RdPC

La pertinence de modélisation des systèmes en RdPC est largement reconnue grâce à leur grande expressivité, leur compacité et leur double caractère graphique/mathématique, entre autres. Les modèles décrivent généralement le fonctionnement d'un système avec une démarche inductive : l'évolution des états (causes) produit des conséquences. Par contre, l'analyse des modèles par accessibilité arrière est une démarche déductive (des conséquences vers les causes). Le principal intérêt de cette démarche est d'apporter une information nouvelle sur un modèle déjà établi. La réalisation de cette démarche peut être formelle, structurelle, etc.

[28]La notion de *description de problème* fait référence dans ce contexte à la modélisation

Dans ce travail, nous nous intéressons à l'analyse des RdPC par accessibilité arrière du point de vue **structurel**. Les travaux précurseurs dans cette voie sont détaillés et discutés dans la section 3.5. Parmi les méthodes explorées, nous avons choisi de nous inspirer des travaux de [Kha03], à savoir, l'inversion des modèles RdP. Le défi principal qui se pose est la généralisation de l'inversion, définie à l'origine pour des RdP ordinaires[29] à la classe des RdP Colorés. D'un point de vue algorithmique, la sémantique liée aux RdPC pose rapidement des problèmes et la définition des RdP inverses est insuffisante dès lors que des expressions d'arcs différentes des constantes sont définies. De plus l'accessibilité arrière utilise un certain nombre de mécanismes, comme l'enrichissement de marquage. Ces mécanismes sont efficaces dans le cas des RdP, mais ils sont incompatibles avec les RdPC du fait de l'évolution de la valeur des jetons en fonction des expressions d'arcs et en étant contraintes par les gardes (de transitions). D'un point de vue formel, l'inversion des RdP est accompagnée d'études basées sur des formalismes tels que la logique linéaire et l'algèbre linéaire. L'adaptation de ces outils mathématiques est intéressante dans le cadre de l'inversion des RdPC pour fournir un appui formel à notre étude.

3.5 État de l'art d'inversion de modèle RdP et RdPC

3.5.1 Travaux menés au LAAS

Une partie de cette thèse se base sur des travaux de recherche dirigés ou co-dirigés par R. Valette et H. Demmou. Ils ont donné lieu à trois thèses [Kha03], [Med06], [Sad07] au sein du laboratoire LAAS de Toulouse. L'objectif général de ces travaux est l'aide à la conception des systèmes mécatroniques. Un système mécatronique est défini par [Mon98] comme un système combinant des technologies qui relèvent des domaines de la mécanique, de l'hydraulique, de la thermique, de l'électronique et des technologies de l'information.

La thèse [Kha03] développe une méthode de recherche des scénarios redoutés basée sur la modélisation du système mécatronique sous la forme d'un Réseau de Petri Prédicats-Transitions Différentiel et Stochastique (RdP PTDS). Cette modélisation hybride sépare les aspects discrets et continus. En effet, un RdP PTDS distingue deux types de transitions : des transitions déterministes et des transitions stochastiques. Aux transitions déterministes sont associées des fonctions de sensibilisation et aux transitions stochastiques sont associées des fonctions sto-

[29]Dans les travaux de [Kha03], les modèles originaux sont des RdP Prédicats-Transitions Différentiels mais l'inversion ne prend en compte que le caractère structurel du modèle ce qui revient à inverser un RdP ordinaire

chastiques. Ces fonctions stochastiques sont associées à toute transition modélisant l'occurrence d'une défaillance d'un composant ou sa réparation (si celle-ci n'est pas déterministe). La recherche des scénarios dits *redoutés* démarre à partir d'un état final (défaillant) pour mettre en exergue tous les scénarios possibles conduisant à cette situation critique. Le moyen utilisé pour effectuer cette démarche est l'inversion du RdP. Lors de la recherche des scénarios redoutés, les aspects prédicatifs et stochastiques sont ignorés et l'algorithme se focalise sur l'aspect chemin menant aux sources de l'état redouté. Un fondement théorique, basé sur la logique linéaire, émaille cette étude. Un avantage de cette approche est qu'elle n'implique pas une énumération globale de tous les états accessibles du système. Au contraire, elle permet de se focaliser sur le voisinage de l'état final en faisant une énumération locale d'états partiels.

La thèse [Med06] reprend l'approche de [Kha03], en précisant plus en détail le concept de scénario et notamment de *scénario minimal*. Ceci a conduit à l'élaboration d'une autre version de l'algorithme de recherche de scénarios redoutés qui tient compte de l'aspect continu (des instants auxquels les seuils associés à certaines transitions dans le modèle RdP sont atteints). Cela permet de déterminer précisément les conditions exactes de l'occurrence de l'événement redouté. Cette approche permet d'éliminer des scénarios incohérents vis-à-vis de la dynamique continue du système. L'aspect continu est pris en compte par l'introduction des abstractions temporelles ce qui permet de vérifier certaines propriétés temporelles telles que la durée maximale d'un scénario ou celle du temps maximal pouvant s'écouler entre deux événements. L'implémentation et donc l'automatisation de l'algorithme de recherche de scénarios redoutés a été réalisée en développant l'outil ESA_PetriNet (*Extraction & Scenarios Analyser by PetriNet model*).

La thèse [Sad07] fait suite à [Kha03] et [Med06]. Le but est de développer une approche multi-modèles pour la conception des systèmes dynamiques sûrs de fonctionnement. Pour prendre en compte l'aspect hybride dans toute sa complexité, une approche de modélisation structurée et modulaire est développée. Cette approche est mise en œuvre sur des réseaux de Petri Prédicats Transition Différentiels Stochastiques Orientés Objets. L'algorithme de recherche de scénarios redoutés est adapté à cette approche. La pertinence des scénarios redoutés est liée à des propriétés comme la minimalité et la complétude. La prise en compte de la dynamique continue nécessite la mise en place d'une simulation hybride qui associe un solveur d'équations différentielles à l'algorithme de recherche de scénarios. C'est l'algorithme de génération de scénarios redoutés qui doit déterminer les équations à intégrer. Ceci permet d'effectuer une simulation locale et de réduire considérablement le coût de la simulation.

3.5.2 Travaux de Portinale

Les Travaux de [AP94] développent une méthode d'analyse par accessibilité arrière appelée *B-W Analysis*. Le but de cette méthode est de mettre en œuvre un outil de résolution de problèmes de diagnostic. Cette technique est appliquée sur une classe particulière des RdP nommée *Behavioral Petri Net* (BPN). Elle utilise deux différents types de jetons : normaux et inhibiteurs.

Le modèle *Behavioral Petri Net*

Le modèle nommé *Behavioral Petri Net*, introduit par [Por93], est un RdP conçu pour traiter des problèmes de diagnostic basé sur modèle. Il est défini comme un quadruplet $N = (P, T_N, T_{OR}, F)$ où :

- ⋆ $T_N \cap T_{OR} = \emptyset$
- ⋆ $(P, (T_N \cup T_{OR}), F)$ est un RdP
- ⋆ La fermeture transitive de F (notée F^+) est non reflexive.
- ⋆ $\forall p \in P(|^\bullet p| \leq 1 \wedge |p^\bullet| \leq 1)$[30]
- ⋆ $\forall p_1, p_2 \in P((^\bullet p_1 =^\bullet p_2) \wedge (p_1^\bullet = p_2^\bullet) \Rightarrow p_1 = p2)$
- ⋆ $\forall t \in T_N(|^\bullet t| = 1 \wedge |t^\bullet| = 0) \vee (|^\bullet t| > 0 \wedge |t^\bullet| = 1)$ [31]
- ⋆ $\forall t \in T_{OR}(|^\bullet t| = 2 \wedge |t^\bullet| = 1)$

L'ensemble des transitions d'un BPN est divisé en deux sous ensembles T_N et T_{OR}. Le sous ensemble T_N est considéré comme des transitions classiques. Elles sont représentées graphiquement par des rectangles pleins. Le sous ensemble T_{OR} représente le connecteur logique *OU*. Les transitions qui y appartiennent sont validées si au moins une précondition est validée. La sémantique formelle peut être définie (dans les RdP ordinaires) par des arcs inhibiteurs. Les transitions sont représentées graphiquement par des rectangles vides.

Certaines propriétés des BPN découlent de leur définition comme le caractère sauf, déterministe, l'existence d'exactement un seul marquage final et l'existence d'un ordre partiel de tir de transition noté " $\prec$ " : $t_i \prec t_j \Leftrightarrow t_i$ est tirée avant t_j.

Analyse B-W

Étant donné un BPN, une technique d'analyse appelée *Analyse B-W* peut être définie. C'est une analyse par accessibilité arrière utilisant deux types de jetons : jetons normaux (cercles pleins) et jetons inhibiteurs (cercles vides). La présence d'un jeton normal dans une classe signifie que la condition qui y est associée est vérifiée. La présence d'un jeton inhibiteur dans une classe signifie que la condition qui y est associée n'est pas vérifiée. Enfin l'absence de jeton dans une place signifie que la véracité de la condition qui y est associée est inconnue.

[30] $^\bullet p$: transitions en amont de la place p. $p^\bullet$: transitions en aval de la place p.
[31] $^\bullet t$: places en amont de la transition t. $t^\bullet$: places en aval de la transition t.

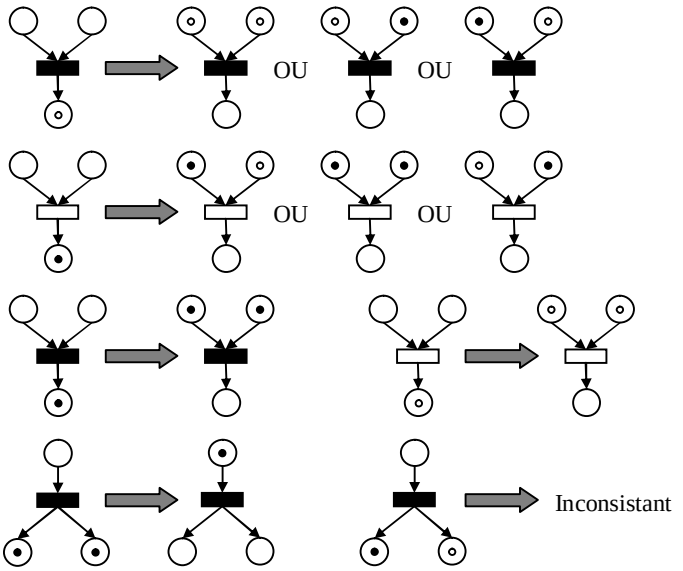


FIG. 3.3 – Règles de tir arrière dans l'analyse B-W

L'Analyse B-W se base sur un ensemble de règles de tir arrière illustrées dans Fig.3.3. Certaines de ces règles ne sont pas déterministes et d'autres produisent des états inconsistants qu'il faudra éliminer lors de l'analyse.

Le déroulement d'une analyse B-W commence par un marquage μ tel que $\mu(p) \neq 0 \Rightarrow p^\bullet = \emptyset$. Une place telle que $p^\bullet = \emptyset$ est appelée puits. Une branche de l'analyse se termine quand soit un marquage initial μ_0 est atteint ($\mu_0(p) \neq 0 \Rightarrow p$ est une place source) soit un état inconsistant est atteint.

L'application donnée est tirée du monde automobile. Le modèle illustré dans Fig.3.4 représente les fautes partielles dans un moteur de véhicule. Par exemple, la transition $t1$ modélise l'augmentation de la consommation d'huile (la présence d'un jeton noir dans la place *Oil_cons(incr)* qui peut être causée soit par un état usagé des pistons ou des chemises de pistons).

Un graphe d'accessibilité est obtenu grâce à l'analyse B-W comme illustré dans Fig.3.5. La notation $p[b]$ signifie que la place p est marquée par un jeton noir et la notation $p[w]$ signifie que la place p est marquée par un jeton blanc. Ce graphe correspond au cas de l'exemple de Fig.3.4 avec le marquage : $Ex\_smoke(black)[w], Oil\_light(red)[b], Temp_ind(red)[b]$.

3.5.3 Travaux de Muller et Schnieder

Les travaux de [MS07] développent une méthode d'analyse arrière dans les réseaux de Petri de haut niveau (notés HLPN pour *High Level Petri Net*). Elle se base sur une relation qui existe entre les HLPN et l'algèbre linéaire ce qui rend

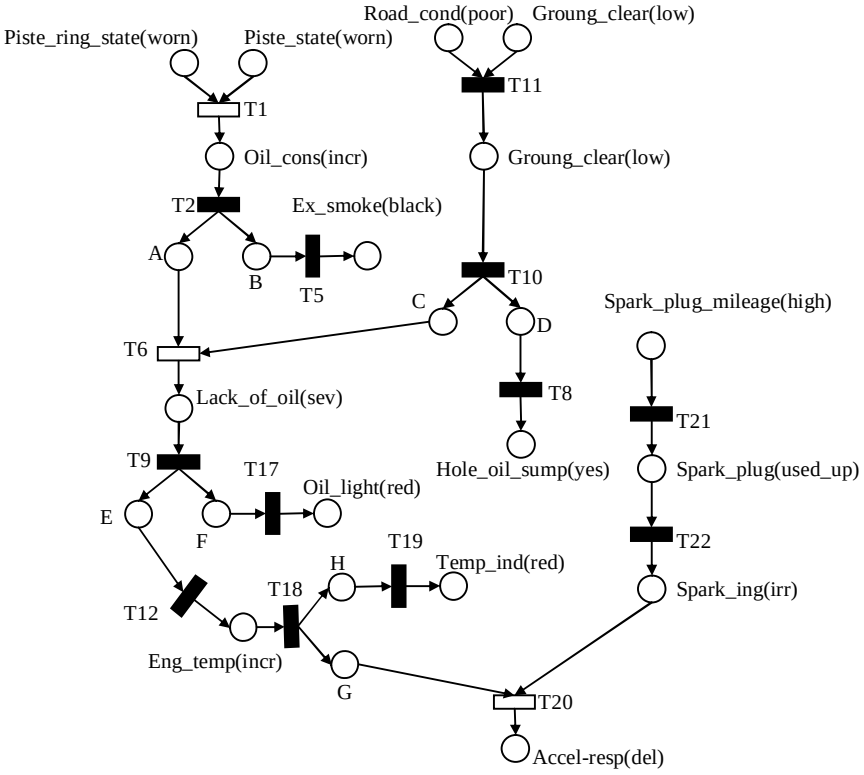


FIG. 3.4 – Exemple de modèle BPN représentant les défaillances d'un moteur

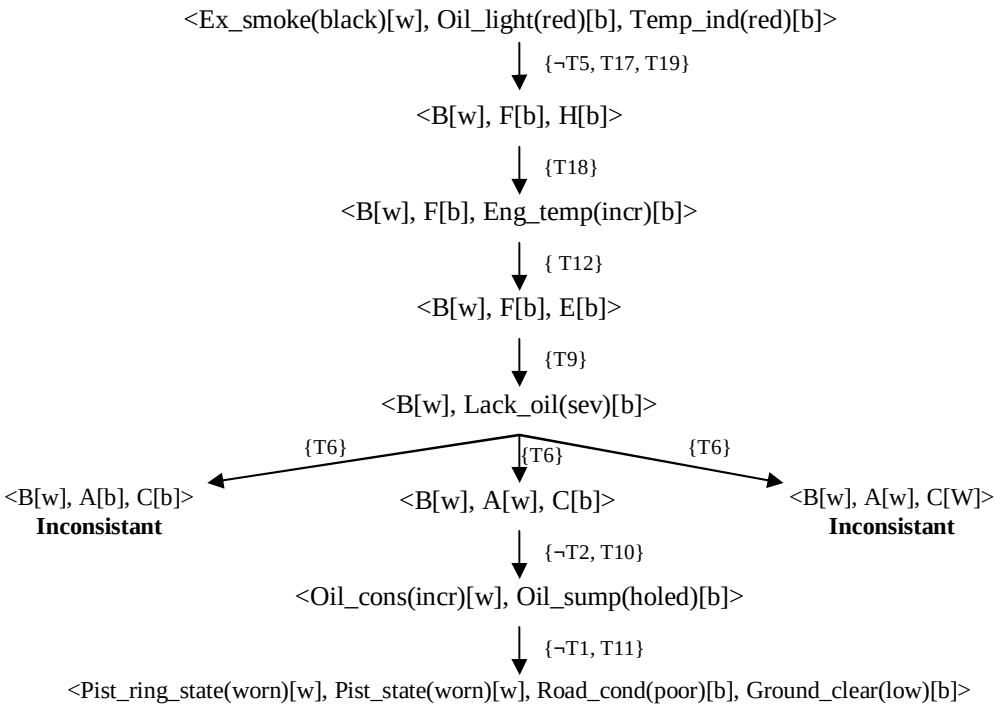


FIG. 3.5 – Résultat de l'exemple d'analyse B-W

possible l'application de méthodes d'analyse structurelle.

Dans les RdP ordinaires, il est possible d'effectuer une analyse arrière sur une transformation du RdP original. Celle-ci est appelée *dualisation*. Elle est obtenue en transposant la matrice d'incidence du RdP ce qui revient à inverser la direction des arcs ainsi qu'à changer les places en transitions et vice versa. Le problème qui se pose est comment généraliser cette approche sur des classes de RdP de haut niveau. En terme d'algèbre linéaire, si φ est une application linéaire correspondant à une matrice d'incidence M, comment trouver l'application linéaire φ^{ad} qui correspond à la matrice d'incidence du RdP dual M^t. L'application φ^{ad} est nommée application adjointe de φ.

Réseau de Petri dual

Soit un RdP ordinaire R tel que l'application linéaire φ correspond à la matrice d'incidence. Si on considère que $\varphi : V \to W$, $\varphi^{ad} : W \to V$ et $\{e_1, e_2, \ldots, e_n\}$ une base standard de V alors selon [Kow79] :

$$\varphi^{ad}(w) = \sum_{i=1}^{n} (w \bullet \varphi(e_i)).e_i$$

Dans cette formule, le symbole $'\bullet'$ désigne le produit vectoriel.

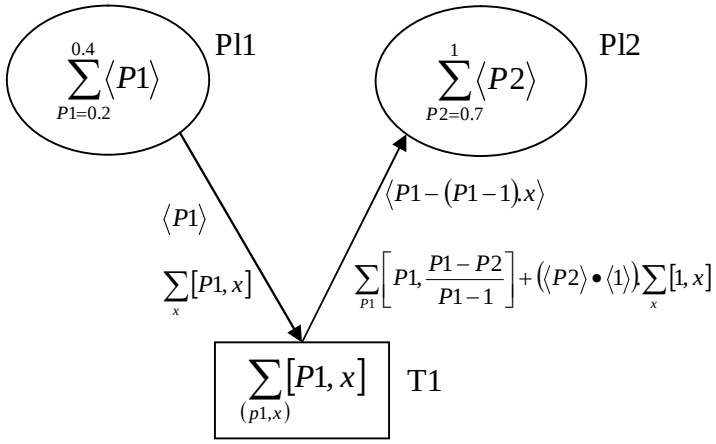


FIG. 3.6 – Réseau de Petri de haut niveau et son dual

Cette formule produit comme résultat une application linéaire (φ^{ad}) telle que la matrice correspondante représente la matrice d'incidence d'un RdP particulier appelé RdP dual. Dans le RdP dual, les transitions de R sont transformées en places, les places de R sont transformées en transitions et la direction des arcs est inversée.

Le travail de [MS07] généralise cette méthode de calcul de RdP dual vers la classe des HLPN en prenant en compte les expressions associées aux arcs et les domaines des variables. Pour y arriver, les termes w, e_i et $\varphi(e_i)$ (dans la formule de [Kow79] présentée ci-dessus) ne représentent plus des vecteurs de constantes numériques mais plutôt des vecteurs d'expressions d'arcs. L'idée fondamentale est donc de reprendre la méthode utilisée pour les RdP ordinaires et de l'appliquer aux HLPN en portant le moins de changements possibles à savoir la nature des termes utilisés dans l'expression de *dualisation*.

L'illustration Fig.3.6 fournit un exemple simple de cette méthode. Le HLPN représenté contient deux places, une transition et deux arcs. Chaque arc est associé à deux expressions : au dessus est marquée celle du réseau original et au dessous est marquée celle du réseau dual. Les expressions d'arcs du HLPN dual sont calculées par la formule précédente. Pour l'arc $(Pl1, T1)$ nous avons : $\varphi_{(Pl1,T1)}([p1, x]) = \langle p1 \rangle$ et $\varphi^{ad}_{(Pl1,T1)}(\langle p1 \rangle) = \sum_x [p1, x]$. Pour l'arc $(T1, Pl2)$, nous avons : $\varphi_{(T1,Pl2)}([p1, x]) = \langle p1 - (1 - p1).x \rangle$ et $\varphi^{ad}_{(T1,Pl1)}(\langle p2 \rangle) = \sum_{p1} [p1, \frac{p1-p2}{p1-1}] + (\langle p2 \rangle \bullet \langle 1 \rangle). \sum_x [1, x]$.

3.5.4 Travaux de Cho et al.

Le but des travaux [CHC96] est de développer une méthode de diagnostic basée sur une analyse arrière des RdPC. Cette méthode met en œuvre une approche

d'accessibilité arrière. Elle est inspirée de [LS87] qui propose une analyse des RdP ordinaires : commençant par un état considéré faute (ou erreur), les marquages prédécesseurs immédiats sont générés et chacun d'entre eux est examiné pour savoir s'il correspond à une situation critique. [CHC96] étend les sémantiques des RdPC sur deux aspects : 1) définition du marquage symbolique et 2) définition du marquage *inconditionnel*. L'accessibilité arrière est définie sur des RdPC ainsi étendus.

Un jeton symbolique est un couple $(p, \hat{v})$ tels que p est une place du RdPC et $\hat{v}$ est un symbole prenant ses valeurs dans l'ensemble des couleurs de p. Un marquage symbolique est un couple $(\hat{M}, \hat{D})$ tel que $\hat{M}$ est une distribution de jetons symboliques sur tout le RdPC et $\hat{D}$ est un prédicat sur les symboles.

Souvent, pour effectuer une analyse arrière, l'information existante (marquage défaillant) se révèle insuffisante car elle ne fournit que la distribution de jetons sur un sous ensemble du réseau. Le marquage inconditionnel enrichit l'information à propos de certaines places par des jetons symboliques prenant chacun ses valeurs dans tout le domaine de la place dans laquelle il est placé. Dans cette optique, l'ensemble des places est subdivisé en deux : le premier contenant les places dont le marquage est connu et le second celles dont le marquage est inconnu. D'une manière formelle, si $SM = (\hat{M}, \hat{D})$ est un marquage symbolique alors pour chaque transition t du RdPC sont définis deux ensembles :

- $\star$ $Out1(t, \hat{M}) = \{p | p \in Out(t) \wedge \hat{M}(p) \neq \emptyset\}$
- $\star$ $Out2(t, \hat{M}) = \{p | p \in Out(t) \wedge \hat{M}(p) = \emptyset\}$

L'algorithme d'accessibilité arrière utilise cette distinction des ensembles pour écrire une variante (inverse) de l'équation fondamentale des RdPC. Soient $SM = (\hat{M}, \hat{D})$ et $SM' = (\hat{M}', \hat{D}')$ deux marquages symboliques. SM' est dit accessible par l'arrière de SM avec l'occurrence d'une transition t si et seulement si :

- $\star$ $\forall p \in (P - Out2(t, \hat{M})) : \hat{M}'(p) = \hat{M}(p) + E(p, t) - E(t, p)$,
- $\star$ $\forall p \in Out2(t, \hat{M}) : \hat{M}'(p) = E(p, t)$,
- $\star$ $\hat{D}_{out} = \bigwedge_{p_i \in out1(t, \hat{M})} (\hat{v} = E(t, p_i))$, tels que $Out(t)$ désigne les post-conditions de t, $E(p, t)$ désigne l'expression associée à l'arc (p, t), $(p_i, \hat{v}) \in \hat{M}$.

Un exemple d'application est illustré dans Fig.3.7.a. Le RdPC contient trois places et une transition. Considérons le marquage symbolique $SM = (\hat{M}, \hat{D}) = ((p_2, \hat{x}_2), \hat{x}_2 < 10)$. Le but est de calculer le marquage $SM' = (\hat{M}', \hat{D}')$ accessible par l'arrière de SM.

- $\star$ D'abord, il faut calculer les ensembles $out1$ et $out2$
  $Out1(t, M) = \{p_2\}$
  $Out2(t, M) = \{p_3\}$
  $P - Out2(t, M) = \{p_1, p_2\}$
- $\star$ Il faut ensuite calculer le marquage M' à partir du marquage M
  - Pour chaque place dans $\{p_1, p_2\}$: $\hat{M}'(p) = \hat{M}(p) + E(p, t) - E(t, p)$

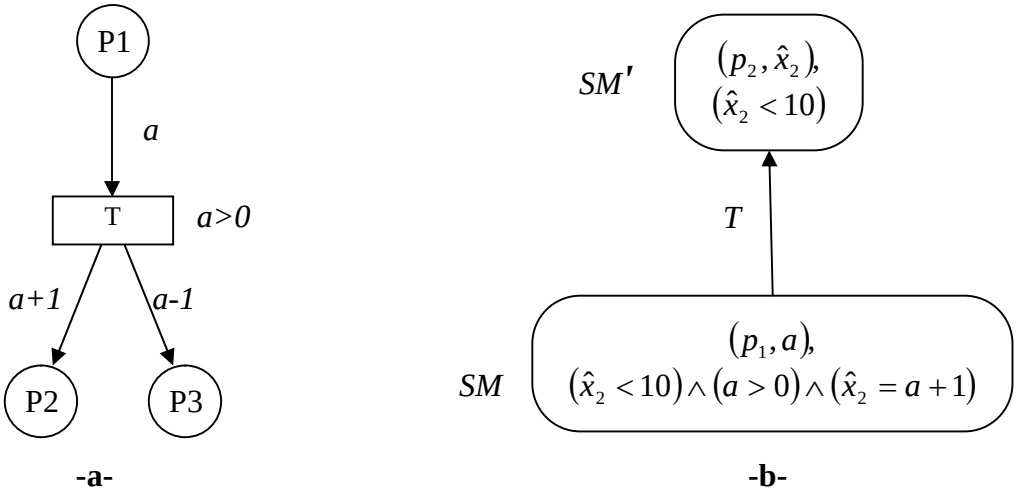


FIG. 3.7 – Exemple d'application de l'algorithme de Cho et al.

$\hat{M'}(p_1) = \hat{M}(p_1) + E(p_1, t) - E(t, p_1) = \emptyset + a - \emptyset = a$
$\hat{M'}(p_2) = \hat{M}(p_2) + E(p_2, t) - E(t, p_2) = \hat{x}_2 + \emptyset - (a + 1) = \emptyset$

- Pour chaque place dans $\{p_3\}$: $\hat{M'}(p) = E(p, t)$
  $\hat{M'}(p_3) = E(p_3, t) = \emptyset$

⋆ Il faut enfin calculer le prédicat D' à partir du prédicat D
$\hat{D'} = \hat{D} \wedge G(t) \wedge \hat{D}_{out}$ où
$\hat{D} = \hat{x}_2 < 10$ et $G(t) = (a > 0)$ et $\hat{D}_{out} = (\hat{x}_2 = a + 1)$

La représentation graphique des résultats de cette application est illustré dans Fig.3.7.b.

3.5.5 Travaux de Haddad et al.

Les travaux décrits dans cette section sont tirés de [ldMD01] et notamment le chapitre 3 qui synthétise un grand ensemble de publications portant sur l'étude et l'analyse des RdP.

Les invariants

Soit un RdP R défini par $R = (P, T, Pre, Post)$. Notons par W la *matrice d'incidence* tel que $W = Post - Pre$. La notion d'invariant dans les RdP est liée à celle des flots. Un flot d'un RdP est une quantité qui annule la matrice d'incidence d'un RdP. Il en existe différents types :

⋆ Un P-flot est un vecteur non nul v de $\mathbb{Z}^P$ tel que $v^t.W = \vec{0}$.
⋆ Un P-semiflot est un vecteur non nul v de $\mathbb{N}^P$ tel que $v^t.W = \vec{0}$.
⋆ Un T-flot est un vecteur non nul v de $\mathbb{Z}^T$ tel que $W.v = \vec{0}$.

⋆ Un T-semiflot est un vecteur non nul v de $\mathbb{N}^T$ tel que $W.v = \vec{0}$.

Dans ces travaux, l'intérêt est porté sur le P-flots. Ils seront notés *flots*.

Si un RdP possède un flot alors il possède une infinité de flots. Il suffit de multiplier le flot par n'importe quel scalaire. Les calculs se limitent alors à chercher une famille génératrice de flots dite minimale, c'est-à-dire qu'elle contient le plus petit nombre d'éléments parmi les familles génératrices.

D'une manière formelle, une famille de flots $\{v_1, \ldots, v_n\}$ définie pour un RdP R est une famille génératrice si :

$$\forall v \ flot, \exists \lambda_1, \ldots, \lambda_n \in \mathbb{Q}^n, \ v = \sum_{i=1}^{n} \lambda_i.v_i$$

L'une des méthodes qui sont utilisées pour réaliser les calculs est *l'élimination de Gauss* (aussi nommée *pivot de Gauss*). Cette méthode propose un algorithme itératif sur les transitions. Soit t la prochaine transition à examiner et $v_1, \ldots, v_n$ la famille courante. Deux cas de figures sont alors possibles. Dans le premier cas, si $\forall v_i, v_i^t.W(t) = 0$ alors la famille est inchangée. Dans le second cas, $\exists v_{i0}$ *t.q.* $v_{i0}^t.W(t) = 0$ alors, le flot v_{i0} sert de pivot pour calculer la famille génératrice $\{v_i'\}_{i \neq i0}$ avec $v_i' = (v_{i0}^t.W(t)).v_i - (v_i^t.W(t)).v_{i0}$.

Les invariants colorés

Soit un RdPC $(P, T, Pre, Post, M_0, C)$ (tel que défini dans [DA89]) où P est un ensemble fini de places, T est un ensemble fini de transitions, $C = \{C_1, C_2, \ldots\}$ est l'ensemble des couleurs, $Pre, Post$ sont des fonctions relatives aux couleurs de franchissement, M_0 est le marquage initial.

La définition des invariants colorés doit satisfaire (au moins) deux exigences. Premièrement, l'invariant d'un RdP Coloré (RdPC) doit être coloré, c'est-à-dire qu'il permet de prendre en considération la valeur des couleurs dans le RdPC. Deuxièmement, l'invariant doit être utilisable mathématiquement. Sous ces exigences, l'invariant coloré peut être défini comme une somme pondérée de places associées à un domaine de couleur : le domaine de l'invariant qui est considéré comme étant son domaine d'interprétation. Chaque poids associé aux places est une fonction du domaine de couleur de la place vers le domaine de couleur de l'invariant.

Notons par $Bag_Q(A)$ le $\mathbb{Q}$-espace vectoriel canonique sur l'ensemble non vide A, W la matrice d'incidence du RdPC ($\forall t \in T, p \in P : W(p,t) = Post(p,t) - Pre(p,t)$). Formellement, l'invariant coloré $\mathcal{F}$ est défini par la somme $\sum_{p \in P} \mathcal{F}(p)\vec{p}$ où

⋆ $\mathcal{C}(\mathcal{F})$ est le domaine de couleur de l'invariant et

⋆ $\forall p \in P, \mathcal{F}(p)$ application linéaire de $Bag_Q(\mathcal{C}(p)) \rightarrow Bag_Q(\mathcal{C}(\mathcal{F}))$ tel que : $\forall t \in T, \sum_{p \in P} \mathcal{F}(p) \circ W(p,t) = 0$

L'invariant s'interprète comme étant un ensemble d'équations liant le marquage d'un sous ensemble de places du RdPC pour tout état accessible à partir du marquage initial. Cette relation est notée comme suit :

$$\forall m \in Acc(R, m_0), \forall c \in \mathcal{C}(\mathcal{F}) : \sum_{p \in P} \sum_{c_p \in \mathcal{C}(p)} [\mathcal{F}(p)(c_p)(c)].m(p(c_p)) = constante$$

Où $Acc(R, m_0)$ désigne tout marquage m du RdPC R accessible à partir du marquage initial m_0, et $p(c_p)$ l'instance c_p de la place p.

Calcul des flots colorés

Ce qui suit aborde deux méthodes de calcul de flots dans les RdPC et plus précisément de calcul de famille génératrice de flots. La première est une extension de l'élimination de Gauss. Cette méthode est simple mais elle trouve rapidement ses limites d'où la seconde méthode se basant sur le concept de la *semi-inverse généralisée*. Cette dernière est plus générale mais elle est plus complexe et pose d'autres problèmes de calcul. Les deux méthodes seront illustrées à travers des exemples.

a) Adaptation de l'élimination de Gauss : Cette méthode adapte, d'une manière intuitive, l'élimination de Gauss aux RdPC, la différence réside dans le contenu de la matrice d'incidence. En effet, dans le cas des RdPC, cette matrice contient les expressions des arcs. Soit par exemple le RdPC de la figure 3.8 dont le domaine de couleur des places est $C = \{1, \ldots, n\}$. Sa matrice d'incidence W s'écrit comme suit :

	T1	T2	T3
$(x)\vec{P1}$	$-x$	x	0
$(x)\vec{P2}$	x	$-x$	0
$(x)\vec{P3}$	$All - x$	0	$-x$
$(x)\vec{P4}$	0	$-(All - x)$	x

Le processus d'élimination de Gauss consiste à réduire successivement W jusqu'à obtenir une matrice dont tous les éléments sont nuls. Il faut choisir à chaque étape le pivot, annuler la colonne pour pouvoir la supprimer. Dans le premier pas, le pivot choisi est $(x)\vec{P2}$ pour éliminer la première colonne. Dans le second pas, le pivot est $(x)\vec{P4}$. Nous atteignons ainsi la matrice nulle à deux lignes. Donc chaque ligne est un flot.

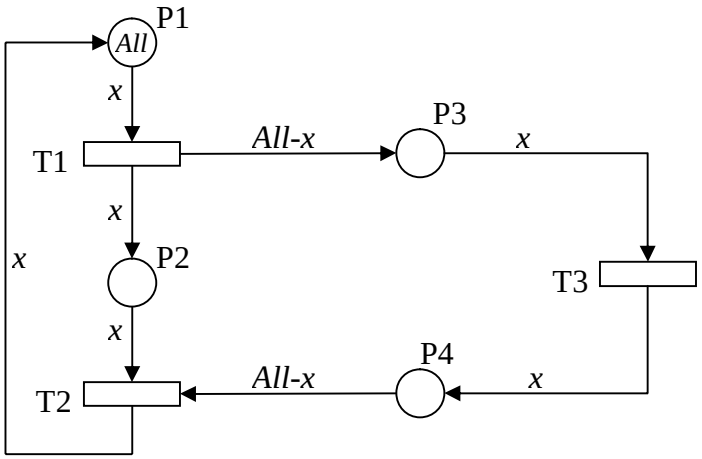


FIG. 3.8 – Exemple d'adaptation de l'élimination de Gauss

	T1	T2	T3
$(x)\vec{P1} + (x)\vec{P2}$	0	0	0
$(x)\vec{P2}$	x	$-x$	0
$(x)\vec{P3} - (All - x)\vec{P2}$	0	$All - x$	$-x$
$(x)\vec{P4}$	0	$-(all - x)$	x

$\Downarrow$

	T1	T2	T3
$(x)\vec{P1} + (x)\vec{P2}$	0	0	0
$(x)\vec{P3} - (All - x)\vec{P2}$	0	$All - x$	$-x$
$(x)\vec{P4}$	0	$-(all - x)$	x

$\Downarrow$

	T1	T2	T3
$(x)\vec{P1} + (x)\vec{P2}$	0	0	0
$(x)\vec{P3} - (All - x)\vec{P2} + (x)\vec{P4}$	0	0	0
$(x)\vec{P4}$	0	$-(all - x)$	x

$\Downarrow$

	T1	T2	T3
$(x)\vec{P1} + (x)\vec{P2}$	0	0	0
$(x)\vec{P3} - (All - x)\vec{P2} + (x)\vec{P4}$	0	0	0

Essayons maintenant d'appliquer le même procédé sur le RdPC illustré par Fig.3.9. Nous débuterons par la matrice d'incidence, puis nous appliquerons l'élimination de Gauss.

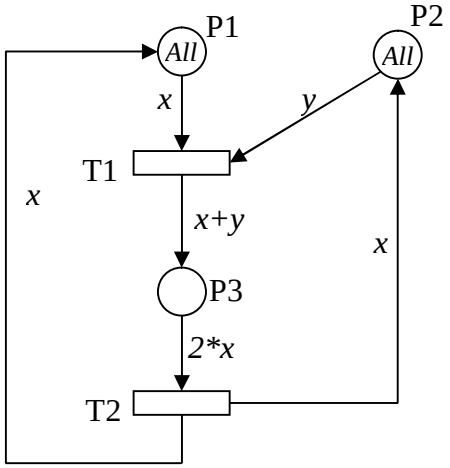


FIG. 3.9 – Exemple d'élimination de Gauss augmentée

$$\begin{array}{c|cc} & \text{T1} & \text{T2} \\ \hline (x)\vec{P1} & -x & x \\ (x)\vec{P2} & -y & x \\ (x)\vec{P3} & x+y & -2x \end{array} \Rightarrow \begin{array}{c|cc} & \text{T1} & \text{T2} \\ \hline (x)\vec{P1}-(x)\vec{P2} & -x+y & 0 \\ 2(x)\vec{P1}+(x)\vec{P3} & -x+y & 0 \end{array} \Rightarrow$$

$$\begin{array}{c|cc} & \text{T1} & \text{T2} \\ \hline (x)\vec{P1}-(x)\vec{P2} & -x+y & 0 \\ (x)\vec{P1}+(x)\vec{P2}+(x)\vec{P3} & 0 & 0 \end{array}$$

Un problème se pose dans la dernière matrice obtenue. En effet, la fonction $y-x$ n'est pas injective donc il n'est pas possible de continuer le processus. Nous aurons un résultat similaire si nous changeons l'ordre des éliminations. Pour résoudre ce problème, une autre règle est ajoutée à l'élimination de Gauss. Elle se base sur la notion de semi-inverse généralisée.

b) Élimination de Gauss augmentée : Pour appliquer cette méthode, il faut d'abord introduire la notion de *Semi-inverse généralisée* [Nas76] : soit f une application de $Bag_Q(C_1)$ dans $Bag_Q(C_2)$. Il existe une application linéaire non unique h de $Bag_Q(C_2)$ dans $Bag_Q(C_1)$ telle que $f \circ h \circ f = f$. Une telle application est appelée *semi-inverse généralisée* de f. De plus $Id - f \circ h$ est une solution génératrice de l'équation $X \circ f = 0$.

Le principe de cette méthode est d'utiliser la notion de semi-inverse généralisée pour augmenter le nombre d'éléments nuls dans une matrice. Cette opération est effectuée grâce à une transformation de la matrice l'incidence (au départ) et des matrices intermédiaires pendant le calcul. Le principe de cette transformation est illustré sur une matrice W dont la première colonne est T et W_1 est la sous matrice définie par les autres colonnes.

	T				T	
$\mathcal{F}_1$	f_1			$\mathcal{F}_1 - \mathcal{F} \circ f_1 \circ h$	$f_1 - f_1 \circ h \circ g$	
$\vdots$	$\vdots$			$\vdots$	$\vdots$	
$\mathcal{F}_k$	f_k			$\mathcal{F}_k - \mathcal{F} \circ f_k \circ h$	$f_k - f_k \circ h \circ g$	
$\mathcal{F}$	ⓖ	W_1	$\Rightarrow$	$\mathcal{F} - \mathcal{F} \circ g \circ h$	⓪	W'_1
$\mathcal{F}'_1$	0			$\mathcal{F}'_1$	0	
$\vdots$	$\vdots$			$\vdots$	$\vdots$	
$\mathcal{F}'_q$	0			$\mathcal{F}'_q$	0	

Chaque ligne $\mathcal{F}_i, i = 1, \ldots, k$ est retranchée de la ligne $\mathcal{F}$ composée avec $f_i \circ h$. Dans le cas particulier de la ligne $\mathcal{F}$, le retranchement se fait de $\mathcal{F} \circ g \circ h$. Ceci fait apparaître l'élément nul à la place de g car $g \circ h \circ g = g$. Par itération de cette opération, tous les éléments de la matrice peuvent ainsi être annulés et donc les flots déduits. Notons que le calcul d'une semi-inverse d'une expression peut conduire à limiter la taille des classes de couleur du RdPC.

Appliquons cette méthode à l'exemple du RdPC de la figure 3.9 dont la matrice d'incidence W est la suivante :

	T1	T2
$(x)\vec{P1}$	$-x$	x
$(x)\vec{P2}$	$-y$	x
$(x)\vec{P3}$	$x+y$	$-2x$

Considérons l'application linéaire $g : (x, y) \mapsto x + y$. Une semi-inverse de g peut être l'application $h : (x) \mapsto \frac{1}{2}(x, x)$. En effet $g \circ h \circ g(x, y) = g \circ h(x + y) = g(\frac{1}{2}(x + y, x + y)) = x + y = g$. Posons $f_1 : (x, y) \mapsto -x$ et $f_2 : (x, y) \mapsto -y$. Nous avons alors :

- $\star$ $f_1 \circ h \circ g : (x, y) \mapsto -\frac{1}{2}(x + y)$
- $\star$ $f_2 \circ h \circ g : (x, y) \mapsto -\frac{1}{2}(x + y)$
- $\star$ $x - (-2)x \circ f_1 \circ h = 0_{Bag(C) \to Bag(C)}$

La transformation de la matrice W donne la matrice W' qui s'écrit comme suit :

	T1	T2
$(x)\vec{P1} + \frac{1}{2}(x)\vec{P3}$	$\frac{1}{2}(y - x)$	0
$(x)\vec{P2} + \frac{1}{2}(x)\vec{P3}$	$\frac{1}{2}(x - y)$	0

Pour continuer, nous devons appliquer une seconde fois le même processus. Posons l'application $g : (x, y) \mapsto \frac{1}{2}(x - y)$. Une semi-inverse de g pourrait être $h : x \mapsto 2(x - \alpha)$ telle que α peut prendre n'importe quelle valeur de l'ensemble C. Nous avons alors :

- $\star$ $g \circ h : (x, y) \mapsto x - \alpha$
- $\star$ $h \circ g : (x, y) \mapsto (x - \alpha, y - \alpha)$

$\star$ $g \circ h \circ g : (x, y) \mapsto \frac{1}{2}(x - \alpha - y + \alpha)$

La transformation de la matrice W' donne la matrice W'' qui s'écrit comme suit :

	T1	T2
$(x)\vec{P1} + (x-\alpha)\vec{P2} + (x-\alpha)\vec{P3}$	0	0
$(\alpha)\vec{P2} + \frac{1}{2}(\alpha)\vec{P3}$	0	0

La matrice W'' étant nulle, chaque ligne correspond à un flot et appartient à la famille génératrice des flots.

L'utilisation pratique de cet algorithme pose certains problèmes. D'abord, le calcul de la semi-inverse peut conduire à déplier partiellement le RdPC. Ceci limite la taille des domaines de couleurs et pose un problème de connaissance a priori sur la possibilité de calcul de la famille génératrice. Le second problème réside dans le fait que cette méthode introduit parfois, dans les flots, des coefficients qu'il est difficile d'interpréter.

3.5.6 Synthèse sur les méthodes d'inversion des RdP et RdPC

L'inversion des modèles RdP ordinaires ainsi que leurs extensions est basée sur différentes techniques suivant le besoin et l'extension. Dans le domaine des RdP ordinaires, les travaux menés au LAAS proposent d'utiliser les modèles RdP inverses. Ceux-ci sont obtenus par une inversion de la direction des arcs. Les avantages de cette méthodes sont sa simplicité ainsi que la mise en œuvre de démarche de recherche de scénarios redoutés sur des modèles RdP PTDS en appliquant l'inversion sur des modèle originaux plus simples. L'inconvénient de cette méthode, par rapport à notre travail, réside dans les limites rapidement atteintes quand cette démarche est appliquées au RdPC. Les travaux de Portinale proposent une analyse par accessibilité arrière sur des modèles introduisant une sémantique (réduite) de différentiation de jeton. Néanmoins, l'extension proposée reste très particulière et limitée en expressivité. Dans la classe des RdPC, Cho et al. propose une version modifiée de l'équation de base d'évolution des RdPC. Elle permet d'exprimer un tir inverse d'une transition. et permet aussi de prendre en compte la notion de tir de transition même en cas de manque de certains jetons. Le problème de cette méthode est l'association de chaque tir inverse à un prédicat ce qui pourrait ralentir l'analyse, voire la rendre impossible à cause de l'introduction de l'indécidabilité systématique sur chaque transition. Dans un contexte très proche des RdPC, Muller et al. proposent une approche applicable sur les RdP haut niveau (HLPN). L'approche consiste à généraliser, vers les HLPN, une formule de dualisation initialement définie pour les modèles RdP ordinaires. Cette méthode transforme le RdP en changeant les transitions en places et vice versa. L'interprétation physique

des phénomènes modélisés devient plus délicate et l'application pratique de cette méthode montre que l'apport principal de la dualisation réside dans la limitation de certains intervalles de calcul mais l'analyse en elle même se faisait toujours sur les HLPN originaux.

3.6 Conclusion

Ce chapitre montre quelques facettes de la modélisation basée sur les modèles RdP et notamment RdPC. L'accent est mis sur les différentes approches d'analyse de ces modèles : simulation, formelle, structurelle. La séparation entre ces deux dernières n'est pas nette dans le sens où des connexions existent entre elles. En effet, des approches structurelles sont utilisées par exemple pour réduire la complexité des calculs formels et l'équivalence est établie entre l'accessibilité et la preuve de séquents [32].

La problématique centrale de cette thèse s'inscrit dans la thématique d'analyse structurelle de modèles RdPC par accessibilité arrière reprenant le modèle d'inversion de RdP de [Kha03]. Une telle analyse introduit une démarche déductive de l'analyse des RdPC et peut constituer une alternative ou un complément aux analyses formelles palliant une partie de la complexité des modèles et/ou de l'explosion combinatoire.

Les verrous et défis à relever sont de proposer une généralisation de la méthode proposées pour les RdP ordinaires vers les RdP Colorés. Cette généralisation doit se conformer à la sémantique des RdPC et doit garder un lien avec les méthodes formelles. D'où les deux axes de travail qui sont : élaboration d'algorithmes de transformation de RdPC et étude d'outils de preuve applicables sur ces transformations ainsi que sur le RdPC transformé.

[32] Voir chapitre 5.

Chapitre 4

Accessibilité arrière

Sommaire

Comme évoqué dans le chapitre 3, les deux axes développés dans notre travail sont d'ordre algorithmique et théorique. Le volet algorithmique, traité dans ce chapitre, consiste à proposer une méthode de transformation des RdPC généralisant l'inversion des modèles et puis proposer une adaptation des mécanismes nécessaires à la conduite de l'analyse structurelle par accessibilité arrière (comme l'enrichissement du marquage).

De même que dans les RdP ordinaires, l'accessibilité arrière dans le domaine des RdPC est un concept dual de l'accessibilité avant, c'est-à-dire, si un marquage

M_f est accessible à partir de M_0, on dira que M_0 est accessible en arrière de M_f. L'expression *accessibilité arrière* signifie que M_0 est une cause ou source de M_f.

Une inversion appliquée sur les RdP ordinaires peut être généralisée aux RdPC en deux étapes : 1) inversion de l'orientation des arcs et 2) adaptation de la sémantique du RdPC. Le résultat de ces deux étapes est un *RdPC inverse*. Se contenter d'appliquer la première étape[33] peut conduire à un RdPC dont l'analyse est difficile voire impossible. De plus, l'accessibilité arrière utilise un certain nombre de mécanismes complémentaires pour pouvoir conduire l'analyse. Ces mécanismes sont intuitifs dans le cas des RdP ordinaires, mais ils le sont moins dans le cas des RdPC du fait de l'évolution de la valeur des jetons en fonction des expressions des arcs et des gardes de transitions.

Dans ce chapitre, nous considérerons $R = (\Sigma, P, T, A, N, C, G, E, I)$ comme le RdPC original et $R' = (\Sigma', P', T', A', N', C', G', E', I')$ le RdPC inverse obtenu par application des transformations nécessaires sur R. Ces transformations dépendent directement de la structure du RdPC. Par conséquent, une transformation spécifique doit être définie pour chaque cas de structure étudiée. Néanmoins, les transformations les plus communes peuvent être regroupées. Dans ce chapitre, chaque groupe sera présenté à part. Les mécanismes complémentaires servant à la conduite d'une analyse en accessibilité arrière utilisant le RdPC inverse seront détaillées après les transformations. Viendront ensuite quelques pistes de travaux qui n'ont pas abouti à des résultats satisfaisants mais qui soulignent la difficulté du sujet traité. Notons dans ce qui suit que E est écrit dans une forme scindée $(Pre, Post)$ ainsi que $E' = (Pre', Post')$.

4.1 Transformations élémentaires

Les transformations qui suivront (transformations triviale, basique, mixte, parallèle, paramétrée) sont des changements de structure appliqués aux RdPC et aboutissant aux RdPC inverses. Leur utilité est de permettre la réalisation d'analyses par accessibilité arrière. Chacune des transformations est un algorithme générique applicable sur un type de transitions possédant des caractéristiques communes [BBS09b], [BBS09c]. Ces caractéristiques sont :

- ⋆ la nature des expressions associées aux arcs (constante, variable, fonction),
- ⋆ le nombre de fonctions utilisant une même variable (aucune, une, deux ou plus),
- ⋆ l'orientation de l'arc (entrant ou sortant)

L'association de ces trois critères donne 81 cas possibles. Ce nombre décroît très rapidement car beaucoup de ces associations sont obsolètes tels que les fonc-

[33] Appliquer la première étape revient à inverser la direction des arcs. Ceci constituerait une généralisation directe de la méthode d'inversion telle qu'elle est énoncée pour les RdP ordinaires

tions en entrées, les variables en sortie[34], et les fonctions n'ayant pas de variable en entrée. Un deuxième critère réduit encore plus le nombre de ces associations : le nombre de constantes et le nombre d'arcs entrants marqués par la même variable n'influe pas sur le type de la transformation à appliquer. Ce qui influe donc c'est leur présence ou leur absence. À la fin, sur les 81 cas initialement possibles, il n'en reste que 5 qui sont : constante en entrée et constante en sortie (transformation triviale), variable en entrée et aucune fonction l'utilisant en sortie (transformation paramétrée), variable en entrée et une seule fonction l'utilisant en sortie (transformation basique) et enfin variable en entrée et plusieurs fonctions l'utilisant en sortie (transformation parallèle). Des associations de ces types sont, bien évidement possibles. Un exemple est donné dans ce sens, il s'agit de la transformation mixte.

4.1.1 Transformations triviale et basique

Transformation triviale

L'illustration Fig.4.1.a montre un cas trivial d'inversion de RdPC. Les arcs entrant et sortant sont associés à des constantes a et b respectivement. Dans ce cas, il est suffisant de généraliser la définition d'inversion du RdP aux RdPC ce qui donne :

$$\begin{cases} Pre'(p2,t) = Post(p2,t) \\ Post'(p1,t) = Pre(p1,t) \end{cases}$$

Considérons, par exemple, le marquage $M_0 = \langle a \rangle.p1$ [35] du RdPC illustré dans Fig.4.1.a. L'évolution dynamique possible de ce RdPC est $M_0[t\rangle M_1$ [36] tel que $M_1 = \langle b \rangle.p2$. Dans le RdPC inverse, l'evolution du marquage $M_1[t\rangle M_0$ est facilement vérifiable. En effet, le tir de la transition t avec le marquage $M_1 = \langle b \rangle.p2$ produit le marquage $M_0 = \langle a \rangle.p1$. Ainsi, l'accessibilité arrière est vérifiée en utilisant le RdPC inverse.

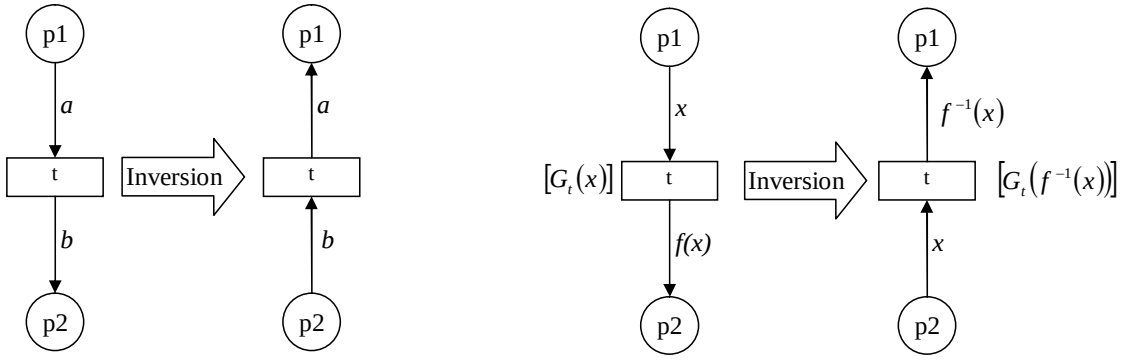
Transformation basique

Fig.4.1.b montre un cas basique où l'arc entrant est associé à une variable x, l'arc sortant est associé à une fonction $f(x)$ et une garde $G(x)$ est associée à la transition t. L'application de la règle précédente (inversion de la direction des arcs) pourrait aboutir à la construction d'un réseau incorrect. Dans cet exemple, l'arc entrant serait marqué avec une fonction tandis que l'arc sortant serait marqué

[34] Une variable associée à un arc sortant est considérée comme une fonction identité.

[35] La notation $M = \langle ValJeton \rangle.p$ signifie que la place p est marquée par un jeton de valeur $ValJeton$.

[36] La notation $M_i[t\rangle M_j$ signifie que le tir de la transition t fait évoluer le RdPC du marquage M_i vers le marquage M_j



-a- **-b-**

FIG. 4.1 – Transformations triviale et basique pour inversion de RdPC

avec la variable de cette fonction ce qui ne permettrait pas d'évaluer la fonction. C'est pourquoi la suggestion est de marquer l'arc entrant avec une variable, l'arc sortant avec l'inverse de la fonction f (noté f^{-1}) et de mettre à jour la garde pour exprimer la nouvelle contrainte associée au tir de la transition. Ceci suppose la nécessité de savoir si la fonction f est inversible. Si oui, il est nécessaire de définir son inverse (noté f^{-1}). Cette transformation donne :

$$\begin{cases} Pre'(p2,t) = Pre^{-1}(p1,t) \\ Post'(p1,t) = Post^{-1}(p2,t) \\ G'_t(Pre'(p2,t)) = G_t(Post^{-1}(p2,t)) \end{cases}$$

$Pre^{-1}(p1,t)$ dénote la transformation de l'arc $Pre(p1,t)$ associé à la nouvelle variable définie dans le domaine de f^{-1}. $Post^{-1}(p2,t)$ dénote la transformation de l'arc $Post(p2,t)$ associé à la fonction f^{-1} (l'inverse de $Post(p2,t)$).

Exemple : Dans l'illustration de Fig.4.1.b, soient les fonctions $f(x) = 2x + 5$ et $G(x) = G_t(x) = (x < 8)$. Dans le RdPC inverse, nous aurons :

$$\begin{cases} Pre'(p2,t) = Pre^{-1}(p1,t) \Rightarrow Pre'(p2,t) = x \\ Post'(p1,t) = Post^{-1}(p2,t) \Rightarrow f^{-1}(x) = (x-5)/2 \\ G'_t(Pre'(p2,t)) = G_t(Post^{-1}(p2,t)) \Rightarrow G'(x) = G((x-5)/2) = (x < 20) \end{cases}$$

En accessibilité avant, nous avons $M_0[t\rangle M_1$ tels que $M_0 = \langle 4\rangle.p1$ et $M_1 = \langle 13\rangle.p2$. En accessibilité arrière (en utilisant donc le RdPC inverse), la relation $M_1[t\rangle M_0$ est vérifiée.

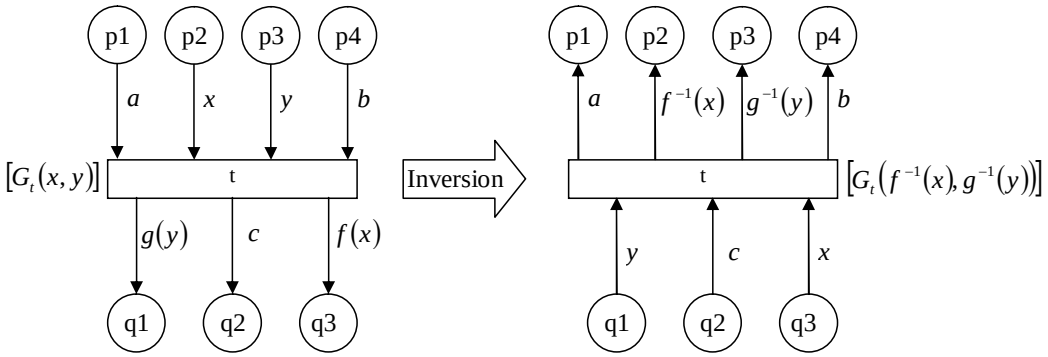


FIG. 4.2 – Transformations mixtes pour inversion de RdPC

4.1.2 Transformations mixtes

Fig.4.2 montre un cas mixte où certains arcs entrants sont associés à des variables $\{x, y\}$ et les autres à des constantes $\{a, b\}$. Les arcs sortants sont associés à des constantes $\{c\}$ et à des fonctions inversibles $\{f, g\}$. Cette inversion de RdPC est une généralisation des transformations triviale et basique. Pour les arcs marqués avec des constantes, il suffit de généraliser la règle appliquée aux RdP ordinaires (inverser la direction des arcs). Pour le reste des arcs, il faut appliquer la règle illustrée sur Fig.4.1.b. Il faut donc associer chaque variable à la fonction qui l'utilise pour respecter les places origines des arcs marqués avec les variables et les places puits recevant les jetons résultant des fonctions. Pour pouvoir le faire, la contrainte suivante doit être vérifiée : chaque variable est utilisée par une et une seule fonction. Si cette condition n'est pas vérifiée, la transformation devient soit paramétrée soit parallèle (voir plus loin). L'algorithme décrivant la transformation mixte peut être écrit comme suit :

- ⋆ $Pre'(qj, t) = Post(qj, t)$, $j = 1 \dots m$ si $Post(qj, t)$ est associé à une constante,
- ⋆ $Post'(pi, t) = Pre(pi, t)$, $i = 1 \dots n$ si $Pre(pi, t)$ est associé à une constante,
- ⋆ $Pre'(qj, t) = Pre^{-1}(pi, t)$, $i = 1 \dots n$, $j = 1 \dots m$ si $Pre(pi, t)$ est associé à une variable et cette variable est utilisée par la fonction $Post(qj, t)$,
- ⋆ $Post'(pi, t) = Post^{-1}(qj, t)$, $i = 1 \dots n$, $j = 1 \dots m$ si $Post(qj, t)$ est associé à une fonction et cette fonction utilise la variable $Pre(pi, t)$,
- ⋆ $G'_t(Pre'(qj, t)) = G_t(Post^{-1}(qj, t))$, $i = 1 \dots n$, $j = 1 \dots m$ si $Post(qj, t)$ est associé à une fonction et cette fonction utilise la variable $Pre(pi, t)$.

Exemple : Dans l'illustration de Fig.4.2, considérons arbitrairement que $a = 1$, $b = 0$ et $c = 2$ et soient les fonctions $f(x) = 2x+5$, $g(x) = x^2$ et $G(x) = (x+2y < 15)$. Dans le RdPC inverse, nous aurons : $f^{-1}(x) = (x - 5)/2$, $g(x) = \sqrt{x}$ et $G'(x, y) = G((x - 5)/2, \sqrt{x}) = (\frac{x-5}{2} + 2\sqrt{x} < 15)$.

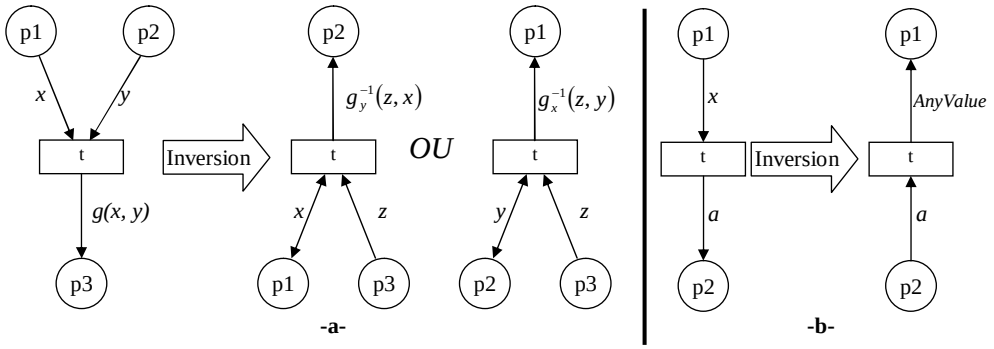


FIG. 4.3 – Transformations paramétrées pour inversion de RdPC

En accessibilité avant, nous avons $M_0[t\rangle M_1$ tels que $M_0 = \langle 1\rangle.p1 + \langle 3\rangle.p2 + \langle 4\rangle.p3 + \langle 0\rangle.p4$ et $M_1 = \langle 16\rangle.q1 + \langle 2\rangle.q2 + \langle 11\rangle.q3$. Il est aisé de vérifier qu'en accessibilité arrière la relation $M_1[t\rangle M_0$ en utilisant le RdPC inverse.

4.1.3 Transformations paramétrées

Certaines structures des RdPC ne peuvent pas être inversées pour retrouver un marquage de manière déterministe en accessibilité arrière. La raison est que le processus d'inversion peut être assimilé à des opérations mathématiques dont les solutions sont des intervalles. L'inversion du RdPC devient alors *paramétrée*. Ceci implique l'introduction d'informations additionnelles pour pouvoir effectuer l'inversion. Deux exemples sont décrits ci-après : le premier traite le cas d'une fonction (associée à l'arc sortant) à plusieurs variables, le second traite le cas de variables en entrée non associées à des fonctions en sortie.

Fig.4.3.a montre le cas d'une fonction à plusieurs variables. Dans ce cas, le RdPC n'est pas directement inversible. Les valeurs des variables $\{x, y\}$ ne peuvent pas être déduites en ne connaissant que la valeur de $g(x, y)$. La solution *partielle (paramétrée)* proposée consiste à trouver la valeur d'une variable connaissant celles des autres variables et le résultat de la fonction. Comme montré dans Fig.4.3.a, le tir de t dans le RdPC inverse requiert des jetons soit dans $\{p2,p3\}$ ou dans $\{p1,p3\}$. Les fonctions $g_x^{-1}(z, y)$ et $g_y^{-1}(z, x)$ sont des inverses partielles de $g(x, y)$.

Fig.4.3.b montre le cas d'une variable qui n'est pas associée à une fonction. Notons juste que l'arc entrant est associé à une variable et l'arc sortant à une constante. L'inversion de ce RdPC est similaire à la transformation basique (illustrée sur Fig.4.3.a) à une exception : l'arc $Post'(p1, t)$ est associée à l'expression paramétrée $AnyValue$. Cette expression d'arc peut prendre toute valeur en sortie de t.

4.1.4 Transformations parallèles

Le terme *'parallèle'* signifie l'existence d'une variable partagée tel qu'utilisée dans [LT07]. Fig.4.4 montre le cas où la même variable est utilisée par plus d'une fonction (deux fonctions dans ce cas). Pour inverser ce RdPC, il faut calculer l'inverse d'une seule fonction utilisant la variable partagée (ceci suppose l'existence d'au moins une fonction inversible). Soit f cette fonction inversible. Dans le RdPC original (illustré à gauche de Fig.4.4), le tir de la transition t produit deux jetons (dans $p2$ et $p3$). Chaque valeur de jeton résulte d'une fonction appliquée à la variable partagée : fonction f au jeton dans $p2$ et fonction h au jeton dans $p3$. Le RdPC inverse doit produire un jeton dans $p1$ en tirant t dont les préconditions sont $p2$ et $p3$. Mais le fait d'avoir des jetons dans $p2$ et $p3$ n'est pas suffisant pour tirer t car les jetons (dans $p2$ et $p3$) doivent avoir des valeurs cohérentes vis-à-vis des fonctions f et h. Pour cette raison, une *garde* est associée à la transition t. Elle vérifie que la valeur de $f^{-1}(x)$ appliquée à la fonction h renvoie bien la valeur de y (jeton initialement dans $p3$). Si la valeur de la garde est *Vrai*, t est tirée et le marquage initial de $p1$ est retrouvé. Si la valeur de la garde est *Faux*, t n'est pas tirée pour cause d'incohérence des valeurs des jetons en entrée.

Pour illustrer cette transformation, étudions l'exemple de Fig.4.4 tel que $f(x) = x + 4$, $h(x) = (x > 10)$, le marquage initial $M_0 = \langle 5 \rangle.p1$ et le marquage final $M_f = \langle 9 \rangle.p2 + \langle Faux \rangle.p3$. Nous remarquons que $M_0[t\rangle M_f$. Dans le RdPC inverse, on a $f^{-1}(x) = x - 4$.

Le premier test consiste à appliquer l'accessibilité arrière à M_f utilisant le RdPC inverse (Fig.4.4). Le résultat attendu est le marquage initial M_0.

$$M_f = \langle 9 \rangle.p2 + \langle Faux \rangle.p3$$
$$t \downarrow$$
$$M_0 = \langle 5 \rangle.p1$$

Notons que la garde renvoie la valeur $'Vrai'$. Cette valeur est obtenue en appliquant la formule $[y == h(f^{-1}(x))]$ qui donne $['Faux' == (5 > 10)]$.

Le second test consiste à appliquer l'accessibilité arrière à un marquage final inaccessible M'_f. Le résultat attendu est de trouver un ensemble vide de marquages initiaux.

$$M'_f = \langle 9 \rangle.p2 + \langle True \rangle.p3$$
$$t \downarrow$$
$$\{\}$$

La notation $\{\}$ désigne un ensemble de marquages vide (pour cause de tir impossible). Notons que le tir est impossible car la garde renvoie la valeur $'Faux'$. Cette valeur est obtenue en appliquant la formule $[y == h(f^{-1}(x))]$ qui donne $['Vrai' == (5 > 10)]$.

L'algorithme décrivant la transformation parallèle peut être écrit comme suit : Soit R un RdPC contenant une variable x ($Pre(p,t) = x$) partagée par n fonctions

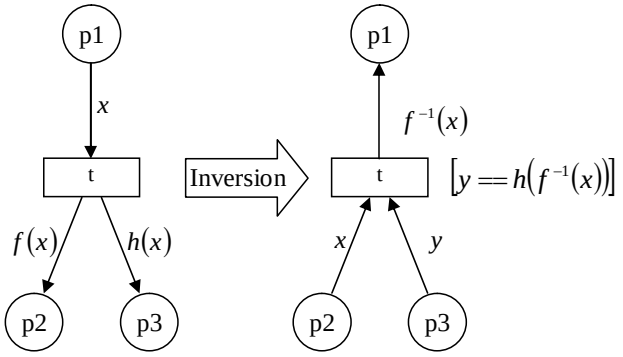


FIG. 4.4 – Transformation parallèle pour inversion de RdPC

$f_1, f_2, \ldots, f_n$ ($Post(qi,t) = f_i(x), i = 1 \ldots n$) tel que f_1 est inversible. Le RdPC inverse est défini par :

- ⋆ $Pre'(q1,t) = Pre^{-1}(p,t)$,
- ⋆ $Pre'(qi,t) = Id_i, i = 2 \ldots n$,
- ⋆ $Post'(p,t) = Post^{-1}(q1,t)$,
- ⋆ $G'_t = \bigwedge_n^{i=2}[Pre'(qi,t) == f_i(f_1^{-1}(Pre'(q1,t)))]$.

4.2 Problèmes inhérents à la structure du modèle

Dans l'infinité de possibilités de modèles qui existent en RdPC, et notamment les RdPC généralisés (concernés par cette étude), un certain nombre de cas peuvent poser des problèmes à l'inversion. Ces problèmes sont inhérents à la structure même du modèle. Dans cette section, nous discuterons des cas les plus fréquents.

4.2.1 Transformations complexes

Les cas des transformations présentées précédemment dans ce chapitre sont *"simples"*, facilement identifiables et transformables. Mais, dans les modèles réels, des transitions regroupent des caractéristiques qui ne leurs permettent pas d'être catégorisées dans un type de transformation unique (ex. transition paramétrée et parallèle à la fois). La transformation de telles transitions est un mélange d'inversions élémentaires. Pour ce faire, il faudrait tout d'abord identifier tous les types d'inversion à appliquer en fonction de la structure de la transition. Ensuite, les appliquer dans un ordre pertinent. Un autre point complique encore cet état de fait : la non-unicité de transformation de certaines transitions. Souvent, la procédure commence par la transformation parallèle, suivie par la paramétrée et finit par la

mixte. Cet ordre n'est qu'une indication générale. Chaque cas étant unique, il faut procéder au cas par cas et proposer l'inversion la plus pertinente, à savoir, celle qui exploite le maximum d'informations disponibles et qui répond au mieux aux besoins exprimés. Dans les cas réels, l'information dynamique disponible influence le choix structurel de la transformation à appliquer (Voir des exemples pratiques dans le chapitre 6 et le chapitre 7).

4.2.2 Présence de cycles

Les modèles sous-jacents aux systèmes embarqués fonctionnent en cycles ou contiennent des sous-systèmes cycliques. La caractérisation de ces cycles peut être très difficile dans le cas des RdPC du fait de l'évolution de la valeur des jetons qui y circulent. Dans le cas des RdP ordinaires, [Sad07] minimise le nombre de chemins critiques retournés par l'accessibilité arrière grâce à la détection des cycles et l'optimisation de l'algorithme d'analyse pour n'analyser le cycle qu'une seule fois et capitaliser cette connaissance pour les détections de cycles suivantes. Cette méthode est difficilement généralisable dans le cadre des RdPC car deux exécutions successives du même cycle peut produire des jetons avec des valeurs complètement différentes. Des cas spéciaux existent (comme les boucles assimilables à des fonctions monotones ou périodiques) mais dans le cas général la difficulté de prédiction des résultats de la boucle fait que son exécution doit être complète (pas de concept de chemin minimum exécutant une boucle une seule fois). Néanmoins, une solution intermédiaire peut être trouvée dans certains cas en utilisant les agglomérations de transitions [Had89] qui composent la boucle quand ceci est possible.

4.2.3 Fonctions non inversibles

L'inversion de modèle suppose souvent l'existence d'au moins une fonction inversible. Celle-ci donne à partir des valeurs connues, celles qui les ont engendrées. Dans certains cas, la fonction en question peut être inversible par morceaux, inversible sur une partie de l'ensemble de couleurs, inversible partiellement ou pseudo-inversible. Le problème se pose si la fonction inversible n'existe pas. Une des solutions envisageables est la programmation par contraintes [FA03]. Cette solution pose la relation entre les sorties et les entrées sous formes de contraintes au lieu qu'elle soit une relation directe. L'introduction des contraintes dans le modèle implique l'introduction de la notion de décidabilité. En effet, si les contraintes sont résolues, la solution existe forcément. Par contre, si les contraintes ne sont pas résolues, la décision d'existence ne peut pas être catégorique (c'est, par exemple, le temps d'analyse imparti qui n'est pas assez long). Un autre problème sous-jacent à cette solution est la qualité du solveur utilisé qui influe directement sur le temps d'analyse.

4.3 Concepts complémentaires pour l'analyse arrière

Les règles de transformation présentées précédemment permettent l'inversion de la structure du RdPC, c'est-à-dire qu'elles permettent d'obtenir une structure sur laquelle une analyse arrière peut être conduite. Cette analyse se heurte parfois à un manque d'informations traduit par la connaissance partielle d'un état du système. En effet, dans certaines situations, le marquage final (ou intermédiaire) ne permet pas une analyse complète. C'est pourquoi, dans le but d'analyser le comportement dynamique inverse, des concepts complémentaires sont proposés.

4.3.1 Transition potentiellement franchissable

Dans un RdPC inverse, une transition avec plusieurs places préconditions est potentiellement franchissable si elle possède au moins : 1) une place précondition marquée avec un jeton dont la valeur est compatible avec le tir de la transition (i.e. un jeton compatible avec l'expression d'arc et la garde) ; 2) une place précondition qui n'est pas marquée avec un jeton dont la valeur est compatible avec le tir de la transition ou qui n'est pas marquée du tout. Ce genre de transition nécessite l'application de l'*Enrichissement de marquage*

4.3.2 Enrichissement de marquage

L'enrichissement de marquage est un mécanisme qui permet de compléter une information dynamique d'un modèle en ajoutant des hypothèses à propos de l'état du système modélisé (jetons additionnels). Cette opération est effectuée en calculant les états prédécesseurs. En effet, il arrive qu'un prédécesseur immédiat d'un état donné (ce qui revient au tir d'une transition du RdPC inverse) peut être construit seulement à l'aide de jetons additionnels ayant les bonnes valeurs aux bonnes places. La signification de l'enrichissement du marquage est de supposer que le système a atteint un certain état global contenant l'état partiel connu. Les questions qui se posent sont quand et comment enrichir un marquage ?

L'enrichissement est une opération *très délicate*. Il faut y recourir le moins possible pour exploiter au mieux l'information connue au détriment de celle supposée. Cette opération se pratique exclusivement sur les transitions potentiellement franchissables et permet, si possible, au modèle inverse d'évoluer jusqu'à atteindre les états initiaux. L'enrichissement ne doit pas apporter une incohérence au modèle. Dans le cas des RdP ordinaires, la cohérence après chaque ajout de jeton est vérifiée grâce au calcul de l'invariant [Kha03]. L'invariant, dans le cas des RdPC étant difficile voire impossible à calculer. Ce moyen de vérification devient par conséquent inefficace. Le meilleur moyen, jusqu'à présent, dans le cas

coloré est l'expertise se rapportant au système physique qui permettra d'établir la cohérence ou non de l'état obtenu après enrichissement. Enfin pour minimiser le nombre d'enrichissements, ceux-ci doivent être effectués suivant l'ordre partiel de tir des transitions [AP94].

4.3.3 Ordre partiel de tir de transitions

L'ordre partiel dans les RdP désigne une relation dans un ensemble de transitions (n'incluant pas forcément toutes les transitions du RdP) notée dans [Por93] par le symbole "$\prec$". Cette relation définit une précédence dans le tir des transition. Pour deux transitions $t1$ et $t2$, la formule $t1 \prec t2$ signifie que le tir de $t1$ survient avant le tir de $t2$ (les propriétés de cette relation sont détaillée dans [Por93]).

La notion d'ordre partiel est primordiale dans les analyses par accessibilité arrière pour désigner les transitions à tirer à chaque étape de l'évolution de l'analyse. L'ordre partiel de tir des transitions, dans l'accessibilité arrière, se trouve inversé par rapport à l'accessibilité avant : si dans un RdP la relation $t1 \prec t2$ est vraie alors dans le RdP inverse la formule $t2 \prec t1$ est également vraie. [Kha03] utilise cette notion pour ordonner les événements dans un chemin redouté. Pour satisfaire l'ordre de tir, il arrive de décider soit de faire évoluer une partie du RdP pour sensibiliser la transition souhaitée, soit d'enrichir le marquage (choix des places auxquelles il faudrait rajouter des jetons pour conduire l'analyse arrière). Nous nous tiendrons dans le cadre des RdPC à cet usage en gardant la même définition de cette notion.

4.4 Pistes supplémentaires

Cette section est un aperçu de certaines réflexions et certains travaux menés lors de cette thèse mais qui ont été abandonnées faute de temps. Leur présentation sert juste à souligner quelques difficultés liées à l'analyse inverse des RdPC ou, plus important, constituer une base d'efforts futurs dans cette voie.

4.4.1 Transformation de boucles

Les modèles visés dans ces études montrent une présence récurrente de petites boucles (représentant souvent des *timers*). Leurs forme minimale est une boucle entre une place et une transition représentant un simple *timer* ou échantillonneur. Pour éviter la répétition des itérations, nous proposons l'utilisation de suites mathématiques, et notamment, les suites arithmétiques et géométriques. Elles permettent de déterminer, en une opération unique, la valeur finale de la boucle après un nombre d'itérations donné ou, par dualité, de calculer le nombre d'itérations nécessaires pour atteindre une valeur sans simuler la boucle.

Les boucles concernées ont une progression comparable à des suites. Par exemple, sur Fig.4.5.a, nous pouvons considérer le marquage initial $\langle U_0 \rangle.p$. Les tirs successifs de t produisent des jetons dont les valeurs sont $U_1 = f(U_0)$, puis $U_2 = f(U_1) = f(f(U_0))$, ..., $U_n = f(U_{n-1}) = \underbrace{f(\ldots(f}_{n\ fois}(U_0)\ldots)$. Les valeurs $U_i, i = 1, \ldots, n$ peuvent être considérées comme des termes d'une suite et peuvent ainsi être déduites facilement.

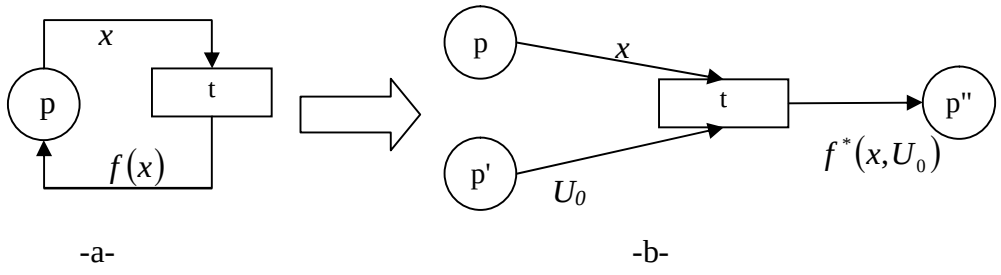


FIG. 4.5 – Transformation de boucle élémentaire

Dans le cas de cette boucle élémentaire, le RdPC est transformé par duplication de la place constituant la boucle. Pour cet exemple, le résultat est illustré sur Fig.4.5.b. La place p est dupliquée en p, p' et p''. La fonction $f(x)$ est transformée en $f^*(x, U_0)$. La fonction f^* retourne le nombre nécessaire d'itérations (étapes) la valeur initiale du jeton dans p.

Si, par exemple, $f(x)$ est écrite sous la forme $f(x) = x + b$ (x est une variable numérique et b est une constante numérique) alors $f^*(x, U_0) = \frac{x - U_0}{b}$ donne la valeur de p". U_0 est la valeur initiale dans p' et elle constitue la valeur d'entrée de la boucle. Notons que pour avoir la forme littérale de f^*, il suffit de considérer f, qui caractérise la boucle, comme une suite arithmétique dont la raison est b et la valeur initiale U_0.

L'algorithme décrivant la transformation de boucle peut être écrit comme suit :

Soit R un RdPC et $p \in P$, $t \in T$ les éléments d'une boucle (élémentaire) dont la progression est caractérisée par une suite. Dans le RdPC inverse nous aurons :

- ⋆ la duplication de p en p, p' et p",
- ⋆ $Pre'(p, t) = Pre(p, t)$,
- ⋆ $Pre'(p', t) = U_0$ tel que U_0 est une variable qui prend ses valeurs dans le domaine de p,
- ⋆ $Post(p", t) = f^*(x, U_0)$ tel que $Post(p, t) = f(x)$.

Certains des problèmes liés aux transformations de boucles sont discutés dans la section 4.2.2. En plus, de la difficulté de caractérisation des cycles eux-mêmes, ceux-ci ne sont pratiquement jamais autonomes dans un modèle RdPC : ils interagissent avec le reste du modèle. Ce qui accroît considérablement leurs traitement et les rend inefficaces.

4.4.2 Transformations conditionnelles

Cette transformation concerne les cas où les expressions associées aux arcs en sortie sont des fonctions conditionnelles. Ceci veut dire que les expressions des arcs sont écrites sous la forme *"Si ... Alors ... Sinon ..."* (Fig.4.6.a).

Cette transformation procède en deux étapes. La première déplie le RdPC pour avoir un des éléments plus simple que l'expression conditionnelle. Quand cette étape est appliquée à l'exemple illustré dans Fig.4.6.a, la fonction f est séparée en f_1, f_2 et f_3 telles que $f_1(x) = x > 10$ (de type booléen), $f_2(x) = x + 4$ et $f_3(x) = x - 2$. Le RdPC déplié est illustré dans Fig.4.6.b.

Le dépliage de l'expression conditionnelle est utile car produisant des structures simples à inverser. En effet, $t2$ et $t3$ de Fig.4.6.b sont des transformations mixtes avec $f_2(x) = f_3(x) = x$ comme fonctions inversibles dont les fonctions inverses sont $f_2^{-1}(x) = f_3^{-1}(x) = x$. L'inversion de la transition $t1$ est parallèle. Le RdPC inverse est montré dans Fig.4.6.c.

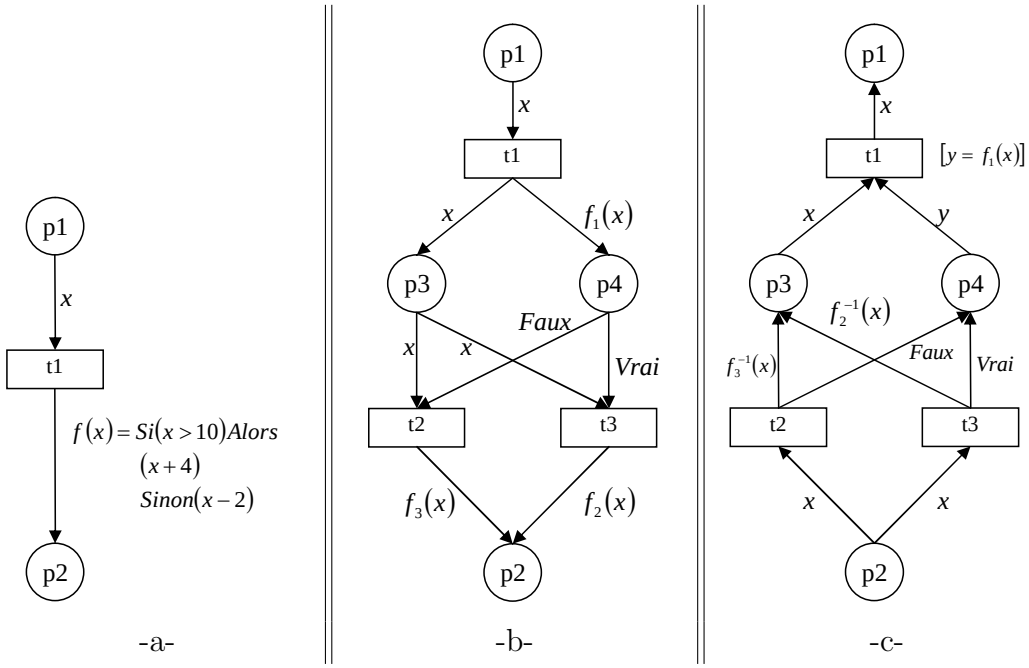


FIG. 4.6 – Transformation conditionnelle

Exemple 1 Cet exemple considère le cas de Fig.4.6 avec deux marquages initiaux M_0 et M_0' tels que $M_0 = \langle 10 \rangle.p1$ et $M_0' = \langle 11 \rangle.p1$. Le tir de $t1$ (Fig.4.6.a) donne :

$$M_0 = \langle 10 \rangle.p1 \xrightarrow{t1} M_f = \langle 8 \rangle.p2$$
$$M_0' = \langle 11 \rangle.p1 \xrightarrow{t1} M_f' = \langle 15 \rangle.p2$$

Vérifions l'équivalence des résultats avec le RdPC illustré dans Fig.4.6.b. Le même marquage initial doit forcément aboutir au même marquage final.

$$\begin{aligned} M_0 = \langle 10\rangle.p1 &\xrightarrow{t1} \langle 10\rangle.p3 + \langle Faux\rangle.p4 \\ &\xrightarrow{t2} \langle 8\rangle.p2 = M_f \\ M_0 = \langle 11\rangle.p1 &\xrightarrow{t1} \langle 11\rangle.p3 + \langle Vrai\rangle.p4 \\ &\xrightarrow{t3} \langle 15\rangle.p2 = M'_f \end{aligned}$$

L'application de l'accessibilité arrière a M_f et M'_f utilisant le RdPC inverse donne :

$$\begin{array}{c} M_f = \langle 8\rangle.p2 \\ t2 \swarrow \quad \| \quad \searrow t3 \\ \langle 10\rangle.p3 + \langle Faux\rangle.p4 \quad \| \quad \langle 4\rangle.p3 + \langle vrai\rangle.p4 \\ t1 \downarrow \quad \| \quad \downarrow t1 \\ \langle 10\rangle.p1 = M_0 \quad \| \quad \{\} \end{array}$$

$$\begin{array}{c} M'_f = \langle 15\rangle.p2 \\ t2 \swarrow \quad \| \quad \searrow t3 \\ \langle 13\rangle.p3 + \langle False\rangle.p4 \quad \| \quad \langle 11\rangle.p3 + \langle vrai\rangle.p4 \\ t1 \downarrow \quad \| \quad \downarrow t1 \\ \{\} \quad \| \quad \langle 11\rangle.p1 = M'_0 \end{array}$$

Ces deux exécutions montrent que pour chaque marquage final considéré, la correspondance avec le marquage initial est trouvée.

Exemple 2 Cet exemple montre l'impossibilité de trouver un marquage initial qui correspondrait à un marquage final inaccessible. Pour ce faire, cet exemple utilise le même RdPC que le précédent exemple et considère un marquage final inaccessible $M_f = \langle 9\rangle.p2$. Comme dans l'exemple précédent, le marquage initial correspondant à M_f est recherché par accessibilité arrière.

$$\begin{array}{c} M_f = \langle 9\rangle.p2 \\ t2 \swarrow \quad \| \quad \searrow t3 \\ \langle 11\rangle.p3 + \langle Faux\rangle.p4 \quad \| \quad \langle 5\rangle.p3 + \langle Vrai\rangle.p4 \\ t1 \downarrow \quad \| \quad \downarrow t1 \\ \{\} \quad \| \quad \{\} \end{array}$$

Le résultat montre qu'aucun marquage initial ne correspond à M_f. Ce résultat est logique puisque M_f est choisi de telle manière qu'il soit inaccessible.

Exemple 3 Le but de cet exemple est de montrer un cas où deux marquages initiaux M_0 et M'_0 aboutissent au même marquage final M_f tel que $M_0 \rightarrow M_f$ execute la clause *ALORS* et $M'_0 \rightarrow M_f$ execute la clause *SINON*. Dans ce cas, l'analyse en accessibilité arrière doit donner deux chemins distincts menant de M_f

vers le marquage initial. Pour ce faire, reprenons l'exemple 1 avec les modifications suivantes : $f_1(x) = x > 10$, $f_2(x) = x - 4$ et $f_3(x) = x + 2$; et $M_0 = \langle 15 \rangle.p1$ et $M'_0 = \langle 9 \rangle.p1$. Dans ces conditions $M_0 \xrightarrow{ALORS} M_f = \langle 11 \rangle.p2$ et $M'_0 \xrightarrow{SINON} M_f = \langle 11 \rangle.p2$. Cherchons, par accessibilité arrière, les marquages initiaux correspondants à $M_f = \langle 11 \rangle.p2$. Le tir de $t1$. Par accessibilité arrière utilisant le RdPC inverse, nous aurons :

$$
\begin{array}{c}
M_f = \langle 11 \rangle.p2 \\
t2 \swarrow \quad \| \quad \searrow t3 \\
\langle 9 \rangle.p3 + \langle Faux \rangle.p4 \quad \| \quad \langle 15 \rangle.p3 + \langle Vrai \rangle.p4 \\
t1 \downarrow \quad \| \quad \downarrow t1 \\
\langle 9 \rangle.p1 = M'_0 \quad \| \quad \langle 15 \rangle.p1 = M_0
\end{array}
$$

Le résultat montre que les deux marquages initiaux possibles correspondants à M_f sont retrouvés par accessibilité arrière (ce qui était le but).

Au delà de la structure de boucle ou la structure conditionnelle, la démarche proposée ici va plus loin que l'inversion des arcs et la caractérisation des expressions associées aux arcs. L'inversion se base sur une transformation complète de la structure du RdPC pour faire apparaître des entités dont le traitement est maîtrisé.

4.4.3 Implémentation des transformations

Un logiciel reprenant des concepts d'analyse par accessibilité arrière a été mis en œuvre. Il est programmé en Java en utilisant l'environnement de programmation MDWorkbench[37]/Eclipse[38] pour les aspects programmation associés à l'outil WindowBuilder[39] pour les interfaces graphiques. L'utilisation de Java permet une Implémentation avec des structures de données dynamiques (listes, vecteurs, etc) et permet d'appeler facilement des applications externes (à travers le package Shell[40] de Java). Cet aspect est très utile pour exécuter du code ML et récupérer les résultats sans passer par des fichiers tampons.

L'illustration Fig.4.7 donne une vue d'ensemble sur la structure du logiciel. L'entrée principale du programme est un fichier XML crée par l'outil *CPN Tools* qui représente la structure du RdPC. Le programme d'analyse contient deux parties principales : un analyseur syntaxique/créateur de structures de données et un moteur de calcul. La première partie analyse la syntaxe du fichier (XML) entrée pour y extraire la structure et les propriétés du RdPC et créer les données manipulées en interne. Le moteur de calcul met en œuvre l'approche développée pour

[37] www.mdworkbench.com/
[38] www.eclipse.org/
[39] www.instantiations.com/windowbuilderpro/
[40] adiguba.developpez.com

l'analyse structurelle par accessibilité arrière. Il répond aux requêtes utilisateur en applicant une partie (ou la totalité) des transformations vues dans ce chapitre : basique, triviale, mixte, paramétrée, parallèle, complexes, et boucles et permet de tester leur application.

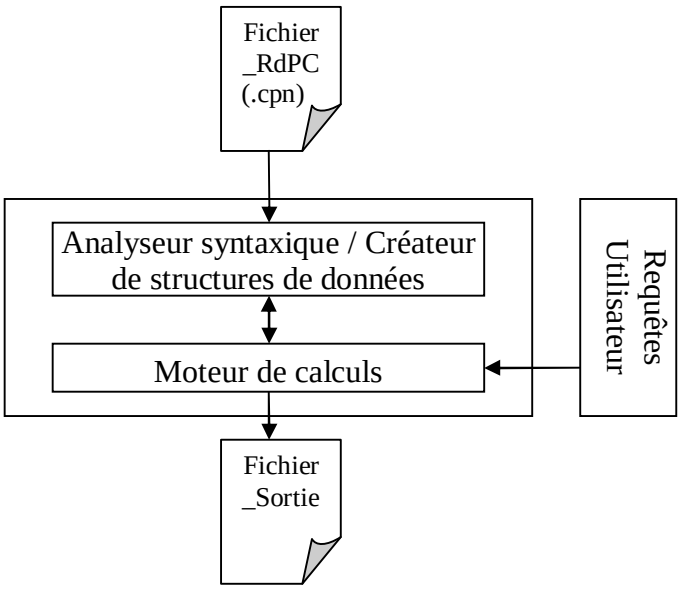


FIG. 4.7 – Structure schématique du logiciel

4.5 Conclusion

Dans ce chapitre nous nous sommes intéressés à l'aspect algorithmique de l'analyse structurelle par accessibilité arrière. Cet aspect se décompose en deux parties : les algorithmes de transformation et la mise en œuvre de l'analyse arrière. Les transformations élémentaires considérées regroupent des ensembles de structures possédants des caractéristiques communes. Elles sont loin d'être exhaustives mais elles peuvent être vues comme des briques de base permettant de définir de nouvelles transformations en considérant simplement les différentes associations possibles. Cet aspect d'association de transformations permet d'augmenter la panoplie des algorithmes de transformations et, par conséquent, d'augmenter considérablement le champ applicatif de la méthode proposée.

La mise en œuvre de l'analyse arrière requiert quelques mécanismes complémentaires. Ceux-ci sont empruntés du domaine des RdP ordinaires et sont soit définis tels quels dans les RdPC (comme l'ordre partiel de tir de transitions), soit adaptés pour refléter les spécificités de la sémantique liée eu RdPC.

La suite du travail traitera l'aspect théorique permettant de donner une base formelle aux algorithmes proposés. Elle contient également des illustrations d'uti-

lisation pratiques de l'approche proposée permettant une prise en main de la méthode proposée et mettant l'accent sur les apports pratiques.

Chapitre 5

Preuves des transformations

Sommaire

L'analyse structurelle par accessibilité arrière utilise le RdPC inverse. Celui-ci est obtenu à l'aide de transformations structurelles sur le RdPC original. L'interrogation porte alors sur deux aspects : 1) les algorithmes des transformations structurelles présentés précédemment sont-ils viables ? et 2) l'analyse arrière effectuée grâce au RdPC inverse est-elle correcte ?

La recherche de la réponse à ces deux questions nous a conduit à explorer deux aspects théoriques de l'inversion des RdPC. Le premier, l'algèbre linéaire, porte sur l'inversion d'une seule transition. Elle permet de fournir un socle formel aux algorithmes de transformations. Pour cela, nous nous sommes inspirés de [Had89] concernant la théorie des RdPC et plus précisément les réseaux de Petri *bien formés* (*WPN : Well-Formed Petri Nets*). Le second aspect est la logique linéaire. Elle établit une équivalence entre l'accessibilité et la preuve de séquents. Elle permet de prouver formellement la justesse des chemins trouvés à l'aide des RdPC inverses. Pour l'utiliser dans notre étude, nous allons étendre cette méthode aux domaines colorés ce qui implique la redéfinition des règles d'inférence sur des classes de langages d'ordre supérieur que ce qui est utilisé dans les travaux dont nous nous sommes inspirés ([Kha03], [VK94], [Lin92]).

5.1 Algèbre linéaire

5.1.1 Définitions et préliminaires

Les définitions qui suivent dont tirées principalement de [Had89], mais aussi de [Gob95], [ldJPR00] et [CE05].

Multi-ensembles

Soit U un ensemble fini. La notation $Bag(U) = \mathbb{N}^U$ désigne le multi ensemble défini sur U. Un Bag (ou *multi-ensemble*) est un ensemble non ordonné où la répétition est permise. L'introduction des multi ensembles permet de traiter l'une des caractéristiques des RdPC stipulant qu'une place peut contenir un ensemble non ordonné de jetons dont certains peuvent avoir des valeurs identiques.

- ⋆ Un élément de $Bag(U)$ est noté $\sum_{u \in U} a_u.u$.
- ⋆ La somme de deux éléments de $Bag(U)$ est définie par $a + b = \sum_{u \in U} (a_u + b_u).u$.
- ⋆ Un ordre partiel dans $Bag(U)$ est défini par la formule : $a \geq b$ si et seulement si $\forall u \in U, a_u \geq b_u$.
- ⋆ La différence entre deux éléments a et b (si $a \geq b$) de $Bag(U)$ est définie par $a - b = \sum_{u \in U} (a_u - b_u).u$.

Forme bilinéaire

Une *forme* désigne une application d'un espace dans son corps de nombres K. Le corps de nombre désigne l'ensemble des nombres définissant la multiplication externe des vecteurs tels que les réels ou les complexes. Une *forme bilinéaire* est une application définie sur un couple de vecteurs x et y, son espace de départ est le produit cartésien de deux espaces E et F ayant le même corps de nombres. Pour une application donnée f, on écrit :

$$\begin{aligned} f : \quad & E \times F \to K \\ & (x, y) \to f(x, y) \end{aligned}$$

La forme est dite linéaire pour sa première variable si pour tout y_0, l'application f qui à x associe $f(x, y_0)$ est linéaire. De même, la forme est dite linéaire pour sa deuxième variable si pour tout x_0, l'application f qui à y associe $f(x_0, y)$ est linéaire. Si les deux propriétés précédentes sont vérifiées, alors la forme est dite bilinéaire.

Similitudes

Considérons l'espace E muni d'une forme bilinéaire $\phi : E \times E \to K$. Soit une application linéaire définie sur $E \times E$. On définit la forme bilinéaire symétrique

ϕ_f par

$$\begin{array}{rl} \phi_f : & E \times E \to K \\ & (x, y) \to \phi(fx, fy) \end{array}$$

On dit que f est une *similitude* de multiplicateur μ (noté aussi $\mu(f)$) si la propriété suivante est vérifiée :

$$\forall x \in E, \forall y \in E, \phi(fx, fy) = \phi_f(x, y) = \mu\phi(x, y)$$

Un résultat engendré par cette définition est que les applications orthogonales sont des similitudes (de multiplicateur 1)

Équivalence des formes linéaires et bilinéaires

Dans un RdPC $R = (P, T, C, Pre, Post, M)$, une fonction de couleur peut être définie soit de $Bag(C(t))$ dans $Bag(C(p))$ ou de $C(t) \times C(p)$ dans $\mathbb{N}$ ($p \in P$, $t \in T$, $\mathbb{N}$ est l'ensemble des entiers naturels). Les deux formes sont utiles pour définir les transformations, c'est pourquoi, le même symbole de fonction est utilisé pour les deux formes. La formule qui donne la relation entre les deux formes s'écrit comme suit :

$$f(c) = \sum_{c' \in C(p)} f(c', c).c'$$

Où $f(c)$ désigne l'image de c dans un élément de $Bag(C(p))$ par f en tant qu'application linéaire et $f(c', c)$ désigne l'image de (c', c) dans une valeur entière. Notons qu'il n'y a pas de confusion entre les deux écritures. En effet, la première forme prend un seul argument, tandis que la seconde prend deux arguments.

Composition

Soit f une fonction de $Bag(C)$ dans $Bag(C')$ et g une fonction de $Bag(C')$ dans $Bag(C'')$. La composition de f et g est une fonction $g \circ f$ de $Bag(C)$ dans $Bag(C'')$ définie par :

$$g \circ f(c) = g(f(c)) = \sum_{c'' \in C''} (\sum_{c' \in C'} g(c'', c').f(c', c)).c''$$

Propriétés

⋆ La fonction *identité* de $Bag(C)$ (notée Id) est définie par

$$Id(c) = c$$

Cette fonction peut aussi être définie par

$$Id(c', c) = \begin{cases} 1 & \text{Si c = c'} \\ 0 & \text{Sinon} \end{cases}$$

⋆ Une fonction f de $Bag(C)$ dans $Bag(C)$ est *orthonormale* si et seulement s'il existe une substitution σ dans C tel que

$$f(c) = \sigma(c)$$

La seconde définition donne

$$f(c', c) = \begin{cases} 1 & \text{Si } \sigma(c) = c' \\ 0 & \text{Sinon} \end{cases}$$

Cette condition peut également être écrite sous la forme d'une similitude [EHPP05] :

$$\forall c \in C, \exists c' \in C', f(c, c') = 1 \text{ et } \forall c' \in C', \exists c \in C, f(c, c') = 1$$

⋆ La *projection* de $Bag(C \times C')$ dans $Bag(C)$ (notée $Proj$) est définie par

$$Proj(\langle c, d \rangle) = c$$

La seconde définition donne

$$Proj(c', \langle c, d \rangle) = \begin{cases} 1 & \text{Si c = c'} \\ 0 & \text{Sinon} \end{cases}$$

⋆ Une fonction f de $Bag(C)$ dans $Bag(C')$ est *quasi-injective* si et seulement si

$$\forall c' \in C', \forall c_1 \in C, \forall c_2 \in C : f(c', c_1) \neq 0 \wedge f(c', c_2) \neq 0 \Rightarrow c_1 = c_2$$

⋆ Une fonction f de $Bag(C)$ dans $Bag(C')$ est *unitaire* si et seulement si

$$\forall c' \in C', \forall c \in C : f(c', c) = 0 \vee f(c', c) = 1$$

Orthonormalisation des transitions

Soit un Réseau de Petri Coloré $R = (P, T, C, Pre, Post, M)$, t une transition de R et f une fonction orthonormale de $C(t)$. Le RdPC R_r obtenu par $f - orthonormalisation$ de t est défini par :

⋆ $P_r = P$
⋆ $T_r = T$
⋆ $\forall t \in T_r, \forall p \in P_r, C_r(t) = C(t)$ et $C_r(p) = C(p)$
⋆ $\forall t' \in T_r - \{t\}, \forall p \in P_r, Post_r(p, t') = Post(p, t')$ et $Pre_r(p, t') = Pre(p, t')$
⋆ $\forall p \in P, Pre_r(p, t) = Pre(p, t) \circ f$ et $Post_r(p, t) = Post(p, t) \circ f$
⋆ $\forall p \in P_r, M_r(p) = M(p)$

Une illustration de ce procédé est donnée dans Fig.5.1.

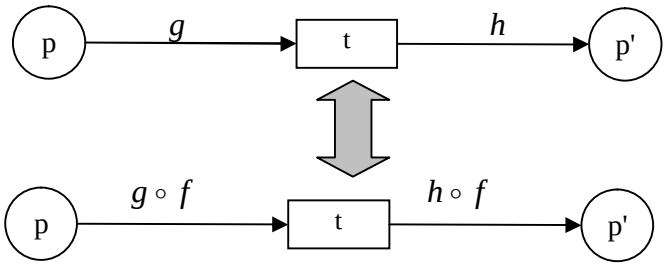


FIG. 5.1 – Exemple d'orthonormalisation de transition

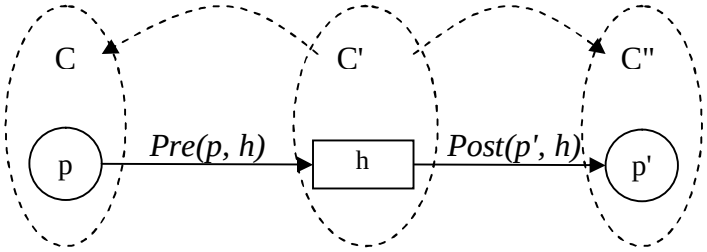


FIG. 5.2 – Mapping des couleurs dans un RdPC

5.1.2 Mapping des couleurs dans un Réseau de Petri Coloré

Considérons le cas illustré dans la Fig.5.2. Le RdPC considéré contient les places p et p' ainsi que la transition h tels que la précondition de h est p et la postcondition de h est p'. La fonction orthonormale $Pre(p,h)$ est définie de $Bag(C(h))$ dans $Bag(C(p))$. La fonction $Post(p',h)$ est définie de $Bag(C(h))$ dans $Bag(C(p'))$. Nous avons alors : $Pre(p,h) : C' \rightarrow C$ et $Pre(p',h) : C' \rightarrow C''$. Le but est d'exprimer la relation entre C et C''.

La fonction $Pre(p,h)$ est orthonormale, c'est-à-dire qu'il existe une substitution σ dans C tel que $f(c) = \sigma(c)$. À partir de cette substitution, la substitution inverse σ^{-1} est définie. Cette définition est possible grâce à une partie de la condition d'orthonormalité qui est $\forall c' \in C', \exists c \in C, f(c,c') = 1$. La substitution inverse σ^{-1} est associée à une nouvelle fonction de couleur notée Pre^{-1} définie de $Bag(C(p))$ dans $Bag(C(h))$. Les relations $Pre(p,h)$ et $Post(p,h)$ peuvent être exprimées comme suit :

$$\begin{cases} Pre^{-1}(p,h)(c) = \sum_{c' \in C(h)} Pre(p,h)(c',c).c' & \text{pour } c \in C(p) \ldots (1) \\ Post(p',h)(c') = \sum_{c'' \in C(p')} Post(p',h)(c'',c').c'' & \text{pour } c' \in C(h) \ldots (2) \end{cases}$$

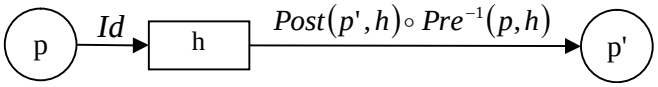


FIG. 5.3 – Exemple d'interprétation d'un RdPC

En remplaçant c' dans (1) par (2). Nous obtenons :

$$\begin{aligned}
&\textstyle\sum_{c'\in C(h)} Pre^{-1}(p,h)(c',c).\sum_{c''\in C(p')} Post(p',h)(c'',c').c'' \\
&= \textstyle\sum_{c''\in C(p')} c''.Post(p',h)(c'',c').\sum_{c'\in C(h)} Pre^{-1}(p,h)(c',c) \\
&= \textstyle\sum_{c''\in C(p')} c''.\sum_{c'\in C(h)} Post(p',h)(c'',c').Pre^{-1}(p,h)(c',c) \\
&= Post(p',h)\circ Pre^{-1}(p,h)(c'',c)
\end{aligned}$$

Donc, la relation entre C et C'' s'exprime sous la forme $Post(p',h)\circ Pre^{-1}(p,h)$ définie de C dans C''. D'où l'équivalence du schéma illustré dans Fig.5.2 avec celui illustré en Fig.5.3

Ce résultat peut être obtenu par orthonormalisation de la transition h avec la fonction orthonormale $Pre^{-1}(p,h)$. En effet, cette opération compose $Pre(p,h)$ avec $Pre^{-1}(p,h)$ et compose aussi $Post(p',h)$ avec $Pre(p,h)$. Notons que $Pre(p,h)\circ Pre^{-1}(p,h)$ est égal à l'identité (Id), d'où le résultat.

Puisque les deux RdPC (Fig.5.2 et Fig.5.3) sont équivalents, nous définissons une relation (notée $\diamond$) telle que $f\diamond g = g\circ f^{-1}$. Cet opérateur permet d'exprimer le fait que f est associée à l'arc entrant et g est est associée à l'arc sortant. Ceci nous permettra de manipuler aisément les opérations symboliques appliquées aux matrices d'incidences du RdPC.

RdPC généralisés et RdPC bien formés

Pour des raisons pratiques, nous travaillerons sur deux classes de RdPC. La première, les RdPC généralisés, est très utile pour modéliser des systèmes réels ou réalistes. Elle est mise en œuvre dans des logiciels industriels tel que *CPN Tools*. Dans cette classe, les arcs entrants sont associés à des variables et les arcs sortants sont associés à des fonctions des variables entrantes. Cette classe est équivalente en expressivité à la classe des RdP bien formés [Had89]. Elle permet de mettre en place des preuves formelles. Dans cette classe, la transition peut prendre ses valeurs dans un ensemble construit par un produit cartésien $C = C_1 \times \cdots \times C_n$ où chaque C_i est un ensemble de départ pour une fonction d'arc $Pre_i = Proj_i(C)$ noté aussi $Proj_i$ (ou $Pre_i = constante$) projetant les valeurs de couleurs de la transition à sa i^{eme} place precondition. La (ou les) variable(s) de la fonction de sortie prend sa (ses) valeur(s) dans C. Fig.5.4 illustre un exemple concret de passage d'une classe à l'autre. $C = C(T) = C_1 \times C_2 \times C_3 \times C_4$. La fonction f_1 prend ses valeurs dans C_1 tel que $C_1 = Proj_1(C)$ (qu'on notera $Proj_1$). De la même manière, f_2 prend ses valeurs dans C_2 qui est une projection de C sur sa seconde composante.

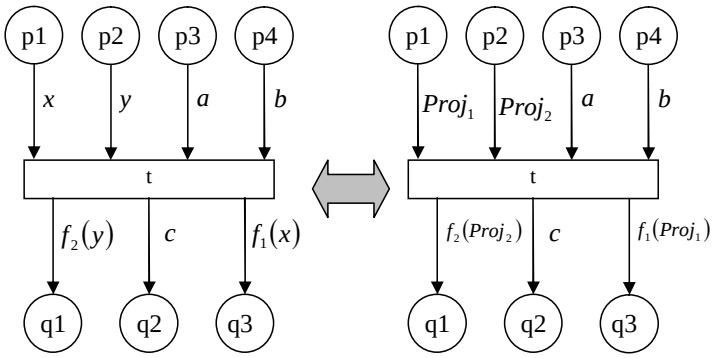


FIG. 5.4 – Passage d'un RdPC généralisé vers un RdPC bien formé

5.1.3 Preuves de l'inversion de RdPC

L'inversion appliquée sur les RdP peut être généralisée aux RdPC en deux étapes : 1) inversion de la direction des arcs et 2) transformation du RdPC pour l'évaluation [BBS09a]. Dans ce qui suit, la direction des arcs est d'abord inversée pour chaque transformation élémentaire. Ensuite, à partir du RdPC résultant, nous prouverons formellement le passage vers le RdPC inverse (structures montrées dans le chapitre 4).

Transformation basique

Fig.5.5 montre l'illustration en RdPC bien formé du cas basique. L'arc entrant est marqué par l'identité Id et l'arc sortant par une fonction f. Si la fonction f est orthonormale ou quasi-injective, les matrices inverses sont définies par :

$$\begin{cases} Pre'(p2,t) = Pre^{-1}(p1,t) \\ Post'(p1,t) = Post^{-1}(p2,t) \end{cases}$$

La preuve est une généralisation de celle appliquée dans le cas des RdP ordinaires. La relation avant (en un pas) entre les places est donnée par $Pre.Post^T$ et la relation arrière (en un pas) est la transposée de cette dernière matrice, à savoir $Post.Pre^T$. Vérifions ces relations dans le cas basique.

Nous avons :

$$Pre = \begin{pmatrix} Id \\ 0 \end{pmatrix}, Post = \begin{pmatrix} 0 \\ f \end{pmatrix}, Pre.Post^T = \begin{pmatrix} 0 & Id \diamond f \\ 0 & 0 \end{pmatrix}$$

Donc la relation entre $p1$ et $p2$ est vérifiée par le terme $Id \diamond f = f \circ Id^{-1} = f$. D'un autre côté, nous avons la relation arrière $Post.Pre^T = \begin{pmatrix} 0 & 0 \\ f \diamond Id & 0 \end{pmatrix}$ qui vérifie une relation de $p2$ vers $p1$ à travers le terme $f \diamond Id$. Comme $f \diamond Id = Id \circ f^{-1}$, l'algorithme fournit bien l'inverse du RdPC.

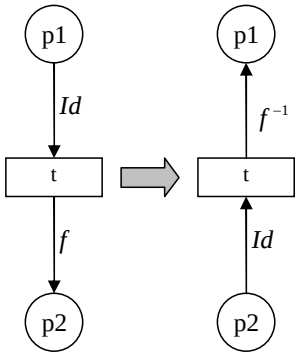


FIG. 5.5 – Transformation basique en RdPC bien formé

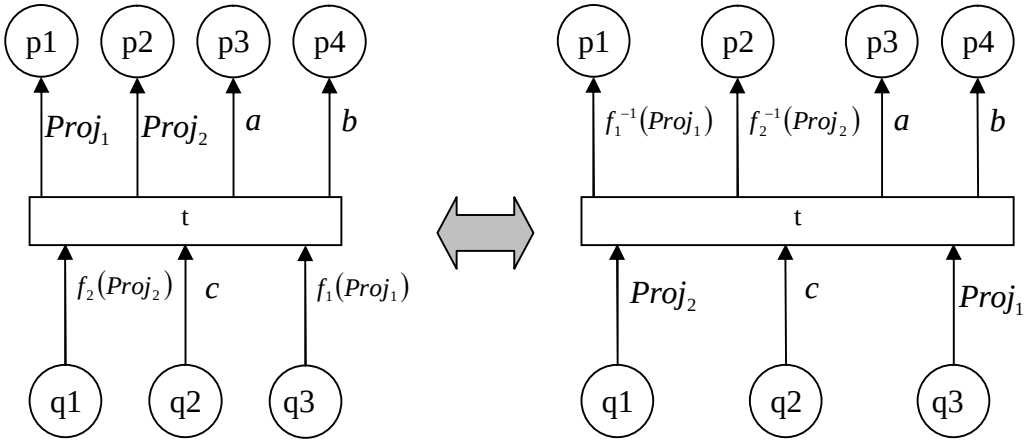


FIG. 5.6 – Transformation mixte en RdPC bien formé

Transformations mixtes

Fig.5.4 montre le passage de la notation RdPC généralisé (variables $\{x, y\}$ et constantes $\{a, b\}$ en entrée) vers la notation RdPC bien formé où les variables sont muées en *projections*. Le schéma Fig.5.6 illustre les deux étapes qui permettent d'inverser le RdPC. La première étape est équivalente à ce qui est appliqué pour la transformation triviale, à savoir, l'inversion de la direction des arcs. Il reste à prouver le passage vers la sémantique des RdPC généralisés.

Notons par C le domaine de couleurs de la transition tel que $C = C_1 \times C_2 \times \cdots \times C_n$. On définit $f_i : C_i \to C'$, et une projection $Proj_i : C \to C_i$. La composition des deux fonctions est définie par $f_i(Proj_i) : C \to C'$.

Notons par $\tilde{C}_i$ le domaine de couleurs défini par $\tilde{C}_i = C_1 \times \ldots C_{i-1} \times C' \times C_{i+1} \times \cdots \times C_n$. On définit $\tilde{f}_i : \tilde{C}_i \to C'$ avec $\tilde{f}_i(x) = (Proj_1(x), \ldots, Proj_{i-1}(x), f_i(Proj_i(x)), Proj_{i+1}(x) \ldots, Proj_n(x))$. On définit également une projection $Proj_i : \tilde{C}_i \to C$. La composition des deux fonc-

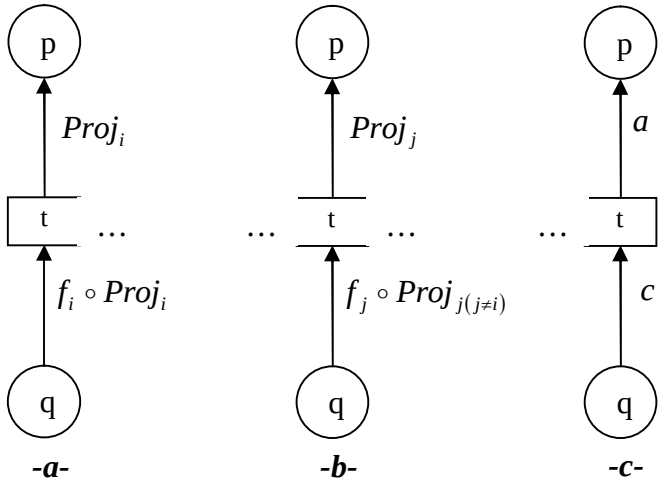


FIG. 5.7 – Preuve des transformations mixtes

tions est définie par $Proj_i(\tilde{f}_i) : C \to C'$.

Les quantités $f(Proj_i)$ et $Proj_i(\tilde{f}_i)$ sont égales. En effet :

$$\begin{aligned} Proj_i(\tilde{f}_i)(x) &= Proj_i(Proj_1(x), \ldots, Proj_{i-1}(x), f_i(Proj_i(x)), Proj_{i+1}(x) \ldots, Proj_n(x)) \\ &= f(Proj_i)(x) \end{aligned}$$

On définit enfin la fonction $\tilde{f_i^{-1}}$ telle que

$$\tilde{f}_i^{-1}(x) = (Proj_1(x), \ldots, Proj_{i-1}(x), f_i^{-1}(Proj_i(x)), Proj_{i+1}(x) \ldots, Proj_n(x))$$

La transformation s'effectue par une série d'orthonormalisations de la transition par les fonctions $\tilde{f}_i^{-1}$. Pour chaque fonction $\tilde{f}_i^{-1}$, trois cas sont identifiés comme illustré dans Fig.5.7.

Partie 1 : Fig.5.7.a montre la partie de la transition composée d'un arc entrant marqué par la fonction $f_i \circ Proj_i$ et d'un arc sortant marqué par $Proj_i$. La composition avec la fonction $\tilde{f}_i^{-1}$ donne les résultats qui suivent. Du côté de l'arc entrant, on aura

$$\begin{aligned} f_i \circ Proj_i \circ \tilde{f}_i^{-1} &= f_i \circ f_i^{-1} \circ Proj_1 \\ &= Proj_i \end{aligned}$$

Du côté de l'arc sortant, on aura $Proj_1 \circ \tilde{f}_i^{-1} = f_i \circ Proj_i$

Donc cette composition produit bien la forme recherchée d'une partie de la transition (fonctions indexées par i).

Partie 2 : Fig.5.7.b montre la partie de la transition composée d'un arc entrant marqué par la fonction $f_j \circ Proj_j$ (tel que $j \neq i$) et d'un arc sortant marqué par $Proj_j$. La composition avec la fonction $\tilde{f}_i^{-1}$ donne les résultats qui suivent. Du point de vue de l'arc entrant, on aura $f_j \circ Proj_j \circ \tilde{f}_i^{-1} = f_j \circ Proj_j$. Du coté de

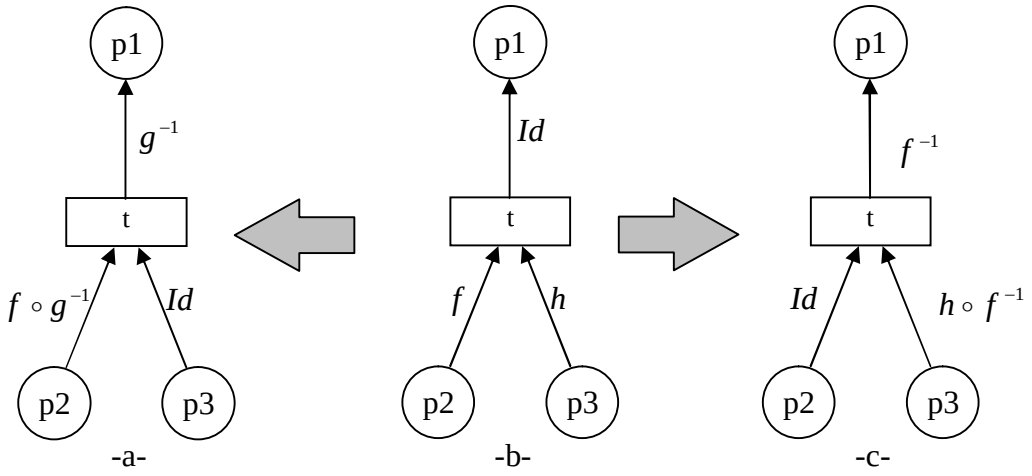


FIG. 5.8 – Preuve des transformations parallèles

l'arc sortant, on aura $Proj_j \circ \tilde{f}_i^{-1} = Proj_j$. Donc cette composition n'affecte pas les fonctions autres que celles indexée par i.

Partie 3 : Fig.5.7.c montre la partie restante de la transition, à savoir, les arcs marqués par les constantes. Sachant que la composition n'affecte pas les constantes, ceci reste vrai pour la composition avec $\tilde{f}_i^{-1}$.

Pour compléter la transformation de la transition, il suffit de refaire les étapes précédentes pour toutes les fonctions marquant les arcs entrants.

Transformations parallèles

La preuve sur les transformations parallèles s'effectue sur le RdPC résultant de l'inversion de la direction des arcs. Il reste donc à prouver le passage vers la sémantique du RdPC généralisé. Cette preuve peut être conduite pour deux fonctions *parallèles* ensuite généralisée à un nombre quelconque de fonctions parallèles.

La transformation, telle qu'illustrée dans Fig.5.8 peut s'effectuer par une orthonormalisation de la transition provoquant le parallélisme. Le choix de la fonction orthonormale est arbitraire (f ou g dans Fig.5.8). Ceci donne deux RdPC qui doivent être équivalents. Le schéma Fig.5.8.b est obtenu par composition avec g^{-1} et Fig.5.8.c est obtenu par composition avec f^{-1}.

Les deux RdPC étant identiques, les fonctions f^{-1} et g^{-1} ne s'appliquent qu'à un sous-ensemble de couleurs de la transition $c \in Bag(C(t))$ où $f^{-1}(c) = g^{-1}(c)$. Pour ce sous-ensemble de couleurs : $f \circ g^{-1} = Id$ et $g \circ f^{-1} = Id$. En faisant le remplacement dans les expressions des arcs entrants illustrés dans Fig.5.8.b et Fig.5.8.c, le résultat trouvé est identique. Le RdPC obtenu est assorti d'une

condition restrictive sur les domaines de couleurs. C'est cette condition qu'on appelle *garde* et qui s'exprime sur la notation liée à la transition.

5.2 Utilisation de la logique linéaire

La logique linéaire (LL pour *Linear Logic*) est le formalisme utilisé par [Kha03] pour donner une base théorique à l'analyse par accessibilité arrière utilisant le RdP inverse. Les avantages qu'offre ce formalisme sont, entre autres, une traduction facile entre RdP et LL, l'équivalence qui existe entre l'accessibilité dans les RdP et la preuve de séquents en LL et le fait de pouvoir inclure facilement tout le RdP dans une preuve (prendre en compte facilement les interactions) contrairement à l'algèbre linéaire où cet aspect est difficile à exprimer. Le défi que nous relevons dans cette partie du travail est de proposer une approche généralisant l'utilisation de la logique linéaire au cadre des RdPC dans un but d'application à l'accessibilité arrière [BRB09].

5.2.1 Présentation de la logique linéaire

La logique linéaire est introduite par Girard [Gir87]. La puissance de son expressivité est démontrée par l'encodage très naturel de certains modèles tels que les RdP ou les machines de Turing [Lin92]. La logique linéaire diffère de la logique classique par l'introduction de la notion de ressources. La logique classique traite des propositions statiques où chacune est soit $Vraie$ soit $Fausse$. À cause de la nature statique des propositions en logique classique, celles-ci peuvent être *dupliquées* $[P \Rightarrow (P \wedge P)]$ et/ou *ignorées* $[(P \wedge Q) \Rightarrow P]$. Ces deux propositions sont valides en logique classique pour chaque P et Q. En logique linéaire, ces propositions ne sont pas valides parce qu'aucune information n'est disponible sur la manière dont le second P est produit (dans la première proposition) ou sur la manière dont Q est consommé (dans la seconde proposition). Les règles de la logique linéaire imposent que les propositions linéaires représentent des propriétés dynamiques ou des ressources finies.

Considérons, par exemple, les propositions E, C, et T conçues comme des ressources : 1)E *: un Euro*, 2)C *: tasse de Café*, 3)T *: tasse de Thé*. Considérons aussi les axiomes suivants pour un distributeur de boissons : 1)$E \Rightarrow C$, 2)$E \Rightarrow T$. Dans ce cas, en logique classique, la proposition $E \Rightarrow (C \wedge T)$ peut être déduite. Elle peut être interprétée comme *"Avec un Euro, je peux acheter une tasse de Café et une tasse de Thé"*. Même si cette déduction est valide en logique classique, elle n'a pas de sens si les propositions sont interprétées comme des ressources : deux tasses de boissons chaudes ne peuvent pas être achetées avec un seul Euro dans le distributeur de boisson décrit.

Ce travail s'intéresse particulièrement à un fragment de la logique linéaire utilisé dans le domaine des RdP, à savoir, le fragment MILL (*Multiplicative Intuitionistic Linear Logic*). Ce fragment contient le connecteur multiplicatif *FOIS* ($\otimes$) et le connecteur de l'implication linéaire ($\multimap$). Le connecteur *FOIS* traduit l'accumulation de ressources. La proposition $A \otimes A$ signifie que deux ressources A sont disponibles. L'implication linéaire ($\multimap$) exprime la causalité entre la production et la consommation de ressources. La proposition $A \multimap B$ signifie que quand la proposition A est consommée, la proposition B est produite. Le meta-connecteur "**,**" (virgule) est cumulatif [Gir87].

5.2.2 Calcul de séquents

La notation de calcul de séquent, due à Gentzen [G.G69], est composée de deux séquences de formules séparées par un symbole tourniquet ($\vdash$). La formule $\Gamma, \Gamma' \vdash \Delta, \Delta'$ signifie que la conjonction de Γ et Γ' permet de déduire la disjonction Δ ou Δ'. Une règle de preuve dans un calcul de séquent est composée d'un ensemble de séquents hypothèses écrit au-dessus une ligne horizontale et une conclusion unique écrite au-dessous de cette même ligne, comme illustré ci-après :

$$\frac{Hypothèse_1 \qquad Hypothèse_2}{Conclusion}\, Règle_{attribut}$$

Le but est de construire un arbre de preuve. En débutant par un séquent, et en appliquant étape par étape une règle adaptée, la preuve consiste à éliminer les connecteurs. Ces règles sont illustrées dans Fig. 5.9 où A est un atome ; F, G et H sont des formules ; Γ et Δ sont des blocs séparés par des virgules. L'attribut indique si la règle est appliquée à gauche (L), à droite (R) ou à tout le séquent (attribut vide).

$$\frac{}{A \vdash A}\, id \qquad \frac{\Gamma \vdash F \qquad \Delta, F \vdash H}{\Gamma, \Delta \vdash H}\, Cut$$

$$\frac{\Gamma, F, G, \Delta \vdash H}{\Gamma, G, F, \Delta \vdash H}\, Exchange$$

$$\frac{\Gamma \vdash F \qquad \Delta, G \vdash H}{\Gamma, \Delta, F \multimap G \vdash H} \multimap_L \qquad \frac{\Gamma, \Delta, F \vdash G}{\Gamma, \Delta \vdash F \multimap G} \multimap_R$$

$$\frac{\Gamma, F, G \vdash H}{\Gamma, F \otimes G \vdash H} \otimes_L \qquad \frac{\Gamma \vdash F \qquad \Delta \vdash G}{\Gamma, \Delta \vdash F \otimes G} \otimes_R$$

FIG. 5.9 – Règle de calculs de séquent du fragment MILL

L'intérêt de la logique linéaire est qu'elle fournit, par exemple, un codage simple et naturel de l'accessibilité dans les RdP [Lin92]. Basée sur le calcul de séquents, la

logique linéaire aide à retrouver les conditions nécessaires et suffisantes pour l'existence de l'accessibilité d'un marquage vers un autre, grâce à l'équivalence entre l'accessibilité dans les RdP et la prouvabilité du séquent correspondant [F.G97]. De plus, la logique linéaire renseigne sur l'ordre partiel de tir des transitions pour atteindre un marquage final M_f à partir d'un marquage initial M_0 [HSER04].

Pour traduire un RdP en logique linéaire, une formule logique est associée à chaque transition. La partie gauche du séquent initial doit contenir la liste de toutes les transitions qui peuvent être tirées pour obtenir un marquage M_f à partir d'un marquage M_0. La construction de la preuve génère un arbre de preuve commençant par un séquent et finissant par l'axiome identité. Pour un RdP donné, la transformation est effectuée comme suit :

- ⋆ Une proposition atomique P est associée à chaque place p du RdP,
- ⋆ Un séquent utilisant la conjonction multiplicative $FOIS$ ($\otimes$) est associé à chaque marquage, pre-condition et post-condition de transition,
- ⋆ Pour chaque transition t, une formule implicative est définie comme suit :

$$t : \bigotimes_{i \in Pre(pi,t)} P_i \multimap \bigotimes_{o \in Post(po,t)} P_o$$

Pour montrer l'accessibilité entre deux marquages M_0 et M_f, la preuve du séquent $M_0, t_1, \ldots, t_n \vdash M_f$ est construite comme suit : d'abord, le marquage initial M_0 est remplacé par des atomes séparés en appliquant la règle $\otimes_L$ aussi souvent que nécessaire. Ensuite, en appliquant $\multimap_L$, les relations de causalité des atomes (de M_0 à M_f) sont extraits. À chaque fois que la règle $\multimap_L$ est appliquée, la règle $\otimes_L$ est appliquée s'il est nécessaire de séparer les atomes liés par $\otimes$. La preuve continue essentiellement du côté droit de l'arbre parce qu'après chaque application de la règle $\multimap_L$, le membre gauche est prouvé en utilisant, si nécessaire, la règle $\otimes_R$. La preuve du séquent se termine quand toutes les formules implicatives (exprimant les transitions) sont éliminées.

5.2.3 Logique linéaire et réseaux de Petri colorés

L'application de la logique linéaire à l'analyse par accessibilité des RdPC nécessite une traduction entre les deux formalismes. Cette traduction doit respecter les caractéristiques des RdPC, particulièrement, les types (valeurs) des jetons et les expressions des arcs qui n'existent pas en RdP ordinaires. Pour exprimer la différentiation des jetons et l'évolution de leurs valeurs en RdPC, la logique linéaire prédicative est très limitée ; c'est pourquoi dans le cadre des RdPC, la logique linéaire du premier ordre est préférée. Ce qui suit présente le fragment de la logique linéaire utilisé pour traduire les RdPC et présente ensuite l'algorithme de traduction. Le fragment en question est *First Order Multiplicative Intuitionistic Linear Logic* (noté MILL1).

First Order Multiplicative Intuitionistic Linear Logic

Le langage MILL1 est défini comme suit :

Alphabet : L'alphabet est constitué d'ensembles disjoints : un ensemble de symboles de variables (ex : x, y), un ensemble de symboles de fonctions (ex : f, g, h), un ensemble de symboles de relations (ex : R, S, T), les connecteurs binaires ($\otimes$, $\multimap$) et le quantificateur $\forall$. Chaque langage $\mathcal{L}$ est doté d'une application de $\mathcal{L}$ vers l'ensemble des entiers naturels $ar : \mathcal{L} \to \mathbb{N}$. Cette application ar exprime l'*arité*.

Termes : Étant donné un langage $\mathcal{L}$, les termes de premier ordre dans $\mathcal{L}$ sont définis par la catégorie syntaxique suivante :

$$\begin{array}{rcl} \tau & ::= & x \\ & | & f\left(\tau_1, \ldots, \tau_n\right) \end{array}$$

où x appartient aux variables et f aux symboles de fonctions de $\mathcal{L}$ tel que $ar(f) = n$.

Formules : Les formules du premier ordre MILL1 sont définies par la recursion syntaxique suivante :

$$\begin{array}{rcl} \varphi, \phi & ::= & R\left(\tau_1, \ldots, \tau_n\right) \\ & | & \varphi \otimes \phi \\ & | & \varphi \multimap \phi \\ & | & \forall x \cdot \varphi \end{array}$$

où R appartient aux symboles de relations de $\mathcal{L}$ tel que $ar(R) = n$ et où les occurrences de la variable x dans la formule φ sont liées dans la formule $\forall x \cdot \varphi$ par le quantificateur universel. Les variables qui ne sont pas liées par le quantificateur sont appelées libres.

Séquents : Si Γ est un multi ensemble de formules séparées par des virgules et φ est une formule alors $\Gamma \vdash \varphi$ est un séquent. En prenant Γ comme multi ensemble nous considérons implicitement que le meta-connecteur virgule (",") est associatif et cumulatif. Γ est appelé l'*antécédent* du séquent et φ le *succédant*.

Les preuves en MILL1 sont données par des termes d'arbres de preuve qui sont une composition de règles d'inférences sur des jugements. Les jugements sont un ensemble de paires de formules Γ et de formule φ écrite sous la forme $\Gamma \vdash \varphi$. Cette écriture signifie que la formule φ est une conséquence logique de celles de Γ. Les règles d'inférences (n-aire) sont des relations entres $n+1$ jugements qui sont notés comme suit :

$$\frac{\Gamma_1 \vdash \varphi_1 \qquad \cdots \qquad \Gamma_n \vdash \varphi_n}{\Gamma \vdash \varphi}$$

Ce qui signifie qu'il est suffisant d'établir la règle du jugement de dessous $\Gamma \vdash \varphi$ si les règles du dessus sont vérifiées ; en d'autres termes, pour établir la règle du jugement de dessous, il est nécessaire de prouver les jugements des règles de dessus $\Gamma_i \vdash \varphi_i$ $(1 \leqslant i \leqslant n)$. Les règles d'inference MILL1 sont donnés dans Fig. 5.10.

$$\frac{}{\varphi \vdash \varphi}\, id \qquad \frac{\Gamma, \varphi \vdash \phi}{\Gamma \vdash \varphi \multimap \phi} \multimap_r \qquad \frac{\Gamma \vdash \varphi \qquad \phi, \Delta \vdash \psi}{\Gamma, \varphi \multimap \phi, \Delta \vdash \psi} \multimap_l$$

$$\frac{\Gamma \vdash \varphi \qquad \Delta \vdash \phi}{\Gamma, \Delta \vdash \varphi \otimes \phi} \otimes_r \qquad \frac{\Gamma, \varphi, \phi \vdash \psi}{\Gamma, \varphi \otimes \phi \vdash \psi} \otimes_l$$

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \forall x \cdot \varphi} \forall_r \quad (*) \qquad \frac{\Gamma, \varphi \vdash \psi}{\Gamma, \forall x \cdot \varphi \vdash \psi} \forall_l$$

La contrainte $(*)$ requiert que la variable x ne soit pas libre dans les formules de Γ.

FIG. 5.10 – Règles de calcul de séquents du fragment MILL1

Traduction d'un RdPC vers MILL1

Pour un RdPC donné, la traduction vers MILL1 est conduite comme suit :

- ⋆ Un prédicat unaire atomique R_i est défini pour chaque place p_i du RdPC.
- ⋆ Un séquent utilisant la conjonction multiplicative $FOIS$ $(\otimes)$ est associé à chaque marquage, precondition et postcondition de transition,
- ⋆ Pour chaque transition t du RdPC, une formule implicative est définie comme suit :

$$t : \forall x_1 \ldots \forall x_i \left(\bigotimes_{i \in Pre(pi,t)} R_i(x_i) \multimap \bigotimes_{o \in Post(po,t)} R_o(f_o(X_o)) \right)$$

Où x_i sont des variables associées aux arcs entrants de t, f_o sont des fonctions associées aux arcs sortants de t et X_o sont des vecteurs composés de différentes associations de x_i.

L'exemple suivant traduit le RdPC illustré dans Fig. 5.11 en son équivalent MILL1

- ⋆ les places p_i sont traduites en prédicats unaires atomiques $x \mapsto R_i(x)$ où $(1 \leqslant i \leqslant 5)$;
- ⋆ la transition t_1 est traduite par la formule : $\varphi_1 \mathrel{\widehat{=}} \forall x \cdot \Big(R_1(x) \multimap R_2\big(f(x)\big) \otimes R_4\big(g(x)\big) \Big)$

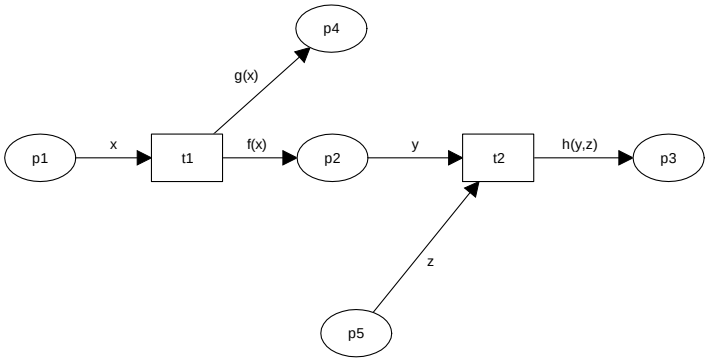


FIG. 5.11 – Exemple de RdPC à traduire en MILL1

- ⋆ la transition t_2 est traduite par la formule : $\varphi_2 \mathrel{\widehat{=}} \forall y \cdot \forall z \cdot \Big(R_2(y) \otimes R_5(z) \multimap R_3\big(h(y,z)\big) \Big)$
- ⋆ l'état initial est traduit par la formule : $\phi_0 \mathrel{\widehat{=}} R_1(i) \otimes R_5(j)$
- ⋆ l'état final est traduit par la formule : $\phi \mathrel{\widehat{=}} R_3(a) \otimes R_4(b)$

Le reste de l'exemple traite le cas d'accessibilité entre deux marquages M_0 et M_f tel que $M_0 = \langle i \rangle.p1 + \langle j \rangle.p5$ et $M_f = \langle a \rangle.p3 + \langle b \rangle.p4$. L'accessibilité entre M_0 et M_f est obtenue par le moyen du jugement $\phi_0, \varphi_1, \varphi_2 \vdash \phi$ suivant une preuve MILL1 (voir l'arbre de preuve Fig.5.12) : d'abord, la formule de l'état initial ϕ_0 est traitée par $\otimes_l$, ensuite la transition σ_1 est traitée par $\forall_l$ et $\multimap_l$. Le résultat de la première transition est traité par $\otimes_l$. La seconde transition σ_2 est traitée deux fois par $\forall_l$ et $\multimap_l$. Le résultat de la seconde transition est traité par $\otimes_r$. En parallèle, la formule ϕ représentant l'état final est traitée par $\otimes_r$. Finalement, on applique la règle *id* et on unifie les équations restantes en utilisant la substitution ς :

$$\begin{array}{rcl} \varsigma(a) & = & h\big(f(i), j\big) \\ \varsigma(b) & = & g(i) \end{array}$$

Cette démonstration, illustrée dans Fig.5.12, montre via une preuve de séquents l'accessibilité entre deux marquages génériques : le marquage initial $M_0 = \langle i \rangle.p1 + \langle j \rangle.p5$ et le marquage final $M_f = \langle a \rangle.p3 + \langle b \rangle.p4$. Elle fournit en plus les conditions nécessaires et suffisantes pour que M_f soit réellement accessibles par M_0. Ces conditions sont exprimées par la substitution ς. Une telle preuve peut être conduite sur un RdPC quelconque et notamment sur les RdPC inverse. Les résultats obtenus en faisant évoluer les jetons dans un RdPC inverses peuvent être vérifiés et prouvés grâce aux preuves de séquents.

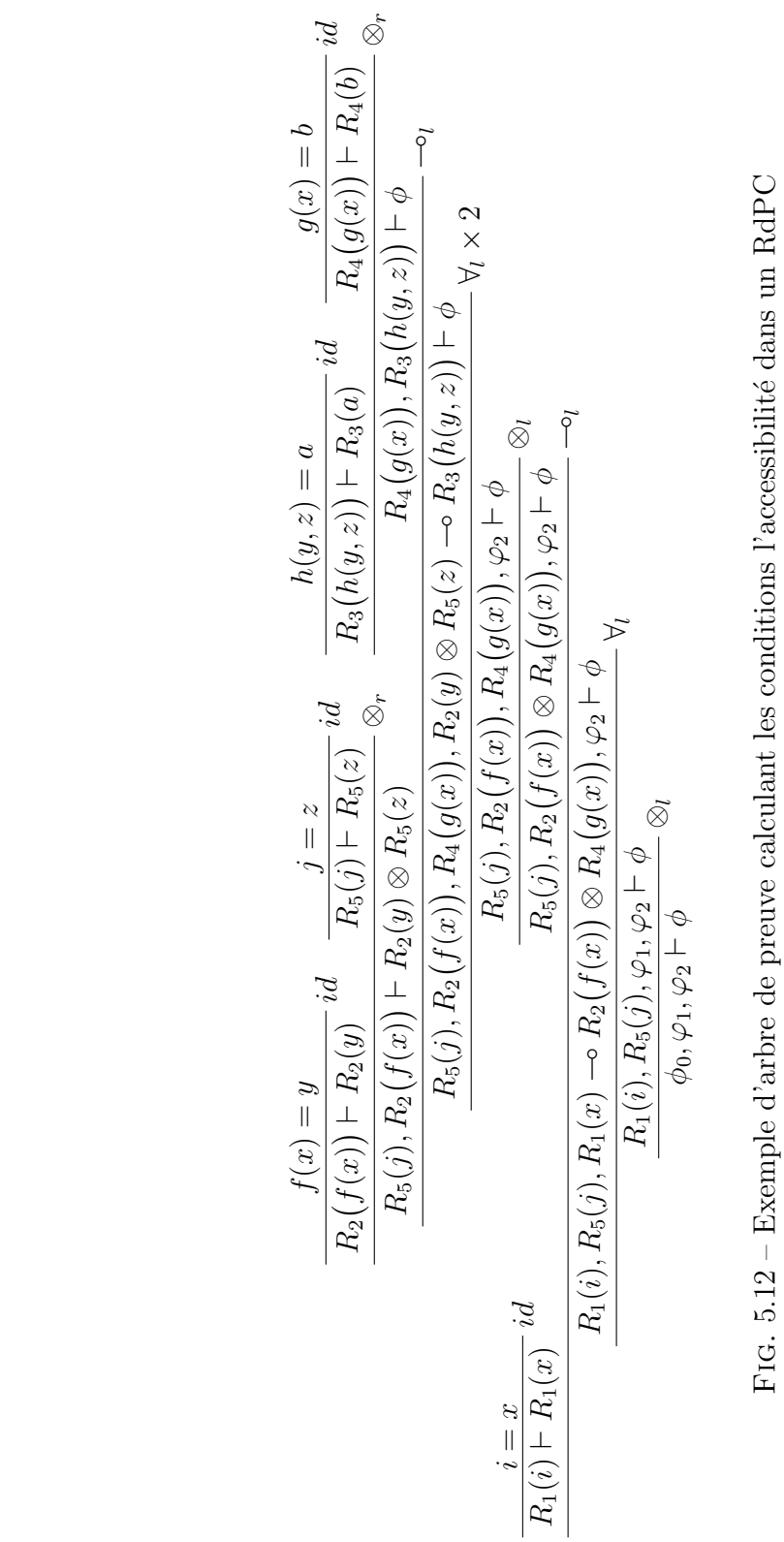


FIG. 5.12 – Exemple d'arbre de preuve calculant les conditions l'accessibilité dans un RdPC

5.3 Conclusion

Les aspects théoriques de l'analyse structurelle des RdPC sont étudiés avec deux outils mathématiques distincts : l'algèbre linéaire et la logique linéaire. Le premier outil (l'algèbre linéaire) permet de valider les transformations structurelles. Il ne se base que sur une théorie définie pour les RdPC et notamment les *Well formed Petri Nets*. La démarche principale, après inversion de la direction des arcs, s'appuie sur le passage vers les RdPC généralisés. Le second outil mathématique utilisé (la logique linéaire) permet de formaliser l'accessibilité dans les RdPC en prenant en compte la structure et la dynamique du modèle. Ceci nécessite une traduction du modèle RdPC en MILL1. Une fois cette traduction réalisée, toutes les techniques applicables sur les preuves de séquents peuvent être exploitées pour étudier le problème d'accessibilité et en particulier dans les RdPC inverse. Même si l'algèbre linéaire est plus proche des concepts structurels purs, la logique linéaire reste plus simple de prise en main et plus expressive. En effet, elle permet non seulement de travailler aisément sur des vue locales (transformation d'une transition) mais aussi sur des vues globales : transformations complexes et accessibilité dans un RdPC entier.

Chapitre 6

Application sur des modèles académiques

Sommaire

Après avoir présenté les aspects algorithmiques et théoriques de l'inversion de modèles RdPC, ce chapitre montre l'utilisation pratique de l'accessibilité arrière structurelle par le biais d'exemples introductifs. Le but est de montrer des situations variées pour lesquelles l'analyse par accessibilité arrière est un moyen de répondre à des questions de diagnostic. Les systèmes étudiés sont : 1) un protocole simple de communication et 2) un automate de surveillance d'une centrale nucléaire. Pour chacun des exemples, la spécification et la modélisation sont présentées en premier, suivies de l'analyse arrière contenant l'inversion des modèles RDPC.

6.1 Protocole de communication

6.1.1 Spécification et modélisation

Cet exemple est inspiré de la documentation de *CPN Tools*, et plus précisément de l'exemple titré *"Simple Protocol"* [41]. Ce protocole est connu dans la littérature,

[41] http ://wiki.daimi.au.dk

il est communément appelé *stop-and-wait*. Il décrit un protocole de communication utilisant un système de contrôle par acquittements. Dans la source documentaire dont il est inspiré, les analyses appliquées à cet exemple n'incluent pas de diagnostic ou d'analyse arrière. Notre objectif est d'appliquer l'approche proposée de l'analyse par accessibilité arrière à cet exemple. Cette analyse doit répondre à la question de savoir si le récepteur peut recevoir un ensemble hypothétique de trames.

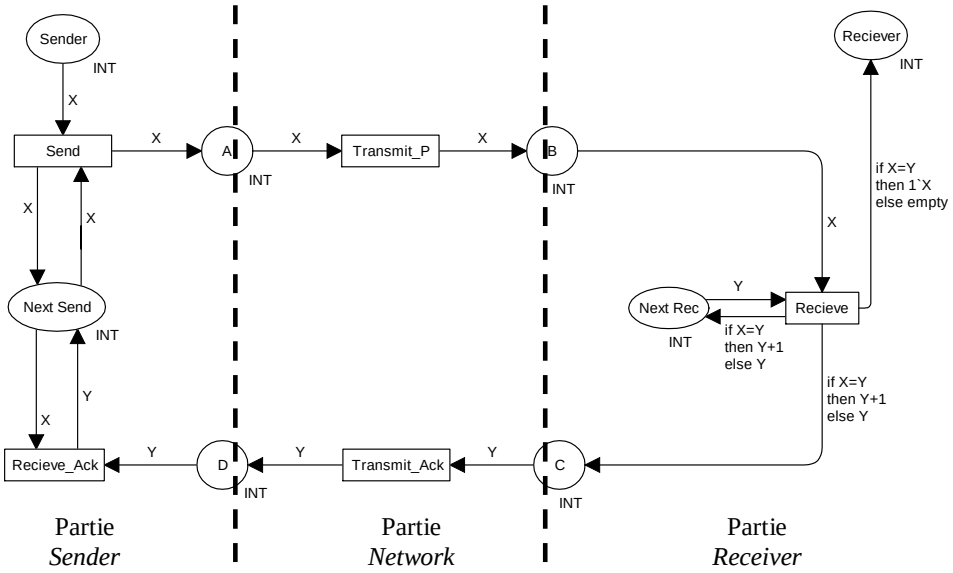


FIG. 6.1 – Modèle RdPC du protocole de communication

Le modèle RdPC de ce protocole est illustré dans Fig.6.1. Il est constitué de trois parties. La partie *Sender* possède deux transitions qui peuvent envoyer des paquets et recevoir des acquittements. La partie *Network* possède deux transitions qui peuvent transmettre des paquets et transmettre des acquittements. Enfin, la partie *Receiver*, modélisée par une transition unique, peut recevoir des paquets et envoyer des acquittements. L'interface entre les parties *Sender* et $Network$ est constituée des places A et D. L'interface entre les parties $Network$ et *Receiver* est constituée des places B et C. Les paquets à envoyer sont placés dans la place *Sender*. Chaque jeton dans cette place contient un paquet représenté par son numéro de séquence. La place $Next\_Send$ contient le numéro du prochain paquet à envoyer et est mis à jour à chaque réception correcte d'un acquittement. Tous les messages correctement reçus, sont stockés dans un buffer représenté par la place *Receiver*. La place $Next\_Rec$ contient le numéro du prochain paquet à recevoir et est mis à jour à chaque reception valide d'un paquet. Dans cette étude, le contenu de la trame n'influe pas sur l'analyse c'est pourquoi il n'est pas pris en compte.

6.1.2 Analyse par accessibilité arrière

Le fonctionnement nominal de ce système ne devrait pas laisser apparaître les doublons (i.e. réception correcte de deux trames identiques). Ainsi par exemple, le marquage avec 3 paquets dans la place *Receiver* où le premier est identifié par 1 et les deux autres égaux identifiés par 2 représente un état inconsistant. Ce marquage peut être noté $\langle 1+2+2\rangle.Receiver$.

Pour vérifier l'accessibilité du marquage donné précédemment ($\langle 1+2+2\rangle.Receiver$), nous procédons à l'analyse par accessibilité arrière en utilisant le RdPC inverse illustré dans Fig.6.2. La transformation des transitions $Transmit\_P$ et $Transmit\_Ack$ est de type basique avec à chacune d'elle la fonction identité comme fonction inversible. Autour de *Receive_Ack*, l'inversion est un cas de transformation paramétrée (la variable entrante X n'est pas utilisée en sortie). Le but de cette transition est d'écraser l'ancienne valeur de $Next\_Send$ et de la remplacer par la valeur du jeton dans la place D. Donc d'un point de vue inverse, le but est de mettre dans D la valeur du jeton de $Next\_Send$ et de mettre dans cette place une valeur arbitraire. Nous avons choisi de remettre la valeur déjà contenue dans cette place. L'inversion autour de la transition *Send* est de type parallèle. En effet, la variable X en entrée est utilisée dans deux fonctions en sortie (vers A et $Next\_Send$) ce qui explique l'apparition de la variable Z et la garde $[X = Z]$. L'inversion de *Receive* est à la fois parallèle et paramétrée (les expressions d'arcs sont des fonctions à deux variables : X et Y). Comme dans le problème inverse, les jetons dans *Receive* sont connus, ceci revient à considérer que les valeurs de X sont connues. La garde est composée de deux partie $Y = Z$ et $X = Y - 1$. La première partie de la garde ($Y = Z$) vient du fait que, dans le RdPC original, les fonctions associées aux arcs sortants vers $Next\_Rec$ et C, sont identiques. Comme la garde dans la transformation parallèle s'écrit sous la forme $Y = h(f^{-1}(Z))$ et que dans, ce cas précis, les fonctions h et f sont identiques alors la garde donne $Y = f(f^{-1}(Z))$ ce qui est égal à $Y = Z$. La seconde partie de la garde ($X = Y-1$) est le résultat de la transformation parallèle en prenant en compte les fonctions associées aux arcs sortants vers $Next\_Rec$ et $Reveiver$. Elles stipule qu'à chaque tir de *Receive* produisant un jeton dans *Receiver*, la valeur de Y (qui est égale à la valeur de X) est incrémentée d'une unité.

Dans le RdPC original, la séquence de tir des transitions est : *Send*, $Transmit\_P$, *Receive*, $Transmit\_Ack$, *Receive_Ack*. Ceci correspond à un ordre partiel de tir des transitions [Kha03] [AP94]. L'analyse du RdPC inverse (doté du marquage $\langle 1+2+2\rangle.Receiver$) doit respecter cet ordre (inversé) commençant par la transition *Receive* (seule transition potentiellement franchissable). Le tir de *Receive* s'accompagne d'un enrichissement de marquage et donc de l'introduction d'hypothèse sur l'état du système. Deux jetons sont à rajouter au modèle : le premier dans la place $Next\_Rec$ et le second dans la place C. Les deux sont de valeur

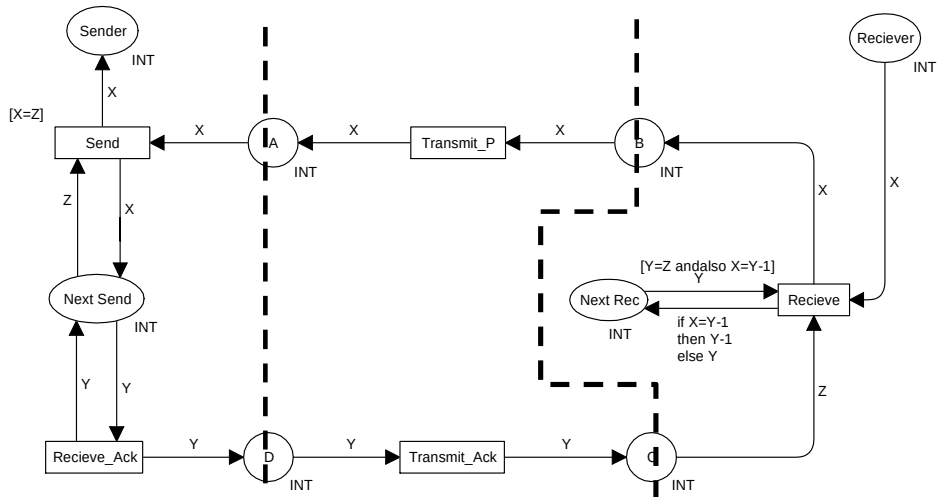


FIG. 6.2 – RdPC inverse du protocole de communication

3 car cette valeur satisfait la garde $[X = Z \wedge X = Y - 1]$. Cet enrichissement reconstitue l'état du système à la réception du paquet estampillé du numéro 2 . Il correspond à faire l'hypothèse que le paquet en question a été bien reçu. Le résultat est un jeton de valeur 2 dans la place $Next\_Rec$ et B puis dans A (après tir de $Transmit\_P$). Le tir de $Send$ suppose que $Next\_Send$ soit marqué d'un jeton de valeur 2, ce qui suppose l'envoi normal du paquet. L'évolution du marquage (sans aucun enrichissement) fait que le paquet estampillé 1 va être traité grâce au tir de toutes les transitions une fois chacune. Ce processus aboutit à une situation dans laquelle un jeton estampillé 2, dans la place *Receive*, ne peut pas évoluer sans un nouvel enrichissement de marquage. Un problème donc se pose dans le fait que la place $Next\_Rec$ contient déjà une autre numéro de paquet reçu.

Cette analyse commence d'un état final (supposé défaillant) du système. Cet état n'est connu que partiellement. L'analyse par accessibilité arrière tente de reconstituer l'ensemble des événements (sous formes de scénarios) faisant aboutir le système vers l'état final. La reconstitution des scénarios nécessite l'augmentation des connaissances sur les états intermédiaires apportées par l'enrichissement du marquage. L'analyse précédente conclut sur le fait qu'aucun marquage initial cohérent n'est possible pour aboutir au marquage $\langle 1 + 2 + 2 \rangle.Receiver$. La défaillance pourrait venir d'un événement extérieur au fonctionnement propre du système telle qu'une erreur d'initialisation qui mettrait le jeton de valeur 2 dans le buffer du *Receiver* ou une faute transitoire affectant le système d'acquittement. À la detection d'un marquage correspondant à l'état final supposé, une faute imprévue doit être déclarée.

Pour généraliser le résultat de cette analyse, il suffit de marquer la place *Receiver* par deux jetons de même valeur générique i. Le marquage final M_f s'écrirait donc $M_f = \langle i + i \rangle.Receiver$. L'analyse montre alors que ce genre de

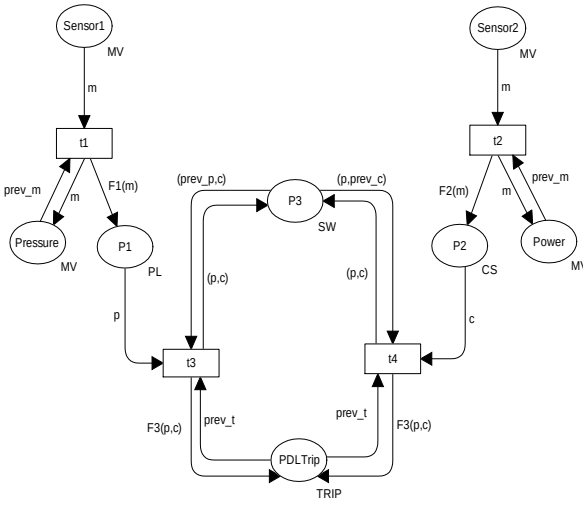


FIG. 6.3 – RdPC du paramètre PDL Trip

marquage n'est pas possible dans le fonctionnement nominal du système. Il survient si une faute extérieure apparaît.

6.2 Centrale Nucléaire

6.2.1 Spécification et modélisation

Cet exemple est inspiré de [CHC96]. Il décrit un système d'arrêt (*shutdown*) d'une centrale électrique nucléaire. Le système surveille des variables d'état du réacteur telles que la pression et la puissance et génère un signal d'erreur (*trip*) si le réacteur nucléaire entre dans un état jugé dangereux (i.e. pression trop haute ou trop basse). Le système réel comporte une multitude de paramètres. Cet exemple en décrit seulement le paramètre *PDL trip* dont le modèle est illustré dans Fig.6.3

Pour contrôler l'état du paramètre *PDL trip*, le système surveille l'état du réacteur à travers des variables comme la différentielle de la pression du noyau (noté ΔP) et le log du signal de puissance (noté ϕ_{LOG}). Dans Fig.6.3, $Sensor1$ et $Sensor2$ représentent les capteurs fournissant les valeurs d'entrée du système. Les places $Pressure$ et $Power$ représentent la valeur courante de ΔP et ϕ_{LOG} respectivement. $P1$, $P2$ et $P3$ représentent les états du système. $PDLTrip$ représente le paramètre *PDL trip* dont la valeur est calculée comme suit :

- ⋆ determiner le conditionnement de l'erreur : si $\phi_{LOG} < 2739mV$ désactiver l'erreur. Sinon activer l'erreur.
- ⋆ Determiner le paramètre *PDL trip* : si le conditionnement de l'erreur est activé et ($\Delta P < 1287mV$ or $\Delta P > 4810mV$) ouvrir le paramètre *PDL trip*. Sinon fermer le paramètre *PDL trip*.

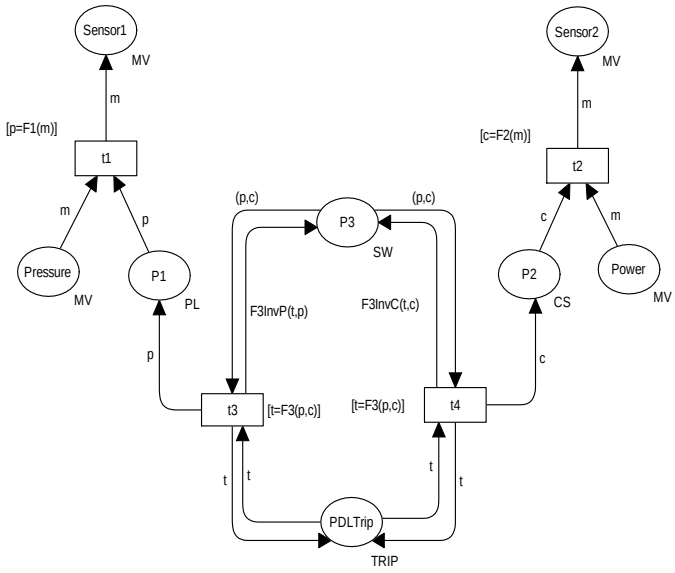


FIG. 6.4 – RdPC inverse du paramètre PDL Trip

Posons le vecteur d'état $(Pressure, Power, P1, P2, P3, PDLtrip)$. L'état $(x, y, ?, ?, ?, z)$[42] est considéré défaillant si $x \leq 1287$ ou $x \geq 4810$, $y \geq 2739$, et $z = close$.

6.2.2 Analyse par accessibilité arrière

L'inversion du modèle RdPC (Fig.6.4) peut être réalisée par l'application d'une transformation parallèle aux transitions $t1$ et $t2$ et d'une transformation parallèle paramétrée aux transitions $t3$ et $t4$.

Transitions $t1$ et $t2$

Le tir de ces transitions correspond à une mise à jour des valeurs de $Pressure$ et $Power$. Les anciennes valeurs sont écrasées et remplacées par celle de la variable m. De nouvelles valeurs de $P1$ et $P2$ sont calculées à travers les fonctions $F1$ et $F2$ respectivement. L'inversion de ces transitions exploite les valeurs produites par $F1$ et $F2$ pour retrouver celles des jetons dans les places $Pressure$ et $Power$ à travers les gardes associées aux transitions $t1$ et $t2$.

Transitions $t3$ et $t4$

Ces transitions sont à la fois parallèles et paramétrées. Parallèles parce que les expressions des arcs sortants sont fonction des mêmes variables (entrantes). Paramétrées pour cause de présence de la fonction $F3$ à plusieurs variables. L'inversion

[42] le caractère "?" désigne une valeur quelconque

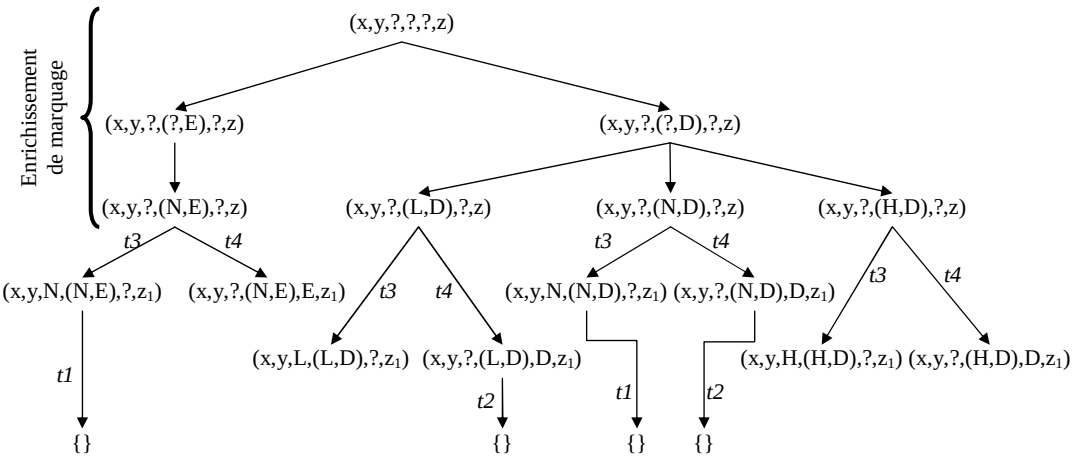


FIG. 6.5 – Arbre d'accessibilité arrière du paramètre PDL trip

fait apparaître la garde $[t = F3(p, c)]$ associée à $t3$ et $t4$. L'inversion de $F3$ est différente selon qu'on inverse $t3$ ou $t4$. Autour de $t3$, la variable c reste constante, par contre, autour de $t4$ c'est la variable p qui reste constante.

Analyse arrière

Considérons dans le RdPC inverse le même vecteur d'état que le RdP original, à savoir : $(Pressure, Power, P1, P2, P3, PDLtrip)$. On pose le marquage défaillant dans le RdPC inverse $(x, y, ?, ?, ?, z)$ tel que $x \leq 1287$ ou $x \geq 4810$, $y \geq 2739$ et $z = close$.

L'enrichissement du marquage consiste à définir les valeurs possibles des jetons inconnus (parmi les places $P1$, $P2$, $P3$) commençant par la composante c du couple (p, c) (jetons dans la place $p3$). L'ensemble des couleurs possibles dans ce cas (pour la variable c) est limité par définition à deux valeurs $Enable$ et $Disable$ (notées E et D respectivement). Les valeurs correspondantes pour la variable p sont calculées à travers l'inverse paramétrée de la fonction $F3$ prenant comme entrée les valeurs de z et c (ce qui revient dans ce cas à résoudre la garde associée aux transitions $t3$ et $t4$). L'analyse se poursuit par le tir des transitions valides. Une partie des résultats est illustrée dans l'arbre d'états Fig.6.5. Au niveau des feuilles de l'arbre, les marquages qui ne peuvent pas produire l'état final défaillant sont noté par l'ensemble vide ($\{\}$). Ils sont discriminés grâce aux grades associées à chacune des transitions $t1$ et $t2$. Les autres marquages mènent donc au marquage défaillant. Le résultat est obtenu malgré la connaissance partielle (matérialisée par le symbole ? dans les vecteurs d'états) des états menant à la défaillance. Notons l'apparition d'une nouvelle valeur notée z_1. Elle correspond au prédicat $z = close \vee z = open$.

Une autre manière d'enrichir le marquage consiste à définir les valeurs possibles des jetons inconnus (parmi $P1$, $P2$, $P3$) commençant par la composante p du

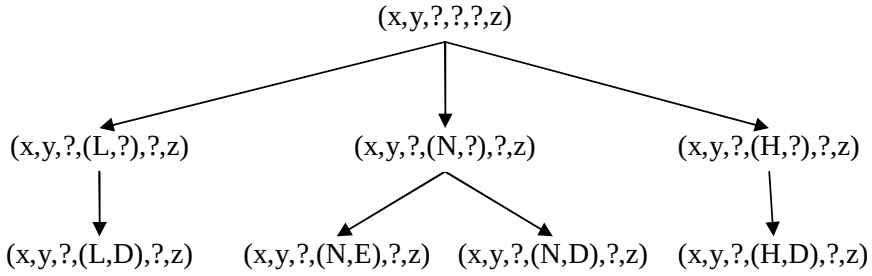


FIG. 6.6 – Seconde méthode d'enrichissement de marquage (PDL trip)

couple (p, c) (jetons dans la place $p3$). L'ensemble des couleurs possibles dans ce cas (pour la variable c) est limité par définition à trois valeurs Low, $Normal$ et $High$ (notées L, N et H respectivement). Les valeurs correspondantes pour la variable c sont calculées à travers l'inverse paramétré de la fonction $F3$ prenant comme entrée les valeurs de z et c. Cette procédure est schématisée par Fig.6.6.

L'étude par accessibilité arrière permet de discriminer rapidement un sous-ensemble de marquages menant au marquage défaillant. Elle permet de reconstituer aussi l'ordre des événements conduisant à ce marquage. Les résultats montrés ne sont pas exhaustifs mais illustrent bien le potentiel de l'accessibilité arrière. La poursuite de l'étude permettrait de remonter de plus en plus loin dans l'historique des événements menant à la défaillance.

6.3 Conclusion

Ce chapitre fournit des exemples introductifs pour une première prise en main de la méthode d'analyse structurelle par accessibilité arrière. La conduite d'une telle analyse nécessite : 1) une phase d'inversion du RdPC, 2) une phase d'enrichissement du marquage, correspondant à l'introduction d'hypothèses sur le fonctionnement passé du système et 3) une phase dans laquelle on fait évoluer dynamiquement le RdPC inverse.

La première phase (inversion du RdPC) est une application des différentes transformations structurelles proposées dans le chapitre 4. Concernant la deuxième phase et la troisième phase, un point important à considérer est la prise en compte maximale de l'information disponible. En effet, la précision et la justesse de l'analyse effectuée repose, en partie, sur les hypothèses complémentaires qu'il faut parfois rajouter. La minimisation du nombre (et donc de l'impact) de ces hypothèses passe par l'exploitation au mieux de la structure du modèle (expressions des arcs, gardes, etc.) et de la distribution des jetons connue au préalable. L'enrichissement du marquage doit donc être effectué

La suite des exemples introductifs est une application de l'approche proposée

pour un cas tiré de l'industrie. L'exemple en question est un système de freinage de tramway. L'analyse de ce système passe par une modélisation en RdPC et une analyse par accessibilité arrière décrite de telle manière à faire ressortir les apports concrets de cette analyse.

Chapitre 7

Application sur un modèle industriel : système de freinage de tramway

Sommaire

7.1 Introduction

Ce chapitre met en application la démarche proposée sur un cas tiré du domaine du transport ferroviaire, à savoir, le système de freinage d'un tramway. L'objectif est de montrer, sur des études de cas, les différentes manières d'exploiter l'analyse par accessibilité arrière et de voir l'étendue de son utilisation.

L'organisation de ce chapitre commence par la spécification du système à étudier. Elle est suivie par une modélisation détaillée en RdPC avec complément d'information, notamment sur les fonctions calculant les valeurs de freinage ainsi que l'impact de chaque type de freinage sur la vitesse. La section suivante explique,

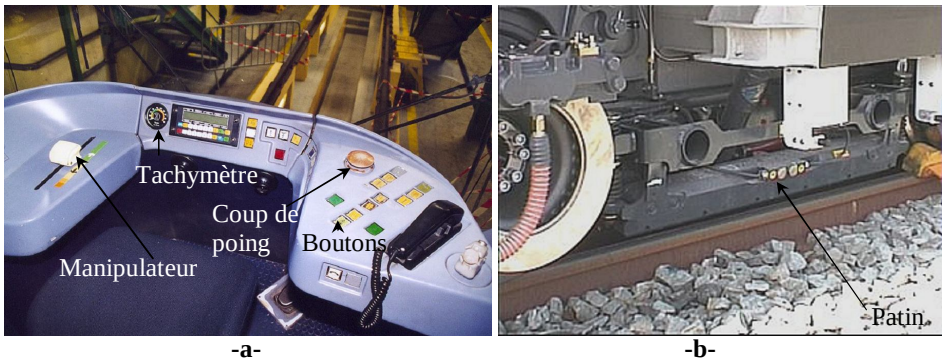


FIG. 7.1 – Vue de cabine de conduite et de patin d'un tramway

transition par transition, l'inversion du modèle. Viendront ensuite, les différentes analyses ainsi que leurs résultats. Ces analyses sont organisées de la plus simple à la plus complète.

7.2 Spécification

Le conducteur possède à sa disposition, au pupitre de conduite (Fig.7.1.a), un manipulateur traction/freinage qui donne les consignes d'effort (accélération, neutre, freinage) ainsi qu'un ensemble de boutons poussoirs : soit marche arrière soit marche avant. Le sens de marche est sélectionné à partir de deux boutons poussoirs arrière et avant. Par défaut, la marche avant est sélectionnée automatiquement à la mise en service de la cabine (obtenue par une clé de prise de service).

Les consignes d'accélération ou de freinage sont transmises aux différentes électroniques de traction freinage (ETF) qui gèrent les différents bogies[43] du train. Le mode de freinage usuel, dit *freinage de service* est obtenu en tirant le manipulateur au delà du point neutre sans toutefois le tirer à fond, position provoquant le *freinage d'urgence* (voir ci-après). Dans le cas du freinage de service, les ETF réalisent l'effort demandé, variable selon la position du manipulateur, par le frein électrodynamique (le moteur travaille en générateur et le courant généré est dissipé dans des résistances). Ce freinage utilise aussi le freinage mécanique à disques en complément au besoin. Pour éviter de provoquer des chutes trop fréquentes dans le train, l'effort de freinage est modulé selon la charge de passagers (mesurée par un dispositif de pesée). Le dispositif appelé anti-enrayage (antiblocage de roue, équivalent de l'ABS automobile) est actif dans ce mode.

Le *freinage d'urgence* est obtenu en tirant à fond vers soi le manipulateur. Il est auto maintenu jusqu'à l'arrêt complet du train. En freinage d'urgence, les performances de décélération maximales sont recherchées. Les ETF commandent donc

[43] ensemble de deux essieux

l'effort maximal de frein électrodynamique et mécanique à disques, la modulation à la charge de passagers et l'anti-enrayage restent actifs. Le sablage (injection de sable avec des buses d'air comprimé, sous les roues) est commandé sur tout le train. En complément des patins électromagnétiques (longs électroaimants visibles sur Fig.7.1.b) sont alimentés et viennent s'appliquer sur le rail. Une fois le véhicule arrêté les patins sont désalimentés et sont rappelés à leur position de repos par des ressorts.

Le *freinage de secours* est obtenu par action sur le coup de poing (Fig.7.1.a). Dans ce cas c'est la sécurité vis à vis de toute défaillance technique qui est recherchée. Tous les moyens d'asservissement complexes sont donc inhibés (anti-enrayage, modulation à la charge), le disjoncteur principal s'ouvre (supprimant toute alimentation en courant de traction, et donc toute possibilité d'accélération intempestive), ce qui a également pour effet de supprimer le frein électrodynamique. Le freinage de secours n'utilise que le frein à mécanique à disques commandé à la valeur maximale, et les patins électromagnétiques. Le sablage est également commandé sur tout le train. Lorsque l'arrêt de véhicule est détecté, le *frein d'immobilisation* est automatiquement appliqué, sollicitant les seuls freins à disques à une valeur d'effort prédéterminée, fonction de la charge des passagers. Le frein d'immobilisation est levé dès qu'une consigne d'accélération est émise depuis le manipulateur.

7.3 Modélisation en réseau de Petri coloré

Le modèle RdPC reprend les différents types de freinage cités dans la spécification : freinage de service, freinage d'urgence, freinage de secours et freinage d'immobilisation (section 7.2). Chaque type de freinage est représenté par une transition qui met en relation les causes (pré conditions) et les organes sollicités (post conditions). Le modèle est illustré dans Fig.7.2. Notons que la spécification ne fournit pas les valeurs ou les fonctions numériques concernant les consignes à appliquer. Les choix de celles-ci affectent très peu l'analyse, principalement la forme des graphique représentant les résultats. Ces choix sont donc faits de manière à mettre l'accent sur le déroulement et les apports de l'analyse structurelle par accessibilité arrière.

La transition *Service* modélise le freinage de service qui est enclenché par une position du manipulateur (représentée par la place *manipulateur*) au-delà du point neutre mais pas en position maximale. Cette condition est représentée par la garde associée à cette transition. Elle permet un tir avec des valeurs de la variable $m \in]0..1[$ tel que 0 est le point neutre et 1 est la position tirée à fond. La consigne de l'effort est proportionnelle à la position du manipulateur. Elle est modulée aussi par la pesée et l'anti-enrayage (représenté par des places

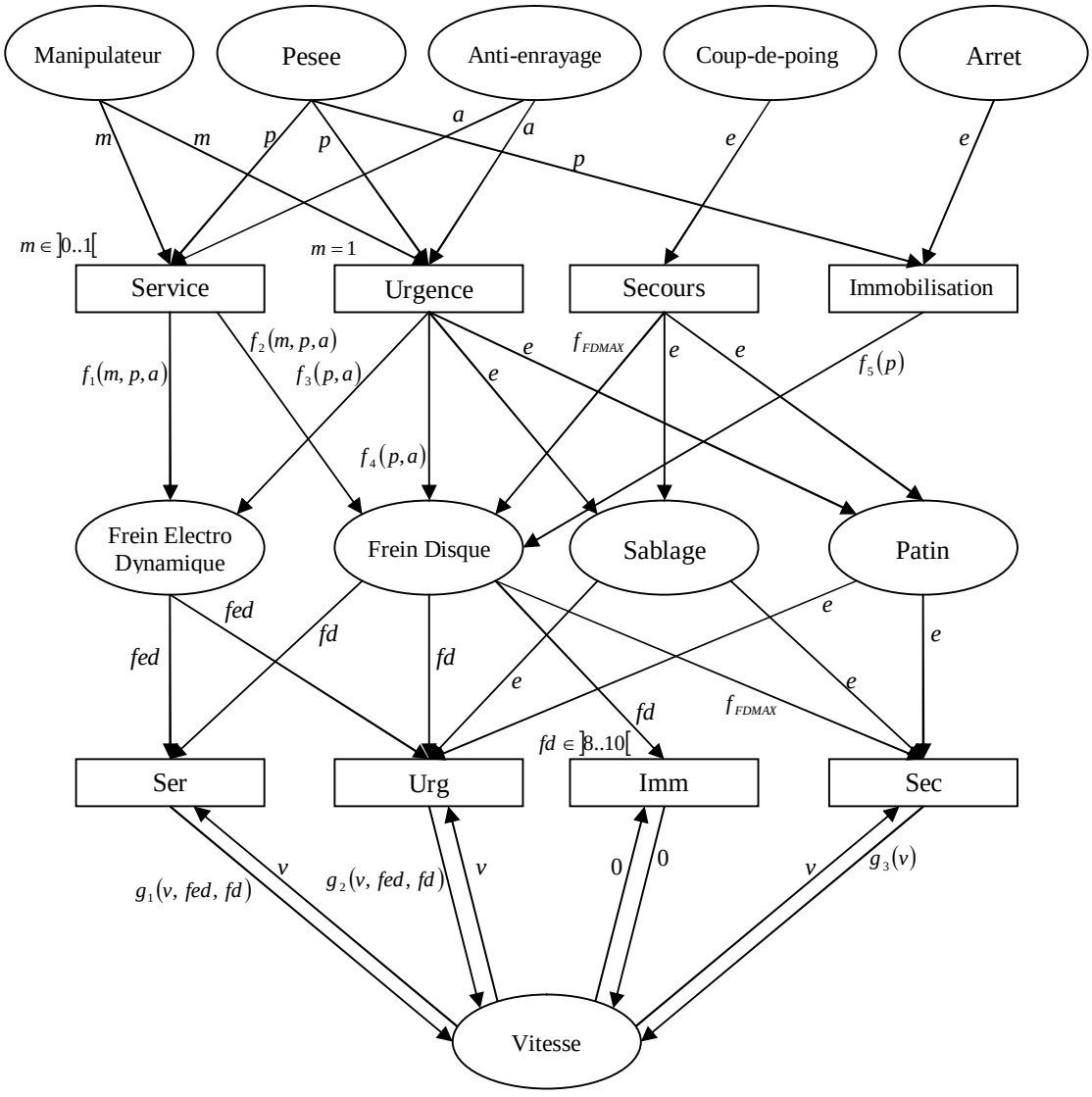


FIG. 7.2 – Modèle RdPC du freinage de tramway

de même nom). Les valeurs des variables p et a (pour pesée et anti-enrayage respectivement) appartiennent toutes les deux à l'intervalle [0..1]. La consigne du freinage en sortie est envoyée vers le frein électrodynamique. Elle est calculée par la fonction $f_1(m, p, a) = 5m + 2p + a$. Quand la valeur cumulée de p et a est supérieure au seuil de $1,5$, une consigne est envoyée aux freins à disques calculée $f_2(m, p, a) = 7m + 2p + a$.

La transition *Urgence* modélise le freinage d'urgence provoqué par une position à fond du manipulateur. Cette condition est exprimée par la garde $m = 1$ associée à la transition. Ce freinage provoque une action sur tous les dispositifs matériels de freinage. Il envoie une consigne modulée par la pesée et l'anti-enrayage au frein électrodynamique et aux freins disques calculées par $f_3(p, a) = 5 + 2p + a$ et $f_4(p, a) = 7 + 2p + a$. Il déclenche aussi le sablage et les patins par l'envoi d'un signal de déclenchement matérialisé par un jeton anonyme e.

La transition *Secours* modélise le freinage de secours provoqué par une action sur le coup-de-poing. Ce type de freinage provoque l'envoi d'une consigne maximale $f_{FDMAX} = 10$ aux freins à disques et le déclenchement du sablage et des patins.

La transition *Immobilisation* modélise le freinage d'immobilisation provoqué par la détection d'un arrêt du tramway. Il enclenche le frein à disques grâce à une consigne modulée par le dispositif de pesée et calculée par la fonction $f_5(p) = 8 + 2p$.

Les transitions *Ser*, *Urg*, *Imm* et *Sec* calculent la nouvelle vitesse en fonction du freinage appliqué (la valeur de la consigne, le cas échéant) et de la vitesse précédente. Les fonctions de réduction de vitesse, suivant le freinage sont formulées comme suit :

- ⋆ transition *Ser* : $g_1(v, fed, fd) = v - \frac{3.v.fed}{40} - \frac{v.fd}{25}$.
- ⋆ transition *Urg* : $g_2(v, fed, fd) = v - \frac{v.fed}{32} - \frac{v.fd}{40} - \frac{v}{5} - \frac{3v}{10}$. La partie $\frac{v}{5}$ représente la contribution du sablage dans le freinage et la partie $\frac{3v}{10}$ celle des patins.
- ⋆ la transition *Imm* ne peut être tirée que lors de la détection de l'arrêt du véhicule. Elle prend donc en entrée une valeur de vitesse nulle et elle la maintient en sortie.
- ⋆ transition *Sec* : $g_3(v) \in [0..\frac{v}{3}]$

7.4 Inversion du modèle

La transition *Service* est à la fois parallèle et paramétrée. Elle est parallèle car les mêmes variables en entrée sont utilisées dans deux fonctions en sortie. La propriété paramétrée vient du fait que les fonctions en sortie sont des multi-variables. Plusieurs transformations sont possibles, chacune dépend de l'entrée à calculer, mais toutes doivent respecter les deux propriétés (parallélisme et paramétrage).

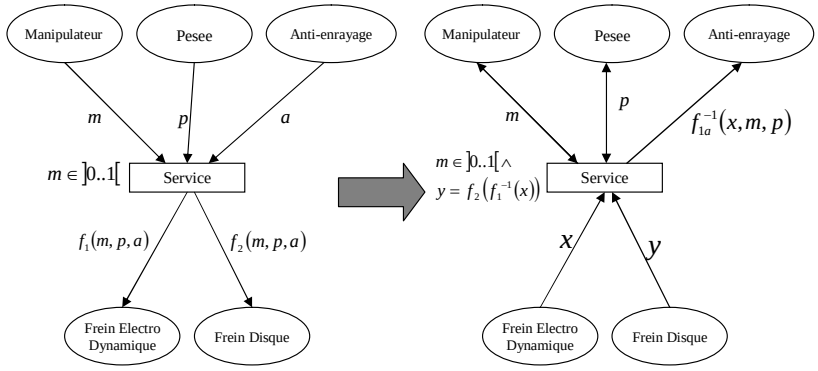


FIG. 7.3 – Inversion de la transition Service

Une des transformations possibles est illustrée dans Fig.7.3. Nous supposons les valeurs m et p. Le calcul se fait alors sur la variable a avec la fonction inverse partielle $f_{1a}^{-1}(x) = x - 5m - 2p$. Lors des calculs, les résultats de cette fonction qui n'appartiennent pas au domaine de couleurs de la place *AntiEnrayage* (et donc de la variable a) sont éliminés ainsi que les combinaisons d'entrée qui les ont engendrés. Une garde supplémentaire est également associée à la transition respectant ainsi la transformation parallèle. Cette garde s'écrit $y = f_2(f_1^{-1}(x))$ tels que

$$f_1^{-1}(x) = (m, p, a) = (\frac{x - 2p - a}{5}, \frac{x - 5m - a}{2}, x - 5m - 2p)$$

$$f_2(f_1^{-1}(x)) = \begin{array}{lll} & Si & (((3x - 15m - 4p - a)/2) < 1.5)\ Alors \\ & & 0 \\ & Sinon & \\ & & (17x - 50m - 24p - 12a)/5 \end{array}$$

La transition *Urgence*, pour les mêmes raisons que la transition *Service*, est parallèle et paramétrée. Elle est de plus mixte car elle possède des expressions d'arc constantes en sortie produisant des jetons anonymes e. Une des transformations possibles de cette transition est illustrée dans Fig.7.4. Le tir de la transition, dans le RdPC original, se fait sur une valeur de $m = 1$. Donc, dans le modèle inverse, l'arc ($Urgence$, $Manipulateur$) est associé à la constante 1. Les valeurs de l'anti-enrayage sont calculées par la fonction $f_{3a}^{-1}(x, p) = x - 2p - 5$. La garde ajoutée exprime l'adéquation des valeurs de x avec celles de y tel que $f_4(f_3^{-1}(x)) = 2x - 2p - a - 3$.

Les transitions *Secours* et *Immobilisation* sont plus simples à transformer. En effet, la transition *Secours* est basique. Il suffit donc d'inverser le sens des arcs entrant et sortants (voir Fig.7.5.a). La transition *Immobilisation* est mixte il suffit

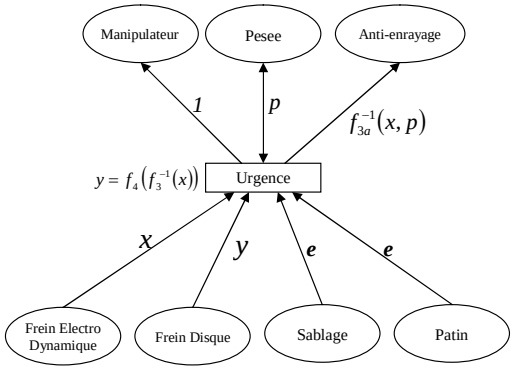


FIG. 7.4 – Inversion de la transition Urgence

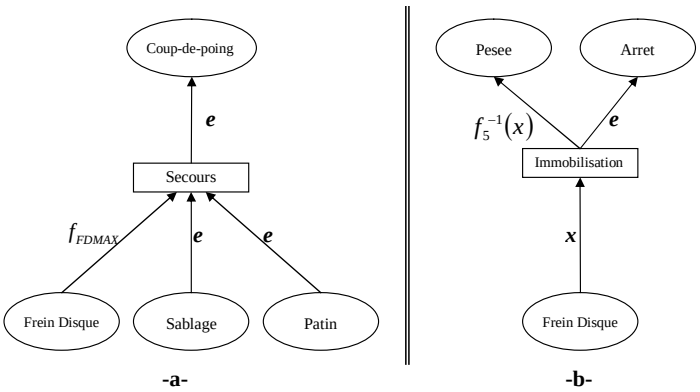


FIG. 7.5 – Inversion des transitions Secours et Immobilisation

donc d'inverser la direction des arcs et d'associer à l'arc $(Immobilisation, Pesee)$ l'inverse de la fonction f_5 qui est $f_5^{-1}(x) = (x-8)/2$ (voir Fig.7.5.b).

L'inversion des transitions *Ser*, *Urg*, *Imm* et *Sec* est montrée dans les schémas de Fig.7.6. Différentes inversions sont possibles. La meilleure d'entre elles est celle qui exploite le maximum d'informations connues. Dans notre cas, l'un des objectifs des analyses est la validation d'un *"profil de vitesse"*, i.e. l'étude du passage imposé d'une vitesse à l'autre à l'approche de certaines sections de voie. Le profil nous renseigne donc sur les valeurs successives des vitesses et donc il n'y a plus besoin de calculer l'inverse des fonctions g_{1v}^{-1}, g_{2v}^{-1} et g_{3v}^{-1} car leurs résultats sont connus (la vitesse avant la baisse sera notée v_1 et après la baisse v_2). Les inverses des fonctions g_1 et g_2 donnant les jetons des freins à disques sont : $g_{1fd}^{-1}(v1, v2, fed) = 25(1 - \frac{v_2}{v_1}) - \frac{15}{8}fed$ et $g_{2fd}^{-1}(v1, v2, fed) = 20 - 40\frac{v_2}{v_1} - \frac{5}{4}fed$.

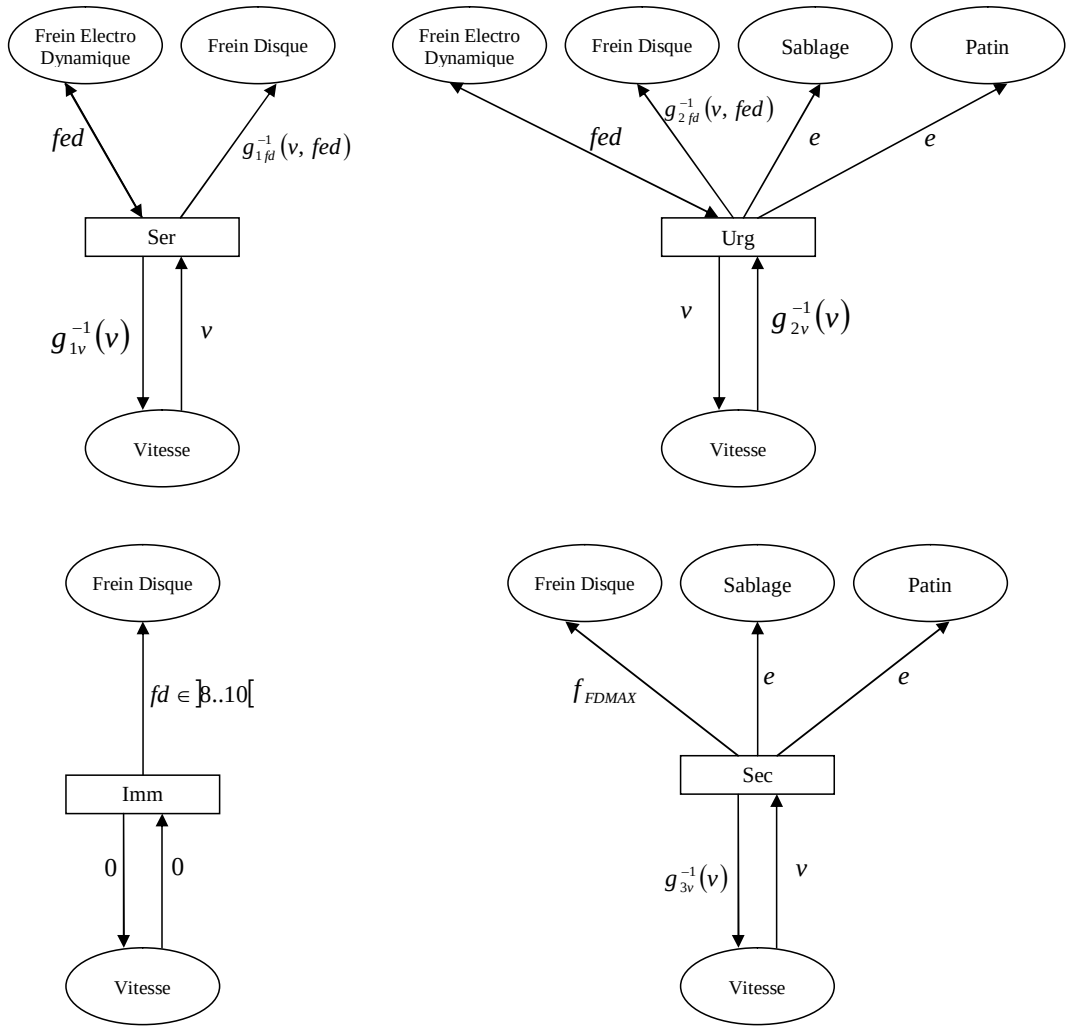


FIG. 7.6 – Inversion des transitions liées au calcul de la vitesse

7.5 Analyse du système

7.5.1 Le freinage de service

La première analyse concerne le freinage de service. Le processus consiste à provoquer un freinage nécessitant l'enclenchement, à la fois, le frein électrodynamique et le frein à disques avec une valeur prédéfinie sur le premier (par exemple 5). Le but est de trouver les valeurs possibles des consignes appliquées au frein à disques ainsi que les valeurs de commande (pesée p et anti-enrayage a) qui les provoquent. Nous considérons dans ce cas que la valeur du manipulateur est connue : $m = 0,5$.

L'analyse par l'accessibilité arrière utilise le RdPC inverse et prend en compte, dans l'état partiel, la valeur 5 du frein électrodynamique et la valeur $0,5$ du manipulateur. Le tir (arrière) de la transition *Service* se fait grâce à un enrichissement de marquage. En effet, les consignes du frein à disques, nécessaires au tir arrière de la transition, sont inconnues. Elles sont donc ajoutées comme hypothèses supplémentaires signifiant que le dispositif de freinage à disques a été enclenché. Le calcul se fait en utilisant la formule de la garde associée à la transition ainsi que la contrainte $2p + a = 2,5$ obtenue et vérifiée aisément dans chaque élément du triplet $f_1^{-1}(x)$ lui même faisant partie de la garde. Les résultats sont illustrés dans Fig.7.7. L'algorithme de recherche par accessibilité arrière des valeurs des entrées s'écrit comme suit :

for $p \in [0..1]$ **do**
  $a \leftarrow 2.5 - 2p$
  if $0 \leq a \leq 1$ **then**
    Return $f_2(f_1^{-1}(x))$ /*$x = 5$, $m = 0,5$*/
  end if
end for

La conclusion de ce cas d'analyse est que le marquage final $M_f = \langle 5 \rangle.FreinElectroDynamique$ est accessible, en tirant la transition *Service*, par le marquage initial $M_0 = \langle 0,5 \rangle.Manipulateur + \langle \hat{p} \rangle.Pesee + \langle \hat{a} \rangle.AntiEnrayage$ tel que $\hat{a} = 2.5 - 2\hat{p}$ et $0 \leq \hat{a} \leq 1$. Le marquage final réellement accessible est le marquage enrichi $M_{fe} = \langle 5 \rangle.FreinElectroDynamique + \langle \hat{y} \rangle.FreinDisque$. Les valeurs de $\hat{y}$, $\hat{p}$ et $\hat{a}$ sont les combinaisons possibles des variables y, p et a (respectivement) qui correspondent au segment de droite de Fig.7.7.

Pour obtenir les mêmes résultats en analyse par accessibilité avant, l'algorithme de recherche de consigne de frein à disques doit discriminer parmi les valeurs de y celles qui sont effectivement possibles en utilisant la contrainte $f_1(m, p, a) = 5 \wedge m = 0,5$. Par contre, dans l'état, aucune contrainte ne lie les variables p et a. L'algorithme d'analyse par accessibilité avant s'écrit comme suit :

for $a \in [0..1]$ **do**

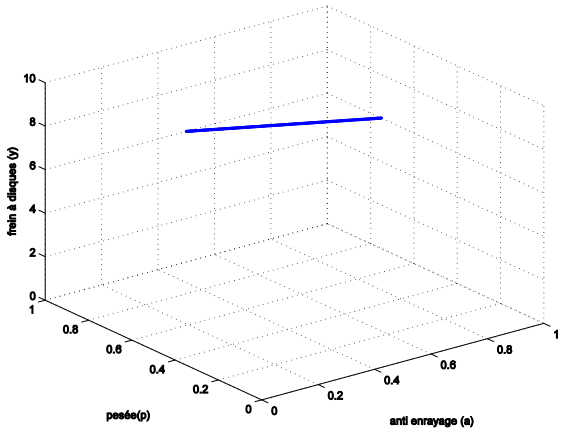


FIG. 7.7 – Analyse du freinage de service

for $p \in [0..1]$ **do**
  if $f_1(m, p, a) = 5$ **then**
    Return $f_2(m, p, a)$
  end if
end for
end for

La différence entre les deux algorithmes (accessibilité arrière et accessibilité avant) réside dans la complexité algorithmique. Sachant que la complexité des fonctions f_2 et $f_2(f^{-1})$ est strictement identique, la différence entre les deux est que le second algorithme présenté (accessibilité avant) comporte une imbrication de deux boucles alors que le premier n'en possède qu'une seule. Ceci est dû au fait que le RdPC inverse exploite la relation de parallélisme (transformation parallèle) et introduit ainsi une connaissance à priori du modèle étudié réduisant de la sorte le nombre de calculs à effectuer.

7.5.2 Le frein à disques

La deuxième analyse concerne le diagnostic d'un état particulier du frein à disques. L'état connu du système est une valeur de consigne du frein à disques. Dans ce cas d'étude, elle est donnée par la valeur 9. Le but est de retrouver les états initiaux (M_0) qui ont pu engendrer l'état partiel final ($M_f = \langle 9 \rangle.FreinDisque$) et ceci à partir de la seule connaissance de ce dernier. L'analyse se fait par accessibilité arrière utilisant le RdPC inverse avec un état initial correspondant à M_f. Le processus d'analyse est détaillé transition par transition de la plus simple à la plus complexe.

La transition *Immobilisation*

L'inversion de la transition *Immobilisation*, illustrée dans Fig.7.5.b, montre que le tir de celle-ci est suffisant connaissant juste la valeur du jeton dans la place *Frein Disque*. Le tir de cette transition produit donc un jeton anonyme dans la place *Arret* et un jeton dans la place *Pesee* dont la valeur est calculée par la formule $f_5^{-1}(x) = (x-8)/2$. Cette formule appliquée à une valeur de $x = 9$ donne $f_5^{-1}(x) = 0,5$. La conclusion est que le marquage $M_f = \langle 9 \rangle.FreinDisque$ peut être produit par un marquage initial $M_0 = \langle 0,5 \rangle.Pesee + \langle e \rangle.Arret$. Ce résultat est obtenu naturellement en une seule opération contrairement à ce qu'aurait été une recherche par accessibilité avant qui testerait toutes les entrées possibles de la pesée (variable p) pour trouver le marquage initial M_0 source de M_f. Le gain peut se résumer en une diminution de la complexité algorithmique dans le processus du diagnostic.

La transition *Secours*

L'inversion de la transition *Secours*, illustrée dans Fig.7.5.a, montre que le tir de celle-ci est possible si et seulement si chacune des places *Sablage* et *Patin* possède un jeton anonyme et si la place $FreinDisque$ possède un jeton d'une valeur f_{FDMAX}. Cette valeur est fixée à 10 [44] par le modèle et est différente du marquage partiel à analyser. La transition *Secours* ne peut donc pas être tirée même en enrichissant le marquage dans *Sablage* et *Patin* ; la transition *Secours* n'est pas potentiellement franchissable. La conclusion est que le marquage $M_f = \langle 9 \rangle.FreinDisque$ est inaccessible par le tir de *Secours*.

La transition *Urgence*

Le tir inverse de la transition *Urgence* nécessite un enrichissement de marquage au niveau des transitions *Sablage*, *Patin* et $FreinElectroDynamique$. Dans les deux premières places les jetons à rajouter sont anonymes et aucun calcul n'est à faire. Par contre, les valeurs des jetons dans $FreinElectroDynamique$ (et donc de la variable x), doivent satisfaire la garde. Pour simplifier les calculs, nous allons réécrire la garde qui donne les valeurs de x en fonction de celle de y. Cette garde peut être écrite sous la forme $x = f_3(f_4^{-1}(y))$ ce qui donne $x = 2y - 2p - a - 9$ sous la contrainte $2p + a = y - 7$.

Par un processus proche de ce qui a été utilisé dans la section 7.5.1, les valeurs des consignes du frein électrodynamique sont calculées ainsi que celles des commandes de pesée et d'anti-enrayage qui les engendrent. Le résultat est illustré dans Fig.7.8. La conclusion est que le marquage $M_f =$

[44] La valeur $f_{FDMAX} = 10$ est obtenue en appliquant les valeurs maximales des consignes a et p dans la formule de la fonction $f_4(p, a)$.

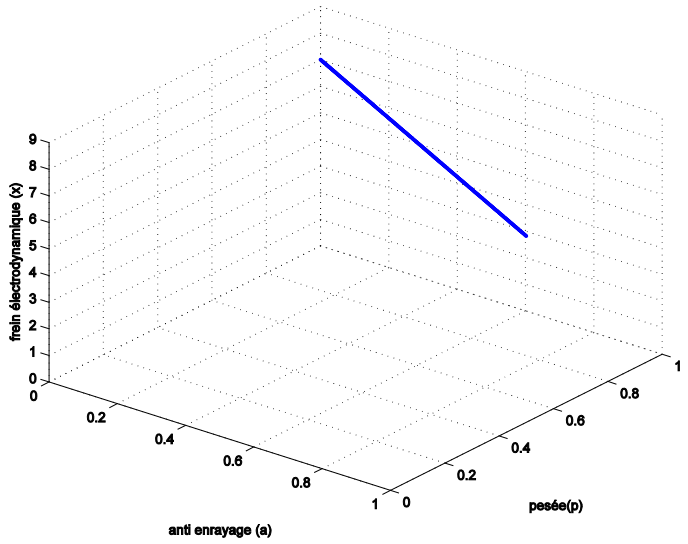


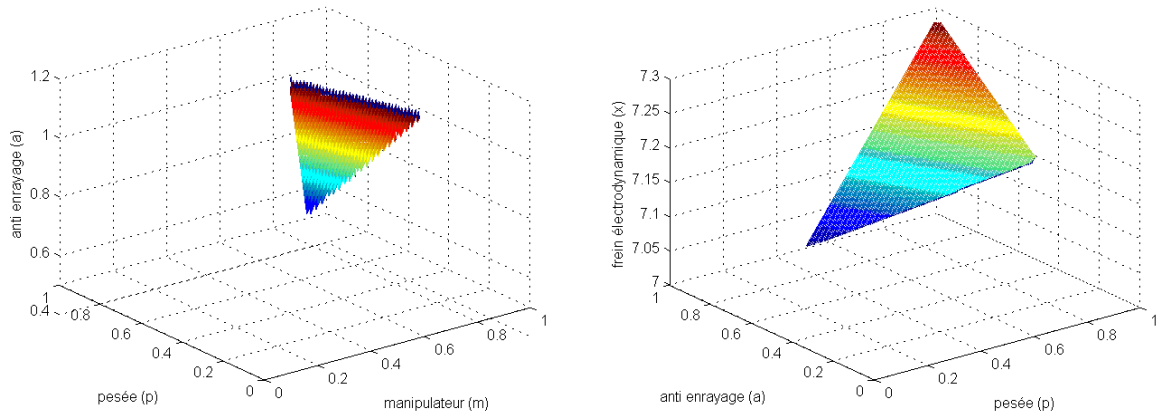
FIG. 7.8 – Freinage d'urgence

$\langle 9 \rangle.FreinDisque$ est accessible à l'arrière par $M_0 = \langle 1 \rangle.Manipulateur + \langle \hat{p} \rangle.Pesee + \langle \hat{a} \rangle.AntiEnrayage$. Cette accessibilité est conditionnée par l'enrichissement du marquage. Le marquage final réellement accessible est le marquage enrichi $M_{fe} = \langle \hat{x} \rangle.FreinElectroDynamique + \langle 9 \rangle.FreinDisque + \langle e \rangle.Sablage + \langle e \rangle.Patin$. Les valeurs de $\hat{x}$, $\hat{p}$ et $\hat{a}$ sont les combinaisons possibles des variables x, p et a (respectivement) qui correspondent au segment de droite de Fig.7.8.

La transition *Service*

Comme pour la transition *Urgence*, le tir inverse de la transition *Service* nécessite un enrichissement de marquage. La place à enrichir est $FreinElectroDynamique$. Les valeurs des jetons dans cette place (donc de la variable x), doivent satisfaire la garde. Comme la valeur de y est connue, la garde est réécrite pour retrouver aisément celles de x. La nouvelle garde est sous la forme $x = f_1(f_2^{-1}(y))$. Sachant que $y \neq 0$, la condition dans la fonction f_2 est valide, i.e. dans le RdPC original, la condition $p + a > 1,5$ est valide et la valeur du jeton dans la place $FreinDisque$ (la valeur 9 dans cet exemple) est obtenue par la formule $7m + 2p + 1$. La nouvelle garde est donnée donc sous la forme $(0 < m < 1) \wedge (3y - 21m - 4p - a > 3) \wedge (x = (19y - 98m - 24p - 12a)/7)$ sous la contrainte $a = y - 7m - 2p$.

En exploitant le RdPC inverse ainsi complété, les valeurs des consignes du frein électrodynamique sont calculées ainsi que celles des commandes de pesée et d'anti-enrayage qui les engendrent. Les résultats ne peuvent pas être montrés en un seul graphique (trop de variables), c'est pourquoi ils sont schématisés sur les deux graphiques de Fig.7.9. Le premier (Fig.7.9.a) montre les combinaisons



-a- **-b-**

FIG. 7.9 – Freinage de service

d'entrée possibles des variables m, p et a. Le second (Fig.7.9.b) reprend deux variables d'entrée (m et p) et les valeurs de x correspondantes. La conclusion est que le marquage $M_f = \langle 9 \rangle.FreinDisque$ est accessible à l'arrière par $M_0 = \langle \hat{m} \rangle.Manipulateur + \langle \hat{p} \rangle.Pesee + \langle \hat{a} \rangle.AntiEnrayage$. Cette accessibilité est conditionnée par l'enrichissement du marquage. Le marquage final réellement accessible est le marquage enrichi $M_{fe} = \langle \hat{x} \rangle.FreinElectroDynamique + \langle 9 \rangle.FreinDisque$. Les valeur de $\hat{x}$, $\hat{m}$, $\hat{p}$ et $\hat{a}$ sont les combinaisons possibles des variables x, m, p et a (respectivement) qui correspondent aux valeurs illustrées dans Fig.7.9.

Récapitulatif du cas *freins à disques*

L'analyse par accessibilité arrière du marquage final considéré dans ce cas d'étude, à savoir $M_f = \langle 9 \rangle.FreinDisque$, montre qu'il est accessible par trois chemins différents. Chaque chemin correspond à un scénario bien spécifique. Le premier chemin passe par la transition *Immobilisation* qui signifie une détection d'arrêt avec une pesée à $0,5$. Les chemins passant par *Service* et *Urgence* nécessitent une des informations complémentaires sur l'état final du système. C'est pourquoi, le marquage final a été enrichi de telle sorte à supposer un état cohérent. Les résultats de ce cas d'étude sont illustrés dans l'arbre de Fig.7.10

7.5.3 Cas de décélération forte

L'action des différents éléments du freinage provoque un effet de baisse directement visible sur la vitesse du véhicule. Le but de cette analyse est d'explorer les causes d'une baisse "brutale" de la vitesse. Les états connus du système sont les vitesses avant et après la baisse. Nous considérons que la vitesse initiale (avant la baisse) est de $100km/h$ et, après le freinage, cette vitesse est passée à $40km/h$.

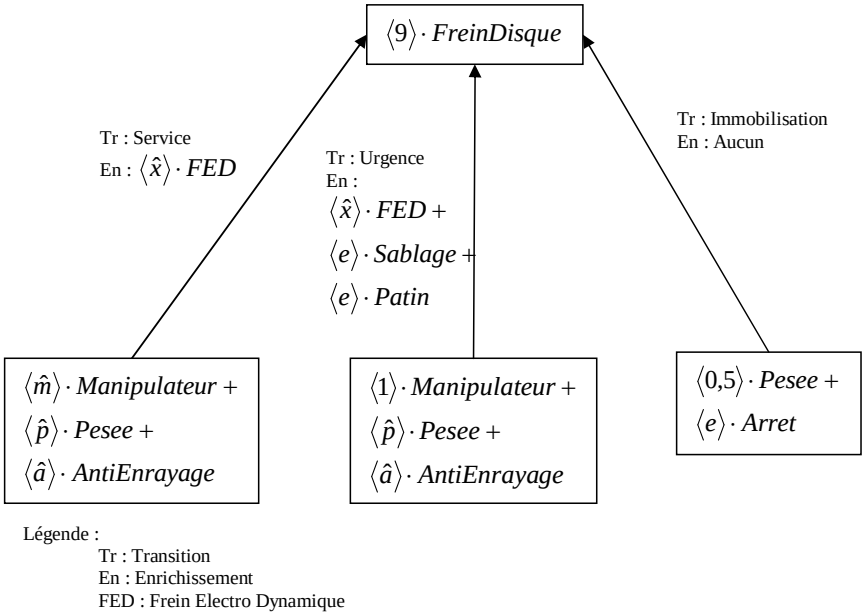


FIG. 7.10 – Arbre d'accessibilité arrière du frein à disques

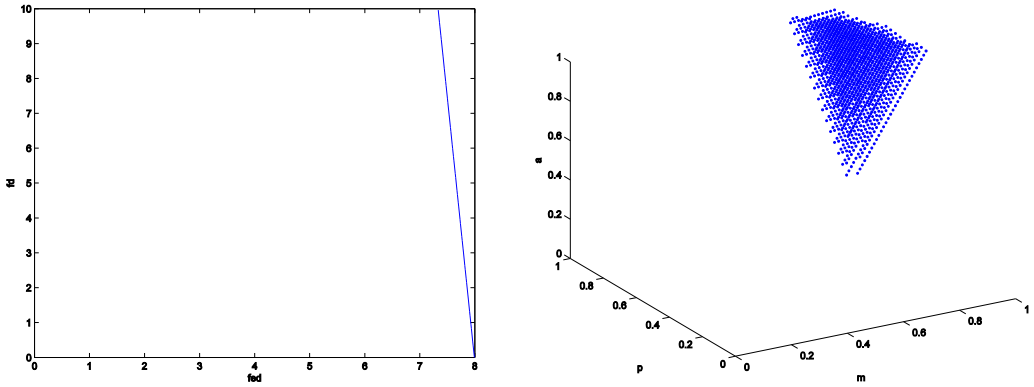
L'analyse par accessibilité arrière va tenter de répondre à la question : quelles sont les causes du passage de la vitesse du véhicule de $100km/h$ à $40km/h$? Posons $M_f = \langle 100 \rangle.Vitesse$ et $M_{f-1} = \langle 40 \rangle.Vitesse$.

Transitions *Ser/Service*

Le tir inverse de *Ser* produit deux jetons (en plus de celui de valeur 30 dans la place *Vitesse*). Le premier dans la place *FreinElectroDynamique* (variable fed), le second dans *FreinDisque* (variable fd). La relation entre les deux est donnée par la formule $fd = 15 - 1,875fed$. Les valeurs possibles de fd et fed sont illustrées dans Fig.7.11.a. À chaque combinaison du couple (fed, fd), correspond un plan des combinaisons possibles des variables m, p et a. Ces valeurs sont illustrées dans Fig.7.11.b.

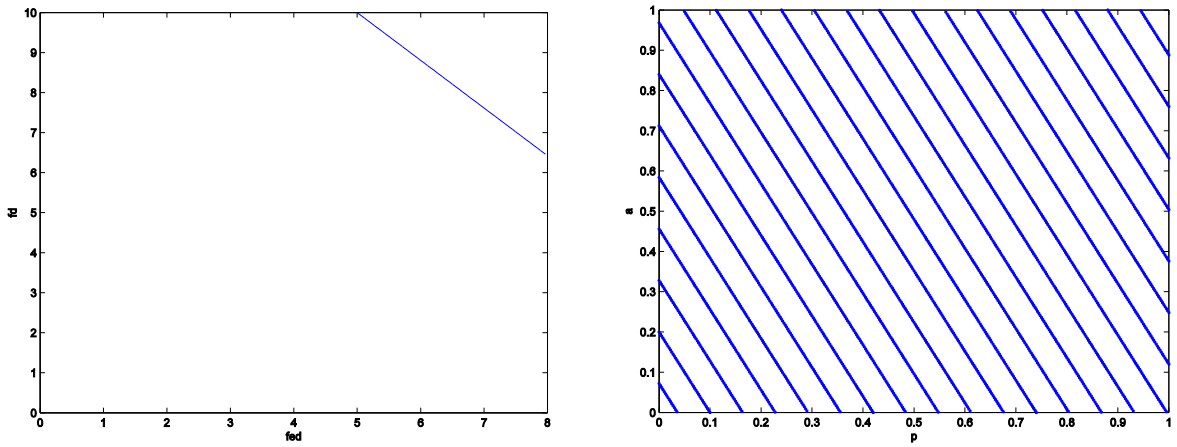
Transitions *Urg/Urgence*

Le tir inverse de *Urg* produit quatre jetons : un dans *FreinElectroDynamique* (variable fed), un dans *FreinDisque* (variable fd) et un jeton anonyme dans chacune des deux places *Sablage* et *Patin*. La relation entre les deux premiers est donnée par la formule et $fd = 16 - 1,2fed$. Les valeurs possibles de fd et fed (après élimination des valeurs impossibles) sont illustrées dans Fig.7.12.a. À chaque combinaison du couple (fed, fd) plus la présence des jetons dans *Sablage* et *Patin*, correspond une droite des combinaisons possibles des variables p et a (la variable m dans ce cas est toujours égale à 1). Ces valeurs sont illustrées dans Fig.7.12.b.



-a- -b-

FIG. 7.11 – Réduction forte de la vitesse : transitions Ser et Service



-a- -b-

FIG. 7.12 – Réduction forte de la vitesse : transitions Urg et Urgence

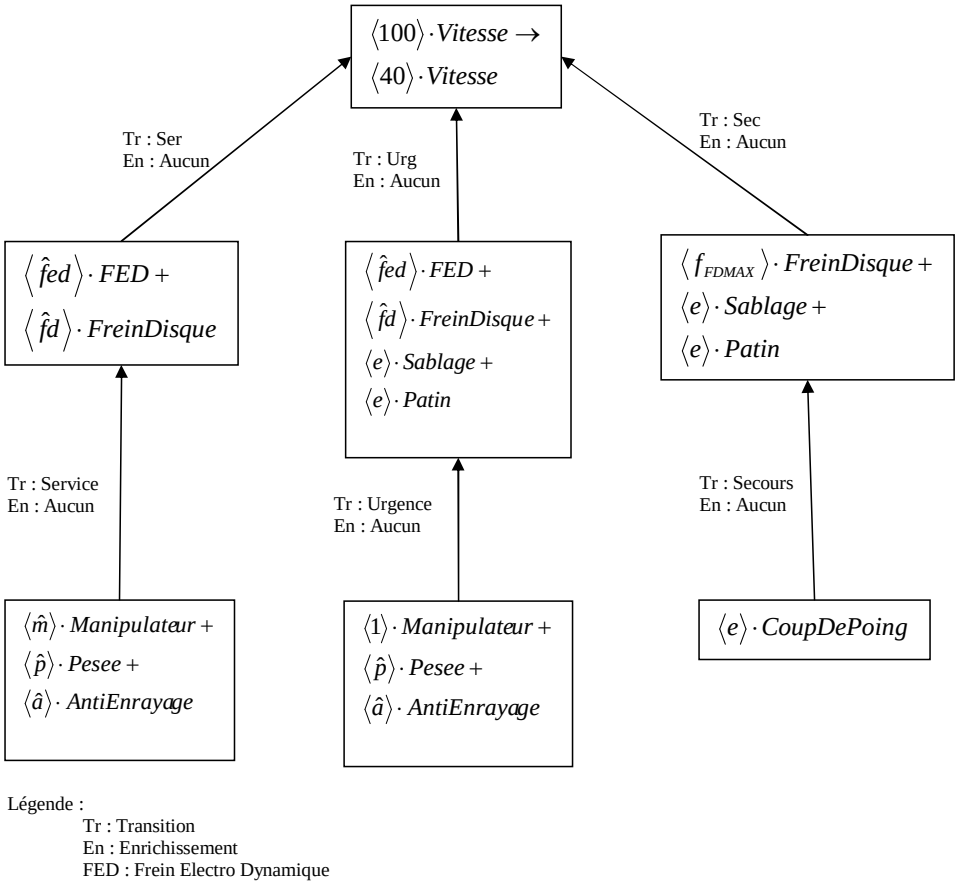


FIG. 7.13 – Arbre d'accessibilité arrière de la réduction forte de la vitesse

Transition *Imm*

Le tir de la transition *Imm* nécessite une valeur initiale nulle de la vitesse. Puisque cette condition n'est pas satisfaite, le tir est donc impossible.

Récapitulatif du cas *décélération forte*

L'arbre de l'accessibilité arrière pour le cas de la décélération forte, illustré dans Fig.7.13, montre trois niveaux. Le premier niveau (la racine) représente le marquage final connu, c'est-à-dire, la réduction forte de la vitesse. Le deuxième niveau représente les causes directes possibles de cette réduction. Elles sont au nombre de trois. Chacune des causes décrit les dispositifs de freinage qui sont mis en œuvre pour réduire fortement la vitesse. Le dernier niveau (les feuilles de l'arbre) représente, à la fois, les causes indirectes du marquage final et les causes directes de l'enclenchement des dispositifs de freinage. Le résultat final est donc un ensemble de scénarios menant des consignes initiales du systèmes vers l'état final connu.

7.5.4 Réduction de la vitesse suivant un profil

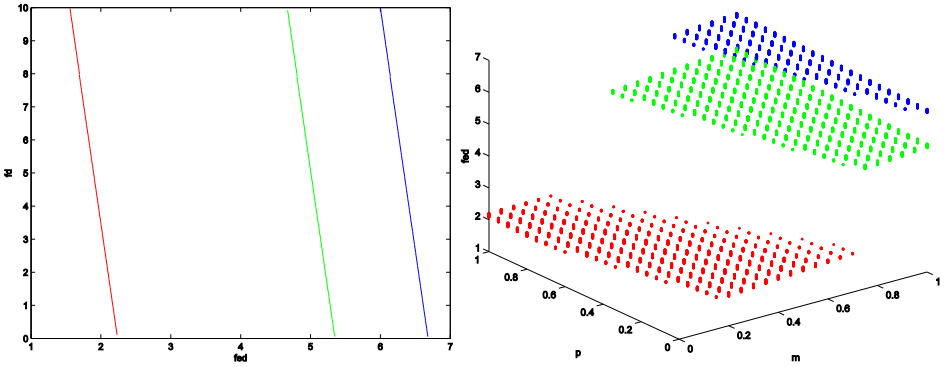
Le but de ce cas d'étude est de trouver, à partir d'un profil donné, les combinaisons de commandes en entrée qui permettent de le réaliser en utilisant uniquement le freinage usuel (freinage de service). Les données en entrée sont une liste des vitesses successives que doit avoir le véhicule. Dans cette étude, elle est donnée par la suite 120, 100, 50, 30 (les vitesses sont exprimées en km/h).

L'analyse par accessibilité arrière procède sur les trois paliers : le passage $120 \rightarrow 100km/h$, le passage $100 \rightarrow 50km/h$ et enfin, le passage $50 \rightarrow 30km/h$. Les codes couleurs associés sont respectivement : rouge, bleu et vert. À chaque étape, deux niveaux sont étudiés. Le premier, calcule les valeurs des consignes appliquées pour le passage d'une vitesse à l'autre tel qu'illustré dans Fig.7.14.a. Ces valeurs sont calculées en exploitant la fonction $g^{-1}_{1fd}(v1, v2, fed)$. Le second niveau donne, en fonction des résultats du premier niveau, les valeurs des consignes à injecter dans le système pour avoir le freinage demandé. Les résultats sont illustrés dans Fig.7.14.b. Chaque volume représente les différentes combinaisons possibles des consignes $Manipulateur$ et $Pesee$ nécessaires pour le passage d'une vitesse à l'autre. L'axe des z est associé à l'effort en freinage électrodynamique et l'épaisseur des volumes représente les variations possibles de l'anti-enrayage.

Dans Fig.7.14.b, nous constatons que le volume rouge (passage $120 \rightarrow 100km/h$) et bleu (passage $100 \rightarrow 50km/h$) ne se recouvrent pas (par rapport à la variable m). Cela signifie que pour toute valeur de $Pesee$, la baisse demandée nécessite le changement de la consigne du $Manipulateur$. La raison principale de ce constat est la grande différence de l'effort de freinage qu'il faudrait appliquer entre les deux étapes successives du profil. Par contre, le volume bleu (passage $100 \rightarrow 50km/h$) et vert (passage de $50 \rightarrow 30km/h$) se recouvrent (par rapport à la variable m). La zone de recouvrement devient de plus en plus large au fur et à mesure que la consigne p est grande. Ceci signifie l'existence, pour toute consigne p, de valeur de m fixe qui permet le passage d'un palier de vitesse à l'autre. Le seul paramètre à régler est la consigne de l'anti-enrayage.

La prochaine étape est d'essayer de corriger le profil pour un moindre mouvement du manipulateur sachant que la pesée est constante d'un palier à l'autre. Nous nous intéresserons particulièrement aux deux premier paliers. Pour ce faire, nous considérons une vitesse intermédiaire inconnue vi telle qu'elle soit la limite entre les deux paliers, i.e. les deux paliers s'écriront $120 \rightarrow vi$ et $vi \rightarrow 50$. L'analyse par accessibilité arrière recherche les différents comportements de passage d'un palier à l'autre en fonction de la vitesse intermédiaire. Quelques résultats sont illustrés dans Fig.7.15. La vue sur l'axe des z, dans les planches, a été volontairement écrasée pour permettre de distinguer clairement le chevauchement (ou non) des volumes.

Les résultats montrent que pour $vi \in [50..63] \cup [96..120]$, aucun recouvrement



-a- **-b-**

FIG. 7.14 – Résultats d'analyse de la réduction de vitesse selon profil

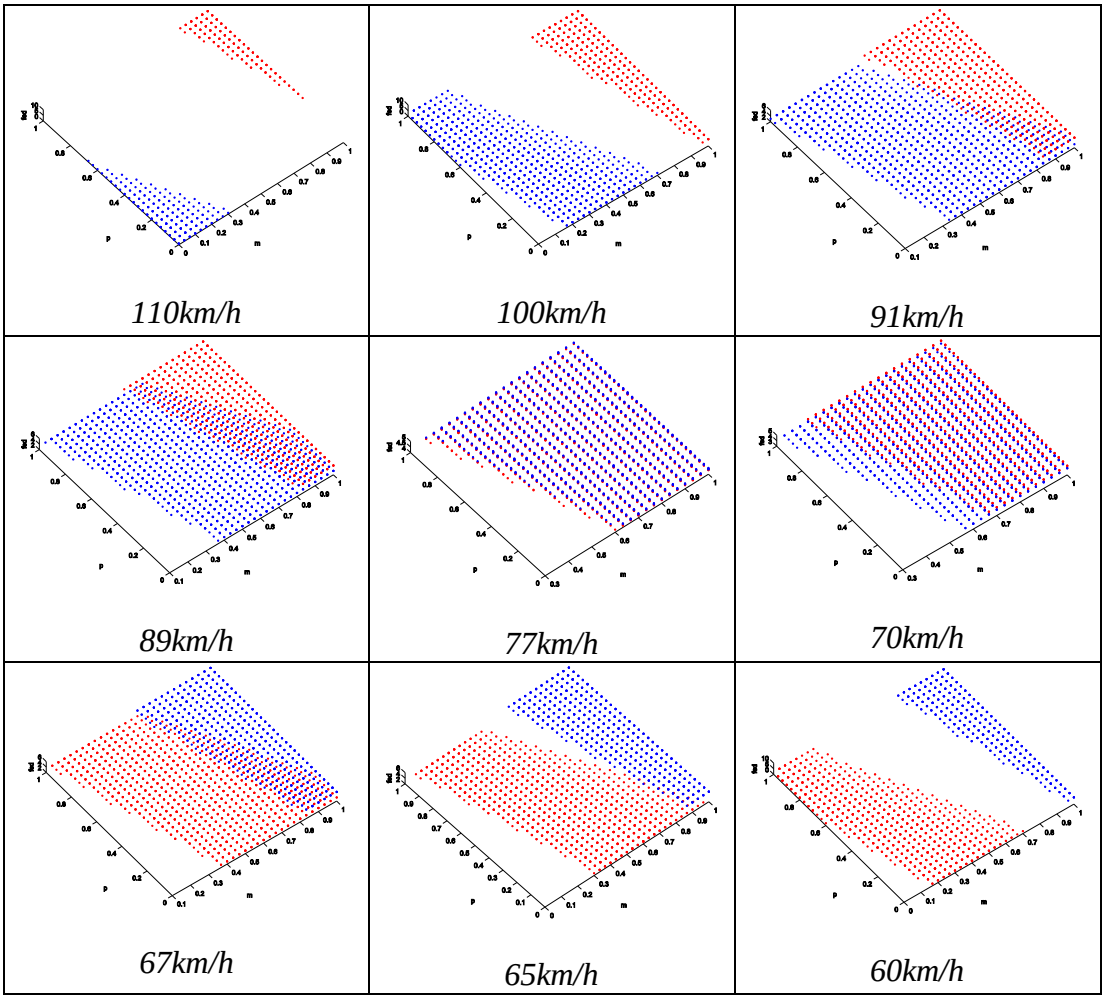


FIG. 7.15 – Résultats d'analyse sur la correction de profil de vitesse

n'existe entre les volumes. Ceci signifie que le passage d'un palier à l'autre nécessite de bouger la position du cran. Pour $vi[64..67] \cup \in [91..95]$, les volumes se superposent partiellement, ce qui veut dire que le passage d'un palier à l'autre peut se faire sans bouger le cran (en agissant juste sur l'anti-enrayage) à condition que la valeur de la pesée soit dans l'intervalle de la superposition. Enfin, pour $vi \in [68..90]$, le recouvrement est total, c'est-à-dire que pour toute valeur de pesée, le passage entre les deux paliers peut se faire sans bouger le manipulateur. Pour choisir une valeur optimale de vi, d'autres critères doivent être pris en compte. Il s'agirait par exemple d'imposer des valeurs de freinage proches entre les paliers (freinage sans à-coups). Dans ce cas, les valeurs du centre de l'intervalle (autour de $80km/h$ et notamment $77km/h$) sont les plus intéressantes.

7.6 Conclusion

Nous avons appliqué, au cours de ce chapitre, la méthode d'analyse par accessibilité arrière utilisant le RdPC inverse. Le cas proposé est un modèle de freinage de tramway. Les différentes analyses effectuées ont été conduites suivant des but précis montrant une étendue de possibilités d'usage de cette technique. La première analyse avait comme point de départ un état final partiel accompagné d'un scénario (connaissance de transitions tirées). Le but est diagnostiquer les causes de cet état suivant le scénario donné. La deuxième analyse part d'un état partiel sans aucun scénario. L'objectif est de trouver les scénarios possibles ainsi que les valeurs initiales des jetons qui ont permis l'état final donné. La troisième analyse prospecte les causes d'une évolution possible d'un marquage final. Ce cas de figure a été étendu pour trouver les entrées possibles du système qui font évoluer un état final selon un profil donné. Enfin, la dernière analyse tente de corriger le profil (l'état final) pour répondre à des contraintes imposées sur les entrées.

Les avantages de la méthode de l'analyse par accessibilité arrière des RdPC sont multiples. Tout comme dans les RdP ordinaires, cette méthode apporte une connaissance sur des scénarios correspondant au diagnostic par modèle. Le cadre des RdPC rajoute toute la puissance et l'expressivité liées au traitement des valeurs des jetons. Dans l'exemple traité, la prise en compte implicite des contraintes (liant les variables entrantes entre elles et/ou les variables entrantes avec les valeurs de sortie) dans le RdPC inverse produit une diminution de la complexité algorithmique par rapport au modèle RdPC original. Dans le cas général, si le RdPC inverse ne réduit pas la complexité de la recherche du marquage source, il ne l'augmente pas. En effet, même dans le pire cas (ou l'inverse est difficile à formuler), le recours à la programmation pas contraintes revient à effectuer la recherche dans le RdPC original. L'inconvénient principal de la méthode proposée est lié à l'effort nécessaire à priori pour inverser le modèle. Cette opération exige

la meilleure prise en compte possible de l'information disponible à propos de la structure statique et du comportement dynamique du système.

Conclusion

Les problématiques de diagnostic, et plus généralement, de sûreté de fonctionnement dans le domaine des systèmes embarqués sont si larges qu'une thèse ne peut en traiter qu'une partie. Par conséquent, ce travail ne prétend ni l'exhaustivité ni la complétude.

Cette thèse a choisi de se focaliser sur angle particulier qui est l'analyse et la vérification basées sur des modèles, représentant dans le même formalisme, les deux facettes d'un système embarqué : la structure statique et le comportement dynamique. L'axe principal de l'étude est de proposer une méthode formelle dont le but est double. Le premier est de pouvoir chercher la source d'une défaillance connue sur le système (diagnostic). Le deuxième est de pouvoir réaliser des analyses pour vérifier/valider un système par rapport à une spécification donnée ; et le cas échéant proposer un paramétrage du système pour qu'il réponde à cette spécification.

Une telle étude nécessite un modèle approprié pouvant représenter la structure et le comportement dynamique du système mais aussi possédant de fortes bases formelles. Le modèle choisi, répondant à ces critères est l'extension colorée des réseaux de Petri. Sur ces RdPC, nous proposons de développer la méthode de l'analyse structurelle par accessibilité arrière basée sur inversion du modèle. Cette méthode est étudiée et a fait ses preuves dans le domaine des RdPC ordinaires.

L'inversion des RdPC pose des défis liés à la sémantique même des modèles. La littérature propose quelques solutions dans ce sens soit sur des modèles particuliers comme le modèle *Behavioral Petri Net* [Por93] soit en écrivant la problématique de l'inversion sous forme de contraintes comme les travaux de [CHC96], ce qui revient autour d'une transition à une analyse par accessibilité avant. Nous avons exploré et généralisé l'approche introduite par [Kha03] en appliquant l'inversion structurelle aux RdPC. Dans cette optique, la sémantique des RdPC, et notamment autour des transitions, affecte directement la transformation à appliquer. C'est pourquoi, la proposition faite inclut ce paramètre et préconise un certain nombre d'algorithmes de transformation. Chacun d'eux est approprié à une famille de structures. La définition du RdPC inverse n'est qu'une partie de la problématique. L'autre partie est la conduite d'une analyse complète par accessibilité arrière. Ceci n'est possible que grâce à la définition (plutôt, généralisation)

de mécanismes complémentaires tels que le repérage des transitions potentiellement franchissables et l'établissement de l'ordre partiel de tir des transitions. Le mécanisme complémentaire le plus important est l'enrichissement de marquage qui permet de poser des hypothèses sur le fonctionnement passé du système.

La définition du RdPC inverse et la mise en œuvre de l'analyse par accessibilité arrière sont présentées au départ d'une manière intuitive. Pour consolider l'approche proposée avec une base formelle, une large partie de cette étude est consacrée aux considérations théoriques. L'aspect théorique de cette étude se subdivise en deux pistes : l'algèbre linéaire et la logique linéaire. Le choix de chaque formalisme s'est effectué pour répondre à des besoins spécifiques en s'inspirant des travaux passés pour définir leurs cadres d'utilisation. Le premier formalisme (l'algèbre linéaire), est largement inspiré de [Had89]. L'objectif de son application à l'inversion des RdPC est de montrer la pertinence des algorithmes de transformation proposés. À partir du constat que l'inversion des modèles, à la base, se résume au changement de la direction des arcs (principe de l'inversion des modèles RdP ordinaire), l'algèbre linéaire permet de prouver le passage à la forme généralisée du RdPC. Le second formalisme étudié est la logique linéaire. Son utilisation est inspirée notamment de [Kha03] qui l'a appliquée à l'accessibilité dans les RdP ordinaire. Cette étude reprend donc le même principe. Mais la sémantique liée aux RdPC fait que la généralisation n'est pas directe. Une phase d'identification du fragment de la logique linéaire vers lequel sera traduit les RdPC était nécessaire. Le fragment en question est MILL1 (*First Order Multiplicative Intuitionistic Linear Logic*). Une fois cette phase d'identification terminée et les RdPC traduits, l'accessibilité en RdPC est vérifiée grâce aux preuves de séquents.

L'étude conjointe des deux formalismes nous a permis de les confronter et d'établir une "*comparaison*" entre les deux. L'avantage de l'algèbre linéaire est de préserver la structure même du modèle et d'établir directement les preuves sans aucune formalisation intermédiaire. Son inconvénient, outre la difficulté de prise en main, est son caractère local. En effet, les preuves dans ce formalisme se compliquent considérablement dans les cas de transitions complexes ou d'analyse sur un modèle complet. Concernant la logique linéaire, les arguments qui plaident pour elle sont sa prise en main aisée, sont automatisation facile et son caractère d'analyse de modèles complets même complexes. Néanmoins, l'utilisation de la logique linéaire a nécessité le développement d'une méthode spécifique de traduction de RdPC en logique linéaire. Ce qui implique donc l'introduction d'un nouveau formalisme, autre que les réseaux de Petri.

Dans les utilisations pratiques, l'apport de la méthode réside dans la prise en compte maximale de l'information connue ou supposée du modèle à étudier. Ceci peut se manifester sous plusieurs formes comme la diminution de la complexité algorithmique dans la recherche de marquages prédécesseurs et les suppositions sur

un état de fonctionnement passé. Cette prise en compte demande une attention et un effort particuliers lors de la construction du modèle RdPC inverse. En effet, plus les informations et les contraintes sont prises en compte, plus précise est l'analyse, et souvent plus simple est l'inversion. Le recours à l'enrichissement du marquage doit être limité au strict minimum pour deux raisons. D'abord, cette opération apporte de l'information supposée sur le fonctionnement du système et pour une meilleure précision de l'analyse il faut maximiser l'utilisation de l'information connue. Ensuite, cette opération ne peut pour l'instant être automatisée et elle l'est difficilement. Une piste très intéressante pour un travail futur consiste à étudier l'automatisation du mécanisme d'enrichissement du marquage. La deuxième piste à explorer dans ce contexte est de profiter des outils existants notamment dans le domaine de la logique linéaire pour mettre en œuvre la démarche d'accessibilité arrière par RdPC.

Comme dans toute méthode, l'approche que nous proposons possède ses limites. D'abord, l'état actuel des travaux, certaines structures ne sont pas traitées. En effet, la variété des structures (de transitions) des RdPC fait qu'une méthode exhaustive est difficilement concevable. En plus de cette variété de structure qui est une vision plutôt locale de l'approche proposée, la mise en œuvre de l'analyse par accessibilité arrière (vision globale) fait intervenir l'interaction entre structures. Ces interactions ont des répercussions directes sur les résultats de l'analyse. L'exemple de la difficulté de traitement des boucles (cycles) est très illustratif. En effet, la difficulté est sur deux niveaux : local et global. Au niveau local, un cycle est souvent difficile à caractériser et donc à inverser dans le cas des RdPC. Les résultats existants dans le domaine des RdP ordinaire, comme la minimalité des cycles, ne sont pas applicables dès que les jetons sont caractérisés par des valeurs qui évoluent. Au niveau global, le fait d'inverser une boucle ne signifie pas forcement une conduite d'analyse facile. L'exemple est donné par une boucle qui se comporte comme générateur de jetons ce qui fait souvent appel, dans le RdPC inverse, à un maximum d'enrichissement de marquages. Les limitations de l'approche proposée à l'état actuel imposent une vigilance également dans la conception du modèle RdPC initial. Si ce modèle comporte des structures limitant l'analyse, il faudrait soit les inverser au cas par cas en prenant en compte l'information disponible (comme dans l'inversion des boucles dans le cas d'étude du chapitre 7), soit agir plus en amont en retravaillant la modélisation initiale. Si les deux solutions s'avèrent insuffisantes, le modèle ne peut donc pas être analysé par l'approche proposée.

Le but visant à exploiter les structures des modèles RdP en général et RdPC en particulier pour des fins de diagnostic et de SdF de systèmes embarqués s'est heurté, tôt dans ce travail, à une carence de références. En effet, dans la littérature, les modèles RdP sont évalués soit statistiquement (simulation) soit formellement

(*model checking*). Les travaux qui abordent l'exploitation des structures des RdPC sont traités, à quelques exceptions, d'une manière intuitive. La proposition d'une approche basée sur l'inversion de modèle RdPC s'est faite, elle aussi, de cette manière. La difficulté était de fournir une base formelle qui consoliderait les algorithmes proposés. Cette démarche a nécessité une exploration des différents outils théoriques applicables aux structures RdPC et une prise en mains de ces outils pour les exploiter convenablement.

Le présent travail comporte une grande composante exploratoire. Le potentiel de suites des travaux est considérable. Les perspectives peuvent se décliner en deux axes principaux : théoriques et pratiques. Les perspectives théoriques peuvent prendre des formes très variées tels que la prise en compte des structures qui ne sont pas encore traitées ; l'incorporation d'approches comme la programmation par contraintes pour apporter une solution au problème d'inversion de fonctions difficiles ; automation possible des mécanismes complémentaires et notamment l'enrichissement de marquage ; et écriture en parallèle de ces études intuitives, les preuves adéquates. Les perspectives pratiques sont aussi variées que les théoriques. Elles peuvent se décliner sous la forme de la mise en œuvre de l'approche sous forme d'un logiciel stable ; l'utilisation des outils déjà développés pour les preuves, notamment la logique linéaire, pour les incorporer dans l'approche et l'utilisation répétée sur des cas d'étude et des projets industriels pour améliorer la prise en main et avoir un retour d'experience permettant d'adapter et d'améliorer la méthode proposée.

Bibliographie

[ABB+00] T. Amnell, G. Behrmann, J. Bengtsson, P. R. D'Argenio, A. David, A. Fehnker, T. Hune, B. Jeannet, K.G. Larsen, O. Möller, P. Pettersson, C. Weise, and W. Yi. UPPAAL - Now, Next, and Future. In *Modeling and Verification of Parallel Processes*, volume 2067 of *LNCS*, June 2000. Revised Tutorial Lectures of 4th Summer School, MOVEP 2000, Nantes, France.

[AD94] R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2) :183–235, 1994.

[AP94] C. Anglano and L. Portinale. B-W analysis : a backward reachability analysis for diagnostic problem solving suitable to parallel implementation. In Valette, R., editor, *Lecture Notes in Computer Science ; Application and Theory of Petri Nets 1994, Proceedings 15th International Conference, Zaragoza, Spain*, volume 815, pages 39–58. Springer-Verlag, 1994.

[Apo76] G.E. Apostolakis. The effect of a certain class of potential common cause failures on reliability of redundant systems. In *Nuclear Engineering and Design*, volume 36, pages 123–133, January 1976.

[Bar03] P. Barger. *Evaluation et validation de la fiabilité et de la disponibilité des systèmes d'automatisation à intelligence distribuée, en phase dynamique*. PhD thesis, Université Henri Poincaré, Nancy 1 (France), Décembre 2003.

[Bar07] M. Barr. *Embedded Systems Glossary.* Netrino Technical Library, http ://www.netrino.com/Embedded-Systems/Glossary, 04 2007.

[BBD03] S. Bernardi, A. Bobbio, and S. Donatelli. Petri nets and dependability. In Springer Berlin / Heidelberg, editor, *Lectures on Concurrency and Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 125–179, 2003.

[BBS09a] Mohamed BOUALI, Pavol BARGER, and Walter SCHÖN. Colored petri nets inversion for backward reachability analysis. In *Second IFAC Workshop on Depandable Control of Descrete Systems (DCDS'09)*, pages 259–264, Bari (Italia), June 10-12 2009.

[BBS09b] Mohamed BOUALI, Pavol BARGER, and Walter SCHÖN. Towards an efficient structural analysis of colored petri nets. In *International Conference on Industrial Engineering and Systems Management (IESM'09),*, Montreal (Canada), May 13-15 2009.

[BBS09c] Mohamed Bouali, Pavol Barger, and Walter SCHON. Inconsistent state analysis of a network receiver with colored petri nets. In *Fourth International Conference on Dependability of Computer Systems (DepCoSRELCOMEX'09)*, pages 152–159, Wroclaw (Poland), Jun 30 - Jul 02 2009. ieee Computer Society.

[Boe88] Barry W. Boehm. A spiral model of software development and enhancement. *Computer*, 21(5) :61–72, May 1988.

[BRB09] Mohamed BOUALI, Jérôme ROCHETEAU, and Pavol BARGER. Backward reachability analysis of colored petri nets. In Guedes Soares & Martorell (eds) Bris, editor, *Reliability, Risk and Safety : theory and application, proc. The European Safety and Reliability Conference (ESREL'09)*, pages 1975–1981, Prague (Czech Republic), September 7-10 2009. Taylor & Francis Group.

[BRJ00] G. Booch, J. Rumbaugh, and I. Jacobson. *Le guide de l'utilisateur UML*. Eyrolles, 2000.

[BSB09a] Pavol BARGER, Walter SCHÖN, and Mohamed BOUALI. Human behavior model for ertms verification. In *GDR E HAMASYT workshop : HumAn-Machine Systems in Transportation*, Reims, France, Sep 2 2009.

[BSB09b] Pavol BARGER, Walter SCHÖN, and Mohamed BOUALI. A study of railway ertms safety with colored petri nets. In Guedes Soares & Martorell (eds) Bris, editor, *Reliability, Risk and Safety : theory and application, proc. The European Safety and Reliability Conference (ESREL'09)*, pages 1303–1309, Prague (Czech Republic), September 7-10 2009. Taylor & Francis Group.

[CDFH93] G. Chiola, C. Dutheillet, G. Franceschinis, and S. Haddad. Stochastic well-formed colored nets and symmetric modeling applications. *IEEE Transactions on Computers*, 42(11) :1343–1360, 1993.

[CE05] F. Cottet-Emard. *Algèbre linéaire et bilinéaire : : cours et exercices corrigés*. de Boeck, Bruxelles (Belgium), 2005.

[CEI98] CEI 50 (191). Voabulaire electronique international. chapitre 191 : Sûreté de fonctionnement et qualité de service. Technical report, Commission Electronique Internationale, Apr 1998.

[CGCL04] M. Chevalier, R. Garnier, P. Chang, and B. Lusson. La sûreté de

fonctionnement (sdf). *Intersections, Le magazine Schneider Electric de l'enseignement technologique et professionnel*, NOV 2004.

[CGP99] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. The MIT Press, USA, 1999.

[CHC96] S. M. Cho, H. S. Hing, and S. D. Cha. Safety analysis using colored petri nets. In *Proc. Asia-Pacific Software Engineering Conference (APSEC-96), 4-7 December 1996, Seoul, Korea*, pages 176–183, 1996.

[CJMK01] S. Christensen, K. Jensen, T. Mailund, and L. Kristensen. State space methods for timed coloured petri nets. In *Proceedings of 2nd International Colloquium on Petri Net Technologies for Modelling Communication Based Systems, Berlin Germany*, September 2001.

[CM01] B. Compin and Y. Mortureux. Définition et mise en place d'un dispositif de diagnostic en sûreté de fonctionnement au profit des pme-pmi : Projet aqcen. Technical report, Institut de sûreté de fonctionnement, Ligeron SA, Centre technique des industries mécaniques, 2001.

[CR03] F. Cassez, , and H.O. Roux. Traduction structurelle des réseaux de petri temporels vers les automates temporisés. In *4ieme Colloque Francophone sur la Modélisation des Systèmes Réactifs (MSR'03)*, Metz, France, OCT 2003.

[DA89] R. David and H. Alla. *Du Grafecet aux réseaux de Petri*. Hermès, Paris (France), 1989.

[Dav09] P. David. *Contribution à l'analyse de sûreté de fonctionnement des systèmes complexes en phase de conception : application à l'évaluation des missions d'un réseau de capteurs de présence humaine*. PhD thesis, Université d'Orléans, France, Nov 2009.

[dSC04] P.J. Portugaland A. da Silva Carvalho. An approach based on stochastic petri nets for dependability evaluation of profibus-dp networks. In *30th Annual Conference of the IEEE Industrial Electronics Society (IECON'04)*, volume 3, pages 2371–2376, Busan, Corea Republic, Nov 2004. IEEE, Piscataway NJ, ETATS-UNIS.

[D.U97] J. D.Ullman. *Elements of ML Programming*. Prentice Hall, New Jersey (USA), 1997.

[EH82] E.A. Emerson and J.Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *14th ACM Symp Theory of Computing (STOC'82), San Francisco, CA, USA*, pages 169–180, 1982.

[EHPP05] S. Evangelista, S. Haddad, and J-F. Pradat-Peyre. Syntactical colored petri nets reductions. *Automated Technology for Verification and Analysis (ATVA 05) Taipei, Taiwan, October 4-7*, pages 202–216, 2005.

[EIA98] EIA632. Processes for engineering a system. Technical report, Electronic Industries Alliance, Apr 1998.

[FA03] T. Fruewirth and S. Abdennadher. *Essentials of Constraint Programming*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.

[Fau04] J. Faucher. *Pratique de l'AMDEC*. Fonctions de l'entreprise. Dunod, L'Usine Nouvelle, France, 1ère édition edition, Jan 2004.

[F.G97] F.Girault. *Formalisation en Logique Linéaire du fonctionnement des réseaux de Petri*. PhD thesis, Université Paul Sabatier, Toulouse, 1997.

[Fre05] G. Frehse. Phaver : Algorithmic verification of hybrid systems past hytech. In Manfred Morari and Lothar Thiele, editors, *HSCC*, volume 3414 of *Lecture Notes in Computer Science*, pages 258–273. Springer, 2005.

[FRH+75] K.N. Fleming, P.H. Raabe, G.W. Hannaman, W.J. Houghton, R.D. Pfremmer, and F.S. Dombek. Htgr accident initiation and progression analysis status report. volume ii. aipa risk assessment methodology. Technical Report GA-A13617, General Atomic Co., San Diego, Calif. (USA), October 1975.

[FTJ07] J. M. Fernandes, S. Tjell, and J. B. Jørgensen. Requirements engineering for reactive systems with coloured petri nets : the gas pump controller example. In *Proceedings of the 8th Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools (CPN 2007), October 22–24, 2007, Aarhus, Denmark*, number PB-584, pages 207–222, Denmark, 2007. Department of Computer Science, University of Aarhus.

[Gab06] H. A. Gabbar. *Modern Formal Methods and Applications*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[GBU03] F. Guerin, B.Dumon, and E. Usureau. Reliability estimation by bayesian method : definition of prior distribution using dependability study. *Reliability Engineering and System Safety*, pages 299–306, December 2003.

[G.G69] G.Gentzen. *The Collected Works of Gerhard Gentzen*. North-Holland Publishing Company, Amsterdam, 1969.

[Gir87] J.Y. Girard. Linear logic. *Theorical Computer Sciences*, 50 :1–102, 1987.

[Gob95] R. Goblot. *Algèbre Linéaire*. Masson, Paris (France), 1995.

[Had89] S. Haddad. A reduction theory for coloured nets. *Lecture notes in Computer science n° 424 Springer-Verlag*, pages 209–235, 1989.

[Han93] H.M. Hanisch. Analysis of place/transition nets with timed arcs and its application to batch process control. In Ajmone Marsan, M., editor, *Lecture Notes in Computer Science ; Application and Theory of Petri Nets 1993, Proceedings 14th International Conference, Chicago, Illinois, USA*, volume 691, pages 282–299. Springer-Verlag, 1993.

[Hav93] B.R. Haverkort. Approximate performability and dependability analysis using generalized stochastic petri nets. *Performance evaluation*, 18(1) :61–78, 1993.

[HB99] M.G. Hinchey and J.P. Bowen, editors. *Industrial-Strength Formal Methods in Practice*. Formal Approaches to Computing and Information Technology. Springer-Verlag, London, 1999.

[Hea97] S. Heath. *Embedded Systems Design*. Butterworth-Heinemann, Newton, MA, USA, 1997.

[Hei95] M. Heiner. Petri net based software dependability engineering. In *Tutorial Notes, Int. Symposium on Software Reliability Engineering (ISSRE'95)*, 1995.

[HS07] T.A. Henzinger and J. Sifakis. The discipline of embedded systems design. *Computer*, 40 :32–40, October 2007.

[HSEG02] L.S. Younes Hakan, Reid.G Simmons, E.Brinksma, and L. K. Gulstrand. Probabilistic verification of discrete event systems using acceptance sampling. *International conference on computer aided verification N°14 (CAV 2002), Copenhagen , DANEMARK*, pages 223–235, July 2002.

[HSER04] H.Demmou, S.Khalfaoui, E.Guilhem, and R.Valette. Critical scenarios derivation methodology for mechatronic systems. *Reliability Engineering and System Safety*, 84 :33–44, 2004.

[IEE99] IEEE 1220. Standard for application and management of the systems engineering process. Technical report, International Electronic and Electrical Engineers, Jan 1999.

[ISO03] ISO/IEC 15288. Systems engineering - system life-cycle processes. Technical report, American National Standards Association, Nov 2003.

[JK09] K. Jensen and L. M. Kristensen. *Coloured Petri Nets : Modelling and Validation of Concurrent Systems*. Computer Science. Springer Berlin Heidelberg, 2009.

[JKW07] K. Jensen, L. M. Kristensen, and L. Wells. Coloured petri nets and cpn tools for modelling and validation of concurrent systems. *International Journal on Software Tools for Technology Transfer (STTT)*, 9(3) :213–254, 2007.

[JLL77] N. D. Jones, L. H. Landweber, and Y. E. Lien. Complexity of some problems in petri nets. *Theor. Comput. Sci.*, 4(3) :277–299, 1977.

[JR91] K. Jensen and G. Rozenberg. *High-level Petri Nets : Theory and Application*. Springer-Verlag, Germany, 1991.

[KDCD96] W. Khansa, J.P. Denat, and S. Collart-Dutilleul. P-time petri nets for manufacturing systems. In *In International Workshop on Discrete Event Systems, WODES'96*, pages 94–102, Edinburgh (UK), 1996.

[Kha03] S. Khalfaoui. *Méthode de recherche des scénarios redoutés pour l'évaluation de la sûreté de fonctionnement des systèmes mécatroniques du monde automobile*. PhD thesis, Institut National polytechnique de Toulouse (France), Septembre 2003.

[KO01] J. Knezevic and E.R. Odoom. Reliability modelling of repairable systems using petri nets and fuzzy lambda-tau methodology. In *Reliability Engineering and System Safety*, volume 73, pages 1–17, 2001.

[Kow79] H. J. Kowalsky. *Lineare Algegra*. de Gruyter Lehrbuch, 1979. 9th Edition.

[LAB+96] J.C. Laprie, J. Arlat, J.P. Blanquart, A. Costes, Y. Crouzet, Y. Deswarte, J.C. Fabre, H. Guillermain, M. Kaaniche, K. Kanoun, C. Mazet, D. Powell, C. Rabejac, and P.Thevenod. *Guide de la sûreté de fonctionnement*. Cepaduès Editions, deuxième edition, 1996.

[Lam80] L. Lamport. 'sometimes' is sometimes 'not never' : On the temporal logic of programs. *7th Annual ACM Symp. on Principles of Programming Languages (POPL 80)*, pages 174–185, August 1980.

[LBBZ97] U. Lefarth, U. Baum, T. Beck, and T. Zurawka. Ascet-sd - development environment for embedded control systems. In *Proc. of IFAC Symposium on Computer Aided Control System Design (CACSD '97)*, Gent, Belgium, April 1997. ETAS-GmbH - Engineering Tools, Schwieberdingen, Germany.

[ldJPR00] Sous la direction J. P. Richard. *Algèbre et analyse pour l'automatique*. Hermès Science Publications, Paris (France), 2000.

[ldMD01] Sous la direction M. Diaz. *Les réseaux de Petri : modèles fondamentaux*. Hermès Science Publications, Paris (France), 2001.

[LE98] P.E. Labeau and E.Zio. The cell-to-boundary method in the frame of memorization-based monte carlo algorithms. a new computational improvement in dynamic reliability. *Mathematics and Computer in Simulation*, pages 347–360, August 1998.

[Lin92] Patrick Lincoln. Linear logic. *SIGACT News*, 23(2) :29–37, 1992.

[LL98] F. Laroussinie and K. G. Larsen. Cmc : A tool for compositional model-checking of real-time systems. In *FORTE XI / PSTV XVIII '98 : Proceedings of the FIP TC6 WG6.1 Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols (FORTE XI) and Protocol Specification, Testing and Verification (PSTV XVIII)*, pages 439–456, Deventer, The Netherlands, The Netherlands, 1998. Kluwer, B.V.

[LM04] T. Latvala and M. Mäkelä. Ltl model checking for modular petri nets. *Applications and Theory of Petri Nets, 25th International Conference, ICATPN'04, Bologna, Italy*, pages 298–311, JUNE 2004.

[LS87] N. G. Leveson and J. L. Stolzy. Safety analysis using petri nets. *IEEE Trans. Softw. Eng.*, 13(3) :386–397, 1987.

[LT07] K.B. Lassen and S. Tjell. Translating colored control flow nets into readable java via annotated java workflow nets. In *Eighth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, pages 127–146, Aarhus, Denmark, 2007.

[Mal02] A. Mallet. Démarche générale de la modélisation et de l'expérimentation in silico. In *Atelier de formation n°134 de l'INSERM (Conférence 2 sur 16)*, La Roche-Posay, France, Fev 2002.

[MBC+95] M. A. Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets.* John Wiley and Sons, 1995.

[MBT02] J.Y Morel, M. Barreau, and A. Todoskoff. Petri nets : A tool adapted to computer system dependability and safety. In *European Conference on System Dependability and Safety (ESREL 2002)*, pages 125–130, Lyon, France, 2002.

[ME00] M.Marseguerra and E.Zio. Monte carlo biasing in reliability calculations with deterministic repair times. *Annals of Nuclear Energy*, pages 639–648, May 2000.

[Med06] M. Medjoudj. *Contribution à l'analyse des systèmes pilotés par calculateurs : Extraction de scénarios redoutés et vérification de contraintes temporelles.* PhD thesis, Université Paul Sabatier de Toulouse (France), March 2006.

[MEJP98] M.Marseguerra, E.Zio, J.Devooght, and P.E.Labeau. A concept paper on dynamic reliability via monte carlo simulation. *Mathematics and Computer in Simulation*, pages 371–382, August 1998.

[Mer74] P.M. Merlin. *A Study of the Recoverability of Computing Systems.* PhD thesis, Dept. of Information and Computer Science, University of California, Irvine, California, 1974.

[Mol82] M. K. Molloy. Performance analysis using stochastic petri nets. *IEEE Transactions on Computers*, 31(9) :913–917, 1982.

[Mon98] G. Moncelet. *Application des réseaux de Petri à l'évaluation de la sûreté de fonctionnement des systèmes mécatroniques du monde automobile*. PhD thesis, Université Paul Sabatier de Toulouse (France), October 1998.

[MS07] J.R. Müller and E. Schnieder. Duality in high level petri-nets : a basis to do diagnoses. In *WSC '07 : Proceedings of the 39th conference on Winter simulation*, pages 629–636, Piscataway, NJ, USA, 2007. IEEE Press.

[MSS09] J.R. Müller, T. Ständer, and E. Schnieder. A comparison of safety analysis techniques : Analytical calculations versus monte carlo simulations. In Guedes Soares & Martorell (eds) Bris, editor, *In proc of Esrel'09, Reliability, Risk and Safety : Theory and Applications*, pages 1483–1487, Prague, czech republic, September 2009. Taylor & Francis Group, London.

[MT95] M. Malhotra and K.S. Trivedi. Dependability modeling using petri-nets. In *IEEE Transactions on Reliability*, volume 44, pages 428–440, 1995.

[MU49] N. Metropolis and S. Ulam. The monte carlo method. *Journal of American Statistical Association*, 44(247) :335–341, September 1949.

[Nas76] M.Z. Nashed. *generalized inverses and applications*. Academic Press, New York, 1976.

[Noe05] T. Noergaard. *Embedded Systems Architecture : A Comprehensive Guide for Engineers and Programmers*. Newnes, 2005.

[OTSW04] F. Ortmeier, A. Thums, G. Schellhorn, and W.Reif. Combining formal methods and safety analysis - the formosa approach. *Integration of Software Specification Techniques for Application in Engineering*, pages 474–493, May 2004.

[Pet62] Carl Adam Petri. *Communication with automata*. PhD thesis, Darmstadt Institut für Instrumentelle Mathematik, Bonn (Germany), 1962.

[PG80] A. Pagès and M. Gondran. *Fiabilité des systèmes*. Eyrolles, Paris, France, 1980.

[Por93] L. Portinale. *Petri Net Models for Diagnostic Knowledge Representation and Reasoning*. PhD thesis, University of Torino (Italy), 1993.

[Ram74] C. Ramchandani. *Analysys of Asynchronous Concurrent Systems by Timed Petri Nets*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachussets, USA, 1974.

[Rip84] J. McDermidand K. Ripken. *Life Cycle Support in the ADA Environment*. Cambridge University Press, New York, NY, USA, 1984.

[RJB04] J. Rumbaugh, I. Jacobson, and G. Booch. *UML 2 : Guide de référence*. CampusPress, 2004.

[Roy70] W. Royce. Managing the development of large software systems. In *Proc. IEEE Wescon*, pages 1–9, August 1970.

[RR06] S. Robertson and J. Robertson. *Mastering the Requirements Process (2nd Edition)*. Addison-Wesley Professional, 2006.

[Rug07] A.E. Rugina. *Modélisation et évaluation de la sûreté de fonctionnement - De AADL vers les réseaux de Petri stochastiques*. PhD thesis, Instutut National Polytechnique de Toulouse, Toulouse, France, November 2007.

[Sad07] N. Sadou. *Aide à la conception des systèmes embarqués sûr de fonctionnement*. PhD thesis, Université Paul Sabatier de Toulouse (France), November 2007.

[Sif09] J. Sifakis. Les systèmes embarqués - nouveaux défis scientifiques pour l'informatique. les défis scientifiques du 21e siècle, Grande salle des séances, Institut de France, 23 quai Conti, 75006 Paris, Mar 2009.

[SK07] T. Sivanthi and U. Killat. Reliable scheduling of a distributed real-time embedded application considering common cause failures. In *IEEE Conference on Emerging Technologies and Factory Automation*, Patras, Greece, sep 2007.

[SS97] Ian Sommerville and Pete Sawyer. *Requirements Engineering : A Good Practice Guide*. John Wiley & Sons, New York, NY, USA, April 1997.

[SS03] H.S. Son and P. H. Seong. Development of a safety critical software requirements verification method with combined CPN and PVS : a nuclear power plant protection system application. *Reliability Engineering & System Safety, Volume 80, Issue 1 , April 2003*, pages 19–32, April 2003.

[Sta88] J.A. Stankovic. Misconceptions about real-time computing : A problem for next generation systems. *IEEE Computer*, 21(10) :10–19, October 1988.

[Sym78] F. J. W. Symons. *Modeling and analysis of communication protocols using numerical Petri nets*. PhD thesis, University of Essex, Great Britain, 1978.

[Val92] F. Vallee. Statemate : une méthode et un outil pour la spécification des systèmes réactifs. In FRANCE Union des syndicats de l'électricité, Paris, editor, *Revue générale de l'électricité*, pages 46–51, 1992.

[vdA93] W. M. P. van der Aalst. Interval timed coloured petri nets and their analysis. In *Proceedings of the 14th International Conference on Application and Theory of Petri Nets*, pages 453–472, London, UK, 1993. Springer-Verlag.

[Ver01] Y. Verot. Retour d'expérience dans les industries de procédé. In *Techniques de l'ingénieur. Sécurité et gestion des risques*, volume SE1, pages AG4610.1–AG4610.4, Paris, FRANCE, 2001. ATOFINA, Techniques de l'ingénieur,.

[Vil88] A. Villemeur. *Sûreté de fonctionnement des systèmes industriels*. Direction des études et recherches d'Electricité de France (EDF). Eyrolles, EDF, 1988.

[VK94] R. Valette and L.A. Künzle. Réseaux de petri pour la détection et le diagnostic. *G.R.Automatique, Journées Nationales Sûreté, surveillance, supervision*, 1994.

[ZBH96] Armin Zimmermann, Stefan Bode, and Günter Hommel. Performance and dependability evaluation of manufacturing systems using petri nets. In *in : 1st Workshop on Manufacturing Systems and Petri Nets, 17th Int. Conf. on Application and Theory of Petri Nets ,Osaka (Japan)*, pages 235–250, 1996.

[Zur06] R. Zurawski. *Embedded Systems Handbook*. CRC Press, Inc., Boca Raton, FL, USA, 2006.

Résumé

Le développement rapide des systèmes embarqués et les exigences croissantes auxquelles ils sont soumis créent un besoin de techniques innovantes en terme de conception, de vérification et de validation. Les méthodes formelles fournissent des approches intéressantes à la conception de ces systèmes, notamment pour des études de sûreté de fonctionnement (SdF).

Le formalisme choisi est basé sur les Réseaux de Petri Colorés (RdPC). L'avantage de ces modèles, en plus d'être très expressifs et formels, est qu'ils permettent d'exprimer le double caractère des systèmes étudiés : statique et dynamique. Le défi relevé par cette thèse est d'utiliser des modèles établis, décrivant l'architecture et/ou le comportement de systèmes, pour en extraire des informations de SdF en général et de diagnostic de défaillances en particulier.

L'approche proposée est une analyse structurelle par accessibilité arrière de RdPC. Elle peut être décomposée en deux parties. La première consiste en la proposition d'un outil pour réaliser cette analyse : le RdPC inverse. Il est obtenu grâce à l'application de transformations structurelles sur le RdPC original. La seconde partie est la mise en œuvre de l'analyse. Cette partie requiert des mécanismes complémentaires dont le plus important est l'enrichissement du marquage.

L'approche proposée est étudiée de deux points de vue complémentaires : algorithmique et théorique. Le point de vue algorithmique consiste à proposer des modèles de transformations pour l'inversion des RdPC et la mise en œuvre de l'analyse. L'aspect théorique vise à offrir une base formelle à l'approche en appliquant deux méthodes (l'algèbre linéaire et la logique linéaire) pour prouver notre approche.

Mots-clés: Réseaux de Petri Colorés (RdPC), Accessibilité arrière, Analyse structurelle, Sûreté de fonctionnement, Systèmes embarqués, diagnostic.

Abstract

Embedded system development creates a need of new design, verification and validation technics. Formal methods appear as a very interesting approach for embedded systems analysis, especially for dependability studies.

The chosen formalism for this study is based on Colored Petri Net (CPN). Among the CPN models interests : expressivity, formal nature. Also, they express easily the double nature of the studied systems : static and dynamic. The main challenge of this thesis is to use existing models, which describe the system structure and/or behavior, to extract dependability information in a most general use and failure diagnosis information in a special use.

The proposed approach is a CPN structural backward reachability analysis. It can be split in two parts. The first is the tool to perform the proposed analysis. This tool is the inverse CPN. It is obtained thanks to structural transformation applied on an original CPN. The second part is the analysis implementation. This part needs some complementary concepts whose the most important is the marking enhancement.

The proposed approach is studied under two complementary aspects : algorithmic aspect and theoretic aspect. The first one proposes transformations for the CPN inversion and the analysis implementation. The second aspect (the theoretical one) aims to offer a formal proof for the approach by applying different methods : linear algebra and linear logic.

Keywords: Colored Petri Nets (CPN), Backward Reachability, Structural Analysis, Dependability, Embedded Systems, diagnosis.