# Génération des tests et placement de capteurs pour le diagnostic des systèmes physiques s'appuyant sur une modélisation structurelle

Abed Alrahim Yassine

**Université de Joseph Fourier - Grenoble I**

No. attribué par la bibliothèque

**THESE**

*pour obtenir le grade de*

**DOCTEUR DE L'UNIVERSITÉ JOSEPH FOURIER**

***Spécialité :*** **AUTOMATIQUE-PRODUCTIQUE**

préparée au Laboratoire G-SCOP de Grenoble

dans le cadre de l'École Doctorale :
**Électronique, Électrotechnique, Automatique, Traitement du Signal**

*présentée et soutenue publiquement*

*par*

**Abed Alrahim Yassine**

*le* 8 décembre 2008

**Titre :**

# Génération des tests et placement de capteurs pour le diagnostic des systèmes physiques s'appuyant sur une modélisation structurelle

***Directeurs de thése:***

Jean Marie Flaus, Stephane Ploix(UJF, INPG)

**JURY :**

| | | |
|---|---|---|
| Mm. Marie-Odile Cordier | Rapporteur | Directeur de Recherche, IRISA, Rennes |
| M. Jean-Philippe Cassar | Rapporteur | Professeur au laboratoire LGBA, Lille |
| Mm. Louise Trave-Massuyès | Examinateur | Directeur de Recherche LAAS,Toulouse |
| M. Didier Georges | Examinateur | Professeur au laboratoire GIPSA, Grenoble |
| M. Franck Barruel | Examinateur | Docteur ingénieur au CEA |

# Table des figures

# Table des matières

# Remerciements

Les travaux présentés dans ce mémoire ont été effectués au Laboratoire des Sciences pour la conception, l'optimisation et la production de Grenoble " G-SCOP ".

Nombreux sont ceux que je voudrais remercier pour m'avoir aidé, soutenu ou accompagné durant cette thèse.

Je tiens tout d'abord à remercier mon Directeur de thèse, Monsieur Jean-Marie FALUS, professeur à l'université Joseph Fourier à Grenoble, de m'avoir fait partager son expérience, de la liberté d'action et d'autonomie dont j'ai bénéficié durant cette thèse.

Je tiens également à adresser mes plus vifs remerciements à mon co-directeur de thèse, Monsieur Stéphane PLOIX, Maître de conférences à l'INPG pour avoir su me guider avec attention et gentillesse pendant la durée de ma thèse. Ses qualités scientifiques et humaines, son encouragement et ses remarques m'ont largement aidé à aboutir cette thèse.

Je tiens aussi à adresser mes remerciements à Madame Louise TRAVE-MASSUYES (Directrice de Recherche LAAS, Toulouse) qui m'a fait le bonheur d'accepter à la fois de présider le jury et d'être examinatrice de ma thèse.

J'exprime mes sincères remerciements à Madame Marie-Odile CORDIER (Directrice de Recherche, IRISA, Rennes), ainsi qu'à Monsieur Jean Philippe CASSAR (Professeur au laboratoire LGBA, Lille), pour l'intérêt qu'ils ont bien voulu me porter en acceptant à la fois de participer à mon jury de thèse et d'être rapporteurs de ce travail. Je vous remercie pour vos conseils et vos suggestions qui ont permis l'amélioration de ce manuscrit. Veuillez accepter mes plus sincères remerciements pour votre présence dans ce jury.

J'exprime également mes sincères remerciements à Monsieur Didier GEORGES (Professeur au laboratoire GIPSA, Grenoble), ainsi qu'à Monsieur Franck BARRUEL (Docteur ingénieur au CEA) d'avoir accepté d'être examinateurs de ce travail.

Je tiens à remercier M. Alain BARAUD (ancien Directeur du Laboratoire Automatique de Grenoble) qui m'a accueilli au sein de son laboratoire comme étudiant en Master de Recherche puis en tant que Doctorant durant les deux premières années de ma thèse.

Je n'oublierai pas l'ensemble de membres de l'équipe administrative et technique du Laboratoire GIPSA-LAB et en particulier Daniel, Marie-Thérèse, Virginie et Patricia. Je les remercie pour leur gentillesse, leur bonne humeur et leur efficacité.

Mes remerciements s'adressent aussi à M. Yannick FREIN (le Directeur du laboratoire G-SCOP) qui m'a accueilli au sein de son laboratoire depuis janvier 2007.
Je n'oublierai pas l'ensemble de membres de l'équipe administrative et technique du laboratoire G-SCOP et en particulier Fadila MESSAOUD-DJEBARA, Chantal PUECH.

Une mention toute particulière pour mes collègues de laboratoire. Merci à tous les membres du laboratoire G-SCOP, à leur enthousiasme, parfois débordant qui contribuent à son ambiance de travail agréable et propice à la recherche.

J'adresse mes remerciements à l'université TICHRINE en Syrie, qui a financé mes études en France. Je remercie également tous mes professeurs à l'université TICHRINE.

La distance qui me sépare de ma famille a toujours été une des difficultés que j'ai eu à surmonter durant mon séjour en France. Je souhaite exprimer toute ma reconnaissance à ma mère, mon père et tous mes frères et soeurs (Fadwa, Ramez, Warda, Ibtessam, Linda, Amjad, Shirine) pour leur soutien moral.

Un remerciement particulier à ma fleur Johanna YASSINE de m'avoir supporté pendant la période de préparation de soutenance. De même, je tiens à remercier la famille de Johanna d'avoir assisté à ma soutenance.

Merci à tous pour votre soutien et votre compréhension sans limites.

# Introduction générale

Le monde de l'industrie dispose de machines et d'installations de plus en plus performantes et complexes. Les exigences de sécurité, la réduction des coûts d'exploitation et la maîtrise de la disponibilité des équipements donnent au diagnostic et à la maintenance des systèmes un rôle prépondérant. Ils doivent permettre de n'intervenir qu'en présence d'éléments défectueux, de minimiser le temps de réparation, et de fournir un diagnostic fiable et facilement interprétable malgré la complexité des équipements.

La complexité des systèmes actuels complexifie la tâche de diagnostic de défauts des installation industrielles. Les procédures de diagnostic de défauts dans les systèmes physiques s'avèrent devenir très complexes dès que les systèmes considérés ne sont plus simples. Il est alors légitime pour les entreprises de chercher à mettre en place des systèmes de diagnostic et de maintenance afin d'améliorer la sécurité des personnels et d'assurer la fiabilité et la disponibilité des installations.

La tâche de diagnostic d'un système est constituée de plusieurs étapes dont la principale consiste à surveiller le système pour détecter tout anomalie. Une fois, l'anomalie détectée, l'étape suivante est d'expliquer les défaillance en fournissant des diagnostics (des listes de composants pouvant être l'origine de défaut).
L'efficacité du système de diagnostic dépend de la pertinence d'information accueillies sur le système à diagnostiquer en utilisant des capteurs. Si cette information n'est pas suffisante, le système de diagnostic ne pourra pas réaliser la tâche pour laquelle il est conçu. L'efficacité du système de capteurs est mesurée par le niveau du "diagnosticabilité" qu'il fournit.

Pour faire face au problème de diagnostic, de nombreuses méthodes et outils (la conception des tests de détection et la conception d'un système de placement de capteurs) ont été mis en place, mais ces méthodes présentent toutes des limites, soient en termes de complexité de calcul, soient en termes de classes des systèmes adressables.

Le travail que nous présentons dans ce document propose d'apporter sa contribution au domaine du diagnostic de défauts. Dans ce travail, nous présentons une approche structurelle pour le conception des sous-systèmes testables permettant de détecter les défauts dans les systèmes à diagnostiquer. Cette approche vise d'une part à étendre les

possibilités des approches existantes et d'autre part à diminuer l'ordre de complexité des méthodes existantes. Nous présentons également une approche structurelle pour le placement de capteurs en prenant en compte les spécifications de diagnosticabilité. Cette approche permet de trouver le meilleur placement de capteurs sans nécessiter la conception préalable des relations de redondance analytique.

Ce mémoire est organisé en sept chapitres dont les thèmes sont donnés ci-après.

Le premier chapitre rappelle les différentes approches du diagnostic de défauts développées par différentes communautés de recherche. Ces approches peuvent être classées, de manière générale, en des méthodes à base de modèles analytiques, à base de reconnaissance de formes, à base de réseaux bayésiens, à base de cas, et des méthodes à base d'arbre de décision. Nous nous concentrons particulièrement sur le diagnostic à base de modèles analytiques représentant l'outil principal de diagnostic dans les communautés de diagnostic **FDI** et **DX** et **bridge**.

Dans le deuxième chapitre, nous présentons quelques représentations structurelles permettant d'analyser les systèmes physiques (matrice structurelle, graphe biparti, digraphe, bond graph). Nous présentons également quelques outils importants utilisés dans la représentation par graphe pour le diagnostic de défauts (couplage, décomposition de Dulmage-Mendelsohn). Ces outils seront utilisés pour analyser les propriété des systèmes qui pourront être utilisées pour la conception des relations de redondance analytique ou le placement de capteurs. À la fin de ce chapitre, nous présentons une représentation structurelle générale permettant d'appréhender la diversité des concepts utiles au diagnostic. Cette représentation sera la base de notre travail.

Le troisième chapitre rappelle des méthodes existantes pour la conception de relations de redondance analytique (**RRAs**) pour le diagnostic de défauts. Nous nous concentrons particulièrement sur les méthodes basées sur la recherche de couplages dans un graphe biparti (Blanke et al. [2003], Blanke et al. [2006]), la propagation de valeurs, le retrait consécutif de capteurs (Travé-Massuyès et al. [2006]) et la décomposition Dulmage-Mendelshon (Krysander et al. [2008]). Nous présentons aussi une méthode de conception de conflits (Pulido and Alonso [2002], Pulido and Alonso [2004]) issue de la communauté de l'intelligence artificielle. Un comparatif entre ces méthodes est représenté à la fin de ce chapitre pour montrer les avantages et les inconvénients de ces méthodes.

Dans le quatrième chapitre, une méthode structurelle pour la conception des sous-systèmes testables est formalisée. La méthode présentée réduira les limites des méthodes déjà proposées dans le chapitre précédent soient en terme de complexité de calcul, soient en terme de classes de systèmes adressables. En utilisant cette méthode, il est possible d'appréhender de nombreux types de système : des systèmes continus ou échantillonés, linéaires ou non linéaires, statiques ou dynamiques, à base de règle ou

non. Un exemple de réseau routier montre qu'il est possible de calculer des relations de redondance analytique (**RRAs**) même si les contraintes comportementales ne sont pas disponibles. Il est aussi possible de tenir compte d'exclusions afin d'éviter les sous-systèmes testables irréalistes.

Le cinquième chapitre rappelle des méthodes existantes pour le placement de capteurs. Plusieurs méthodes du placement de capteurs sont présentées et nous nous concentrons spécialement sur des méthodes satisfaisant des critères de diagnosticabilité. Dans un premier temps, nous présentons les méthodes présentées dans (Travé-Massuyès et al. [2006], Travé-Massuyès et al. [2001]). Ces méthodes sont basées sur l'addition ou le retrait consécutif des capteurs. Ensuite, nous présentons la méthode présentée dans (Frisk and Krysander [2007]). Cette méthode est basée sur la représentation structurelle du système (décomposition Dulmage-Mendelshon (Dulmage and Mendelsohn [1959], Pothen and Chin-Ju [1990]). À la fin de ce chapitre, nous montrons les limites de ces méthodes.

Dans le sixième chapitre, nous propose une approche pour le placement de capteurs prenant en compte des spécifications de diagnosticabilité et détectabilité. Deux types de spécifications sont considérés : spécifications complètes, où tous les ensembles de contraintes diagnosticables, discriminables et non détectables sont spécifiés, et les spécifications simplifiées, où seulement l'ensemble de contraintes qui doivent être diagnosticables et l'ensemble de contraintes qui doivent être au moins détectables sont spécifiés. Ces méthodes s'appliquent aux systèmes pour lesquels seule la structure est connue. Elles s'appliquent aussi à n'importe quel système (sur-déterminé, juste-déterminé ou sous-déterminé). Grâce à ces méthodes, le placement de capteurs optimal par rapport au coût satisfaisant des objectifs de diagnosticabilité peut être déduit sans nécessiter la conception préalable de relations de redondance analytique. Ces méthodes ne prennent pas en compte la notion de "déductibilité" (toutes les variables du système sont déductibles). À la fin de ce chapitre, nous présentons une méthode alternative pour le placement de capteurs prenant en compte la notion de déductibilté des variables du système. L'inconvénient de cette méthodes est qu'elle nécessite la conception préalable des relations de redondance analytique.

Dans la septième chapitre, nous présentons un outil informatique pour l'aide au diagnostic appelé **DXLAB**. Afin de concevoir les sous-systèmes testable et valider les résultats de placement de capteurs que nous avons développés, nous illustrons cet outil sur deux systèmes : un système électrique ("un amplificateur') et un système logique. Pour finir, nous allons valider notre approche de placement de capteurs sur une véritable application industrielle complexe. Cette application industrielle (installation photovoltaïque) va être utilisée pour illustrer l'intérêt de notre méthode de placement de capteurs. Afin de trouver le meilleur placement de capteur satisfaisant des spécifications de diagnosticabilité pour cette application, nous allons appliquer une approche hiérarchisée permettant

d'appréhender des systèmes complexes contenant un grand nombre de composants et faisant intervenir un grand nombre de variables. Cette hiérarchisation est juste une astuce qui correspond à une réalité pratique pour réduire la complexité.

# Première partie

# Etat de l'art

# Chapitre 1

# Introduction au diagnostic de défauts

Les systèmes conçus et fabriqués par l'homme (véhicules, avions, réseaux de télécommunications, usines...) sont de plus en plus complexes. Cette complexité est due au grand nombre de composants constituant ces systèmes. Malgré les besoins de haute sécurité, la réduction des coûts d'exploitation et la maîtrise de la disponibilité des équipements, ces systèmes ne sont pas à l'abri de défaillances. C'est pourquoi les activités de surveillance, diagnostic (détection, localisation, identification de défaillances), réparation ou reconfiguration sont très importantes. Ces activités permettent de détecter et localiser les défauts, de minimiser le temps de réparation, et de fournir un diagnostic fiable et facilement interprétable malgré la complexité des équipements.
De nos jours, l'implémentation des systèmes automatisés suppose la mise en place d'outils importants pour le diagnostic et la surveillance pour aider les entreprises dans leur recherche permanente d'un meilleur fonctionnement de leurs systèmes à moindre coût.

Dans ce contexte, de nombreuses approches ont été développées, en vue de la détection de défauts et du diagnostic, par différentes communautés de recherche en automatique. Ces approches peuvent être classées, de manière générale, comme des méthodes à base de modèles, à base de reconnaissance de formes, à base de réseaux bayésiens, à base de cas, et des méthodes à base d'arbre de décision. Les méthodes à base de modèles considèrent un modèle de comportement du système basé sur des principes physiques fondamentaux. Ces modèles peuvent être de type quantitatif exprimés sous forme d'équations mathématiques (contraintes) ou bien de type qualitatifs, exprimés par exemple sous forme de relations logiques. Les méthodes à base de reconnaissance de formes vise à identifier les zones d'un espace de valeurs qui correspondent à des états défaillants. Les méthodes à base de réseaux bayésiens sont des méthodes probabilistes qui peuvent aider à faire le diagnostic. Les méthodes à base d'arbre de décision consistent à construire un arbre à questions successives. Selon la réponse, cet arbre peut être construit et il permet de

réaliser le diagnostic. Les méthodes à base de cas consistent à enregistrer dans une base de connaissances les effets observés des défauts qui se sont produit dans le passé. Puis, lorsque un fait anormal se produit, on cherche des cas similaires dans la base de connaissances pour trouver les diagnostics possibles.

## 1.1 Étapes de conception d'un système de diagnostic

Pour introduire notre vision du diagnostic de défauts, il est nécessaire d'introduire quelques définitions :

- Défaut de composant : un comportement non conforme au comportement normal d'un composant.
- Défaillance : un défaut qui implique l'altération d'une ou de plusieurs fonctions du système dans des conditions d'opération spécifiées.
- Perturbation : une entrée inconnue et non contrôlée agissant sur un système.
- Symptôme : signe comportant une information révélatrice de l'état réel du système.
- Détection de défauts : action de générer des symptômes.
- Localisation de défauts : action de déterminer l'état d'un système à partir des symptômes. Elle se nomme aussi analyse diagnostique.
- Diagnostic de défauts : action générique regroupant la détection et la localisation de défauts.
- Résidu : indicateur de la présence d'un défaut basé sur la déviation entre les mesures et le calcul basé sur un modèle. Un résidu est un signal porteur de symptômes.

Pour bien comprendre les étapes nécessaires pour concevoir un système de diagnostic de défauts, le schéma 1.1 représente la modularisation de la résolution. Ce schéma montre que la conception se base sur des étapes différentes :

- la première étape de conception d'un système de diagnostic est le placement de capteurs. Cette étape permet de trouver le placement optimal de capteurs satisfaisant des objectifs prévus (observabilité, contrôlabilité, monitorabilité, diagnosticabilité...).
- la deuxième étape est la modélisation du système à diagnostiquer avec les capteurs trouvés en première étape. Cette étape consiste à modéliser le comportement d'un système avec ses capteurs par des relations analytiques (contraintes).
- La troisième étape est la conception de sous systèmes testables (**SST**s) sachant qu'un SST est un ensemble de contraintes qui conduit à un test. Cette étape permet de chercher les sous ensembles de contraintes nécessaires pour construire les relations de redondance analytique **RRA**s.
- la quatrième étape est la conception de relations de redondance analytique (**RRA**s) sachant qu'une RRA est une contrainte qui ne contient plus de variables inconnues. Cette étape consiste à construire les relations de redondance analytique nécessaires

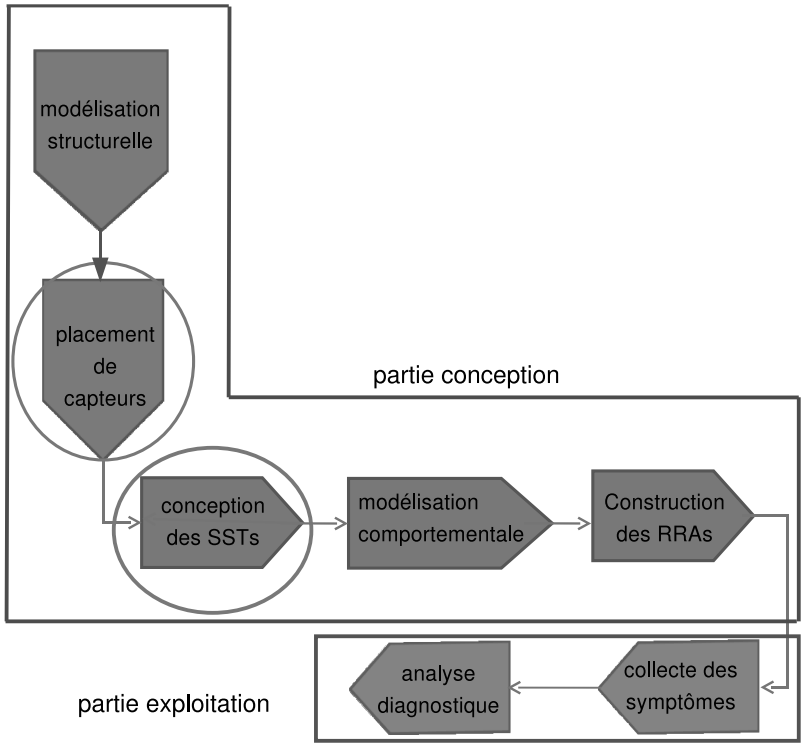


FIG. 1.1 – Système de diagnostic de défauts

pour trouver les symptômes.
- la cinquième étape est la collecte des symptômes. Cette étape consiste à utiliser les **RRA**s pour générer des symptômes.
- la dernière étape est l'analyse diagnostique. Cette étape permet de trouver les diagnostics exprimés en terme de modes de comportement de composants.

Dans le cadre de notre thèse, nous nous concentrons d'une part sur l'étape du placement de capteurs et d'autre part sur les méthodes de conception de **SST**s. Nous allons proposer une méthode de placement de capteurs satisfaisant des spécifications de diagnosticabilité et une méthode pour la conception des **SST**s diminuant la complexité des méthodes existantes.

## 1.2 Rappel sur le diagnostic de défauts dans les système industriels

Diagnostiquer un système afin de trouver les composants défectueux est une tâche très importante qui a inspiré beaucoup de chercheurs afin d'améliorer les méthodes déjà existantes. Le but principal du diagnostic de défauts est de fournir au monde industriel la possibilité de détecter précocément des défauts. Par exemple, la détection d'un défaut dans un procédé au moment de son apparition permet de le réparer rapidement avant que son comportement devienne critique, en évitant l'arrêt du procédé entier qui peut augmenter le prix de maintenance, et la réparation qui peut coûter très cher.
La sûreté, disponibilité et fiabilité des systèmes sont des objectifs importants dans le monde industriel. Ces caractéristiques peuvent être augmentées considèrabelement par le diagnostic précoce des changements de performance des composants. Dans ce but, concevoir un système efficace de diagnostic est devenu une tâche inévitable. De nombreuses méthodes existantes de diagnostic et s'appuient sur différents principes d'analyse.

### 1.2.1 Méthode de diagnostic à base de modèles

Parmi les différentes méthodes proposées pour le diagnostic, celle nommée diagnostic à base de modèles, présente l'avantage de ne nécessiter aucune connaissance préalable des défauts ou symptômes possibles. Elle s'appuie uniquement sur la vérification de la consistance entre le comportement réellement observé du système et le comportement attendu de ce système. Selon la connaissance du processus, il est possible de définir trois formulations différentes de cette approche à base de modèles : l'approche **FDI** (Fault Détection and Isolation), issue de la communauté de l'Automatique, s'appuie sur des modèles quantitatifs, l'approche **DX** de la communauté de l'Intelligence Artificielle, s'appuie sur les modèles qualitatifs et l'approche **bridge** entre **FDI** et **DX**. Les méthodes de diagnostic à base de modèles sont utilisées notamment dans (Willsky [1975], Frank [1996], Pulido and Alonso [2000], Travé-Massuyès et al. [2001], Chittaro and Ranon [2004], Isermann [2004], Dimitrova et al. [2007], Asokan and Sivakumar [2007]).

#### A Approche FDI

La communauté **FDI** (Fault Detection and Isolation) (Patton and Chen [1991], Gertler [1993], Cassar et al. [1995], Frank [1996], Isermann [1997], Isermann [2004]) utilise des techniques provenant de la théorie de la commande et de la décision statique. Cette approche se consacre principalement à la partie détection qui consiste à générer des symptômes les plus révélateurs possible de l'état courant du système. Cette approche se base sur les modèles quantitatifs. Ces modèles sont construits à partir des lois fonda-

mentales (physique, chimie,..) et sont décrits par des équations mathématiques qui sont la plupart du temps des équations différentielles.
Les méthodes de la communauté **FDI** sont nommées "méthode des résidus". Ces méthodes comportent deux étapes :

- génération des résidus (Frisk [2000], Staroswiecki and Comtet-varga [2001], Dustegor et al. [2004], Nyberg and Frisk [2006])
- le choix d'une règle de décision pour l'analyse diagnostique

Les résidus représentent des changements entre le comportement réel du système et celui attendu par le modèle. Ces résidus doivent avoir une valeur nulle en fonctionnement normal (pas de défauts). Au contraire, en présence de défauts, la valeur de ces résidus sera non nulle. Le principe le plus général pour produire les résidus est illustré dans la figure 1.2.

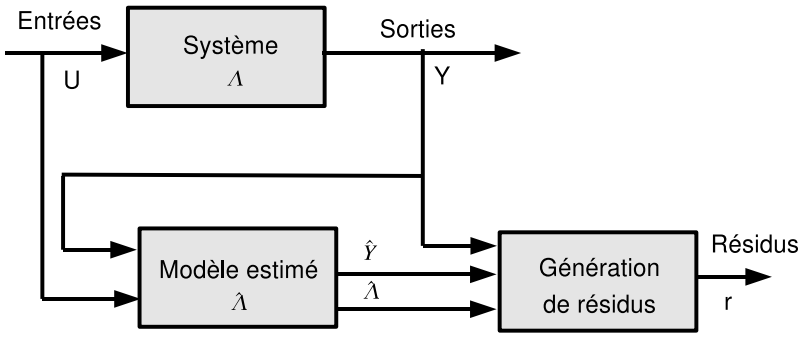


FIG. 1.2 – Génération des résidus

Ces résidus doivent être sensibles aux défauts et insensibles aux perturbations (bruit de mesure et bruit de structure qui sont conçus comme normaux). L'approche **FDI** vise autant que possible à faire qu'un résidu soit sensible à un seul défaut. Lorsque cela est possible, l'analyse diagnostique devient triviale mais n'offre malheureusement aucune garantie de résultat c'est-à-dire qu'il n'est pas possible de prouver que le diagnostic fourni est juste même si la modélisation l'est : cela est dû à l'hypothèse d'exonération qui consiste à considérer qu'en l'absence d'anomalie, tous les composants couverts par un test fonctionnent correctement. Dans la pratique, cette hypothèse est rarement vérifiée car un modèle de défaut ne se révèle pas en permanence. Des techniques pour la génération des résidus à partir de modèles analytiques sont proposées ci-dessous :

- Estimation d'état à partir d'un filtre de kalman (Frank and Wunnenberg [1989], Caliskan and Hajiyev [1998])
- Relation de parité (Gertler [1987], Ploix and Adrot [2006])
- Estimation paramétrique (Isermann [1993])

Les résidus générés doivent être évalués pour déterminer la présence des défauts. Cette évaluation est faite par l'utilisation de seuils fixes pour éviter les fausses alarmes.

Pour analyser la diagnosticabilité des systèmes, l'approche **FDI** utilise un outil central nommé matrice de signatures des défauts "$MSF$" (Patton and Chen [1991], Cassar and Staroswiecki [1997]) qui est obtenue à partir de l'ensemble des résidus représentant les lignes de la matrice et l'ensemble des défauts représentant les colonnes de la matrice. Un élément $MSF_{ij}$ a une valeur "1" si le défaut de la colonne "j" influence le résidu de la ligne "i", sinon la valeur sera "0". Nous pouvons localiser les défauts en faisant la comparaison d'une signature observée avec les différentes colonnes de la matrice de signature de défauts.

La limite de l'approche **FDI** est la nécessité des modèles mathématiques précis et complets, ce qui n'est pas facile avec les systèmes complexes tels que les processus chimiques. Un autre inconvénient de cette approche est la modélisation des perturbations qui peuvent causer des problèmes dans le modèle. En plus, l'approche **FDI** est liée à l'hypothèse implicite d'exonération : elle considère que si un résidu est quasiment nul, cela signifie qu'il n'y a pas de défaut. Cette approche appréhende difficilement les défauts multiples car pour ce faire, il faut envisager toutes les combinaisons possibles de défauts ce qui conduit à une complexité très grande.

## B Approche DX

Dans certains cas, à cause de la complexité des systèmes, il n'est pas facile de disposer des connaissances complètes sur le comportement des systèmes afin de faire un modèle analytique décrit par des équations mathématiques. Une approche pour traiter les connaissances incomplètes sur les systèmes à diagnostiquer est nommée approche **DX**. L'approche **DX**, issue de la communauté de l'Intelligence Artificielle exprime explicitement le lien entre un composant et les formules décrivant son comportement. Les travaux fondateurs de l'approche **DX** sont ceux de (Reiter [1987], De Kleer and Williams [1987]). Reiter [1987] propose une théorie logique du diagnostic intitulée diagnostic issu des principes premiers (diagnosis from the first principles). Le but de cette théorie est de déterminer les composants défectueux d'un système physique. Cette approche est basée sur l'analyse d'inconsistance entre les observations et le comportement attendu (figure 1.3). Cette approche a été formalisée dans (De Kleer et al. [1992]). Le cadre théorique de cette approche est bien développé dans de nombreux travaux (Travé-Massuyès et al. [1997], Dague [2001], De Kleer [2003],Pulido and Alonso [2004]).

L'approche **DX** est basée sur un raisonnement à base logique et ne repose sur aucune hypothèse implicite telle l'hypothèse d'exonération. Elle offre par conséquent une garantie de résultat : si les modèles de comportement sont justes, alors nécessairement l'état réel du système fait partie de la liste des diagnostics calculés. Cette approche est adaptée

aux systèmes qui peuvent être modélisés sous formules logiques et elle s'adapte mal aux systèmes dynamiques.
Pour analyser la diagnosticabilité des systèmes, l'approche **DX** utilise le concept de conflit. Un conflit est un ensemble de composants dont les hypothèses de comportement correct ne sont pas consistantes avec les observations réelles. Si un défaut se produit dans le système, alors l'utilisation de la technique de raisonnement à base de consistance nous permet de localiser les composants défectueux.

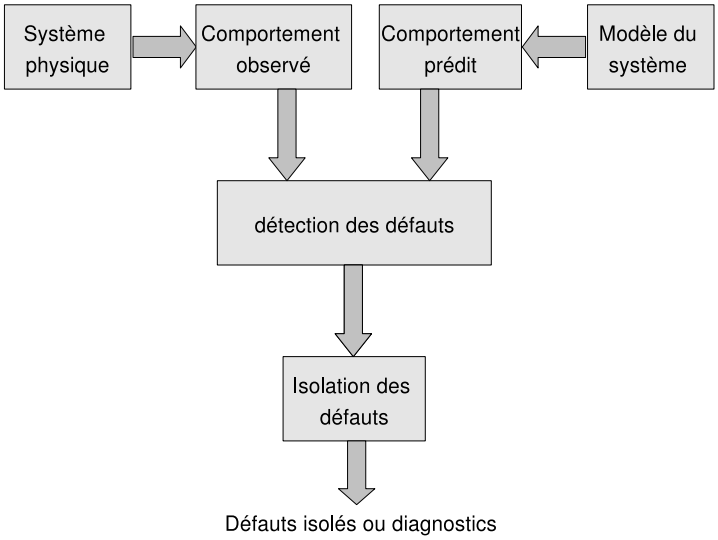


FIG. 1.3 – Diagnostics à base de consistance

## C Approche bridge entre FDI et DX

Une nouvelle approche de diagnostic à base de modèles est conçue en utilisant les idées des communautés **FDI** et **DX** (Ploix et al. [2003], Nyberg and Krysander [2003], Cordier et al. [2004]). Des études comparatives entre l'approche **FDI** et l'approche **DX** ont été réalisées par le groupe de travail français IMALAIA (Cordier et al. [2000b], Cordier et al. [2004]). Dans cette approche, le mécanisme de localisation de défauts est basé sur l'approche **DX**, et l'avantage principal est que les défauts multiples peuvent être implicitement manipulés.

L'approche **DX** propose un raisonnement logiquement rigoureux basé sur les modèles représentant le comportement normal des composants du système. Or, cette approche ne distingue pas la génération de symptômes de raisonnement du diagnostic. Tandis que l'approche **FDI** utilise principalement la génération de symptômes basée sur les modèles représentant le comportement anormal. L'approche **bridge** prend les avantages des deux

approches **FDI** et **DX**. Elle distingue la phase de génération des symptômes, qui peut se faire à base de techniques variées et parfois très sophistiquées de la phase de raisonnement diagnostique, qui doit permettre de garantir ce qui peut l'être et d'analyser toutes les informations disponibles pour en déduire le diagnostic le plus juste et complet possible.

Dans cette approche, nous supposons que le système est composé d'un ensemble de composants $C$. Le comportement de chaque composant $c \in C$ est modélisé par un ensemble de relations. Ce comportement peut varier selon le mode comportemental du composant. Il existe différents modes de comportement $(ok(c), \neg ok(c), defaut_1(c), defaut_2(c), ...)$ tels que $ok(c)$ signifie que le composant $c$ se comporte normalement et $\neg ok(c)$ signifie qu'il y a une anomalie sur le comportement du composant $c$ .

L'outil principal de diagnostic dans l'approche **bridge** est la conception des **RRAs**. Une relation de redondance analytique **RRA** est une contrainte déduite du modèle du système, qui ne contient que des variables connues et qui peut être évaluée à chaque observation. Les composants (où les contraintes modélisant les composants) utilisés pour obtenir les **RRAs** sont nommés les supports de **RRAs** ($supp - \mathbf{RRA}s$).

La conception du système de diagnostic peut être décomposée en trois étapes principales (voir figure 1.4).

- extraire les symptômes des informations disponibles sur l'état actuel du système en étudiant la consistance entre les relations de redondance analytique obtenues et les modèles du comportement du système à diagnostiquer.
- déterminer les composants défectueux dans le système
- appliquer la stratégie de diagnostic pour satisfaire des conditions telles que la fiabilité de diagnostic, temps requis par les algorithmes de diagnostic...

Dans l'approche **bridge**, la matrice de signature des défauts (Cordier et al. [2000a]) est un concept central pour analyser la diagnosticabilité des systèmes. Dans cette matrice, les relations de redondance analytique **RRAs** correspondent aux lignes et les défauts aux colonnes. Supposons que $\mathcal{F}_j$ représente un défaut sur le composant $c_j$. La signature de $\mathcal{F}_j$ est donnée par un vecteur binaire $S\mathcal{F}_j = [\lambda_{1j}, \lambda_{2j}, ..., \lambda_{nj}]^T$, où $n$ est le nombre de **RRA** dans la matrice de signature de défauts, où $\lambda_{ij}$ est donné par :

$(RRA_i, \mathcal{F}_j) \rightarrow \lambda_{ij} = 0$ si le composant $c_j$ n'appartient pas au support de $RRA_i$, $\lambda_{ij} = 1$ sinon.
Les signatures de l'ensemble de défauts $\{\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_m\}$ constituent la matrice de signature de défauts **MSD** de dimension $n \times m$.

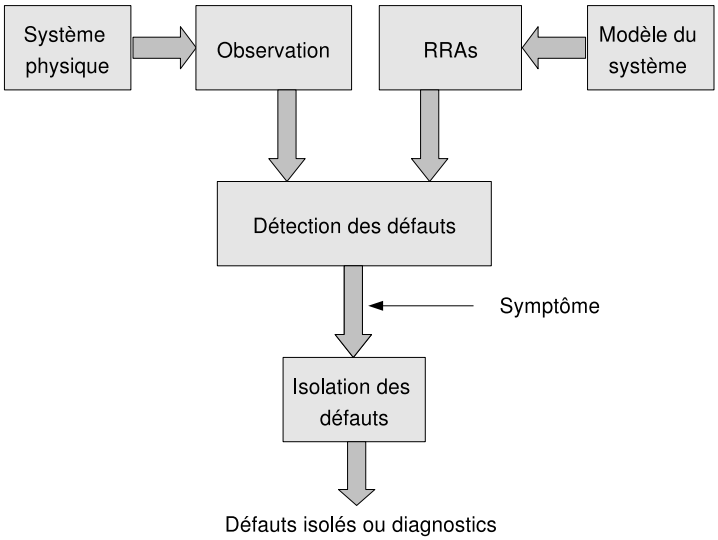


FIG. 1.4 – Diagnostics à base de **RRAs**

### 1.2.2 Méthode de diagnostic à base de reconnaissance de formes

Les techniques à base de reconnaissance de formes (Dubuisson [2001]) visent à identifier les zones d'un espace de valeur qui correspondent à des états défaillants identifiés. Il y a généralement de forts recouvrements entre ces zones. Cela montre bien que, dans le cas général, il existe plusieurs diagnostics possibles et qu'il est illusoire de penser obtenir directement la bonne explication d'un système de diagnostics automatique.

L'approche par reconnaissance de formes ne présente pas d'obstacle à la garantie des résultats. Si les zones sont correctement délimitées, il est possible d'établir avec certitude si l'on se trouve ou non dans un état anormal et, si des états défaillants ont été modélisés, d'obtenir la liste des états défaillants modélisés qui sont compatibles avec les observations.
Cette approche présente néanmoins l'inconvénient de ne s'appliquer qu'à des systèmes statiques (sans mémoire).

### 1.2.3 Méthode de diagnostic à base de réseaux bayésiens

L'un des objectifs principaux dans le domaine de la recherche en Intelligence Artificielle est d'être capable de construire et développer des systèmes dynamiques évolutifs. Ces systèmes doivent être équipés de comportements intelligents qui peuvent apprendre et raisonner.
En général, la connaissance acquise n'est pas toujours adéquate pour permettre au sys-

tème de prendre la décision la plus convenable. Pour résoudre ce genre de problème, certaines méthodologies ont été proposées. Seules les approches probabilistes s'adaptent le mieux à ce genre de problème. Ces approches probabilistes sont nommées "réseaux bayésiens" (Pearl [1988]).
Les réseaux bayésiens sont la combinaison des approches probabilistes et la théorie des graphes. Un réseau bayésien est un graphe acyclique orienté sans aucun circuit. Chaque noeud de ce graphe porte une étiquette qui est un des attributs du problème. Ces attributs sont binaires qui prennent la valeur **VRAI** ou **FAUX**.
La notion de causalité est très importante pour construire les réseaux bayésiens. La causalité signifie que deux noeuds représentant deux faits différents peuvent être en relation causale sans que l'un implique l'autre. Une fois que le réseau bayésien est consrtuit, nous cherchons souvent à déterminer des probabilités correspondant à certains événements, certaines questions.

Les réseaux bayésiens constituent un outil important pour le diagnostic des systèmes (Delcroix et al. [2002]). D'une part, ils permettent de construire un modèle de bon et de mauvais fonctionnement du système, ce qui permet de chercher les diagnostics sans disposer d'un historique de pannes. D'autre part, ces réseaux bayésiens permettent de prendre en compte les probabilités de défaillance *a priori* des composants. Ils permettent aussi de calculer facilement les probabilités de défaillance *a posteriori* des composants, c'est-à-dire avec les observations.

Le principe du diagnostic de défauts avec les réseaux bayésiens est le suivant. Il s'agit de recenser l'ensemble de faits possibles observés ou non et les relier par un graphe de causalité. Il s'agit ensuite de définir les probabilités conditionnelles associées à chaque arc du graphe de causalité avec toutes les difficultés que l'on peut imaginer pour le concepteur lors du calage de ces probabilités. L'analyse diagnostique consiste alors à partir des faits observés pour rechercher les autres faits qui ont pu vraisemblablement conduire à ces faits observables. Les diagnostics les plus vraisemblables sont alors déduits. Les défauts multiples peuvent être trouvés mais cela démultiplie le nombre de faits.

Cette approche statistique requiert la construction de nombreuses probabilités qu'il n'est pas aisé de régler. Le contexte probabiliste conduit à une absence de garantie : fondamentalement, tout est possible avec des probabilités plus ou moins importantes. En plus, cette méthode reste spécifique à certaines catégories d'application.

### 1.2.4 Méthode de diagnostic à base d'arbre de décision

Les arbres de décision sont un outil graphique puissant et très répandu pour la classification et la prédiction. Ces arbres sont des classificateurs avec une structure arborescente (voir figure 1.5) et ils utilisent une technique de recherche "top-dwon" (de haut en bas). Ils se composent de plusieurs noeuds où chaque noeud représente une décision, ou un

test, à effectuer sur un attribut donné des individus. Les arbres de décision commencent par un noeud initial nommé “racine” et ils se terminent par des noeuds terminaux nommés “feuilles” qui aident à prendre la décision pour effectuer un individu à une classe existante. Deux branches au moins peuvent être liées à chaque noeud et cela dépend des types de décision à prendre (binaire ou non). Les noeuds intermédiaires sont des noeuds de décision.
Les arbres de décision sont largement utilisés en contexte industriel pour déterminer des

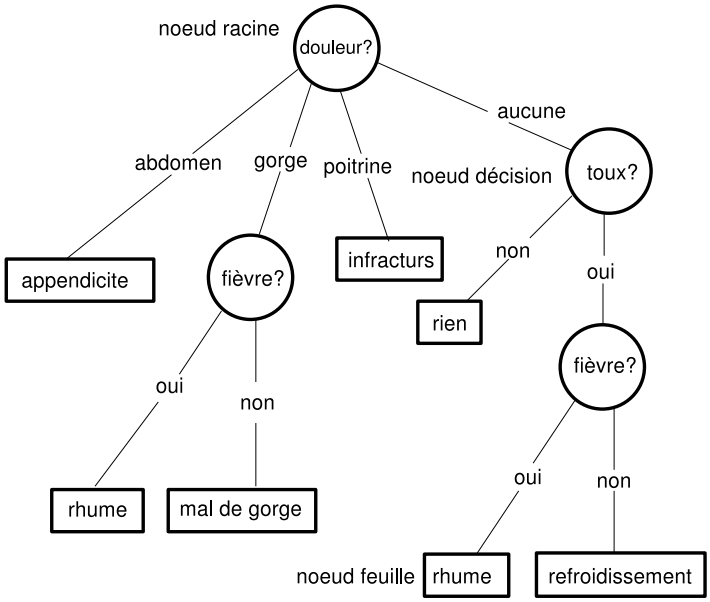


Fig. 1.5 – Exemple d’un arbre de décision

diagnostics. Généralement, les arbres de décision ont pour intrants des symptômes qui peuvent être de diverses natures.
Potentiellement, il est possible de diagnostiquer des défauts multiples en garantissant les diagnostics obtenus. Néanmoins, la garantie requiert que l’arbre de décision soit complet et que toutes les situations soient représentées, y compris les défauts multiples. Or, pratiquement, la combinatoire est telle qu’il est irréaliste d’atteindre cette complétude. Pratiquement, les arbres de décision ne garantissent donc pas que la situation réelle appartienne nécessairement aux diagnostics déduits.

### 1.2.5 Méthode de diagnostic à base de cas

Parmi les approches de diagnostic à base de cas, on trouve les techniques de fouille de données. Il s’agit d’enregistrer dans une base de connaissances les effets observés des

défauts qui se sont produits par le passé. Puis, lorsque des faits anormaux se produisent, il s'agit de rechercher des cas similaires dans la base de connaissances pour trouver des diagnostics possibles.

Cette approche est une approche par coïncidence et repose donc sur une hypothèse d'exonération implicite. Si cette approche est relativement simple à mettre en place, elle n'offre aucune garantie sur les résultats comme toute approche par coïncidence. Elle pose aussi d'autres problèmes. Si l'on considère la possibilité de défauts multiples, on réalise combien il est difficile d'obtenir une base de connaissances complète. De plus, comme nous l'avons déjà vu précédemment, chaque système est particulier. Il est donc difficile d'exploiter la connaissance acquise sur un système pour enrichir la base d'un autre. Il faut donc regénérer la base de connaissances pour chaque système. Or, une base de connaissances n'est complète que si tous les défauts, les variantes de défauts et les combinaisons de défauts possibles se sont déjà produits. On comprend aisément les problèmes que cela pose pour les défauts critiques qui ne se produisent que rarement. Enfin, cette approche ne permet pas de prendre en compte les modèles de comportement connus du système.

D'autres méthodes de diagnostic sont présentés dans Desinde [2006].

## 1.3 Difficultés liées aux applications

La modélisation d'un système représente la deuxième étape du diagnostic de défauts à base de modèles (voir figure 1.1). Cette modélisation représente un point important du diagnostic pour les approches **FDI**, **DX** et **bridge**. Mais il y a une grande difficulté de mise en oeuvre des méthodes usuelles de diagnostic :

- Difficulté due au fait que la connaissance experte est partiellement formalisée, c'est à dire qu'il est difficile de modéliser le comportement du système. Prenons par exemple le réseau informatique représenté dans la figure 1.6, nous remarquons qu'il n'est pas facile de modéliser le comportement des composants du système (ordinateurs, imprimantes,...) exhaustivement.
- Difficultés liées à la concommittance de différents types de modèles (voir figure 1.7) :
    1. basé sur des équations différentielles
    2. basé sur des équations statiques
    3. basé sur des réseaux de Petri
    4. Basé sur des règles qualitatives

Devant ce constat, nous en déduisons qu'il faut privilégier les outils qui permettent d'appréhender cette diversité.

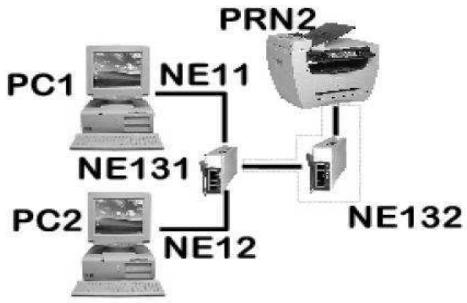


FIG. 1.6 – Réseau informatique

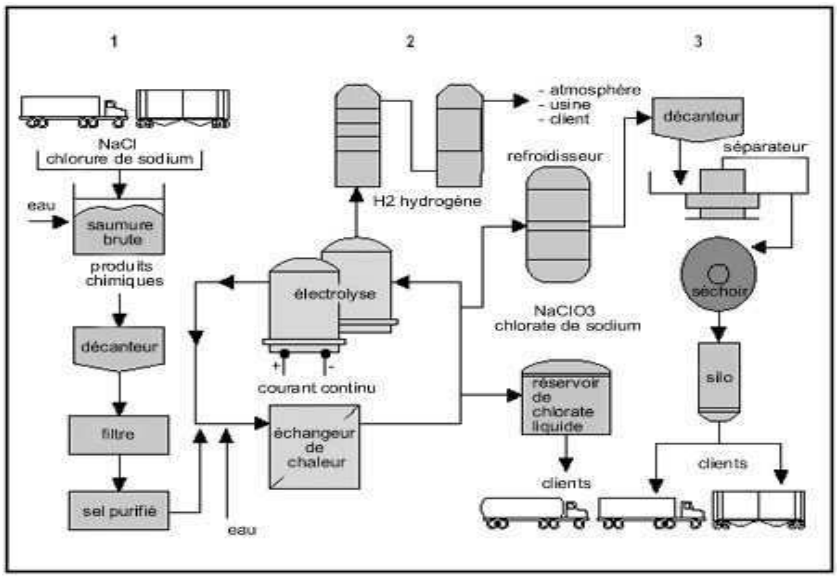


FIG. 1.7 – Procédé chimique

## 1.4 Conclusion

Dans ce chapitre, nous avons montré les étapes nécessaires pour concevoir un système de diagnostic de défauts et nous nous concentrerons particulièrement sur l'étape du placement de capteurs et l'étape de conception des **SST**s dans la suite de notre thèse. Nous avons aussi introduit les principaux travaux sur les méthodes de diagnostic de défauts. Nous nous sommes concentrés particulièrement sur le diagnostic à base de modèles représentant l'outil principal de diagnostic dans les communautés de diagnostic **FDI**, **DX** et **bridge**. Nous avons montré pourquoi il fallait privilégier les méthodes permettant d'appréhender la diversité des modèles. Ce constat nous a conduit à nous intéresser à la modélisation structurelle qui sera présentée dans le chapitre suivant.

# Chapitre 2

# Modélisation structurelle pour le diagnostic des systèmes physiques

## 2.1 Introduction

Des outils performants et génériques permettant d'analyser les systèmes physiques dans toute leur diversité sont nécessaires à la conception de systèmes de diagnostic de défauts. Dans ce chapitre, la représentation structurelle des systèmes est analysée. La représentation structurelle consiste à décrire les composants et les flux qui les lient. Ces flux sont généralement représentés par des variables. Nous examinerons différentes modélisations structurelles adoptées par (Declerck and Staroswiecki [1991], Krysander and Nyberg [2002], Izadi-Zamanabadi et al. [2003], Blanke et al. [2006], Dustegor et al. [2006], Commault et al. [2006b]) avant d'évoquer celle que nous avons adoptée dans la suite.

## 2.2 Modélisation des systèmes

Établir un modèle pour le diagnostic est une tâche plus complexe qu'établir un modèle pour commander un système. Il faut représenter les modes de comportements ou de fonctionnements associés aux contraintes de manière à être capable d'analyser les symptômes engendrés (un test ou une relation de redondance analytique étant une contrainte particulière).

La modélisation d'un système physique se base sur différents niveaux de représentation (Chittaro and Kumar [1998]), parmi lesquels : le niveau fonctionnel et comportemental, et le niveau structurel.

### 2.2.1 Modélisation comportementale et fonctionnelle

Le modèle du comportement d'un système $\Sigma$ peut être défini par un ensemble de relations (contraintes $K$), définies sur un espace correspondant à un ensemble de variables $V$. L'ensemble de contraintes $K$ est partitionné en deux sous-ensembles : $K_{proc}$, qui correspond aux composants du système sans capteurs et actionneurs et $K_{obs}$, qui correspond aux capteurs et actionneurs que nous ajoutons au système. En plus de la notion de contraintes, la notion de réalisation doit être introduite : elle permet de tenir compte du fait qu'une relation ne peut pas toujours être utilisée de n'importe quelle façon, en particulier lorsqu'il peut y avoir des problèmes d'inversibilité et plus généralement, certaines variables ne peuvent pas être déduites d'autres pour une contrainte donnée.

### 2.2.2 Modélisation structurelle

Un système automatisé est constitué d'un ensemble de composants qui interagissent entre eux afin de réaliser la (les) tâche(s) pour laquelle (lesquelles) le système a été conçu. Chaque composant peut être décrit par un ensemble de relations (contraintes) algébriques, continues ou discrètes, linéaires ou non linéaires, statiques ou dynamiques qui définissent le comportement attendu des phénomènes liés.
Les variables impliquées dans ces contraintes peuvent être classées en deux types :

- les variables inconnues.
- les variables connues, soit parce qu'elles sont mesurées, contrôlées ou connues par hypothèses.

Le modèle structurel ne considère que les interconnexions entre les contraintes décrivant les modes de comportement des composants d'un système. Cela revient à voir un système comme un ensemble de composants interconnectés.

## 2.3 Rappel des travaux sur le raisonnement structurel pour le diagnostic de défauts

L'analyse structurelle est un outil puissant et performant qui permet de déterminer de nombreuses propriétés d'un système. Ces propriétés sont obtenues à partir de la seule connaissance de l'existence de liens entre les composants d'un système sans que les modèles de comportement détaillés soient nécessaires. Ce pré-traitement, qui peut être entièrement automatisé, même pour de très grands systèmes, permet par exemple de connaître les conditions structurelles d'observabilité ou de commandabilité (Staroswiecki [2002], Blanke et al. [2003], ), les possibilités de détection et de localisation des défauts

(Pulido and Alonso [2000], Izadi-Zamanabadi and Staroswiecki [2000], Izadi-Zamanabadi [2002], Krysander and Nyberg [2002], Dustegor et al. [2006], Blanke et al. [2006]). Cette analyse structurelle permet aussi de déterminer un placement de capteurs satisfaisant un cahier des charges lié à des critères de diagnosticabilité (Maquin et al. [1997], Carpentier et al. [1997], Travé-Massuyès et al. [2006], Commault and Dion [2007], Frisk and Krysander [2007]). Pour toutes les propriétés structurelles étudiées, les conditions obtenues sont nécessaires mais en général non suffisantes car elles sont indépendantes de la valeur réelle des paramètres.

## 2.4 Représentation structurelle des systèmes

Il y a plusieurs méthodes pour représenter structurellement le modèle d'un système. Dans cette section, différentes représentations structurelles des systèmes à diagnostiquer sont présentées.

### 2.4.1 Modélisation par matrice structurelle

Considérons un système $\Sigma$. Le modèle du système $\Sigma = (K, V)$ peut être défini par un ensemble de $n$ contraintes (relations) $K$, qui lient un ensemble de $m$ variables $V$. L'ensemble de variables $V$ est constitué de deux ensembles : l'ensemble de variables connues $O$ et l'ensemble de variables inconnues $X$ (Blanke et al. [2006]).
Nous savons qu'une contrainte ne peut pas être toujours utilisée de n'importe quelle façon car parfois certaines variables ne peuvent pas être déduites des autres variables au moyen de la contrainte. Pour des modèles mathématiques, on parle d'inversibilité. C'est pourquoi, nous allons définir la notion de causalité et la notion de déductibilité (calculabilté).
L'analyse de la causalité dans les systèmes physiques a donné lieu à plusieurs conceptions dont les plus connues ont été proposées par (Iwasaki and Simon [1986], De Kleer and Brown [1986]). Iwasaki et Simon déduisent l'ordre causal à partir d'une analyse purement structurelle d'un modèle comportemental (décrit par un ensemble d' équations). Avec cette approche, il n'est pas nécessaire de résoudre les équations, contrairement à l'approche proposée par De Kleer et Brown qui consiste à déterminer les chemins de propagation suivis par les perturbations sur les variables en réalisant une propagation de contraintes. La notion de causalité est importante pour la conception des systèmes de diagnostic.
La notion de la causalité étant assimilée à la notion de déductibilité (calculabilité), un ordonnancement causal permet d'orienter les relations entre les variables apparaissant dans les contraintes du système.
Nous allons définir maintenant cette notion de déductibilité (calculabilité) assimilée à la

notion de causalité.

**Définition 2.4.1.** *déductibilité (calculabilité)*
*Soient $v_i, i = 1, ..., p, ...m$ des variables intervenant dans la contrainte $k_j$. La variable $v_p$ est déductible (calculable) si ses valeurs peuvent être déduites à travers la contrainte $k_j$ en utilisant les autres variables $v_i, i = 1, ..., m$ avec $i \neq p$, tel que les valeurs de ces variables sont connues.*

Le modèle du système peut être représenté par une matrice structurelle dont les colonnes représentent l'ensemble des variables et les lignes représentent l'ensemble des contraintes du modèle. Dans cette matrice, l'intersection entre chaque colonne et ligne $a(i, j)$ prend une de ces 3 valeurs :

- $a(i, j) = 0$ si le variable $v_i$ n'est pas apparue dans la contrainte $k_j$
- $a(i, j) = 1$ si le variable $v_i$ est déductible pour la contrainte $k_j$
- $a(i, j) = -1$ si le variable $v_i$ n'est pas déductible pour la contrainte $k_j$

Considérons un système de deux bacs présenté dans le figure 2.1.

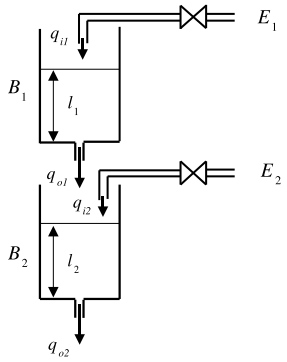


FIG. 2.1 – Système de deux bacs

Les composants de ce système sont : deux bacs $B_1$ et $B_2$, deux vannes $E_1$ et $E_2$ et deux capteurs de hauteur d'eau $L_1$ et $L_2$. Les variables du système sont : les débits d'eau en entrée des bacs $q_{i_1}$ et $q_{i_2}$, les débits d'eau en sortie des bacs $q_{o_1}$ et $q_{o_2}$ et les hauteurs d'eau dans les deux bacs $l_1$ et $l_2$

Le modèle de comportement $MC$ du système de deux bacs est décrit par quatre contraintes :

$$\begin{aligned} k_1 &: S\frac{dl_1}{dt} = q_{i_1} - q_{o_1} \\ k_2 &: q_{o_1} = kl_1 \\ k_3 &: S\frac{dl_2}{dt} = q_{i_2} + q_{o_1} - q_{o_2} \\ k_4 &: q_{o_2} = kl_2 \end{aligned} \tag{2.4.1}$$

Et le modèle d'observation $MO$ est décrit par les contraintes suivant :

$$\begin{aligned} k_5 &: l_1 = \tilde{l_1} \\ k_6 &: l_2 = \tilde{l_2} \\ k_7 &: q_{i1} = \tilde{q_{i_1}} \\ k_8 &: q_{i2} = \tilde{q_{i_2}} \end{aligned} \tag{2.4.2}$$

Concernant les systèmes comportant des équations différentielles ordinaires, différentes représentations structurelles sont proposées.
Concernant la première représentation structurelle, les contraintes $k_1$ et $k_3$ induisent deux contraintes implicites :

$$\begin{aligned} k_9 &: l_1 = \int_0^t \frac{dl_1}{dt} \\ k_{10} &: l_2 = \int_0^t \frac{dl_2}{dt} \end{aligned} \tag{2.4.3}$$

Les variables connues sont : $(\tilde{q_{i_1}}, \tilde{q_{i_2}}, \tilde{l_1}, \tilde{l_2})$ et les variables inconnues sont : $(q_{i_1}, q_{i_2}, q_{o_1}, q_{o_2}, l_1, l_2, \frac{dl_1}{dt}, \frac{dl_2}{dt})$.
Pour éviter les inconvénients liés à la dérivation, dans les contraintes $k_9$ et $k_{10}$, les variables $\frac{dl_1}{dt}$ et $\frac{dl_2}{dt}$ ne sont pas considérées comme déductibles (calculables). La première représentation est donnée par la matrice structurelles suivante :

Tab. 2.1 – La matrice structurelle du système de deux bacs

| | $q_{i_1}$ | $q_{o_1}$ | $q_{i_2}$ | $q_{o_2}$ | $l_1$ | $\frac{dl_1}{dt}$ | $l_2$ | $\frac{dl_2}{dt}$ | $\tilde{q_{i_1}}$ | $\tilde{q_{i_2}}$ | $\tilde{l_1}$ | $\tilde{l_2}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_2$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_3$ | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_4$ | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $k_5$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $k_6$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| $k_7$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $k_8$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $k_9$ | 0 | 0 | 0 | 0 | 1 | -1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{10}$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | -1 | 0 | 0 | 0 | 0 |

Dans les contraintes $k_9$ et $k_{10}$, seulement les variables $l_1$ et $l_2$ sont considérées comme déductibles afin d'éviter les dérivations qui amplifient les bruits de haute fréquence. Cette représentation est utilisée pour le diagnostic dans (Blanke et al. [2006]).

Concernant la deuxième représentation structurelle du système, les contraintes implicites $k_9$ et $k_{10}$ ne sont pas présentes et les variables inconnues sont : $(q_{i_1}, q_{i_2}, q_{o_1}, q_{o_2}, l_1, l_2)$.

Dans ce cas, les variable $(q_{i_1}, q_{o_1})$ dans la contrainte $k_1$ et les variables $(q_{i_2}, q_{o_1}, q_{o_2})$ dans la contrainte $k_3$ ne sont pas calculables afin d'éviter les dérivations qui amplifient les bruits hautes fréquences. Cette représentation est donnée par la matrice structurelle suivante :

TAB. 2.2 – La matrice structurelle du système de deux bacs

| | $q_{i_1}$ | $q_{o_1}$ | $q_{i_2}$ | $q_{o_2}$ | $l_1$ | $l_2$ | $\tilde{q_{i_1}}$ | $\tilde{q_{i_2}}$ | $\tilde{l_1}$ | $\tilde{l_2}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $k_1$ | -1 | -1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $k_2$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $k_3$ | 0 | -1 | -1 | -1 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_4$ | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_5$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| $k_6$ | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| $k_7$ | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $k_8$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

Cette représentation est utilisée dans (Frisk et al. [2003]).

### 2.4.2 Représentation structurelle par graphe

Dans de nombreux domaines de la science, l'utilisateur est amené à traduire un problème par des graphes avec des noeuds représentant des individus, des objets et avec des arcs orientés ou non, reliant les noeuds et symbolisant une relation entre eux. Ces graphes sont utilisés dans différentes applications telles que les réseaux de communication, les circuits électriques, les réseaux de Petri, etc. La théorie des graphes est apparue pour la première fois dans (Köing [1989] puis dans Berge [1958]). La théorie des graphes a énormément progressé et a été utilisée pour représenter les structures d'une grande variété de problèmes pratiques. Les types de graphes utilisés pour représenter les problèmes sont nombreux : graphe linéaire, graphe fonctionnel, réseau de Petri, graphe orienté (digraphe), graphe biparti, etc.

#### A Représentation structurelle par digraphe associé à la représentation d'état

En donnant une orientation aux arêtes d'un graphe, on obtient un digraphe (ou graphe orienté). Un digraphe est défini par deux ensembles : un ensemble d'éléments appelés "**sommets**" et un ensemble d'éléments appelés "**arcs**".
Dans ce paragraphe, nous présentons une utilisation d'un digraphe pour représenter un système donné par des équations d'état (Blanke et al. [2003]). Supposons un système

représenté par les équations suivantes :

$$\begin{aligned} \frac{dx_i(t)}{dt} &= g(x(t), u(t), \Theta) \\ y(t) &= h(x(t), u(t), \Theta) \end{aligned} \tag{2.4.4}$$

tel que $x(t)$ représente l'état du système, $u(t)$ les entrées du système, $y(t)$ les sorties du système et $\Theta$ un vecteur de paramètres.
L'ensemble de variables et contraintes de ce système sont :

$$\begin{aligned} V &= \{x, u, y\} \\ K &= \{g, h\} \end{aligned} \tag{2.4.5}$$

où $g$ représente l'ensemble des contraintes différentielles : $\frac{dx_i(t)}{dt} - g_i(x(t), u(t), \Theta) = 0, i = 1, ..., n$ et $h$ représente l'ensemble des contraintes de mesure : $y_j(t) - h_j(x(t), u(t), \Theta) = 0, j = 1, ..., n$.

Le comportement de ce système peut être représenté structurellement par un graphe orienté (Blanke et al. [2003], Blanke et al. [2006]).
En effet, le digraphe lié avec le système représenté par les équations 2.4.4 est un graphe dont les sommets sont les variables d'entrée, sortie et d'état, et dont les arcs sont définis par les règles suivantes :

- il existe un arc du sommet $x_k$ (respectivement du sommet $u_l$) au sommet $x_i$ si et seulement si la variable d'état $x_k$ (respectivement la variable d'entrée $u_l$) apparaît dans la fonction $g_i$ (c'est à dire $\frac{dg_i}{dx_k}$, respectivement $\frac{dg_i}{du_l}$ est non nul).
- il existe un arc du sommet $x_k$ au sommet $y_j$ si et seulement si la variable d'état $x_k$ apparaît dans la fonction $h_j$.

En effet, un arc du sommet $x_k$ (respectivement $u_l$) vers le sommet $x_i$ signifie que l'évolution de la dérivée de $x_i(t)$ dépend de l'évolution de $x_k(t)$ (respectivement $u_l(t)$). De la même manière, un arc du sommet $x_k$ au sommet $y_j$ signifie que l'évolution de la sortie $y_j(t)$ dépend de l'évolution de la variable d'état $x_k(t)$.

Considérons le système linéaire suivant :

$$\begin{aligned} \frac{dx(t)}{dt} &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

tel que les matrices $A$, $B$ et $C$ sont données par : $A = \begin{bmatrix} 0 & 0 & a \\ 0 & 0 & b \\ c & d & e \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 0 \\ f \end{bmatrix}$, $C = \begin{bmatrix} p & 0 & 0 \end{bmatrix}$.

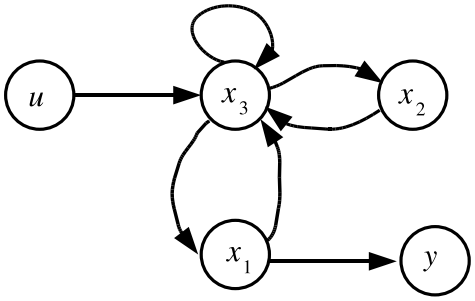


FIG. 2.2 – Digraphe du système linéaire

Leurs structures sont :

$$S_A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}, S_B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, S_C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

Le digraphe associé au système est donné par la figure 2.2 où $x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$.

Reprenons l'exemple de deux bacs et supposant que les variables d'état sont : ($l_1$,$l_2$), les variables d'entrée sont : $\tilde{q_{i_1}}, \tilde{q_{i_2}}$ et les variables de sortie sont : ($\tilde{l_1}$,$\tilde{l_2}$). En remplaçant les contraintes $k_2$ et $k_4$ dans les contraintes $k_1$ et $k_3$ et en remplaçant les variables $q_{i_1}, q_{i_2}$ par leur mesures, nous obtenons les équations suivantes :

$$\begin{array}{l} \frac{dl_1}{dt} = \frac{1}{S}\tilde{q_{i_1}} - \frac{k}{S}l_1 \\ \frac{dl_2}{dt} = \frac{1}{S}\tilde{q_{i_2}} + \frac{k}{S}l_1 - \frac{k}{S}l_2 \end{array} \tag{2.4.6}$$

Nous pouvons déduire la représentation d'état suivante :

$$\frac{d}{dt}\begin{bmatrix} l_1 \\ l_2 \end{bmatrix} = \begin{bmatrix} -\frac{k}{S} & 0 \\ \frac{k}{S} & -\frac{k}{S} \end{bmatrix}\begin{bmatrix} l_1 \\ l_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{S} & 0 \\ 0 & \frac{1}{S} \end{bmatrix}\begin{bmatrix} \tilde{q_{i_1}} \\ \tilde{q_{i_2}} \end{bmatrix}.$$

$$\begin{bmatrix} \tilde{l_1} \\ \tilde{l_2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} l_1 \\ l_2 \end{bmatrix}$$

Leurs structures sont :

$$S_A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, S_B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, S_C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Le digraphe associé avec le système est donné par la figure 2.3.

Cette représentation structurelle peut être trouvée dans (Commault et al. [2002], Dion et al. [2003], Commault and Dion [2004]) ou dans (Commault et al. [2005]) pour le placement de capteurs.

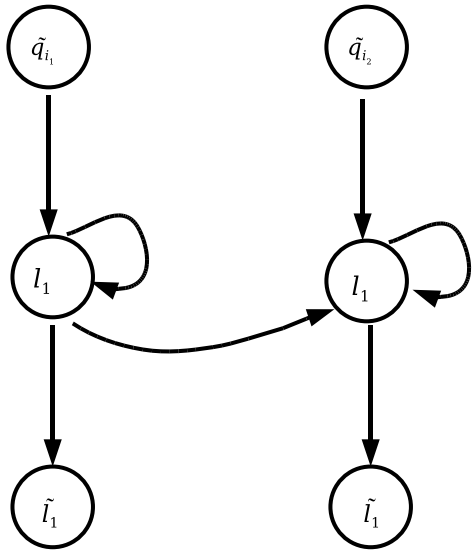


FIG. 2.3 – Digraphe de deux bacs

## B Représentation structurelle par graphe biparti

Dans ce paragraphe, nous allons nous intéresser à la représentation structurelle des problèmes à l'aide de graphes bipartis (Blanke et al. [2006]). Cette représentation est un formalisme graphique très intéressant. Elle met en évidence de nombreuses propriétés difficilement observables avec d'autres représentations.

Ce paragraphe introduit la modélisation structurelle d'un système par graphe biparti. Un graphe biparti est un graphe dont l'ensemble de noeuds est divisé en deux sous-ensembles tel que chaque arc de ce graphe relie un noeud du premier ensemble à un noeud de l'autre ensemble. Un graphe biparti peut être un digraphe et un digraphe est généralement un graphe biparti en diagnostic. En ce qui concerne le digraphe, les noeuds représentent les variables du système et les arcs représentent les équations d'état (contraintes). Ce digraphe peut être représenté comme graphe biparti en considérant que les variables et les contraintes sont les noeuds.

Le modèle d'un système $\Sigma = (K, V)$ peut être défini par un ensemble de $n$ contraintes $K$, qui relient un ensemble de $m$ variables $V$. L'ensemble des variables $V$ est constitué de deux sous-ensembles (Blanke et al. [2006]) : l'ensemble de variables connues $O$ et l'ensemble de variables inconnues $X$, $V = X \cup O$. La structure d'un modèle peut être représentée par un graphe biparti $G = (K, V, A)$, où $A$ est un ensemble d'arcs tel que $a(i, j) \in A$ si et seulement si la variable $v_j \in V$ apparaît dans la contrainte $k_i \in K$. Le graphe biparti associé à un système $\Sigma = (K, V)$ est un graphe non orienté avec deux ensembles de sommets $K$ et $V$ et un ensemble d'arcs définit par la règle suivante :

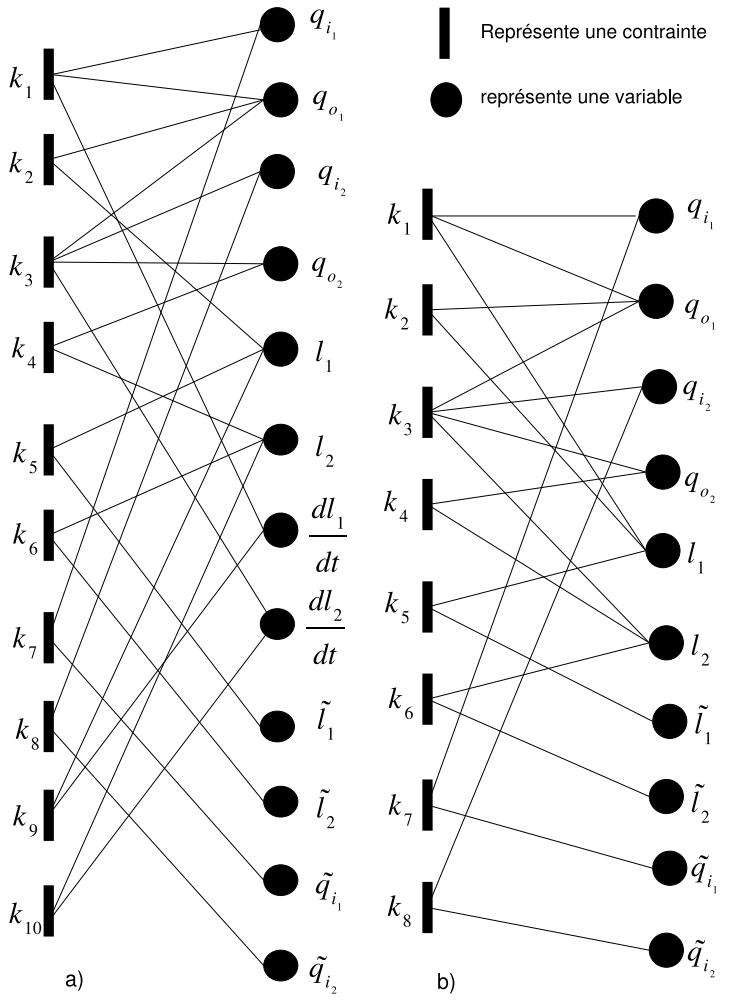


FIG. 2.4 – Graph biparti du système de deux bacs

– un arc existe entre le sommet $k_i \in K$ et le sommet $v_i \in V$ si et seulement si le variable $v_i$ apparaît dans la contrainte $k_i$.

La matrice d'incidence $\mathcal{M}$ correspondant à un graphe biparti $G = (K, V, A)$ est une matrice booléenne où les lignes correspondent aux contraintes et les colonnes correspondent aux variables et $\mathcal{M} = \{m_{ij} | m_{ij} = 1$ si $v_j$ apparaît dans $k_i$, 0 autrement$\}$.

Reprenons l'exemple de deux bacs. Nous allons représenter le graphe biparti correspondant à cet exemple avec ou sans les contraintes implicites. Le graphe biparti correspondant à la représentation avec contraintes implicites est représenté par la figure 2.4.a, tandis que le graphe biparti correspondant à la représentation sans contraintes implicites est représenté par la figure 2.4.b.

Le modèle structurel d'un système est une abstraction de son modèle de comportement. Deux systèmes sont équivalents structurellement s'ils ont la même structure.

L'analyse structurelle se sert de la structure d'un système pour etudier ses propriétés. Plusieurs outils spécifiques ont été développés dans ce but.

**B.1 Couplage** La notion de couplage est centrale dans les approches liées au diagnostic exploitant les graphes bipartis.
Pour introduire la notion de couplage, supposons que dans un ensemble de $Q$ hommes et $N$ femmes, on désire créer des couples en tenant compte des préférences de chacun. Dans un premier temps, nous construisons une table de deux colonnes avec les noms des hommes dans l'une, ceux des femmes dans l'autre. Nous relions les noms de l'une des colonnes à ceux de l'autre selon les préférences des personnes. Nous obtenons alors un graphe biparti $G(Y, Z, A)$ où $Y$ est l'ensemble des hommes et $Z$ l'ensemble des femmes et $A$ est l'ensemble des arcs reliant $Y$ et $Z$.

Supposons que le nombre $q = card(Y)$ d'éléments de $Y$ est inférieur ou égal au nombre $n = card(Z)$ d'éléments de $Z$. Trouver un couplage de $Y$ dans $Z$ consiste à trouver un sous-ensemble d'arcs deux à deux non adjacents (sans extrémités communes). Tous les sommets de $Y$ sont ainsi associés à un arc unique. Les sommets de $Z$ sont associés à un arc au plus. Le couplage obtenu est alors maximal.

**Définition 2.4.2.** *(Couplage)*
*Soit un graphe biparti $G(K, X, A)$ où $K$ est un ensemble des contraintes du système. $X$ est l'ensemble des variables $X \subset V$ et $A$ est l'ensemble des arcs liant $K$ à $X$. Un couplage $M$ dans $G(K, X, A)$ est un ensemble d'arcs tel qu'il n'y pas un couple d'arcs dans $M$ qui ont un sommet commun.*

**Définition 2.4.3.** *(Couplage maximal)*
*Soit un graphe biparti $G(K, X, A)$. Un couplage maximal est un couplage $M$ tel que aucun arc ne peut être ajouté sans violer la propriété selon laquelle deux arcs ne doivent pas avoir de sommet commun.*

**Définition 2.4.4.** *(Couplage complet)*
*Soit un graphe biparti $G(K, X, A)$. Un couplage $M$ dans ce graphe est dit complet par rapport à $K$ si $|M| = |K|$ (respectivement un couplage est dit complet par rapport à $X$ si $|M| = |X|$).*

En général, différents couplages peuvent être trouvés dans un graphe biparti. Par exemple, dans la figure 2.4.b, plusieurs couplages peuvent être trouvés dans le graphe biparti $G(K, X, A)$. Deux d'entre eux sont illustrés dans la figure 2.5.a et 2.5.b. Nous remarquons que ces deux couplages présentés sont complets par rapport aux variables inconnues $X$.

- **Graphe biparti orienté associé avec un couplage**
  Définir un couplage dans un graphe biparti $G(K, X, A)$ impose quelques orientations. Le graphe biparti devient alors être orienté. Le problème d'orientation ne

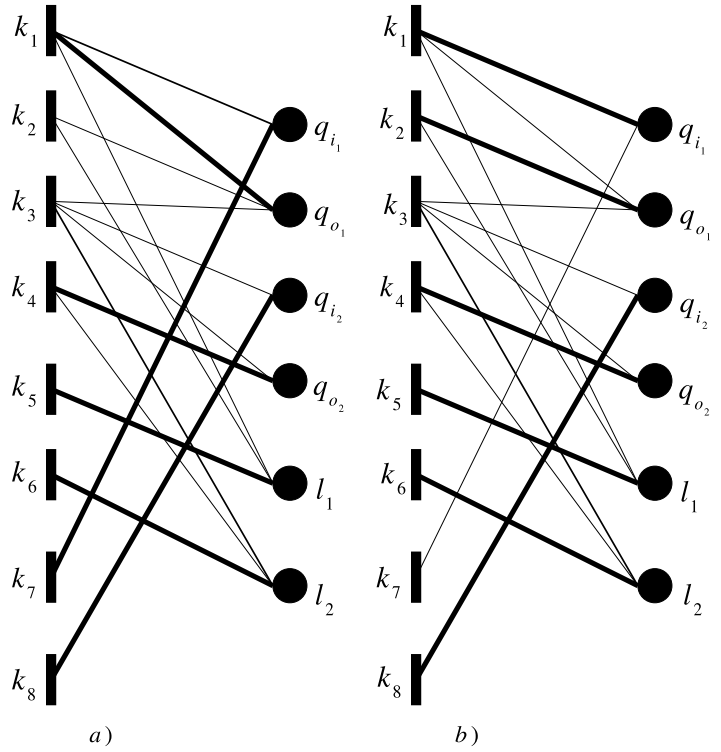


FIG. 2.5 – Deux couplages possibles pour le système de deux bacs

s'applique pas aux contraintes mais juste aux variables. Par exemple, reprenons la contrainte suivante du système de deux bacs avec les contraintes implicites (première représentation, voir tableau 2.1) :

$$k_1 : S\frac{dl_1}{dt} = q_{i_1} - q_{o_1} \tag{2.4.7}$$

Les trois variables $(\frac{dl_1}{dt}, q_{i_1}, q_{o_1})$ sont considérées comme déductibles, alors la contrainte $k_1$ peut être utilisée pour calculer chacune de ces trois variables quand les deux autres variables sont connues ou déduites. Une fois un couplage choisi, nous remarquons que chaque contrainte couplée est maintenant liée avec une variable couplée et plusieurs variables non couplées.
Puisque quelques contraintes ne pourraient pas être couplées, les règles suivantes sont appliquées :

- Une contrainte est dite couplée si les arcs liés à cette contrainte sont équipés d'orientation :
  - des variables non couplées (entrées) à la contrainte
  - de la contrainte à la variable couplée (sortie)
- Une contrainte est dite non couplée si toutes les variables sont considérées comme entrées, c'est-à-dire cette contrainte ne sera pas utilisée pour déduire une variable et par conséquent, tous les arcs sont orientés des variables à la contrainte.

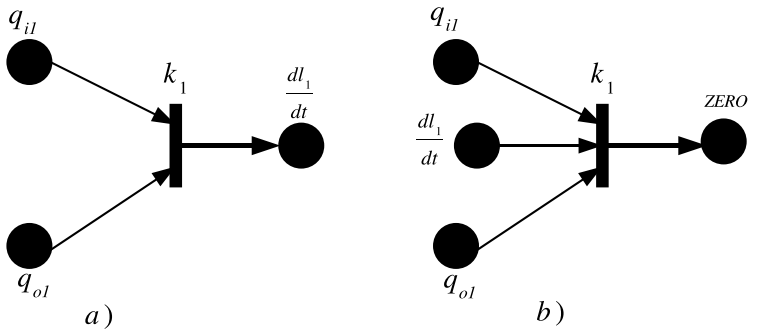


FIG. 2.6 – a) une contrainte couplée b) une contrainte non couplée

La figure 2.6 représente une contrainte couplée et une autre non couplée : Dans la figure 2.6.a, nous remarquons que la contrainte $k_1$ est couplée avec la variable $\frac{dl_1}{dt}$, c'est-à-dire, la variable $\frac{dl_1}{dt}$ sera déduite des variables $q_{i_1}$ et $q_{o_1}$ en utilisant la contrainte $k_1$. Dans la figure 2.6.b, nous remarquons que la contrainte $k_1$ n'est couplée avec aucune variable, c'est-à-dire cette contrainte ne sera pas utilisée pour déduire une de ces variables.

Pour bien comprendre l'interêt de ces règles, considérons un couplage $M$ et un arc $(k, x)$ où $k$ est une contrainte et $x$ est une variable inconnue. La variable $x$ peut être considérée comme une sortie de la contrainte $k$ tandis que les autres variables dans $k$ sont les entrées. Le couplage représente une certaine tâche de causalité par laquelle la contrainte $k$ est utilisée pour calculer la variable $x$ en supposant que les autres variables sont connues. La causalité est dépeinte en orientant tous les arcs $(k_i, x_j)$ du graphe biparti $G(K, X, A)$ de la façon suivante :

– si $k_i$ et $x_i$ sont couplés, $k_i \longrightarrow v_i$
– sinon $v_i \longrightarrow k_i$

Reprenons le graphe biparti représenté dans la figure 2.4.a et choisissons un couplage représenté par la figure 2.7. Le graphe orienté associé avec ce couplage est représenté dans la figure 2.8. Dans cette figure, nous remarquons que les contraintes $(k_5, k_6, k_7, k_8)$ sont utilisées pour déduire les variables $(l_1, l_2, q_{i_1}, q_{i_2})$ des observations $(\tilde{l_1}, \tilde{l_2}, \tilde{q_{i_1}}, \tilde{q_{i_2}})$. Ces variables déduites permettent de déduire les autres variables du système. Nous remarquons aussi que les contraintes $(k_9, k_{10})$ sont des contraintes non couplées c'est-à-dire qu'elles ne seront pas utilisées pour déduire des variables, c'est pourquoi elles sont liées au zéro.

– **Interprétation causale**

Le but de ce paragraphe est de discuter de l'interprétation causale du graphe biparti orienté lié avec un couplage. En effet, sélectionner une paire $(k, x)$ pour appartenir à un couplage, implique que la variable $x$ doit être déductible dans la contrainte $k$. Cependant si la déductibilité n'est pas systématique, tous les arcs $(k, x)$ ne peuvent

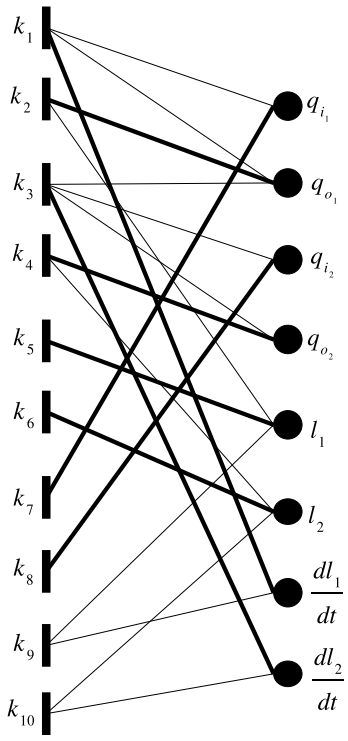


FIG. 2.7 – Un couplage

pas intervenir dans un couplage. Une situation évidente dans laquelle $(k, x)$ ne peut pas être couplé est quand $k$ est non inversible en ce qui concerne $x$ ($x$ est une variable non déductible dans $k$). Reprenons la contrainte suivante du système de deux bacs sans les contraintes implicites (deuxième représentation, voir figure 2.2) :

$$k_1 : S\frac{dl_1}{dt} = q_{i_1} - q_{o_1} \tag{2.4.8}$$

Dans cette contrainte, la variable $l_1 = \int_0^t \frac{1}{S}(q_{i_1} - q_{o_1})dt$ peut être calculée en supposant que les variables $(q_{i_1}, q_{o_1})$ sont connues c'est-à-dire la variable $l_1$ peut être couplée avec la contrainte $k_1$. Par contre la variable $q_{i_1}$ ne peut pas être déduite en supposant que les variables $q_{o_1}$ et $l_1$ sont connues pour éviter les dérivations qui amplifient les bruits hautes fréquences. De la même façon, la variable $q_{o_1}$ ne peut pas être déduite en supposant que les variables $q_{i_1}$ et $l_1$. Par conséquent, les variables $q_{i_1}$ et $q_{o_1}$ ne peuvent jamais être couplées avec la contrainte $k_1$.
La figure 2.9 représente les couplages possibles et impossibles .

**B.2 Décomposition de Dulmage-Mendelsohn** La décomposition de Dulmage-Mendelsohn notée décomposition DM est un outil intéressant pour l'analyse structurelle des systèmes à diagnostiquer. La décomposition DM peut être obtenue en appliquant des algorithmes classiques de la théorie des graphes (Dulmage and Mendelsohn [1959],

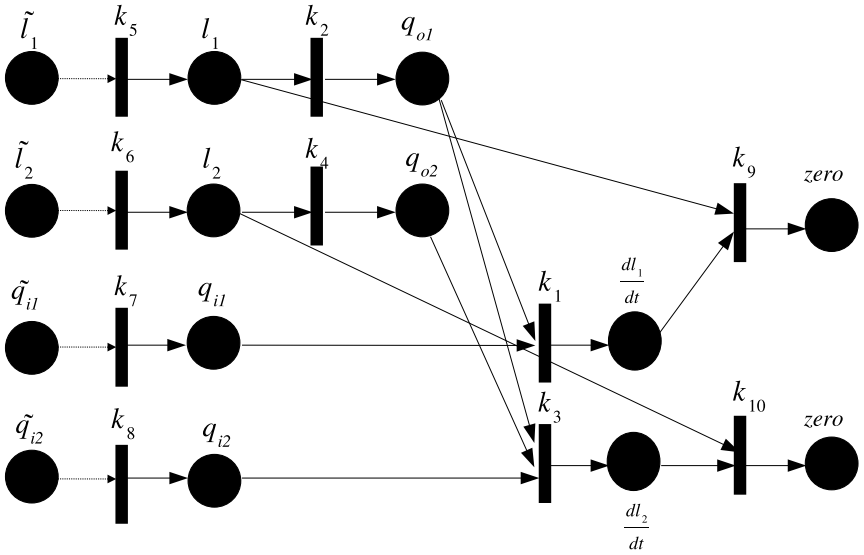


FIG. 2.8 – Graphe biparti orienté associé avec le couplage

Murota [1987]). Ceux-ci consistent à permuter les colonnes et les lignes de la matrice d'incidence représentant le graphe afin d'obtenir une représentation bloc triangulaire inférieur.

Quelques définitions sont proposées pour illustrer la base de la décomposition DM.

**Définition 2.4.5.** *graphe sur-déterminé ou (sur-contraint)*
*Un graphe biparti $G(K, X, A)$ est nommé sur-déterminé ou (sur-contraint) si il existe un couplage complet par rapport aux variables $X$ et non par rapport aux contraintes $K$*

**Définition 2.4.6.** *graphe juste-déterminé ou (juste-contraint)*
*Un graphe biparti $G(K, X, A)$ est nommé juste-déterminé ou (juste-contraint) si il existe un couplage complet par rapport aux variables $X$ et aux contraintes $K$*

**Définition 2.4.7.** *graphe sous-déterminé ou (sous-contraint)*
*Un graphe biparti $G(K, X, A)$ est nommé sous-déterminé ou (sous-contraint) si il existe un couplage complet par rapport aux contraintes $K$ et non par rapport aux variables $X$*

Une décomposition Dulmage-Mendelshon (Dulmage and Mendelsohn [1959]) d'un graphe biparti $G = (K, X, A)$ conduit à trois composants canoniques nommés $K^+$, $K^0$, $K^-$ (voir la figure 2.10) :

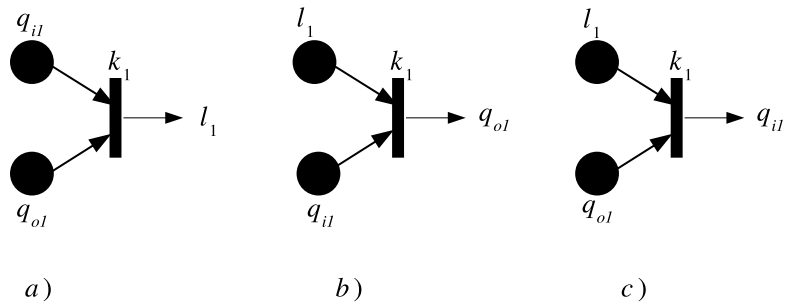


FIG. 2.9 – a) Un couplage possible, b) Un couplage impossible et c) Un autre couplage impossible

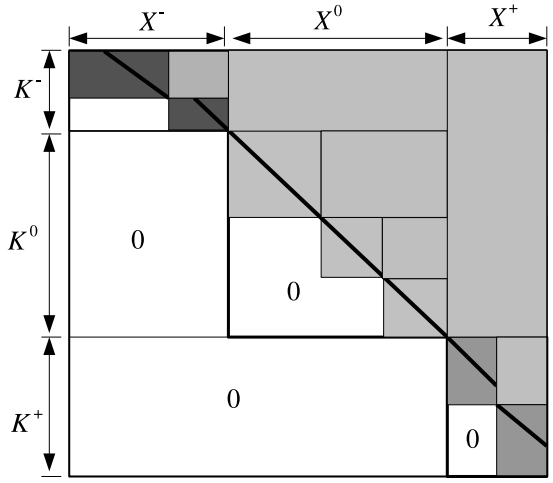


FIG. 2.10 – Décomposition Dulmage-Mendelsohn d'une représentation structurelle

Dans cette figure, nous remarquons qu'il y a trois sous systèmes :

$$\begin{aligned} S^+ &= (K^+, X^+) \\ S^0 &= (K^0, X^+ \cup X^0) \\ S^- &= (K^-, X^+ \cup X^0 \cup X^-) \end{aligned} \tag{2.4.9}$$

tel que

- $(K^+, K^0, K^-)$ est la partition des contraintes du système $K$
- $(X^+, X^0, X^-)$ est la partition des variables inconnuess du système $X$
- $(K^+, X^+)$ sur-déterminé ou (sur-contraint)
- $(K^0, X^0)$ juste-déterminé ou (juste-contraint)
- $(K^-, X^-)$ sous-déterminé ou (sous-contraint)

TAB. 2.3 – Matrice structurelle

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ |
|---|---|---|---|---|---|---|
| $k_1$ | 1 | 0 | 0 | 0 | 0 | 1 |
| $k_2$ | 0 | 1 | 0 | 0 | 0 | 1 |
| $k_3$ | 0 | 0 | 1 | 1 | 1 | 1 |
| $k_4$ | 1 | 0 | 0 | 1 | 0 | 0 |
| $k_5$ | 0 | 1 | 0 | 0 | 1 | 0 |
| $k_6$ | 0 | 0 | 1 | 0 | 0 | 1 |
| $k_7$ | 0 | 0 | 0 | 0 | 0 | 1 |

- $S^+$ est nommé un bloc sur-déterminé. Il est caractérisé par $|K^+| > |X^+|$. Dans $S^+$, il peut y avoir plusieurs sous blocs sur-déterminés ou juste-déterminés (pas de sous-blocs sous-déterminés). Mais au moins un sous-bloc sur-déterminé doit exister sinon $S^+$ sera juste-déterminé.
- $S^0$ est nommé un bloc juste-déterminé. Il est caractérisé par $|K^0| = |X^0|$. Dans $S^0$, il peut avoir plusieurs sous-bloc juste-déterminés (pas de sous-bloc sous-déterminés ou sur-déterminés).
- $S^-$ est nommé un sous-ensemble sous-déterminé. Il est caractérisé par $|X^-| > |K^-|$. Dans $S^-$, il peut avoir plusieurs sous-ensembles sous-déterminés (pas de sous-ensembles juste-déterminés ou sur-déterminés).

Cette décomposition représente un outil très important pour la conception des sous systèmes testables **SST**s (Cassar and Staroswiecki [1997], Krysander et al. [2008]) et pour la conception des placements de capteurs (Frisk and Krysander [2007]). Considérons un système représenté par la matrice structurelle 2.3.

Ce système contient six sous-blocs, un sous-bloc sur-déterminés ($\{k_3, k_5\}$, $\{x_2, x_3, x_4, x_5, x_6\}$) et quatre sous-blocs juste-déterminés ($\{k_4\}, \{x_1, x_4\}$), ($\{k_6\}, \{x_3, x_6\}$), ($\{k_2\}, \{x_2, x_6\}$), ($\{k_1\}$), $\{x_1, x_6\}$) et ($\{k_7\}, \{x_6\}$). Le décomposition DM est présenté dans la matrice structurelle 2.4.

### 2.4.3 Représentation structurelle par bond graph (graphe de liaison ou graphe à liens)

Le bond graph (**BG**) est un outil de modélisation graphique qui peut être appliqué à tous les domaines des sciences. Il s'agit d'une approche structurelle de la modélisation des systèmes. L'outil bond-graph (ou graphe de liaison), défini par Paynter [1961] et formalisé par Thoma [1975], est considéré comme intermédiaire entre le système physique et les équations mathématiques décrivant le comportement de ce système.

TAB. 2.4 – décomposition Dulmage-Mendelshon

| | $x_5$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $x_6$ |
|---|---|---|---|---|---|---|
| $k_3$ | 1 | 1 | 1 | 0 | 0 | 1 |
| $k_5$ | 1 | 0 | 0 | 1 | 0 | 0 |
| $k_4$ | 0 | 1 | 0 | 0 | 1 | 0 |
| $k_6$ | 0 | 0 | 1 | 0 | 0 | 1 |
| $k_2$ | 0 | 0 | 0 | 1 | 0 | 1 |
| $k_1$ | 0 | 0 | 0 | 0 | 1 | 1 |
| $k_7$ | 0 | 0 | 0 | 0 | 0 | 1 |

L'approche **BG** (Borne et al. [1992], Vergé and Jaume [2004]) est basée essentiellement sur la caractérisation des phénomènes d'échange d'énergie au sein de système. L'énergie est un concept important pour décrire l'évolution des systèmes technologiques. La méthodologie **BG** permet de traiter les chaînes d'énergie et d'information et elle peut être appliquée sur tous les systèmes dans tous les domaines (linéaire, non linéaire, continue, discrète, numérique, électronique, mécanique, hydraulique...). La démarche se décompose en plusieurs étapes :

- la première étape consiste à étudier l'architecture du système, soit l'interconnexion des composants, soit le couplage des phénomènes physiques retenus, et à la reproduire graphiquement avec un langage unique pour tous les domaines de la physique
- la deuxième étape consiste à décrire les lois consécutives des composants ou des phénomènes, linéaires ou non linéaires. Pour cela la notion de causalité est un atout majeur de cette technique.

Le bond-graph obtenu peut facilement évoluer en ajoutant simplement d'autres éléments bond-graph, sans recommencer la démarche du début. De plus, grâce à son caractère graphique et à sa structure causale, le modèle bond-graph représente un excellent outil pour analyser structurellement un système.

La modélisation par bond graph des systèmes à structure rigide (mécanique, électrique) a connu d'importants développements. Par contre, la modélisation des systèmes en génie énergétique restent encore un domaine ouvert en raison de la complexité de ces phénomènes. Ce sont pourtant ces types de processus, présents dans le monde industriel à grand risque qui nécessitent, pour leur contrôle, des modèles de plus en plus précis et exploitables.

Le bond graph est un graphe orienté qui fait apparaître des variables dynamiques, qui traduisent des transferts de puissance entre systèmes. Considérons le système mécanique représenté dans la figure 2.11. Dans ce système, il y a une liaison physique entre les deux éléments $A$ et $B$ par l'intermédiaire d'un barre. Dans ce système il y a conservation de l'énergie et continuité de puissance. Le flux d'énergie entre $A$ et $B$ est représenté par un

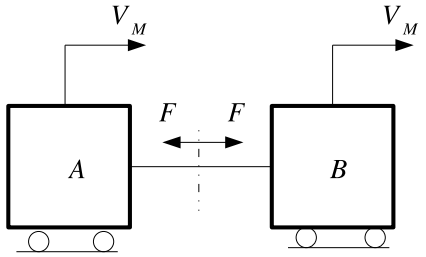


FIG. 2.11 – Système mécanique

lien de puissance, caractérisé par le symbole suivant : qui correspond au lien du bond

graph. La puissance échangée entre $A$ et $B$ est donnée par $P = F_M V_M$ où l'action d'une force $F_M$ représente l'effort et la vitesse $V_M$ représente le flux. La schéma physique de la figure 2.11 est traduit par la figure 2.12.

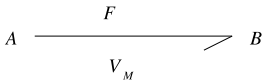


FIG. 2.12 – Transfert de puissance

Après avoir défini le transfert de puissance, nous montrerons le bond graph du système de deux bacs qui est plus compliqué que le système présenté par la figure 2.11. Le bond graph correspondant à ce système est représenté par la figure 2.13. Cette représentation structurelle a été utilisée dans (Ould Bouamama et al. [2005], Samantaray et al. [2006]) pour la détection et la localisation de défauts et dans (Khemliche et al. [2006]) pour trouver le placement de capteurs.

## 2.5 Comparatif entre les représentations structurelles présentées

Dans ce chapitre, nous avons discuté des représentations structurelles introduites précédemment afin d'identifier celles qui seront les plus pertinentes dans les chapitres suivants. Ces représentations sont utilisées par plusieurs auteurs.

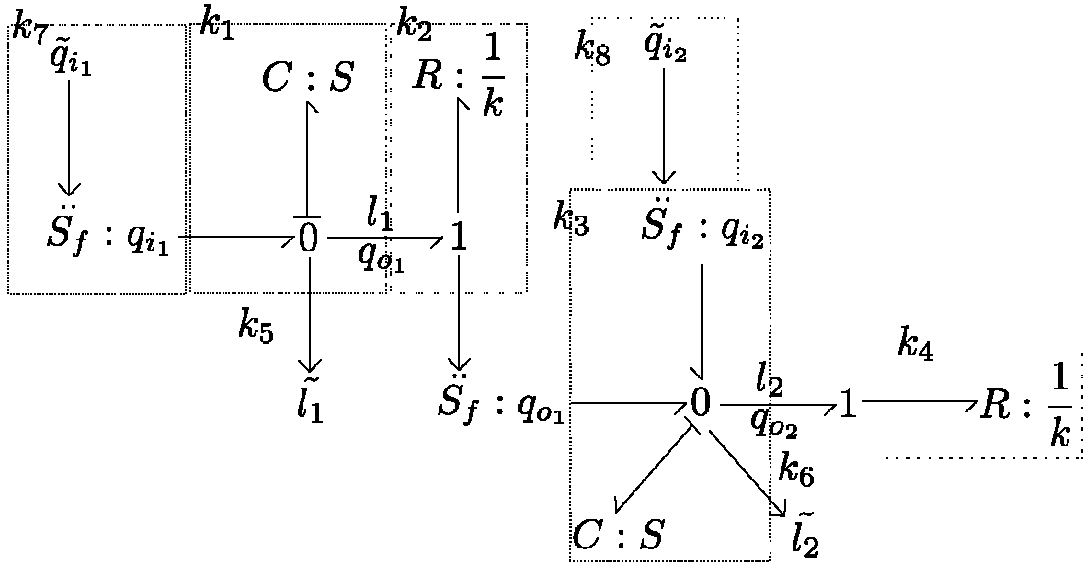


FIG. 2.13 – Modèle bond graph du système de deux bacs

La représentation par matrice structurelle permet de représenter la structure d'un système en définissant deux sous-ensembles : les colonnes représentant les contraintes $K$ modélisant les composants du système et les lignes représentant les variables $V$ apparaissant dans ces contraintes. Cette représentation traite les systèmes constitués d'un ensemble de composants modélisés par un ensemble de contraintes (un composant peut être modélisé par une ou plusieurs contraintes). Par conséquent, elle permet de déterminer le(les) composant(s) défectueux.

La représentation par digraphe associé à la représentation d'état est une abstraction structurelle du comportement du système où les arcs peuvent être interprétés comme des "influences mutuelles" entre des variables. Cette représentation peut être appliquée aux systèmes représentés par les équations d'état afin de diagnostiquer les défauts dans ces systèmes. L'inconvénient de cette représentation est qu'elle ne présente pas les contraintes algébriques du système à diagnostiquer. Il est difficile d'établir un lien avec les composants physiques. De plus, la représentation par digraphe associé à la représentation d'état impose un choix d'entrées et de sorties. Fixer les entrées et les sorties signifie qu'un point de vue particulier est adopté car une variable n'est pas par essence entrée ou sortie. Ce choix d'entrées et de sorties est nécessaire pour trouver une représentation d'état du système à diagnostiquer et par suite pour construire le digraphe. Or, ce choix est réducteur parce qu'il représente un point de vue particulier qui ne permet de trouver qu'une partie des tests. Trouver tous les tests exige de trouver toutes les combinaisons d'entrées/sorties possibles c'est-à-dire tous les digraphes correspondants.
Reprenons l'exemple des deux bacs déjà mentionnés dans le paragraphe A et supposons que la variable d'état est $l_2$, les variables d'entrée sont $(l_1, q_{i_2})$ et la variable de sortie est

$l_2$, une autre représentation d'état peut être obtenue :

$\frac{d}{dt}[l_2] = [-\frac{k}{S}][l_2] + [\frac{k}{S}, \frac{1}{S}] \begin{bmatrix} \tilde{l_1} \\ \tilde{q_{i_2}} \end{bmatrix}$

$[\tilde{l_2}] = [1][l_2]$

Nous remarquons que cette représentation est différente de la représentation présentée dans le paragraphe A. Par conséquent, le digraphe associé à cette représentation d'état sera différent du digraphe présenté par la figure 2.3. Cette différence est due aux différents choix d'entrée et de sortie.

La représentation par graphe biparti telle que celle utilisée par Blanke et al. [2006] a les même caractéristiques que celles de la représentation par matrice structurelle. Dans la représentation par graphe biparti, il n' y pas d'ordonnancement à priori.

Les trois représentations précédentes (matrice structurelle, digraphe et graphe biparti) sont équivalentes mais elles sont utilisées par différents auteurs. Ces représentations sont intéressantes dans le cas où les contraintes et les variables sont disponibles. Dans le cas contraire, l'utilisateur doit les générer en utilisant des heuristiques spécifiques. Pour les systèmes complexes, générer les contraintes du processus n'est pas une tâche facile (par exemple les système thermiques). La représentation par bond graph est un outil à considérer car il introduit une heuristique pour la construction d'un modèle.

L'intérêt du bond graph est de décrire de manière graphique tous les échanges énergétiques. Cette approche est intrinsèquement multidisciplinaire. Les modèles développés sont implicitement dynamiques et permettent de conserver un lien avec les phénomènes physiques. Les difficultés principales sont que la description, les échanges et les couplages énergétiques ne sont pas toujours simples à établir. De plus, cette représentation ne s'applique pas aux systèmes à paramètres distribués et aux systèmes discrets. La grande difficulté d'utilisation de cette représentation réside dans la connaissance de la physique. La représentation par bond graph apporte en plus de la modélisation structurelle une modélisation comportementale et une sémantique qui importe peu au diagnostic et qui demmande un grand effort de modélisation. l'interêt d'ordre sémantique est la vérification de la cohérence énergétique mais dans certaines situations (capteur, actionneur, découplage du système), il n'y a pas de cohérence et cela réduit l'interêt sémantique du bond graph. En diagnostic, l'aspect sémantique n'est pas primordiale. Comme dans un graphe biparti et une matrice structurelle, les contraintes et les variables apparaissent dans le bond grah mais il y a une dimension sémantique supplémentaire qui n'est pas primordiale et cette dimension sémantique pose des problèmes lorsqu'il n'y pas de continuité énergétique. Ce cas est très présent dans le diagnostic (capteurs, actionneurs) ou dans le cas où le système est décomposé de quelques parties séparées physiquement (deux bacs).

La différence entre ces représentations structurelles est représentée par le tableau

suivant :

| Représentation par | Adapté à | Causalité à priori |
|---|---|---|
| Matrice structurelle | représentation sous forme de contraintes, ou sous forme d'éléments connectés:<br>•permet de combiner les contraintes<br>•permet conserver les liens entre les contraintes et les composants. | possible |
| Digraphe | représentation avec orientation:<br>•ordonnancement a priori, fixer un ensemble d'entrées et de sorties ne permet pas de trouver tous les RRA possibles.<br>•ne permet pas de conserver les liens entre les contraintes et les composants. | oui |
| Graphe biparti | représentation sous forme de contraintes, ou sous forme d'éléments connectés:<br>•permet de combiner les contraintes et pas d'ordonnancement à priori.<br>•permet de conserver les liens entre les contraintes et les composants. | Possible si digraphe |
| Bond graph | la représentation de contraintes explicites (équations) ou implicites:<br>•permet de vérifier de la cohérence énergétique.<br>•ne permet pas de vérifier la cohérence énergétique pour les capteurs, actionneurs...<br>•ajoute une dimension sémantique aux contraintes (dimension pas nécessaire en diagnostic). | possible |

Les représentations structurelles de type matrice structurelle et par graphe biparti, très proches, nous semblent les plus adaptées aux problématiques du diagnostic qu'il s'agisse de générer les relations de redondance analytique ou de trouver un placement de capteurs satisfaisant des cahiers de charges. Néanmoins, ces représentations doivent être adaptées car les concepts importants du diagnostic n'apparaissent pas toujours explicitement. Les notions de variables inconnues et de variables connues ne sont pas précises et les notions de contraintes, de **SST** et de composants sont plus ou moins confondues. Dans la section suivante, nous présentons une représentation structurelle spécialisée pour les problématique de diagnostic, permettant d'appréhender la diversité des modèles et qui utilise des notions plus précises. Cette représentation sera la base de notre travail.

## 2.6 Modélisation structurelle utilisée

La connaissance comportementale commence par les phénomènes. Un phénomène est un élément d'information potentiellement observable sur l'état réel d'un système. Il est

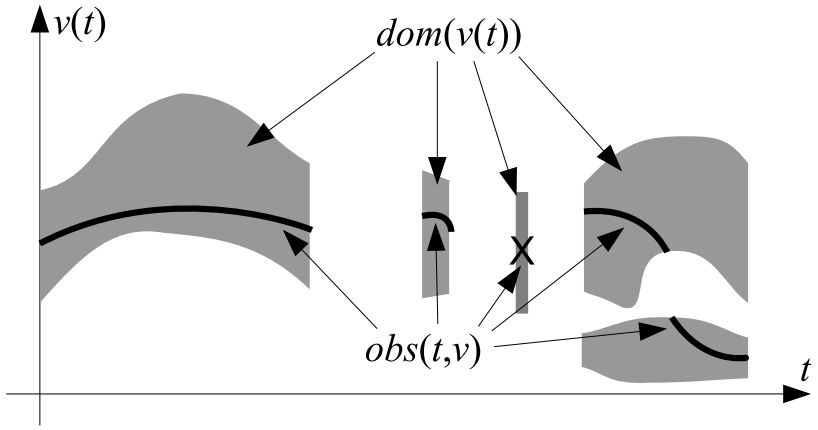


FIG. 2.14 – Un tube modélisant une variable et une observation relative

modélisé par une variable (appelée variable inconnue dans (Blanke et al. [2003], Krysander and Nyberg [2005], Travé-Massuyès et al. [2006]) implicitement temps-variant, qui doit être distinguée d'un paramètre qui dépend du modèle. D'une manière générale, même si un phénomène est observable, il n'est pas possible de le fusionner avec des données relevées parce que, dans le diagnostic de défauts, les données sont connues seulement à condition que des actionneurs ou des capteurs se comportent correctement. Les phénomènes $V(t) = \{\ldots, v_i(t), \ldots\}$ sont modélisés par un espace phénoménologique $F(T, V) = \{V(t); t \in T\}$ où $T$ représente l'ensemble de temps continu ou discret. À tout instant donné $t$ dans $T$, ces phénomènes appartiennent à un domaine $dom(t, V) = dom(V(t))$ représentant toutes les valeurs possibles que peuvent avoir les phénomènes. Par conséquent, en considérant tout $t \in T$, $\{dom(V(t)); t \in T\}$ représente un tube dans l'espace phénoménologique $F(T, V)$.
Tous les phénomènes sont considérés comme inconnus parce que les phénomènes observables ne sont pas des observations. Introduisons le concept de *flot de données* (appelée variable connue dans (Blanke et al. [2003], Krysander and Nyberg [2005], Travé-Massuyès et al. [2006]) pour modéliser les données réelles récoltées d'un système. Un *flot de données* modélise les données fournis par une source d'information concernant un phénomène. Un *flot de données* concernant un phénomène $v$ est noté $val(t, v)$ avec $val(t, v) \in dom(v(t))$. Il correspond à une trajectoire appartenant au tube $\{dom(v(t)); t \in T\}$ (voir figure 2.14).

Quand l'information sur un phénomène $v$ provient de différentes sources, les différents *flots de données* peuvent être dénotés : $val_i(t, v)$. Formellement, des *flots de données* fournis par un composant $c$ peuvent être liés à un phénomène : $ok(c) \longrightarrow \forall t \in T, val(t, v) = v$, qui signifie que si le composant nommé $c$ est dans le mode normal $ok(c)$ alors le *flot de données* $val(t, v)$ correspond à la valeur réelle du phénomène $v$ à tout instant $t \in T$.
Dans le diagnostic de défauts, un système n'est pas supposé rester dans un mode

donné. En effet, l'analyse diagnostique vise à rechercher les modes comportementaux des composants d'un système. Au moins deux modes sont définis : le mode $ok(c)$, qui correspond au comportement normal prévu et le mode $cfm$, qui est le mode complémentaire de défauts : il fait référence à tous les comportements qui ne s'adaptent pas au comportement normal prévu. Parfois, des modes spécifiques de défauts peuvent être modélisés (Struss [1992], De Kleer and Williams [1992]). Ils sont dénotés par une étiquette spécifique, par exemple, le mode $fuite$. Considérant par exemple un tuyau où $ok$ et $fuite$ sont modélisés. Cela donne : $Mode(tuyau) = \{ok, fuite, cfm\}$ où $cfm(tuyau)$ fait référence au comportement qui ne correspond pas à $ok(c)$ ou à $fuite(tuyau)$. Excepté le mode complémentaire de défaut, les modes sont modélisés par des relations cause-effet entre les phénomènes, qui sont représentés par des contraintes. Chaque contrainte fait référence à un ensemble de fonctions contenant les variables inconnues et les *flots de données* connus.
D'une manière générale, une fonction sur $dom(t, V)$ est définie d'un sous-espace $dom(t, V_1)$ à un autre sous-espace $dom(t, V_2)$, où $\{V_1, V_2\}$ est une partition de $V$. Notons que plusieurs fonctions $\kappa_i$ peuvent modéliser la même contrainte $k$. Si $\kappa_i : dom(t, V_1) \longrightarrow dom(t, V_2)$ est une fonction représentant la contrainte $k$ qui modélise, par exemple un composant $c_1$ dans le mode $mode_1$ et un composant $c_2$ dans le mode $mode_2$, cela donne :

$$\begin{aligned} mode_1(c_1) \wedge mode_2(c_2) \rightarrow \\ V_2 = \kappa_i(t, V_1, val(V_3)); \\ V_1 \in dom(t, V_1), V_2 \in dom(t, V_2) \end{aligned} \tag{2.6.1}$$

tel que le *flot de données* $val(V_3)$ est considéré comme inclus dans la fonction. Une *contrainte* n'est pas strictement équivalente à une $fonction$. Une contrainte correspond à un ensemble de fonctions équivalentes.
Premièrement, bien que des vecteurs de fonctions pourraient être utilisés, elles sont difficiles à gérer. Il vaut mieux les diviser en fonctions élémentaires. Dans la suite, les fonctions modélisant une contrainte $k$ sont nommées les réalisations de $k$. Supposons que $\kappa_i$ est une réalisation de $V \setminus \{v\}$ vers $\{v\}$. Il peut y avoir des réalisations équivalentes définies sur V qui modélisent également la contrainte. Donc, la notion de *contrainte* peut être prolongée pour représenter toutes les réalisations équivalentes représentant un sous-ensemble donné de $dom(V)$.
Prenons par exemple les équations suivantes modélisant un *xor* logique :

$$x_1 = x_2 \oplus x_3$$
$$x_2 = x_1 \oplus x_3$$
$$x_3 = x_1 \oplus x_2$$

Ces équations sont 3 réalisations différentes d'une même contrainte. D'un point de vue structurel, n'importe quella variable se déduit des autres.
Prenons l'équation suivante : $x_1 = x_2 \vee x_3$ modélisant un *or* logique. Cette équation est la seule réalisation de la contrainte. D'un point de vue structurel, seule la variable $x_3$ se déduit des autres variables.
Dans la suite, une contrainte $k$ sera comprise comme un ensemble de réalisations équivalentes. Elle est résumée par un ensemble des variables apparaissant dans les réalisations : $var(k)$ dont une partie est déductible, s'il existe une réalisation correspondante.
Pour résumer, un système $\Sigma$ est composé d'un ensemble de contraintes $K_\Sigma$ et un ensemble de modes comportementaux $Modes(\Sigma)$ liés aux composants dans $\Sigma$. $var(K_\Sigma)$ est l'ensemble des variables, nommées "ports" dans (Chittaro and Ranon [2004]), qui modélisent les phénomènes observables impliqués dans $\Sigma$. En effet, par prolongation, l'ensemble de variables apparaissant dans un ensemble de contraintes $K$ est dénoté $var(K) = \bigcup_{k \in K} var(k)$. Chaque contrainte $k \in K_\Sigma$ est liée à un mode $m \in Modes(\Sigma)$ par une relation du premier ordre : $m \rightarrow k$.

La notion de *structure de contrainte* est introduite : en général, la structure d'une contrainte $k$ représente les variables $var(k)$ en les partitionnant entre les variables auxquelles correspond une réalisation : elles sont déductibles, et les autres sont non déductibles. Cette notion représente toutes les réalisations existantes correspondant à une contrainte donnée $k$. Par exemple, considérons une réalisation $\kappa$ modélisant un *xor* logique : $x_3 = x_1 \oplus x_2$. Il y a une fonction menant à $x_3$ de $x_1$ et $x_2$ mais il y a aussi une autre fonction menant à $x_1$ de $x_2$ et $x_3$ et une autre menant à $x_2$ de $x_1$ et $x_3$. Dans ce cas, ces variables peuvent être qualifiées comme variables déductibles. Elles sont dénotées : $var(\kappa) = var^+(\kappa) = \{x_1, x_2, x_3\}$. La structure de contrainte $\kappa$ est différente de la structure de contrainte $\kappa^{'}$ modélisant un *or* logique : $x_3 = x_1 \vee x_2$ où seulement $x_3$ est déductible. Elle est notée : $var(\kappa^{'}) = var^+(\kappa) \cup var^-(\kappa)$ avec $var^+(\kappa) = \{x_3\}$ et $var^-(\kappa) = \{x_1, x_2\}$. L'ensemble des réalisations équivalentes modélisant une contrainte $k$ est noté $\mathcal{K}(k)$. La structure d'une contrainte $k$ modélise en même temps les variables de son espace phénoménologique et l'ensemble des réalisations possibles modélisant cette contrainte. Une structure peut représenter une réalisation particulière ou un ensemble de réalisations équivalentes.

Les variables $V = var(k)$ peuvent être décomposées en un ensemble de variables déductibles et un ensemble de variables non déductibles. Une structure $s$ associée à une contrainte $k$ sera écrite $\llcorner V^-, V^+ \lrcorner$ où $V^-$ et $V^+$ satisfont $V^- \cap V^+ = \emptyset$ et $var(k) = V^- \cup V^+$. Donc, la structure modélisant une contrainte $k$ est notée : $s(k) = \llcorner V^-, V^+ \lrcorner$. $\forall v \in V^+$, il existe une réalisation de $\mathcal{K}(k)$ menant à $v$ : $\forall v \in V^+ \Longleftrightarrow \exists \llcorner (V^-, V^+) \backslash \{v\}, \{v\} \lrcorner$

Pour simplifier, les notations suivantes sont adoptées : $\llcorner \emptyset, V^+ \lrcorner = \llcorner V^+ \lrcorner$ et si $S$ est un ensemble de structures, $var(S) = \bigcup_{s \in S} var(s)$. Finalement, une structure vide $s$ est une structure satisfaisant : $var(S) = \emptyset$. Il est noté : $s = \llcorner \emptyset \lrcorner$.

Cette représentation structurelle sera utilisée dans les chapitres 4 et 6 pour concevoir d'autres méthodes pour le placement de capteurs et la conception de tests.

## 2.7 Conclusion

Dans cette section, nous avons étudié les représentations structurelles des systèmes physiques en vue de leur utilisation dans le contexte du diagnostic de défauts (bond graphe, matrice structurelle et graphe). Nous avons présenté également quelques outils importants utilisés dans la représentation par graphe pour le diagnostic de défauts (couplage, décomposition de Dulmage-Mendelsohn). Ces outils seront utilisés pour analyser les propriétés des systèmes qui pourront être utilisées pour la conception des relations de redondance analytique ou le placement de capteurs.
À la fin de ce chapitre, nous avons présenté une représentation structurelle générale permettant d'appréhender la diversité des modèles et elle utilise des notations plus précises. Cette représentation sera utilisée dans cette thèse pour concevoir des nouvelles méthodes de génération des tests et du placement de capteurs pour le diagnostic.

# Deuxième partie

# Conception de RRAs

# Chapitre 3

# Etat de l'art des méthodes de conception de RRAs pour le diagnostic des défauts

La plupart des méthodes de diagnostic de défauts se décomposent en deux étapes : détection et localisation de défauts. Détecter les défauts exige généralement de disposer de **RRAs**. La procédure de conception de **RRAs** s'appuie sur des techniques d'élimination de variables dans le système à diagnostiquer. Une fois les **RRAs** conçues, la procédure de détection de défauts vérifie à chaque moment si ces **RRAs** sont satisfaites ou pas. Si ces **RRAs** ne sont pas satisfaites, la procédure de localisation de défauts identifie les composants du système qui peuvent être défectueux. Afin que la procédure de diagnostic fonctionne correctement, les **RRAs** doivent représenter au mieux l'ensemble des comportements possibles du système.

Dans ce chapitre, nous nous concentrerons sur les méthodes déjà proposées pour la conception de relations de redondance analytique après avoir présenté quelques définitions.

**Définition 3.0.1.** *Une contrainte terminale $k$ est une contrainte qui satisfait : $card(var(k)) = 1$ où $var(k)$ est l'ensemble de variables apparaissant dans la contrainte $k$.*

**Définition 3.0.2.** *Supposons que $K$ soit un ensemble de contraintes, $var(K)$ est l'ensemble des variables apparaissant dans $K$ et $v$ une variable dans $var(K)$ caractérisée par son domaine $dom(v)$. $K$ est un ensemble de contraintes solution pour $v$ si en utilisant $K$, il est possible de trouver un ensemble de valeurs $S$ pour $v$ tel que $S \subset dom(v)$. Un ensemble de contraintes solution pour $v$ est minimal si il n'existe pas de sous ensemble de $K$ qui soit également un ensemble de contraintes solution pour $v$. Un ensemble minimal de contraintes solution $K$ pour $v$ est noté : $K \vdash v$.*

**Définition 3.0.3.** *Supposons que $K$ soit un ensemble de contraintes. $K$ est testable si et seulement si il existe deux sous ensembles distincts $K_1 \subset K$, $K_2 \subset K$ tel que $K_1 \nsubseteq K_2$ et $K_2 \nsubseteq K_1$, et une variable $v \in var(K)$ tel que $K_1 \vdash v$ et $K_2 \vdash v$. Si cette propriété est satisfaite, il est en effet possible de vérifier si l'ensemble de valeurs $S_1$ déduit de $K_1$ est consistant avec l'ensemble de valeurs $S_2$ déduit de $K_2$ : $S_1 \cap S_2 \neq \emptyset$.*

Ajouter n'importe quelle contrainte à un ensemble de contraintes testable mène également à un ensemble de contraintes testable. Néanmoins, la contrainte ajoutée n'est pas nécessaire : seuls les ensembles testables minimaux sont intéressants.

**Définition 3.0.4.** *Un ensemble de contraintes testable est minimal s'il n'est pas possible de retirer une contrainte sans perdre la testabilité.*

Une contrainte globale qui peut être déduite d'un ensemble testable est nommée une relation de redondance analytique **RRA**. Les contraintes de cet ensemble testable sont le support de la **RRA** résultant.

## 3.1 Méthodes existantes pour la conception de RRAs

Dans la communautés **bridge**, la conception des relations de redondance analytique **RRAs** a été l'objet de nombreuses publications. Ce problème de conception a inspiré beaucoup de chercheurs afin d'améliorer les méthodes existantes. Toutes les méthodes de détection et de localisation des défauts sont basées sur l'utilisation de redondance. D'une manière générale, les procédures de génération des relations de redondance sont basées sur l' élimination de variables et par conséquent, nous obtenons des relations qui ne contiennent que les flots de données.
Dans cette section, nous rappellerons quelques méthodes déjà proposées qui permettent de générer les relations de redondance analytique.

### 3.1.1 Méthode basée sur la recherche de couplages dans un graphe biparti

(Blanke et al. [2003], Blanke et al. [2006]) ont proposé une méthode pour générer les relations de redondance analytique **RRAs**. Cette méthode est basée sur l'élimination des variables sur l'ensemble de contraintes du système. Elle consiste à représenter le système structurellement par un graphe biparti $G = (K, V, A)$ où $K$ est un ensemble des contraintes du système, $V$ est l'ensemble des variables $X$ plus les flots de données $O$ et $A$ est l'ensemble des arcs liant $K$ à $V$, et ensuite chercher les couplages complets par

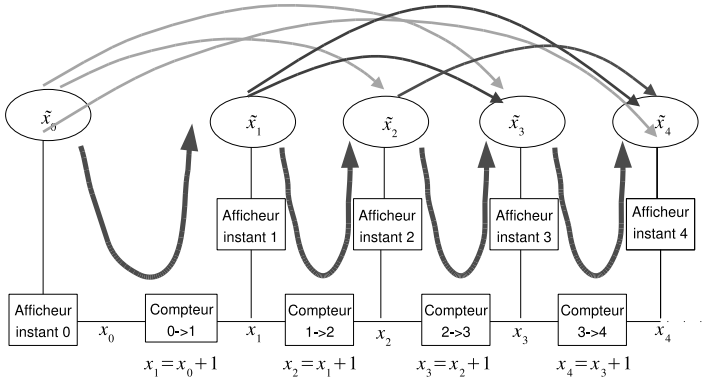


FIG. 3.1 – Système du compteur digital

rapport aux variables $X$ tel que $X \subset V$.
Les relations de redondance sont des sous-graphes du graphe biparti $G$, qui sont associées à des couplages complets par rapport aux variables $X$. Ces relations de redondance sont composées des chaînes alternées, qui commencent par des flots de données et qui finissent par des contraintes non couplées. Ces relations de redondance permettent de concevoir les relations de redondance analytique. Un algorithme permettant de trouver les couplages complets est présenté dans (Blanke et al. [2006]).
Afin de mettre en évidence cette approche, appliquons-la sur un système de compteur digital représenté sur la figure 3.1, qui peut être décomposé en un afficheur et une carte électronique. Pour être capable de déterminer l'instant où un défaut se produit, nous choisissons de modéliser les opérations de comptage et d'affichage plutôt que les composants.

Les decriptions retenues de ce compteur, $\forall i \in \mathbb{N}$ :

$$\begin{aligned} &\text{afficheur} : ok(A_i) \longrightarrow x_i = \tilde{x_i} \\ &\text{compteur} : ok(C_{i \longrightarrow i+1}) \longrightarrow x_{i+1} = x_i + 1 \end{aligned} \tag{3.1.1}$$

où $A_i$ représente l'opération d'affichage á l'instant $i = 0, 1, 2....$ $C_{i \longrightarrow i+1}$ représente l'opération de comptage de l'instant $i$ à l'instant $i+1$ et $x_{i+1}$ correspond à la valeur sur la sortie du compteur. Le symbôle $ok(A_i)$ signifie que le composant $A_i$ se comporte normalement. Le modèle de comportement de 4 opérations de comptage du système de compteur digital est décrit par les contraintes suivantes :

$$\begin{aligned} k_1 &: x_1 = x_0 + 1 \\ k_2 &: x_2 = x_1 + 1 \\ k_3 &: x_3 = x_2 + 1 \\ k_4 &: x_4 = x_3 + 1 \end{aligned} \tag{3.1.2}$$

Le modèle d'observation (afficheurs) est décrit par les contraintes suivantes :

$$\begin{aligned} k_5 &: x_0 = \tilde{x_0} \\ k_6 &: x_1 = \tilde{x_1} \\ k_7 &: x_2 = \tilde{x_2} \\ k_8 &: x_3 = \tilde{x_3} \\ k_9 &: x_4 = \tilde{x_4} \end{aligned} \tag{3.1.3}$$

tel que $V = X \cup O$ où les variables $X$ sont : $(x_0, x_1, x_2, x_3, x_4)$ et les flots de données $O$ sont : $(\tilde{x}_0, \tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4)$. Ce système peut être représenté par la matrice d'incidence 3.1

TAB. 3.1 – La matrice structurelle du compteur

| | $x_0$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $\tilde{x}_0$ | $\tilde{x}_1$ | $\tilde{x}_2$ | $\tilde{x}_3$ | $\tilde{x}_4$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_2$ | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_3$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_4$ | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| $k_5$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_6$ | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $k_7$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $k_8$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| $k_9$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

Le graphe biparti $G = (K, V, A)$ correspondant à cette matrice d'incidence est représenté par le figure 3.2

Trouver toutes les relations de redondance analytique *RRAs* commence par chercher tous les couplages complets possibles par rapport à $X$ dans le graphe biparti $G' = (K, X, A')$ tel que $A'$ représente l'ensemble des arcs liant $K$ à $X$.
Prenons le couplage représenté dans la figure 3.3.a. Nous rappelons que les relations de redondance sont composées de chaînes alternées, qui commencent par des flots de données et qui finissent par des contraintes non couplées. Dans le couplage proposé, il y a quatre contraintes non couplées, par conséquent quatre relations de redondance peuvent être trouvées. Elle sont représentées par la figure 3.4.

Dans la figure 3.4, nous remarquons que les contraintes $k_5$, $k_6$ ,$k_7$ ,$k_8$ et $k_9$ sont couplées avec les variables $x_0$, $x_1$, $x_2$, $x_3$ et $x_4$ respectivement. Nous remarquons aussi que les contraintes $k_1$, $k_2$, $k_3$, $k_4$ ne sont pas couplées avec des variables, c'est pourquoi elles sont liées au zeéro (voir le paragraphe B.1). Cette figure montre comment les variables

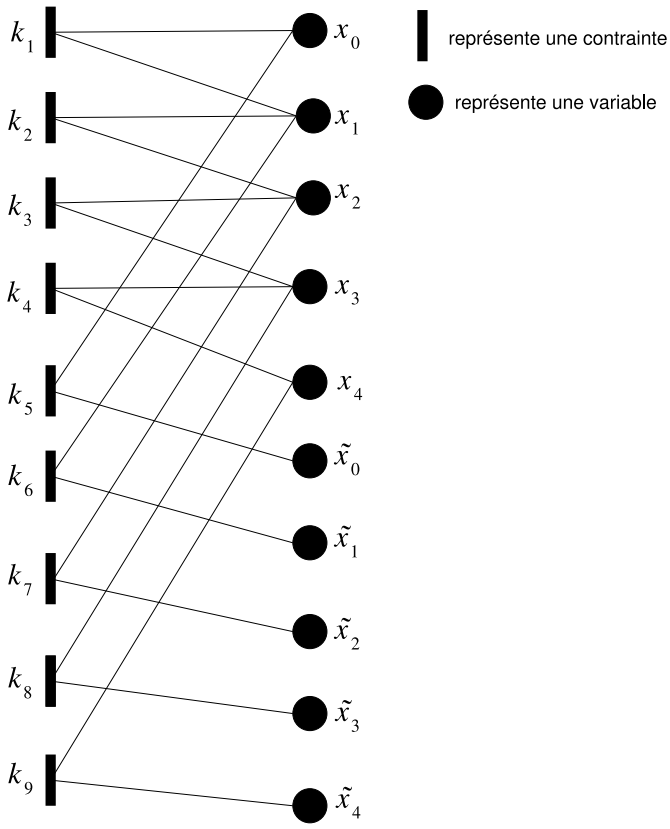


FIG. 3.2 – Graphe biparti du système de compteur digital

$(x_0, x_1, x_2, x_3, x_4)$ sont déterminées. Les contraintes non couplées $(k_1, k_2, k_3, k_4)$ sont utilisées pour construire les *RRAs*. Les contraintes utilisées pour construire la première *RRA* sont :

$$\begin{aligned} k_1 &: x_1 = x_0 + 1 \\ k_5 &: x_0 = \tilde{x_0} \\ k_6 &: x_1 = \tilde{x_1} \end{aligned} \tag{3.1.4}$$

Cet ensemble de contraintes est intitulé sous système testable **(SST)**. Il peut être utilisé pour obtenir la relation de redondance analytique suivante $(RRA_1) : \tilde{x_0} - \tilde{x_1} + 1 = 0$. Notons que $(x_0, x_1)$ sont des flots de données. Alors, cette équation peut être testée pour les valeurs données : $\tilde{x_0}$ et $\tilde{x_1}$.
De la même façon, nous obtenons les autres relations de redondance analytique :
$RRA_2 : \tilde{x_1} - \tilde{x_2} + 1 = 0$
$RRA_3 : \tilde{x_2} - \tilde{x_3} + 1 = 0$
$RRR_4 : \tilde{x_3} - \tilde{x_4} + 1 = 0$
La relation de redondance analytique $RRA_2$ est obtenue de l'ensemble de contraintes testables $\{k_2, k_6, k_7\}$, $RRA_3$ est obtenue de l'ensemble de contraintes testables $\{k_3, k_7, k_8\}$

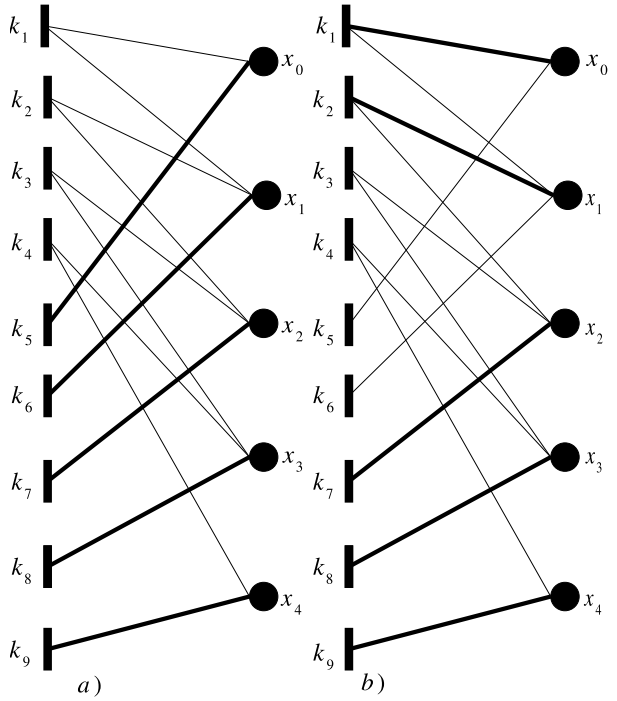


FIG. 3.3 – Deux couplages du système de compteur digital

et $RRA_4$ est obtenue de l'ensemble de contraintes testables $\{k_4, k_8, k_9\}$.
Notons que nous avons obtenus quatre *RRAs* avec un seul couplage proposé.
Prenons un autre couplage représenté dans la figure 3.3.b. Nous obtenons les quatre sous systèmes testables suivants :
$\{k_1, k_2, k_5, k_7\}$, $\{k_2, k_6, k_7\}$, $\{k_3, k_7, k_8\}$ et $\{k_4, k_8, k_9\}$.
Ces sous-systèmes testables conduisent aux relations de redondance analytique suivantes.
$RRA_1 : \tilde{x_0} - \tilde{x_2} + 2 = 0$
$RRA_2 : \tilde{x_1} - \tilde{x_2} + 1 = 0$
$RRA_3 : \tilde{x_2} - \tilde{x_3} + 1 = 0$
$RRR_4 : \tilde{x_3} - \tilde{x_4} + 1 = 0$
Dans ce couplage, nous avons obtenu quatre **RRAs**, une d'entre elles ($RRA_1$) est nouvelle alors que les 3 autres sont les mêmes que nous avons obtenues avec le couplage 3.3.a.
Par conséquent, trouver toutes les *RRAs* du système exige de trouver tous les couplages possibles.

Concernant cette méthode, nous avons remarqué que certaines **RRA** sont trouvées de nombreuses fois : il y a donc des redondances dans les calculs. Le nombre de redondances résultant dépend du nombre de couplages possibles. Ceci représente l'inconvénient de cette méthode. Dans l'exemple précédent, un couplage conduit à construire quatre **RRAs**. Trouver toutes les **RRAs** du système exige de chercher tous les couplages possibles : 62 couplages sont possibles ici. Pour chaque couplage, le nombre

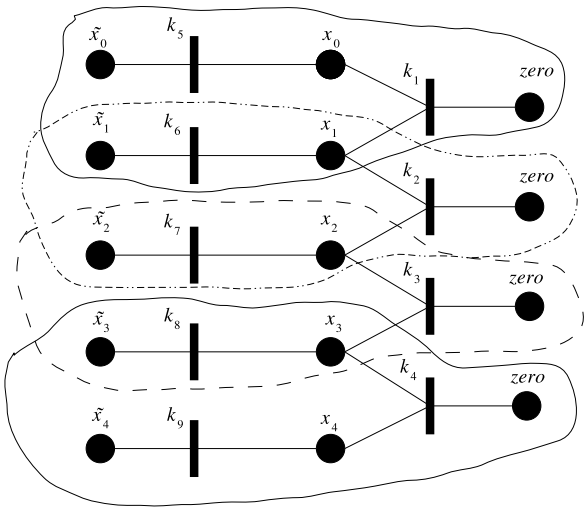


FIG. 3.4 – Les RRAs engendrées par le couplage présenté par la figure 3.3.a

de **RRAs** possibles qui peuvent être obtenus est égale à 4. Or, pour les 10 **RRAs** possibles par ce système (voir la figure 3.1), $62 \times 4$ **RRAs** sont obtenues.
Cette approche du graphe biparti induit une grande complexité même pour des systèmes simples.

### 3.1.2 Méthode basée sur la propagation de valeurs

Les algorithmes de propagations comme la propagation de valeurs (Apt [2003]) ou la propagation de contraintes (Russell and Norvig [2003]) peuvent être considérés comme des solutions possibles pour concevoir les relations de redondance analytique. S'il est possible de propager une valeur, alors il est possible de pré-calculer les chemins de propagation. Un pré-calcul de chemin de propagation tel qu'à une même variable, deux valeurs issues de propagations différentes peuvent être affectées, correspond à un ensemble testable de contraintes qui mène à une relation de redondance analytique.
Reprenons l'exemple de compteur digital proposé dans la section 3.1.1. Les tests performants peuvent être réalisés en propageant les valeurs connues dans les contraintes. Les différentes manières de propager peuvent être représentées par des arbres commençant par des contraintes terminales c'est-à-dire contraintes modélisant des capteurs ou des actionneurs. La figure 3.5 illustre deux arbres de propagations complètes. L'un commence par la contrainte terminale $k_5$ et l'autre commence par la contrainte terminale $k_7$.

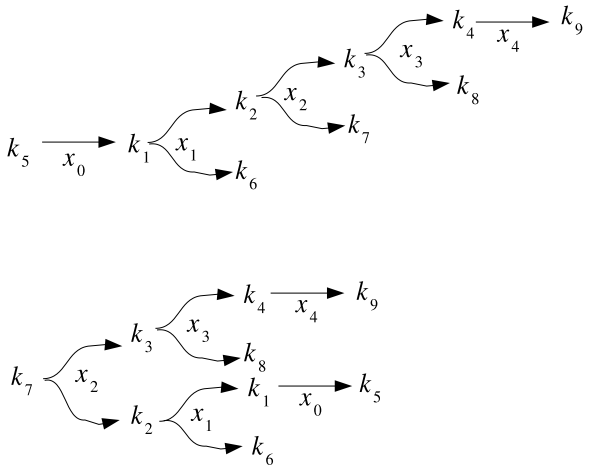


FIG. 3.5 – Deux manières différentes de propagations de valeurs

Dans la figure 3.5, nous remarquons qu' il y a 8 propagations élémentaires. Nous remarquons que l'arbre de propagation commençant par $k_5$ ne contient pas les tests partant de $k_7$ à $k_6$, de $k_7$ à $k_8$ et de $k_7$ à $k_9$, qui sont présents dans le deuxième arbre. Par contre, les tests partant de $k_5$ à $k_6$, $k_5$ à $k_8$, et de $k_5$ à $k_9$ ne sont pas contenus dans le deuxième arbre de propagation.
Les contraintes intervenant dans chaque propagation élémentaire composent les sous systèmes testables conduisant aux relations de redondance analytique. Prenons par exemple la propagation partant de $k_5$ à $k_6$, la variable $x_1$ peut être calculée en utilisant deux sous-ensembles différents de contraintes : $K_1 = \{k_6\}$ et $K_2 = \{k_1, k_5\}$. En utilisant le premier sous-ensemble $K_1$, la valeur $\tilde{x_1}$ peut être propagée à la variable $x_1$ :

$$x_1 = \tilde{x_1} \tag{3.1.5}$$

En utilisant le deuxième sous-ensemble de contraintes $K_2$, la valeur $\tilde{x_0}$ peut être propagée à la variable $x_0$, et puis la valeur résultant peut être propagée à la variable $x_1$ :

$$x_1 = \tilde{x_0} + 1 \tag{3.1.6}$$

Par conséquent, la variable $x_1$ peut être calculée de deux façons différentes et selon la définition 3.0.3, l'ensemble de contraintes $K = \{k_1, k_5, k_6\}$ est un sous système testable. En remplaçant la variable $x_1$ de l'équation 3.1.5 dans l'équation 3.1.6, nous obtenons la relation de redondance analytique suivante :
$\tilde{x_1} = \tilde{x_0} + 1$.
De la même façon, nous obtenons les autres relations de redondance analytique suivantes.
$RRA_2 : \tilde{x_0} - \tilde{x_2} + 2 = 0$.
$RRA_3 : \tilde{x_0} - \tilde{x_3} + 3 = 0$.
$RRA_4 : \tilde{x_0} - \tilde{x_4} + 4 = 0$.

$RRA_5 : \tilde{x_1} - \tilde{x_2} + 1 = 0.$
$RRA_6 : \tilde{x_2} - \tilde{x_3} + 1 = 0.$
$RRA_7 : \tilde{x_2} - \tilde{x_4} + 2 = 0.$

La matrice de signature de défauts est donnée par le tableau 3.2

TAB. 3.2 – La matrice signature de défauts du compteur

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $RRA_1$ | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| $RRA_2$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| $RRA_3$ | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| $RRA_4$ | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| $RRA_5$ | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| $RRA_6$ | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| $RRA_7$ | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

Concernant cette méthode, supposons que le nombre de contraintes terminales est égale à $n$. Chaque arbre de propagations commençant par une contrainte terminale conduit à $m_i$ propagations élémentaires. Ces propagations élémentaires sont utilisées pour construire des **RRA**s. Donc, trouver toutes **RRA**s exige de chercher tous les arbres de propagations possibles. Par conséquent, le degré de complexité est égal à $\sum_{i=1,2,...,n}(m_i)$. Dans l'exemple précédent, un arbre de propagation conduit à quatre propagations élémentaires permettant de construire quatre **RRA**s. Pour trouver les 10 **RRAs** possibles au système, 20 propagations élémentaires sont obtenues.
Nous remarquons que ne nombre de redondances résultant par cette méthode est inférieur au nombre de redondances résultant par la méthode basée sur la recherche de couplages. L'inconvénient de cette méthode est que la construction des propagations élémentaires n'est pas une tâche facile et la complexité de cette construction augmente dans les systèmes complexes.

### 3.1.3 Méthode basée sur le retrait consécutif de capteurs

(Travé-Massuyès et al. [2006]) ont proposé une méthode pour la conception de **RRAs**. L'idée principale de cette méthode est d'utiliser le modèle du comportement du système pour analyser d'une façon approfondie les redondances analytiques introduites par les capteurs hypothétiques. La conception des relations de redondance analytique hypothétique (**H-RRAs**) permet de construire une matrice de signature de défauts hypothétique (**HFS** matrice). Le point du départ de cette méthode est de faire l'hypothèse que toutes

les variables du système sont mesurées. On procède alors aux retraits consécutifs de capteurs. Cette méthode est la continuation des travaux proposés dans (Travé-Massuyès et al. [2001], Travé-Massuyès et al. [2003]).
Le travail effectué dans (Travé-Massuyès et al. [2001]) s'appuie sur une interprétation causale et, par conséquent, il n'y a pas de garantie que toutes les relations de redondance analytique (**RRAs**) peuvent être trouvées. Mais la méthode (Travé-Massuyès et al. [2006]) traite toutes les interprétations causales possibles des relations du modèle du système et par conséquent, toutes les **RRAs** peuvent être obtenues.
Le support d'un **RRA** (Support(RRA)) est composé d'un ensemble de composants et un ensemble de capteurs et actionneurs (un ensemble de contraintes élémentaires modélisant un ensemble de composants et un ensemble de contraintes modélisant un ensemble de capteurs et actionneurs).

Quelques définitions et propositions avec les démonstrations sont proposées dans (Travé-Massuyès et al. [2006]).

**Définition 3.1.1.** *Supposons que $\mathcal{A}$ est un ensemble de **RRAs**. Nous définissons $\mathcal{A}_b$ comme une base de $\mathcal{A}$ si toutes les **RRAs** de $\mathcal{A}$ peuvent être obtenues en combinant deux ou plusieurs **RRAs** de $\mathcal{A}_b$ et il n'existe pas $\mathcal{A}' \subset \mathcal{A}_b$ qui soit générateur de $\mathcal{A}$.*

**Proposition 3.1.1.** *Soit un système $\Sigma = (K, X \cup O)$ où $K$ est l'ensemble des contraintes du système et $V = X \cup O$, l'ensemble de variables du système $X$ et les flots de données $O$. Supposons que $M$ soit un couplage complet par rapport aux variables $X$, alors l'ensemble des **RRAs** résultant de ce couplage $(M)$ est une base pour les **RRAs** du système $\Sigma$.*

*Démonstration.* Une **RRA** est une contrainte déduite du modèle du système. Cette contrainte ne contient que les flots de données. La recherche d'un couplage complet dans un graphe biparti est une manière d'appliquer l'élimination structurelle des variables, qui conduit aux relations de redondance analytique **RRA**s. Par conséquent, chaque combinaison de **RRA**s est également une **RRA**. □

**Proposition 3.1.2.** *Considérons le modèle du système $\Sigma = (K, X \cup O)$, où $K$ est l'ensemble des contraintes du système, composé de deux sous-ensembles : les contraintes modélisant les composants du système sans capteurs et les actionneurs : $(K_{proc})$ et les contraintes modélisant les capteurs et les actionneurs : $(K_{obs})$. Supposons que toutes les variables du système $\Sigma$ sont mesurées $(card(X) = card(O))$. Alors l'ensemble des contraintes de $K_{proc}$ instanciées par les observations constitue une base $\mathcal{A}_b$ qui permet de construire toutes les **RRAs** du système.*

*Démonstration.* Comme il y a une contrainte d'observation $k_{obs} \in K_{obs}$ de la forme $o_i = f_i(x_i)$ pour chaque paire $(x_i, o_i)$, un couplage complet entre les contraintes du système $K$ et les variables $X$ est obtenu en associant les variables $X$ à leurs contraintes d'observation correspondantes $(K_{obs})$. Par conséquent, les contraintes dans $(K_{proc})$ sont

toutes les relations de redondance, et les **RRA**s correspondantes sont obtenues en remplaçant les variables par leurs expressions d'observation $x_i = f_i^{-1}(o_i)$. L'ensemble des **RRA**s est une base pour les **RRAs** du système $\Sigma$. □

**Proposition 3.1.3.** *Considérons le modèle du système $\Sigma^{'} = (K^{'}, X \cup O^{'})$ résultant du modèle $\Sigma = (K, X \cup O)$ après le retrait d'un capteur $\mathbf{s}(x_i)$ mesurant la variable $x_i$. Dans ce modèle $K^{'} = K_{proc} \cup K^{'}_{obs}$ tel que $K^{'}_{obs} = K_{obs} \setminus k^{i}_{obs}$ où $k^{i}_{obs}$ est la contrainte modélisant le capteur $\mathbf{s}(x_i)$ et $O^{'} = O \setminus o_i$ où $o_i$ est l'observation de $x_i$. Alors une base $\mathcal{A}^{'}_b$ est donnée par $\{\mathcal{A}_b \setminus \mathcal{A}_b(o_i) \cup Comb(\mathcal{A}_b(o_i))\}$ telle que $\mathcal{A}_b(o_i)$ est l'ensemble de **RRA**s contenant l'observation $o_i$ et $Comb(\mathcal{A}_b(o_i))$ est l'ensemble de **RRA**s combiné obtenu de $\mathcal{A}_b(o_i)$ en éliminant la variable $o_i$, c'est-à-dire extraire $o_i$ d'une relation dans $\mathcal{A}_b(o_i)$ et la remplacer dans les autres relations.*

*Démonstration.* Construisons un couplage complet par rapport aux variables entre les contraintes $K^{'}$ et les variables $X$. Toutes les variables de $X$ peuvent être couplées à l'exception de $x_i$ parce que la contrainte d'observation $k^{i}_{obs}$ n'est pas dans $(K^{'}_{obs})$ (elle est retirée). Alors $x_i$ doit être couplée avec une contrainte $k^{*}_{proc} \in K_{proc}(x_i)$ tel que $K_{proc}(x_i)$ est l'ensemble de contraintes de $K_{proc}$ qui contient la variable $x_i$. Ces contraintes sont seulement celles qui correspondent à $\mathcal{A}_b(o_i)$. Les **RRA**s dans $\mathcal{A}_b(o_i)$ ne sont pas une partie des **RRA**s du $\Sigma^{'}$. Un nouvel ensemble des **RRA**s intitulé $Comb(\mathcal{A}_b(o_i))$ peut être ajouté aux **RRA**s du $\Sigma^{'}$. Cet ensemble est obtenu en utilisant $k^{*}_{proc}$ pour produire $x_i$, puis $x_i$ sera remplacée dans $K_{proc}(x_i) \setminus \{k^{*}_{proc}\}$, puis les variables $x_j, j \neq i$ seront remplacées par leurs observations $o_j$. D'une façon équivalente, en considérant $\mathcal{A}_b(o_i)$, $o_i$ est extrait de la **RRA** correspondant à $k^{*}_{proc}$ et remplacée dans les autres **RRA**s de $\mathcal{A}_b(o_i)$. $\mathcal{A}_b(o_i)$ est alors remplacée par $Comb(\mathcal{A}_b(o_i))$ dans les **RRA**s du $\Sigma^{'}$. □

Pour bien comprendre cette méthode, nous allons prendre l'exemple simple d'un additionneur connecté à un inverseur représenté dans la figure 3.6.

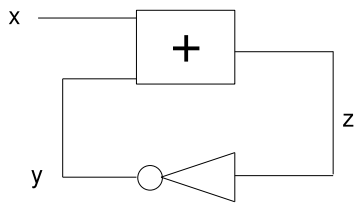


FIG. 3.6 – additionneur connecté à un inverseur

Le modèle de comportement du système est décrit par les contraintes $K_{proc}$ suivantes :

$$\begin{aligned} A : k_1 : z &= x + y \\ I : k_2 : y &= -z \end{aligned} \tag{3.1.7}$$

Et le modèle d'observation est décrit par les contraintes $K_{obs}$ suivantes :

$$\begin{aligned} \mathbf{s}(x) : k_5 : x &= \tilde{x} \\ \mathbf{s}(y) : k_6 : y &= \tilde{y} \\ \mathbf{s}(z) : k_7 : z &= \tilde{z} \end{aligned} \tag{3.1.8}$$

tel que $X = \{x, y, z\}$ est l'ensemble de variables et $O = \{\tilde{x}, \tilde{y}, \tilde{z}\}$ est l'ensemble de flots de données (observations données par les capteurs et actionneurs). Quand les trois variables sont mesurées, alors selon la proposition 3.1.2, une base $\mathcal{A}_b$ des **RRA**s est simplement donnée par les contraintes primaires du système $K_{proc}$ dans lesquelles chaque variable est remplacée par sa mesure.

$$\begin{aligned} RRA_1^0 : \tilde{z} = \tilde{x} + \tilde{y} \quad & Support(RRA_1^0) = \{A, \mathbf{s}(x), \mathbf{s}(y), \mathbf{s}(z)\} \\ RRA_2^0 : \tilde{y} = -\tilde{z} \quad & Support(RRA_2^0) = \{I, \mathbf{s}(y), \mathbf{s}(z)\} \end{aligned} \tag{3.1.9}$$

et $\mathcal{A}_b = \{RRA_1^0, RRA_2^0\}$.
$RRA_1^0$ et $RRA_2^0$ peuvent être interprétées de la façon suivante

$$\begin{aligned} RRA_1 : \tilde{z} = \tilde{x} + \tilde{y} \quad & Support(RRA_1) = \{A, \mathbf{s}(x), \mathbf{s}(y), \mathbf{s}(z)\} \\ RRA_2 : \tilde{x} = \tilde{z} - \tilde{y} \quad & Support(RRA_2) = \{A, \mathbf{s}(x), \mathbf{s}(y), \mathbf{s}(z)\} \\ RRA_3 : \tilde{y} = -\tilde{x} + \tilde{z} \quad & Support(RRA_3) = \{A, \mathbf{s}(x), \mathbf{s}(y), \mathbf{s}(z)\} \\ RRA_4 : \tilde{y} = -\tilde{z} \quad & Support(RRA_4) = \{I, \mathbf{s}(y), \mathbf{s}(z)\} \\ RRA_5 : \tilde{z} = -\tilde{y} \quad & Support(RRA_5) = \{I, \mathbf{s}(y), \mathbf{s}(z)\} \end{aligned} \tag{3.1.10}$$

Les formules dans 3.1.10 représentent les interprétations causales des $RRA_1^0$ et $RRA_2^0$ et correspondent aux **H-RRA**s primaires (c'est-à-dire les **RRAs** résultant en remplaçant les mesures dans les contraintes $K_{proc}$, dans le cas où toutes les variables du système sont mesurées (proposition 3.1.2)). Ces **H-RRA** sont représentées par le tableau 3.7.

Nous remarquons qu'il y a 5 **H-RRA**s et nous n'avons pas besoin de *conditions de validité (contraintes de validité)* : il s'agit d'une ou de plusieurs contraintes supplémentaires qui déterminent quand la *contrainte de modèle (ou de comportement)* peut être utilisée. Si les *contraintes de validité* ne sont pas satisfaites, cela signifie que la contrainte de modèle ne peut pas être utilisée dans le contexte courant. Il devient beaucoup plus facile de construire des modèles précis pour des contextes particuliers.

Considérons le système $\Sigma' = (K', X \cup O')$ correspondant à la situation dans laquelle seulement $\mathbf{s}(x)$ et $\mathbf{s}(z)$ sont disponibles mais pas $\mathbf{s}(y)$, $\mathbf{s}(y)$ a été retiré (marqué

| | Support-relation | | Support-relation causale | | | | | CV | Structure | | | Support-composant | | support-capteur | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $k_1$ | $k_2$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ | | $x$ | $y$ | $z$ | $A$ | $I$ | $s(x)$ | $s(y)$ | $s(z)$ |
| RRA1 | x | | x | | | | | | x | x | ⊗ | x | | x | x | x |
| RRA2 | x | | | x | | | | | ⊗ | x | x | x | | x | x | x |
| RRA3 | x | | | | x | | | | x | ⊗ | x | x | | x | x | x |
| RRA4 | | x | | | | x | | | | ⊗ | x | | x | | x | x |
| RRA5 | | x | | | | | x | | | x | ⊗ | | x | | x | x |

FIG. 3.7 – La matrice de H-RRAs primaires

| | Suuport-relation | | Support-relation causale | | | | | CV | Structure | | | Support-composant | | support-capteur | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $k_1$ | $k_2$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ | | $x$ | $y$ | $z$ | $A$ | $I$ | $s(x)$ | $s(y)$ | $s(z)$ |
| RRA1 | x | | x | | | | | | x | x | ⊗ | x | | x | x | x |
| RRA2 | x | | | x | | | | | ⊗ | x | x | x | | x | x | x |
| RRA3 | x | | | | x | | | | x | ⊗ | x | x | | x | x | x |
| RRA4 | | x | | | | x | | | | ⊗ | x | | x | | x | x |
| RRA5 | | x | | | | | x | | | x | ⊗ | | x | | x | x |
| RRA6 | x | x | x | | | x | | | x | # | ⊗ | x | x | x | | x |
| RRA7 | x | x | | | x | x | | | ⊗ | # | x | x | x | x | | x |
| RRA8 | x | x | x | | | x | | | x | ⊗ | # | x | x | x | x | |
| RRA9 | x | x | | | x | | x | | ⊗ | x | # | x | x | x | x | |

FIG. 3.8 – La matrice de H-RRAs finals

par # dans la sixième ligne dans le tableau 3.8). $RRA_1^0$ et $RRA_2^0$ ne sont plus des **RRA**s dans le système $\Sigma'$. Une nouvelle **RRA** vient de la combinaison d'une **RRA** de $\{RRA1, RRA_2, RRA3\}$ avec une **RRA** de $\{RRA4, RRA_5\}$, parce qu'elles partagent la même variable $\tilde{y}$.
Par exemple, en combinant $RRA1$ avec $RRA4$ ou $RRA3$ avec $RRA4$, la relation de redondance analytique résultant est :

$$RRA_3^0 : \tilde{x} = 2\tilde{z} \quad Support(RRA_3^0) = \{A, I, \mathbf{s}(x), \mathbf{s}(z)\}. \tag{3.1.11}$$

Dans ce cas, $\mathcal{A}_b^{\{y\}} = \{RRA_3^0\}$ tel que $\mathcal{A}_b^{\{y\}}$ est la base des **RRA**s après avoir retiré le capteur mesurant la variable $y$. $RRA_3^0$ a deux interprétations causales :

$$\begin{aligned} RRA_6 &: \tilde{x} = 2\tilde{z} \quad Support(RRA_6) = \{A, I, \mathbf{s}(x), \mathbf{s}(z)\}. \\ RRA_7 &: \tilde{z} = \tfrac{1}{2}\tilde{x} \quad Support(RRA_7) = \{A, I, \mathbf{s}(x), \mathbf{s}(z)\}. \end{aligned} \tag{3.1.12}$$

$RRA_6$ et $RRA_7$ sont représentées dans le tableau 3.8.

Considérons la situation dans laquelle seuls $\mathbf{s}(x)$ et $\mathbf{s}(y)$ sont disponibles mais pas $\mathbf{s}(z)$, $\mathbf{s}(z)$ a été retiré (marquée par # dans la huitième ligne dans le tableau 3.8). La combinaison d'une **RRA** de $\{RRA1, RRA_2, RRA3\}$ avec une **RRA** de $\{RRA4, RRA_5\}$ par rapport à la variable $z$, fournit $RRA_4^0$ :

$$RRA_4^0 : \tilde{x} = -2\tilde{y} \quad Support(RRA_4^0) = \{A, I, \mathbf{s}(x), \mathbf{s}(y)\}. \tag{3.1.13}$$

Dans ce cas, $\mathcal{A}_b^{\{z\}} = \{RRA_4^0\}$ telle que $\mathcal{A}_b^{\{z\}}$ est la base des **RRA**s après avoir retiré le capteur mesurant la variable $z$. $RRA_4^0$ a deux interprétations causales :

$$\begin{aligned} RRA_8 &: \tilde{x} = -2\tilde{y} \quad Support(RRA_8) = \{A, I, \mathbf{s}(x), \mathbf{s}(y)\}. \\ RRA_9 &: \tilde{y} = -\tfrac{1}{2}\tilde{x} \quad Support(RRA_9) = \{A, I, \mathbf{s}(x), \mathbf{s}(y)\}. \end{aligned} \tag{3.1.14}$$

$RRA_8$ et $RRA_9$ sont représentées dans le tableau 3.8.

Considérons la situation dans laquelle seule $\mathbf{s}(y)$ et $\mathbf{s}(z)$ sont disponibles mais pas $\mathbf{s}(x)$, $\mathbf{s}(x)$ a été retiré. Il n'est pas possible de combiner une **RRA** de $\{RRA1, RRA_2, RRA3\}$ avec une **RRA** de $\{RRA4, RRA_5\}$ par ce que aucune **RRA** de $\{RRA4, RRA_5\}$ contient la variable $x$. Par conséquent aucune nouvelle RRA ne peut être obtenue. Dans ce cas $\mathcal{A}_b^{\{x\}} = \{\emptyset\}$ telle que $\mathcal{A}_b^{\{y\}}$ est la base des **RRA**s après avoir retiré le capteur mesurant la variable $x$.

Finalement, dans le cas où les deux capteurs $\mathbf{s}(y)$ et $\mathbf{s}(z)$ sont retirés, aucune RRA ne peut être obtenue : $\mathcal{A}_b^{\{y,z\}} = \{\emptyset\}$ telle que $\mathcal{A}_b^{\{y,z\}}$ est la base des **RRA**s après avoir retiré les capteurs mesurant les variables $y, z$. En effet, $RRA_3^0$ partage la variable $\tilde{z}$ avec $RRA_1^0$ et $RRA_2^0$, mais $RRA_3^0$ ne peut pas être combinée avec $RRA_1^0$ et $RRA_2^0$ qui ont produit $RRA_3^0$.
Un algorithme produisant les **H-RRAs** est proposé dans (Travé-Massuyès et al. [2006]).

La méthode proposée par Travé-Massuyès et al. [2006] permet de trouver les relations de redondance analytique des systèmes. Cette méthode consiste á retirer toutes les combinaisons possibles des variables, alors sa complexité est exponentielle par rapport aux variables. En plus cette complexité augmente encore en prenant en compte l'interprétation causale présentée dans l'équation 3.1.10. Ceci représente l'inconvénient de cette méthode. Nous avons remarqué que pour un simple exemple qui contient 3 **RRAs**, 9 **RRAs** sont obtenues.

### 3.1.4 Méthode basée sur la décomposition de Dulmage-Mendelshon

(Krysander et al. [2008]) ont proposé un algorithme pour la conception de relations de redondance analytique en partant d'un modèle structurel. Cet algorithme est basé

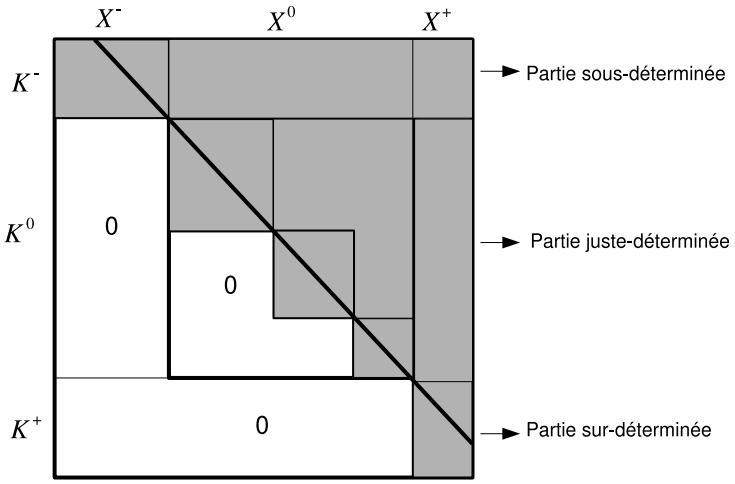


FIG. 3.9 – Décomposition Dulmage-mendelshon d'un graphe bipartie

sur la décomposition de Dulmage-Mendelshon et permet de trouver les sous-systèmes sur-déterminées minimaux(les sous-systèmes testables minimaux) qui permettent de générer les **RRAs**. Les résultats présentés dans (Krysander et al. [2008]) sont l'extension des résultats présentés dans (Krysander and Nyberg [2005]). La complexité de cet algorithme dépend de l'ordre des redondances structurelles c'est-à-dire de la différence entre le nombre d'équations et le nombre de variables. Cette complexité est inférieure à la complexité des algorithmes proposés dans (Krysander and Nyberg [2002], Frisk et al. [2003]). Une décomposition de Dulmage-Mendelshon d'un graphe biparti $G = (K, X, A)$ est présentée par la figure 3.9 tel que $K$ est l'ensemble des contraintes d'un système $\Sigma$, $X$ est l'ensemble des variables du système et $A$ est l'ensemble des arcs reliant $K$ et $X$ .
Dans cette figure, $K$ et $X$ représentent respectivement les contraintes et les variables du système. La zone grise contient des 1 et des 0, tandis que la zone blanche ne contient que des 0. La ligne épaisse représente un couplage maximal dans ce graphe.

Les contraintes du système sont décomposées en trois parties : $K^+$ est la partie sur-déterminée avec plus de contraintes que de variables, $K^-$ est la partie sous-déterminée avec plus de variables que de contraintes et $K^0$ est la partie juste-déterminée avec le même nombre de contraintes que de variables.
Quelques définitions importantes pour la conception des sous-systèmes sur-déterminés minimaux sont rappelées.

**Définition 3.1.2.** *(Structurellement sur-déterminé (SO))*
*Un ensemble de contraintes $K$ est structurellement sur-déterminé $(SO)$ si $K$ a plus de contraintes que de variables.*

**Définition 3.1.3.** *(Structurellement rationnellement sur-déterminé $(PSO)$)*
*Un ensemble de contraintes $K$ structurellement sur-déterminé $(SO)$ est un ensemble*

*structurellement rationnellement sur-déterminé* $(PSO)$ *si* $K = K^+$.

Un ensemble $PSO$ conduit généralement à un ensemble de sous-systèmes testables.

**Définition 3.1.4.** *(Structurellement sur-déterminé minimal* $(MSO)$*)*
*Un ensemble* $K$ *structurellement sur-déterminé est un ensemble structurellement sur-déterminé minimal* $(MSO)$ *si aucun sous-ensemble est un ensemble structurellement sur-déterminé* $(SO)$.

Notons qu'un ensemble $(MSO)$ est aussi un ensemble $(PSO)$.
Considérons, par exemple, un système modélisé par les équations différentielles suivantes :

$$\begin{aligned} k_1 &: x_1 = x_2 + x_3 - 4 \\ k_2 &: x_2 = x_3 + 1 \\ k_3 &: x_1 = x_3 \\ k_4 &: x_1 = \tilde{x_1} \end{aligned} \tag{3.1.15}$$

Dans cet exemple, les variables sont $x_1$, $x_2$ et $x_3$ et le flot de données est $\tilde{x_1}$.
En ne considérant que les variables et pas les flots de données, ce système peut être représenté par la matrice structurelle 3.3.

Tab. 3.3 – La matrice structurelle du système

| | $x_1$ | $x_2$ | $x_3$ |
|---|---|---|---|
| $k_1$ | 1 | 1 | 1 |
| $k_2$ | 1 | 1 | 0 |
| $k_3$ | 0 | 1 | 1 |
| $k_4$ | 1 | 0 | 0 |

Dans cette matrice, les contraintes implicites ne sont pas prises en compte. Cet exemple est un $MSO$. Il a plus de contraintes que de variables. L'équation $\tilde{x_1} - 3 = 0$ peut être trouvée par l'élimination des variables. Cette équation est appelée relation de redondance analytique et elle peut être utilisée pour vérifier si la valeur de $\tilde{x_1}$ est consistante avec le modèle du système. L'ensemble des contraintes $\{k_1, k_2, k_3, k_3\}$ est un sous-système testable.

L'algorithme proposé dans (Krysander et al. [2008]) est basé sur une approche descendante, c'est-à-dire qu'il commence avec le modèle complet du système et il réduit la taille du modèle étape par étape jusqu'à obtenir un $MSO$. Pour bien comprendre cet algorithme, nous avons besoin de la notion de redondance structurelle.

Considérons un graphe biparti $G = (K, X, A)$ et supposons que $var_X(K) \subseteq X$ est l'ensemble de variables apparaissant dans un ensemble donné de contraintes $K$.
La redondance structurelle de l'ensemble $K$ est donnée par :
$\varphi K = |K^+| - |var_X(K^+)|$.
L'algorithme est basé sur les trois lemmes suivants.

**Lemme 3.1.4.** *Si $K$ est un ensemble de contraintes PSO et $k \in K$, alors*

$$\varphi(K \setminus \{k\}) = \varphi(K) - 1. \tag{3.1.16}$$

**Lemme 3.1.5.** *L'ensemble des contraintes $K$ est un ensemble MSO si et seulement si $K$ est un ensemble PSO et $\varphi K = 1$.*

**Lemme 3.1.6.** *Si $K$ est un ensemble de contraintes, $E \subseteq K$ est un ensemble PSO et $k \in K \setminus E$, alors $E \subseteq (K \setminus \{k\})^+$*

Considérons, par exemple, un système modélisé par la matrice structurelle 3.4 telles que les variables du système sont : $(x_1, x_2)$.

Tab. 3.4 – La matrice structurelle d'un système

| | $x_1$ | $x_2$ |
|---|---|---|
| $k_1$ | 1 | 0 |
| $k_2$ | 1 | 0 |
| $k_3$ | 1 | 1 |
| $k_4$ | 0 | 1 |

La redondance structurelle de ces contraintes est 2.
Appliquons l'algorithme présenté dans (Krysander et al. [2008]) :

- si $k_1$ est supprimé, La décomposition Dulmage-Mendelshon de ce nouveau sous-ensemble $K \setminus k_1 = \{k_2, k_3, k_4\}$ est $(K \setminus \{k_1\})^+ = \{k_2, k_3, k_4\}$, $(K \setminus \{k_1\})^- = \{\emptyset\}$ et $(K \setminus \{k_1\})^0 = \{\emptyset\}$. L'ensemble $K^{'} = (K \setminus \{k_1\})^+ = \{k_2, k_3, k_4\}$ vérifie : $\varphi K^{'} = 1$, et cet ensemble de contraintes $K^{'}$ est un MSO.
- si $k_2$ est supprimé, La décomposition Dulmage-Mendelshon de ce nouveau sous-ensemble $K \setminus k_2 = \{k_1, k_3, k_4\}$ est $(K \setminus \{k_2\})^+ = \{k_1, k_3, k_4\}$, $(K \setminus \{k_2\})^- = \{\emptyset\}$ et $(K \setminus \{k_2\})^0 = \{\emptyset\}$. L'ensemble $K^{'} = (K \setminus \{k_2\})^+ = \{k_1, k_3, k_4\}$ vérifie : $\varphi K^{'} = 1$, et cet ensemble de contraintes $K^{'}$ est un MSO.
- si $k_3$ est supprimé, La décomposition Dulmage-Mendelshon de ce nouveau sous-ensemble $K \setminus k_3 = \{k_1, k_2, k_4\}$ est $(K \setminus \{k_3\})^+ = \{k_1, k_2\}$, $(K \setminus \{k_3\})^- = \{\emptyset\}$ et $(K \setminus \{k_3\})^0 = \{k_4\}$. L'ensemble $K^{'} = (K \setminus \{k_3\})^+ = \{k_1, k_2\}$ vérifie : $\varphi K^{'} = 1$, et cet ensemble de contraintes $K^{'}$ est un MSO.

– si $k_4$ est supprimé, La décomposition Dulmage-Mendelshon de ce nouveau sous-ensemble $K \setminus k_4 = \{k_1, k_2, k_3\}$ est $(K \setminus \{k_4\})^+ = \{k_1, k_2\}$, $(K \setminus \{k_4\})^- = \{\emptyset\}$ et $(K \setminus \{k_4\})^0 = \{k_3\}$. L'ensemble $K^{'} = (K \setminus \{k_4\})^+ = \{k_1, k_2\}$ vérifie : $\varphi E = 1$, et cet ensemble de contraintes $K^{'}$ est un $MSO$ déjà obtenu dans le cas où $k_3$ a été supprimé.

Le même ensemble $MSO$ $\{k_1, k_2\}$ est obtenu deux fois : cet algorithme n'est pas donc optimal. (Krysander et al. [2008]) ont présenté des améliorations afin d'augmenter l'efficacité de cet algorithme. Les $MSO$ résultants de cet algorithme conduisent aux relations de redondance analytique.

Reprenons l'exemple de compteur digital proposé et appliquons l'algorithme proposé dans (Krysander et al. [2008]). Nous obtenons les $SMO$ suivants :
$\{k_1, k_5, k_6\}$, $\{k_2, k_6, k_7\}$, $\{k_3, k_7, k_8\}$, $\{k_4, k_8, k_9\}$, $\{k_1, k_2, k_5, k_7\}$, $\{k_2, k_3, k_6, k_8\}$, $\{k_3, k_4, k_7, k_9\}$, $\{k_1, k_2, k_3, k_5, k_8\}$, $\{k_2, k_3, k_4, k_6, k_9\}$ et $\{k_1, k_2, k_3, k_4, k_5, k_9\}$.
Ces $SMO$ résultants permettent de consrtuire les 10 relations de redondance analytique possibles du système.
Concernant cette méthode, nous avons remarqué qu'il y a des redondances. (Krysander et al. [2008]) a mené des améliorations sur l'algorithme afin d'augmenter son efficacité et réduire sa complexité. Il est donc possible d'utiliser la structure réduite pour trouver tous les ensembles $MSO$ conduisant aux **RRA**s dans la structure originale. L'inconvénient de cette méthode est qu'elle ne prend pas en compte la notion de déductibilité des variables : toutes les variables du système sont considérées comme déductibles et, de ce fait, certaines **RRAs** peuvent ne pas être réalisables.

### 3.1.5 Approche DX de la génération de RRAs

Dans la communauté de l'Intelligence Artificielle **DX**, le diagnostic à base de consistance est le plus répandu pour diagnostiquer les défauts. Plusieurs approches de diagnostic à base de consistance exigent de calculer les ensembles de conflits possibles et de générer les diagnostics candidats. Un conflit (Reiter [1987]) est un ensemble de composants dont l'hypothèse de comportement normal est inconsistante avec les observations courantes. (Pulido and Alonso [2002], Pulido and Alonso [2004]) ont proposé une méthode pour générer les conflits possibles nécessaires pour la détection et la localisation des défauts. Cette méthode est composée de deux étapes. La première étape consiste à chercher les systèmes sur-déterminées. La deuxième étape consiste à vérifier si ces systèmes peuvent être résolus en utilisant la propagation locale.
La définition de hyper-graphe est nécessaire à la conception de cette méthode.

**Définition 3.1.5.** *Un hyper-graphe est représenté par : $H(V, E)$ tel que $V$ est un ensemble de sommets de $H$ et $E$ est une partie de la partition $\mathcal{P}(V)$. Un élément de $E$ est*

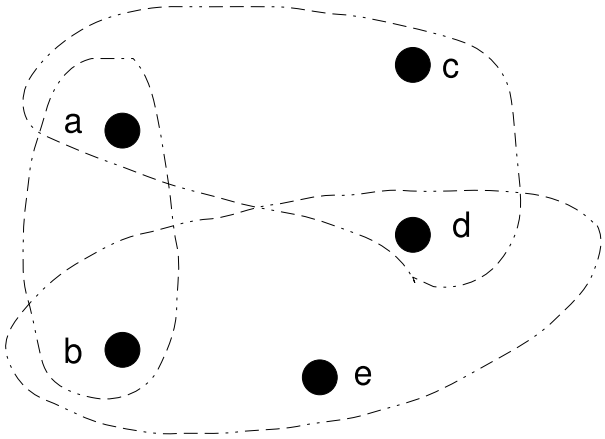


FIG. 3.10 – Un hyper-graphe

*appelé hyper-arrête.*

Soitent $V = \{a, b, c, d, e\}$, $E = \{\{a, b\}, \{a, c, d\}, \{b, d, e\}\}$. Le hyper-graphe est représenté par la figure 3.10.

Le modèle **DS** (description du système) peut être représenté par un hyper-graphe $H_{SD} = \{V, K\}$ tel que :

- $V = \{v_1, v_2, ..., v_n\}$ est l'ensemble de variables du systèmes $X$ plus les flots de données $O$, $V = X \cup O$.
- $K = \{k_1, k_2, ..., k_m\}$ est une famille de sous-ensembles dans $V$, tel que chaque $k_i \in K$ représente une contrainte du modèle. Chacune de ces contraintes contient des variables et des flots de données.

Les sous systèmes sur-déterminés dans $H_{SD}$ seront appelés "*chaînes d'évaluation*"

## A Chaîne d'évaluation

$H_{EC} \subseteq H_{SD}$ est un sous hyper-graphe dans $H_{SD}$ : $H_{EC} = \{V_{EC}, K_{EC}\}$, où $V_{EC} = X_{EC} \cup O_{EC} \in V$, $K_{EC} \in K$, $X_{EC}$ est l'ensemble de variables dans de la chaîne d'évaluation $EC$, $O_{EC}$ est l'ensemble des flots de données dans $EC$ et $H_{EC}$ vérifie :

1. $H_{EC}$ est un hyper-graphe connecté
2. $O_{EC} \neq \emptyset$
3. $\forall v \in X_{EC} \Rightarrow d_{H_{EC}}(v) \geq 2$
4. soit $G(H_{EC})$ un graphe biparti dont l'ensemble de noeuds est divisé en deux sous ensembles : $X_{EC}$, $K_{EC}$ tel que chaque arc de ce graphe relie un noeud du premier ensemble $x_i \in X_{EC}$ à un noeud de l'autre ensemble $k_i \in K_{EC}$. Alors, il y a un couplage complet par rapport aux variables.

Comme nous sommes intéressés aux conflits minimaux, seules les chaînes d'évaluation minimales $MEC$ sont utiles.

## B Chaîne d'évaluation minimale

$H_{EC}$ est une chaîne d'évaluation minimale si il n'y pas une autre chaîne $H'_{EC} \subset H_{EC}$. (Pulido and Alonso [2002], Pulido and Alonso [2004]) ont proposé des algorithmes pour construire l'ensemble des chaînes d'évaluation minimales **MEC**s.
Prenons le système polybox (Cordier et al. [2004]). Ce système se compose de trois multiplicateurs $M_1, M_2, M_3$, et deux additionneurs $A_1, A_2$ (figure 3.11). Les composants

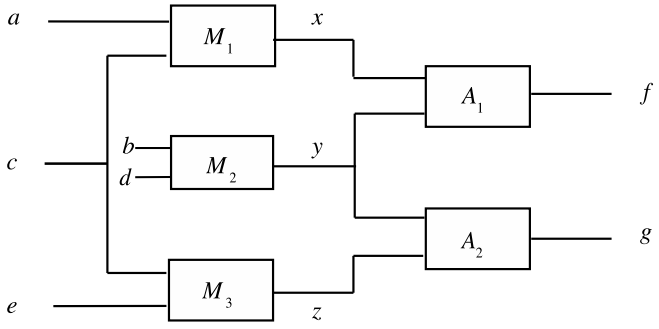


FIG. 3.11 – Système polybox

du système $(M_1, M_2, M_3, A_1, A_2)$ sont décrits par les modèles élémentaires suivants :

$$\begin{aligned} k_1 &: x = a \times c \\ k_2 &: y = b \times d \\ k_3 &: z = c \times e) \\ k_4 &: f = x + y \\ k_5 &: g = y + z \end{aligned} \tag{3.1.17}$$

Les capteurs et les actionneurs sont décrits par les modèles élémentaires suivants :

$$\begin{aligned} k_6 &: a = \tilde{a} \\ k_7 &: b = \tilde{b} \\ k_8 &: c = \tilde{c} \\ k_9 &: d = \tilde{d} \\ k_{10} &: e = \tilde{e} \\ k_{11} &: f = \tilde{f} \\ k_{12} &: g = \tilde{g} \end{aligned} \tag{3.1.18}$$

Appliquons les algorithmes proposés dans (Pulido and Alonso [2002], Pulido and Alonso [2004]), nous obtenons les $MECs$ suivantes :

$$\begin{aligned} H_{EC1} &= \{\{a,b,c,d,f,x,y\},\{k_1,k_2,k_4\}\} \\ H_{EC2} &= \{\{b,c,d,e,g,y,z\},\{k_2,k_3,k_5\}\} \\ H_{EC3} &= \{\{a,c,e,f,g,x,y,z\},\{k_1,k_3,k_4,k_5\}\} \end{aligned} \tag{3.1.19}$$

où les contraintes $k_1$, $k_2$, $k_3$, $k_4$, $k_5$ modélisent les composants $M_1$, $M_2$, $M_3$, $A_1$, $A_2$ respectivement.

## C La résolution d'une chaîne d'évaluation

Un conflit minimal est un système structurellement sur-déterminé que nous voulons résoudre en utilisant la propagation locale. Cependant, le hyper-graphe n'a pas suffisamment d'informations sur la façon dont chaque contrainte peut être résolue. Pour résoudre ce problème, nous créons un graphe AND-OR pour chaque chaîne d'évaluation minimale. Dans un tel graphe, il y a un ou plusieurs arcs AND-OR pour chaque hyper-arc dans la **MEC**. Chaque arc AND-OR représente de quelle manière le hyper-arc pourrait être résolu. En effet, pour résoudre une **MEC**, nous devons choisir un arc AND-OR de chaque contrainte. Par conséquent, le choix de différents arcs AND-OR du graphe AND-OR génère différentes manières pour résoudre la **MEC**. En plus, les systèmes sur-déterminés peuvent être résolus seulement en utilisant les critères de la propagation locale. Chacune des manières différentes de résoudre un **MEC** est appelée "modèle d'évaluation minimal **MEM**".
Dans l'exemple de polybox, chaque contrainte produit quelques interprétations pour le graphe AND-OR. Prenons par exemple la contrainte $k_1$ modélisant le multiplicateur $M_1$. Les trois interprétations possibles sont :

$$\begin{aligned} &k_{1_1} : x = a \times c \\ &k_{1_2} : a = x/c, \quad si \quad c \neq \emptyset \\ &k_{1_3} : c = x/a, \quad si \quad a \neq \emptyset \end{aligned} \tag{3.1.20}$$

**C.1 Modèle d'évaluation minimale** Un graphe partiel AND-OR, $H_{MEM} = \{V_{MEM}, K_{MEM}\} \subseteq AOG(H_{EC})$ tel que $AOG(H_{EC})$ est le graphe AND-OR obtenu de $H_{EC} = \{V_{EC}, K_{EC}\}$, est un modèle d'évaluation minimale si et seulement si :

1. $K_{MEM}$ est un hitting-set minimal pour la partition induite par $k_i \in K_{EC}$ dans $K_{EM}$.
2. $\forall v_i | v_i \in X_{MEM}$ and $v_i$ est un noeud feuille, $v_i$ est un flot de données.

3. $\exists x_j \in V_{MEM} | x_j$ est un noeud intermédiaire.

4. si $x_j$ est un noeud intermédiaire, alors il existe un chemin orienté et acyclique dans $H_{MEM} : \{x_i, x_i + 1, ..., xi + k, x_j\}$ pour chaque noeud de $x_i$ à $x_j$.

(Pulido and Alonso [2002], Pulido and Alonso [2004]) ont proposé des algorithmes pour calculer les **MEM**s. Par exemple, reprenons la chaîne d'évaluation minimale $H_{EC_1}$. Cette chaîne a un graphe $AND - OR$ lié : $AOG(H_{EC_1}) = \{\{a, b, c, d, f, x, y\}, \{k_{1_1}, k_{1_2}, k_{1_3}, k_{2_1}, k_{2_2}, k_{2_3}, k_{4_1}, k_{4_2}, k_{4_3}\}\}$.
En utilisant $H_{EC_1}$ et ses interprétations disponibles dans $AOG(H_{EC_1})$, l'algorithme proposé dans (Pulido and Alonso [2002], Pulido and Alonso [2004]) trouve les **MEM**s suivantes :
$\{k_{1_1}, k_{2_1}, k_{4_1}\} : \tilde{f} \equiv f_{pred} = a \times c + b \times d$
$\{k_{1_1}, k_{2_1}, k_{4_2}\} : x_{pred_1} = a \times c \equiv x_{pred_2} = f - b \times d$
$\{k_{1_2}, k_{2_1}, k_{4_2}\} : \tilde{a} \equiv a_{pred} = (f - b \times d)/c$, if $c \neq \emptyset$
$\{k_{1_3}, k_{2_1}, k_{4_2}\} : \tilde{c} \equiv c_{pred} = (f - b \times d)/a$, if $a \neq \emptyset$
$\{k_{1_1}, k_{2_1}, k_{4_3}\} : y_{pred_1} = f - (a \times c) \equiv y_{pred_2} = b \times d$
$\{k_{1_1}, k_{2_2}, k_{4_3}\} : \tilde{b} \equiv b_{pred} = (f - a \times c)/d$, if $d \neq \emptyset$
$\{k_{1_1}, k_{2_3}, k_{4_3}\} : \tilde{d} \equiv d_{pred} = (f - a \times c)/b$, if $b \neq \emptyset$
Il est important de noter qu'une **MEC** ne peut pas produire une **MEM** si le système sur-déterminé ne peut pas être résolu en utilisant les interprétations et la propagation locale.
Ces **MEM**s pourraient conduire à des conflits. L'avantage de cette méthode de conception de conflits est que l'ensemble de **MEM**s peuvent être calculés hors ligne sans aucune évaluation du modèle. Les conflits peuvent apparaître seulement quand les observations sont introduites et l'évaluation du modèle est calculée. Ces conflits résultant seront utilisés pour la détection et la localisation de défauts.
Reprenons l'exemple du polybox, il y a trois conflits possibles : $\{\{k_1, k_2, k_4\}, \{k_2, k_3, k_5\}, \{k_1, k_3, k_4, k_5\}\}$, parce que chaque **MEC** a au moins un **MEM**.
Considérons les observations suivantes : $OBS = \{ENTRE1(M_1) : (a = 3), ENTRE2(M_1) : (c = 5), ENTRE1(M_2) : (b = 3), ENTRE2(M_2) : (d = 5), ENTRE1(M_3) : (c = 5), ENTRE2(M_3) : (e = 3), SORTIE(A_1) : (f = 20), SORTIE(A_2) : (g = 30)\}$, en utilisant les composants à la place des contraintes, les conflits sont : $\{M_1, M_2, A_2\}$ et $\{M_1, A_3, A_1, A_3\}$.
Dans le cas où $(f = 20)$ et $(g = 20)$, les conflits sont : $\{M_1, M_2, A_1\}$ et $\{M_2, M_3, A_2\}$.
Dans le cas où $(f = 30)$ et $(g = 30)$, il n'y a pas de conflit. Aucune anomalie n'a été détectée.

La méthode proposée dans (Pulido and Alonso [2004]) permet de trouver les modèles d'évaluation minimaux conduisant aux conflits pour la détection et la localisation des défauts. Elle présente la particularité d'être difficile d'accès et complexe à mettre en oeuvre.

## 3.2 Comparatif entre les méthodes existantes pour la conception de RRAs

Dans la section précédente, nous avons synthétisé certaines méthodes existantes pour la conception de relations de redondance analytique (**RRAs**) pour le diagnostic de défauts.
Ces méthodes sont basées sur la recherche de couplages dans un graphe biparti, la propagation des valeurs, le retrait consécutif de capteurs, la décomposition Dulmage-Mendelshon. Nous avons synthétisé également une méthode DX pour la génération des **RRA**s. Nous avons remarqué que chacune de ces méthodes a des avantages et des inconvénients. Ces avantages et inconvénients sont représentés par le tableau suivant :

| Conception les RRAs par | Avantages | Inconvénients |
|---|---|---|
| Couplage | •facile à construire.<br>•prend en compte la notion de déductibilité. | •beaucoup de redondance dans les calculs. |
| Propagation de valeurs | • prend en compte la notion de déductibilité. | •difficile à construire toutes les propagations élémentaires.<br>•redondance dans les calculs |
| Retrait consécutif des capteurs | •prend en compte la notion de causalité (déductibilité). | •complexité exponentielle par rapport aux variables. |
| Décomposition Dulmage-Mendelshon | •pas de redondance dans les calculs.<br>•degré de complexité faible. | •prend pas en compte la notion de déductibilité<br>•ne s'applique qu'aux systèmes juste-déterminés . |
| Approche DX | •ne nécessite pas de modèle analytique. | • difficile d'accès.<br>•complexe à mettre en oeuvre. |

## 3.3 Conclusion

Dans ce chapitre, nous avons synthétisé des méthodes existantes pour la conception de relations de redondance analytique (**RRAs**) pour le diagnostic de défauts. Ces **RRAs** constituent un outil essentiel pour la détection et la localisation de défauts. Nous sommes particulièrement concentré sur les méthodes basées sur la recherche de couplages dans

un graphe biparti, la propagation des valeurs, le retrait consécutif de capteurs et la décomposition Dulmage-Mendelshon. Nous avons remarqué que les méthodes basées sur la recherche de couplages dans un graphe biparti, la propagation des valeurs permet de trouver les **RRA**s en prenant en compte la notion de déductibilité mais l'inconvénient de ces méthodes est qu'il y a beaucoup de redondances dans les calculs. La méthode basée sur le retrait consécutif de capteurs résout le problème de redondances mais la complexité est exponentielle par rapport au nombre de variables du système. La méthode basée sur la décomposition Dulmage-Mendelshon résout le problème de redondance et la complexité, mais l'inconvénient de cette méthode est qu'elle ne prend pas en compte la notion de déductibilité (calculabilité). Nous avons présenté aussi une méthode de conception de conflits issue de la communauté de l'intelligence artificielle. Cette méthode permet de trouver les modèles d'évaluation minimale conduisant aux conflits. Elle présente la particularité d'être difficile d'accès et complexe à mettre en oeuvre. Vu des inconvénients des méthodes existantes pour la conception des **RRA**s, nous en déduisons qu'il faut trouver une méthode pour la conception des **RRA**s qui résout le problème de redondances, la complexité et la déductibilité. Dans le chapitre suivant, nous allons présenter cette méthode qui réduit les limites des méthodes déjà existantes.

# Chapitre 4

# Approche structurelle proposée pour la conception de RRAs

## 4.1 Introduction

Dans l'approche **FDI**, l'analyse diagnostique des systèmes physiques se base sur les relations de redondance analytique **RRAs** qui fournissent des symptômes nécessaires pour détecter et isoler les défauts. Dans le chapitre précédent, nous avons présenté quelques méthodes structurelles pour la conception de **RRAs** et nous avons remarqué que les méthodes basées sur le couplage, la propagation des valeurs, le retrait consécutif de capteurs et la décomposition Dulmage-Mendelshon présentaient toutes des limites, soient en termes de complexité de calcul, soient en termes de classes des systèmes adressables. Dans ce chapitre, nous présentons une méthode plus générale et d'un ordre de complexité moindre que ces méthodes existantes, qui peut fournir toutes les relations de redondance analytique d'un système à diagnostiquer pour n'importe quelle classe de systèmes et qui permet de tenir compte de la déductibilité (calculabilité) des variables par rapport aux contraintes. Cette méthode améliore les résultats présentés dans (Ploix et al. [2005]). Notre algorithme est basé sur un opérateur "jointure" venant de l'algèbre relationnel. Il se fonde également sur une abstraction structurelle des contraintes et trace de toutes les contraintes impliquées dans les sous systèmes testables obtenus. Ce point est crucial en analyse diagnostique où il s'agit de comprendre les causes possibles d'un test révélant une anomalie.

## 4.2 Algèbre relationnelle et opérateur jointure

L'algèbre relationnelle est un support mathématique cohérent sur lequel s'appuie aujourd'hui les bases de données relationnelles. L'objectif de cette section est d'aborder l'algèbre relationnelle dans le but de décrire les opérations qu'il est possible d'appliquer sur des relations (contraintes) pour produire de nouvelle relation. Cette approche est donc plus opérationnelle que mathématique. Nous allons utiliser cette approche pour trouver une nouvelle approche permettant à concevoir les sous-systèmes testables.

### 4.2.1 Famille d'opérateurs relationnels

L'algèbre relationnel porte sur des ensembles caractérisés par des attributs ou variables caractérisées par un domaine de valeurs, et des relations entre les attributs.
Une relation $R$ peut être une classe ou une table. Elle est notée : $R(A_1, A_2, ..., A_n)$ tel que $A_1, A_2, ..., A_n$ représentent les attributs de la relation $R$.
Nous pouvons distinguer trois familles d'opérateurs relationnels :

- Les opérateurs unaires **(Projection, Sélection)** : ces opérateurs sont les plus simples. Ils permettent de produire une nouvelle relation à partir d'une autre relation.
- les opérateurs binaires ensemblistes **(Union, Différence, Intersection)** : ces opérateurs permettent de produire une nouvelle relation à partir de deux relations de même degré et de même domaine.
- les opérateurs binaires ou n-aires **(Produit cartésien, Jointure)** : ces opérateurs permettent de produire une nouvelle relation à partir de deux ou plusieurs autres relations.

Détaillons ces différents opérateurs :

**Définition 4.2.1.** *(Sélection)*
*La sélection (parfois appelée restriction) génère une relation regroupant exclusivement toutes les occurrences de la relation $R$ satisfaisant une expression logique $E$. Elle est notée : $\sigma_E R$. En d'autres termes, la sélection permet de choisir des lignes dans un tableau. Le résultat de la sélection est une nouvelle relation qui a les mêmes attributs que $R$.*

**Définition 4.2.2.** *(Projection)*
*La projection consiste à supprimer les attributs autres que $A_1, ..., A_n$ d'une relation et à éliminer les n-uplets en double apparaissant dans la nouvelle relation. Elle est notée : $\Pi_{(A_1,...,A_n)} R$. En d'autres termes, la projection permet de choisir des colonnes de la relation $R$.*

**Définition 4.2.3.** *(Union)*
*L'union est une opération portant sur deux relations $R_1$ et $R_2$ ayant le même schéma*

*et construisant une autre relation R constituée des n-uplets appartenant à chacune de relations $R_1$ et $R_2$ sans doublons. Elle est notée : $R = R_1 \cup R_2 = \{t : t \in R_1$ ou $t \in R_2\}$.*

**Définition 4.2.4.** *(Intersection)*
*L'intersection est une opération portant sur deux relations $R_1$ et $R_2$ ayant le même schéma et construisant une autre relation R dont les n-uplets sont constitués de ceux appartenant aux deux relations. Elle est notée : $R = R_1 \cap R_2 = \{t : t \in R_1$ et $t \in R_2\}$ où t représente un tuple d'une relation.*

**Définition 4.2.5.** *(Différence)*
*La différence est une opération portant sur deux relations $R_1$ et $R_2$ ayant le même schéma et construisant une autre relation R dont les n-uplets sont constitués de ceux ne se trouvant que dans la relations $R_1$. Elle est notée : $R = R_1 - R_2 = \{t : t \in R_1$ et $t \notin R_2\}$.*

**Définition 4.2.6.** *(Produit cartésien)*
*Le produit cartésien est une opération portant sur deux relations $R_1$ et $R_2$ et qui construit une autre relation regroupant toutes les possibilités de combinaison des occurrences ou tuples des relations $R_1$ et $R_2$. Il est noté : $R_1 \times R_2 = \{(r_i, r_j) : r_i \in R_1$ et $r_j \in R_2\}$.*
*Prenons par exemple deux relations $R_1$ et $R_2$ représentées par le figure 4.1. Le produit cartésien est donné par la relation R présentée par le figure 4.1.*

**Définition 4.2.7.** *(Jointure)*
*L'opérateur jointure est un des opérateurs définis dans le modèle de données relationnelle (Codd [1970]). Il est utilisé pour construire une nouvelle relation R en combinant deux relations $R_1$ et $R_2$. Cette relation R regroupe toutes les possibilités de combiner des occurrences des relations $R_1$ et $R_2$ qui satisfont une expression logique E. La jointure est notée $R_1 \bowtie_E R_2$ tel que E est une condition "jointure" spécifiée.*

En fait, la jointure est un produit cartésien suivi d'une sélection : $R_1 \bowtie_E R_2 = \sigma_E(R_1 \times R_2)$.

**Définition 4.2.8.** *(Theta-jointure)*
*Un theta-jointure est une jointure dans laquelle l'expression logique E est une simple comparaison entre un attribut $A_1$ de la relation $R_1$ et un attribut $A_2$ de la relation $R_2$. Cette jointure est notée $R_1 \bowtie_E R_2$. L'expression logique peut être l'un des opérateurs suivants : $=, \neq, \leq, \geq, <, >$.*

**Définition 4.2.9.** *(Equi-jointure)*
*Une equi-jointure est une theta-jointure dans laquelle l'expression logique E est un test d'égalité entre un attribut $A_1$ de la relation $R_1$ et un attribut $A_2$ de la relation $R_2$. L'equi-jointure est notée : $R_1 \bowtie_{A_1=A_2} R_2$. En fait, l'equi-jointure est un produit cartésien suivi d'une sélection : $R_1 \bowtie_{A_1=A_2} R_2 = \sigma_{A_1=A_2}(R_1 \times R_2)$*

*Relation : $R_1$*

| A | B |
|---|---|
| 1 | 7 |
| 4 | 2 |
| 7 | 3 |

*Relation : $R_2$*

| C | D | E |
|---|---|---|
| 3 | 2 | 5 |
| 7 | 1 | 6 |
| 5 | 6 | 4 |
| 2 | 8 | 1 |

*Relation : R*

| A | B | C | D | E |
|---|---|---|---|---|
| 1 | 7 | 3 | 2 | 5 |
| 1 | 7 | 7 | 1 | 6 |
| 1 | 7 | 5 | 6 | 4 |
| 1 | 7 | 2 | 8 | 1 |
| 4 | 2 | 3 | 2 | 5 |
| 4 | 2 | 7 | 1 | 6 |
| 4 | 2 | 5 | 6 | 4 |
| 4 | 2 | 2 | 8 | 1 |
| 7 | 3 | 3 | 2 | 5 |
| 7 | 3 | 7 | 1 | 6 |
| 7 | 3 | 5 | 6 | 4 |
| 7 | 3 | 2 | 8 | 1 |

FIG. 4.1 – Exemple de produit cartésien

**Définition 4.2.10.** *(Jointure naturelle)*
*Une jointure naturelle est une jointure dans laquelle l'expression logique $E$ est un test d'égalité entre des attributs identiques dans les relations $R_1$ et $R_2$. Dans la relation construite, ces attributs ne sont pas dupliqués mais fusionnés. Cette jointure est notée : $R_1 \bowtie R_2$. On peut préciser explicitement les attributs communs à $R_1$ et $R_2$ sur lesquels porte la jointure : $R_1 \bowtie_{A_1,\ldots,A_2} R_2$.*

Si $R_1$ et $R_2$ ont un seul attribut commun, alors la jointure naturelle est équivalente à un equi-jointure dans laquelle l'attribut de la relation $R_1$ et l'attribut de la relation $R_2$ sont justement deux attributs (variables) identiques. Lorsque nous désirons effectuer une jointure naturelle entre les relations $R_1$ et $R_2$ sur un attribut commun $A_1 = A_2$, nous devons écrire $R_1 \bowtie_{A_1=A_2} R_2$.
Reprenons les deux relations $R_1$ et $R_2$ représentées par le figure 4.1 et supposons que $age = ageC$. En appliquant la jointure naturelle, nous obtenons la relation $R = R_1 \bowtie R_2$ ou $R = R_1 \bowtie_{B=C} R_2$. Cette relation $R$ est représentée par le figure 4.2.
D'autres types de jointures et plusieurs algorithmes permettant exécuter ces jointures sont présentées dans (Mishra and Eich [1992]).

*Relation : $R_1$*

| A | B |
|---|---|
| 1 | 7 |
| 4 | 2 |
| 7 | 3 |

*Relation : $R_2$*

| C | D | E |
|---|---|---|
| 3 | 2 | 5 |
| 7 | 1 | 6 |
| 5 | 6 | 4 |
| 2 | 8 | 1 |

*Relation : R*

| A | B | C | D | E |
|---|---|---|---|---|
| 1 | 7 | 7 | 1 | 6 |
| 4 | 2 | 2 | 8 | 1 |
| 7 | 3 | 3 | 2 | 5 |

FIG. 4.2 – Exemple de jointure naturelle

### 4.2.2 Propagation de valeurs et equi-jointure

La propagation de valeurs (Apt [2003]) peut être considérée comme un outil pour concevoir les relations de redondance analytique (**RRA**s). Cette propagation consiste à calculer la valeur d'une variable $v_i$ en utilisant des observations (mesures, commande,...) au sein d'une contrainte $k_i$. Cette valeur sera propagée à la variable $v_i$ dans la contrainte $k_j$. Par conséquent, la même valeur de $v_i$ doit être présentée dans les deux contraintes $k_i$ et $k_j$. Nous appelons cela "la condition de propagation".
Cette principe est représentée par la figure 4.3.
La figure 4.3 montre comment la valeur de la variable $v_2$ est calculée au sein de la

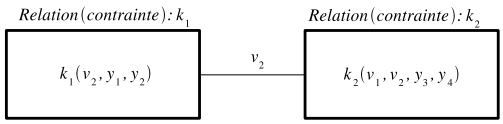


FIG. 4.3 – Propagation de valeurs

contrainte $k_1$ en utilisant les mesures de $y_1$ et $y_2$. La valeur résultant sera propagée à la variable $v_2$ au sein de la contrainte $k_2$ afin de calculer la valeur de $v_1$. Par conséquent, la variable $v_2$ sera éliminée dans les deux contraintes $k_1$ et $k_2$.

Maintenant, nous allons montrer le point commun entre la propagation de valeurs et l'opérateur equi-jointure.
La présence de la même valeur dans les contraintes $k_1$ et $k_2$ ("la condition de propagation") correspond à une equi-joiture et la propagation de la valeur de $v_2$ résultant de

la contrainte $k_1$ à la variable $v_2$ dans la contrainte $k_2$ correspond à une projection sur l'equi-jointure des deux contraintes $k_1$ et $k_2$ : $\Pi_{(var(k_1)\cup var(k_2)\setminus\{v_2\})}(k_1 \bowtie_{v_2\in k_1=v_2\in k_2} k_2)$ tel que $var(k_1)$ et $var(k_2)$ représentent les variables intervenant dans les contraintes $k_1$ et $k_2$ (c'est-à-dire, les attributs de $k_1$ et $k_2$).
Nous allons montrer ce résultat en analysant l'exemple ci-dessous.
Nous avons deux relations (contraintes) : $k_1$ dont les attributs sont : $(v_2, y_1, y_2)$ et $k_2$ dont les attributs sont : $(v_1, v_2, y_3, y_4)$. Supposons que ces deux relations sont données par les tableaux 4.4.
L'equi-jointure correspond à un produit cartésien suivi d'une sélection $(k_1 \bowtie_{v_2\in k_1=v_2\in k_2}$

| $y_1$ | $y_2$ | $v_2$ |
|---|---|---|
| 1 | 2 | 1 |
| 3 | 4 | 3 |
| 9 | 3 | 6 |

| $v_2$ | $v_1$ | $y_3$ | $y_4$ |
|---|---|---|---|
| 1 | 5 | 4 | 2 |
| 2 | 7 | 3 | 3 |
| 6 | 1 | 2 | 7 |

FIG. 4.4 – Deux relations (contraintes) $k_1$ et $k_2$

$k_2) = \sigma_{v_2\in k_1=v_2\in k_2}(R_1 \times R_2)$. Cette equi-jointure est représentée par la tableau 4.5.
Nous remarquons que, dans cet equi-jointure, la valeur de la variable $v_2 \in k_1$ est la

| $y_1$ | $y_2$ | $v_2$ | $v_2$ | $v_1$ | $y_3$ | $y_4$ |
|---|---|---|---|---|---|---|
| 1 | 2 | 1 | 1 | 5 | 4 | 2 |
| 9 | 3 | 6 | 6 | 1 | 2 | 7 |

FIG. 4.5 – L'equi-jointure de $k_1$ et $k_2$

même de la variable $v_2 \in k_2$. Cela correspond à la condition de propagation.

La projection de l'equi-jointure résultant par rapport à $(var(k_1) \cup var(k_2) \setminus \{v_2 \in k_1 \cup v_2 \in k_2\})$ est donnée par le tableau 4.6.
Nous remarquons que la variable $v_2$ n'est plus présente dans ce tableau, cette projection correspond à l'idée de propager la valeur de $v_2$ à la variable $v_3$ (autrement dit, éliminer la variable $v_2$ des contraintes $k_1$ et $k_2$).

Nous concluons que l'opérateur jointure peut être utilisé comme la propagation de valeurs pour concevoir les relations de redondance analytique.
Dans la suite, nous utilisons la notation $s_i \bowtie_v s_j$ pour représenter l'opérateur jointure permettant de propager la variable $v$ entre deux structures de contraintes $s_i$ et $s_j$. Cet opérateur est donné par : $s_i \bowtie_v s_j = \Pi_{(var(s_i)\cup var(s_j)\setminus\{v\})}(s_i \bowtie_{v\in s_i=v\in s_j} s_j)$ tel que $s_i$ et

| $y_1$ | $y_2$ | $v_1$ | $y_3$ | $y_4$ |
|---|---|---|---|---|
| 1 | 2 | 5 | 4 | 2 |
| 9 | 3 | 1 | 2 | 7 |

FIG. 4.6 – La projection de l'equi-jointure de l'exemple

$s_j$ représentent les structures des contraintes $k_i$ et $k_j$, $var(s_i)$ et $var(s_j)$ représentent les ensembles de variables des deux structures $s_i$ et $s_j$.

## 4.3 Formulation de problème

La résolution d'un problème de diagnostic est généralement décomposée en deux étapes consécutives. La génération des symptômes, nommée aussi "détection de défauts" dans la communauté d'Automatique, et l'analyse diagnostique nommée parfois "localisation de défauts". La première étape consiste en des tests de consistance ou (**RRA**s) s'appuyant sur des sous système testables **SST**. Par conséquent, cette étape nécessite la génération des SST. Une approche pour générer les sous systèmes testables est proposée dans ce chapitre. Cette approche est présentée dans (Ploix et al. [2008])
Dans la section 2.6, nous avons montré que plusieurs réalisations d'une contrainte $k$ peuvent être équivalentes. Supposant que $\kappa$ est une réalisation de $V \backslash v$ vers $v$ tel que $V$ est l'ensemble des variables apparaissant dans la contrainte $k$. Nous avons montré que la structure modélisant une contrainte $k$ peut être dénotée : $s(k) = \llcorner V^-, V^+ \lrcorner$. $\forall v \in V^+$, il existe une réalisation de $\mathcal{K}(k)$ menant à $v : \llcorner (V^- \cup V^+) \backslash \{v\}, \{v\} \lrcorner$
Pour simplifier, les notations suivantes sont adoptées : $\llcorner \emptyset, V^+ \lrcorner = \llcorner V^+ \lrcorner$ et si $S$ est un ensemble de structures, $var(S) = \bigcup_{s_i \in S} var(s_i)$. Finalement, une structure vide $s$ est une structure satisfaisant : $var(S) = \emptyset$. Elle est notée : $s = \llcorner \emptyset \lrcorner$.

Pour la conception des sous-systèmes testables, certaines structures de contraintes sont particulièrement utiles parce qu'elles permettent de modéliser ce qui est connu dans un système c'est-à-dire les informations liées aux actionneurs et aux capteurs. Ces structures sont nommées "structures terminales".

**Définition 4.3.1.** *Une structure terminale $s$ est une structure satisfaisant $card(var(s)) = 1$.*

Elle implique généralement un flot de données et modélise le fait qu'une trajectoire peut être attribuée à une variable. Les approches structurelles induisent des sur-

estimations : prenons par exemple un système modélisé par les contraintes suivantes :

$$\begin{aligned} k_1 &: x_1 + x_2 - x_3 = 0 \\ k_2 &: x_2 - x_3 + x_4 = 0 \\ k_3 &: x_1 = \tilde{x_1} \\ k_4 &: x_2 = \tilde{x_2} \\ k_5 &: x_3 = \tilde{x_3} \\ k_6 &: x_4 = \tilde{x_4} \end{aligned} \tag{4.3.1}$$

Supposons maintenant que la variable $x_2$ est éliminée entre les contraintes $k_1$ et $k_2$, la contrainte résultant est : $x_1 - x4 = 0$. En remplaçant les variables $x_1$ et $x_4$ par leurs flots de données, on obtient la relation de redondance suivante : $\tilde{x_1} - \tilde{x_4} = 0$. Donc, le test est composé des contraintes $(k_1, k_2, k_3, k_6)$.
Le système ci-dessus peut être modélisé par les structures suivantes :

$$\begin{aligned} s(k_1) &= \llcorner\{x_1, x_2, x_3\}\lrcorner \\ s(k_2) &= \llcorner\{x_2, x_3, x_4\}\lrcorner \\ s(k_3) &= \llcorner\{x_1\}\lrcorner \\ s(k_4) &= \llcorner\{x_2\}\lrcorner \\ s(k_5) &= \llcorner\{x_3\}\lrcorner \\ s(k_6) &= \llcorner\{x_4\}\lrcorner \end{aligned} \tag{4.3.2}$$

Supposons que la variable $x_2$ doit être éliminée entre les structures $s(k_1)$ et $s(k_2)$ en utilisant l'opérateur jointure, nous obtenons : $s(k_1) \bowtie_{x_2} s(k_2) = \llcorner\{x_1, x_3, x_4\}\lrcorner$.
La variable $x_1$ peut être éliminée entre les structures $s(k_1) \bowtie_{x_3} s(k_2)$ et $s(k_3)$, nous obtenons : $(s(k_1) \bowtie_{x_3} s(k_2)) \bowtie_{x_1} s(k_3) = \llcorner\{x_3, x_4\}\lrcorner$.
La variable $x_3$ peut être éliminée entre les structures $(s(k_1) \bowtie_{x_3} s(k_2)) \bowtie_{x_1} s(k_3)$ et $s(k_5)$, Nous obtenons : $((s(k_1) \bowtie_{x_2} s(k_2)) \bowtie_{x_1} s(k_3)) \bowtie_{x_3} s(k_5) = \llcorner\{x_4\}\lrcorner$.
La variable $x_4$ peut être éliminée entre les structures $((s(k_1) \bowtie_{x_2} s(k_2)) \bowtie_{x_1} s(k_3)) \bowtie_{x_3} s(k_5)$ et $s(k_6)$, nous obtenons : $(((s(k_1) \bowtie_{x_2} s(k_2)) \bowtie_{x_1} s(k_3)) \bowtie_{x_3} s(k_5)) \bowtie_{x_4} s(k_6) = \llcorner\{\emptyset\}\lrcorner$. Cette formule représente un test composé des contraintes $(k_1, k_2, k_3, k_5, k_6)$.
Par conséquent, nous remarquons que le test résultant du raisonnement structurel contient une contrainte qui n'est pas nécessaire. Cela signifie que le test résultant de l'élimination des variables entre les contraintes 4.3.1 est inclus dans le test résultant de l'utilisation de l'opérateur jointure entre les structures de ces contraintes 4.3.2. Cette différence est due à la suppression de la variable $x_3$ en même temps que la variable $x_2$ entre les contraintes $k_1$ et $k_2$.
Du fait de ces sur-estimations induites par le raisonnement structurel, plusieurs définitions utiles sont introduites.

**Définition 4.3.2.** *Une structure $s_1$ enveloppe une autre structure $s_2$ si $V(s_1) \supseteq V(s_2)$ et $V^+(s_1) \supseteq V^+(s_2)$. Elle est dénotée $s_1 \supseteq s_2$*

Pour formaliser les propagations de valeurs, un opérateur jointure est défini. Par nécessité, la notion de variable propageable doit être introduite.

**Définition 4.3.3.** *Supposons que $s_1$ et $s_2$ sont deux structures liées à $\kappa_1 \in \mathcal{K}(k_1)$ et $\kappa_2 \in \mathcal{K}(k_2)$. La propagation d'une variable $v$ de $s_1 = s_1(\kappa_1)$ vers $s_2 = s_2(\kappa_2)$ est possible seulement si $v \in V_1^+$ et $v \in V_2$. La variable $v$ est qualifiée de propageable entre $s_1$ et $s_2$ si elle est propagable de $s_1$ vers $s_2$ ou propagable de $s_2$ vers $s_1$. Par extension, $v$ est dite propageable entre $\kappa_1$ et $\kappa_2$ et aussi entre $\mathcal{K}(k_1)$ et $\mathcal{K}(k_2)$.*

L'opérateur jointure peut maintenant être défini.

**Définition 4.3.4.** *Supposons que $s_1$ et $s_2$ soient deux structures. L'opérateur jointure dénoté $\bowtie_v$, où $v$ est une variable propageable entre $s_1$ et $s_2$, est défini seulement dans les trois situations suivantes :*

- *si $v \in V_1^+ \cap V_2^-$ alors*
  $s_1 \bowtie_v s_2 = \llcorner(V_1^- \cup V_1^+ \cup V_2^-) \setminus (V_2^+ \cup \{v\}), V_2^+\lrcorner$
- *si $v \in V_2^+ \cap V_1^-$ alors*
  $s_1 \bowtie_v s_2 = \llcorner(V_1^- \cup V_2^+ \cup V_2^-) \setminus (V_1^+ \cup \{v\}), V_1^+\lrcorner$
- *si $v \in V_1^+ \cap V_2^+$ alors*
  $s_1 \bowtie_v s_2 = \llcorner(V_1^- \cup V_2^-) \setminus (V_1^+ \cup V_2^+), (V_1^+ \cup V_2^+) \setminus \{v\}\lrcorner$

*Si une formule $s_1 \bowtie_v s_2$ satisfait un de trois points ci-dessus, elle est dite évaluable*

L'utilisation de cet opérateur permet de formaliser une propagation entre deux contraintes c'est-à-dire le fait qu'une valeur d'une variable $v$ donnée peut être déduite à partir d'une réalisation d'une contrainte et introduite dans une autre. Un lien est ainsi créé entre deux contraintes : il correspond partiellement à l'opérateur jointure de l'algèbre relationnel.

**Théorème 4.3.1.** *Supposons que $\mathcal{K}_1$ et $\mathcal{K}_2$ sont deux contraintes qui représentent deux ensembles de réalisations. Alors une structure enveloppante de la contrainte resultant de la propagation d'une variable $v$ entre $\mathcal{K}_1$ et $\mathcal{K}_2$ peut être obtenue en appliquant l'opérateur $\bowtie_v$ entre $s(\mathcal{K}_1)$ et $s(\mathcal{K}_2)$*

*Démonstration.* Notons $s(\mathcal{K}_1) = \llcorner V_1^-, V_1^+\lrcorner$, $s(\mathcal{K}_2) = \llcorner V_2^-, V_2^+\lrcorner$ et $\mathcal{K}$, l'ensemble des réalisations équivalentes résultant de la propagation de $v$ entre $\mathcal{K}_1$ et $\mathcal{K}_2$. $\mathcal{K}$ dépend de la présence d'autres variables propageables. Si $v$ est la seule variable propageable entre $\mathcal{K}_1$ et $\mathcal{K}_2$, la structure exacte de $\mathcal{K}$ peut être déduite, sinon, seule une structure enveloppante peut être trouvée.

Considérons la situation suivante où il y a une seule variable propageable $v \in V_1^+ \cap V_2^-$. $\exists \kappa_{1,v} \in \mathcal{K}_1$ tel que $v = \kappa_{1,v}((V_1^+ \backslash \{v\}) \cup V_1^-)$ et $v$ peut être propagé vers $\mathcal{K}_2$. Si $V_2^+ \neq \emptyset$, $\forall w \in V_2^+$, $\exists \kappa_{2,w} \in \mathcal{K}_2 / w = \kappa_{2,w}((V_2^+ \backslash \{w\}) \cup V_2^-)$. Puisque $v \in V_2^-$, il existe $\kappa_w$ tel que : $w = \kappa_w(((V_2^+ \setminus \{w\}) \cup (V_2^- \setminus \{v\})) \cup ((V_1^+ \setminus \{v\}) \cup (V_1^-))$

Parce que $v$ est la seule variable propageable, $w \notin (V_1^+ \cup V_1^-)$ et donc, $\kappa_w$ est une réalisation. Comme ce résultat est vrai $\forall w \in V_2^+$, la structure de contrainte résultant est : $\llcorner(V_1^- \cup V_1^+ \cup V_2^-) \setminus (V_2^+ \cup \{v\}), V_2^+ \lrcorner$. Si $V_2^+$ est vide, le résultat précédent reste vrai : $\llcorner(V_1^- \cup V_1^+ \cup V_2^-) \setminus \{v\}), \emptyset \lrcorner$. Cela conduit à : $\forall v \in V_1^+ \cap V_2^-$, $s(\mathcal{K}) = s(\mathcal{K}_1) \bowtie_v s(\mathcal{K}_2)$.
Si la seule variable propageable satisfait $v \in V_1^+ \cap V_2^+$, des variables déductibles additionnelles peuvent être trouvées. En effet,

- si $V_2^+ \setminus \{v\} \neq \emptyset, \exists \kappa_{1,v} \in \mathcal{K}_1$ tel que $v = \kappa_{1,v}((V_1^+ \setminus \{v\}) \cup V_1^-)$ et $v$ peut être propagé vers $\mathcal{K}_2 : \forall w \in V_2^+ \setminus \{v\}$, $\exists \kappa_{2,w} \in \mathcal{K}_2 / w = \kappa_{2,w}((V_2^+ \setminus \{w\}) \cup V_2^-)$. Puisque $v \in V_2^+$, on déduit :
  $\exists \kappa_w / w = \kappa_w(((V_2^+ \cup V_1^+) \setminus \{v, w\}) \cup V_1^- \cup V_2^-)$
- si $V_1^+ \setminus \{v\} \neq \emptyset, \exists \kappa_{2,v} \in \mathcal{K}_2$ tel que $v = \kappa_{2,v}((V_2^+ \setminus \{v\}) \cup V_2^-)$ et $v$ peut être propagé vers $\mathcal{K}_1 : \forall w \in V_1^+ \setminus \{v\}$, $\exists \kappa_{1,w} \in \mathcal{K}_1 / w = \kappa_{1,w}((V_1^+ \setminus \{w\}) \cup V_1^-)$. Puisque $v \in V_1^+$, on déduit : $\exists \kappa_w / w = \kappa_w(((V_1^+ \cup V_2^+) \setminus \{v, w\}) \cup V_1^- \cup V_2^-)$

Parce que $v$ est la seule variable propageable, les $\kappa_w$ sont des réalisations et la structure de la contrainte résultant est donnée par : $\llcorner(V_1^- \cup V_2^- \setminus (V_1^+ \cup V_2^+), (V_1^+ \cup V_2^+) \setminus \{v\} \lrcorner$. Par conséquent, $\forall v \in (V_1^+ \cap V_2^+)$, $s(\mathcal{K}) = s(\mathcal{K}_1) \bowtie_v s(\mathcal{K}_2)$. Ces résultats restent vrais si $V_1^+$ ou $V_2^+$ sont vides.

Quand il y a plusieurs variables propagables, il n'est plus possible de montrer que les fonctions $k_w$ sont des réalisations et les résultats précédents deviennent :

- $\forall v \in V_1^+ \cap V_2^-$, $s(\mathcal{K}) \subseteq s(\mathcal{K}_1) \bowtie_v s(\mathcal{K}_2)$
- $\forall v \in V_1^+ \cap V_2^+$, $s(\mathcal{K}) \subseteq s(\mathcal{K}_1) \bowtie_v s(\mathcal{K}_2)$

□

**Remarque 4.3.1.** Selon la définition, l'opérateur jointure n'est ni commutatif ni associatif. En effet, considérons un système modélisé par les trois contraintes suivante :

$$k_1 : x_2 = x_1 + 1$$
$$k_2 : x_1 = \tilde{x_1}$$
$$k_3 : x_2 = \tilde{x_2}$$

Les structures de ces contraintes sont : $s_1 = \llcorner\{x_2\}, \{x_1\} \lrcorner$, $s_2 = \llcorner\{x_1\} \lrcorner$ et $s_3 = \llcorner\{x_2\} \lrcorner$. Ces structures satisfont : $x_2 \in V^-(s_1) \cap V^+(s_3)$, $x_2 \notin V(s_2)$, $x_1 \in V^+(s_1) \cap V^+(s_2)$. Alors, la formule $(s_1 \bowtie_{x_1} s_2) \bowtie_{x_2} s_3$ peut être évaluée tandis que $(s_2 \bowtie_{x_2} s_3)$ ne le peut pas parce que la variable $x_2$ n'appartient pas à la structure $s_2$. Par conséquent, $(s_1 \bowtie_{x_1} s_2) \bowtie_{x_2} s_3 \neq s_1 \bowtie_{x_1} (s_2 \bowtie_{x_2} s_3)$

Dans cette section, nous avons introduit la notion de structure de contraintes et l'opérateur jointure qui modélise les propagations de valeurs entre des structures. Ces outils peuvent être utilisés pour concevoir des sous-systèmes testables pour tout type d'applications. La manière d'utiliser ces outils est expliquée dans la section suivante.

## 4.4 Conception des sous-systèmes testables

La conception des sous systèmes testables consiste à combiner la description présentée dans la section 2.6 :

$$\begin{aligned} mode_1(c_1) \wedge mode_2(c_2) \rightarrow \\ V_2 = \kappa_i(t, V_1, val(V_3)); \\ V_1 \in dom(t, V_1), V_2 \in dom(t, V_2) \end{aligned} \tag{4.4.1}$$

c'est-à-dire, à combiner des modes sous forme de conjonction et combiner les contraintes associées en éliminant les variables.
Avant d'aborder la conception de sous-systèmes testables, nous allons introduire quelques concepts.

### 4.4.1 Notion de formule de propagation

Un test résulte de propagations consécutives qui peuvent être modélisées par une formule de propagation, par exemple : $f = (((s_1 \bowtie_{v_1} s_2) \bowtie_{v_2} s_3) \bowtie_{v_3} (s_4 \bowtie_{v_4} s_2))$. Une formule est composée des sous-formules comportant l'opérateur jointure, où les formules élémentaires sont des structures. Grâce à cet opérateur, une formule de propagation $f$ peut être évaluée comme une structure $s(f)$. L'ensemble de toutes les formules est dénoté : $F$.

**Définition 4.4.1.** *Une formule est qualifiée d'évaluable si toutes ses sous-formules sont aussi évaluables.*

**Définition 4.4.2.** *Le support d'une formule $f \in F$, dénoté $\sigma(f)$, est l'ensemble de toutes les structures impliquées dans cette formule.*

**Définition 4.4.3.** *Le degré d'une formule $f \in F$, dénoté $d(f)$, est égal au nombre de fois où l'opérateur jointure apparaît dans la formule : il représente le nombre de propagations élémentaires.*

**Définition 4.4.4.** *Deux formules $f_1$ et $f_2$ sont structurellement équivalentes si $\sigma(f_1) = \sigma(f_2)$ et si $var(s(f_1)) = var(s(f_2))$ c'est à dire elles ont les mêmes contraintes et les mêmes variables. Elles sont dénotées : $f_1 \sim f_2$.*

Au sein d'un même **SST**, une variable donnée ne peut être instanciée qu' une fois seulement. Cependant, dans une même formule, une variable peut apparaître plusieurs fois et peut de ce fait être instanciée de différentes manières. Dans ce type de situation, il existera nécessairement une formule plus simple au sens de la définition qui suit, où les variables ont été instanciées seulement une fois.

**Définition 4.4.5.** *Une formule $f_1$ est plus simple qu'une formule $f_2$ si :*
- $\sigma(f_1) \subset \sigma(f_2)$ *et* $var(s(f_1)) \subset var(s(f_2))$
- *ou si* $f_1 \sim f_2$ *et* $d(f_1) < d(f_2)$
- *ou si* $f_1 \sim f_2$ *et* $d(f_1) = d(f_2)$ *et* $var^+(s(f_1)) \subset var^+(s(f_2))$

On note : $f_1 \prec f_2$. On dit aussi que $f_2$ sur-estime $f_1$. Une formule de propagations testable $f \in F$ est une formule évaluable qui mène à une structure vide. Une formule testable de propagations est minimale sur un ensemble de structures $S$ s'il n'existe aucune formule plus simple sur $S$. Elle est nommée formule de propagation testable minimale **MTPF** sur $S$.
Selon la définition de l'opérateur jointure, toutes les formules correspondent à des structures enveloppantes de contraintes qui peuvent être trouvées en combinant les contraintes élémentaires d'un problème de diagnostic. D'une manière générale, les formules peuvent sur-estimer les propagations qui mènent à un test. Heureusement, il est facile de vérifier si toutes les contraintes liées au support $\sigma(f)$ d'un **MTPF** $f$ ont été utilisées pendant la conception d'un test. Si certaines contraintes ne sont pas nécessaire à la construction d'un test, les modes associés doivent être retirés des symptômes correspondant au test.

### 4.4.2 Caractéristique de l'opérateur jointure

En appliquant l'opérateur jointure $\bowtie_v$ sur une variable $v$ appartenant à deux structures $s(f_i)$ et $s(f_j)$, nous obtenons une nouvelle structure : $s(f) = s(f_i \bowtie_v f_j)$ alors que les deux structures de départ peuvent être supprimées. En bilan, il y a une formule de moins dans l'ensemble des formules utiles. En appliquant l'opérateur jointure $\bowtie_v$ sur une variable $v$ appartenant à trois structures $s(f_i)$, $s(f_j)$ et $s(f_k)$, trois nouvelles structures sont obtenues : $s(f_1) = s(f_i \bowtie_v f_j)$, $s(f_2) = s(f_i \bowtie_v f_k)$ et $s(f_3) = s(f_j \bowtie_v f_k)$. Au total, le nombre de formules ne change pas. En appliquant l'opérateur jointure $\bowtie_v$ sur une variable $v$ appartenant à quatre structures $s(f_i)$, $s(f_j)$, $s(f_k)$ et $s(f_l)$, six nouvelles structures sont obtenues : $s(f_1) = s(f_i \bowtie_v f_j)$, $s(f_2) = s(f_i \bowtie_v f_k)$, $s(f_3) = s(f_i \bowtie_v f_l)$, $s(f_4) = s(f_j \bowtie_v f_k)$, $s(f_5) = s(f_j \bowtie_v f_l)$, $s(f_6) = s(f_k \bowtie_v f_l)$ alors que quatre disparaissent.

Le schéma 4.4.2 montre la liaison entre le nombre des structures avant et après l'application de l'opérateur jointure (propagation) à une même variable.
Prenons un système modélisé par les structures suivanntes :

$$\begin{aligned} s(f_1) &= \llcorner\{x_1, x_2, x_3\}\lrcorner \\ s(f_2) &= \llcorner\{x_1, x_2, x_3\}\lrcorner \\ s(f_3) &= \llcorner\{x_2, x_3\}\lrcorner \\ s(f_4) &= \llcorner\{x_3\}\lrcorner \end{aligned} \tag{4.4.2}$$

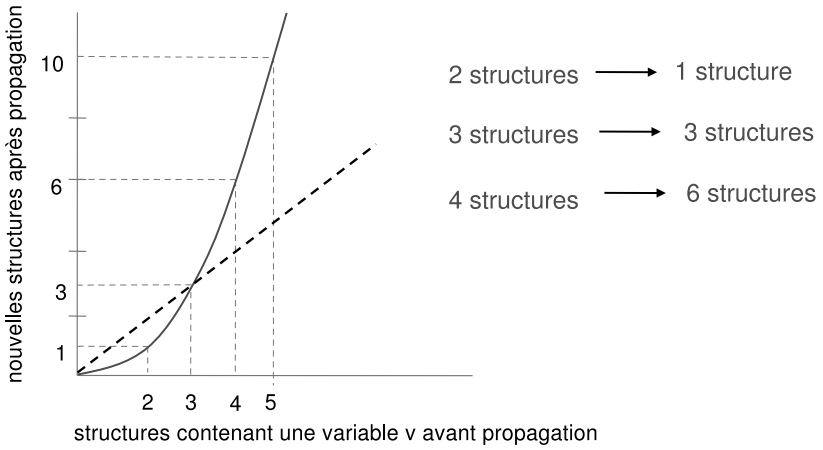


FIG. 4.7 – Influence de l'opérateur jointure sur le nombre de formules

- L'application de l'opérateur jointure $\bowtie_{x_1}$ entre les structures $s(f_1)$ et $s(f_2)$ afin d'enlever la variable $x_1$ donne la nouvelle structure suivante : $s(f_1 \bowtie_{x_1} f_2) = \llcorner\{x_2, x_3\}\lrcorner$ alors que les formules $f_1$ et $f_2$ sont supprimées.
- L'application de l'opérateur jointure $\bowtie_{x_2}$ entre les structures $s(f_1)$, $s(f_2)$ et $s(f_3)$ afin d'enlever la variable $x_2$ donne les quatre nouvelles structures suivantes : $s(f_1 \bowtie_{x_2} f_2) = \llcorner\{x_1, x_3\}\lrcorner$, $s(f_1 \bowtie_{x_2} f_3) = \llcorner\{x_1, x_3\}\lrcorner$, $s(f_2 \bowtie_{x_2} f_3) = \llcorner\{x_1, x_3\}\lrcorner$ alors que les formules $f_1$, $f_2$ et $f_3$ sont supprimées.
- L'application de l'opérateur jointure $\bowtie_{x_3}$ entre les structures $s(f_1)$, $s(f_2)$, $s(f_3)$ et $s(f_4)$ afin d'enlever la variable $x_1$ donne les six nouvelles structures suivantes : $s(f_1 \bowtie_{x_2} f_2) = \llcorner\{x_1, x_2\}\lrcorner$, $s(f_1 \bowtie_{x_2} f_3) = \llcorner\{x_1, x_2\}\lrcorner$, $s(f_1 \bowtie_{x_2} f_4) = \llcorner\{x_1, x_2\}\lrcorner$, $s(f_2 \bowtie_{x_2} f_3) = \llcorner\{x_1, x_2\}\lrcorner$, $s(f_2 \bowtie_{x_2} f_4) = \llcorner\{x_1, x_2\}\lrcorner$ et $s(f_3 \bowtie_{x_2} f_4) = \llcorner\{x_2\}\lrcorner$ alors que les formules $f_1$, $f_2$, $f_3$ et $f_4$ sont supprimées.

Afin de diminuer le complexité du calcul des sous systèmes testables, nous proposons un algorithme basant sur la caractéristique de l'opérateur jointure ci-dessus.

### 4.4.3 Algorithme pour la conception des sous systèmes testables

Pour la conception de sous-systèmes testables (**SST**s), tous les **MTPF**s possibles doivent être trouvés parce que les **SST**s sont donnés par le support des **MTPF**s. Le principe est de propager les valeurs itérativement jusqu'à ce que les **MTPF**s soient trouvés. Mais, contrairement à la propagation de valeurs, une propagation peut être envisagée même si les variables relatives n'ont pas été instanciées. Afin de réduire le nombre de

calculs, les propagations liées aux variables présentées dans le moins de structures sont réalisées d'abord. Cette caractéristique représente le point fort de notre algorithme permettant de diminuer la complexité de la conception des **SST**s.

L'ensemble des formules est représenté par la lettre $F$ et les structures correspondant par la lettre $S(F) = \{s(f); f \in F\}$. Considérons un ensemble de contraintes $K$ intervenant dans un problème de diagnostic et $F_0$ est l'ensemble des structures correspondant à $K$, qui sont aussi des formules élémentaires. $S(F_0)$ représente les structures correspondant à $F_0$.
Le nombre de structures d'un ensemble $S(F)$ où une variable $v$ apparaît, est nommé ordre de $v$ dans $\sigma(F)$. Il est noté $o_F(v)$.

Tout d'abord, une propagation ne peut pas être réalisée quand une variable apparaît seulement une fois dans les structures $S(F_0)$. Les structures contenant ces variables, et leur formule correspondante peuvent être enlevées parce qu'elles ne peuvent pas être impliquées dans des **MTPF**s. Néanmoins, quand des structures sont enlevées, il est possible de trouver de nouvelles variables qui apparaissent une seule fois. Cette procédure sera donc répétée jusqu'à ce que plus aucune variable n'apparaisse qu'une seule fois. Cette étape est une étape de nettoyage et elle est formalisée par l'algorithme 1. Prenons par

**Algorithm 1** Supprimer les structures inutiles

**Require:** $F_0$, un ensemble de formules
  $F \leftarrow F_0$
  **repeat**
    $V \leftarrow \{v \in var(S(F)); o_F(v) = 1\}$
    $F \leftarrow \{f \in F; var(s(f)) \cap V = \emptyset\}$
  **until** $V = \emptyset$
  **return** $F$

exemple le système modélisé par l'ensemble de contraintes $K$ suivant :

$$\begin{aligned} k_1 &: x_2 = x_1 + 1 \\ k_2 &: x_3 = x_2 + 3 \\ k_3 &: x_4 = x_3 + 4x_5 \\ k_4 &: x_5 = x_3 + 2 \\ k_5 &: x_3 = \tilde{x_3} \\ k_6 &: x_4 = \tilde{x_4} \\ k_7 &: x_5 = \tilde{x_5} \end{aligned} \tag{4.4.3}$$

Ce système peut être modélisé par l'ensemble des structures $F_0$ correspondant à $K$ :

$$s(f_1) \quad = \quad \llcorner\{x_1, x_2\}\lrcorner$$

$$
\begin{aligned}
s(f_2) &= \llcorner\{x_2, x_3\}\lrcorner \\
s(f_3) &= \llcorner\{x_3, x_4, x_5\}\lrcorner \\
s(f_4) &= \llcorner\{x_3, x_5\}\lrcorner \\
s(f_5) &= \llcorner\{x_3\}\lrcorner \\
s(f_6) &= \llcorner\{x_4\}\lrcorner \\
s(f_7) &= \llcorner\{x_5\}\lrcorner
\end{aligned}
$$

Nous remarquons que la variable $x_1$ apparaît seulement une fois dans les structures du système. La structure contenant cette variable $s(f_1)$ et sa formule $f_1$ seront enlevées. Par conséquent, la variable $x_2$ apparaît seulement une fois dans les structures restantes du système. La structure contenant cette variable $s(f_2)$ et sa formule $f_2$ seront enlevées. Il n'y a plus de variable qui apparaissent seulement une fois dans les structures restantes.

L'ensemble nettoyé résultant de l'algorithme 1 est nommé $F_1$. Les propagations peuvent être maintenant réalisées selon les ordres des variables. Les variables d'ordre le plus faible sont sélectionnées d'abord. Supposons que $v$ est une de ces variables. Toutes les formules contenant la variable $v$ sont sélectionnées et en utilisant l'opérateur jointure, des nouvelles formules évaluables sont déduites et ajoutées à l'ensemble de formules courantes. Les formules qui sur-estiment d'autres sont enlevées de même que les formules qui contiennent $v$ et cela constitue l'intérêt de cet algorithme par rapport aux méthodes existantes. Cette procédure est répétée jusqu'à ce que toutes les variables aient été considérées. Les structures restantes sont alors des structures vides et ainsi, toutes les **MTPF**s ont été trouvés. Cette procédure est synthétisée par l'algorithme 2.
Pour comprendre cet algorithme, nous reprenons l'exemple ci-dessus. L'ensemble $F_1$ est

**Algorithm 2** Conception de **MTPF**s

**Require:** $F_1$, un ensemble de formules
  $F \leftarrow F_1$
  **while** $var(S(F)) \neq \emptyset$ **do**
    **select** $v \in var(S(F))$ **such that** $o_F(v) \leq o_F(v_i), \forall v_i \in var(S(F))$
    $F' \leftarrow \{f / v \in var(s(f))\}$
    $F'' \leftarrow \{f_i \bowtie_v f_j; (f_i, f_j) \in F'^2,\ i \neq j,\ f_i \bowtie_v f_j\ evaluable\}$
    $F \leftarrow (F \backslash F') \cup F''$
    $F \leftarrow F \backslash \{f \in F; \exists f_i \in F, f_i \neq f, f \succ f_i\}$
  **end while**
  **return** $F$

composé des formules suivantes :

$$
\begin{aligned}
s(f_3) &= \llcorner\{x_3, x_4, x_5\}\lrcorner \\
s(f_4) &= \llcorner\{x_3, x_5\}\lrcorner
\end{aligned}
$$

$$\begin{aligned} s(f_5) &= \llcorner\{x_3\}\lrcorner \\ s(f_6) &= \llcorner\{x_4\}\lrcorner \\ s(f_7) &= \llcorner\{x_5\}\lrcorner \end{aligned}$$

La variable $x_4$ a le plus bas ordre dans $F_1$. Elle est sélectionnée d'abord. Ensuite, l'opérateur jointure $\bowtie_{x_4}$ est utilisé entre toutes les formules où cette variable est impliquée. Alors, en enlevant les formules contenant la variable $x_4$, l'ensemble de formules suivantes $F$ est obtenu :

$$\begin{aligned} s(f_3 \bowtie_{x_4} f_6) &= \llcorner\{x_3, x_5\}\lrcorner \\ s(f_4) &= \llcorner\{x_3, x_5\}\lrcorner \\ s(f_5) &= \llcorner\{x_3\}\lrcorner \\ s(f_7) &= \llcorner\{x_5\}\lrcorner \end{aligned}$$

Les variables $x_3$ et $x_5$ ont le même ordre. Choisissons la variable $x_3$ et utilisons l'opérateur jointure $\bowtie_{x_3}$. En enlevant les formules contenant la variable $x_3$, nous obtenons :

$$\begin{aligned} s((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_4) &= \llcorner\{x_5\}\lrcorner \\ s((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5) &= \llcorner\{x_5\}\lrcorner \\ s(f_4 \bowtie_{x_3} f_5) &= \llcorner\{x_5\}\lrcorner \\ s(f_7) &= \llcorner\{x_5\}\lrcorner \end{aligned}$$

Choisissons la variable $x_5$ et utilisons l'opérateur jointure $\bowtie_{x_5}$. En enlevant les formules contenant la variable $x_5$, nous obtenons :

$$\begin{aligned} s(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_4) \bowtie_{x_5} ((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5)) &= \llcorner\{\emptyset\}\lrcorner \\ s(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_4) \bowtie_{x_5} (f_4 \bowtie_{x_3} f_5)) &= \llcorner\{\emptyset\}\lrcorner \\ s(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_4) \bowtie_{x_5} f_7) &= \llcorner\{\emptyset\}\lrcorner \\ s(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5) \bowtie_{x_5} (f_4 \bowtie_{x_3} f_5)) &= \llcorner\{\emptyset\}\lrcorner \\ s(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5) \bowtie_{x_5} f_7) &= \llcorner\{\emptyset\}\lrcorner \\ s((f_4 \bowtie_{x_3} f_5) \bowtie_{x_5} f_7) &= \llcorner\{\emptyset\}\lrcorner \end{aligned}$$

Les formules $(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_4) \bowtie_{x_5} ((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5))$ et $(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5) \bowtie_{x_5} (f_4 \bowtie_{x_3} f_5))$ sont enlevées parce que la formule $(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_4) \bowtie_{x_5} (f_4 \bowtie_{x_3} f_5))$ est plus simples. Les **MTPF**s résultants sont :

– $(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_4) \bowtie_{x_5} (f_4 \bowtie_{x_3} f_5))$

- $(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_4) \bowtie_{x_5} f_7)$
- $(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5) \bowtie_{x_5} f_7)$
- $((f_4 \bowtie_{x_3} f_5) \bowtie_{x_5} f_7)$

Parfois, parce que le nombre de formules de propagations testables est grand, il est plus intéressant de trouver un sous-ensemble de toutes les **MTPF**s qui testent au moins une fois toutes les contraintes. Afin de réduire le nombre de propagations tout en testant toutes les contraintes, les propagations des variables qui sont connues parce qu'elles sont mesurées ou contrôlées, peuvent être effectuées dans une étape préalable. Les **MTPF**s résultant sont appelées **MTPF**s de base (une notion d'ensemble de **RRA**s de base est déja définie dans la section 3.1.3). Ces **MTPF**s sont intéressantes parce que les **MTPF** de base sont un petit sous-ensemble de toutes les **MTPF** qui couvrent toutes les structures de contraintes testables. L'algorithme 2 peut être utilisé mais quand une variables $v$ est impliquée dans une structure terminale (la structure d'un capteur par exemple), l'opérateur jointure $\bowtie_v$ est appliqué seulement, d'une part, entre les structures terminales et d'autres structures impliquant $v$, et d'autre part, entre les structures terminales elles-mêmes si beaucoup d'entres elles contiennent la variable $v$ : il correspond à ce qu'on appelle "redondance matérielle".

Reprenons l'exemple ci-dessus. Nous allons chercher les **MTPF**s de base. L'ensemble $F_1$ est composé des formules suivantes :

$$\begin{aligned} s(f_3) &= \llcorner\{x_3, x_4, x_5\}\lrcorner \\ s(f_4) &= \llcorner\{x_3, x_5\}\lrcorner \\ s(f_5) &= \llcorner\{x_3\}\lrcorner \\ s(f_6) &= \llcorner\{x_4\}\lrcorner \\ s(f_7) &= \llcorner\{x_5\}\lrcorner \end{aligned}$$

La variable $x_4$ a le plus bas ordre dans $F_1$. Elle est sélectionnée d'abord. Ensuite, l'opérateur jointure $\bowtie_{x_4}$ est utilisé entre la structure terminale $s(f_6)$ et la stucture $s(f_3)$. Alors, en enlevant les formules contenant la variable $x_4$, l'ensemble de formules suivantes $F$ est obtenu :

$$\begin{aligned} s(f_3 \bowtie_{x_4} f_6) &= \llcorner\{x_3, x_5\}\lrcorner \\ s(f_4) &= \llcorner\{x_3, x_5\}\lrcorner \\ s(f_5) &= \llcorner\{x_3\}\lrcorner \\ s(f_7) &= \llcorner\{x_5\}\lrcorner \end{aligned}$$

Les variables $x_3$ et $x_5$ ont le même ordre. La variable $x_3$ est sélectionnée et l'opérateur jointure $\bowtie_{x_3}$ est utilisé entre la structure terminale $s(f_5)$ et les structures $s((f_3 \bowtie_{x_4} f_6))$

et $s(f_4)$. En enlevant les formules contenant la variable $x_3$, nous obtenons :

$$\begin{aligned} s((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5) &= \llcorner\{x_5\}\lrcorner \\ s(f_4 \bowtie_{x_3} f_5) &= \llcorner\{x_5\}\lrcorner \\ s(f_7) &= \llcorner\{x_5\}\lrcorner \end{aligned}$$

La variable $x_5$ est sélectionnée et l'opérateur jointure $\bowtie_{x_5}$ est utilisée entre la structure terminale $s(f_7)$ et les structures $s((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5)$ et $s(f_4 \bowtie_{x_3} f_5)$. En enlevant les formules contenant la variable $x_5$, nous obtenons :

$$\begin{aligned} s(((f_3 \bowtie_{x_4} f_6) \bowtie_{x_3} f_5) \bowtie_{x_5} f_7) &= \llcorner\{\emptyset\}\lrcorner \\ s((f_4 \bowtie_{x_3} f_5) \bowtie_{x_5} f_7) &= \llcorner\{\emptyset\}\lrcorner \end{aligned}$$

Ces deux **MTPF**s sont les **MTPF**s de base qui couvrent toutes les structures de contraintes testables.

### 4.4.4 Contextes particuliers de modélisation

Dans cette section, deux contextes particuliers sont examinés : les systèmes dynamiques et les systèmes avec embranchements.

**.2 Système dynamiques** D'une manière générale, un modèle est dit dynamique si :
- une variable apparaît plusieurs fois dans un système mais à différents instants.
- une variable et ses dérivées ou intégrales apparaissent dans ce système.

Le premier cas concerne principalement les systèmes avec retard et les systèmes échantillonnés. Selon la modélisation structurelle (section 2.6), chaque variable représente un tube dans un espace phénoménologique par conséquent, un retard de temps, modélisé par $y(t + \triangle) = x(t)$, est une contrainte qui établit un lien entre deux tubes : $\{dom(y(t + \triangle)); \forall t\}$ et $\{dom(x(t)); \forall t\}$. Par conséquent, même si les deux variables modélisent le même phénomène, dans le modèle structurel, elles ne peuvent pas être confondues.
Considérons maintenant le modèle échantillonné suivant :

$$\begin{aligned} x((k+1)T_e) &= Ax(kT_e) + Bu_k(kT_e) \\ y(kT_e) &= Cx(kT_e) \qquad ; k \in \mathbb{N} \end{aligned}$$

où $T_e$ correspond à la période d'échantillonnage. Le phénomène modélisé par $x$ apparaît deux fois. Par conséquent, la contrainte doit être implicitement complétée par

un retard entre les variables $x((k+1)T_e)$ et $x(kT_e)$ (en utilisant l'opérateur retard : $x((k+1)T_e) = qx(kT_e)$). De manière structurelle, les contraintes sont modélisées par les structures suivantes :

$$\llcorner\{x(kT_e), x((k+1)T_e), u(kT_e)\}\lrcorner$$
$$\llcorner\{x(kT_e), x((k+1)T_e)\}\lrcorner$$
$$\llcorner\{x(kT_e), y(kT_e)\}\lrcorner$$

En plus, si le tube correspondant à $x((k+1)T_e)$ apparaît seulement dans ces contraintes (ce qui est habituellement le cas dans la pratique), l'opérateur jointure $\bowtie_{x((k+1)T_e)}$ peut être appliqué entre $\llcorner\{x(kT_e), x((k+1)T_e), u(kT_e)\}\lrcorner$ et $\llcorner\{x(kT_e), x((k+1)T_e)\}\lrcorner$ et on obtient :

$$\llcorner\{x(kT_e), u(kT_e)\}\lrcorner$$
$$\llcorner\{x(kT_e), y(kT_e)\}\lrcorner$$

Si nous voulons obtenir des **SST**s causaux, il suffit de modéliser le retard par $\llcorner\{x(kT_e), x((k+1)T_e)\}\lrcorner$ afin d'interdire de remonter à $x(kT_e)$ en partant de $x((k+1)T_e)$.

Reprenons l'exemple de compteur digital traité dans le chapitre 3 et supposons que toutes les variables sont déductibles (calculables).
Parce qu'il n'y a pas de structure qui peuvent être enlevées, nous allons commencer avec la définition de l'ensemble $F_1$, qui est composé des formules suivantes :

$$\begin{aligned}
s(f_1) &= \llcorner\{x_0, x_1\}\lrcorner \\
s(f_2) &= \llcorner\{x_1, x_2\}\lrcorner \\
s(f_3) &= \llcorner\{x_2, x_3\}\lrcorner \\
s(f_4) &= \llcorner\{x_3, x_4\}\lrcorner \\
s(f_5) &= \llcorner\{x_0\}\lrcorner \\
s(f_6) &= \llcorner\{x_1\}\lrcorner \\
s(f_7) &= \llcorner\{x_2\}\lrcorner \\
s(f_8) &= \llcorner\{x_3\}\lrcorner \\
s(f_9) &= \llcorner\{x_4\}\lrcorner
\end{aligned}$$

La variable $x_0$ est une des variables avec le plus bas ordre dans $F_1$. Elle est sélectionnée d'abord. Ensuite, l'opérateur jointure $\bowtie_{x_0}$ est utilisé entre toutes les formules où cette variable est impliquée. Alors, l'ensemble de formules suivantes $F$ est obtenu :

$$s(f_1 \bowtie_{x_0} f_5) = \llcorner\{x_1\}\lrcorner$$

$$
\begin{aligned}
s(f_2) &= \llcorner\{x_1, x_2\}\lrcorner \\
s(f_3) &= \llcorner\{x_2, x_3\}\lrcorner \\
s(f_4) &= \llcorner\{x_3, x_4\}\lrcorner \\
s(f_6) &= \llcorner\{x_1\}\lrcorner \\
s(f_7) &= \llcorner\{x_2\}\lrcorner \\
s(f_8) &= \llcorner\{x_3\}\lrcorner \\
s(f_9) &= \llcorner\{x_4\}\lrcorner
\end{aligned}
$$

Ensuite, $x_4$ est sélectionnée. Cela donne :

$$
\begin{aligned}
s(f_1 \bowtie_{x_0} f_5) &= \llcorner\{x_1\}\lrcorner \\
s(f_2) &= \llcorner\{x_1, x_2\}\lrcorner \\
s(f_3) &= \llcorner\{x_2, x_3\}\lrcorner \\
s(f_4 \bowtie_{x_4} f_9) &= \llcorner\{x_3\}\lrcorner \\
s(f_6) &= \llcorner\{x_1\}\lrcorner \\
s(f_7) &= \llcorner\{x_2\}\lrcorner \\
s(f_8) &= \llcorner\{x_3\}\lrcorner
\end{aligned}
$$

Ensuite, les variables $x_1$, $x_2$ et $x_3$ peuvent être sélectionnées parce qu'elles ont le même ordre 3. Choisissons la variable $x_1$, nous obtenons :

$$
\begin{aligned}
s(f_3) &= \llcorner\{x_2, x_3\}\lrcorner \\
s(f_4 \bowtie_{x_4} f_9) &= \llcorner\{x_3\}\lrcorner \\
s(f_7) &= \llcorner\{x_2\}\lrcorner \\
s(f_8) &= \llcorner\{x_3\}\lrcorner \\
s((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_2) &= \llcorner\{x_2\}\lrcorner \\
s((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_6) &= \llcorner\{\emptyset\}\lrcorner \\
s(f_2 \bowtie_{x_1} f_6) &= \llcorner\{x_2\}\lrcorner
\end{aligned}
$$

La formule $(f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_6$ est a **MTPF**. Ensuite, la variable $x_3$ doit être sélectionnée. Il mène à :

$$
\begin{aligned}
s((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_6) &= \llcorner\{\emptyset\}\lrcorner \\
s(f_7) &= \llcorner\{x_2\}\lrcorner \\
s((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_2) &= \llcorner\{x_2\}\lrcorner \\
s(f_2 \bowtie_{x_1} f_6) &= \llcorner\{x_2\}\lrcorner
\end{aligned}
$$

$$
\begin{aligned}
s(f_3 \bowtie_{x_3} (f_4 \bowtie_{x_4} f_9)) &= \llcorner\{x_2\}\lrcorner \\
s(f_3 \bowtie_{x_3} f_8) &= \llcorner\{x_2\}\lrcorner \\
s((f_4 \bowtie_{x_4} f_9) \bowtie_{x_3} f_8) &= \llcorner\{\emptyset\}\lrcorner
\end{aligned}
$$

Une nouvelle **MTPF** a été trouvée : $(f_4 \bowtie_{x_4} f_9) \bowtie_{x_3} f_8$. Il reste seulement la variable $x_2$. En appliquant l'opérateur jointure $\bowtie_{x_2}$ entre les formules contenant la variable $x_2$, nous obtenons :

$$
\begin{aligned}
s((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_6) &= \llcorner\{\emptyset\}\lrcorner \\
s((f_4 \bowtie_{x_4} f_9) \bowtie_{x_3} f_8) &= \llcorner\{\emptyset\}\lrcorner \\
s(f_7 \bowtie_{x_2} ((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_2)) &= \llcorner\{\emptyset\}\lrcorner \\
s(f_7 \bowtie_{x_2} (f_2 \bowtie_{x_1} f_6)) &= \llcorner\{\emptyset\}\lrcorner \\
s(f_7 \bowtie_{x_2} (f_3 \bowtie_{x_3} (f_4 \bowtie_{x_4} f_9))) &= \llcorner\{\emptyset\}\lrcorner \\
s(f_7 \bowtie_{x_2} (f_3 \bowtie_{x_3} f_8)) &= \llcorner\{\emptyset\}\lrcorner \\
s(((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_2) \bowtie_{x_2} (f_3 \bowtie_{x_3} (f_4 \bowtie_{x_4} f_9))) &= \llcorner\{\emptyset\}\lrcorner \\
s(((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_2) \bowtie_{x_3} (f_3 \bowtie_{x_3} f_8)) &= \llcorner\{\emptyset\}\lrcorner \\
s((f_2 \bowtie_{x_1} f_6) \bowtie_{x_2} (f_3 \bowtie_{x_3} (f_4 \bowtie_{x_4} f_9))) &= \llcorner\{\emptyset\}\lrcorner \\
s((f_2 \bowtie_{x_1} f_6) \bowtie_{x_2} (f_3 \bowtie_{x_3} f_8)) &= \llcorner\{\emptyset\}\lrcorner
\end{aligned}
$$

Les formules $((f_1 \bowtie_{x_0} f_5) \bowtie_{x_2} f_2) \bowtie_{x_2} (f_2 \bowtie_{x_1} f_6)$ et $(f_3 \bowtie_{x_3} (f_4 \bowtie_{x_4} f_9)) \bowtie_{x_2} (f_3 \bowtie_{x_3} f_8)$ ont été calculées et puis enlevées parce que les formules $(f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_6$ et $(f_4 \bowtie_{x_4} f_9) \bowtie_{x_3} f_8$ sont respectivement plus simples.
Les **MTPF**s résultant sont :

- $((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_6)$
- $((f_4 \bowtie_{x_4} f_9) \bowtie_{x_3} f_8)$
- $(f_7 \bowtie_{x_2} ((f_1 \bowtie_{x_0} f_5)$
- $(f_7 \bowtie_{x_2} (f_2 \bowtie_{x_1} f_6))$
- $(f_7 \bowtie_{x_2} (f_3 \bowtie_{x_3} (f_4 \bowtie_{x_4} f_9)))$
- $(f_7 \bowtie_{x_2} (f_3 \bowtie_{x_3} f_8))$
- $(((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_2) \bowtie_{x_2} (f_3 \bowtie_{x_3} (f_4 \bowtie_{x_4} f_9)))$
- $(((f_1 \bowtie_{x_0} f_5) \bowtie_{x_1} f_2) \bowtie_{x_3} (f_3 \bowtie_{x_3} f_8)$
- $((f_2 \bowtie_{x_1} f_6) \bowtie_{x_2} (f_3 \bowtie_{x_3} (f_4 \bowtie_{x_4} f_9)))$
- $((f_2 \bowtie_{x_1} f_6) \bowtie_{x_2} (f_3 \bowtie_{x_3} f_8))$

La matrice de signature de défauts est donnée par le tableau 4.1.
Seulement 17 propagations élémentaires sont nécessaires pour trouver toutes les **MTPF**s à la place de 20 avec la méthode de la propagation de valeurs (chapitre 3) et 248 manières de propagations avec la méthode basée sur le couplage (chapitres 3). Nous remarquons aussi que cette méthode est moins complexe que la méthode proposée dans (Travé-Massuyès et al. [2006]).

TAB. 4.1 – Les sous-systèmes testables

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| $SST_2$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| $SST_3$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| $SST_4$ | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| $SST_5$ | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| $SST_6$ | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| $SST_7$ | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| $SST_8$ | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| $SST_9$ | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| $SST_{10}$ | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |

Le second cas concerne les intégrations et les équations différentielles. Considérons par exemple le modèle suivant : $\frac{dx}{dt} = u$. $\frac{dx}{dt}$ correspond à un tube, qui peut être connecté à $x$ en ajoutant la contrainte implicite suivante : $x = \int \frac{dx}{dt} dt$. La condition initiale pourrait également être prise en compte en considérant $x = \int_0^{t_f} \frac{dx}{dt} dt + x_0$. Dans ce cas, les structures deviennent : $\llcorner\{\frac{dx}{dt}, u\}\lrcorner$ and $\llcorner\{x, \frac{dx}{dt}, x_0\}\lrcorner$. De la même manière que les systèmes avec retard, l'opérateur jointure $\bowtie_{\frac{dx}{dt}}$ peut être appliqué pour obtenir la structure suivante : $\llcorner\{u, x, x_0\}\lrcorner$. Ce résultat reste vrai pour les intégrales et les dérivées de n'importe quel ordre.
Avec ce type de systèmes, la notion de causalité (déductibilité) peut être prise en compte pour éviter les inconvénients liés à la dérivation ou l'intégration. Considérons maintenant l'équation différentielle :

$$\frac{dx}{dt} = Ax + Bu$$
$$y = Cx$$

Ici aussi, une contrainte implicite doit être ajoutée : $x = \int \frac{dx}{dt} dt$. Les contraintes sont modélisées par les structures suivantes :

$$\llcorner\{x, \frac{dx}{dt}, u\}\lrcorner$$
$$\llcorner\{x, \frac{dx}{dt})\}\lrcorner$$
$$\llcorner\{x, y\}\lrcorner$$

En utilisant l'opérateur jointure $\bowtie_{\frac{dx}{dt}}$, nous obtenons :

$$\llcorner\{x, u\}\lrcorner$$
$$\llcorner\{x, y\}\lrcorner$$

Reprenons l'exemple de bacs avec les contraintes implicites (section 2.4.1).
Parce qu'il n'y a pas de structure qui peuvent être enlevées, nous allons commencer avec la définition de l'ensemble $F_1$, qui est composé des formules suivantes :

$$\begin{aligned}
s(f_1) &= \llcorner\{q_{i_1}, q_{o1}, \frac{dl_1}{dt}\}\lrcorner \\
s(f_2) &= \llcorner\{q_{o1}, l_1\}\lrcorner \\
s(f_3) &= \llcorner\{q_{o_1}, q_{o_2}, q_{i_2}, \frac{dl_2}{dt}\}\lrcorner \\
s(f_4) &= \llcorner\{q_{o_1}, l_2\}\lrcorner \\
s(f_5) &= \llcorner\{l_1\}\lrcorner \\
s(f_6) &= \llcorner\{l_2\}\lrcorner \\
s(f_7) &= \llcorner\{q_{i_1}\}\lrcorner \\
s(f_8) &= \llcorner\{q_{i_2}\}\lrcorner \\
s(f_9) &= \llcorner\{\frac{dl_1}{dt}\}, \{l_1\}\lrcorner \\
s(f_{10}) &= \llcorner\{\frac{dl_2}{dt}\}, \{l_2\}\lrcorner
\end{aligned}$$

La variable $q_{i_1}$ est une des variables avec le plus bas ordre dans $F_1$. Elle est sélectionnée d'abord. Ensuite, l'opérateur jointure $\bowtie_{q_{i_1}}$ est utilisé entre toutes les formules où cette variable est impliquée. Alors, l'ensemble de formules suivantes $F$ est obtenu :

$$\begin{aligned}
s(f_1 \bowtie_{q_{i_1}} f_7) &= \llcorner\{q_{o1}, \frac{dl_1}{dt}\}\lrcorner \\
s(f_2) &= \llcorner\{q_{o1}, l_1\}\lrcorner \\
s(f_3) &= \llcorner\{q_{o_1}, q_{o_2}, q_{i_2}, \frac{dl_2}{dt}\}\lrcorner \\
s(f_4) &= \llcorner\{q_{o_2}, l_2\}\lrcorner \\
s(f_5) &= \llcorner\{l_1\}\lrcorner \\
s(f_6) &= \llcorner\{l_2\}\lrcorner \\
s(f_8) &= \llcorner\{q_{i_2}\}\lrcorner \\
s(f_9) &= \llcorner\{\frac{dl_1}{dt}\}, \{l_1\}\lrcorner \\
s(f_{10}) &= \llcorner\{\frac{dl_2}{dt}\}, \{l_2\}\lrcorner
\end{aligned}$$

Choisissons la variable $q_{i_2}$, nous obtenons :

$$\begin{aligned}
s(f_1 \bowtie_{q_{i_1}} f_7) &= \llcorner\{q_{o1}, \frac{dl_1}{dt}\}\lrcorner \\
s(f_2) &= \llcorner\{q_{o1}, l_1\}\lrcorner \\
s(f_3 \bowtie_{q_{i_2}} f_8) &= \llcorner\{q_{o_1}, q_{o_2}, \frac{dl_2}{dt}\}\lrcorner \\
s(f_4) &= \llcorner\{q_{o_1}, l_2\}\lrcorner \\
s(f_5) &= \llcorner\{l_1\}\lrcorner \\
s(f_6) &= \llcorner\{l_2\}\lrcorner \\
s(f_9) &= \llcorner\{\frac{dl_1}{dt}\}, \{l_1\}\lrcorner \\
s(f_{10}) &= \llcorner\{\frac{dl_2}{dt}\}, \{l_2\}\lrcorner
\end{aligned}$$

Ensuite, $q_{o_2}$ est sélectionnée. Cela donne :

$$\begin{aligned}
s(f_1 \bowtie_{q_{i_1}} f_7) &= \llcorner\{q_{o1}, \frac{dl_1}{dt}\}\lrcorner \\
s(f_2) &= \llcorner\{q_{o1}, l_1\}\lrcorner \\
s((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) &= \llcorner\{q_{o_1}, \frac{dl_2}{dt}, l_2\}\lrcorner \\
s(f_5) &= \llcorner\{l_1\}\lrcorner \\
s(f_6) &= \llcorner\{l_2\}\lrcorner \\
s(f_9) &= \llcorner\{\frac{dl_1}{dt}\}, \{l_1\}\lrcorner \\
s(f_{10}) &= \llcorner\{\frac{dl_2}{dt}\}, \{l_2\}\lrcorner
\end{aligned}$$

Ensuite, $\frac{dl_1}{dt}$ est sélectionnée. Cela donne :

$$\begin{aligned}
s((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) &= \llcorner\{q_{o1}\}, \{l_1\}\lrcorner \\
s(f_2) &= \llcorner\{q_{o1}, l_1\}\lrcorner \\
s((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) &= \llcorner\{q_{o_1}, \frac{dl_2}{dt}, l_2\}\lrcorner \\
s(f_5) &= \llcorner\{l_1\}\lrcorner \\
s(f_6) &= \llcorner\{l_2\}\lrcorner \\
s(f_{10}) &= \llcorner\{\frac{dl_2}{dt}\}, \{l_2\}\lrcorner
\end{aligned}$$

Ensuite, $\frac{dl_2}{dt}$ est sélectionnée. Cela donne :

$$s((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) = \llcorner\{q_{o1}\}, \{l_1\}\lrcorner$$

$$
\begin{aligned}
s(f_2) &= \llcorner\{q_{o1}, l_1\}\lrcorner \\
s(((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) \bowtie_{\frac{dl_2}{dt}} f_{10}) &= \llcorner\{q_{o_1}\}, \{l_2\}\lrcorner \\
s(f_5) &= \llcorner\{l_1\}\lrcorner \\
s(f_6) &= \llcorner\{l_2\}\lrcorner
\end{aligned}
$$

Ensuite, la variable $l_2$ est sélectionnée. Cela donne :

$$
\begin{aligned}
s((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) &= \llcorner\{q_{o1}\}, \{l_1\}\lrcorner \\
s(f_2) &= \llcorner\{q_{o1}, l_1\}\lrcorner \\
s((((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) \bowtie_{\frac{dl_2}{dt}} f_{10}) \bowtie_{l_2} f_6) &= \llcorner\{q_{o_1}\}, \{\emptyset\}\lrcorner \\
s(f_5) &= \llcorner\{l_1\}\lrcorner
\end{aligned}
$$

La variable $l_1$ est sélectionnée. Cela donne :

$$
\begin{aligned}
s(((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_2) &= \llcorner\{q_{o1}\}\lrcorner \\
s(((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_5) &= \llcorner\{q_{o1}\}, \{\emptyset\}\lrcorner \\
s(f_2 \bowtie_{l_1} f_5) &= \llcorner\{q_{o1}\}\lrcorner \\
s((((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) \bowtie_{\frac{dl_2}{dt}} f_{10}) \bowtie_{l_2} f_6) &= \llcorner\{q_{o_1}\}, \{\emptyset\}\lrcorner
\end{aligned}
$$

Enfin, la variable $q_{o1}$ est sélectionnée. Nous obtenons :

$$
\begin{aligned}
s((((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_2) \bowtie_{q_{o_1}} (f_2 \bowtie_{l_1} f_5)) &= \llcorner\{\emptyset\}\lrcorner \\
s((((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_2) \bowtie_{q_{o_1}} ((((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) \bowtie_{\frac{dl_2}{dt}} f_{10}) \bowtie_{l_2} f_6)) &= \llcorner\{\emptyset\}\lrcorner \\
s((f_2 \bowtie_{l_1} f_5)((((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) \bowtie_{\frac{dl_2}{dt}} f_{10}) \bowtie_{l_2} f_6)) &= \llcorner\{\emptyset\}\lrcorner
\end{aligned}
$$

Les formules $s((((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_2) \bowtie_{q_{o_1}} (((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_5))$ et $((((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_5) \bowtie_{q_{o_1}} (f_2 \bowtie_{l_1} f_5))$ ont été calculées et puis enlevées parce que la formule $((((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_2) \bowtie_{q_{o_1}} (f_2 \bowtie_{l_1} f_5))$ est plus simple.
Les **MTPF**s résultant sont :

- $s((((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_2) \bowtie_{q_{o_1}} (f_2 \bowtie_{l_1} f_5))$
- $s((((f_1 \bowtie_{q_{i_1}} f_7) \bowtie_{\frac{dl_1}{dt}} f_9) \bowtie_{l_1} f_2) \bowtie_{q_{o_1}} ((((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) \bowtie_{\frac{dl_2}{dt}} f_{10}) \bowtie_{l_2} f_6))$
- $s((f_2 \bowtie_{l_1} f_5)((((f_3 \bowtie_{q_{i_2}} f_8) \bowtie_{q_{o_2}} f_4) \bowtie_{\frac{dl_2}{dt}} f_{10}) \bowtie_{l_2} f_6))$

Nous remarquons que cette méthode est plus générale que la méthode proposée dans (Krysander et al. [2008]) parce qu'elle prend en compte la notion de déductibilité (calculabilité).

**.3 Système avec embranchements** L'utilisation d'une approche structurelle pour la conception de sous-systèmes testables mène à une méthode générale qui peut être appliquée potentiellement à n'importe quel type de systèmes. Le cas des systèmes dynamiques a été examiné mais un autre contexte exige également une attention particulière : le système avec embranchements. Ces systèmes peuvent être gérés par la méthode présentée dans la section 4.4.3 mais le coût de calcul peut être considérablement réduit. En effet, considérons par exemple les routes dans la figure 4.8. Supposons que $R_{XY}$ est une section de routes liant $X$ à $Y$, un modèle structurel du réseau de routes est donné par :

$$s(R_{AB_1}) = \llcorner A, B_1 \lrcorner$$
$$s(R_{B_2C}) = \llcorner B_2, C \lrcorner$$
$$s(R_{B_3D}) = \llcorner B_3, D \lrcorner$$
$$s(\text{carrefour}) = \llcorner B_1, B_2 \lrcorner$$
$$s(\text{carrefour}) = \llcorner B_1, B_3 \lrcorner$$
$$s(\text{carrefour}) = \llcorner B_2, B_3 \lrcorner$$

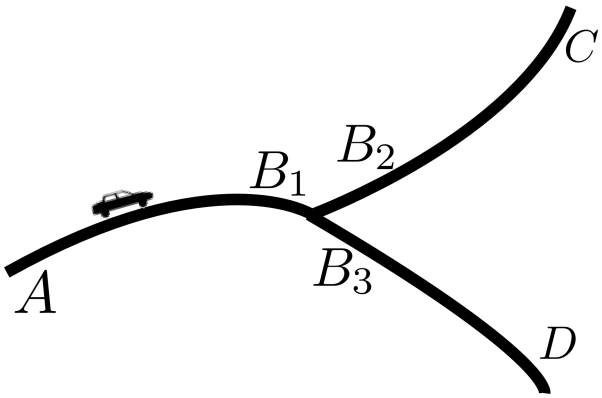


FIG. 4.8 – Un réseau de routes dans un système avec embranchements

D'une manière générale, si une voiture va de $A$ à $C$, il est improbable que sa route passe par $B_3$. Alors, un sous système testable contenant deux ou toutes les contraintes du carrefour n'aurait aucun sens. Si cette spécifité n'est pas prise en compte, le nombre de $SST$ générés sera très important et la plupart d'entres eux ne seront d'aucune utilité. Dans ce cas simple, des routes irréalistes vont apparaître : $(A, B_1, B_2, B_3, D)$, $(A, B_1, B_3, B_2, C)$, $(C, B_2, B_1, B_3, D)$,... Imaginons maintenant un réseau complet avec plusieurs carrefours. Des combinaisons irréalistes de contraintes seront recombinées et elles vont produire de nouvelles contraintes irréalistes et ainsi de suite jusqu'à ce qui un très grand nombre de $SST$ irréalistes soit obtenu. Afin d'éviter ce phénomène, les jonctions doivent être prises en compte pendant la génération des formules.

Afin d'éviter les sous-systèmes testables irréalistes, un ensemble d'exclusions peut être défini. Afin de modéliser que seules deux contraintes du carrefour peuvent être ensemble dans une formule, l'ensemble d'exclusions suivant peut être défini : $\{\llcorner B_1, B_2 \lrcorner, \llcorner B_1, B_3 \lrcorner \llcorner B_2, B_3 \lrcorner\}$. Cela signifie que deux structures ou plus de cet ensemble ne doivent pas apparaître dans une même formule. D'une manière générale, un ensemble d'exclusions, noté $\mathbb{X}$, peut être défini. Dès lors, pendant le calcul des formules (algorithme 2) à chaque fois qu'une nouvelle formule est trouvée, elle est examinée afin de déterminer si elle ne rassemble pas toutes les structures appartenant à un ensemble d'exclusion de $\mathbb{X}$. Dans ce cas, la nouvelle formule est enlevée de $F''$ dans l'algorithme 2.
Les systèmes avec embranchements ne sont pas rares : ils peuvent être trouvés dans les systèmes à événements discrets avec convoyeurs, les réseaux de transport mais aussi en diagnostic de compétences cognitives (Ploix et al. [2004]) où des différents modes de raisonnements sont adoptés.

## 4.5 Exemples

Dans cette section, quelques exemples sont choisis pour clarifier la méthode proposée pour la conception des **MTPF**s. Dans un premier temps, un réseau routier sera présenté. Ce réseau montrera l'intérêt de notre approche proposée permettant d'éviter les sous-systèmes testables irréalistes. Dans un deuxième temps, nous présentons un système électrique (amplificateur). À la fin, nous allons étudier un système binaire. Ce dernier est un circuit logique présenté par la figure 4.11, qui est un additionneur constitué d'un ensemble de portes logiques issu de (Reiter [1987]).

### 4.5.1 Réseau routier

Considérons le réseau routier représenté par la figure 4.9. Il est constitué de trois carrefours et sept routes. Supposons que le nombre de voitures sur les entrées et les sorties sont connues. Notre objectif est de trouver toutes les **MTPF**s en supposant que la voiture va d'un noeud des extrémités du réseau à un autre noeud. Le modèle structurel du réseau est donné par :

$$s(R_{AB_1}) = \llcorner A, B_1 \lrcorner$$

$$s(\text{carrefour1}) = \llcorner B_1, B_2 \lrcorner$$

$$s(\text{carrefour1}) = \llcorner B_1, B_3 \lrcorner$$

$$s(\text{carrefour1}) = \llcorner B_2, B_3 \lrcorner$$

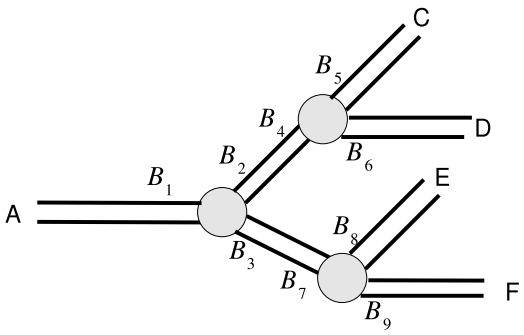


FIG. 4.9 – Un réseau de routes avec trois carrefours

$$s(R_{B_2B_4}) = \llcorner B_2, B_4 \lrcorner$$
$$s(\text{carrefour2}) = \llcorner B_4, B_5 \lrcorner$$
$$s(\text{carrefour2}) = \llcorner B_4, B_6 \lrcorner$$
$$s(\text{carrefour2}) = \llcorner B_5, B_6 \lrcorner$$
$$s(R_{B_3B_7}) = \llcorner B_3, B_7 \lrcorner$$
$$s(\text{carrefour3}) = \llcorner B_7, B_8 \lrcorner$$
$$s(\text{carrefour3}) = \llcorner B_7, B_9 \lrcorner$$
$$s(\text{carrefour3}) = \llcorner B_8, B_9 \lrcorner$$
$$s(R_{B_5C}) = \llcorner B_5, C \lrcorner$$
$$s(R_{B_6D}) = \llcorner B_6, D \lrcorner$$
$$s(R_{B_8E}) = \llcorner B_8, E \lrcorner$$
$$s(R_{B_9F}) = \llcorner B_9, F \lrcorner$$

Pour chercher tous les parcours possibles dans ce réseau, nous allons appliquer la méthode de la conception de tests présentée dans ce chapitre.
Parce qu'il n'y a pas de structures qui peuvent être enlevées, nous allons commencer avec la définition de l'ensemble $F_1$, qui se compose des formules suivantes :

$$s(f_1) = \llcorner \{A, B_1\} \lrcorner$$
$$s(f_2) = \llcorner \{B_1, B_2\} \lrcorner$$
$$s(f_3) = \llcorner \{B_1, B_3\} \lrcorner$$
$$s(f_4) = \llcorner \{B_2, B_3\} \lrcorner$$

$$
\begin{aligned}
s(f_5) &= \llcorner\{B_2, B_4\}\lrcorner \\
s(f_6) &= \llcorner\{B_4, B_5\}\lrcorner \\
s(f_7) &= \llcorner\{B_4, B_6\}\lrcorner \\
s(f_8) &= \llcorner\{B_5, B_6\}\lrcorner \\
s(f_9) &= \llcorner\{B_3, B_7\}\lrcorner \\
s(f_{10}) &= \llcorner\{B_7, B_8\}\lrcorner \\
s(f_{11}) &= \llcorner\{B_7, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s(f_{13}) &= \llcorner\{B_5, C\}\lrcorner \\
s(f_{14}) &= \llcorner\{B_6, D\}\lrcorner \\
s(f_{15}) &= \llcorner\{B_8, E\}\lrcorner \\
s(f_{16}) &= \llcorner\{B_9, F\}\lrcorner \\
s(f_{17}) &= \llcorner\{A\}\lrcorner \\
s(f_{18}) &= \llcorner\{C\}\lrcorner \\
s(f_{19}) &= \llcorner\{D\}\lrcorner \\
s(f_{20}) &= \llcorner\{E\}\lrcorner \\
s(f_{21}) &= \llcorner\{F\}\lrcorner
\end{aligned}
$$

tel que les structures $s(f_{17})$, $s(f_{18})$, $s(f_{19})$, $s(f_{20})$, $s(f_{21})$ représentent les observations sur l'entrée et les sorties du réseau.
Afin d'éviter les sous systèmes testables irréalistes, nous définissons les ensembles d'ensembles d'exclusions suivants : $\{\llcorner B_1, B_2\lrcorner, \llcorner B_1, B_3\lrcorner, \llcorner B_2, B_3\lrcorner\}$, $\{\llcorner B_4, B_5\lrcorner, \llcorner B_4, B_6\lrcorner, \llcorner B_5, B_6\lrcorner\}$ et $\{\llcorner B_7, B_8\lrcorner, \llcorner B_7, B_9\lrcorner, \llcorner B_8, B_9\lrcorner\}$. La variable $A$ est une variable avec le plus bas ordre dans $F_1$. Elle est sélectionnée d'abord. Ensuite, l'opérateur jointure $\bowtie_A$ est utilisé entre toutes les formules où cette variable est impliquée. Alors l'ensemble de formules suivantes $F$ est obtenu :

$$
\begin{aligned}
s(f_1 \bowtie_A f_{17}) &= \llcorner\{B_1\}\lrcorner \\
s(f_2) &= \llcorner\{B_1, B_2\}\lrcorner \\
s(f_3) &= \llcorner\{B_1, B_3\}\lrcorner \\
s(f_4) &= \llcorner\{B_2, B_3\}\lrcorner
\end{aligned}
$$

$$
\begin{aligned}
s(f_5) &= \llcorner\{B_2, B_4\}\lrcorner \\
s(f_6) &= \llcorner\{B_4, B_5\}\lrcorner \\
s(f_7) &= \llcorner\{B_4, B_6\}\lrcorner \\
s(f_8) &= \llcorner\{B_5, B_6\}\lrcorner \\
s(f_9) &= \llcorner\{B_3, B_7\}\lrcorner \\
s(f_{10}) &= \llcorner\{B_7, B_8\}\lrcorner \\
s(f_{11}) &= \llcorner\{B_7, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s(f_{13}) &= \llcorner\{B_5, C\}\lrcorner \\
s(f_{14}) &= \llcorner\{B_6, D\}\lrcorner \\
s(f_{15}) &= \llcorner\{B_8, E\}\lrcorner \\
s(f_{16}) &= \llcorner\{B_9, F\}\lrcorner \\
s(f_{18}) &= \llcorner\{C\}\lrcorner \\
s(f_{19}) &= \llcorner\{D\}\lrcorner \\
s(f_{20}) &= \llcorner\{E\}\lrcorner \\
s(f_{21}) &= \llcorner\{F\}\lrcorner
\end{aligned}
$$

Ensuite, la variable $C$ est sélectionnée. Cela donne :

$$
\begin{aligned}
s(f_1 \bowtie_A f_{17}) &= \llcorner\{B_1\}\lrcorner \\
s(f_2) &= \llcorner\{B_1, B_2\}\lrcorner \\
s(f_3) &= \llcorner\{B_1, B_3\}\lrcorner \\
s(f_4) &= \llcorner\{B_2, B_3\}\lrcorner \\
s(f_5) &= \llcorner\{B_2, B_4\}\lrcorner \\
s(f_6) &= \llcorner\{B_4, B_5\}\lrcorner \\
s(f_7) &= \llcorner\{B_4, B_6\}\lrcorner \\
s(f_8) &= \llcorner\{B_5, B_6\}\lrcorner \\
s(f_9) &= \llcorner\{B_3, B_7\}\lrcorner \\
s(f_{10}) &= \llcorner\{B_7, B_8\}\lrcorner
\end{aligned}
$$

$$
\begin{aligned}
s(f_{11}) &= \llcorner\{B_7, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s(f_{13} \bowtie_C f_{18}) &= \llcorner\{B_5\}\lrcorner \\
s(f_{14}) &= \llcorner\{B_6, D\}\lrcorner \\
s(f_{15}) &= \llcorner\{B_8, E\}\lrcorner \\
s(f_{16}) &= \llcorner\{B_9, F\}\lrcorner \\
s(f_{19}) &= \llcorner\{D\}\lrcorner \\
s(f_{20}) &= \llcorner\{E\}\lrcorner \\
s(f_{21}) &= \llcorner\{F\}\lrcorner
\end{aligned}
$$

Ensuite, les variables $D$,$E$,$F$ sont sélectionnées. Cela donne :

$$
\begin{aligned}
s(f_1 \bowtie_A f_{17}) &= \llcorner\{B_1\}\lrcorner \\
s(f_2) &= \llcorner\{B_1, B_2\}\lrcorner \\
s(f_3) &= \llcorner\{B_1, B_3\}\lrcorner \\
s(f_4) &= \llcorner\{B_2, B_3\}\lrcorner \\
s(f_5) &= \llcorner\{B_2, B_4\}\lrcorner \\
s(f_6) &= \llcorner\{B_4, B_5\}\lrcorner \\
s(f_7) &= \llcorner\{B_4, B_6\}\lrcorner \\
s(f_8) &= \llcorner\{B_5, B_6\}\lrcorner \\
s(f_9) &= \llcorner\{B_3, B_7\}\lrcorner \\
s(f_{10}) &= \llcorner\{B_7, B_8\}\lrcorner \\
s(f_{11}) &= \llcorner\{B_7, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s(f_{13} \bowtie_C f_{18}) &= \llcorner\{B_5\}\lrcorner \\
s(f_{14} \bowtie_D f_{19}) &= \llcorner\{B_6\}\lrcorner \\
s(f_{15} \bowtie_E f_{20}) &= \llcorner\{B_8\}\lrcorner \\
s(f_{16} \bowtie_F f_{21}) &= \llcorner\{B_9\}\lrcorner
\end{aligned}
$$

Les variables $B_1$, $B_2$, $B_3$, $B_4$ ,$B_5$, $B_6$, $B_7$, $B_8$, $B_9$ ont le même ordre "3 "dans les formules. Choisissons la variable $B_2$, nous obtenons :

$$
\begin{aligned}
s(f_1 \bowtie_A f_{17}) &= \llcorner\{B_1\}\lrcorner \\
s(f_2 \bowtie_{B_2} f_5) &= \llcorner\{B_1, B_4\}\lrcorner \\
s(f_4 \bowtie_{B_2} f_5) &= \llcorner\{B_3, B_4\}\lrcorner \\
s(f_3) &= \llcorner\{B_1, B_3\}\lrcorner \\
s(f_6) &= \llcorner\{B_4, B_5\}\lrcorner \\
s(f_7) &= \llcorner\{B_4, B_6\}\lrcorner \\
s(f_8) &= \llcorner\{B_5, B_6\}\lrcorner \\
s(f_9) &= \llcorner\{B_3, B_7\}\lrcorner \\
s(f_{10}) &= \llcorner\{B_7, B_8\}\lrcorner \\
s(f_{11}) &= \llcorner\{B_7, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s(f_{13} \bowtie_C f_{18}) &= \llcorner\{B_5\}\lrcorner \\
s(f_{14} \bowtie_D f_{19}) &= \llcorner\{B_6\}\lrcorner \\
s(f_{15} \bowtie_E f_{20}) &= \llcorner\{B_8\}\lrcorner \\
s(f_{16} \bowtie_F f_{21}) &= \llcorner\{B_9\}\lrcorner
\end{aligned}
$$

Pendant le calcul des formules (algorithme 2), nous remarquons que la formule $(f_2 \bowtie_{B_2} f_4)$ rassemble deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_1, B_2\lrcorner, \llcorner B_1, B_3\lrcorner, \llcorner B_2, B_3\lrcorner\}$. Alors cette formule a été enlevée de l'ensemble de formules.
Choisissons la variable $B_5$, nous obtenons :

$$
\begin{aligned}
s(f_1 \bowtie_A f_{17}) &= \llcorner\{B_1\}\lrcorner \\
s(f_2 \bowtie_{B_2} f_5) &= \llcorner\{B_1, B_4\}\lrcorner \\
s(f_4 \bowtie_{B_2} f_5) &= \llcorner\{B_3, B_4\}\lrcorner \\
s(f_3) &= \llcorner\{B_1, B_3\}\lrcorner \\
s(f_7) &= \llcorner\{B_4, B_6\}\lrcorner \\
s(f_9) &= \llcorner\{B_3, B_7\}\lrcorner
\end{aligned}
$$

$$
\begin{aligned}
s(f_{10}) &= \llcorner\{B_7, B_8\}\lrcorner \\
s(f_{11}) &= \llcorner\{B_7, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6) &= \llcorner\{B_4\}\lrcorner \\
s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_8) &= \llcorner\{B_6\}\lrcorner \\
s(f_{14} \bowtie_D f_{19}) &= \llcorner\{B_6\}\lrcorner \\
s(f_{15} \bowtie_E f_{20}) &= \llcorner\{B_8\}\lrcorner \\
s(f_{16} \bowtie_F f_{21}) &= \llcorner\{B_9\}\lrcorner
\end{aligned}
$$

Pendant le calcul des formules (algorithme 2), nous remarquons que la formule $(f_6 \bowtie_{B_5} f_8)$ rassemble deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_4, B_5 \lrcorner, \llcorner B_4, B_6 \lrcorner, \llcorner B_5, B_6 \lrcorner\}$. Alors cette formule a été enlevée de l'ensemble de formules.

Choisissons la variable $B_7$, nous obtenons :

$$
\begin{aligned}
s(f_1 \bowtie_A f_{17}) &= \llcorner\{B_1\}\lrcorner \\
s(f_2 \bowtie_{B_2} f_5) &= \llcorner\{B_1, B_4\}\lrcorner \\
s(f_4 \bowtie_{B_2} f_5) &= \llcorner\{B_3, B_4\}\lrcorner \\
s(f_3) &= \llcorner\{B_1, B_3\}\lrcorner \\
s(f_7) &= \llcorner\{B_4, B_6\}\lrcorner \\
s(f_9 \bowtie_{B_7} f_{10}) &= \llcorner\{B_3, B_8\}\lrcorner \\
s(f_9 \bowtie_{B_7} f_{11}) &= \llcorner\{B_3, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6) &= \llcorner\{B_4\}\lrcorner \\
s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_8) &= \llcorner\{B_6\}\lrcorner \\
s(f_{14} \bowtie_D f_{19}) &= \llcorner\{B_6\}\lrcorner \\
s(f_{15} \bowtie_E f_{20}) &= \llcorner\{B_8\}\lrcorner \\
s(f_{16} \bowtie_F f_{21}) &= \llcorner\{B_9\}\lrcorner
\end{aligned}
$$

Pendant le calcul des formules (algorithme 2), nous remarquons que la formule $(f_{10} \bowtie_{B_7} f_{11})$ rassemble deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_7, B_8 \lrcorner, \llcorner B_7, B_9 \lrcorner, \llcorner B_8, B_9 \lrcorner\}$. Alors cette formule a été enlevée de l'ensemble de formules.
Choisissons la variable $B_1$, nous obtenons :

$$\begin{aligned}
s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) &= \llcorner\{B_4\}\lrcorner \\
s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3) &= \llcorner\{B_3\}\lrcorner \\
s(f_4 \bowtie_{B_2} f_5) &= \llcorner\{B_3, B_4\}\lrcorner \\
s(f_7) &= \llcorner\{B_4, B_6\}\lrcorner \\
s(f_9 \bowtie_{B_7} f_{10}) &= \llcorner\{B_3, B_8\}\lrcorner \\
s(f_9 \bowtie_{B_7} f_{11}) &= \llcorner\{B_3, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6) &= \llcorner\{B_4\}\lrcorner \\
s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_8) &= \llcorner\{B_6\}\lrcorner \\
s(f_{14} \bowtie_D f_{19}) &= \llcorner\{B_6\}\lrcorner \\
s(f_{15} \bowtie_E f_{20}) &= \llcorner\{B_8\}\lrcorner \\
s(f_{16} \bowtie_F f_{21}) &= \llcorner\{B_9\}\lrcorner
\end{aligned}$$

Pendant le calcul des formules (algorithme 2), nous remarquons que la formule $((f_2 \bowtie_{B_2} f_5) \bowtie_{B_1} f_3)$ rassemble deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_1, B_2 \lrcorner, \llcorner B_1, B_3 \lrcorner, \llcorner B_2, B_3 \lrcorner\}$. Alors cette formule a été enlevée de l'ensemble de formules.
Choisissons la variable $B_6$, nous obtenons :

$$\begin{aligned}
s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) &= \llcorner\{B_4\}\lrcorner \\
s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3) &= \llcorner\{B_3\}\lrcorner \\
s(f_4 \bowtie_{B_2} f_5) &= \llcorner\{B_3, B_4\}\lrcorner \\
s(f_9 \bowtie_{B_7} f_{10}) &= \llcorner\{B_3, B_8\}\lrcorner \\
s(f_9 \bowtie_{B_7} f_{11}) &= \llcorner\{B_3, B_9\}\lrcorner \\
s(f_{12}) &= \llcorner\{B_8, B_9\}\lrcorner \\
s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6) &= \llcorner\{B_4\}\lrcorner
\end{aligned}$$

$$s(((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_8) \bowtie_{B_6} (f_{14} \bowtie_D f_{19})) = \llcorner\{\emptyset\}\lrcorner$$
$$s((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7) = \llcorner\{B_4\}\lrcorner$$
$$s(f_{15} \bowtie_E f_{20}) = \llcorner\{B_8\}\lrcorner$$
$$s(f_{16} \bowtie_F f_{21}) = \llcorner\{B_9\}\lrcorner$$

Pendant le calcul des formules (algorithme 2), nous remarquons que la formule $(((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_8)) \bowtie_{B_6} f_7)$ rassemble deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_4, B_5\lrcorner, \llcorner B_4, B_6\lrcorner, \llcorner B_5, B_6\lrcorner\}$. Alors cette formule a été enlevée de l'ensemble de formules.
La formule $(((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_8) \bowtie_{B_6} (f_{14} \bowtie_D f_{19}))$ est un **MTPF**.

Choisissons la variable $B_8$, nous obtenons :

$$s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) = \llcorner\{B_4\}\lrcorner$$
$$s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3) = \llcorner\{B_3\}\lrcorner$$
$$s(f_4 \bowtie_{B_2} f_5) = \llcorner\{B_3, B_4\}\lrcorner$$
$$s((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20})) = \llcorner\{B_3\}\lrcorner$$
$$s((f_{15} \bowtie_E f_{20}) \bowtie_{B_8} f_{12}) = \llcorner\{B_9\}\lrcorner$$
$$s(f_9 \bowtie_{B_7} f_{11}) = \llcorner\{B_3, B_9\}\lrcorner$$
$$s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6) = \llcorner\{B_4\}\lrcorner$$
$$s((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7) = \llcorner\{B_4\}\lrcorner$$
$$s(f_{16} \bowtie_F f_{21}) = \llcorner\{B_9\}\lrcorner$$

Pendant le calcul des formules (algorithme 2), nous remarquons que la formule $((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} f_{12})$ rassemble deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_7, B_8\lrcorner, \llcorner B_7, B_9\lrcorner, \llcorner B_8, B_9\lrcorner\}$. Alors cette formule a été enlevée de l'ensemble de formules.
Choisissons la variable $B_9$, nous obtenons :

$$s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) = \llcorner\{B_4\}\lrcorner$$
$$s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3) = \llcorner\{B_3\}\lrcorner$$
$$s(f_4 \bowtie_{B_2} f_5) = \llcorner\{B_3, B_4\}\lrcorner$$
$$s((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20})) = \llcorner\{B_3\}\lrcorner$$

$$s(((f_{15} \bowtie_E f_{20}) \bowtie_{B_8} f_{12}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21})) = \quad \llcorner\{\emptyset\}\lrcorner$$
$$s((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21})) = \quad \llcorner\{B_3\}\lrcorner$$
$$s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6) = \quad \llcorner\{B_4\}\lrcorner$$
$$s((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7) = \quad \llcorner\{B_4\}\lrcorner$$

Pendant le calcul des formules (algorithme 2), nous remarquons que la formule $(((f_{15} \bowtie_E f_{20}) \bowtie_{B_8} f_{12}) \bowtie_{B_9} (f_9 \bowtie_{B_7} f_{11}))$ rassemble deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_7, B_8\lrcorner, \llcorner B_7, B_9\lrcorner, \llcorner B_8, B_9\lrcorner\}$. Alors cette formule a été enlevée de l'ensemble de formules.
La formule $(((f_{15} \bowtie_E f_{20}) \bowtie_{B_8} f_{12}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))$ est un **MTPF**.

Les variables $B_3$, $B_4$ ont le même ordre "4 " dans les formules. Choisissons la variable $B_3$, nous obtenons :

$$s((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) = \quad \llcorner\{B_4\}\lrcorner$$
$$s(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20}))) = \quad \llcorner\{\emptyset\}\lrcorner$$
$$s(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3)) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))) = \quad \llcorner\{\emptyset\}\lrcorner$$
$$s((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20}))) = \quad \llcorner\{B_4\}\lrcorner$$
$$s((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))) = \quad \llcorner\{B_4\}\lrcorner$$
$$s((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6) = \quad \llcorner\{B_4\}\lrcorner$$
$$s((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7) = \quad \llcorner\{B_4\}\lrcorner$$

Pendant le calcul des formules (algorithme 2), nous remarquons que la formule $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3) \bowtie_{B_3} (f_4 \bowtie_{B_2} f_5))$ rassemble deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_1, B_2\lrcorner, \llcorner B_1, B_3\lrcorner, \llcorner B_2, B_3\lrcorner\}$ et la formule et $(((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20})) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21})))$ rassemblent deux structures appartenant à l'ensemble d'exclusions de $\{\{\llcorner B_7, B_8\lrcorner, \llcorner B_7, B_9\lrcorner, \llcorner B_8, B_9\lrcorner\}$. Alors ces formules ont été enlevées de l'ensemble de formules.
Les formules $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20})))$ et $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3)) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21})))$ sont des **MTPF**s.
Choisissons la variable $B_4$, nous obtenons :

$$s(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) \bowtie_{B_4} ((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6)) = \quad \llcorner\{\emptyset\}\lrcorner$$
$$s(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) \bowtie_{B_4} ((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7)) = \quad \llcorner\{\emptyset\}\lrcorner$$

$s(((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20}))) \bowtie_{B_4} ((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6)) = \llcorner\{\emptyset\}\lrcorner$

$s(((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20}))) \bowtie_{B_4} ((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7)) = \llcorner\{\emptyset\}\lrcorner$

$s(((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))) \bowtie_{B_4} ((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6)) = \llcorner\{\emptyset\}\lrcorner$

$s(((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))) \bowtie_{B_4} ((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7)) = \llcorner\{\emptyset\}\lrcorner$

Pendant le calcul des formules (algorithme 2), nous remarquons que les formules $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) \bowtie_{B_4} ((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20}))))$, $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) \bowtie_{B_4} ((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))))$, $s(((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20}))) \bowtie_{B_4} ((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))))$ et $s(((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6) \bowtie_{B_4} ((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7))$ rassemblent deux structures appartenant aux ensembles d'exclusions de $\{\{\llcorner B_1, B_2 \lrcorner, \llcorner B_1, B_3 \lrcorner, \llcorner B_2, B_3 \lrcorner\}$ et $\{\{\llcorner B_4, B_5 \lrcorner, \llcorner B_4, B_6 \lrcorner, \llcorner B_5, B_6 \lrcorner\}$. Alors ces formules ont été enlevées de l'ensemble de formules.
Les formules ci-dessus sont des **MTPF**s.

L'ensemble de **MTPF**s résultant de cet exemple est :
- $(((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_8) \bowtie_{B_6} (f_{14} \bowtie_D f_{19}))$
- $(((f_{15} \bowtie_E f_{20}) \bowtie_{B_8} f_{12}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))$
- $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20})))$
- $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} f_3)) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21})))$
- $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) \bowtie_{B_4} ((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6))$
- $(((f_1 \bowtie_A f_{17}) \bowtie_{B_1} (f_2 \bowtie_{B_2} f_5)) \bowtie_{B_4} ((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7))$
- $((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20}))) \bowtie_{B_4} ((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6))$
- $(((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{10}) \bowtie_{B_8} (f_{15} \bowtie_E f_{20}))) \bowtie_{B_4} ((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7))$
- $(((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))) \bowtie_{B_4} ((f_{13} \bowtie_C f_{18}) \bowtie_{B_5} f_6))$
- $(((f_4 \bowtie_{B_2} f_5) \bowtie_{B_3} ((f_9 \bowtie_{B_7} f_{11}) \bowtie_{B_9} (f_{16} \bowtie_F f_{21}))) \bowtie_{B_4} ((f_{14} \bowtie_D f_{19}) \bowtie_{B_6} f_7))$

Nous avons obtenu seulement 10 tests possibles avec notre méthode. Par contre, en appliquant les méthodes existantes proposées dans le chapitre 3 pour concevoir les tests, 89 tests possibles sont obtenus. Cela signifie que les autres méthodes ne sont pas utiles pour ce type de système. Ils n'arrivent pas à éliminer les tests irréalistes.

### 4.5.2 Amplificateur

La méthode proposée pour la conception des tests présentées dans ce chapitre a été appliquée à un circuit électronique (figure 4.10). Ce circuit est modélisé par les contraintes suivantes :

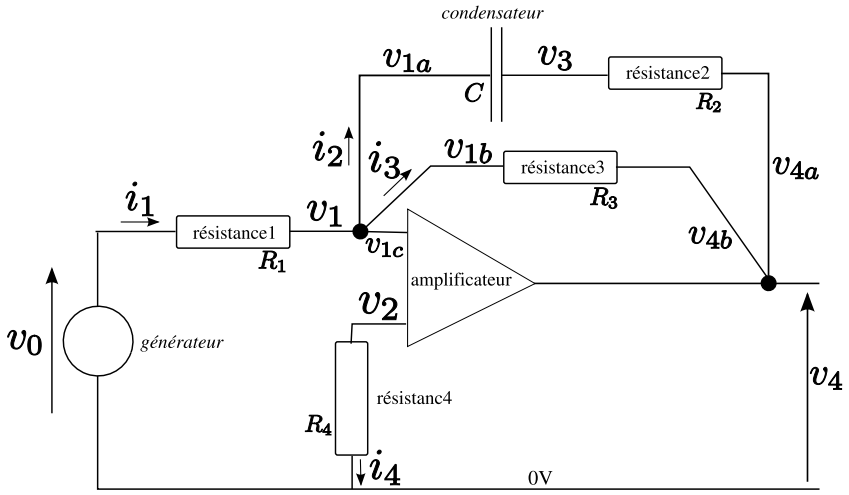


FIG. 4.10 – Un circuit électronique

$$amplificateur : k_1 : v_{1c} = v_2$$
$$connection1 : k_2 : i_1 = i_2 + i_3$$
$$connection1 : k_3 : v_1 = v_{1a}$$
$$connection1 : k_4 : v_1 = v_{1b}$$
$$connection1 : k_5 : v_1 = v_{1c}$$
$$\text{résistance1} : k_6 : v_0 - v_1 = R_1 i_1$$
$$condensateur : k_7 : C(v_{1a} - v_3) = \int_0^t i_2 dt$$
$$\text{résistance2} : k_8 : v_3 - v_{4a} = R_2 i_2$$
$$\text{résistance3} : k_9 : v_1 - v_{4b} = R_3 i_3$$
$$\text{résistance4} : k_{10} : v_2 = R_4 i_4$$
$$connection2 : k_{11} : v_4 = v_{4a}$$
$$connection2 : k_{12} : v_4 = v_{4b}$$
$$\text{générateur} : k_{13} : v_0 = \tilde{v_0}$$
$$capteur1 : k_{14} : v_1 = \tilde{v_1}$$
$$capteur2 : k_{15} : v_2 = \tilde{v_2}$$
$$capteur3 : k_{16} : v_3 = \tilde{v_3}$$
$$capteur4 : k_{17} : v_4 = \tilde{v_4}$$

avec $K_\Sigma = \{k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}, k_{11}, k_{12}, k_{13}\}$, $K_{obs} = \{k_{14}, k_{15}, k_{16}, k_{17}\}$. Ce système peut être modélisé par l'ensemble des structures $F_0$ correspondant à $K$ :

$$s(f_1) \quad = \quad \llcorner\{v_{1c}, v_2\}\lrcorner$$

$$
\begin{aligned}
s(f_2) &= \llcorner\{i_1, i_2, i_3\}\lrcorner \\
s(f_3) &= \llcorner\{v_1, v_{1a}\}\lrcorner \\
s(f_4) &= \llcorner\{v_1, v_{1b}\}\lrcorner \\
s(f_5) &= \llcorner\{v_1, v_{1c}\}\lrcorner \\
s(f_6) &= \llcorner\{v_0, v_1, i_1\}\lrcorner \\
s(f_7) &= \llcorner\{i_2\}, \{v_{1a}, v_3\}\lrcorner \\
s(f_8) &= \llcorner\{v_3, v_{4a}, i_3\}\lrcorner \\
s(f_9) &= \llcorner\{v_1, v_{4b}, i_3\}\lrcorner \\
s(f_{10}) &= \llcorner\{v_2, i_4\}\lrcorner \\
s(f_{11}) &= \llcorner\{v_4, v_{4a}\}\lrcorner \\
s(f_{12}) &= \llcorner\{v_4, v_{4b}\}\lrcorner \\
s(f_{13}) &= \llcorner\{v_0\}\lrcorner \\
s(f_{14}) &= \llcorner\{v_1\}\lrcorner \\
s(f_{15}) &= \llcorner\{v_2\}\lrcorner \\
s(f_{16}) &= \llcorner\{v_3\}\lrcorner \\
s(f_{17}) &= \llcorner\{v_4\}\lrcorner
\end{aligned}
$$

Nous remarquons que la variable $i_4$ apparaît seulement une fois dans les structures du système. La structure contenant cette variable $s(f_{10})$ et sa formule $f_{10}$ seront enlevées. Nous remarquons aussi que la variable $v_{1b}$ apparaît seulement une fois dans les structures du système. La structure contenant cette variable $s(f_4)$ et sa formule $f_4$ seront enlevées. Il n'y a plus de variable qui apparaissent seulement une fois dans les structures restantes. L'ensemble $F_1$ est composé des formules suivantes :

$$
\begin{aligned}
s(f_1) &= \llcorner\{v_{1c}, v_2\}\lrcorner \\
s(f_2) &= \llcorner\{i_1, i_2, i_3\}\lrcorner \\
s(f_3) &= \llcorner\{v_1, v_{1a}\}\lrcorner \\
s(f_5) &= \llcorner\{v_1, v_{1c}\}\lrcorner \\
s(f_6) &= \llcorner\{v_0, v_1, i_1\}\lrcorner \\
s(f_7) &= \llcorner\{i_2\}, \{v_{1a}, v_3\}\lrcorner \\
s(f_8) &= \llcorner\{v_3, v_{4a}, i_3\}\lrcorner \\
s(f_9) &= \llcorner\{v_1, v_{4b}, i_3\}\lrcorner \\
s(f_{11}) &= \llcorner\{v_4, v_{4a}\}\lrcorner \\
s(f_{12}) &= \llcorner\{v_4, v_{4b}\}\lrcorner \\
s(f_{13}) &= \llcorner\{v_0\}\lrcorner \\
s(f_{14}) &= \llcorner\{v_1\}\lrcorner
\end{aligned}
$$

$$
\begin{aligned}
s(f_{15}) &= \llcorner\{v_2\}\lrcorner \\
s(f_{16}) &= \llcorner\{v_3\}\lrcorner \\
s(f_{17}) &= \llcorner\{v_4\}\lrcorner
\end{aligned}
$$

En utilisant l'opérateur jointure entre toutes les variables restant, nous obtenons les **MTPF**s suivantes :

- $(f_{14} \bowtie_{v_1} (f_{16} \bowtie_{v_3} ((((((f_7 \bowtie_{i_2}) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} ((f_{12} \bowtie_{v_{4b}} f_9) \bowtie_{i_3} (f_{11} \bowtie_{v_{4a}} f_8)))))$
- $(f_{14} \bowtie_{v_1} (f_{16} \bowtie_{v_3} ((((((f_7 \bowtie_{i_2}) f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} f17)))$
- $(f_{14} \bowtie_{v_1} (((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} ((f_{12} \bowtie_{v_{4b}} f_9) \bowtie_{i_3} (f_{11} \bowtie_{v_{4a}} f_8))) \bowtie_{v_3} ((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} f_{17})))$
- $(((f_5 \bowtie_{v_{1c}} f_1) \bowtie_{v_2} f_{15}) \bowtie_{v_1} (f_{16} \bowtie_{v_3} ((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} ((f_{12} \bowtie_{v_{4b}} f_9) \bowtie_{i_3} (f_{11} \bowtie_{v_{4a}} f_8)))))$
- $(((f_5 \bowtie_{v_{1c}} f_1) \bowtie_{v_2} f_{15}) \bowtie_{v_1} (f_{16} \bowtie_{v_3} ((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} f_{17})))$
- $(((f_5 \bowtie_{v_{1c}} f_1) \bowtie_{v_2} f_{15}) \bowtie_{v_1} (((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} ((f_{12} \bowtie_{v_{4b}} f_9) \bowtie_{i_3} (f_{11} \bowtie_{v_{4a}} f_8))) \bowtie_{v_3} ((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} f_{17})))$
- $((f_{16} \bowtie_{v_3} ((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{11} \bowtie_{v_{4a}} f_8)) \bowtie_{v_4} f_{17})) \bowtie_{v_1} f_{14})$
- $((f_{16} \bowtie_{v_3} ((((((f_7 \bowtie_{i_2}) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13} \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} f_{17})) \bowtie_{v_1} (f_{16} \bowtie_{v_3} ((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a} f_3}) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{12} \bowtie_{v_{4b}} f_9)) \bowtie_{v_4} (f_{12} \bowtie_{v_{4b}} f_9) \bowtie_{i_3} (f11 \bowtie_{v_{4a}} f_8)))))$
- $(((f_5 \bowtie_{v_{1c}} f_1) \bowtie_{v_2} f_{15}) \bowtie_{v_1} f_{14})$
- $((f_{16} \bowtie_{v_3} (f_{17} \bowtie_{v_4} ((f_{12} \bowtie_{v_{4b}} f_9) \bowtie_{i_3} (f_{11} \bowtie_{v_{4a}} f_8)))) \bowtie_{v_1} f_{14})$
- $((f_{16} \bowtie_{v_3} (f_{17} \bowtie_{v_4} ((f_{12} \bowtie_{v_{4b}} f_9) \bowtie_{i_3} (f_{11} \bowtie_{v_{4a}} f_8)))) \bowtie_{v_1} ((f_5 \bowtie_{v_{1c}} f_1) \bowtie_{v_2} f_{15}))$
- $((f_{16} \bowtie_{v_3} ((((((f_7 \bowtie_{i_2} f_2) \bowtie_{i_1} f_6) \bowtie_{v_{1a}} f_3) \bowtie_{v_0} f_{13}) \bowtie_{i_3} (f_{11} \bowtie_{v_{4a}} f_8)) \bowtie_{v_4} f_{17})) \bowtie_{v_1} ((f_5 \bowtie_{v_{1c}} f_1) \bowtie_{v_2} f_{15}))$

### 4.5.3 Système logique

Le circuit logique qui représente l'additionneur est constitué de cinq composants : deux portes $X-OR$ ($OU-EXCLUSIF$), deux portes $OR$ ($OU$) et une porte $AND$ ($ET$) (la figure 4.11). La porte $OR$ prédit un 1 en sortie à partir du moment où une des entrées vaut 1, prédit un 0 du moment que les deux entrées valent 0. La porte logique $AND$ prédit un 0 en sortie à partir du moment où une des entrées vaut 0, et prédit un 1 en sortie dès que les deux entrées valent 1. Enfin, la porte logique $X-OR$ prédit un

1 en sortie du moment qu'une entrée vaut 0 (respectivement 1) et l'autre entrée vaut 1 (respectivement 0), et prédit un 0 en sortie du moment que les deux entrées sont à 0 ou à 1.
Les composants de ce système sont : $XOR_1$, $XOR_2$, $AND_1$, $AND_2$, $OR$. Les variables sont : $x_1$, $x_2$, $x_3$, $x_4$, $x_5$, $x_6$, $x_7$, $x_8$. Les variables $x_1$, $x_2$, $x_3$, $x_7$, $x_8$ sont mesurées
Ce système peut être représenté par l'ensemble des structures $F_0$ correspondant à $K$ :

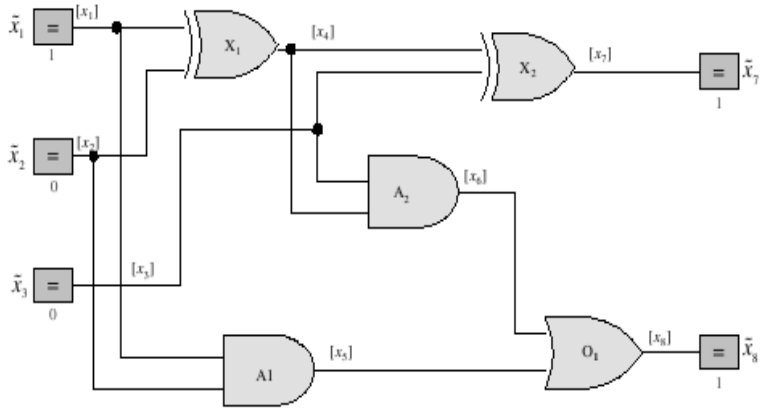


FIG. 4.11 – Un système logique

$$
\begin{aligned}
s(f_1) &= \llcorner\{x_1, x_2, x_4\}\lrcorner \\
s(f_2) &= \llcorner\{x_1, x_2\}, \{x_5\}\lrcorner \\
s(f_3) &= \llcorner\{x_3, x_4\}, \{x_6\}\lrcorner \\
s(f_4) &= \llcorner\{x_3, x_4, x_7\}\lrcorner \\
s(f_5) &= \llcorner\{x_5, x_6\}, \{x_8\}\lrcorner \\
s(f_6) &= \llcorner\{\tilde{x}_1\}\lrcorner \\
s(f_7) &= \llcorner\{\tilde{x}_2\}\lrcorner \\
s(f_8) &= \llcorner\{\tilde{x}_3\}\lrcorner \\
s(f_9) &= \llcorner\{\tilde{x}_7\}\lrcorner \\
s(f_{10}) &= \llcorner\{\tilde{x}_8\}\lrcorner
\end{aligned}
$$

Parce qu'il n'y pas de structure qui peuvent être enlevées, alors $F_1 \equiv F_0$. En appliquant l'algorithme 2, on obtient les **MTPF**s suivantes :

- $(((((((f_5 \bowtie_{x_8} f_{10}) \bowtie_{x_6} f_3) \bowtie_{x_5} f_2) \bowtie_{x_4} (f_4 \bowtie_{x_7} f_9)) \bowtie_{x_3} f_8) \bowtie_{x_2} f_7) \bowtie_{x_1} ((((f_4 \bowtie_{x_7} f_9) \bowtie_{x_4} f_1) \bowtie_{x_3} f_8) \bowtie_{x_2} f_7))$
- $(((((((f_5 \bowtie_{x_8} f_{10}) \bowtie_{x_6} f_3) \bowtie_{x_5} f_2) \bowtie_{x_4} (f_4 \bowtie_{x_7} f_9)) \bowtie_{x_3} f_8) \bowtie_{x_2} f_7) \bowtie_{x_1} f_6$
- $(((((((f_5 \bowtie_{x_8} f_{10}) \bowtie_{x_6} f_3) \bowtie_{x_5} f_2) \bowtie_{x_4} f_1) \bowtie_{x_3} ((f_4 \bowtie_{x_7} f_9) \bowtie_{x_4} f_1)) \bowtie_{x_2} f_7) \bowtie_{x_1} f_6)$

- $(((((((f_5 \bowtie_{x_8} f_{10}) \bowtie_{x_6} f_3) \bowtie_{x_5} f_2) \bowtie_{x_4} f_1) \bowtie_{x_3} f_8) \bowtie_{x_2} f_7) \bowtie_{x_1} f_6)$
- $(((((((f_5 \bowtie_{x_8} f_{10}) \bowtie_{x_6} f_3) \bowtie_{x_5} f_2) \bowtie_{x_4} f_1) \bowtie_{x_3} ((f_4 \bowtie_{x_7} f_9)) \bowtie_{x_2} (((f_4 \bowtie_{x_7} f_9) \bowtie_{x_4} f_1) \bowtie_{x_3} f_8)) \bowtie_{x_1} f_6)$
- $((((((f_4 \bowtie_{x_7} f_9) \bowtie_{x_4} f_1) \bowtie_{x_3} f_8) \bowtie_{x_2} f_7) \bowtie_{x_1} f_6)$

## 4.6 Conclusion

Dans ce chapitre, nous avons formalisé une méthode structurelle pour la conception de tests de détection. Nous avons montré comment elle peut être utilisée pour déduire tous les tests d'un système à diagnostiquer. La méthode proposée fournit toutes les contraintes qui doivent être utilisées pour la conception de chacun des tests.
L'avantage principal de cette méthode est qu'elle permet d'une part de tenir compte des variables non déductibles par certaines contraintes, et d'autre part, qu'elle permet d'intégrer des ensembles d'exclusions qui permettent d'éviter l' explosion combinatoire dans les systèmes discrets. Il s'agit d'une approche structurelle permettant de manipuler tout type de système : des systèmes continus échantillonés ou discrets, linéaires ou non linéaires, statiques ou dynamiques, à base de règle ou non. Dans ce chapitre, nous avons montré avec l'exemple du réseau de routes qu'il est possible de calculer toutes les relations de redondance analytique **RRAs** même si les contraintes comportementales ne sont pas disponibles. Nous avons aussi montré qu'il était aussi possible de tenir compte d'exclusions afin d'éviter les sous-systèmes testables irréalistes.
L'inconvénient de l'approche structurelle présentée dans ce chapitre est que des sur-estimations de solutions peuvent se produire. Même si elles peuvent être partiellement évitées en tenant compte de la déductibilité des variables par rapport aux contraintes, des sur-estimations restent possibles. La conséquence est que certaines contraintes intervenant dans une **SST** peuvent ne pas être nécessaires. Elles doivent être enlevées a posteriori au moment de la conception des tests de détection. Cet inconvénient n'est pas un problème important parce que la principale difficulté reste la détermination de l'ensemble des contraintes qui mènent au RRAs.
La procédure proposée a été conçue afin de réduire autant que possible le nombre de calculs. Comme les contraintes impliquées dans toutes les **MTPF**s, et donc les **SST** sont obtenues, une matrice complète de signature de défauts peut être générée automatiquement. Il devient donc possible de déterminer la meilleure performance atteignable par un système de diagnostic.

# Troisième partie

# Placement de capteurs

# Chapitre 5

# Etat de l'art des méthodes de placement de capteurs

## 5.1 Introduction

L'efficacité d'un système de diagnostic dépend grandement de la pertinence des informations recueillies sur le système à diagnostiquer. Si cette information n'est pas suffisante, le système de diagnostic produira des diagnostics peu précis. Pour recueillir ces informations, des capteurs mesurant les variables du système doivent être installés.

L'efficacité d'un système de capteurs s'apprécie par le niveau de diagnosticabilité qu'il fournit, sachant qu'un même niveau de diagnosticabilité peut être obtenu par différents ensembles de capteurs. Une définition du concept de diagnosticabilité peut être trouvée dans (Console et al. [2000]). Le critère de coût doit être pris en compte dès la conception du système, qui inclut la conception du système de diagnostic, lors du choix des capteurs. Comme un critère de diagnosticabilité peut être satisfait par différentes combinaisons de capteurs, un critère de coût permet d'orienter vers certains choix plus judicieux.

Dans ce chapitre, nous présentons les méthodes de placement de capteurs satisfaisant des objectifs de contrôlabilité, d'observabilité, de monitorabilité ou de diagnosticabilité. Nous nous concentrerons particulièrement sur les méthodes satisfaisant des critère de diagnosticabilité.

## 5.2 Divers formalismes et méthodes

Une méthode de placement de capteurs dépend fondamentalement des objectifs qu'elle vise à atteindre. Par exemple, dans la théorie de la commande, le placement

de capteurs est utilisé pour fournir des informations suffisantes pour la commande des systèmes c'est-à-dire des critères basés sur l'observabilité, la contrôlabilité et la monitorabilité.
(Madron and Veverka [1992]) a proposé une méthode de placement de capteurs dédiée aux systèmes linéaires. Cette méthode utilise l'élimination de Gauss-Jordan pour trouver un ensemble minimal de variables à mesurer. Ceci assure l'observabilité des variables tout en réduisant simultanément au minimum le coût de capteurs. Dans cette théorie, les variables observables incluent les variables mesurées plus les variables non mesurées mais déductibles.
Un outil original pour la conception d'un placement de capteurs minimal est proposé dans (Luong et al. [1994]). L'objectif de cette méthode est de concevoir un système de mesure prenant en compte les contraintes suivantes : l'observabilité des variables requises pour le contrôle et la maintenance, la satisfaction de différents degrés de redondance imposés pour quelques variables et la satisfaction de critères de coût et de fiabilité de capteurs. L'algorithme proposé est basé sur l'analyse des cycles du graphe associé avec le système.
(Sen et al. [1998]) a utilisé un algorithme génétique (GA) pour la conception optimale de réseaux de capteurs pour les processus linéaires. Les algorithmes génétiques s'appuient sur la théorie des graphes afin d'exploiter les propriétés structurelles du problème. Le même algorithme peut être utilisé pour optimiser plusieurs objectifs.
(hoblos et al. [2000]) a développé un algorithme pour la conception de réseaux de capteurs. L'objectif est d'améliorer la robustesse de l'observabilité des variables en cas de pertes de capteurs pour que le processus de contrôle reste satisfait en présence de défaillances des capteurs. Cet algorithme est basé sur la construction des graphes PMSS (pseudo minimal sensor sets) et MSS (minimal sensor sets) considérés dans (Staroswiecki et al. [1999]) afin de sélectionner un ensemble de capteurs garantissant les propriétés d'observabilité.
(Commault et al. [2006a]) propose aussi une méthode structurelle pour le placement de capteurs dans le problème de détection et localisation de défauts ($FDI$). Cette méthode améliore l'observabilité des systèmes. Les résultats de cette méthode prolongent les résultats présentés dans (Commault et al. [2006b]). Dans cette méthode un ensemble de séparateurs (séparateurs d'entrée et de sortie) est défini dans le graphe du système, qui génère des ensembles de variables dans lesquelles des capteurs additionnels doivent être ajoutés pour résoudre le problème considéré. Ces séparateurs paramétrisent toutes les solutions, et permettent de choisir parmi elles un ensemble de capteurs additionnels potentiels pour résoudre le problème de FDI.

Une méthode de placement de capteurs pour l'analyse de monitorabilité est proposée dans (Khemliche et al. [2006]). Un système est monitorable s'il peut être déterminé en utilisant seulement les trajectoires des flots de données (mesures, commandes connues), si les contraintes du système sont satisfaites ou non (Blanke et al. [2003]). Cette méthode est basée sur l'outil "bond graph". Le but de cette méthode est de satisfaire des contraintes

de monitorabilité pour tous les composants. En utilisant cette méthode, la position des capteurs physiques apparaît explicitement dans le modèle graphique. L'idée générale consiste à analyser les combinaisons de placements, qui est un ensemble de vecteurs binaires afin d'obtenir simplement les lignes de la matrice de signature. Intuitivement, cette méthode est une heuristique qui se traduit par un ensemble de règles appliquées lors des combinaisons de placement.

Cependant, dans le diagnostic de défauts, le but du placement de capteurs devrait être de satisfaire des propriétés de détectabilité et de diagnosticabilité. La détectabilité est la possibilité de détecter un défaut sur un composant et la diagnosticabilité est la possibilité de localiser un défaut sur un composant sans ambiguïté avec les autres composants défectueux. (Maquin et al. [1997]) a proposé aussi une méthode pour le placement de capteurs. Cette méthode vise à garantir la détectabilité et la diagnosticabilité des défauts de capteurs. Cette méthode est basée sur le concept de degré de redondance dans les variables et sur l'analyse structurelle du modèle du système. Le placement de capteurs peut être résolu par l'analyse d'une matrice de cycle ou en utilisant la technique de programmation linéaire mixte.
(Travé-Massuyès et al. [2006]) a proposé une méthode de placement de capteurs satisfaisant des critères de détectabilité et de diagnosticabilité. Cette méthode est basée sur le retrait consécutif des capteurs. Cette méthode consiste à faire l'hypothèse que toutes les variables du système sont mesurées. Puis on procède au retrait des capteurs jusqu'à la satisfaction du cahier des charges considéré.
Une méthode de placement de capteurs qui satisfait les critères de détectabilité et diagnosticabilité est proposée dans (Frisk and Krysander [2007]). Cette méthode est basée sur la décomposition Dulmage-Mendelshon (Dulmage and Mendelsohn [1959]). Cette méthode ne s'applique qu'aux systèmes juste-déterminés qui ne contiennent pas de partie sous-déterminée.
Dans la section suivante, nous présentons en détails les méthodes de placement de capteurs les plus récentes qui satisfont des critères de détectabilité et de diagnosticabilité présentées dans (Travé-Massuyès et al. [2006],Frisk and Krysander [2007])

### 5.2.1 Méthode basée sur l'addition ou le retrait consécutif de capteurs

Avant de présenter ces méthodes de placement de capteurs, nous rappelons quelques définitions intéressantes issues de (Travé-Massuyès et al. [2006]).
Considérons un système $\Sigma$, un ensemble de capteurs $\mathcal{S}$ et un ensemble de défauts (uniques ou multiples) $\mathbf{F} = \{\mathcal{F}_i\}$, représentés par un triplet $(\Sigma, \mathcal{S}, \mathbf{F})$. Supposons que $OBS_j$ est un tuple de flots de données $O$ délivrées par les capteurs jusqu à un instant donné $t_j$.
Supposons que $OBS_{\mathcal{F}_j}$ est l'ensemble de tous les tuples possibles de flots de données sous

le défaut $\mathcal{F}_j$.

**Définition 5.2.1.** *(Diagnostic candidat)*
*Étant donné le triplet $(\Sigma, \mathcal{S}, \mathbf{F})$, $\mathcal{F}_i$ est un diagnostic candidat au moment $t_j$ si et seulement si $OBS_j \in OBS_{\mathcal{F}_i}$. $\mathcal{F}_i$ est un diagnostic candidat minimal si $\forall \mathcal{F}_k \subset \mathcal{F}_i$ ; $\mathcal{F}_k$ n'est pas un diagnostic candidat.*

Les propriétés de la diagnosticabilité d'un triplet $(\Sigma, \mathcal{S}, \mathbf{F})$ dépendent des propriétés de discriminabilité de chaque paire de défauts dans $\mathbf{F}$.

**Définition 5.2.2.** *(Discriminabilité)*

1. *deux défauts $\mathcal{F}_i$ et $\mathcal{F}_j$ sont (fortement) discriminables si et seulement si, pour chaque observation $OBS$, quand $\mathcal{F}_i$ est parmi les diagnostics candidats, $\mathcal{F}_j$ ne l'est jamais, et inversement. Autrement dit, $OBS_{\mathcal{F}_i} \cap OBS_{\mathcal{F}_j} = \emptyset$.*
2. *deux défauts $\mathcal{F}_i$ et $\mathcal{F}_j$ sont non discriminables si et seulement si, quelque soit $OBS$, quand $\mathcal{F}_i$ est parmi les diagnostics candidats, $\mathcal{F}_j$ l'est également, et inversement. Autrement dit, $OBS_{\mathcal{F}_i} = OBS_{\mathcal{F}_i}$.*
3. *un défaut $\mathcal{F}_i$ est (faiblement) discriminables d'un défaut $\mathcal{F}_j$ si et seulement si, quand $\mathcal{F}_i$ est parmi les diagnostics candidats, il existe au moins une observation $OBS^k$, telle que $\mathcal{F}_j$ et $\mathcal{F}_i$ apparaissent ensemble dans les diagnostics candidats et au moins un $OBS^l$ avec $OBS^l \neq OBS^k$ tel que $\mathcal{F}_j$ et $\mathcal{F}_i$ n'apparaîssent pas ensemble. Autrement dit, $\mathcal{F}_j$ et $\mathcal{F}_j$ ne sont pas fortement discriminables et $OBS_{\mathcal{F}_i} \setminus OBS_{\mathcal{F}_j} \neq \{\emptyset\}$. Si $\mathcal{F}_i$ est faiblement discriminable de $\mathcal{F}_j$ et $\mathcal{F}_j$ est faiblement discriminable de $F_i$, alors la paire $(\mathcal{F}_i, \mathcal{F}_j)$, est faiblement discriminable.*

Deux défauts $\mathcal{F}_i$ et $\mathcal{F}_j$ sont discriminables s'ils sont soit faiblement discriminables, soit fortement discriminables. La figure 5.1 montre que :

1. $\mathcal{F}_1$ et $\mathcal{F}_2$ sont fortement discriminables
2. $\mathcal{F}_3$ et $\mathcal{F}_4$ sont faiblement discriminables
3. $\mathcal{F}_5$ et $\mathcal{F}_6$ sont non discriminables

**Définition 5.2.3.** *(Diagnosticabilité forte )*
*Un triplet $(\Sigma, \mathcal{S}, \mathbf{F})$ est fortement diagnosticable si et seulement si pour chaque paire $(\mathcal{F}_i, \mathcal{F}_j) \in \mathbf{F} \times \mathbf{F}$ avec $\mathcal{F}_i \neq \mathcal{F}_j$, $OBS_{\mathcal{F}_i} \cap OBS_{\mathcal{F}_j} = \{\emptyset\}$, c'est-à-dire, chaque paire $(\mathcal{F}_i, \mathcal{F}_j)$ est fortement discriminable.*

**Définition 5.2.4.** *(Diagnosticabilité faible)*
*Un triplet $(\Sigma, \mathcal{S}, \mathbf{F})$ est faiblement diagnosticable si et seulement si pour chaque paire $(\mathcal{F}_i, \mathcal{F}_j) \in \mathbf{F} \times \mathbf{F}$ avec $\mathcal{F}_i \neq \mathcal{F}_j$, $OBS_{\mathcal{F}_i} \setminus OBS_{F_j} \neq \emptyset$ ou $OBS_{\mathcal{F}_j} \setminus OBS_{\mathcal{F}_i} \neq \{\emptyset\}$ c'est-à-dire, chaque paire $(\mathcal{F}_i, \mathcal{F}_j)$ est faiblement discriminable.*

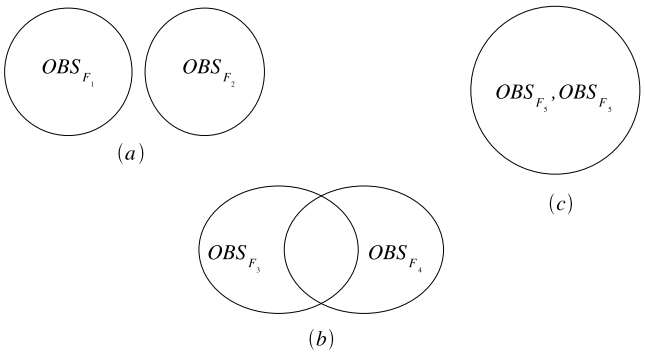


FIG. 5.1 – (a) Deux défauts fortement discriminables. (b) Deux défauts faiblement discriminables. (c) Deux défauts non discriminables

Un système est diagnosticable s'il est soit faiblement diagnosticable soit fortement diagnosticable.

**Définition 5.2.5.** *(D-classe)*
*Deux défauts $\mathcal{F}_i$ et $\mathcal{F}_j$ sont dans la même $D-classe$ si et seulement s'ils sont non discriminables. Notons que deux défauts de différentes D-classes peuvent être fortement ou faiblement discriminables.*

**Définition 5.2.6.** *(niveau de discriminabilité)*
*Étant donné un triplet $(\Sigma, \mathcal{S}, \mathbf{F})$, son niveau de discriminabilité $D_{\mathcal{S}}$ est donné par le nombre de D-classes obtenues pour l'ensemble de capteurs $\mathcal{S}$.*

**Définition 5.2.7.** *(degré de diagnosticabilité)*
*Étant donné un triplet $(\Sigma, \mathcal{S}, \mathbf{F})$, son degré de diagnosticabilité $d_{\mathcal{S}}$ est défini par la fraction suivant : $d_{\mathcal{S}} = \frac{D_{\mathcal{S}}}{Card(\mathbf{F})}$ où $Card(\mathbf{F})$ est le nombre de défauts dans $\mathbf{F}$*

Pour un système entièrement diagnosticable, le nombre $D_{\mathcal{S}}$ est égal au nombre de défauts et $d_s = 1$. En revanche, $d_s < 1$ pour un système partiellement diagnosticable.

**Définition 5.2.8.** *(Ensemble additionnel minimal de capteurs)*
*Soit un triplet partiellement diagnosticable $(\Sigma, \mathcal{S}, \mathbf{F})$. Un ensemble additionnel de capteurs est défini comme un ensemble de capteurs $S$ tel que $(\Sigma, \mathcal{S} \cup S, \mathbf{F})$ est entièrement diagnosticable.Un ensemble additionnel minimal de capteurs est un ensemble additionnel de capteurs $S$ tel qu'il n'existe pas de sous-ensemble de $S$ qui soit un ensemble additionnel de capteurs.*

(Travé-Massuyès et al. [2001]) a proposé une méthode de placement de capteurs basée sur des additions consécutives de capteurs, qui prend en compte des critères de diagnosticabilité. Le principe de cette méthode est d'analyser le modèle physique d'un système d'un point de vue structurel. Cette approche structurelle est basée sur les relations de redondance analytique (**RRA**) (Cassar and Staroswiecki [1997]) qui peuvent être obtenues à partir de combinaisons des contraintes constitutives du modèle de comportement en utilisant les graphes bipartis (Staroswiecki [2003]) ou sur des règles d'élimination (Ploix et al. [2005]).
Cette méthode consiste à construire un graphe AND-OR qui représente la liaison entre les variables et les contraintes. Le noeud "variable" est associé à un OR qui signifie que la valeur de cette variable peut être obtenue de plusieurs façons. Tandis que le noeud "contrainte" est associé à un AND, qui signifie que plusieurs variables sont nécessaires pour instancier la contrainte. Dans une première étape, tous les capteurs additionnels sont ajoutés un par un et la matrice de signature de défauts hypothétique (HFS matrice) est construite pour analyser les relations de redondance analytique. La matrice HFS établit une correspondance entre le capteur additionnel, les relations de redondance résultant et les composants impliqués. L'étape suivante consiste à construire la matrice de signature de défauts hypothétique étendue (EHFS matrix). Cette matrice prend en compte l'addition de plusieurs capteurs à la fois. La matrice EHFS récapitule toute l'information exigée pour fournir tous les ensembles additionnels minimaux de capteurs satisfaisant le critère de diagnosticabilité.
Une autre méthode de placement de capteurs est proposée dans (Travé-Massuyès et al. [2006]). Cette méthode généralise les travaux effectués dans (Travé-Massuyès et al. [2001], Travé-Massuyès et al. [2003]) qui ne considèrent qu'une interprétation causale. Ceci ne permet pas de trouver toutes les relations de redondance analytique **RRA**s. En revanche, la méthode (Travé-Massuyès et al. [2006]) considère toutes les interprétations causales possibles des contraintes du modèle de comportement du système et par conséquent, toutes les **RRA**s peuvent être obtenues. Le point de départ de cette méthode consiste à supposer que toutes les variables du système sont mesurées. Puis on procède au retrait des capteurs jusqu'à la satisfaction du cahier des charges considéré.

Dans la continuité du travail présenté dans (Travé-Massuyès et al. [2003]), une méthode pour concevoir un système de capteurs satisfaisant un cahier des charges avec un coût minimal est proposée dans (Spanache et al. [2004]). Cette méthode permet de trouver la combinaison de capteurs additionnels la plus adaptée qui vérifie un certain degré de diagnosticabilité. L'algorithme utilisé pour trouver le système optimal de capteurs commence par un ensemble initial d'individus candidats appelé *population initiale* et en utilisant des opérateurs génétiques **croisement**, **mutation**, **selection**, une nouvelle population avec les meilleurs individus est produite à chaque étape itérative. Après un nombre d'itérations qui dépend de la complexité du problème, l'algorithme trouve une bonne solution au problème.

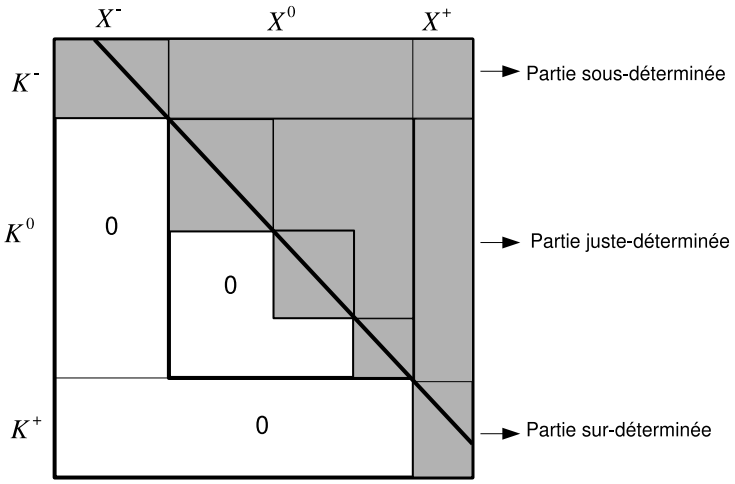


FIG. 5.2 – La décomposition Dulmage-Mendelsohn

### 5.2.2 Méthode basée sur la décomposition Dulmage-Mendelshon

(Frisk and Krysander [2007]) ont proposé un algorithme pour déterminer les capteurs qu'il faut ajouter pour obtenir la détectabilité et la localisabilité maximales de défauts spécifiés. Cet algorithme permet de trouver tous les ensembles minimaux de capteurs satisfaisant les contraintes de détectabilité et de localisabilité en prenant en compte le coût associé à chaque capteur. Cette méthode est basée sur la représentation structurelle du système (décomposition Dulmage-Mendelshon (Dulmage and Mendelsohn [1959], Pothen and Chin-Ju [1990]) représentée par la figure 5.2). Comme la plupart des approches structurelles, elle peut manipuler les modèles très diveres : non linéaires et algébriques différentiels d'une façon efficace.
Supposons que $K = \{k_1, ..., k_n\}$ l'ensemble des contraintes modélisant le comportement d'un système $\Sigma$ et $\mathbf{F} = \{\mathcal{F}_1, ..., \mathcal{F}_m\}$ soit l'ensemble des défauts considérés dans ce système. Supposons que $k_{\mathcal{F}_i} \in K$ représente la contrainte qui est non satisfaite du fait d'un défaut $\mathcal{F}_i \in \mathbf{F}$. Ces caractérisations de détectabilité et localisabilité sont définies ci-après :

**Définition 5.2.9.** *Un défaut $\mathcal{F}_i$ est structurellement détectable dans un modèle $K$ si $k_{\mathcal{F}_i} \in K^+$ avec $K^+$ est l'ensemble de contraintes sur-déterminé (voir la section B.2).*

**Définition 5.2.10.** *Un défaut $\mathcal{F}_i$ est structurellement discriminable d'un défaut $\mathcal{F}_j$ dans un modèle $K$ si $k_{\mathcal{F}_i} \in (K^+ \backslash \{k_{\mathcal{F}_j}\})$*

Ces deux définitions sont utilisées pour résoudre le problème du placement de capteurs. Une hypothèse de base de cette approche est que le modèle du système doit être juste déterminé et qu'il ne doit pas avoir une partie sous-déterminée.

TAB. 5.1 – La matrice structurelle du système

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 0 | 1 |
| $k_2$ | 0 | 1 | 1 | 1 | 0 |
| $k_3$ | 0 | 0 | 1 | 0 | 1 |
| $k_4$ | 0 | 0 | 0 | 1 | 1 |
| $k_5$ | 0 | 0 | 0 | 0 | 1 |

Considérons, par exemple, un système modélisé par les contraintes suivantes :

$$\begin{aligned} k_1 &: \dot{x_1} = x_1 + x_2 + x_5 \\ k_2 &: \dot{x_2} = x_2 + x_3 + x_4 \\ k_3 = k_{\mathcal{F}_1} &: \dot{x_3} = x_3 + x_5 + \mathcal{F}_1 \\ k_4 = k_{\mathcal{F}_2} &: \dot{x_4} = x_4 + x_5 + \mathcal{F}_2 \\ k_5 = k_{\mathcal{F}_3} &: \dot{x_5} = x_5 + \mathcal{F}_3 + u \end{aligned} \tag{5.2.1}$$

où $K = \{k_1, k_2, k_3, k_4, k_5\}$ représente le modèle comportemental du système, $\mathbf{F} = \{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3\}$ représente l'ensemble de défauts que nous devons détecter et localiser, $X = \{x_1, x_2, x_3, x_4, x_5\}$ représente l'ensemble des variables d'état utilisées et $u$ représente le flot de données (signal de commande).
En ne prenant en compte que les variables, ce système peut être représenté par la matrice structurelle 5.1.

Supposons que nous souhaitions détecter tous les défauts du système. Le modèle comportemental du système est un ensemble de contraintes juste-déterminé avec 5 contraintes et 5 variables inconnues. Dans ce cas, tous les défauts sont non détectables (selon la définition 5.2.9).
Considérons le défaut $\mathcal{F}_2$. Afin de rendre ce défaut détectable, selon la définition 5.2.9, un capteur doit être ajouté tel que la contrainte $k_4$ devienne un élément de la partie sur-déterminée du modèle. Il est facile de vérifier que le défaut $\mathcal{F}_2$ devient détectable si une des variables de cet ensemble $\{x_1, x_2, x_4\}$ est mesurée. Par exemple, en mesurant la variable $x_1$ par un capteur, la contrainte modélisant ce capteur forme une partie sur-déterminée avec les contraintes $k_4$ et $k_5$ et par conséquent, le défaut $\mathcal{F}_2$ est détectable. Par contre, en ne mesurant que la variable $x_3$, la contrainte $k_4$ sera un élément d'une partie juste-déterminée et le défaut $\mathcal{F}_2$ est non détectable.
En appliquant l'algorithme proposé dans (Frisk and Krysander [2007]), les ensembles minimaux de capteurs réalisant la détectabilité de tous les défauts sont : $\{x_1\}, \{x_2\}, \{x_3, x_4\}$.

Supposons maintenant que nous souhaitions localiser tous les défauts détectables ci-dessus. Ce problème de localisation de l'ensemble de défauts $\mathbf{F}$ peut être divisé en $|\mathbf{F}|$

sous-problèmes, un pour chaque défaut. Pour chaque défaut $\mathcal{F}_j \in \mathbf{F}$, il faut trouver toutes les mesures maximisant le nombre possible de défauts $\mathbf{F}^{'} = \{..., \mathcal{F}_i, ...\} \in F \setminus \{\mathcal{F}_j\}$ qui peuvent être discriminables du défaut $\mathcal{F}_j$. La solution du problème de localisabilité sera obtenue en combinant les résultats des sous-problèmes.
Supposons que $K^{'}$ soit l'ensemble des contraintes modélisant le système avec les capteurs ajoutés tel que tous les défauts soient détectables et $K_{\mathcal{S}}$ soit l'ensemble des contraintes terminales modélisant un ensemble additionnel de capteurs $\mathcal{S}$.
Étant donné un ensemble additionnel de capteurs $\mathcal{S}$, un défaut $\mathcal{F}_i$ est discriminable d'un défaut $\mathcal{F}_j$ dans le modèle $(K^{'} \cup K_{\mathcal{S}})$ si $k_{\mathcal{F}_i} \in ((K^{'} \cup K_{\mathcal{S}}) \setminus \{k_{\mathcal{F}_j}\})^{+}$ (selon la définition 5.2.10).

Revenons à l'exemple précédent et supposons que tous les défauts $\mathcal{F}_1$, $\mathcal{F}_2$ et $\mathcal{F}_3$ doivent être discriminables. Supposons que deux capteurs mesurant les variables $\{x_3, x_4\}$ sont ajoutés afin que tous les défauts soient détectables. Considérons le sous-problème lié au défaut $\mathcal{F}_1$, nous remarquons que les autres défauts $\{\mathcal{F}_2, \mathcal{F}_3\}$ sont discriminables de $\mathcal{F}_1$ (selon la définition 5.2.10). Par conséquent, l'ensemble $\{x_3, x_4\}$ représente une solution à notre problème. Ce résultat est obtenu en considérant qu'il n'y pas de défaut sur les capteurs.

En appliquant l'algorithme proposé dans (Frisk and Krysander [2007]), les ensembles minimaux de capteurs réalisant la dédectabilité et la localisabilité de tous les défauts sont : $\{x_1, x_3\}$, $\{x_1, x_4\}$, $\{x_2, x_3\}$, $\{x_2, x_4\}$, $\{x_3, x_4\}$.
Dans (Frisk and Krysander [2007]), le cas où les capteurs peuvent être défectueux est aussi considéré.

Ces résultats présentés démontrent qu'il est possible de trouver tous les ensembles minimaux de capteurs satisfaisant les critères de détectabilité et localisabilité maximale en utilisant seulement la structure du modèle.

## 5.3 Limites des méthodes de placement de capteurs existantes

Dans cette partie, nous avons présenté des méthodes de placement de capteurs satisfaisant des critères de contrôlabilité, d'observabilité, de monitorabilité ou de diagnosticabilité. Nous nous concentrerons particulièrement sur les méthodes satisfaisant des critère de diagnosticabilité parce que les autres approches ne répondent pas au problème posé.

La méthode proposée dans (Travé-Massuyès et al. [2001]) est basée sur l'addition consécutive des capteurs jusqu'à la satisfaction du critère de diagnosticabilité. Le degré de complexité de cette méthode est très grand parce que nous devons chercher les relations de redondance analytique pour chaque ensemble de capteurs ajouté. Le nombre de **RRA**s

résultant de chaque ensemble de capteurs est exponentiel avec la redondance du système. Cela suppose que l'on soit capable de trouver toutes ces **RRA**s.

La méthode proposée dans (Travé-Massuyès et al. [2006]) est basée sur le retrait consécutif des capteurs jusqu'à la satisfaction du critère de diagnosticabilité. Cette méthode conduit aussi à une très grande complexité parce que le nombre de **RRAs** croit exponentiellement avec la redondance d'un système, et notamment avec son nombre de capteurs. Le fait de mesurer toutes les variables du système dans une première étape conduit à une redondance maximale.

(Frisk and Krysander [2007]) propose une méthode de placement de capteurs basée sur la représentation structurelle du système (décomposition Dulmage-Mendelshon). La complexité de cette méthode est inférieure à la complexité des méthodes ci-dessus et elle n'exige pas la conception de **RRA**s au préalable. Mais cette méthode n'est pas générale dans le sens où elle ne prend en compte que les systèmes juste-déterminés qui ne contiennent pas de partie sous-déterminée.

## 5.4 Conclusion

Le problème du placement de capteurs a été introduit dans ce chapitre. Il s'agit de chercher les ensembles minimaux de capteurs qui satisfont des critère différents. Plusieurs méthodes de placement de capteurs ont été présentées et nous nous sommes concentrés spécialement sur les méthodes qui satisfont des critères de diagnosticabilité. Nous avons remarqué que les méthodes proposées dans (Travé-Massuyès et al. [2001], Travé-Massuyès et al. [2006]) sont limitées par une grande complexité de calcul et par les méthodes de génération de **RRA**s qui doivent être exhaustives. Ces méthodes exigent de chercher au préalable toutes les **RRA**s du système. Nous avons remarqué aussi que la méthode proposée dans (Frisk and Krysander [2007]) permet de trouver les placements de capteurs sans la conception des **RRA**s au préalable mais ne permet de traiter que les systèmes juste-déterminés qui ne contiennent pas de partie sous-déterminée. Dans le chapitre suivant, nous allons présenter une méthode de placement de capteurs avec un degré de complexité comparable et qui permet de traiter tous les systèmes décrit par leur structure. Cette méthode n'exige pas la conception des **RRA**s au préalable.

# Chapitre 6

# Nouvelle approche structurelle pour le placement de capteurs

## 6.1 Introduction

Les principales méthodes du placement de capteurs ont été présentées dans le chapitre 5. Nous avons présenté en détail une méthode basée sur l'addition ou le retrait consécutif de capteurs afin de trouver le placement de capteurs qui satisfait des critères de diagnosticabilité. Cette méthode requiert la conception préalable de relations de redondance analytique. Par conséquent, le degré de complexité de cette méthode est très grand. Nous avons aussi présenté une autre méthode basée sur une représentation structurelle d'un système (décomposition Dulmage-Mendelshon). Cette méthode n'exige pas la conception de **RRA**s au préalable et la complexité de cette méthode est faible. Néanmoins cette méthode ne s'applique qu'aux systèmes juste-déterminés qui ne contiennent pas de parties sous-déterminées. Or, dans la pratique, un système sans ses capteurs est souvent sous-déterminé.

Dans ce chapitre, nous allons présenter une nouvelles approche pour le placement de capteurs prenant en compte des spécifications de diagnosticabilité et de détectabilité. Cette approche est présentée dans (Yassine et al. [2007a], Yassine et al. [2007b], Yassine et al. [2008]). Deux types de spécifications sont considérées : des spécifications complètes, où tous les ensembles diagnosticables, discriminables et non détectables sont spécifiés, et des spécifications partielles, où seul l'ensemble des contraintes qui doivent être diagnosticables et l'ensemble des contraintes qui doivent être au moins détectables sont spécifiés. Cette approche pourrait conduire facilement à des méthodes permettant d'appréhender d'autres types de spécifications. Nous nous sommes néanmoins limités à deux cas : l'un où les spécifications sont complètes et l'autre où elles sont très partielles. Il est à noter

que l'ingénieur devra préférer la seconde car les spécifications complètes sont difficiles à établir. Ces méthodes s'appliquent aux systèmes pour lesquels seule la structure est connue et ne requiert pas un calcul préalable des relations de redondance analytique.
Elles présentent aussi l'avantage de s'appliquer à des systèmes sur-déterminés, juste-déterminés ou sous-déterminés.
Les méthodes proposées sont basées sur l'étude des propriétés des matrices structurelles pour établir les propriétés de détectabilité, de discriminabilité et de diagnosticabilité. Elles se basent aussi sur les algorithmes d'optimisation combinatoire pour déterminer de placement de capteurs satisfaisant au cahier des charges de coût minimal.

## 6.2 Formulation du problème

Dans le chapitre 3, nous avons rappelé qu'un problème de diagnostic se résoud en deux étapes : la détection de défauts et la localisation de défauts. La première étape est basée sur des tests de consistance reposant sur des sous-systèmes testables. Supposons que $\mathbb{K}$ soit l'ensemble des sous-systèmes testables d'un système $\Sigma$. Si $K \in \mathbb{K}$ est inconsistant, cela signifie qu'au moins, un des modes correspondant aux contraintes de $K$ n'est pas réel. Pour le diagnostic, il est important de tracer toutes les contraintes appartenant aux sous systèmes testables de $\mathbb{K}$ parce que c'est nécessaire pour résoudre le sous-problème de localisation. L'analyse diagnostique qui fournit des conclusions générales en termes de modes sur l'état actuel du système s'appuie sur les modes de défauts testés par une **RRA**. La performance d'un système de diagnostic est fortement liée à l'ensemble $\mathbb{K}$ et par conséquent, liée à l'ensemble des contraintes du système $\Sigma$ qui inclut des contraintes terminales faisant apparaître des flots de données (observations, consignes, commandes). Des capteurs additionnels engendrent des contraintes terminales additionnelles dans le système $\Sigma$, et par conséquent mènent à de nouveaux sous-systèmes testables dans $\mathbb{K}$. L'ensemble de sous-systèmes testables $\mathbb{K}$ peut être obtenus par les méthodes présentées dans les chapitres 3 et 4. De manière simple, une fois que $\mathbb{K}$ est obtenu, il est facile d'étudier la performance du système de diagnostic en terme de détectabilité, discriminabilité et diagnosticabilité. Si la performance résultant ne correspond pas à la performance exigée, l'ensemble de contraintes du système $\Sigma$ peut être complété jusqu'à la satisfaction de la performance recherchée. Cependant, ce processus requiert beaucoup de calculs parce que la génération des sous-systèmes testables requiert beaucoup de temps.

Un autre algorithme de placement de capteurs est proposé. Cet algorithme ne requiert pas le calcul préalable des sous-systèmes testables. Il résout directement le problème en étudiant la structure du système $\Sigma$.
Supposons que $K_\Sigma$ soit l'ensemble des contraintes modélisant les modes *ok* d'un système

$\Sigma$. Supposons que $var(K_\Sigma) = \bigcup_{k \in K_\Sigma} var(k)$ soit l'ensemble des variables apparaissant dans les contraintes $K_\Sigma$. Le problème à résoudre est : quelles doivent être les contraintes complémentaires modélisant les capteurs à ajouter pour satisfaire la performance de diagnosticabilité requise.

Pour résoudre le problème de placement de capteurs, nous allons définir les notions de déductibilité, de discriminabilité et de diagnosticabilité présentées dans le chapitre 5 de façon plus globale car les notions de faible et de forte discrimination ne font pas de différence dans l'algorithme que nous allons proposer. Ces définitions seront utilisées pour les preuves des théorèmes et des lemmes pour la conception du placement de capteurs. Afin d'analyser les notions de déductibilité, de discriminabilité et de diagnosticabilité, nous devons constuire la matrice de siganture des défauts. Les lignes de cette matrice correpondent à l'ensemble de **SST**s déduit de $K_\Sigma$ et les colonnes correpondent aux contraintes construisant ces **SST**s.

**Définition 6.2.1.** *Supposons que $\mathbb{K}$ soit un ensemble de **SST**s déduit de $K_\Sigma$. Une contrainte $k \in K_\Sigma$ est détectable (Struss et al. [2002]) dans $\mathbb{K}$ si et seulement si $\exists K_i \in \mathbb{K} / k \in K_i$. Par extension, un ensemble de contraintes $K \subset K_\Sigma$ est détectable dans $\mathbb{K}$ si $\forall k_i \in K$, $k_i$ est détectable dans $\mathbb{K}$.*

Dans la section 5.2.1, des définitions de discriminabilité faible, discriminabilité forte, diagnosticabilité faible et diagnosticabilité forte sont présentées. Dans ce chapitre nous nous intéressons aux définitions de discriminabilité et diagnosticabilité sans prendre en compte la notion de faible et de forte.

**Définition 6.2.2.** *Deux contraintes $(k_1, k_2) \in K_\Sigma^2$ sont discriminables (Struss et al. [2002]) dans $\mathbb{K}$ si et seulement si : $\exists K_i \in \mathbb{K} / \, k_1 \in K_i \, and \, k_2 \notin K_i$ ou si $\exists K_j \in \mathbb{K} / \, k_2 \in K_j \, and \, k_1 \notin K_j$. Par extension, les contraintes d'un ensemble $K \subset K_\Sigma$ sont discriminables dans $\mathbb{K}$ si : $\forall (k_i, k_j) \in K^2$ avec $k_i \neq k_j$, $k_i$ et $k_j$ sont discriminables dans $\mathbb{K}$.*

Évidemment, la non détectabilité de deux contraintes $(k_1, k_2)$ implique non discriminabilité de ces contraintes.

**Définition 6.2.3.** *Une contrainte $k \in K_\Sigma$ est diagnosticable (Console et al. [2000], Struss et al. [2002]) dans $\mathbb{K}$ si et seulement si : elle est détectable et $\forall k_j \in (K_\Sigma \backslash k)$, $(k, k_j)$ est discriminable dans $\mathbb{K}$. Par extension, les contraintes $K \subset K_\Sigma$ sont diagnosticables dans $\mathbb{K}$ si et seulement si : $\forall k_i \in K$, $k_i$ sont diagnosticables dans $\mathbb{K}$.*

Nous avons défini dans le chapitre 4 qu'une contrainte terminale satisfaisant $card(var(k)) = 1$, modélise habituellement un capteur ou un actionneur. Ce concept est aussi important pour le placement de capteurs. Notons que si un capteur candidat mesure non pas une variable $v$ mais une combinaison de plusieurs variables $v_1, v_2, ..., v_n$,

une nouvelle contrainte $k$ satisfaisant $var(k) = \{v_1, ..., v_n, v_*\}$, où $v_* = f(v_1, v_2, ..., v_n)$ est une variable virtuellement mesurée, doit être ajoutée dans $K_\Sigma$. Alors, la résolution est similaire au problème standard.

Dans le diagnostic de défauts, le placement de capteurs doit satisfaire à des spécifications liées à la détectabilité et à la diagnosticabilité. Pour simplifier les écritures, nous identifierons les contraintes avec les composants qu'elles modélisent en considérant une relation bijective entres ces deux ensembles. Cela conduit à définir des spécifications plus précises : non pas sur les composants, mais sur les contraintes. Cette hypothèse n'est pas nouvelle puisque toutes les méthodes de placement de capteurs pour le diagnostic la font.
Dans ce chapitre deux types de spécifications sont considérés : des spécifications complètes et des spécifications partielles.

Les spécifications complètes consistent à partitionner l'ensemble des contraintes $K_\Sigma$ en les sous-ensembles suivants :

- l'ensemble de contraintes $K_{diag}$ qui doit être diagnosticable.
- l'ensemble des sous-ensembles de contraintes $\mathbb{K}_{nondis} = \{\ldots, K_i, \ldots\}$ où chaque ensemble $K_i$ doit être non discriminable mais détectable.
- l'ensemble des contraintes $K_{nondet}$ qui doit être non détectable.

Les spécifications complètes $K_{diag}$, $\mathbb{K}_{nondis}$ et $K_{nondet}$ pour le problème du placement de capteurs sont consistantes si les deux propriétés suivantes sont satisfaites :

1. les ensembles de contraintes doivent satisfaire :
   - $K_{nondet} \cap K_{diag} = \phi$.
   - $\forall K_i \in \mathbb{K}_{nondis}, K_i \cap K_{nondet} = \phi$.
   - $\forall K_i \in \mathbb{K}_{nondis}, K_i \cap K_{diag} = \phi$.
   - $\forall (K_i, K_j) \in \mathbb{K}^2_{nondis}$, $K_i \cap K_j = \phi$ si $K_i \neq K_j$.
2. l'union de toutes les contraintes apparaissant dans $K_{diag}$, $\mathbb{K}_{nondis}$ et $K_{nondet}$ doit correspondre à $K_\Sigma$ : $K_\Sigma = K_{diag} \cup K_{nondet} \cup \bigcup_{K_i \in \mathbb{K}_{nondis}} K_i$.

Si ces propriétés sont satisfaites, ces spécifications complètes sont qualifiées de consistantes dans $K_\Sigma$.

Les spécifications partielles consistent à partitionner l'ensemble des contraintes $K_\Sigma$ en les sous-ensembles suivants :

- l'ensemble de contraintes $K_{diag}$ qui doit être diagnosticable.
- l'ensemble de contraintes $K_{det}$ qui doit être au moins détectable.

Les spécifications partielles $K_{diag}$ et $K_{det}$ pour le problème de placement de capteurs sont significatifs si ces propriétés suivantes sont satisfaites :

- $K_{diag} \cap K_{det} = \emptyset$.
- $(K_{diag} \cup K_{det}) \subseteq K_\Sigma$.

Si ces propriétés sont satisfaites, les spécifications partielles sont qualifiées de consistantes

dans $K_\Sigma$.

Satisfaire les spécifications de diagnosticabilité exige des informations délivrées par des capteurs. Supposons que $\Sigma'$ représente le système $\Sigma$ avec des capteurs additionnels representés par un ensemble de contraintes terminales où $K_{\Sigma'}$ contient les contraintes $K_\Sigma$ du système $\Sigma$ plus les contraintes terminales additionnelles modélisant les capteurs additionnels $K^{ter}$ : $K_{\Sigma'} = K_\Sigma \cup K^{ter}$. Par conséquent, résoudre le problème du placement de capteurs consiste à déterminer les contraintes terminales additionnelles $K^{ter}$ dans $K_{\Sigma'}$ qui mènent à la satisfaction de spécifications complètes ou partielles.

Dans les sections suivantes, les propriétés de diagnosticabilité des matrices structurelles sont établies et utilisées pour la conception de placements de capteurs satisfaisant à des spécifications de diagnosticabilité.

## 6.3 Propriétés de base des matrices structurelles

Avant de préciser les propriétés de diagnosticabilité, quelques propriétés de base des matrices structurelles doivent être établies.

Selon la définition présentée dans le chapitre 3, un **SST** est un ensemble minimal de contraintes $K$ tel qu'il y a au moins une variable qui peut être instanciée au moyen de deux ensembles différents de $K$. L'instanciation d'une variable $v$ s'appuyant sur un ensemble de contraintes correspond à une propagation de valeurs (Apt [2003]) commençant habituellement, mais pas nécessairement, par des contraintes terminales et menant à $v$. Par conséquent, un **SST** peut également être vu comme la résultante de deux propagations de valeurs distinctes menant à une même variable. Ce point de vue a été adopté comme un outil théorique pour développer les preuves qui vont suivre.

Supposons que $k_1$ et $k_2$ soient deux contraintes. La propagation d'une variable $v$ entre $k_1$ et $k_2$ est possible seulement si $v \in var(k_1) \cap var(k_2)$. La variable $v$ est propageable entre $k_1$ et $k_2$ : $v$ constitue un lien entre $k_1$ et $k_2$. Dans la matrice structurelle correspondant, ce lien est représenté par une ligne épaisse :

| | $v$ | ... |
|---|---|---|
| $k_1$ | 1 | ... |
| $k_2$ | 1 | ... |

Considérons maintenant un système, défini par $K_\Sigma = \{k_1, k_2, k_3, k_4, k_5\}$ avec $var(k_1) = \{v_1, v_3\}$, $var(k_2) = \{v_1, v_2\}$, $var(k_3) = \{v_2, v_3\}$, $var(k_4) = \{v_2\}$ et $var(k_5) = \{v_3\}$. Les contraintes terminales $k_4$ et $k_5$ modélisent des capteurs ou actionneurs. Chaque contrainte terminale contient des informations connues sous la forme de flots de données

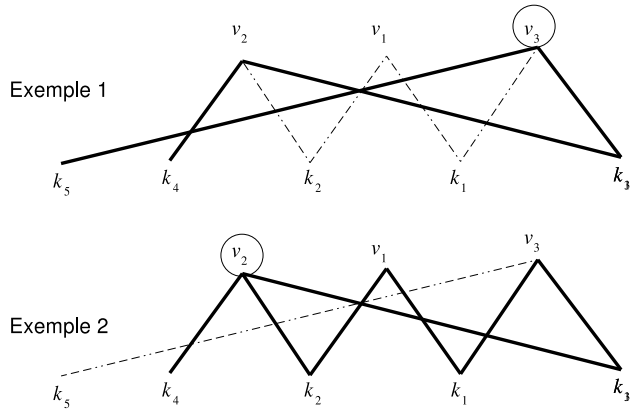


FIG. 6.1 – Lien entre propagations et RRA

(mesures,..).
La figure 6.1 représente des exemples de propagations qui mènent à des **SST**s en utilisant un graphe bipartie. Mais dans un graphe bipartie, les liens n'apparaissent pas clairement : ils correspondent à des chemins alternatifs (ou chaînes alternées) avec ce modèle : contrainte-variable-contrainte. Les liens apparaissent plus clairement dans les matrices structurelles comme des lignes liant deux contraintes. Dans les matrices structurelles suivantes, les variables entourées par un cercle représentent les variables qui peuvent être instanciées deux fois. Les chemins correspondant aux propagations des **SST**s ont été dessinés par la figure 6.1.

[H]

Example 1

| | $v_1$ | $v_2$ | $v_3$ |
|---|---|---|---|
| $k_1$ | 1 | | 1 |
| $k_2$ | 1 | 1 | |
| $k_3$ | | 1 | (1) |
| $k_4$ | | 1 | |
| $k_5$ | | | (1) |

Example 2

| | $v_1$ | $v_2$ | $v_3$ |
|---|---|---|---|
| $k_1$ | 1 | | 1 |
| $k_2$ | 1 | (1) | |
| $k_3$ | | 1 | 1 |
| $k_4$ | | (1) | |
| $k_5$ | | | 1 |

Le concept des contraintes liées doit être formalisé parce que la discriminabilité dépend de ce concept. Avant de définir *contraintes liées*, le concept de *contraintes interconnectées* doit être introduit. Les contraintes du système $\Sigma$ peuvent être modélisées par un graphe

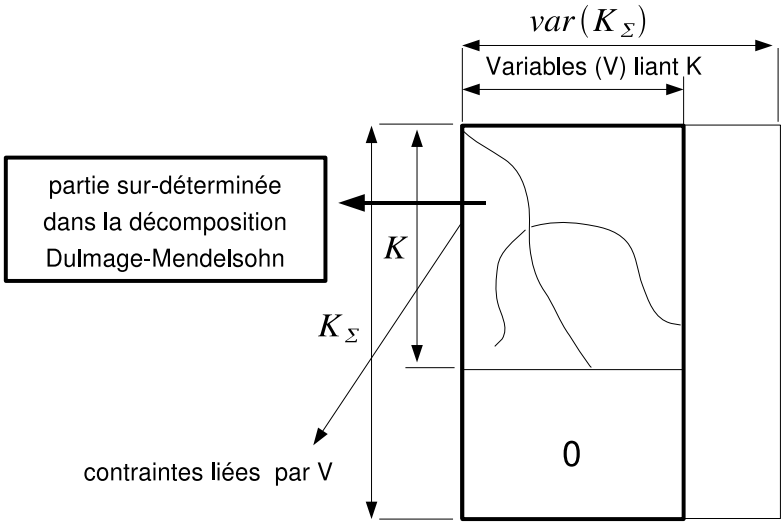


FIG. 6.2 – Matrice structurelle d'un ensemble de contraintes $K$, qui est lié par un ensemble de variables $V$

bipartie non orienté $(K_\Sigma, var(K_\Sigma), A_\Sigma)$ où $A_\Sigma$ est l'ensemble des arcs. Chaque arc $e = (k, v)$ signifie que $v \in var(k)$.

**Définition 6.3.1.** *Un ensemble de contraintes $K \subset K_\Sigma$ est interconnecté par un ensemble de variables $V \subset var(K_\Sigma)$ si et seulement s'il y a un arbre $(K, V, A) \subset (K_\Sigma, var(K_\Sigma), A_\Sigma)$ avec des contraintes aux extrémités (voir Bollobás [1998] par exemple), qui satisfait $card(V) = card(K) - 1$.*

Pour préciser le lien avec la théorie des graphes, si $K$ est interconnecté par $V$ dans $K_\Sigma$, il y a nécessairement un couplage complet par rapport aux variables. La notion d' *ensemble de contraintes liées* peut maintenant être introduite

**Définition 6.3.2.** *Un ensemble de contraintes $K \subset K_\Sigma$ est lié dans $K_\Sigma$ par un ensemble de variables $V \subset var(K_\Sigma)$ si et seulement si $K$ est interconnecté par $V$ et si et seulement si les autres contraintes de $K_\Sigma$ (c'est-à-dire, $K_\Sigma \backslash K$) ne contiennent plus aucune variable de $V$. Les variables de $V$ sont appelées variables liant $K$. Elles sont dénotées : $var_{linking}(K, K_\Sigma)$.*

La forme d'une matrice structurelle comportant des contraintes liées est représentée par la figure 6.2.

Nous remarquons que cette forme correspond bien à une partie sur-déterminée dans la décomposition Dulmage-Mendelshon vérifiant : $card(K) = card(V) + 1$ (c'est-à-dire qu'il y a un couplage complet par rapport aux variables et qu'il y a une contrainte en

plus qui n'appartient pas au couplage).
Le concept de *contraintes liées* est fortement connecté à la notion de discriminabilité.

**Lemme 6.3.1.** *Un ensemble de contraintes $K \subset K_\Sigma$ lié par un ensemble de variables $V \subset var(K_\Sigma)$ est nécessairement non discriminable.*

*Démonstration.*

1. Parce que les variables dans $V$ apparaissent seulement dans les contraintes appartenant à $K$, la seule manière de propager des variables est d'utiliser les contraintes dans $K$ et les variables dans $V$.
2. Parce qu'il existe un arbre $(K, V, A) \subset (K_\Sigma, var(K_\Sigma), A_\Sigma)$ avec des contraintes aux extrémités, permettant d'instancier toutes les variables dans $V$, cela implique au moins les propagations définies par l'arbre.

Par conséquent, toutes les contraintes sont invariablement trouvées ensemble dans les **SST**s. Afin d'améliorer la clarté de ces explications, introduisons la notion des variables souches (stump variables). □

**Définition 6.3.3.** *Un ensemble de variables $var(K)$ apparaissant exclusivement dans un ensemble de contraintes $K$ (c'est-à-dire pas dans $K_\Sigma \backslash K$) est nommé variables souches dans $K_\Sigma$ par rapport à $K$. Elles sont notées : $var_{stump}(K, K_\Sigma)$.*

Par exemple, l'ensemble de variables $V$ qui lient un ensemble de contraintes $K$ appartient aux variables souches $var_{stump}(K, K_\Sigma)$ avec $K \subset K_\Sigma$.

Un ensemble de contraintes ne peut pas être utilisé pour générer un **SST** si les contraintes de cet ensemble sont liées et s'il y a des variables additionnelles qui ne peuvent pas être propagées. Ces contraintes seront qualifiées d'isolées. La détectabilité dépend de ce concept.

**Définition 6.3.4.** *Un ensemble de contraintes $K \subset K_\Sigma$ est isolé dans $K_\Sigma$ par un ensemble de variables $V \subset var(K_\Sigma)$ s'il est lié par $V$ et il y a au moins une variable dans $var(K) \backslash V$ qui n'appartient pas aux autres contraintes de $K_\Sigma$ (c'est-à-dire $K_\Sigma \backslash K$). Si cet ensemble contient seulement une contrainte, la condition (lié par $V$) disparait.*

La forme d'une matrice structurelle comportant des contraintes isolées est représentée par la figure 6.3.

Nous remarquons que cette forme correspond bien à une partie juste ou sous-déterminée dans la décomposition Dulmage-Mendelshon vérifiant : $card(K) \leq card(V)$ .
Le concept de contraintes isolées est fortement lié à la notion de détectabilité.

**Lemme 6.3.2.** *Un ensemble de contraintes $K \subset K_\Sigma$ isolé dans $K_\Sigma$ par $V$ est nécessairement non détectable.*

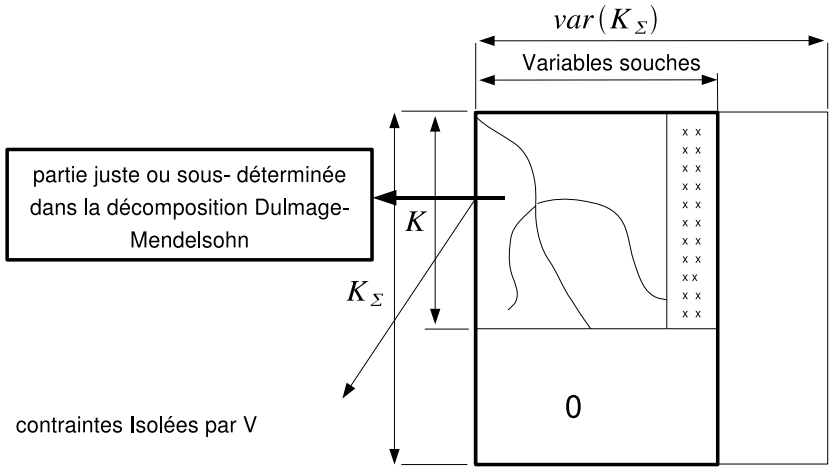


FIG. 6.3 – Matrice structurelle d'un ensemble de contraintes, qui est isolé par un ensemble de variables $V$

*Démonstration.* Les contraintes $K$ isolées dans $K_\Sigma$ par $V$, apparaissent toujours ensemble dans les **SST**s parce que, par définition, elles sont liées par $V$. Puisque dans des contraintes isolées, il y a au moins une variable additionnelle dans $var(K)$, qui n'apparaît pas dans les autres contraintes (c'est-à-dire $K_\Sigma \backslash K$), il n'est pas possible d'instancier cette variable donc, cet ensemble de contraintes ne peut pas être impliqué dans un **SST** : les contraintes $K$ sont alors nécessairement non détectables. □

## 6.4 Propriétés de diagnosticabilité des matrices structurelles

Cette section montre le lien entre les ensembles de contraintes et les propriétés de détectabilité et de diagnosticabilité. Premièrement, il est évident qu'ajouter des contraintes additionnelles terminales à toutes les variables $var(k)$ apparaissant dans une contrainte $k$, assure la diagnosticabilité de $k$.

**Lemme 6.4.1.** *Soit $k \in K_\Sigma$ une contrainte. Si des contraintes terminales additionnelles connectées à toutes les variables de $var(k)$ sont ajoutées, alors la contrainte $k$ est nécessairement diagnosticable.*

*Démonstration.* Puisqu'il y des contraintes terminales additionnelles connectées à chaque variable dans $var(k)$, une valeur peut être assignée pour chaque variable. En conséquence, il y a au moins un **SST**s contenant la contrainte $k$ plus les contraintes terminales addition-

nelles connectées aux variables $var(k)$. Par conséquent, la contrainte $k \in K_\Sigma$ est nécessairement diagnosticable parce qu'il y a un **SST** qui ne contient pas d'autres contraintes de $K_\Sigma$ (c'est-à-dire $K_\Sigma \backslash \{k\}$). □

Le lemme 6.4.1 peut être appliqué directement à toutes les contraintes d'un ensemble de contraintes.

**Corollaire 6.4.2.** *Si des contraintes terminales additionnelles connectées à toutes les variables $var(K)$, avec $K \in K_\Sigma$, sont ajoutées, alors chaque contrainte $k \in K$ est diagnosticable.*

*Démonstration.* Puisqu'il y des contraintes terminales additionnelles connectées à toutes les variables apparaissant dans les contraintes du système, une valeur peut être assignée pour chaque variable. En conséquence, selon le lemme 6.4.1, il y a au moins $card|K_\Sigma|$ **SST** tel que chaque **SST** contient une seule contrainte $k \in K_\Sigma$ du système plus les contraintes terminales additionnelles connectées aux variables $var(k)$, et ne contient pas d'autres contraintes de $K_\Sigma$. Par conséquent, toutes les contraintes du système sont nécessairement diagnosticables. □

Dans le lemme 6.3.2, un rapport entre les contraintes isolées et la propriété de détectabilité a été présenté. Le lemme suivant généralise les résultats précédents.

**Lemme 6.4.3.** *Une condition suffisante pour qu'un sous-ensemble de contraintes $K \subset K_\Sigma$ soit non détectable est qu'il y ait un tuple $(K_1, \ldots, K_m)$ de $m$ ensembles de contraintes qui soit une partition $\mathcal{P}(K)$ de $K$ telle que chaque $K_i$ soit isolé dans $K_\Sigma \backslash \bigcup_{j<i} K_j$ ($K_1$ est un cas limite : il doit être isolé dans $K_\Sigma$).*

*Démonstration.* Le cas de $K_1$ a été discuté dans le lemme 6.3.2 : puisque l'ensemble des contraintes $K_1$ sont isolées dans $K_\Sigma$, elles sont non détectables et par conséquent elles ne peuvent pas être incluses dans les **SST**s. Alors, les contraintes candidates restantes pour les **SST**s appartiennent à $K_\Sigma \backslash K_1$. Puisque $K_2$ est isolé dans $K_\Sigma \backslash K_1$, les contraintes de $K_2$ sont non détectables. Le raisonnement peut être prolongé ainsi à chaque $K_i$. Par conséquent, les contraintes dans $K = \bigcup_i K_i$ sont non détectables. □

La figure 6.4 indique la forme d'une matrice structurelle des contraintes non détectables.

Nous remarquons que cette matrice correspond bien à plusieurs parties juste ou sous-déterminées dans la décomposition Dulmage-Mendelshon. Considérons, par exemple, un système modélisé par la matrice structurelle suivante :

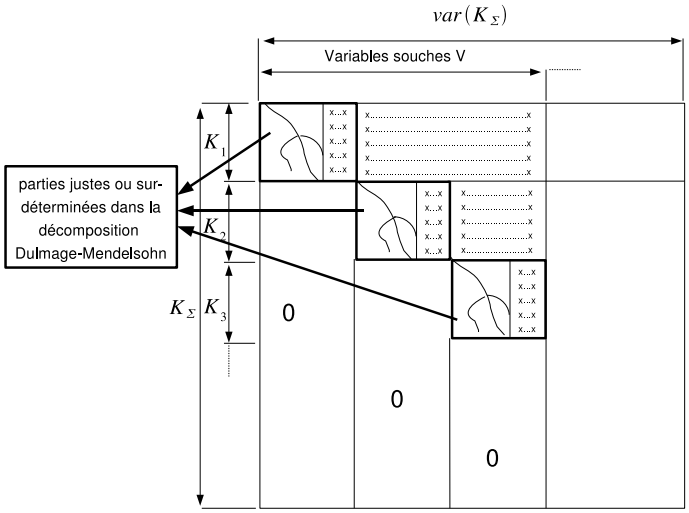


FIG. 6.4 – Matrice structurelle des contraintes non détectables

| | $v_1$ | $v_2$ | $v_3$ | $v_4$ | $v_5$ | $v_6$ |
|---|---|---|---|---|---|---|
| $k_1$ | 1 | 0 | 0 | 1 | 0 | 0 |
| $k_2$ | 0 | 1 | 1 | 0 | 1 | 0 |
| $k_3$ | 0 | 1 | 1 | 0 | 1 | 0 |
| $k_4$ | 0 | 0 | 0 | 1 | 0 | 1 |
| $k_5$ | 0 | 0 | 0 | 1 | 1 | 1 |

Supposons que $K = \{k_1, k_2, k_3\}$ soit un sous-ensemble de contraintes qui doit être non détectables.
Dans cet exemple, il y a un tuple $(\{k_1\}, \{k_2, k_3\})$ tel que chaque élément $K_i$ satisfait le lemme 6.4.3. S'il n'y a pas de contrainte terminale additionnelle contenant $v_1$, $v_2$ et $v_3$, le sous-ensemble $K$ est nécessairement non détectable.

**Lemme 6.4.4.** *Une condition suffisante pour que chaque ensemble $K_i \subset \mathbb{K}$ appartenant à un ensemble de m ensembles de contraintes $\mathbb{K} = \{K_1, \ldots, K_m\}$ tel que $\forall K_i \neq K_j, K_i \cap K_j = \emptyset$, soit non discriminable est que chaque $K_i$ soit lié par un ensemble de variables $V_i$.*

*Démonstration.* Ce lemme est une application directe du lemme 6.3.1 pour plusieurs ensembles de contraintes. □

Considérons, par exemple, un système modélisé par la matrice structurelle suivante :

| | $v_1$ | $v_2$ | $v_3$ | $v_4$ | $v_5$ |
|---|---|---|---|---|---|
| $k_1$ | 1 | 0 | 1 | 1 | 1 |
| $k_2$ | 1 | 1 | 1 | 1 | 0 |
| $k_3$ | 1 | 1 | 1 | 0 | 1 |
| $k_4$ | 0 | 1 | 1 | 0 | 0 |
| $k_5$ | 0 | 0 | 0 | 1 | 1 |

Supposons que $K = \{k_1, k_2, k_3, k_4\}$ soit un sous-ensemble de contraintes qui doit être non discriminable. Puisque les contraintes $k_1$, $k_2$, $k_3$ et $k_4$ sont liées par $V = \{v_1, v_2, v_3\}$, le lemme 6.4.4 est satisfait. Par conséquent, $k_1$, $k_2$, $k_3$ et $k_4$ sont non discriminables s'il n'y a pas de contrainte terminale additionnelle contenant une variable de $V$.

La démonstration du théorème suivant s'appuie sur les lemmes 6.4.1, 6.4.3 et 6.4.4.

**Théorème 6.4.5.** *Soit $K_\Sigma$ un ensemble de contraintes et $K_{nondet}$, $\mathbb{K}_{nondis}$ et $K_{diag}$ des spécifications complètes pour un problème de placement de capteurs consistant dans $K_\Sigma$. Les conditions suffisantes pour que les spécifications soient satisfaites sont :*

1. *chaque ensemble $K_i$ appartenant à $\mathbb{K}_{nondis} = \{K_1, \ldots, K_m\}$ tel que $\forall K_i \neq K_j, K_i \cap K_j = \emptyset$, est lié par un ensemble de variables $V_i$ en considérant seulement les contraintes détectables $K_\Sigma \backslash K_{nondet}$*
2. *des contraintes terminales additionnelles sont ajoutées sur les variables $V_{candidat} = var(K_\Sigma) \backslash (var_{stump}(K_{nondet}, K_\Sigma) \cup \bigcup_{K_j \in \mathbb{K}_{nondis}} var_{linking}(K_j, K_\Sigma \backslash K_{nondet}))$ (voir figure 6.5).*

*Démonstration.* La preuve se fonde sur la structure de la matrice structurelle, qui s'analyse à partir du corollaire 6.4.2 et des lemmes 6.4.3 et 6.4.4. Notez que le point 2 pourrait également être énoncé pour l'ensemble de contraintes $K_\Sigma$. Cependant, il n'est pas utile d'inclure les contraintes non détectables qui n'apparaîtront pas dans les **SST**s résultants.

Conformément aux lemmes 6.4.3 et 6.4.4, les variables de $var(K_{diag})$ ne peuvent pas contenir de variable apparaissant dans les variables impliquées dans (1) et (2) c'est-à-dire, dans $var_{stump}(K_{nondet}, K_\Sigma)$ et dans $\bigcup_{K_j \in \mathbb{K}_{nondis}} var_{linking}(K_j, K_\Sigma \backslash K_{nondet})$. Alors, $var(K_{diag})$ satisfait : $var(K_{diag}) \subset V_{candidat}$. Puisque les variables de $V_{candidat}$ peuvent être instanciées avec les valeurs mesurées, toutes les contraintes de $K_{diag}$ sont diagnosticables selon le corollaire 6.4.2.

Le point qui doit être prouvé est que, dans les spécifications, $\mathbb{K}_{nondis}$ engendre bien des ensembles non discriminables et détectables et pas seulement des ensembles non discriminables comme dans le lemme 6.4.4 : la détectabilité des ensembles dans $\mathbb{K}_{nondis}$ doit être prouvée.

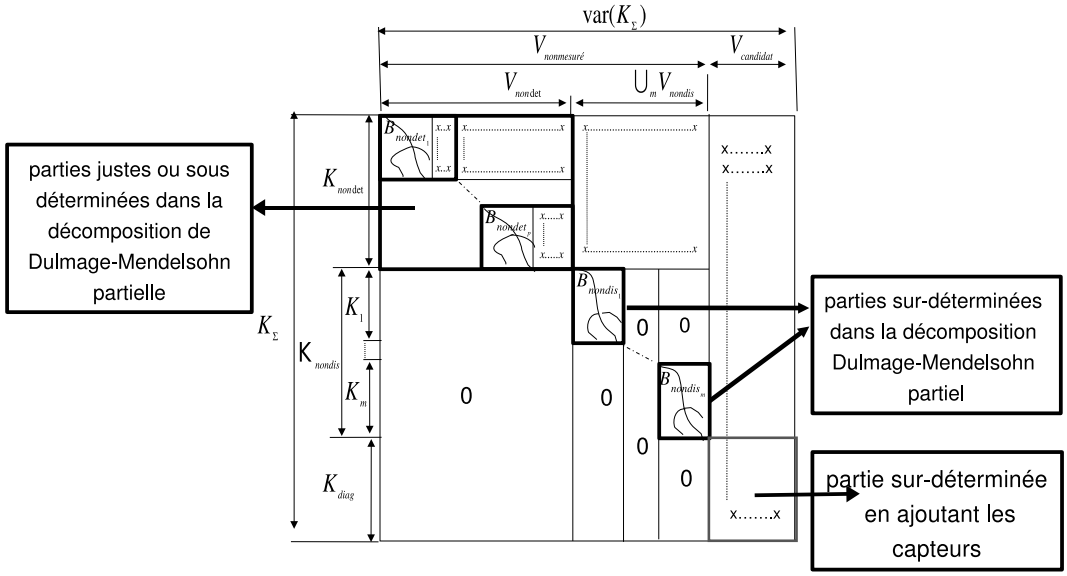


FIG. 6.5 – Forme d'une matrice structurelle satisfaisant le théorème 6.4.5

Les variables $var(K_i)$ d'un ensemble de contraintes $K_i \in \mathbb{K}_{nondis}$ peuvent être décomposées en deux ensembles : $V_i^-$ et $V_i^+$, où $V_i^- = var_{linking}(K_i, K_\Sigma \backslash K_{nondet})$ contient les variables liant $K_i$ et $V_i^+$ contient les variables restantes $V_i^+ = var(K_i) \backslash V_i^-$. Selon les lemmes 6.4.3 et 6.4.4, l'ensemble $V_i^+$ ne peut pas contenir de variables de $var_{stump}(K_{nondet}, K_\Sigma)$ et de $\bigcup_{K_j \in \mathbb{K}_{nondis}; K_j \neq K_i} var_{linking}(K_j, K_\Sigma)$. Par conséquent, $V_i^+$ satisfait : $V_i^+ \subset V_{candidat}$.

Du fait du troisième point du théorème, toutes les variables de $V_{candidat}$ sont connues : des contraintes terminales additionnelles sont en effet ajoutées. Il y a donc nécessairement un **SST** contenant toutes les contraintes dans $K_i$. Cela prouve que l'ensemble de contraintes $K_i$ est nécessairement détectable. Puisque le résultat est vérifié pour chaque $K_i \in \mathbb{K}_{nondis}$, cela prouve le théorème. □

Satisfaire le théorème 6.4.5 garantit que les spécifications sont satisfaites. Cependant, parce que le théorème fournit seulement une condition suffisante pour la diagnosticabilité, le nombre de contraintes terminales additionnelles n'est pas nécessairement minimal. Il doit être vérifié a posteriori.

Dans la section suivante, un algorithme pour extraire des blocs à partir d'une matrice structurelle est présenté. Cet algorithme est requis par des méthodes de placement de capteurs visant à la fois à satisfaire des spécifications complètes ou des spécifications partielles.

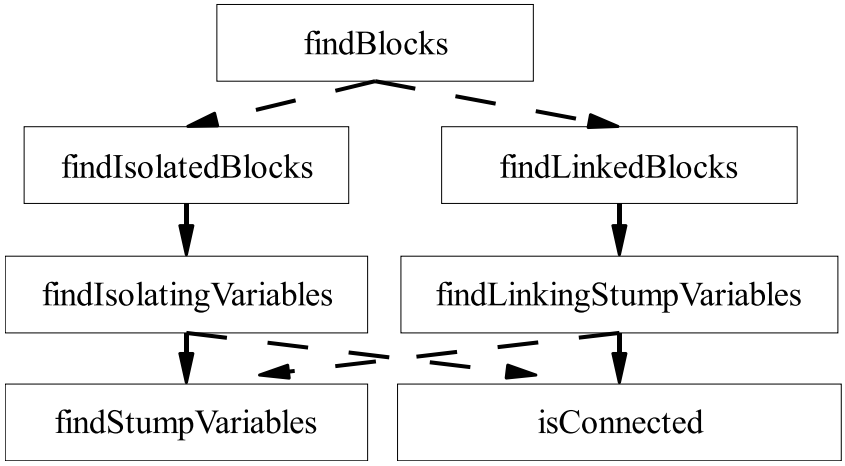


FIG. 6.6 – Schéma de dépendance de l'algorithme d'extraction des blocs

## 6.5 Extraction des blocs d'une matrice structurelle

Avant de présenter un algorithme pour extraire des blocs à partir d'une matrice structurelle $K_\Sigma$, introduisant quelques notations. Premièrement, la notion de bloc est formalisée : un *bloc* est un couple défini par $bloc = (K, V)$ où $bloc.cons = K$ et $bloc.var = V$ représentent respectivement un ensemble de contraintes et un ensemble de variables avec $bloc.var \in var(K)$. Deux blocs peuvent être fusionnés :

$$merge(block_1, block_2) = \\ block(block_1.cons \cup block_2.cons, block_1.var \cup block_2.var)$$

Un ensemble de blocs est représenté par le symbole $\mathbb{B}$. Par extension, le bloc résultant de la fusion d'un ensemble de blocs $\mathbb{B}$ est noté : $merge(\mathbb{B})$.

La Figure 6.6 représente le schéma de dépendance entre les méthodes qui vont être définies.

L'algorithme principal est appelé findBlocks (algorithm 3, page 143). Il extrait les différents blocs qui apparaissent dans le théorème 6.4.5, en considérant seulement les variables $V_\Delta$.

Afin de décrire les méthodes findIsolatedBlocks() (page 144) et findLinkedBlocks() (page 146), les notions de $Knode$ et de *buffer* de $Knodes$ sont introduites. Un $Knode$ est un couple d'ensembles de contraintes : $Knode = Knode(K^-, K^+)$, où $Knode^- = K^-$ et $Knode^+ = K^+$. Un *buffer* est un buffer de type "First In First Out" qui contient des éléments par exemple de type $Knode$. Les fonctionnalités de base sont : $buffer.push(Knode)$ et $buffer.pop()$. Ils correspondent respectivement à l'ajout d'un $Knode$ dans le buffer et

**Algorithm 3** findBlocks($K_\Sigma, V_\Delta$) : Un tuple d'un ensemble de blocs ($\mathbb{B}_{nondet}, \mathbb{B}_{nondis}, B_{diag}$), en considérant seulement les variables $V_\Delta$

**Require:** $V_\Delta \subseteq var(K_\Sigma)$
$K_\Delta \leftarrow K_\Sigma$
$\mathbb{B}_{nondet} \leftarrow$ findIsolatedBlocks($K_\Sigma, K_\Delta, V_\Delta$)
$K_\Delta \leftarrow K_\Delta \backslash merge(\mathbb{B}_{nondet}).cons$
$\mathbb{B}_{nondis} \leftarrow$ findLinkedBlocks($K_\Sigma \setminus merge(\mathbb{B}_{nondet}).\ cons, K_\Delta, V_\Delta \setminus merge(\mathbb{B}_{nondet}).var$)
$K_{diag} \leftarrow K_\Delta \backslash merge(\mathbb{B}_{nondis}).cons$
**return** $(\mathbb{B}_{nondet}, \mathbb{B}_{nondis}, block(K_{diag}, var(K_{diag})))$

au retrait d'un *Knode* du buffer.

En utilisant ces notions, l'algorithme findIsolatedBlocks() (algorithme 4, page 144) extrait l'ensemble des blocs isolés d'un ensemble de contraintes $K_\Delta \subseteq K_\Sigma$, en considérant seulement les variables $V_\Delta$. Selon le lemme 6.4.3, les contraintes appartenant aux blocs résultants sont non détectables.

Cet algorithme dépend de la méthode findIsolatingVariables(). Elle est donnée par l'algorithme 5 (page 145).

L'algorithme findLinkedBlocks() (algorithme 6, page 146) extrait l'ensemble de blocs liés d'un ensemble $K_\Delta \subseteq K_\Sigma$, en considérant seulement les variables $V_\Delta$. La structure de cet algorithme ressemble à la structure de l'algorithme 4. Selon le lemme 6.4.4, les contraintes appartenant aux blocs résultants sont non discriminables.

Cet algorithme dépend de la méthode findLinkingStumpVariables() qui est donné par l'algorithme 7 (page 146).

Finalement, selon la figure 6.6, les deux algorithmes findIsolatingVariables() et findLinkedBlocks() dépendent des deux méthodes findStumpVariables() (algorithme 8, page 147) et isInterconnected() (algorithme 9, page 147). Supposons que $cons(v)$ soit l'ensemble des contraintes contenant la variable $v$ et $cons(V) = \bigcup_{v \in V} cons(v)$ soit l'ensemble des contraintes contenant l'ensemble des variables $V$.

La méthode findBlocks($K_\Sigma$) calcule des blocs représentés dans la figure 6.5. Comme nous le verrons plus loin, ces résultats sont très utiles pour le placement de capteurs.
En effet, les contraintes appartenant à $B_{diag}.cons$ sont déjà diagnosticables. Par conséquent, chercher un placement de capteurs satisfaisant les spécifications exige que l'ensemble $K^{spec}_{diag}$ spécifié inclut $B_{diag}.cons$ :

$$B_{diag}.cons \subset K^{spec}_{diag} \tag{6.5.1}$$

De la même manière, les contraintes $merge(\mathbb{B}_{nondis}).cons \cup B_{diag}.cons$ sont déjà détectables. Par conséquent, chercher un placement de capteurs satisfaisant les spécifications

**Algorithm 4** findIsolatedBlocks($K_\Sigma, K_\Delta, V_\Delta$) : Un bloc contenant l'ensemble de sous-ensembles de contraintes isolés de $K_\Delta$ dans $K_\Sigma$ et les variables isolant ces sous-ensembles de contraintes, en considérant seulement les variables $V_\Delta$

**Require:** $K_\Delta \subseteq K_\Sigma$
**Require:** Un *buffer* existe
  $buffer \leftarrow \emptyset$
  $\mathbb{B} \leftarrow \emptyset$ {une liste vide de blocs}
  $buffer.push(Knode(\emptyset, K_\Delta))$
  **while** $buffer \neq \emptyset$ **do**
    $Knode \leftarrow buffer.pop()$
    $K \leftarrow K_\Delta \backslash Knode^-$
    $V \leftarrow$ findIsolatingVariables$(K_\Sigma, K, V_\Delta)$
    **if** $V \neq \emptyset$ **then**
      $\mathbb{B} \leftarrow (\mathbb{B}, block(Knode^+, V))$
      $K_\Delta \leftarrow Knode^-$
      $K_\Sigma \leftarrow K_\Sigma \backslash K$
      $V_\Delta \leftarrow V_\Delta \backslash V$
      $buffer \leftarrow \emptyset$
      $buffer.push(Knode(\emptyset, K_\Delta))$
    **else**
      $K^+ \leftarrow Knode^+$
      **for all** $k \in Knode^+$ **do**
        $K^- \leftarrow Knode^- \cup \{k\}$
        $K^+ \leftarrow K^+ \backslash \{k\}$
        **if** $K^- \neq K_\Delta$ **then**
          $buffer.push(Knode(K^-, K^+))$
        **end if**
      **end for**
    **end if**
  **end while**
  **return** $\mathbb{B}$

**Algorithm 5** findIsolatingVariables($K_\Sigma, K_\Delta, V_\Delta$) : Un ensemble de variables isolant $K_\Delta$ dans $K_\Sigma$, en considérant seulement les variables $V_\Delta$

**Require:** $K_\Delta \subseteq K_\Sigma$
  $V_{sutmps} \leftarrow$ findStumpVariables($K_\Sigma, K_\Delta, V_\Delta$)
  **if** $card(V_{stump}) \geq card(K_\Delta)$ **then**
    **if** $card(K_\Delta) = 1$ **then**
      **return** $V_{stump}$
    **else**
      **for** $V \in$ combinaisons de $card(K_\Delta) - 1$ variables de $V_{stump}$ **do**
        **if** isInterconnected($K_\Delta, V$) **then**
          **return** $V_{stump}$
        **end if**
      **end for**
    **end if**
  **end if**
  **return** $\emptyset$

exige que l'ensemble $K^{spec}_{nondet}$ spécifié soit inclus dans $merge(\mathbb{B}_{nondet}).cons$ :

$$K^{spec}_{nondet} \subset merge(\mathbb{B}_{nondet}).cons \tag{6.5.2}$$

## 6.6 Méthodes pour le placement de capteurs ne tenant pas compte de la déductibilité

Deux méthodes pour le placement de capteurs satisfaisant des spécifications de diagnosticabilité sont proposées dans cette section. La première est liée aux spécifications complètes : $K^{spec}_{diag}$, $\mathbb{K}^{spec}_{nondis}$ et $K^{spec}_{nondet}$ et la seconde méthode est liée aux spécifications partielles : $K^{spec}_{diag}$ et $K^{spec}_{det}$.

Il peut y avoir plusieurs placements de capteurs qui satisfont les spécifications de diagnosticabilité. Afin de choisir le plus intéressant, des critères basés sur le coût des capteurs sont considérés. Le coût de la mesure d'une variable $v$ est noté $cost(v)$. Par extension, le coût de la mesure d'un ensemble de variables $V$ est noté : $cost(V) = \sum_{v \in V} cost(v)$.

Ajouter des capteurs est équivalent à ajouter des contraintes terminales (voire la définition 3.0.1). En effet, un capteur mesurant une variable $v$ est modélisé par une contrainte : $val(t, v) = v$ où $val(t, v)$ est un flot de données venant d'un capteur. Par conséquent, d'une manière structurelle, un capteur mesurant $v$ est modélisé par une

**Algorithm 6** findLinkedBlocks($K_\Sigma, K_\Delta, V_\Delta$) : Un ensemble de blocs, où chaque bloc correspond d'une part à un ensemble de contraintes lié mais non isolé et d'autre part aux variables liantes correspondant, en considérant seulement les variables de $V_\Delta$

**Require:** $K_\Delta \subseteq K_\Sigma$
**Require:** un *buffer* est créé
  $buffer \leftarrow \emptyset$
  $\mathbb{B} \leftarrow \emptyset$ {une liste vide de blocs, an empty list of blocks}
  $buffer.push(Knode(\emptyset, K_\Delta))$
  **while** $buffer \neq \emptyset$ **do**
    $Knode \leftarrow buffer.pop()$
    $K \leftarrow K_\Delta \backslash Knode^-$
    $V \leftarrow$ findStumpLinkingVariables$(K_\Sigma, K, V_\Delta)$
    **if** $V \neq \emptyset$ **then**
      $\mathbb{B} \leftarrow (\mathbb{B}, block(Knode^+, V))$
      $K_\Delta \leftarrow Knode^-$
      $buffer \leftarrow \emptyset$
      $buffer.push(Knode(\emptyset, K_\Delta))$
    **else**
      $K^+ \leftarrow Knode^+$
      **for all** $k \in Knode^+$ **do**
        $K^- \leftarrow Knode^- \cup \{k\}$
        $K^+ \leftarrow K^+ \backslash \{k\}$
        **if** $K^- \neq K_\Delta$ **then**
          $buffer.push(Knode(K^-, K^+))$
        **end if**
      **end for**
    **end if**
  **end while**
  **return** $\mathbb{B}$

**Algorithm 7** findLinkingStumpVariables($K_\Sigma, K_\Delta, V_\Delta$) : Un ensemble de variables souches liant $K_\Delta$ dans $K_\Sigma$ en considérant seulement les variables $V_\Delta$

**Require:** $K_\Delta \subseteq K_\Sigma$
  $V_{stump} \leftarrow$findStumpVariables$(K_\Sigma, K_\Delta, V_\Delta)$
  **for** $V \in$ combinaisons de $card(K_\Delta) - 1$ variables de $V_{stump}$ **do**
    **if** isInterconnected($K_\Delta$,$V$) **then**
      **return** $V_{linkingStump}$
    **end if**
  **end for**
  **return** $\emptyset$

**Algorithm 8** findStumpVariables($K_\Sigma, K_\Delta, V_\Delta$) : Un ensemble de variables souches pour $K_\Delta$ dans $K_\Sigma$, en considérant seulement les variables $V_\Delta$

**Require:** $K_\Delta \subseteq K_\Sigma$
  $V_{stump} \leftarrow \emptyset$
  $V_{nonstump} \leftarrow \emptyset$
  **for all** $v \in V_\Delta$ **do**
    **if** $cons(v) \subset K_\Delta$ **then**
      $V_{stump} \leftarrow V_{stump} \cup \{v\}$
    **else**
      $V_{nonstump} \leftarrow V_{nonstump} \cup \{v\}$
    **end if**
  **end for**
  **return** $V_{stump}$

**Algorithm 9** isInterconnected($K_\Delta, V$) : Vrai si les contraintes $K_\Delta$ sont interconnectées par $V$

**Require:** $K_\Delta \subseteq K_\Sigma$
**Require:** Un *buffer* vide est créé
  **if** $V \subseteq var(K_\Delta) \wedge card(V) = card(K_\Delta) - 1$ **then**
    $buffer.push(Vnode(\emptyset, V))$
    **while** $buffer \neq \emptyset$ **do**
      $Vnode \leftarrow buffer.pop()$
      $V^+ \leftarrow Vnode^+$
      **for all** $v \in Vnode^+$ **do**
        $V^- \leftarrow Vnode^- \cup \{v\}$
        $K^+ \leftarrow cons(V^-)$
        **if** $card(K^+) = card(V^-)$ **then**
          **return** false
        **else**
          $V^+ \leftarrow V^+ \backslash \{v\}$
          **if** $V^+ \neq \emptyset$ **then**
            $buffer.push(Knode(V^-, V^+))$
          **end if**
        **end if**
      **end for**
    **end while**
    **return** true
  **else**
    **return** false
  **end if**

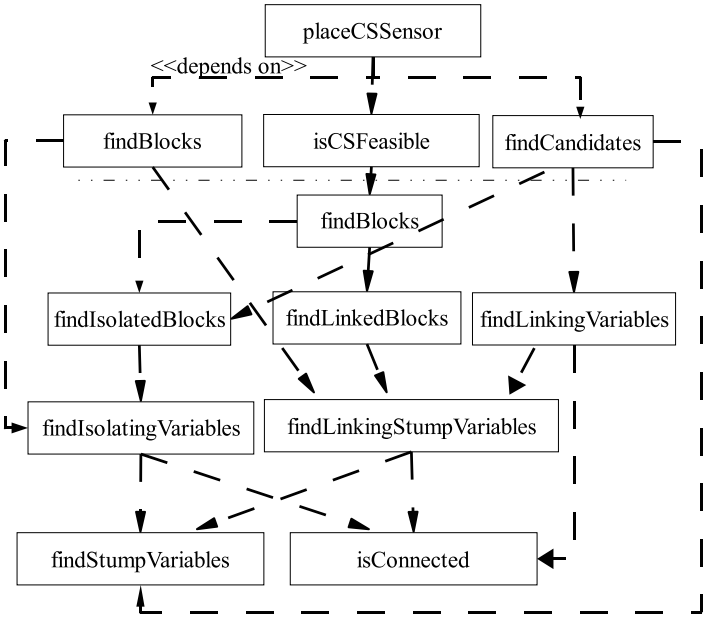


FIG. 6.7 – Schéma de dépendance de la méthode du placement de capteurs pour les spécifications complètes

contrainte terminale $k$. Une telle contrainte $k$ sera notée $k_{sensor}(v)$. Par extension, les contraintes terminales modélisant les capteurs mesurant $V$ seront notées : $K_{sensor}(V)$.

### 6.6.1 Une méthode dédiée aux spécifications complètes

Une méthode de placement de capteurs permettant de résoudre ces spécifications est proposée dans cette section. Elle peut être décomposée en deux étapes : détermination des variables candidates pour le placement de capteurs en utilisant le théorème 6.4.5 et réduction des variables candidates afin de trouver le placement de capteurs optimal par rapport au coût qui satisfait les spécifications complètes de diagnosticabilité en utilisant un algorithme d'optimisation combinatoire. La figure 6.7 présente le schéma de dépendance de la méthode.

La méthode findCandidates() (algorithme 10, page 149) est basée sur le théorème 6.4.5. Elle prend en compte les spécifications pour déterminer un ensemble de variables à mesurer. Si ces variables sont mesurées, les spécifications complètes seront satisfaites. La méthode findCandidates() commence par l'élimination des contraintes qui doivent être non détectables.

Cet algorithme dépend de la méthode findLinkingVariables(), qui est donnée par l'algorithme 11 (page 149). Cet algorithme utilise les résultats trouvés par l'algorithme 7 (page 146) pour trouver un sous-ensemble de variables liant un sous-ensemble de

**Algorithm 10** findCandidates($K_\Sigma, K^{spec}_{nondet}, \mathbb{K}^{spec}_{nondis}, K^{spec}_{diag}$) : Un ensemble de variables à mesurer pour satisfaire les spécifications complètes, $\emptyset$ si pas de solution

**Require:** Spécifications sont consistantes dans $K_\Sigma$
  $K_\Sigma \leftarrow K_\Sigma \backslash K^{spec}_{nondet}$
  $V_{nomes} \leftarrow$ findStumpVariables($K_\Sigma, K_{nodet}, var(K_\Sigma)$)
  $V_\Sigma \leftarrow var(K_\Sigma) \setminus V_{nomes}$
  **for all** $K \in \mathbb{K}^{spec}_{nondis}$ **do**
    $V \leftarrow findLinkingVariables(K_\Sigma, K, V_\Sigma)$
    **if** $V \neq \emptyset$ **then**
      $V_{nomes} \leftarrow V_{nomes} \cup V$
    **else**
      **return** $\emptyset$
    **end if**
  **end for**
  **return** $var(K_\Sigma) \backslash V_{nomes}$

contraintes $K_\Delta$, en considérant seulement les variables $V_\Delta$. Dans cet algorithme, le coût des variables est considéré.

**Algorithm 11** findLinkingVariables($K_\Sigma, K_\Delta, V_\Delta$, $cost(V_\Delta)$) : Un ensemble de variables liant $K_\Delta$ dans $K_\Sigma$, inclus dans $V_\Delta$

  $V_{linkedStump} \leftarrow$ findStumpLinkingVariables($K_\Sigma, K_\Delta, V_\Delta$)
  $V_{sorted} \leftarrow$ sortVariables($V_{linkedStump}$, $cost(V_{linkedStump})$)
  **for** $V \in$ combinaisons de $card(K_\Delta) - 1$ variables de la liste assortie (sorted list) $V_{sorted}$ **do**
    **if** isInterconnected($K_\Delta$,$V$) **then**
      **return** $V$
    **end if**
  **end for**
  **return** $\emptyset$

Cet algorithme dépend de la méthode sortVariables(), qui donne une liste de variables ordonnées selon les coûts de mesure décroissants.

Un sous-ensemble de variables candidates peut également mener à la satisfaction des spécifications. Un algorithme d'optimisation combinatoire est utilisé pour choisir les variables candidates les plus intéressantes à mesurer afin de trouver le placement de capteurs optimal. Avant de définir cet algorithme d'optimisation, il est nécessaire de pouvoir vérifier si les spécifications complètes sont satisfaites pour un sous-ensemble donné de variables candidates. La méthode isCSFeasible() (algorithme 12, page 150) réalise ceci.

**Algorithm 12** isCSFeasible($K_\Sigma, V_{measured}, \mathbb{K}^{spec}_{nondis}, K^{spec}_{diag}$) : Vrai si le placement de capteurs satisfait les spécifications

**Require:** Spécifications consistantes dans $K_\Sigma$

```
K_global ← K_Σ ∪ sensor(V_measured)
(B_nondet, B_nondis, B_diag)=findBlocks(K_Σ, var(K_Σ) \ V_measured)
if B_diag.cons ≠ K^spec_diag then
  return false
else if merge(B_nondet).cons ≠ ∅ then
  return false
else
  for all K^spec ∈ K^spec_nondis do
    found ← false
    for all B ∈ B_nondis do
      if B.cons = K^spec then
        found ← true
      end if
    end for
    if found = false then
      return false
    end if
  end for
end if
return true
```

Le critère d'optimalité pour un placement de capteurs faisable défini par $V_{measured}$ est donné par : $cost(V_{measured})$. L'algorithme d'optimisation combinatoire est mis en application dans la méthode placeCSSensor() (algorithme 13, page 151) en utilisant un simple buffer (First In First out).

**Algorithm 13** placeCSSensor($K_\Sigma, K^{spec}_{nondet}, \mathbb{K}^{spec}_{nondis}, K^{spec}_{diag}$) : Un ensemble de variables à mesurer

**Require:** Spécifications consistantes in $K_\Sigma$
**Require:** $cost()$ est défini pour chaque variable dans $V_\Sigma$
  $criteria \leftarrow cost(var(K_\Sigma))$
  $V_{candidat} \leftarrow$ findCandidates($K_\Sigma, K^{spec}_{nondet}, \mathbb{K}^{spec}_{nondis}, K^{spec}_{diag}, var(K_\Sigma)$
  $V_{measured} \leftarrow V_{candidat}$
  $buffer \leftarrow \emptyset$
  $buffer.push(Vnode(\emptyset, V_{candidat}))$
  **while** *buffer* est non vide **do**
    $Vnode \leftarrow buffer.pop()$
    $V_{remaining} \leftarrow Vnode^+$
    **for all** $v \in Vnode^+$ **do**
      $V_{selected} \leftarrow Vnode^- \cup \{v\}$
      **if** $cost(V_{selected}) < criteria$ **then**
        **if** isFeasible($K_\Sigma, V_{selected}, \mathbb{K}^{spec}_{nondis}, K^{spec}_{diag}$) **then**
          $criteria \leftarrow cost(V_{selected})$
          $V_{measured} \leftarrow V_{selected}$
        **else**
          $V_{remaining} \leftarrow V_{remaining} \backslash \{v\}$
          $buffer.push(Vnode(V_{selected}, V_{remaining}))$
        **end if**
      **end if**
    **end for**
  **end while**
  **return** $V_{measured}$

### 6.6.2 Une méthode pour les spécifications partielles

La section 6.6.1 a fourni des outils pour calculer un placement de capteurs satisfaisant des spécifications complètes. Néanmoins, pratiquement, un concepteur définirait plutôt d'une part, les contraintes qui doivent être diagnosticables et d'autre part, les contraintes qui doivent être au moins détectables. Une méthode pour résoudre à ces spécifications partielles est proposée dans cette section. Elle se fonde sur l'algorithme 3 pour l'extraction de blocs et sur un algorithme d'optimisation combinatoire, qui mène à une solution

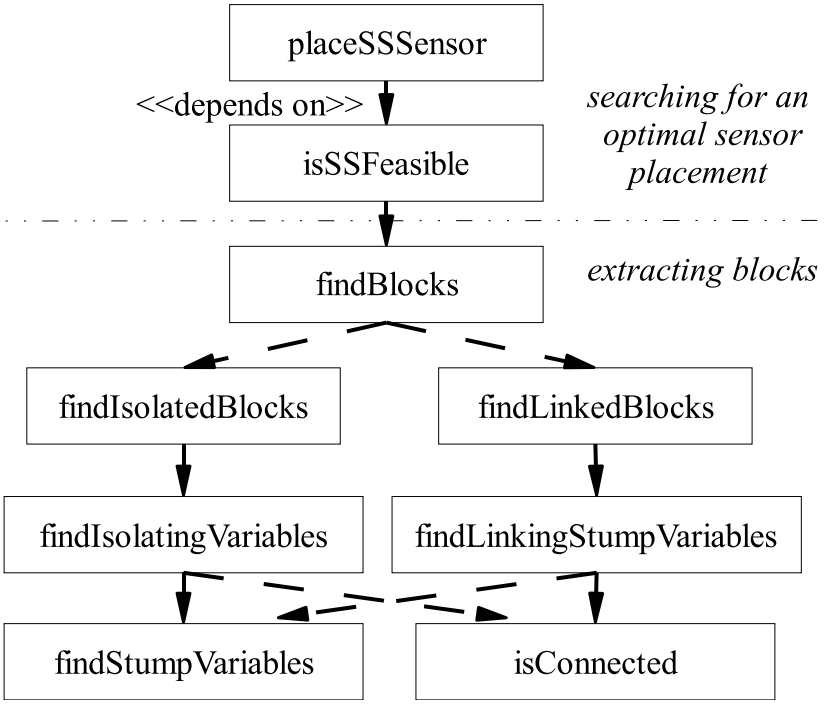


FIG. 6.8 – Schéma de dépendance de la méthode de placement de capteurs pour les spécifications partielles

optimale, en prenant en compte le coût des capteurs et satisfaisant les spécifications partielles de diagnosticabilité. La figure 6.8 représente le schéma de dépendance entre les algorithmes.

Avant de définir l'algorithme d'optimisation, introduisons la méthode isSSFeasible() (algorithme 14, page 153) qui vérifie si un placement de capteurs satisfait les spécifications partielles qui sont exprimées par : un ensemble de contraintes $K_{diag}^{spec}$ qui doivent être diagnosticables et un ensemble de contraintes $K_{det}^{spec}$ qui doivent être détectables mais pas nécessairement discriminables (les contraintes restantes peuvent être non détectables) (voir section 6.2).

Le critère d'optimalité pour un placement de capteurs faisable défini par $V_{measured}$ est donné par : $cost(V_{measured})$. L'algorithme d'optimisation combinatoire est mis en application dans la méthode placeSSSensor() (algorithme 15, page 154) en utilisant un simple buffer "First In First Out".

**Algorithm 14** isSSFeasible($K_\Sigma, V_{measured}, K^{spec}_{diag}, K^{spec}_{det}$) : Vrai si le placement de capteurs satisfait les spécifications

**Require:** Spécifications sont consistantes dans $K_\Sigma$

$K_{global} \leftarrow K_\Sigma \cup sensor(V_{measured})$
$(\mathbb{B}_{nondet}, \mathbb{B}_{nondis}, B_{diag})$=findBlocks($K_\Sigma, var(K_\Sigma) \setminus V_{measured}$)
$K_{diag} = B_{diag}.cons$
$K_{nondet} = merge(\mathbb{B}_{nondet}).cons$
**for all** $k \in K^{spec}_{diag}$ **do**
  **if** $k \notin K_{diag}$ **then**
    **return** false
  **end if**
**end for**
**for all** $k \in K^{spec}_{det}$ **do**
  **if** $k \in K_{nondet}$ **then**
    **return** false
  **end if**
**end for**
**return** true

### 6.6.3 Exemples

#### A Compteur digital

Reprenons l'exemple du compteur digital présenté dans la section 3.1.1. La matrice structurelle correspondant est donnée par le tableau 6.1

Tab. 6.1 – La matrice structurelle du compteur

| | $x_0$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 0 | 0 |
| $k_2$ | 0 | 1 | 1 | 0 | 0 |
| $k_3$ | 0 | 0 | 1 | 1 | 0 |
| $k_4$ | 0 | 0 | 0 | 1 | 1 |

.

Supposons que les coûts de mesure sont :

$$cost(x_0) = cost(x_1) = cost(x_2) = cost(x_3) = cost(x_4) = 1$$

**Algorithm 15** placeSSSensor($K_\Sigma, K_{diag}^{spec}, K_{det}^{spec}$) : Un ensemble de variables à mesurer

**Require:** Specifications sont consistantes dans $K_\Sigma$
**Require:** $cost()$ est défini pour chaque variable dans $V_\Sigma$
  $criteria \leftarrow cost(var(K_\Sigma))$
  $V_{measured} \leftarrow var(K_\Sigma)$
  $buffer \leftarrow \emptyset$
  $V_{candidat} \leftarrow var(K_\Sigma)$
  $buffer.push(Vnode(\emptyset, V_{candidat}))$
  **while** *buffer* est non vide **do**
    $Vnode \leftarrow buffer.pop()$
    $V_{remaining} \leftarrow Vnode^+$
    **for all** $v \in Vnode^+$ **do**
      $V_{selected} \leftarrow Vnode^- \cup \{v\}$
      **if** $cost(V_{selected}) < criteria$ **then**
        **if** isFeasible($K_\Sigma, V_{selected}, K_{diag}^{spec}, K_{det}^{spec}$) **then**
          $criteria \leftarrow cost(V_{selected})$
          $V_{measured} \leftarrow V_{selected}$
        **else**
          $V_{remaining} \leftarrow V_{remaining} \backslash \{v\}$
          $buffer.push(Vnode(V_{selected}, V_{remaining}))$
        **end if**
      **end if**
    **end for**
  **end while**
  **return** $V_{measured}$

TAB. 6.2 – Les relations analytiques pour les spécifications complètes

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ |
|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| $SST_2$ | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| $SST_3$ | 0 | 1 | 1 | 1 | 1 | 0 | 1 |

Considérons les spécifications complètes suivantes :

$$\mathbb{K}_{nondis} = \{\{k_2, k_3\}\}$$
$$K_{nondet} = \{k_1\}$$
$$K_{diag} = \{k_4\}$$

La contrainte $K_{nondet} = \{k_1\}$ doit être non détectable. Elle doit donc être éliminée du système. L'algorithme 4 (page 144) fournit la variable isolant la contrainte $\{\{k_1\}\}$ : $V_{isolating} = \{x_0\}$.

Afin de vérifier si les spécifications $\mathbb{K}_{nondis}$ sont satisfiables, l'algorithme 11 (page 149) est utilisé avec le sous-ensemble $K_\Delta = \{k_2, k_3\}$ en considérant $V_\Delta = var(K_\Sigma \setminus V_{isolating})$. L'algorithme 11 calcule le sous-ensemble de variables liant les contraintes $\{k_2, k_3\}$ : $V_{linking} = \{x_2\}$.

Afin de chercher les variables candidates à être mesurer pour satisfaire les spécifications complètes, l'algorithme 10 (page 149) est utilisé. L'algorithme 10 conduit aux contraintes terminales suivantes liées aux variables $\{x_1, x_3, x_4\}$ .

Afin de trouver le meilleur placement de capteurs qui satisfait les spécifications, l'algorithme 13 (page 151) est utilisé. Il donne le résultat suivant : $V_{minimal} = \{x_1, x_3, x_4\}$ avec un coût de 3 en supposant que le coût d'un capteur est fixé à 1 quelque soit la variable.

Pour valider ce résultat, la méthode proposée dans le chapitre 4 a été utilisée pour concevoir toutes les **MTPFs**. Elle a mené à la matrice de signature de défauts donnée par le tableau 6.2.

Selon ces résultats, l'ensemble de contraintes qui ne peuvent pas être discriminables sont : $\{k_2, k_3\}$, la contrainte qui ne peut pas être détectable est : $\{k_1\}$ et la contrainte diagnosticable est : $\{k_4\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui ne peuvent pas être discriminables sont : $\{c_2, c_3\}$, le composant qui ne peut pas être détectable est : $\{c_1\}$ et le composant diagnosticable est : $\{c_4\}$.

Maintenant, considérons les spécifications partielles :

$$K_{diag} = \{k_1, k_2, k_3, k_4\}$$
$$K_{det} = \{\emptyset\}$$

Afin de trouver le meilleur placement de capteurs qui satisfait les spécifications, l'algorithme 15 (page 154) est utilisé. Il conduit au résultat suivant : $V_{minimal} = \{x_0, x_1, x_2, x_3, x_4\}$ avec un coût de 5.

Afin de valider ce résultat, la méthode proposée dans le chapitre 4 a été utilisée pour concevoir toutes les **MTPFs**. Elle a mené à la matrice de signature de défauts donnée par le tableau 6.3.

TAB. 6.3 – Les relations de redondance analytique pour les spécification partielles

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| $SST_2$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| $SST_3$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| $SST_4$ | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| $SST_5$ | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| $SST_6$ | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| $SST_7$ | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| $SST_8$ | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| $SST_9$ | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| $SST_{10}$ | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |

Selon ces résultats, toutes les contraintes du système $K_{diag} = \{k_1, k_2, k_3, k_4\}$ sont diagnosticables et les spécifications sont satisfaites. Appliquons la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que tous les composants du système $\{c_1, c_2, c_3, c_4\}$ sont diagnosticables.

## B Le système de deux bacs

Reprenons l'exemple de deux bacs mentionné dans la section 2.4.1 et considérons le cas où les contraintes implicites ne sont pas prises en compte.
Supposons que toutes les variables du système sont déductibles. Ce système est représenté par la matrice structurelle 6.4.

TAB. 6.4 – La matrice structurelle du système de deux bacs

| | $q_{i_1}$ | $q_{o_1}$ | $q_{i_2}$ | $q_{o_2}$ | $l_1$ | $l_2$ |
|---|---|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 0 | 1 | 0 |
| $k_2$ | 0 | 1 | 0 | 0 | 1 | 0 |
| $k_3$ | 0 | 1 | 1 | 1 | 0 | 1 |
| $k_4$ | 0 | 0 | 0 | 1 | 0 | 1 |

Supposons que les coûts de mesure sont :

$$cost(q_{o_1}) = cost(q_{o_1}) = 3$$

et

$$cost(q_{i_1}) = cost(q_{i_1}) = 2$$

et

$$cost(l_1) = cost(l_2) = 4$$

Supposons que les spécifications complètes sont :

$$\mathbb{K}_{nondis} = \{\{k_3, k_4\}\}$$
$$K_{nondet} = \{k_1\}$$
$$K_{diag} = \{k_2\}$$

La contrainte $K_{nondet}$ doit être non détectable, alors elle peut être éliminée du système. L'algorithme 4 (page 144) fournit la variable isolant la contrainte $\{\{k_1\}\}$ : $V_{isolating} = \{q_{i_1}\}$.

Afin de vérifier si les spécifications $\mathbb{K}_{nondis}$ sont satisfiables, algorithme 11 (page 149) est utilisé avec le sous-ensemble $K_\Delta = \{k_3, k_4\}$ en considérant $V_\Delta = var(K_\Sigma \setminus V_{isolating})$. L'algorithme 11 calcule le sous-ensemble de variables liant les contraintes $\{k_3, k_4\}$ : $V_{linking} = \{l_2\}$.

Afin de chercher les variables candidates pour la mesure, l'algorithme 10 (page 149) est utilisé. L'algorithme 10 détermine les contraintes terminales candidates associées aux variables $\{q_{o_1}, q_{i_2}, q_{o_2}, l_1\}$.

Afin de trouver le meilleur placement de capteurs satisfaisant les spécifications, l'algorithme 13 est utilisé. Il donne le résultat suivant : $V_{minimal} = \{q_{o_1}, q_{i_2}, q_{o_2}, l_1\}$ avec un coût de 12.

TAB. 6.5 – Les relations analytiques pour les spécifications complètes

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ |
|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| $SST_2$ | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| $SST_3$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

TAB. 6.6 – Les relations analytiques pour les spécifications partielles

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ |
|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| $SST_2$ | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| $SST_3$ | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| $SST_4$ | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

Pour valider ce résultat, la méthode proposée dans le chapitre 4 a été utilisée pour concevoir toutes les **MTPFs**. Elle a conduit à la matrice de signature de défauts donnée par le tableau 6.5.
Selon ces résultats, l'ensemble de contraintes qui ne peuvent pas être discriminables sont : $\{k_3, k_4\}$. L'ensemble de contraintes qui ne peuvent pas être détectables sont : $\{k_1\}$ et les contraintes diagnosticables sont : $\{k_2\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui ne peuvent pas être discriminables sont : $\{c_3, c_4\}$, les composants qui ne peuvent pas être détectables sont : $\{c_1\}$ et les composants diagnosticables sont : $\{c_2\}$.

Maintenant, considérons les spécifications partielles :

$$K_{diag} = \{k_1, k_2\}$$

$$K_{det} = \{k_3, k_4\}$$

Afin de trouver le meilleur placement de capteurs satisfaisant les spécifications partielles, l'algorithme 15 (page 154) est utilisé. Il conduit au résultat suivant : $V_{minimal} = \{q_{i_1}, q_{i_2}, q_{o_2}, l_1\}$ avec un coût de 11.

Afin de valider ce résultat, la méthode proposée dans le chapitre 4 a été utilisée pour concevoir toutes les **MTPFs**. Elle a mené à la matrice de signature de défauts donnée par le tableau 6.6.

Selon ces résultats, les contraintes qui peuvent être détectées sont : $\{k_3, k_4\}$. Les contraintes diagnosticables sont : $\{k_1, k_2\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui peuvent être détectés sont : $\{c_3, c_4\}$ et les composants

diagnosticables sont : $\{c_1, c_2\}$.

Les résultats présentés dans ce chapitre démontrent qu'il est possible de concevoir des placements de capteurs optimaux satisfaisant des critères de diagnosticabilité sans concevoir les **MTPF**s au préalable.
L'inconvénient de cette méthode est qu'elle permet difficilement de tenir compte de la notion de la déductibilité (calculabilité).
Dans la section suivante, nous allons présenter une méthode simple de placement de capteurs permettant de concevoir le placement de capteurs pour les systèmes où la notion de déductibilité est importante.

## 6.7 Méthode de placement de capteurs prenant en compte la notion de déductibilité

Dans cette section, nous allons présenter une méthode pour le placement de capteurs satisfaisant des critères de diagnosticabilité dans les systèmes où la notion de déductibilité est très importante. Cette méthode, bien que requérant plus de calcul que la précédente, présente l'avantage de permettre la prise en compte de la déductibilité des variables par rapport aux contraintes
Cette méthode se décompose en deux étapes :

- faire l'hypothèse que toutes les variables du système sont mesurées puis calculer tous les sous-systèmes testables (**SST**s) et ensuite construire la matrice de signature des défauts.
- enlever itérativement des contraintes terminales et les sous-systèmes testables contenant ces contraintes de la matrice de signature de défauts jusqu' à ce que les spécifications complètes soient satisfaites (il n'y a pas toujours de solution), ou jusqu'à ce que les spécifications partielles restent satisfaites (il y a toujours des solutions). Le coût des mesures doit être pris en compte.

La première étape est synthétisée par l'algorithme 2 (page 89) présenté dans le chapitre 4. Cet algorithme permet de trouver tous les **SST**s du système.
La deuxième étape sera synthétisée par deux algorithmes : le premier algorithme prend en compte les spécifications complètes tandis que le deuxième prend en compte les spécifications partielles

### 6.7.1 Algorithme dédié aux spécifications complètes

Une méthode de placement de capteurs permettant de résoudre les spécifications complètes est proposée dans cette section. Cette méthode est synthétisée par l'algorithme 16 (page 161). L'algorithme 16 dépend de la procédure $models(V_{candidat})$ permettant de modéliser chaque variable de $V_{candidat}$ par une contrainte terminale et de l'algorithme 17 (page 161) permettant d'enlever les sous-systèmes testables qui comportent des contraintes appartenant à $K$. Elle dépend aussi de l'algorithme 18 (page 162) qui est nécessaire pour vérifier si les spécifications complètes sont satisfaites.

L'algorithme 18 (page 162) dépend de l'algorithme 19 (page 162) et de l'algorithme 20 (page 162) qui sont construits en se basant sur les définitions présentées dans la section 6.2.

L'algorithme 20 (page 162) dépend de l'algorithme 21 (page 163) permettant de vérifier si un ensemble de contraintes est détectable ou pas. Il dépend aussi de l'algorithme 22 (page 163) permettant de vérifier si un ensemble de contraintes est discriminable ou pas.

### 6.7.2 Algorithme pour les spécifications partielles

La section 6.7.1 a fourni une méthode pour calculer un placement de capteurs satisfaisant des spécifications complètes. Néanmoins, pratiquement, un concepteur définirait plutôt d'une part, les contraintes qui doivent être diagnosticables et d'autre part, les contraintes qui doivent être au moins détectables. Une méthode pour résoudre ces spécifications partielles est proposée dans cette section.

Cette méthode est synthétisée par l'algorithme 23 (page 164). L'algorithme 23 dépend de la procédure $models(V_{candidat})$ permettant de modéliser chaque variable de $V_{candidat}$ par une contrainte terminale et de l'algorithme 17 (page 161) permettant d'enlever les sous-systèmes testables qui comportent des contraintes appartenant à $K$. Il dépend de l'algorithme 24 (page 165) qui est nécessaire pour vérifier si les spécifications sont toujours satisfaites ou pas. L'algorithme 24 dépend de l'algorithme 21 (page 163) permettant de vérifier si un ensemble de contraintes est détectable ou non et de l'algorithme 20 (page 162) pemettant de vérifier si un ensemble de contraintes est diagnosticable ou non.

Reprenons l'exemple du compteur digital proposé dans la section précédente et supposons que nous voulions satisfaire les spécifications complètes suivantes :

$$\mathbb{K}_{nondis} = \{\{k_2, k_3\}\}$$

$$K_{nondet} = \{k_1\}$$

$$K_{diag} = \{k_4\}$$

**Algorithm 16** placeCSSensorDeductiblity($K_\Sigma, K^{spec}_{nondet}, \mathbb{K}^{spec}_{nondis}, K^{spec}_{diag}, \mathbb{K}_{SST}$) : Un ensemble de variables à mesurer pour satisfaire les spécifications complètes

**Require:** $\mathbb{K}_{SST}$ est l'ensemble de tous les sous-systèmes testables d'un système en supposant que les contraintes $K_{nondet}$ sont enlevées et que toutes les variables restantes sont mesurées
**Require:** Spécifications consistantes dans $K_\Sigma$
**Require:** $cost()$ est défini pour chaque variable dans $V_\Sigma$
$K_{final} \leftarrow$ l'ensemble des contraintes terminales modélisant l'ensemble minimal de capteurs qui doit être ajouté
$V_{candidat} \leftarrow var(K_\Sigma)$
$criteria \leftarrow \emptyset$
$K_{added} \leftarrow models(V_{candidat})$
$buffer \leftarrow \emptyset$
$buffer.push(Knode(\emptyset, K_{added}))$
**while** $buffer$ est non vide **do**
  $Knode \leftarrow buffer.pop()$
  $K_{remaining} \leftarrow Knode^+$
  **for all** $k \in Knode^+$ **do**
    $K_{selected} \leftarrow Knode^- \cup \{k\}$
    **if** $cost(var(K_{selected})) > criteria$ **then**
      $remove(K_{selected}, \mathbb{K}_{SST})$
      **if** isFeasibles($\mathbb{K}^{spec}_{nondis}, K^{spec}_{diag}, \mathbb{K}_{SST}$) **then**
        $criteria \leftarrow cost(var(K_{selected}))$
        $K_{final} \leftarrow K_{Sigma} \setminus K_{selected}$
      **else**
        $K_{remaining} \leftarrow K_{remaining} \setminus \{k\}$
        $buffer.push(K_{selected}, K_{remaining}))$
      **end if**
    **end if**
  **end for**
**end while**
**return** $K_{final}$

**Algorithm 17** $remove(K_{selected}, \mathbb{K}_{SST})$ : cette procédure enlève les sous-systèmes testables de $\mathbb{K}_{SST}$ qui comportent des contraintes appartenant à $K_{selected}$

**for all** $K \in \mathbb{K}_{SST}$ **do**
  **if** $K \cap K_{selected} \neq \emptyset$ **then**
    $\mathbb{K}_{SST} \leftarrow \mathbb{K}_{SST} \setminus K$
  **end if**
**end for**

**Algorithm 18** isCSFeasibleDeductibility ($K_\Sigma, K^{spec}_{nondet}, \mathbb{K}^{spec}_{nondis}, K^{spec}_{diag}, \mathbb{K}_{SST}$) : Vrai si les spécifications sont satisfaites

**Require:** Spécifications consistantes dans $K_\Sigma$

```
for all K ∈ 𝕂_nondis do
  if isNoDiscriminable(K, 𝕂_SST) = false then
    return false
  end if
end for
if isDiagnosable(K^spec_diag, 𝕂_SST) = false then
  return false
end if
return true
```

**Algorithm 19** isNoDiscriminable($K_{nondis}, \mathbb{K}_{SST}$) : Vrai si l'ensemble $K^{spec}_{nondis}$ est non discriminable

```
for (i = 0, i < card(K_nondis), i++) do
  for (j = i + 1, j ≤ card(K_nondis), j++) do
    for all K ∈ 𝕂_test do
      if K_nondis(i) ∈ K ∧ K_nondis(j) ∉ K then
        return false
      end if
    end for
  end for
end for
return true
```

**Algorithm 20** isDiagnosable($K^{spec}_{diag}, \mathbb{K}_{SST}$) : Vrai si $K^{spec}_{diag}$ est diagnosticable

```
if isDetectable(K^spec_diag, 𝕂_SST) then
  if isDiscriminable(K^spec_diag, 𝕂_SST) then
    return true
  else
    return false
  end if
else
  return false
end if
```

**Algorithm 21** isDetectable($K^{spec}_{nondet}, \mathbb{K}_{SST}$) : Vrai si $K^{spec}_{det}$ est détecable

```
for all k ∈ K^spec_det do
  if k ∉ ⋃_{K_i ∈ 𝕂_SST / i=0,1...} (K_i) then
    return false
  end if
end for
return true
```

**Algorithm 22** isDiscriminable($K_{dis}, SSTs$) : Vrai si un ensemble $K^{spec}_{dis}$ est discriminable

```
for (i = 0; i < card(K_dis), i++) do
  for (j = i + 1; j ≤ card(K_dis), j++) do
    for all K ∈ 𝕂_SST do
      if K_dis(i) ∈ K ∧ K_dis(i) ∉ K then
        return true
      end if
    end for
  end for
end for
return false
```

Supposons que toutes les variables aient le même coût :

$$cost(x_0) = cost(x_1) = cost(x_2) = cost(x_3) = cost(x_4) = 1$$

La contrainte $k_1$ doit être non détectable. Elle doit donc être enlevée du système.
La première étape consiste à mesurer toutes les variables du système restant après avoir enlevé la contrainte $k_1$. Les contraintes terminales ajoutées sont :

$$\begin{aligned} k_5 &: x_1 = \tilde{x_1} \\ k_6 &: x_2 = \tilde{x_2} \\ k_7 &: x_3 = \tilde{x_3} \\ k_8 &: x_4 = \tilde{x_4} \end{aligned} \tag{6.7.1}$$

En appliquant l'algorithme de conception des **MTPFs** présenté dans le chapitre 4, nous obtenons les sous-systèmes testables représentés par la matrice de signature 6.7.

Afin de chercher les variables candidates pour la mesure, l'algorithme 16 (page 161) est utilisé. L'algorithme 16 trouve les contraintes $\{k_5, k_7, k_8\}$ associées aux variables :

**Algorithm 23** placeSSSensorDeductibility $(K_\Sigma, K_{diag}^{spec}, K_{det}^{spec}, \mathbb{K}_{SST})$ : Un ensemble de contraintes terminales à mesurer pour satisfaire des spécifications partielles

**Require:** Spécifications consistantes dans $K_\Sigma$
**Require:** $cost()$ est défini pour chaque variable dans $V_\Sigma$
  $K_{final} \leftarrow$ l'ensemble des contraintes terminales modélisant l'ensemble minimal de capteurs qui doit être ajouté
  $V_{candidat} \leftarrow var(K_\Sigma)$
  $criteria \leftarrow \emptyset$
  $K_{added} \leftarrow models(V_{candidat})$
  $buffer \leftarrow \emptyset$
  $buffer.push(Knode(\emptyset, K_{candidat}))$
  **while** *buffer* est non vide **do**
    $Knode \leftarrow buffer.pop()$
    $K_{remaining} \leftarrow Knode^+$
    **for all** $k \in Knode^+$ **do**
      $K_{selected} \leftarrow Knode^- \cup \{k\}$
      **if** $cost(var(K_{selected})) > criteria$ **then**
        $remove(K_{selected}, \mathbb{K}_{SST})$
        **if** isFeasibles$(K_{diag}^{spec}, K_{det}^{spec}, \mathbb{K}_{SST})$ **then**
          $criteria \leftarrow cost(var(K_{selected}))$
          $K_{final} \leftarrow K_{Sigma} \setminus K_{selected}$
        **else**
          $K_{remaining} \leftarrow K_{remaining} \setminus \{k\}$
          $buffer.push(Knode(K_{selected}, K_{remaining}))$
        **end if**
      **end if**
    **end for**
  **end while**
  **return** $K_{final}$

**Algorithm 24** isSSFeasibleDeductibility($K_{diag}^{spec}, K_{det}^{spec}, \mathbb{K}_{SST}$) : Vrai si les spécifications sont satisfaites

**Require:** Spécifications sont consistantes dans $K_\Sigma$

```
if isDiagnosable(K_diag^spec, K_SST) then
  if isDetectable(K_det^spec, K_SST then
    return true
  else
    return false
  end if
else
  return false
end if
```

TAB. 6.7 – Les relations de redondance analytique pour les spécifications complètes en enlevant la contrainte $k_1$ du système

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ |
|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| $SST_2$ | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| $SST_3$ | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| $SST_4$ | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| $SST_5$ | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| $SST_6$ | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 |

$\{x_1, x_3, x_4\}$ avec un coût de 3. La matrice de signature de défauts résultant est représentée par le tableau 6.8.

Selon ces résultats, l'ensemble de contraintes qui ne peuvent pas être discriminables sont : $\{k_2, k_3\}$, l'ensemble de contraintes qui ne peuvent pas être détectables sont : $\{k_1\}$ et les contraintes diagnosticables sont : $\{k_4\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui ne peuvent pas être discriminables sont : $\{c_2, c_3\}$, les composants qui ne peuvent pas être détectables sont : $\{c_1\}$ et les composants diagnosticables sont : $\{c_4\}$.

Maintenant, considérons les spécifications partielles :

$$K_{diag} = \{k_1, k_2, k_3, k_4\}$$

$$K_{det} = \{\emptyset\}$$

En appliquant l'algorithme de conception des **MTPFs** présenté dans le chapitre 4, nous

TAB. 6.8 – Les sous-systèmes testables pour les spécifications complètes

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_7$ | $k_8$ |
|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| $SST_5$ | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| $SST_6$ | 0 | 1 | 1 | 0 | 1 | 1 | 0 |

obtenons les sous-systèmes testables représentés par la matrice de signature 6.3.

Afin de chercher les variables candidates pour la mesure, l'algorithme 23 est utilisé. L'algorithme 23 (page 164) trouve les contraintes $\{k_5, k_6, k_7, k_8, k_9\}$ associées aux variables : $\{v_0, v_1, v_2, v_3, v_4\}$ avec un coût de 5. La matrice de signature de défauts résultant est représentée par le tableau 6.3.

Selon ces résultats, toutes les contraintes du système $K_{diag} = \{k_1, k_2, k_3, k_4\}$ sont diagnosticables et les spécifications sont satisfaites. Appliquons la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$. Il est clair que tous les composants du système $\{c_1, c_2, c_3, c_4\}$ sont diagnosticables.
Nous remarquons que nous avons obtenu les mêmes résultats qu'avec la méthode présentée dans la section précédente.

Le problème de déductibilité (causalité) peut être pris en compte avec cette méthode de placement de capteurs. Reprenons l'exemple de deux bacs mentionné dans la section 2.4.1 et considérons le cas où les contraintes implicites ne sont pas prises en compte. En prenant en compte la notion de déductibilité, ce système est représenté par la matrice structurelle 6.9.

TAB. 6.9 – La matrice structurelle du système de deux bacs

| | $q_{i_1}$ | $q_{o_1}$ | $q_{i_2}$ | $q_{o_2}$ | $l_1$ | $l_2$ |
|---|---|---|---|---|---|---|
| $k_1$ | -1 | -1 | 0 | 0 | 1 | 0 |
| $k_2$ | 0 | 1 | 0 | 0 | 1 | 0 |
| $k_3$ | 0 | -1 | -1 | -1 | 0 | 1 |
| $k_4$ | 0 | 0 | 0 | 1 | 0 | 1 |

Supposons que les coûts des mesures sont :

$$cost(q_{o_1}) = cost(q_{o_2}) = 3$$

et

$$cost(q_{i_1}) = cost(q_{i_2}) = 2$$

et

$$cost(l_1) = cost(l_2) = 1$$

Supposons que les spécifications complètes sont :

$$\begin{aligned} \mathbb{K}_{nondis} &= \{\{k_3, k_4\}\} \\ K_{nondet} &= \{k_1\} \\ K_{diag} &= \{k_2\} \end{aligned}$$

La contrainte $k_1$ doit être non détectable, alors elle doit être enlevée du système.
La première étape consiste à mesurer toutes les variables du système restant après avoir enlevé la contrainte $k_1$. Les contraintes terminales ajoutées sont :

$$\begin{aligned} &k_5 : q_{o_1} = \tilde{q_{o_1}} \\ &k_6 : q_{i_2} = \tilde{q_{i_2}} \\ &k_7 : q_{o_2} = \tilde{q_{o_2}} \\ &k_8 : l_1 = \tilde{l_1} \\ &k_9 : l_2 = \tilde{l_2} \end{aligned} \tag{6.7.2}$$

En appliquantla méthode de conception des **MTPFs** présentée dans le chapitre 4, nous obtenons les sous-systèmes testables représentés par la matrice de signature 6.10.

Afin de chercher les variables candidates pour la mesure, l'algorithme 16 (page 161) est utilisé. L'algorithme 16 trouve les contraintes $\{k_5, k_6, k_8, k_9\}$ associées aux variables : $\{q_{o_1}, q_{i_2}, l_1, l_2\}$ avec un coût de 7. La matrice de signature de défauts résultant est représentée par le tableau 6.11.

Selon ces résultats, l'ensemble de contraintes qui ne peuvent pas être discriminables sont : $\{k_3, k_4\}$, l'ensemble de contraintes qui ne peuvent pas être détectables sont : $\{k_1\}$ et les contraintes diagnosticables sont : $\{k_2\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui ne peuvent pas être discriminables sont : $\{c_3, c_4\}$. Les composants qui ne peuvent pas être détectables sont : $\{c_1\}$ et le composant diagnosticable est : $\{c_2\}$.

TAB. 6.10 – Sous-systèmes testables obtenus lorsque toutes les variables du système sont mesurées ($k_1$ est enlevé)

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| $SST_2$ | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| $SST_3$ | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| $SST_4$ | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| $SST_5$ | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| $SST_6$ | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| $SST_7$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| $SST_8$ | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |

TAB. 6.11 – Les sous-systèmes testables pour les spécifications complètes

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_8$ | $k_9$ |
|---|---|---|---|---|---|---|---|---|
| $SST_5$ | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| $SST_6$ | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| $SST_7$ | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |

TAB. 6.12 – Les sous-systèmes testables en mesurant toutes les variables du système

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_9$ | $k_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| $SST_2$ | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| $SST_3$ | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| $SST_4$ | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| $SST_5$ | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| $SST_6$ | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| $SST_7$ | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| $SST_8$ | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| $SST_9$ | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| $SST_{10}$ | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| $SST_{11}$ | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| $SST_{12}$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| $SST_{13}$ | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| $SST_{14}$ | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |

Maintenant, considérons les spécifications partielles :

$$K_{diag} = \{k_1, k_2\}$$

$$K_{det} = \{k_3, k_4\}$$

La première étape consiste à mesurer toutes les variables du système. Appliquons la méthode de conception des **MTPFs** présentée dans le chapitre 4, nous obtenons les sous-systèmes testables représentées par la matrice de signature 6.12.

Afin de chercher les variables candidates pour la mesure, l'algorithme 23 (page 164) est utilisé. L'algorithme 23 trouve les contraintes $\{k_5, k_7, k_8, k_9\}$ associées au variables : $\{q_{i_1}, q_{i_2}, q_{o_2}, l_1\}$ avec un coût de 8. La matrice de signature de défauts résultant est représentée par le tableau 6.13.

Selon ces résultats, les contraintes qui peuvent être détectées sont : $\{k_3, k_4\}$. Les contraintes diagnosticables sont : $\{k_1, k_2\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui peuvent être détectés sont : $\{c_3, c_4\}$ et les composants diagnosticables sont : $\{c_1, c_2\}$.

Nous remarquons que nous avons obtenu les mêmes résultats qu'avec la méthode

TAB. 6.13 – Les sous-systèmes testables pour les spécifications partielles

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_7$ | $k_8$ | $k_9$ |
|---|---|---|---|---|---|---|---|---|
| $SST_6$ | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| $SST_7$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| $SST_{14}$ | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

TAB. 6.14 – La matrice structurelle de l'exemple

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $k_1$ | -1 | 1 | 0 | 0 |
| $k_2$ | -1 | 1 | 0 | 0 |
| $k_3$ | -1 | 1 | 1 | -1 |
| $k_4$ | 0 | 0 | 1 | -1 |

présentée dans la section précédente.

La méthode de placement de capteurs prenant en compte la notion de déductibilité peut fournir des résultats différents qu'avec la méthode présentée dans la section 6.6. Prenons l'exemple présenté par la matrice structurelle 6.14. Supposons que les coûts des mesures sont :

$$cost(x_1) = 4, cost(x_2) = 1, cost(x_3) = 2, cost(x_4) = 3$$

Supposons que nous voulions satisfaire les spécifications partielles suivantes :

$$\begin{aligned} K_{diag} &= \{\{k_1, k_2\}\} \\ K_{det} &= \{k_3, k_4\} \end{aligned} \tag{6.7.3}$$

Dans un premier temps, nous appliquons la méthode de placement de capteurs ne prenant pas en compte la notion de déductibilité.
Supposons que toutes les variables soient déductibles. La matrice structurelle est donnée par le tableau 6.15.

Afin de trouver le meilleur placement de capteurs satisfaisant les spécifications partielles, l'algorithme 15 (page 154) est utilisé. Il mène au résultat suivant : $V_{minimal} = \{x_2, x_3\}$ avec un coût de 3.
Afin de valider ce résultat, la méthode proposée dans le chapitre 4 a été utilisée pour concevoir toutes les **MTPFs**. Elle a mené à la matrice de signature de défauts donnée par le tableau 6.16.

TAB. 6.15 – La matrice structurelle de l'exemple en supposant que toutes les variables sont déductibles

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 0 |
| $k_2$ | 1 | 1 | 0 | 0 |
| $k_3$ | 1 | 1 | 1 | 1 |
| $k_4$ | 0 | 0 | 1 | 1 |

TAB. 6.16 – Les sous-systèmes testables pour les spécifications partielles

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ |
|---|---|---|---|---|---|---|
| $SST_1$ | 1 | 0 | 1 | 1 | 1 | 1 |
| $SST_2$ | 1 | 1 | 0 | 0 | 1 | 0 |
| $SST_3$ | 0 | 1 | 1 | 1 | 1 | 1 |
| $SST_4$ | 1 | 1 | 1 | 1 | 0 | 1 |

Selon ces résultats, les contraintes qui peuvent être détectées sont : $\{k_3, k_4\}$. Les contraintes diagnosticables sont : $\{k_1, k_2\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui peuvent être détectés sont : $\{c_3, c_4\}$ et les composants diagnosticables sont : $\{c_1, c_2\}$.

Prenons maintenant en compte la déductibilité des variables et supposons que les capteurs trouvés ci-dessus (les capteurs mesurant les variables $\{x_2, x_3\}$) soient ajoutés au système.
Afin de valider si les spécifications partielles sont toujours satisfaites en prenant en compte la notion de déductibilité, la méthode proposée dans le chapitre 4 a été utilisée pour concevoir toutes les **MTPFs**. Nous remarquons qu'il n'y a aucun test trouvé. Par conséquent, la méthode présentée dans la section n'est pas toujours utile pour le placement de capteurs en prenant en compte la notion de déductibilité.

Dans un deuxième temps, nous appliquons la méthode prenant en compte la déductibilité des variables présentée dans cette section pour trouver le placement de capteurs satisfaisant les spécifications partielles (6.7.3).
La première étape consiste à mesurer toutes les variables du système. Appliquons la méthode de conception des **MTPFs** présentée dans le chapitre 4, nous obtenons les sous-systèmes testables représentés par la matrice de signature 6.17.

TAB. 6.17 – Les sous-systèmes testables pour les spécifications partielles de l'exemple

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ |
|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| $SST_2$ | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| $SST_3$ | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| $SST_4$ | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| $SST_5$ | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| $SST_6$ | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| $SST_7$ | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| $SST_8$ | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| $SST_9$ | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| $SST_{10}$ | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

TAB. 6.18 – Les sous-systèmes testables pour les spécifications partielles de l'exemple

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_8$ |
|---|---|---|---|---|---|---|
| $SST_2$ | 0 | 1 | 1 | 1 | 1 | 1 |
| $SST_6$ | 1 | 0 | 1 | 1 | 1 | 1 |
| $SST_8$ | 1 | 1 | 0 | 0 | 1 | 0 |

Afin de chercher les variables candidates pour la mesure, l'algorithme 23 (page 164) est utilisé. L'algorithme 23 trouve les contraintes $\{k_5, k_8\}$ associées aux variables : $\{x_1, x_4\}$ avec un coût de 7. La matrice de signature de défauts résultant est représentée par le tableau 6.18.

Selon ces résultats, les contraintes qui peuvent être détectées sont : $\{k_3, k_4\}$. Les contraintes diagnosticables sont : $\{k_1, k_2\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui peuvent être détectés sont : $\{c_3, c_4\}$ et les composants diagnosticables sont : $\{c_1, c_2\}$.

Nous remarquons que le placement de capteurs satisfaisant les spécifications partielles (6.7.3) en prenant en compte la notion de déductibilité (les variables qui doivent être mesurées sont $V_{minimal} = \{x_1, x_4\}$ est différent du placement de capteurs satisfaisant les mêmes spécifications mais sans prendre en compte la notion de déductibilité (les variables qui doivent être mesurées sont $V_{minimal} = \{x_2, x_3\}$.

Dans cette section, nous avons présenté une méthode pour le placement de capteurs, qui prend en compte la notion de déductibilité (causalité). L'inconvénient de cette méthode est qu'elle exige de concevoir les **SST**s au préalable.

## 6.8 Conclusion

De nouvelles méthodes de placement de capteurs ont été proposées dans ce chapitre. En ce qui concerne la première méthode proposée, les lemmes, théorèmes et les algorithmes ont l'avantage d'être des propriétés générales. Par conséquent, différents types de spécifications de diagnosticabilité peuvent être satisfaits. Avec cette méthode, des placements de capteurs satisfaisant des spécifications complètes et partielles ont été présentés. Les spécifications complètes traitent les éléments qui doivent être diagnosticables, discriminables et non détectables tandis que les spécifications partielles traitent seulement les éléments qui doivent être diagnosticables et ceux qui doivent être au moins détectables. Nous avons montré que ces méthodes proposées s'appliquent à tout système modélisable structurellement. En effet, les contraintes sont décrites seulement par les variables apparaissant dans ces contraintes. Grâce à cette méthode, le placement optimal de capteurs satisfaisant les spécifications de diagnosticabilité devient possible sans concevoir les **SST**s au préalable. C'est un dispositif très important puisqu'il n'est plus nécessaire de concevoir les **SST**S possibles. L'inconvénient de cette méthode est qu'elle ne prend pas en compte la notion de déductibilité. Une seconde méthode traitant le cas où la notion de déductibilité est prise en compte a été aussi proposée. Cette méthode exige de concevoir les **RRA**s au préalable.

# Quatrième partie

# Applications

# Chapitre 7

# Applications

Dans ce chapitre, nous présentons un logiciel pour le diagnostic appelé **DXLAB** pour lequel nous avons contribué au développement. Afin de synthétiser nos résultats, cet outil a été utilisé pour concevoir des tests et pour valider les résultats de placement de capteurs pour deux systèmes différents : l'un tirer du monde continu (amplificateur électrique), l'autre d'un monde logique. Le circuit électrique évoque le cas de composants qui peuvent être modélisés par plusieurs contraintes. Pour finir une application industrielle réelle est présentée : il s'agit du placement de capteurs dans un système photovoltaïque. Ce travail applicatif s'inscrit dans le cadre du projet DLDPV supporté par l'Agence Nationale de la Recherche (ANR) et réunissant le CEA (institut national de l'Énergie Solaire), Transénergie, le G2ELAB, et notre laboratoire G-SCOP.

## 7.1 L'outil DXLAB

**DXLAB** est un outil d'aide au diagnostic qui s'appuie sur l'expérience de l'utilisateur pour obtenir une description correcte du système (qui est modélisé ici par un modèle structurel). À partir de cette description, on obtient des variables et des ressources et on peut déterminer les structures de relations entre variables, à partir desquelles **DXLAB** peut calculer les tests possibles (**MTPF**s). À partir des tests trouvés, on peut analyser la performance résultante (les critères de diagnosticabilité, de détectabilité, et de discriminabilité) et calculer des diagnostics. Cet outil a été développé en Java. Son modèle de données est présenté par la figure 7.1.

L'architecture de **DXLAB** est présenté par la figure 7.2. et l'interface graphique de **DXLAB** est donnée par la figure 7.3.

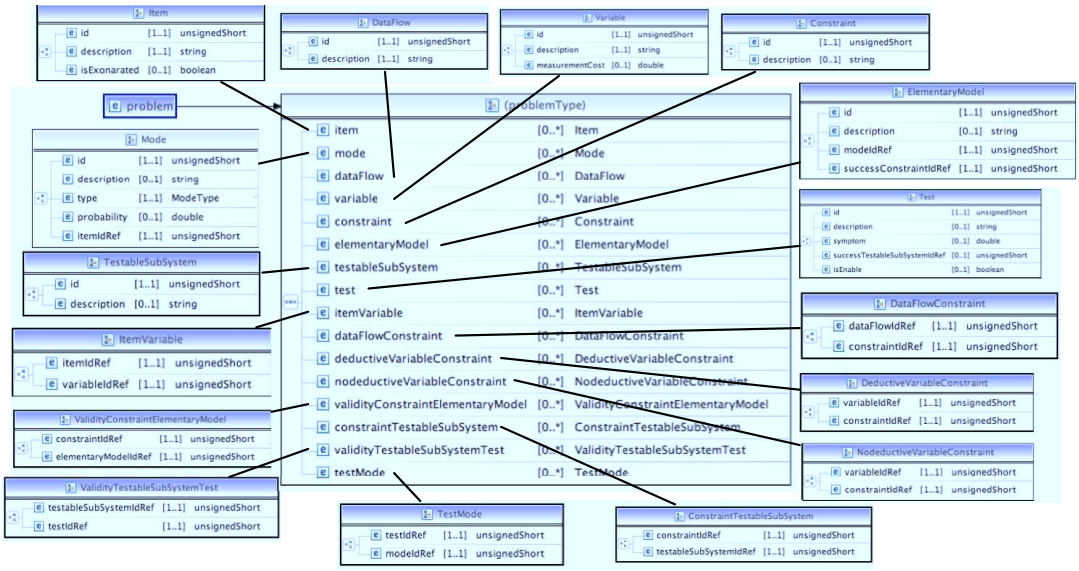


Fig. 7.1 – Le modèle de données de **DXLAB**

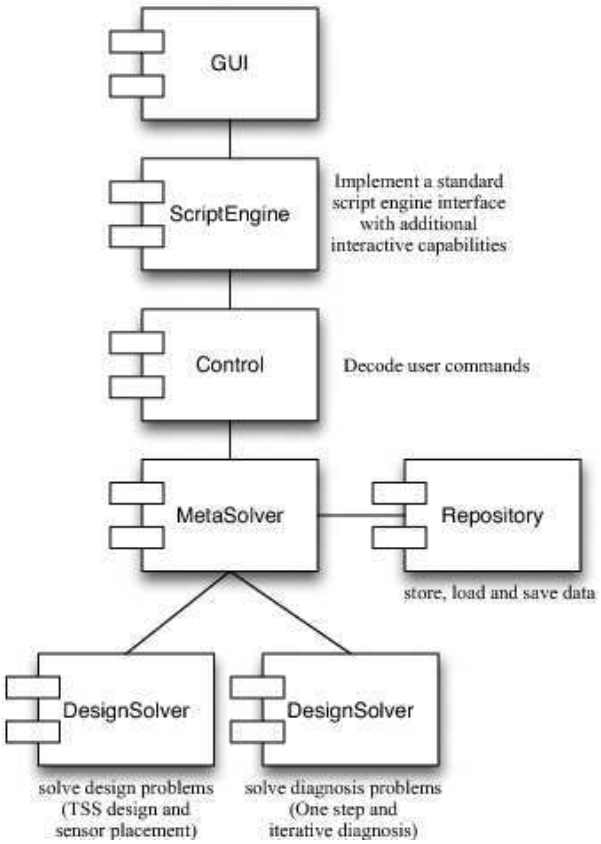


Fig. 7.2 – L'architecture de **DXLAB**

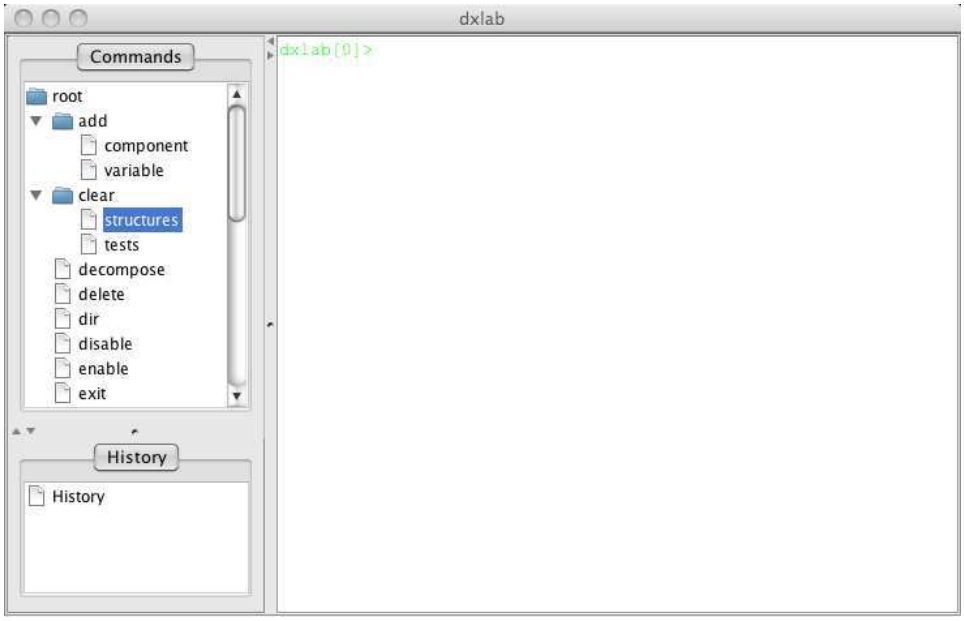


FIG. 7.3 – L'interface graphique de **DXLAB**

Afin d'illustrer l'intérêt de cet outil, appliquons-le sur un compteur digital qui se compose de 4 comptages consécutifs $C_1$, $C_2$, $C_3$ et $C_4$ et 4 affichages $A_1$, $A_2$, $A_3$ et $A_4$. Ce compteur est décrit par les contraintes suivantes :

$$\begin{aligned} C_1 : k_1 : x_1 &= x_0 + 1 \\ C_2 : k_2 : x_2 &= x_1 + 1 \\ C_3 : k_3 : x_3 &= x_2 + 1 \\ C_4 : k_4 : x_4 &= x_3 + 1 \end{aligned} \tag{7.1.1}$$

Le modèle d'observation (afficheurs) est décrit par les contraintes suivantes :

$$\begin{aligned} A_1 : k_5 : x_1 &= \tilde{x_1} \\ A_2 : k_6 : x_2 &= \tilde{x_2} \\ A_3 : k_7 : x_3 &= \tilde{x_3} \\ A_4 : k_8 : x_4 &= \tilde{x_4} \end{aligned} \tag{7.1.2}$$

Pour trouver les tests de ce système, nous commençons par entrer la description du système c'est-à-dire, entrer les structures des contraintes dans le logiciel **DXLAB**. Cette étape est décrite par les instructions suivantes :

```
Welcome into DXLAB version 0.2.2

[1] structure x0 x1 named s1 models C1
s1 = l{-},{x1,x0}l models C1
[2] structure x1 x2 named s2 models C2
s2 = l{-},{x2,x1}l models C2
[3] structure x2 x3 named s3 models C3
s3 = l{-},{x3,x2}l models C3
[4] structure x3 x4 named s4 models C4
s4 = l{-},{x4,x3}l models C4
[5] structure x1 named s5 models A1
s5 = l{-},{x1}l models A1
[6] structure x2 named s6 models A2
s6 = l{-},{x2}l models A2
[7] structure x3 named s7 models A3
s7 = l{-},{x3}l models A3
[8] structure x4 named s8 models A4
s8 = l{-},{x4}l models A4
```

Afin de trouver tous les sous-système testables (**MTPF**s), l'instruction "show tss" est utilisée :

```
[10] show tss
tss001: support=s5[A1],s2[C2],s6[A2] (weight=2)
  Formula=((s5-x1-s2)-x2-s6)
tss002: support=s4[C4],s3[C3],s8[A4],s6[A2] (weight=3)
  Formula=((s3-x3-(s4-x4-s8))-x2-s6)
tss003: support=s5[A1],s4[C4],s3[C3],s2[C2],s8[A4] (weight=4)
  Formula=((s3-x3-(s4-x4-s8))-x2-(s5-x1-s2))
tss004: support=s3[C3],s7[A3],s6[A2] (weight=2)
  Formula=((s3-x3-s7)-x2-s6)
tss005: support=s4[C4],s8[A4],s7[A3] (weight=2)
  Formula=((s4-x4-s8)-x3-s7)
tss006: support=s5[A1],s3[C3],s2[C2],s7[A3] (weight=3)
 Formula=((s3-x3-s7)-x2-(s5-x1-s2))
```

Nous remarquons que l'instruction "show tss" permet de trouver tous les sous-ensembles de composants qui peuvent être testés. Elle montre également les formules **MTPF**s (voir chapitre 4).

Afin de trouver les **MTPF**s de base et donc, indirectement, les sous-systèmes tes-

tables de base (voir chapitre chapitre 4), l'instruction "show tss minimal" suivante est utilisée :

```
[11] show tss minimal
tss001: support=s3[C3],s7[A3],s6[A2] (weight=2)
  Formula=((s3-x3-s7)-x2-s6)
tss002: support=s4[C4],s8[A4],s7[A3] (weight=2)
  Formula=((s4-x4-s8)-x3-s7)
tss003: support=s5[A1],s2[C2],s6[A2] (weight=2)
  Formula=((s2-x2-s6)-x1-s5)
```

Pour extraire les composants des sous-systèmes testables et ainsi de pouvoir analyser la diagnosticabilité du système, nous utilisons l'instruction "make tests" :

```
[12] make tests
tss001->test t001 [enabled], symptom:50.0% (dubious)
   A2,   A1,   C2
tss002->test t002 [enabled], symptom:50.0% (dubious)
   A4,   A2,   C4,   C3
tss003->test t003 [enabled], symptom:50.0% (dubious)
   A4,   A1,   C4,   C3,   C2
tss004->test t004 [enabled], symptom:50.0% (dubious)
   A3,   A2,   C3
tss005->test t005 [enabled], symptom:50.0% (dubious)
   A4,   A3,   C4
tss006->test t006 [enabled], symptom:50.0% (dubious)
   A3,   A1,   C3,   C2
```

Les tests minimaux sont obtenue grâce à l'instruction"make tests minimal" :

```
[13] make tests minimal
tss001->test t007 [enabled], symptom:50.0% (dubious)
   A3,   A2,   C3
tss002->test t008 [enabled], symptom:50.0% (dubious)
   A4,   A3,   C4
tss003->test t009 [enabled], symptom:50.0% (dubious)
   A2,   A1,   C2
```

Afin de déterminer les composants diagnosticables et les composants détectables mais non discriminables, nous utilisons l'instruction "show diagnosability" suivante :

```
[16] show diagnosability
Diagnosable components are: A2,C3,A3
Detectable but not discriminable components are:
 -A1,C2
 -A4,C4
```

L'exemple traité permet d'illustrer un langage de script dédié au diagnostic. L'outil **DXLAB** que nous avons présenté permet de chercher les sous-systèmes testables et d'évaluer les critères de diagnosticabilité conformément aux développements théoriques de notre travaux. Cet outil ne permet pas pour le moment de chercher les placements de capteurs satisfaisant des critères de diagnosticabilité car les codes développés n'ont pas encore été intégrés dans l'environnement **DXLAB**. Ceci sera fait dans une perspective proche. Cela permettra d'avoir un outil complet pour le diagnostic des défauts.

## 7.2 Exemples d'utilisation

Considérons d'autres exemples un peu plus complexes avant d'aborder une application industrielle réelle. Certaines méthodes permettant de reformuler les problèmes sous une forme standard vont être introduites.

### 7.2.1 Amplificateur

La méthode du placement de capteurs présentée dans la section 6.6 va être appliquée sur le circuit électronique (figure 4.10) présenté dans la section 4.5.2. Ce circuit est modélisé par les contraintes suivantes :

$$amplificateur : k_1 : v_{1c} = v_2$$
$$connection1a : k_2 : i_1 = i_2 + i_3$$
$$connection1b : k_3 : v_1 = v_{1a}$$

$$
\begin{aligned}
&\textit{connection}1c : k_4 : v_1 = v_{1b}\\
&\textit{connection}1d : k_5 : v_1 = v_{1c}\\
&\text{résistance1} : k_6 : v_0 - v_1 = R_1 i_1\\
&\textit{condensateur} : k_7 : C(v_{1a} - v_3) = \int_0^t i_2 dt\\
&\text{résistance2} : k_8 : v_3 - v_{4a} = R_2 i_2\\
&\text{résistance3} : k_9 : v_1 - v_{4b} = R_3 i_3\\
&\text{résistance4} : k_{10} : v_2 = R_4 i_4\\
&\textit{connection}2a : k_{11} : v_4 = v_{4a}\\
&\textit{connection}2b : k_{12} : v_4 = v_{4b}\\
&\text{générateur} : k_{13} : v_0 = \tilde{v_0}
\end{aligned}
$$

Dans cet exemple, le composant *connection*1 est modélisé par quatre contraintes ($k_2$, $k_3$, $k_4$ et $k_5$). Il peut être représenté par quatre sous-composants *connection*1*a*, *connection*1*b*, *connection*1*c*, *connection*1*d*. Le composant *connection*2 est modélisé par deux contraintes ($k_{11}$ et $k_{12}$). Il peut être représenté par deux sous-composants *connection*2*a* et *connection*2*b*. Les contraintes modélisant les composants du systèmes sont : $K_\Sigma = \{k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}, k_{11}, k_{12}, k_{13}\}$.

Dans le chapitre 6, nous avons considéré des spécifications de diagnosticabilité par rapport aux contraintes et non par rapport aux composants. Nous avons de surcroît considéré qu'un composant ne peut être modélisé que par une seule contrainte. Dans cet exemple, nous allons traiter les spécifications par rapport à des composants. Comme cet exemple contient des composants modélisés par plusieurs contraintes, certaines définitions importantes sont définies avant de chercher le placement de capteurs satisfaisant à des critères de diagnosticabilité.

**Définition 7.2.1.** *Un composant $c_i$ modélisé par plusieurs contraintes est diagnosticable si et seulement si une de ces contraintes est diagnosticable.*

**Définition 7.2.2.** *Un composant $c_i$ modélisé par plusieurs contraintes est non discrimiable d'un autre composant $c_j$ modelisé par une seule contrainte si et seulement si aucune des contraintes de $c_i$ n'est diagnosticable et s'il y a au moins une contrainte de $c_i$ qui est non discriminble de la contrainte de $c_j$. Par extension, un composant $c_i$ modélisé par plusieurs contraintes est non discrimiable d'un autre composant $c_j$ modelisé par plusieurs contrainte si et seulement si aucune des contraintes de $c_i$ ou de $c_j$ n'est diagnosticable et s'il y a au moins une contrainte de $c_i$ qui est non discriminble d'une des contrainte de $c_j$.*

**Définition 7.2.3.** *Un composant $c_i$ modélisé par plusieurs contraintes est non détectable si et seulement si toutes ces contraintes sont non détectables.*

TAB. 7.1 – La matrice structurelle du circuit électronique

| | $v_0$ | $v_1$ | $v_2$ | $v_3$ | $v_4$ | $i_1$ | $i_2$ | $i_3$ | $i_4$ | $v_{1a}$ | $v_{1b}$ | $v_{1c}$ | $v_{4a}$ | $v_{4b}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_1$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $k_2$ | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_3$ | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_4$ | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $k_5$ | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $k_6$ | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_7$ | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_8$ | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $k_9$ | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| $k_{10}$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $k_{11}$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| $k_{12}$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| $k_{13}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Dans cet exemple, nous supposons que toutes les variables sont déductibles. La matrice structurelle correspondante est donnée par le tableau 7.1. Nous commençons par entrer la description du système dans **DXLAB**. Cette étape est décrite par les instructions suivantes :

```
Welcome into DXLAB version 0.2.2

[1] structure v1c v2 named s1 models amplificateur
s1 = l{-},{v2,v1c}l models amplificateur
[2] structure i1 i2 i3 named s2 models connection1a
s2 = l{-},{i3,i2,i1}| models connection1a
[3] structure v1 v1a named s3 models connection1b
s3 = l{-},{v1a,v1}l models connection1b
[4] structure v1 v1b named s4 models connection1c
s4 = l{-},{v1b,v1}l models connection1c
[5] structure v1 v1c named s5 models connection1d
s5 = l{-},{v1c,v1}l models connection1d
[6] structure v0 v1 i1 named s6 models résistance1
s6 = l{-},{v0,i1,v1}| models résitance1
[7] structure v1a v3 i2 named s7 models condensateur
s7 = l{-},{v1a,i2,v3}l models condensateur
[8] structure v3 v4a i3 named s8 models résistance2
s8 = l{-},{i3,v4a,v3}l models résitance2
[9] structure v1 v4b i3 named s9 models résistance3
s9 = l{-},{i3,v4b,v1 }l models résitance3
[10] structure v2 i4 named s10 models résistance4
s10 = l{-},{i4,v2}l models résitance4
[11] structure v4 v4a named s11 models connection2a
s11 = l{-},{v4a,v4}l models connection2a
[12] structure v4 v4b named s12 models connection2b
s12 = l{-},{v4b,v4}l models connection2b
[13] structure v0 named s13 models générateur
s13 = l{-},{v0}l models générateur
```

Supposons que les coûts des mesures sont :

$$cost(v_0) = cost(v_1) = cost(v_2) = cost(v_3) = cost(v_4) \ldots$$
$$\ldots = cost(v_{1a}) = cost(v_{1b}) = cost(v_{1c}) = cost(v_{4a}) \ldots$$
$$\ldots = cost(v_{4b}) = 1$$

et

$$cost(i_1) = cost(i_2) = cost(i_3) = cost(i_4) = 2$$

Considérons les spécifications complètes suivantes :

$$\mathbb{C}_{nondis} = \{condensateur, \text{résistance2}, connection2\}$$
$$C_{nondet} = \{amplificateur, \text{résistance4}\}$$
$$C_{diag} = \{connection1, \text{résistance1}, \text{résistance3}, \text{générateur}\}$$

Afin d'appliquer les algorithmes du placement de capteurs proposés dans le chapitre 6, nous utilisons les contraintes modélisant les composants.
L'ensemble de composants $C_{nondet}$ doit être non détectable, alors les contraintes $K_{nondet}$ modélisant ces composants doivent donc être enleveés du système.

Maintenant nous allons vérifier si les spécifications $\mathbb{C}_{nondis}$ sont satisfiables. Nous remarquons que le composant $connection2$ est modélisé par deux contraintes $k_{11}$ et $k_{12}$. Nous devons appliquer l'algorithme 11 deux fois, une fois avec la contrainte $k_{11}$ et l'autre fois avec la contrainte $k_{12}$ et nous choisissons la meilleure solution. Dans un premier temps, l'algorithme 11 (page 149) est utilisé sur l'ensemble de contraintes $K_\Delta = \{k_7, k_8, k_{11}\}$, en considérant $v_\Delta = var(K_\Sigma)$ représentant les variables restant après avoir enlevé les contraintes $K_{nondet}$ du système. L'algorithme 11 calcule l'ensemble de variables liantes suivantes : $v = \{v_3, v_{4a}\}$. Dans un deuxième temps, l'algorithme 11 est utilisé avec $K_\Delta = \{k_7, k_8, k_{12}\}$, en considérant $v_\Delta = var(K_\Sigma)$ représentant les variables restant après avoir enlevé les contraintes $K_{nondet}$ du système. L'algorithme 11 ne trouve pas de solution. Par conséquent, l'ensemble des variables satisfaisant $\mathbb{C}_{nondis}$ est $v = \{v_3, v_{4a}\}$.

Afin de chercher les variables candidates à la mesure pour satisfaire les spécifications, l'algorithme 10 est utilisé. L'algorithme 10 (page 149) calcule les contraintes terminales suivantes contenant les variables de $\{v_0, v_2, v_4, i_1, i_2, i_3, v_{1a}, v_{1b}, v_{1c}, v_{4b}\}$.

Afin de trouver le meilleur placement de capteurs qui satisfait les spécifications, l'algorithme 13 (page 151) est utilisé en tenant compte qu'aucune des contraintes modélisant le

composant *connection*2 doit être diagnosticable (définition 7.2.2). Il donne le résultat suivant : $V_{minimal} = \{v_0, i_1, i_3, v_{1a}, v_{4b}\}$ avec un coût de 7. Les capteurs : *capteur*1, *capteur*2, *capteur*3, *capteur*4 et *capteur*5 mesurant les variables de $V_{minimal}$ sont respectivement modélisés par les contraintes terminales : $k_{14}$, $k_{15}$, $k_{16}$, $k_{17}$ et $k_{18}$.

Afin de valider ce résultat, la méthode proposée dans le chapitre 4 sera utilisée pour concevoir toutes les formules **MTPF**s. Les structures des capteurs ajoutés sont entrées dans **DXLAB** :

```
[14] structure v0 named s14 models capteur1
s14 = l{-},{v0}l models capteur1
[15] structure i1 named s15 models capteur2
s15 = l{-},{i1}l models capteur2
[16] structure i3 named s16 models capteur3
s16 = l{-},{i3}l models capteur3
[17] structure v1a named s17 models capteur4
s17 = l{-},{v1a}l models capteur4
[18] structure v4b named s18 models capteur5
s18 = l{-},{v4b}l models capteur5
```

En utilisant **DXLAB** qui utilise les résultats du chapitre 4, on obtient toutes les **MTPF**s. Pour des raisons de simplification, nous ne détaillons que les **MTPF**s de base (voir la section 4.4.3) :

```
[21] show tss minimal
tss001: support=s16[capteur3],s15[capteur2],s13[générateur],s9[résistance3],s6[résistance1],s18[capteur5]
(weight=5)
Formula=(((s6-v0-s13)-i1-s15)-v1-((s9-i3-s16)-v4b-s18))

tss002: support=s16[capteur3],s15[capteur2],s14[capteur1],s9[résistance3],s6[résistance1],s18[capteur5]
(weight=5)
Formula=(((s6-v0-s14)-i1-s15)-v1-((s9-i3-s16)-v4b-s18))

tss003: support=s17[capteur4],s16[capteur3],s3[connection1b],s9[résistance3],s18[capteur5] (weight=4)
Formula=((s3-v1a-s17)-v1-((s9-i3-s16)-v4b-s18))

tss004: support=s17[capteur4],s3[connection1b],s15[capteur2],s13[générateur],s6[résistance1] (weight=4)
Formula=((s3-v1a-s17)-v1-((s6-v0-s13)-i1-s15))

tss005: support=s17[capteur4],s3[connection1b],s15[capteur2],s14[capteur1],s6[résistance1] (weight=4)
Formula=((s3-v1a-s17)-v1-((s6-v0-s14)-i1-s15))

tss006: support=s18[capteur5],s8[résistance2],s17[capteur4],s7[condensateur],s16[capteur3],s15[capteur2],
s12[connection2b],s11[connection2a],s2[connection1a] (weight=9)
Formula=(((s7-v1a-s17)-i2-((s2-i3-s16)-i1-s15))-v3-((s11-v4a-(s8-i3-s16))-v4-(s12-v4b-s18)))

tss007: support=s14[capteur1],s13[générateur] (weight=1)
Formula=(s14-v0-s13)
```

Afin de vérifier si les spécifications complètes sont satisfaites, nous utilisons l'instruction "show diagnosability" dans **DXLAB** :

```
[23] show diagnosability
Diagnosable components are:
-capteur5,capteur3,capteur1,résistance3,résistance1,connection1b,capteur2,générateur,capteur4
Detectable but not discriminable components are:
-condensateur,résistance2,connection2b,connection2a,connection1a
```

Selon ces résultats, l'ensemble des composants du système qui ne peuvent pas être discriminables sont : $\{condensateur, \text{résistance2}, connection2\}$. L'ensemble des composants qui ne peuvent pas être détectables sont : $\{amplificateur, \text{résistance4}\}$ et les composants diagnosticables sont : $\{connection1, \text{résistance1}, \text{résistance3}, \text{générateur}\}$.
Ces résultats conforment aux attentes et ça valide notre approche.

Supposons maintenant que les spécifications soient données par :

$$\mathbb{C}_{nondis} = \{\text{résistance2}, \text{résistance3}\}$$

$$C_{nondet} = \{amplificateur, \text{résistance4}\}$$

$$C_{diag} = \{connection1, \text{résistance1}, condensateur, connection2, \text{générateur}\}$$

L'ensemble de contraintes $C_{nondet}$ doit être non détectable. Les contraintes $K_{nondet}$ modélisant ces composants doivent donc être enlevées du système.

Afin de vérifier si les spécifications $\mathbb{C}_{nondis}$ sont satisfiablees, l'algorithme 11 (page 149) est utilisé avec l'ensemble $K_\Delta = \{k_8, k_9\}$, en considérant $v_\Delta = var(K_\Sigma)$ représentant les variables restant après avoir enlevé les contraintes $K_{nondet}$ du système. Puisque l'algorithme 11 ne trouve pas d'ensemble de variables liant les deux contraintes $k_8$ et $k_9$, il n'y a pas de solution qui satisfasse ces spécifications.

Maintenant, considérons les spécifications partielles :

$$C_{diag} = \{\text{résistance1}, \text{résistance2}, \text{résistance3}, \text{résistance4}, condensateur, \text{générateur}\}$$

$$K_{det} = \{amplificateur, connection1, connetion2\}$$

Afin de trouver le meilleur placement de capteurs qui satisfait ces spécifications, l'algorithme 15 (page 154) est utilisé en tenant en compte les définitions 7.2.1, 7.2.2 et 7.2.3. Il mène au resultat suivant : $V_{minimal} = \{v_0, v_2, v_3, i_1, i_3, i_4, v_{1a}, v_{1b}, v_{4a}, v_{4b}\}$

avec un coût de 13. Les capteurs : $capteur1$, $capteur2$, $capteur3$, $capteur4$, $capteur5$, $capteur6$, $capteur7$, $capteur8$, $capteur9$ et $capteur10$ mesurant les variables de $V_{minimal}$ sont respectivement modélisés par les contraintes terminales : $k_{14}$, $k_{15}$, $k_{17}$, $k_{17}$, $k_{18}$, $k_{19}$, $k_{20}$, $k_{21}$, $k_{22}$ et $k_{23}$.

Afin de valider ce résultat, la méthode proposée dans la chapitre 4 sera utilisée pour concevoir toutes les **MTPF**s.
Les structures de capteurs ajoutés sont entrées dans **DXLAB** :

```
[14] structure v0 named s14 models capteur1
s14 = l{-},{v0}l models capteur1
[15] structure v2 named s15 models capteur2
s15 = l{-},{v2}l models capteur2
[16] structure v3 named s16 models capteur3
s16 = l{-},{v3}l models capteur3
[17] structure i1 named s17 models capteur4
s17 = l{-},{i1}l models capteur4
[18] structure i3 named s18 models capteur5
s18 = l{-},{i3}l models capteur5
[19] structure i4 named s19 models capteur6
s19 = l{-},{i4}l models capteur6
[20] structure v1a named s20 models capteur7
s20 = l{-},{v1a}l models capteur7
[21] structure v1b named s21 models capteur8
s21 = l{-},{v1b}l models capteur8
[22] structure v4a named s22 models capteur9
s22 = l{-},{v4a}l models capteur9
[23] structure v4b named s23 models capteur10
s23 = l{-},{v4b}l models capteur10
```

En utilisant **DXLAB** qui utilise les résultats du chapitre 4, on obtient toutes les **MTPF**s. Pour des raisons de simplification, nous n'allons montrer que les **MTPF**s de base (voir la section 4.4.3) :

```
[25] show tss minimal
tss001:
support=s17[capteur4],s2[connection1a],s13[générateur],s23[capteur10],s9[résistance3],
s18[capteur5],s6[résistance1]  (weight=7)
Formula=((((s9-i3-s18)-v4b-s23)-v1-(((s2-i3-s18)-i1-s17)-i2-(s6-v0-s13)))
tss002:
support=s5[connection1d],s4[connection1c],s15[capteur2],s1[amplificateur],s21[capteur8]
(weight=4)
Formula=((s5-v1c-(s1-v2-s15))-v1-(s4-v1b-s21))
                    :
                    :
tss027:
support=s17[capteur4],s3[connection1b],s2[connection1a],s14[capteur1],s20[capteur7],
s18[capteur5],s6[résistance1] (weight=6)
Formula=((((s2-i3-s18)-i1-s17)-i2-(s6-v0-s14))-v1-(s3-v1a-s20)
```

Afin de vérifier si les spécifications partielles sont satisfaites, nous utilisons l'instruc-

tion “show diagnosability” dans **DXLAB** :

```
[27] show diagnosability
Diagnosable components are: capteur10,capteur5,générateur,résistance3,résistance 1,capteur2,
capteur7,connection1b,condensateur,capteur3,capteur9,capteur1,résistance2
Detectable but not discriminable components are:
-connection1a,capteur4
-capteur8,connection1c
-connection1d,amplificateur
-connection2b,connection2a
-capteur6,résistance4
```

Selon ces résultats, les composants qui peuvent être détectées sont : $\{amplificateur, connection1, connetion2\}$. Les composants diagnosticables sont : {résistance1, résistance2, résistance3, résistance4, *condensateur*, générateur}.
Ces résultats conforment aux attentes et ça valide notre approche.

### 7.2.2 Système logique

La méthode de placement de capteurs présentée dans la section 6.7 va être appliquée sur le circuit logique (figure 4.11) présenté dans la section 4.5.3. Ce système est modélisé par les contraintes suivantes :

$$\begin{aligned} XOR_1 : k_1 : x_4 &= x_1 \oplus x_2 \\ AND_1 : k_2 : x_5 &= x_1 \wedge x_2 \\ AND_2 : k_3 : x_6 &= x_3 \wedge x_4 \\ XOR_2 : k_4 : x_7 &= x_3 \oplus x_4 \\ OR_1 : k_5 : x_8 &= x_5 \vee x_6 \end{aligned}$$

Dans cet exemple, nous tenons compte la notion de déductibilité des variables. Dans les contraintes $k_2$, $k_3$ et $k_5$, seules les variables $x_5$, $x_6$ et $x_8$ sont déductibles et les autres variables sont non déductibles. La matrice structurelle est représentée par le tableau 7.2.
Nous commençons par rentrer la description de ce système dans le logiciel **DXLAB** :

TAB. 7.2 – La matrice structurelle du sysytème logique

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ | $x_8$ |
|---|---|---|---|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_2$ | -1 | -1 | 0 | 0 | 1 | 0 | 0 | 0 |
| $k_3$ | 0 | 0 | -1 | -1 | 0 | 1 | 0 | 0 |
| $k_4$ | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| $k_5$ | 0 | 0 | 0 | 0 | -1 | -1 | 0 | 1 |

```
Welcome into DXLAB version 0.2.2

[1] structure x1 x2 x4 named s1 models XOR1
s1 = I{-},{x4,x2,x1}I models XOR1
[2] structure <x1 x2>I<x5> named s2 models AND1
s2 = I{x2,x1},{x5}I models AND1
[3] structure <x3 x4>I<x6> named s3 models AND2
s3 = I{x4,x3},{x6}I models AND2
[4] structure x3 x4 x7 named s4 models XOR2
s4 = I{-},{x4,x3,x7}I models XOR2
[5] structure <x5 x6>I<x8> named s5 models OR1
s5 = I{x6,x5},{x8}I models OR1
```

Supposons que les coûts des mesures des variables soient :

$$cost(x_1) = cost(x_2) = cost(x_3) = cost(x_4) = cost(x_5) \ldots$$
$$\ldots = cost(x_6) = cost(x_7) = cost(x_8) = 1$$

Supposons maintenant que nous devions trouver le placement de capteurs qui satisfait aux spécifications complètes suivantes :

$$\mathbb{C}_{nondis} = \{\{AND_2, XOR_2\}\}$$
$$C_{nondet} = \{XOR_1, AND_1\}$$
$$C_{diag} = \{OR_1\}$$

Nous remarquons que cet exemple contient des variables non déductibles. Trouver le placement de capteurs satisfaisant les contraintes de diagnosticabilité requiert alors l'utilisation de la méthode proposée dans la section 6.7.
Les composants $C_{nondet} = \{XOR_1, AND_1\}$ doivent être non détectables. Les contraintes

$K_{nondet} = \{k_1, k_1\}$ modélsiant ces composants doivent donc être enlevées du système. La première étape consiste à mesurer toutes les variables du système après avoir enlevé les contraintes $K_{nondet}$. Les capteurs $capteur1$, $capteur2$, $capteur3$, $capteur4$, $capteur5$ et $capteur6$ mesurant toutes les variables du système sont respectivement modélisés par les contraintes terminales : $k_6$, $k_7$, $k_8$, $k_9$, $k_{10}$ et $k_{11}$. Ces contraintes terminales sont entrées dans **DXLAB** :

```
[6] structure x3 named s6 models capteur1
s6 = l{-},{x3}l models capteur1
[7] structure x4 named s7 models capteur2
s7 = l{-},{x4}l models capteur2
[8] structure x5 named s8 models capteur3
s8 = l{-},{x5}l models capteur3
[9] structure x6 named s9 models capteur4
s9 = l{-},{x6}l models capteur4
[10] structure x7 named s10 models capteur5
s10 = l{-},{x7}l models capteur5
[11] structure x8 named s11 models capteur6
s11 = l{-},{x8}l models capteur6
```

En enlevant les contraintes modélisant les composants $\{XOR_1, AND_1\}$ du système et en utilisant **DXLAB** qui utilise les réslutats du chapitre 4, nous obtenons les **MTPF**s suivantes :

```
[12] show tss
tss001: support=s4[XOR2],s3[AND2],s9[capteur4],s10[capteur5],s6[capteur1] (weight=6)
Formula=(((s3-x3-(s4-x7-s10))-x6-s9)-x4-((s4-x7-s10)-x3-s6))
tss002: support=s3[AND2],s9[capteur4],s7[capteur2],s6[capteur1] (weight=3)
Formula=(((s3-x3-s6)-x6-s9)-x4-s7)
tss003: support=s4[XOR2],s3[AND2],s10[capteur5],s9[capteur4],s7[capteur2] (weight=4)
Formula=(((s3-x3-(s4-x7-s10))-x6-s9)-x4-s7)
tss004: support=s5[OR1],s3[AND2],s11[capteur6],s8[capteur3],s7[capteur2],s6[capteur1]
(weight=5)
Formula=((((s5-x8-s11)-x6-(s3-x3-s6))-x5-s8)-x4-s7)
tss005: support=s5[OR1],s4[XOR2],s3[AND2],s11[capteur6],s10[capteur5],s8[capteur3],
s7[capteur2](weight=6)
Formula=((((s5-x8-s11)-x6-(s3-x3-(s4-x7-s10)))-x5-s8)-x4-s7)
tss006:
support=s5[OR1],s4[XOR2],s3[AND2],s11[capteur6],s10[capteur5],s8[capteur3],
s6[capteur1] (weight=8)
Formula=((((s5-x8-s11)-x6-(s3-x3-(s4-x7-s10)))-x5-s8)-x4-((s4-x7-s10)-x3-s6))
tss007: support=s4[XOR2],s10[capteur5],s7[capteur2],s6[capteur1] (weight=3)
Formula=(((s4-x7-s10)-x3-s6)-x4-s7)
tss008: support=s5[OR1],s11[capteur6],s9[capteur4],s8[capteur3] (weight=3)
Formula=(((s5-x8-s11)-x6-s9)-x5-s8)
```

TAB. 7.3 – Les sous systèmes testables du système logique

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_9$ | $k_{10}$ | $k_{11}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| $SST_2$ | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| $SST_3$ | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| $SST_4$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| $SST_5$ | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| $SST_6$ | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| $SST_7$ | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| $SST_8$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |

TAB. 7.4 – Les sous systèmes testables du système logique pour les spécifications complètes

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_8$ | $k_9$ | $k_{10}$ | $k_{11}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| $SST_6$ | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| $SST_8$ | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |

La matrice de signature de défauts est représentée par le tableau 7.3.

Afin de chercher les variables candidates pour la mesure, l'algorithme 16 est utilisé. L'algorithme 16 (page 161) trouve les contraintes $\{k_6, k_8, k_9, k_{10}, k_{11}\}$ associées aux variables : $\{x_3, x_5, x_6, x_7, x_8\}$ avec un coût de 5.
La matrice de signature de défauts résultant est représentée par le tableau 7.4.

Selon ces résultats, l'ensemble de contraintes du système qui ne peuvent pas être discriminables sont $\{k_3, k_4\}$. L'ensemble de contraintes qui ne peuvent pas être détectables sont $\{k_1, k_2\}$ et la contrainte diagnosticable est $\{k_5\}$. Appliquons la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$. Il est clair que les composants qui ne peuvent pas être discriminables sont : $\{AND_2, XOR_2\}$, les composants qui ne peuvent pas détactables sont $\{XOR_1, AND_1\}$ et le composant diagnoticable est $\{OR_1\}$. Ces résultats conforment aux attentes et ça valide notre approche.

L'algorithme 16 (page 161) trouve un autre ensemble de contraintes $\{k_9, k_{10}, k_{11}, k_{12}, k_{13}\}$ associées aux variables : $\{v_4, v_5, v_6, v_7, v_8\}$ avec un coût de 5.

Maintenant, considérons les spécifications partielles :

$$C_{diag} = \{XOR_1, AND_1, OR_1\}$$

$$C_{det} = \{XOR_2, AND_2\}$$

Appliquons la méthodes de conception des **MTPFs** présentée dans le chapitre 4, nous obtenons les sous-systèmes testables représentés par la matrice de signature de défauts 7.5.

Afin de chercher les variables candidates pour la mesure, l'algorithme 23 (page 164) est utilisé. L'algorithme 23 trouve les contraintes $\{k_6, k_7, k_9, k_{10}, k_{11}, k_{12}, k_{13}\}$ associées aux variables : $\{v_1, v_2, v_4, v_5, v_6, v_7, v_8\}$ avec un coût de 7. La matrice de signature de défauts résultant est représentée par le tableau 7.6.

Selon ces résultats, les contraintes qui peuvent être détectées sont : $\{k_3, k_4\}$. Les contraintes diagnosticables sont : $\{k_1, k_2, k_5\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les composants qui peuvent être détectés sont : $\{XOR_2, AND_2\}$ et les composants diagnosticables sont : $\{XOR_1, AND_1, OR_1\}$. Ces résultats conforment aux attentes et ça valide notre approche.

L'algorithme 23 (page 164) fournit un autre ensemble de capteurs satisfaisant les spécifications proposées. Ces capteurs sont modélisés par les contraintes terminales $\{k_6, k_7, k_8, k_{10}, k_{11}, k_{12}, k_{13}\}$ associées aux variables $\{v_1, v_2, v_3, v_5, v_6, v_7, v_8\}$ .

## 7.3 Application industrielle "générateur photovoltaïque raccordé au réseau"

Dans cette section, nous appliquons les méthodes de placement de capteurs et de conception de **MTPF**s à un système photovoltaïque qui sert à transformer l'énergie solaire en courant électrique. Ce projet est supporté par l'Agence Nationale de la Recherche (ANR). Il réunit le CEA (institut national de l'Énergie Solaire), la société Transénergie, le G2ELAB, et le laboratoire G-SCOP.

### 7.3.1 Introduction aux systèmes photovoltaïques

Cent ans après la découverte de l'effet photovoltaïque par Edouard Becquerel (1839), la première cellule capable de transformer l'énergie solaire en courant électrique fut mise au point par un groupe de chercheurs américains de Bell Labs. Aujourd'hui, les avancées technologiques ont permis un plus grand déploiement du photovoltaïque (PV) en particulier celui des installations connectées au réseau.

TAB. 7.5 – Les sous systèes testables du système logique

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_9$ | $k_{10}$ | $k_{11}$ | $k_{12}$ | $k_{13}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| $SST_2$ | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| $SST_3$ | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| $SST_4$ | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| $SST_5$ | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| $SST_6$ | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| $SST_7$ | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| $SST_8$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| $SST_9$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| $SST_{10}$ | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| $SST_{11}$ | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| $SST_{12}$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| $SST_{13}$ | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| $SST_{14}$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| $SST_{15}$ | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| $SST_{16}$ | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| $SST_{17}$ | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| $SST_{18}$ | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| $SST_{19}$ | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| $SST_{20}$ | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| $SST_{21}$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| $SST_{22}$ | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| $SST_{23}$ | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| $SST_{24}$ | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| $SST_{25}$ | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| $SST_{26}$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| $SST_{27}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| $SST_{28}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| $SST_{29}$ | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| $SST_{30}$ | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| $SST_{31}$ | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| $SST_{32}$ | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 |
| $SST_{33}$ | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| $SST_{34}$ | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| $SST_{35}$ | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |

TAB. 7.6 – Les sous systèmes testables du système logique pour les spécifications partielles

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_9$ | $k_{10}$ | $k_{11}$ | $k_{12}$ | $k_{13}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| $SST_3$ | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| $SST_4$ | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| $SST_5$ | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| $SST_8$ | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| $SST_{11}$ | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| $SST_{12}$ | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| $SST_{14}$ | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| $SST_{15}$ | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| $SST_{19}$ | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| $SST_{20}$ | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| $SST_{24}$ | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| $SST_{28}$ | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| $SST_{29}$ | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| $SST_{31}$ | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| $SST_{34}$ | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

Les installations photovoltaïques peuvent être de deux types : centrales au sol ou intégrées au bâtiment (BIPV pour Building Integrated PV). Dans le cas de centrales au sol, les panneaux sont posés sur des supports à même le sol. Certains de ces supports peuvent pivoter afin de capter le maximum d'énergie solaire, ils sont appelés “trackers”. Les installations intégrées au bâtiment sont implantées dans des constructions, souvent sur la toiture mais aussi sur les façades des bâtiments, et subissent des contraintes de fonctionnement différentes (volume à respecter, environnement de fonctionnement, etc...).

On peut également distinguer différentes applications de ces types d'installations (voir la figure 7.4) : résidentiel et tertiaire. C'est la puissance de l'installation qui distingue ces applications. Les installations résidentielles sont d'une puissance inférieure à 5 kVA, alors que celles des tertiaires vont de 5 à 100 kvA, pour les petites, et de 100 à plus de 1000 kVA pour les grandes.

Les générateurs photovoltaïques connectés au réseau sont composés de plusieurs éléments :

– un assemblage de modules photovoltaïques constituant des cellules photovoltaïques.

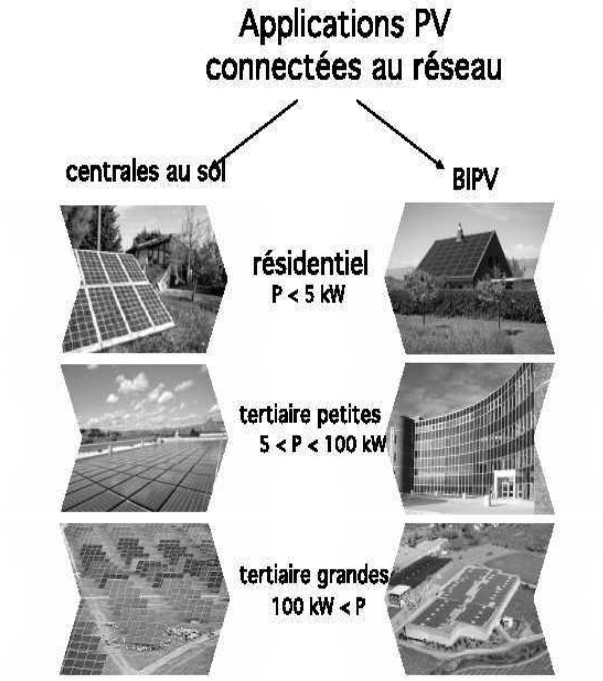


FIG. 7.4 – Applications photovoltaïques connectées au réseau

– un/des convertisseur(s) de puissance.
– les organes de sécurité et de raccordement au réseau.

Les cellules actuellement industrialisées en grande quantité sont produites à partir de silicium. Ces cellules photovoltaïques sont fragiles et sensibles à l'environnement extérieur. Elles sont donc munies d'une protection mécanique (l'encapsulation). La tension et la puissance d'une cellule ne sont pas adaptées aux applications courantes, il est donc nécessaire de les associer. Pour toutes ces raisons, les cellules sont assemblées en modules photovoltaïques.

La mise en série des cellules permet d'augmenter la puissance sur une ligne en veillant à un bon appairage maximiser la production. Le courant reste identique tandis que la tension est multipliée par le nombre de cellules en série.

Un module photovoltaïque est un générateur de courant dont la caractéristique électrique varie en fonction de l'éclairement reçu et de sa température. Sa puissance augmente de façon proportionnelle avec l'éclairement et diminue lorsque la température de fonctionnement des cellules augmente (environ 0,4% par degré d'élévation de la température).

Les convertisseurs de puissance ont plusieurs fonctions :

– ils convertissent le courant continu en courant alternatif en phase avec le réseau.
– ils font fonctionner les capteurs PV au maximum de leur puissance quelque soient l'ensoleillement et la température.

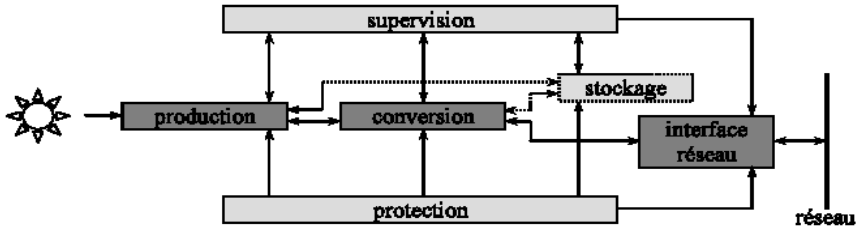


FIG. 7.5 – Fonctionnalités générales des systèmes photovoltaïques connectés au réseau

– ils se déconnectent en cas d'absence de tension du réseau.

Les convertisseurs de puissance sont caractérisés par :

– un rendement élevé au niveau de puissance usuelle de l'installation (de 92% à 96%).
– une durée de garantie étendue et extensible.
– une faible consommation.
– un taux d'harmonique et un niveau sonore faibles.
– de faibles perturbations électromagnétiques.

La connection onduleur réseau lors de l'injection provoque une augmentation de la tension réseau qui peut atteindre ou dépasser la tension maximale admissible en BT 230/400v qui est de + 6% max soit 244v/424v.

## A Fonctions macroscopiques des installations Pv

Ayant comme but final la production d'énergie, une installation photovoltaïque représente la mise en place d'une chaîne de conversion de l'irradiation lumineuse en énergie électrique qui se base sur l'effet photovoltaïque. Quelque soit son architecture détaillée, cette chaîne de conversion peut se décomposer en plusieurs sous-systèmes délimités selon leurs fonctionnalités. Un premier aperçu d'un schéma conceptuel des systèmes photovoltaïques connectés au réseau, identifiant les principaux sous-systèmes fonctionnels et leurs fonctions générales, est donné par la figure 7.5.

En regardant la figure 7.5, nous pouvons séparer les fonctions de base "production, conversion, interface réseau", représentées par des boites grisées foncées des fonctions auxiliaires "stockage, protection et supervision", grisées claires. Les fonctions de base

sont associées au système de production d'énergie proprement dit (elles apparaissent dans toutes les configurations du systèmes photovoltaïques), tandis que les fonctions auxiliaires aident au bon fonctionnement du système de base. Nous détaillons ensuite chaque groupe de fonctions.

**A.1 Fonctions de base** La fonction de production s'exerce au niveau des cellules qui composent les panneaux photovoltaïques. Plusieurs technologies de fabrication de cellules peuvent être actuellement utilisées : monocristallin, polycristallin ou amorphe. Chacune ayant des avantages et des inconvénients. Quant aux panneaux, ils peuvent être agencés de trois manières principales : série, parallèle, ou série parallèle sur le même champ photovoltaïque.
Les fonctions de conversion et d'interfaçage avec le réseau sont réalisées à l'aide d'un ou de plusieurs étages de traitement de la puissance captée, ayant comme but final de conditionner la puissance conformément aux demandes des standards de réseau en vigueur. Différentes architectures d'électronique de puissance sont utilisées à cet effet. La variété des concepts des systèmes est principalement due à la variabilité des structures réalisant ces fonctions. Il existe des installations photovoltaïques où, pour des raisons de coût moins élevé, l'on préfère qu'un seul dispositif "l'onduleur centralisé" soit chargé de toutes les fonctions. En effet, il est assez difficile de différencier nettement les actions appartenant à chacune des deux. En principe, la conversion regroupe :

- la maximisation de la puissance captée.
- l'élévation de tension.

tandis que la fonction d'interface réseau électrique comprend :

- la formation des courants sinusoïdaux.
- la limitation de valeurs de courants.
- le contrôle des puissances active et réactive.
- le contrôle des harmoniques envoyées vers le réseau.
- la gestion des perturbations induites par le réseau, dont les principales sont :
  - réaction aux creux, dépassements et fluctuations de tension.
  - réaction aux court-circuits.
  - gestion des harmoniques et des inter-harmoniques superposées.
  - îlotage en tant que mode d'opération dégradé.

Le stockage est une fonction optionnelle pour les systèmes connectés au réseau, qui suppose l'envoi d'une partie de l'énergie produite vers un élément de stockage habituellement une batterie). Le bloc correspondant au stockage est représenté en pointillés dans la figure 7.5.

**A.2 Fonctions auxiliaires** L'implémentation de fonctions auxiliaires s'appuie sur un échange d'informations avec le système de base. Dans le cas de la protection, cet échange est représenté par la figure 7.5 avec des flèches unidirectionnelles, puisqu'il s'agit

des protections primaires, ne supposant pas un traitement spécial de l'information de retour :

- protection anti-foudre.
- diodes/fusibles anti-retour de courant.
- diodes by-pass pour la limitation du mismatching, etc.

## B Topologies des installations photovoltaïques

Les modules PV peuvent être agencés de plusieurs manières en amont de la connection au réseau de distribution. Dans cette section, nous présentons quelques topologies d'installations photovoltaïques.

- Onduleur central (Abella and Chenlo [2004]) : L'architecture la plus classique est composée d'un seul onduleur réalisant l'interface entre le réseau et le champ photovoltaïque, où des chaînes de modules sont connectées en parallèle. Ce montage est généralement utilisé pour des installations de grandes puissances (20-400 kW) dans lequel des protections anti-retour de courant sont implantées par strings.
- Onduleur rangée (Abella and Chenlo [2004]) : L'architecture la plus employée est celle de l'onduleur rangée, qui consiste à implanter un onduleur au bout de chaque chaîne (ou "string "). Les onduleurs sont ensuite connectés en parallèle au réseau électrique, nécessitant une coordination entre les onduleurs (du type maître-esclave) lors d'îlotages. Cette architecture est représentée par la figure 7.6. Le projet traité dans cette section utilise cette architecture.
- Onduleur rangée "team concept" (Myrzik and Calais [2003]) : Une variante de l'onduleur rangée consiste à introduire des sectionneurs en amont des onduleurs, côté courant continu. La configuration "team concept" de l'onduleur rangée permet d'améliorer le rendement de conversion d'énergie lors de faible production des panneaux photovoltaïques. Il permet d'utiliser moins d'onduleurs pour convertir l'énergie issue des strings pour qu'ils fonctionnent au plus proche de leurs puissances nominales.%.
- Hacheur rangée (Meinhardt and Cramer [2001]) : L'utilisation de plusieurs étages de conversion pourrait améliorer le rendement de conversion du système et la fiabilité du système en dissociant les fonctionnalités de l'onduleur. Souvent présenté comme une symbiose de l'onduleur central et de l'onduleur rangée, le hacheur rangée, aussi appelé convertisseur multi-string, utilise un hacheur au bout de chaque string du système PV. L'interfaçage avec le réseau est réalisée grâce à un seul onduleur, facilitant la participation aux services systèmes de l'installation PV.
- Onduleur individuel (Abella and Chenlo [2004]) : Aujourd'hui, beaucoup d'espoirs résident dans la réalisation de convertisseurs individuels pour les modules photovoltaïques. La configuration onduleur individuel permet de connecter chaque module au réseau grâce à un onduleur intégré.

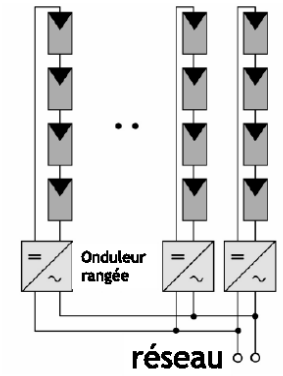


FIG. 7.6 – Onduleur Rangée

– Onduleur cellulaire (Wuest et al. [1994]) : La structure la plus prometteuse semble être celle comprenant un convertisseur au niveau de la cellule photovoltaïque. Dans le cas d'un onduleur cellulaire, un onduleur à deux étages, d'environ 200 W, est placée aux bornes de la cellule photovoltaïque.

### 7.3.2 Installation photovoltaïque étudiée

L'installation a été conçu pour une ville qui souhaite soutenir la promotion des énergies renouvelables. Cette ville est un espace culturel qui abrite : un théâtre-cinéma, une bibliothèque, un conservatoire répartis sur 5 niveaux. La couverture du bâtiment où le système PV doit être installé est une toiture terrasse technique sur laquelle est disposé un ensemble d'équipements de ventilation et de chauffage.

#### A Caractéristiques de l'installation photovoltaïque raccordée au réseau

La puissance crête nominale du générateur photovoltaïque est de 10 800 Wc. Le dimensionnement de l'installation est le suivant :

| | |
|---|---|
| ***Mode d'implantation*** | Modules photovoltaïques fixés sur bacs en PEHD lestés posés en sur imposition de la toiture terrasse |
| ***Type de modules*** | Modules photovoltaïques de type standard « verre tedlar avec cadre » |
| **Nombre prévisionnel de modules photovoltaïques** | 54 modules photovoltaïques de 200 Wc (ou autres nombre et puissance unitaire permettant de satisfaire la puissance crête nominale globale exigée) |
| ***Connexions entre les modules*** | 6 chaînes de 9 modules en série |
| **Nombre prévisionnel d'onduleurs spécifiques à la connexion réseau et puissance nominale globale des onduleurs** | 3 onduleurs de puissance unitaire 3300 VA (soit une puissance globale de 9,9 kVA) |

Le générateur photovoltaïque utilisée pour satisfaire la puissance crête nominale est représenté par la figure 7.7.

## B Diagnostic du système photovoltaïque proposé

Afin de prévenir tout défaut sur un générateur solaire photovoltaïque qui peuvent ne se révéler que longtemps après l'occurrence, l'objectif du projet de recherche est de concevoir un système de diagnostic permettant de détecter et localiser les défauts dans le système.
Dans cette section, nous appliquons la méthodes du placement de capteurs proposées dans le chapitre 6 afin de trouver le placement de capteurs optimal satisfaisant des contraintes de détectabilité et de diagnosticabilité. Nous appliquons aussi la méthode de conception de **MTPF**s proposée dans la section 4 pour trouver tous les tests nécessaires pour la détection et la localisation des défauts.

Reprenons le système photovoltaïque présenté par la figure 7.7 et supposons que les spécifications globales requises soient : tous les onduleurs du système doivent être diagnosticables car leur **MTTF** et **MTBF** sont plus faibles et les autres composants du système doivent être au moins détectables. Nous devons chercher les variables qui doivent être mesurées pour satisfaire ces spécifications en considérant que toutes les variables ont le même coût de mesure.

Nous remarquons que ce système représente une certaine complexité parce qu'il contient 78 composants et 96 variables. Nous allons simplifier la résolution du problème en adoptant une approche de résolution hiérarchisée.
Une définition importante est introduite avant d'expliquer l'approche hiérarchisée.

**Définition 7.3.1.** *Un macro-composant est un ensemble de composants vérifiant la*

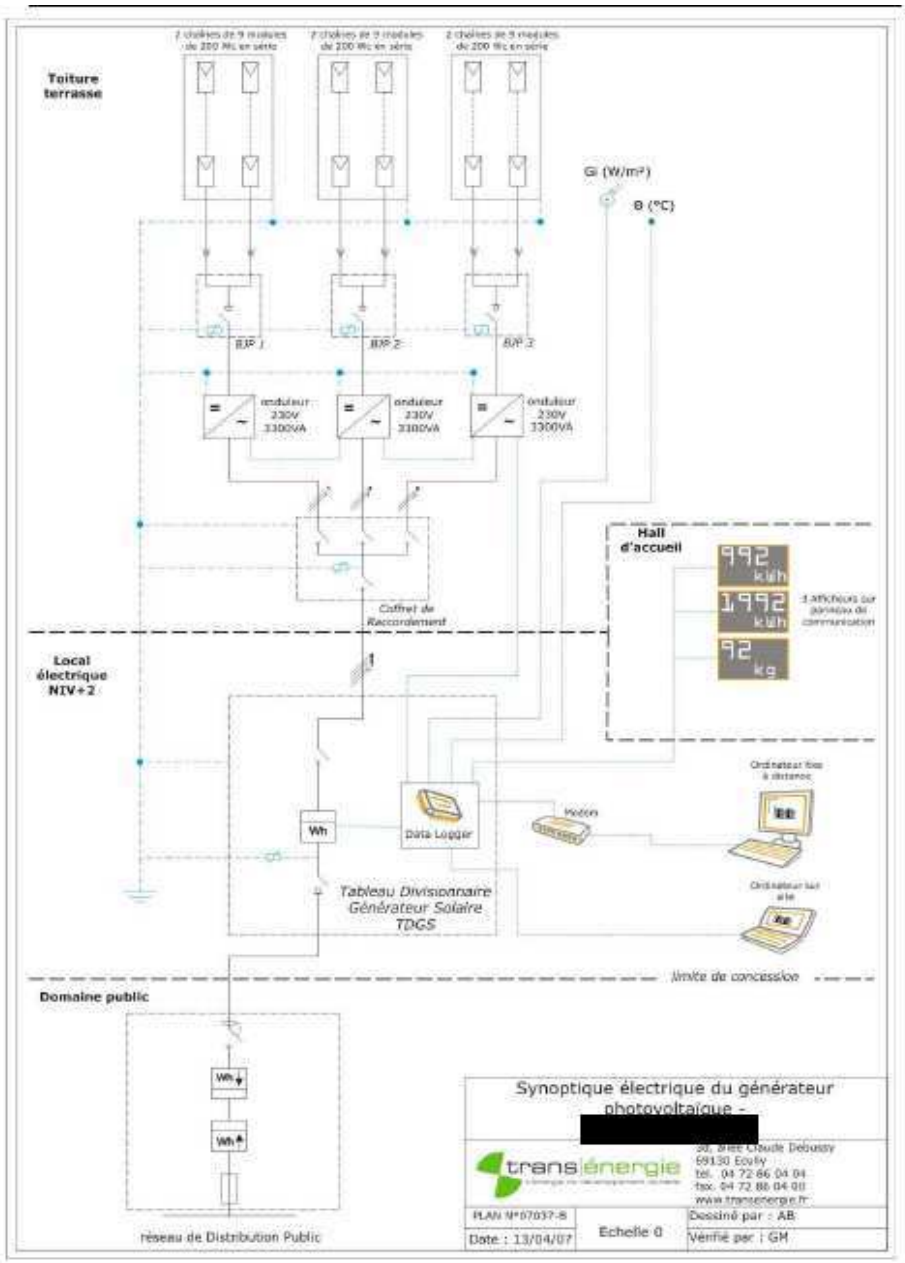


Fig. 7.7 – Le schéma du système photovoltaïque

*condition suivante : en mesurant les variables d'entrées et de sorties, on obtient un et un seul sous-système testable.*

L'approche de résolution hiérarchisée se compose en plusieurs étapes :
- on regroupe les composants en macro-composants
- on transforme les spécifications globales associées aux composants en spécifications globales associées aux macro-composants
- on résout le problème de capteurs au niveau des spécifications globales associées aux macro-composants
- on ajoute les capteurs nécessaires à la satisfaction des spécifications globales associées aux macro-composants
- pour chaque macro-composant,
  - on rentre à l'intérieur de chaque macro-composant en le détaillant en composants
  - on extrait les spécifications locales en partant des spécifications globales associées aux composants
  - on résout le problème de placement de capteurs au niveau des composants du macro-composant
  - on ajoute les capteurs nécessaire pour la satisfaction des spécifications locales.

Cette approche hiérarchisée va être maintenant appliquée au système photovoltaïque.

La première étape consiste à regrouper les composants du système en six macro-composants : les macro-composants $MC_1$, $MC_2$, $MC_3$ constitués chacun de 18 modules, un boitier de jonction et un onduleur, le coffret de raccordement $MC_4$, le capteur mesurant le rayonnement solaire, déjà implanté dans le système $MC_5$ et le macro-composant constitué des autres composants du système $MC_6$.

Le système photovoltaïque simplifié est représenté par la figure 7.8. Ce système simplifié est constitué de 6 macro-composants $MC_1$, $MC_2$, $MC_3$, $MC_4$, $MC_5$ et $MC_6$ qui sont modélisés par les contraintes $k_1$, $k_2$, $k_3$, $k_4$, $k_5$ et $k_6$. Les variables du système sont : le rayonnement solaire $r.sol$, les courants : $i1r$, $i2r$, $i3r$, $ira$, $i.res$ et les tensions : $v1r$, $v2r$, $v3r$, $v.ra$,$v.res$. Ce système peut être représenté par la matrice structurelle 7.7.
Pour que les spécifications globales requises associées aux composants soient satisfaites, nous devons transformer ces spécifications en spécifications globales associées aux macro-composants. Les spécifications associées aux macro-composant sont : les macro-composants $MC_1$, $MC_2$, $MC_3$ contenant les onduleurs doivent être diagnosticables et les macro-composants $MC_4$, $MC_5$, $MC_6$ doivent être au moins détectables.
Nous devons chercher le placement de capteurs qui satisfait ces nouvelles spécifications. Afin de trouver le meilleur placement de capteurs, l'algorithme 15 (page 154) est utilisé. En prenant en compte que toutes les variables ont le même coût de mesure, l'algorithme 15 conduit au résultat suivant : $V_{minimal} = \{i1r, i2r, i3r, i.res, v1r, v2r, v3r, v.ra, v.res\}$. Les capteurs : $capteur1$, $capteur2$, $capteur3$, $capteur4$, $capteur5$, $capteur6$, $capteur7$,

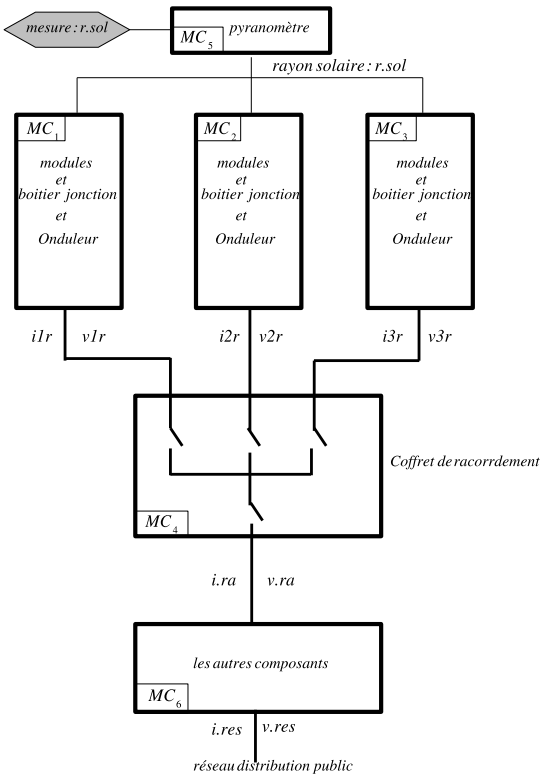


FIG. 7.8 – Le schéma simplifié du système photovoltaïque

TAB. 7.7 – La matrice structurelle du système photovoltaïque simplifié

| | $r.sol$ | $i1r$ | $i2r$ | $i3r$ | $i.ra$ | $i.res$ | $v1r$ | $v2r$ | $v3r$ | $v.ra$ | $v.res$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_2$ | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $k_3$ | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $k_4$ | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| $k_5$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_6$ | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

*capteur*8 et *capteur*9 mesurant les variables $i1r$, $i2r$, $i3r$, $i.res$, $v1r$, $v2r$, $v3r$, $v.ra$ et $v.res$ sont respectivement modélisés par les contraintes terminales : $k_7$, $k_8$, $k_9$, $k_{10}$, $k_{11}$, $k_{12}$, $k_{13}$, $k_{14}$ et $k_{15}$.
Afin de valider ce résultat, la méthode proposée le chapitre 4 a été utilisée pour concevoir toutes les **MTPF**s. En utilisant **DXLAB**, nous obtenons 57 **MTPFs** . Pour des raisons de simplification, nous ne détaillons que les **MTPF**s de base (voir la section 4.4.3).

```
18] show tss minimal
tss001:
support=s9[capteur3],s8[capteur2],s7[capteur1],s6[MC6],s15[capteur9],s14[capteur8],
s13[capteur7],s4[MC4],s12[capteur6],s11[capteur5],s10[capteur4] (weight=11)
  Formula=((((s6-v.ra-s14)-v.res-s15)-i.res-s10)-i.ra-(((((((s4-i3r-s9)-v2r-s12)-v.ra-s14)-i2r-s8)-v1r-
s11)-i1r-s7)-v3r-s13))
tss002:
support=s5[MC5],s1[MC1],s11[capteur5],s7[capteur1] (weight=3)
  Formula=(((s1-v1r-s11)-r.sol-s5)-i1r-s7)
tss003:
support=s5[MC5],s3[MC3],s13[capteur7],s9[capteur3] (weight=3)
  Formula=(((s3-i3r-s9)-r.sol-s5)-v3r-s13)
tss004:
support=s5[MC5],s2[MC2],s12[capteur6],s8[capteur2] (weight=3)
  Formula=(((s2-v2r-s12)-i2r-s8)-r.sol-s5)
```

Les **MTPF**s de base sont représentées par la figure 7.9. Ces **MTPF**s de base sont représentées par la matrice de signature de défauts donnée par le tableau 7.8.

Tab. 7.8 – Les sous-systèmes testables du système photovoltaïque simplifié

| | $k_1$ | $k_2$ | $k_3$ | $k_4$ | $k_5$ | $k_6$ | $k_7$ | $k_8$ | $k_9$ | $k_{10}$ | $k_{11}$ | $k_{12}$ | $k_{13}$ | $k_{14}$ | $k_{15}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $SST_1$ | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $SST_2$ | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| $SST_3$ | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| $SST_4$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

Selon ces résultats, les contraintes qui peuvent être détectées sont : $\{k_4, k_5, k_6\}$. Les contraintes diagnosticables sont : $\{k_1, k_2, k_3\}$. En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que les macro-composants qui peuvent être détectés sont : $\{MC_4, MC_5, MC_6\}$ et les macro-composants diagnosticables sont : $\{MC_1, MC_2, MC_3\}$.

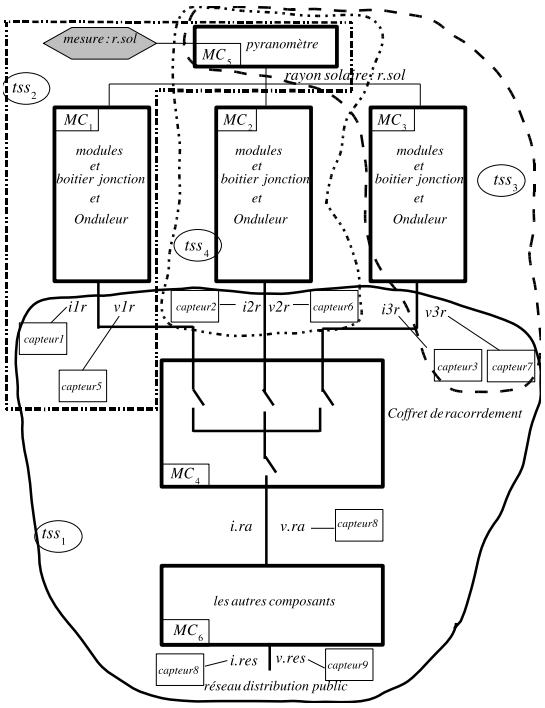


FIG. 7.9 – Les **MTPF**s de base du système photovoltaïque simplifié

Nous pouvons utiliser l'instruction "show diagnosability" pour vérifier que les spécifications sont satisfaites.

```
[17] show diagnosability
Diagnosable components are: MC5,MC1,MC3,MC2
Detectable but not discriminable components are:
  -capteur7,capteur3
  -capteur6,capteur2
  -MC6,capteur4,MC4,capteur9,capteur8
  -capteur5,capteur1
```

L'étape suivante consiste à faire le diagnostic à l'intérieur des macro-composants : $MC_1$, $MC_2$ et $MC_3$ :

– en ce qui concerne le macro-composant $MC_1$ :
Pour satisfaire les spécifications globales associées aux composants proposées dans cette application (c'est-à-dire, les onduleurs doivent être diagnosticables), nous allons faire le diagnostic à l'intérieur du macro-composant $MC_1$. Les spécifications locales associées au macro-composant $MC_1$ sont : l'onduleur1 associé à $MC_1$ doit être diagnosticable et les autres composants associés à $MC_1$ doivent être au moins détectables. Alors, nous devons chercher le placement de capteurs qui permet de distinguer le défaut qui peut affecter l'onduleur1 des autres défauts sur les autres

composants associés à $MC_1$.
Le macro-composant $MC_1$ peut être modélisé par le schéma 7.10.

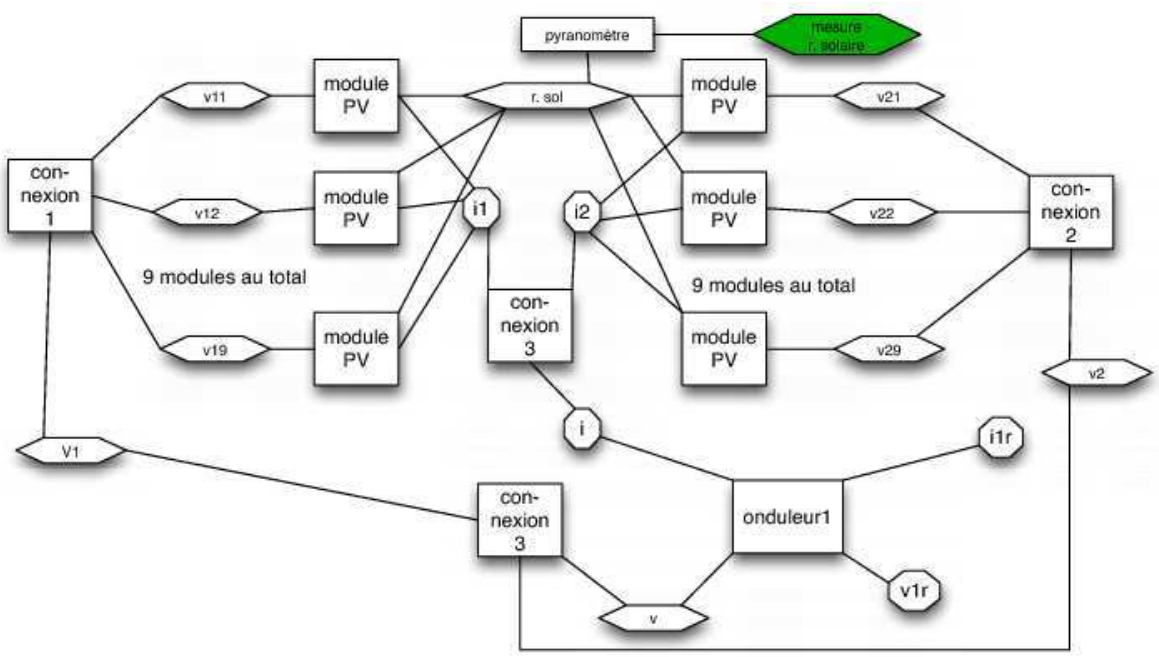


FIG. 7.10 – Le schéma représentant la modélisation du macro-composant $MC_1$

Il est constitué de deux séries de modules (chaque série contient 9 modules), d'un onduleur, d'un boitier jonction, et de trois connections. Il fait intervenir 27 variables.

$MC_1$ est modélisé par les structures des contraintes suivantes :

$module1 : s(k_1) = \llcorner v11, i1, r.sol \lrcorner$
$module2 : s(k_2) = \llcorner v12, i1, r.sol \lrcorner$
$module3 : s(k_3) = \llcorner v13, i1, r.sol \lrcorner$
$module4 : s(k_4) = \llcorner v14, i1, r.sol \lrcorner$
$module5 : s(k_5) = \llcorner v15, i1, r.sol \lrcorner$
$module6 : s(k_6) = \llcorner v16, i1, r.sol \lrcorner$
$module7 : s(k_7) = \llcorner v17, i1, r.sol \lrcorner$
$module8 : s(k_8) = \llcorner v18, i1, r.sol \lrcorner$
$module9 : s(k_9) = \llcorner v19, i1, r.sol \lrcorner$
$module10 : s(k_{10}) = \llcorner v21, i2, r.sol \lrcorner$
$module11 : s(k_{11}) = \llcorner v22, i2, r.sol \lrcorner$
$module12 : s(k_{12}) = \llcorner v23, i_2, r.sol \lrcorner$
$module13 : s(k_{13}) = \llcorner v24, i2, r.sol \lrcorner$
$module14 : s(k_{14}) = \llcorner v25, i2, r.sol \lrcorner$
$module15 : s(k_{15}) = \llcorner v26, i2, r.sol \lrcorner$
$module16 : s(k_{16}) = \llcorner v27, i2, r.sol \lrcorner$
$module17 : s(k_{17}) = \llcorner v28, i2, r.sol \lrcorner$
$module18 : s(k_{18}) = \llcorner v29, i2, r.sol \lrcorner$
$connnexion1 : s(k_{19}) = \llcorner v11, v12, v13, v14, v15, v16, v17, v18, v19, v1 \lrcorner$
$connection2 : s(k_{20}) = \llcorner v21, v22, v23, v24, v25, v26, v27, v28, v29, v2 \lrcorner$
$connection3a : s(k_{21}) = \llcorner i1, i2, i \lrcorner$
$connection3b : s(k_{22}) = \llcorner v1, v2, v \lrcorner$
$onduleur1 : s(k_{23}) = \llcorner v, i, v1r, i1r \lrcorner$

Le macro-composant $MC_1$ peut être représenté par la matrice structurelle 7.9.
Pour que l'onduleur1 soit diagnosticable et que les autres composants associés à $MC_1$ soient au moins détectables, l'algorithme 15 (page 154) est utilisé afin de trouver le meilleur placement de capteurs qui satisfait ces spécifications. En tenant compte que toutes les variables ont le même coût de mesure et que les variables $v1r$ et $i1r$ sont mesurées (étape précédente) et que le rayonnement solaire est mesuré, l'agorithme 15 mène au résultat suivant : $V_{minimal_1} = \{v1, v, i\}$. Les capteurs : *capteur*1, *capteur*2, *capteur*3, *capteur*4 et *capteur*5 mesurant les variables $v1$, $v$, $i$, $v1r$ et $i1r$ sont respectivement modélisés par les contraintes terminales : $k_{25}$, $k_{26}$, $k_{27}$, $k_{28}$ et $k_{29}$.

TAB. 7.9 – La matrice structurelle du le macro-composant $MC_1$

| | r.sol | v11 | v12 | v13 | v14 | v15 | v16 | v17 | v18 | v19 | v21 | v22 | v23 | v24 | v25 | v26 | v27 | v28 | v29 | i1 | i2 | v1 | v2 | v | i | v1r | i1r |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_1$ | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_2$ | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_3$ | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_4$ | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_5$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_6$ | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_7$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_8$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_9$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{10}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{11}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{12}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{13}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{14}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{15}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{16}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{17}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{18}$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k_{19}$ | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $k_{20}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $k_{21}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| $k_{22}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| $k_{23}$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

TAB. 7.10 – Les sous systèmes testables pour le macro-composant $MC_1$

| | k1 | k2 | k3 | k4 | k5 | k6 | k7 | k8 | k9 | k10 | k11 | k12 | k13 | k14 | k15 | k16 | k17 | k18 | k19 | k20 | k21 | k22 | k23 | k24 | k25 | k26 | k27 | k28 | k29 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SST1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| SST2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| SST3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| SST4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |

Afin de valider ce résultat, la méthode proposée dans le chapitre 4 a été utilisée pour concevoir toutes les **MTPF**s. En appliquant l'outil DXLAB, nous obtenons 4 **MTPF**s. La matrice de signature de défauts est donnée par le tableau 7.10. Selon ces résultats la contrainte diagnosticable est : $\{k_{23}\}$. Les contraintes qui peuvent être au moins détectées sont :
$\{k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}, k_{11}, k_{12}, k_{13}, k_{14}, k_{15}, k_{16}, k_{17}, k_{18}, k_{19}, k_{20}, k_{21}, k_{22}\}$.
En appliquant la fonction $\Phi : K_\Sigma \longrightarrow C_\Sigma$, il est clair que le composant qui peut être diagnosticable est l'onduleur1 et les autres composants peuvent être au moins détectables. Ces résultats conforment aux attentes et ça valide notre approche.
Nous pouvons utiliser l'instruction "show diagnosability" pour vérifier que les spécifications sont satisfaites.

```
[7] show diagnosability
Diagnosable components are: capteur3,capteur2
Detectable but not discriminable components are:
   -connection3b,module9,connection3a,module8,connection2,module7,module6,module5,module4,
   module3,module2,pyranomètre,module1,capteur1,connection1,module18,module17,module16,
   module15,module14,module13,module12,module11,module10
  -onduleur,capteur5,capteur4
```

– macro-composant $MC_2$ :
Pour satisfaire les spécifications globales associées aux composants proposées dans cette application (les onduleurs doivent être diagnosticables), nous allons faire le diagnostic à l'intérieur du macro-composant $MC_2$.
Les spécifications locales associées au macro-composant $MC_2$ sont : l'onduleur2 associé à $MC_2$ doit être diagnosticable et les autres composants associés à $MC_2$ doivent être au moins détectables. Alors, nous devons chercher le placement de capteurs qui permet de distinguer le défaut qui peut affecter l'onduleur2 des autres défauts sur les autres composants associés à $MC_2$.
Pour satisfaire les spécifications locales associées à $MC_2$, l'algorithme 15 (page 154) est utilisé afin de trouver le meilleur placement de capteurs satisfaisant ces spécifications. En tenant compte que toutes les variables ont le même coût de mesure, que

les variables $v2r$ et $i2r$ sont mesurées (étape précédente) et que le rayonnement solaire est mesuré, l'algorithme 15 mène au résultat suivant : $V_{minimal_2} = \{v1^{'}, v^{'}, i^{'}, \}$. Les variables $v1^{'}$, $v^{'}$ et $i^{'}$ sont comparables aux variables $v1$, $v$ et $i$ représentées sur le schéma 7.10.

– macro-composant $MC_3$ :
Pour satisfaire les spécifications globales associées aux composants proposées dans cette application (les onduleurs doivent être diagnosticables), nous allons faire le diagnostic à l'intérieur du macro-composant $MC_3$. Les spécifications locales associées au macro-composant $MC_3$ est : l'onduleur3 associé à $MC_3$ doit être diagnosticable et les autres composants associés à $MC_3$ doivent être au moins détectables. Nous devons chercher le placement de capteurs qui permet de distinguer le défaut qui peut affecter l'onduleur3 des autres défauts sur les autres composants associés à $MC_3$.
Pour satisfaire les spécifications locales associées à $MC_3$, l'algorithme 15 (page 154) est utilisé afin de trouver le meilleur placement de capteurs qui satisfait ces spécifications. En tenant compte que toutes les variables ont le même coût de mesure, que les variables $v3r$ et $i3r$ sont mesurées (étape précédente) et que le rayonnement solaire est mesuré, l'algorithme 15 mène au résultat suivant : $V_{minimal_3} = \{v1^{''}, v^{''}, i^{''}\}$. Les variables $v1^{''}$, $v^{''}$ et $i^{''}$ sont comparables aux variables $v1$, $v$ et $i$ représentées sur le schéma 7.10.

Par conséquent, les variables qui doivent être mesurées pour satisfaire les spécifications globales associées aux composants dans cette application (c'est-à-dire, les 3 onduleurs doivent être diagnosticables et les autres composants doivent être au moins détectables) sont :
$V_{minimal} = V_{minimal} \cup V_{minimal_1} \cup V_{minimal_2} \cup V_{minimal_3} = \{i1r, i2r, i3r, i.res, v1r, v2r, v3r, v.ra, v.res, v1, v, i, v1^{'}, v^{'}, i^{'}, v1^{''}, v^{''}, i^{''}\}$.

Dans cette section, nous avons appliqué les méthodes du placement de capteurs et de la conception des **MTPFs** présentées dans cette mémoire sur une applications industrielle (installation photovoltaïque) grâce au logiciel **DXLAB**. Nous avons introduit une approche hiérarchisée permettant de trouver le meilleur placement de capteurs satisfaisant des spécifications de diagnosticabilité pour des systèmes complexes. Cette approche consiste à regrouper les composants d'un système en plusieurs macro-composants. En utilisant cette approche, nous avons analysé la diagnosticabilité du système constitué de plusieurs macro-composants. La diagnosticabilité à l'intérieur de chaque macro-composant peut alors être analysée. Cette approche présente deux avantages principaux :

– réduire la complexité du problème de placement de capteurs.
– raisonnement sur des macro-composants représente une démarche naturelle pour un expert.

**remarque 1.** Les composants associés à un macro-composant peut être un sous-macro-

composant. Dans ce cas nous pouvons appliquer la méthode hiérarchique jusqu'à arriver aux niveau des composants primaires. Par exemple, dans l'application que nous avons traitée, nous pouvons analyser la diagnosticabilité à l'intérieur des modules si le cahier des charges requis contient des cellules.

## 7.4 Conclusion

Dans ce chapitre, nous avons présenté un outil pour l'aide au diagnostic appelé **DXLAB**. Cet outil est un logiciel développé en Java. Nous avons illustré **DXLAB** sur deux systèmes : l'un est tiré du monde continu (amplificateur) et l'autre tiré du monde logique. En ce qui concerne l'amplificateur, nous avons appliqué les méthodes de placement de capteurs présenté dans le chapitre 6 en tenant compte qu'un composant peut être modélisé par plusieurs contraintes. Nous avons montré comment utiliser le logiciel **DXLAB** sur une application de nature très différente : un circuit logique. Nous avons alors validé notre résultat sur une véritable application industrielle avec 78 composants et 96 variables. Cette application industrielle (installation photovoltaïque) a été utilisée pour illustrer l'intérêt de notre méthode de placement de capteurs. La complexité du système nous a conduit à développer une approche hiérarchisée. Les trois exemples présentés dans ce chapitre sont des systèmes sous-déterminés (le nombre de variables est plus que le nombre de contraintes). Par conséquent, la méthode proposée dans (Frisk and Krysander [2007]) ne s'applique pas à ces systèmes parce qu'elle n'arrive à traiter que les systèmes juste-déterminés qui ne contiennent pas de partie sous-déterminée. Puisque le système photovoltaïque contient 78 composants et 96 variables, la méthode du placement de capteurs proposée dans (Travé-Massuyès et al. [2006]) est très difficile d'être appliquée sur ce système parce que le grand nombre de variables impose une grande complexité de calcul.

# Conclusion générale

Les travaux que nous avons présentés dans ce mémoire introduisent de nouvelles méthodes pour le diagnostic de défauts : des méthodes qui permettent de concevoir les sous-systèmes testables et des méthodes qui permettent de trouver le meilleur placement de capteurs satisfaisant des spécifications de diagnosticabilité.

En ce qui concerne de la conception des sous-systèmes testables, nous avons formalisé une nouvelle méthode structurelle. Cette méthode est basée sur un opérateur jointure venant de l'algèbre relationnel. Elle est basée également sur une abstraction structurelle des contraintes. En conséquence de quoi elle peut être appliquée à une très grande classe de systèmes : des systèmes continus échantillonés, linéaires ou non linéaires, statiques ou dynamiques, à base de règle ou non. Elle permet aussi de trouver tous les tests de détection en prenant en compte la notion de déductiblité des variables. Cette méthode permet aussi de prendre en compte les ensembles d'exclusions permettant d'éviter les sous-systèmes testables irréalistes. Nous avons illustrer cet avantage par l'exemple de réseau routier.
L'inconvénient des méthodes structurelles, et en particulier de celle présentée est que des sur-estimations de solutions peuvent se produire. Ces sur-estimations peuvent être partiellement évitées en tenant compte de la déductibilité des variables par rapport aux contraintes. Mais d'autres sur-estimations restent possibles. Par conséquent, certaines contraintes intervenant dans une **SST** peuvent ne pas être nécessaires. Ces contraintes doivent être enlevées a posteriori en concevant les tests de détection. Cet inconvénient ne pose pas un grand problème parce que la principale difficulté reste la détermination de l'ensemble des contraintes qui mènent aux relations de redondance analytique.

Nous avons aussi proposé une nouvelles approche pour le placement de capteurs prenant en compte des spécifications de diagnosticabilité et détectabilité. Nous avons considéré deux types de spécifications : des spécifications complètes, où tous les ensembles de contraintes diagnosticables, discriminables et non détectables sont spécifiés, et des spécifications partielles, où seule l'ensemble des contraintes qui doivent être diagnosticables et l'ensemble des contraintes qui doivent être au moins détectables sont spécifiés. Cette

approche pourrait conduire facilement à d'autres méthodes permettant d'appréhender d'autres types de spécifications. En pratique, nous intéressons du second cas car les spécifications complètes sont difficiles à établir.
Les méthodes proposées sont basées sur l'étude des propriétés des matrices structurelles pour établir les propriétés de détectabilité, de discriminabilité et de diagnosticabilité. Elles se basent aussi sur les algorithmes d'optimisation combinatoire pour déterminer de placement de capteurs satisfaisant au cahier des charges de coût minimal. Ces méthodes s'appliquent à une grande classe de systèmes parce qu'elles sont basées sur une approche structurelle. Les méthodes proposées permettent de trouver le meilleur placement de capteurs satisfaisant des spécifications de diagnosticabilité sans concevoir les relations de redondance analytique au préalable. Elles peuvent être appliquées à tous les systèmes (sur-déterminés, juste-déterminés et sous déterminés).

L'inconvénient de ces méthodes est qu'elles ne prennent pas en compte la déductibilité des variables des systèmes. Comme la déductibilité peut être très importante dans certains systèmes, nous avons proposé une autre méthode pour le placement de capteurs satisfaisant des critères de diagnosticabilité prenant en compte la notion de déductibilité. Cette méthode se décompose en de deux étapes : la première étape consiste à mesurer toutes les variables du système et à calculer tous les sous-systèmes testables et la deuxième consiste à enlever itérativement des contraintes terminales (capteurs), et les sous-systèmes testables contenant ces contraintes, de la matrice de signature de défauts jusqu' à ce que les spécifications complètes soient satisfaites, ou jusqu'à ce que les spécification partielles restent satisfaites. L'inconvénient de cette méthodes est qu'elle nécessite la conception préalable des relations de redondance analytique.

Plusieurs algorithmes, lemmes, et théorèmes sont proposés dans ce mémoire. Ils sont utilisés pour la conception des sous-systèmes testables et pour trouver des placements de capteurs minimaux. Nous avons également présenté un outil informatique développé en java pour l'aide au diagnostic qui capitalise nos résultats. Cet outil est appelé **DX-LAB**. Nous avons illustré cet outil avec deux systèmes : un vient du monde continu (amplificateur) et l'autre vient du monde logique. L'exemple de l'amplificateur a montré la possibilité d'appliquer notre méthode de placement de capteurs sur les systèmes où un composant peut être modélisé par plusieurs contraintes. À la fin du mémoire, nous avons validé nos résultats sur une véritable application industrielle : un système photovoltaïque. Cette application a permit de montrer l'intérêt de notre méthode de placement de capteurs. Afin de résoudre le problème du placement de capteurs pour cette application, nous avons développé une approche hiérarchisée qui permet d'appréhender des systèmes complexes.

Au terme de ces travaux, plusieurs axes de recherche se dégagent pour envisager, du point de vue des perspectives, de prolonger l'étude menée pendant cette thèse.

A court terme, quelques axes d'investigations sont à envisager :

- En ce concerne le placement de capteurs, nous n'avons pas pris en compte les défauts sur les capteurs. Nous pouvons traiter le cas où les défaillances des capteurs peuvent être pris en compte. Il faut investiguer cette problématique.
- les algorithmes concernant le placement de capteurs doivent être intégrés dans l'outil "**DXLAB**"
- les résultat doivent être valorisés notamment dans le cadre de l'application photovoltaïque que nous avons présenté dans ce mémoire.

A moyen terme, l'approche structurelle pour le placement de capteurs que nous avons proposée peut être prolongée afin qu'elle puisse traiter le cas où la notion de déductibilité des variables est prise en compte.

# Table des figures

# Liste des tableaux

# Bibliographie


Abella, M. and Chenlo, F. (2004). Choosing the right inverter for grid-connected pv system. *Renewable Energy World*.

Apt, K. (2003). *Principles of Constraint Programming*. Cambridge University Press.

Asokan, A. and Sivakumar, D. (2007). Model-based fault detection and diagnosis using structured residual approach in a multi-input multi-output system. *Serbian Journal of electrical Engineering*, 4 :133–145.

Berge, C. (1958). *Théorie des Graphes et ses applications*. B.G.Teubner.

Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. (2003). *Diagnosis and fault tolerant control*. Springer-Verlag.

Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. (2006). *Diagnosis and Fault-tolerant Control*. Springer-Verlag.

Bollobás, B. (1998). *Modern graph theory*. Graduate Texts in Mathematics. Springer, New York, U.S.A.

Borne, P., Dauphin Tanguy, G., Richard, J., Rotella, F., and Zambettakis, I. (1992). *Modélisation identification des processus*. TECHNIP.

Caliskan, F. and Hajiyev, C. (1998). Aircraft sensor fault diagnosis based on kalman filter innovation sequence. In *the 37 IEEE Conference on Decision and Control*, Tampa, Folrida, USA.

Carpentier, T., Litwak, R., and Cassar, P. (1997). Criteria for the evaluation of FDI systems-application to sensor location. In *IFAC Symp Fault Detection, Supervision and Safety for Technical Processes*, U.K.

Cassar, J. and Staroswiecki, M. (1997). A structural approach for the design of failure detection and identification systems. In *IFAC, IFIP,IMACS Conference on control of industrial systems*, pages 329–334, Belfort, France.

Cassar, J., Staroswiecki, M., and Cocquempot, V. (1995). Optimal residual design for model-based fault detection and isolation. In *3rd European Control Conference EC-CâĂŹ95*, Roma, Italy.

Chittaro, L. and Kumar, A. (1998). Reasoning about function and its applications to engineering. *Artificial Intelligence in Engineerings*, 12 :331–336.

Chittaro, L. and Ranon, R. (2004). Hierarchical model-based diagnosis based on structural abstraction. *Artificial Intelligence*, 155 :147–182.

Codd, E. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, 13 :377–387.

Commault, C. and Dion, J.-M. (2004). A system decomposition for failure detection and isolation. In *the 2nd IFAC Symposium on System, Structure and Control (SSSC'04)*, Oaxaca, Mexique.

Commault, C. and Dion, J.-M. (2007). Sensor location for diagnosis in linear system :a structural analysis. *IEEE Transactions on Automatic Control*, 52 :155–169.

Commault, C., Dion, J.-M., Sename, O., and Moteyian, R. (2002). Observer-based fault detection and isolation for structured systems. *IEEE Transactions on Automatic Control*, pages 2074–2079.

Commault, C., Dion, J.-M., and Yacoub Agha, S. (2005). A system decomposition for sensor location in fault detection and isolation. In *the 16th IFAC World Congress*, Prague, Czech Republic.

Commault, C., Dion, J.-M., and Yacoub Agha, S. (2006a). Location of additional sensors for FDI. In *4th Workshop on Advanced Control and Diagnosis*, Nancy, France.

Commault, C., Dion, J.-M., and Yacoub Agha, S. (2006b). Structural analysis for the sensor location problem in fault detection and isolation. In *SAFEPROCESS'2006*, Beijing, China.

Console, L., Picardi, C., and Ribando, M. (2000). Diagnosis and diagnosability analysis using process algebra. In *The 11th International Workshop on Principles of Diagnosis, DX'2000*, Morelia, Mexico.

Cordier, M.-O., Dague, P., Lévy, F., Dumas, M., Montmain, J., Staroswiecki, M., and Travé-Massuyès, L. (2000a). AI and automatic control approaches of model-based diagnosis : links and underlying. In *SAFEPROCESS'2000*, Budapest, Hungary.

Cordier, M.-O., Dague, P., Lévy, F., Dumas, M., Montmain, J., Staroswiecki, M., and Travé-Massuyès, L. (2000b). A comparative analysis of AI and control theory approaches to model-based diagnosis. In *14th European Conference on Artificial Intelligence*, Berlin, German.

Cordier, M.-O., Dague, P., Lévy, F., Montmain, J., Staroswiecki, M., and Travé-Massuyès, L. (2004). Conflicts versus analytical redundancy relations, a comparative analysis of the model-based diagnosis approach from the artificial intelligence and automatic control perspectives. *IEEE Transactions on Systems, Man, and Cybernetics*, pages 2163–2177.

Dague, P. (2001). *Diagnosis, intelligence artificielle et reconnaissance des formes*, chapter Théorie logique du diagnostic à base de modèles. Hermes, Paris.

De Kleer, J. (2003). Fundamentals of model-based diagnosis. In *the 14th International Workshop on Principles of Diagnosis, DX'03*, Washington, DC, USA.

De Kleer, J. and Brown, J. (1986). Theories of causal ordering. *Artificial intelligence*, 29 :33–62.

De Kleer, J., Mackworth, R., and Reiter, R. (1992). Characterizing diagnoses and systems. *Artificial Intelligence*, 56 :197–222.

De Kleer, J. and Williams, B. (1987). Diagnosis multiple faults. *Artificial intelligence*, 32(1) :97–130.

De Kleer, J. and Williams, B. (1992). Diagnosis with behavioral modes. *Readings in model-based diagnosis*, pages 124–130.

Declerck, P. and Staroswiecki, M. (1991). Characterization of the canonical components of a structural graph for fault detection in large scale industrial plant. In *European control conference*, Grenoble, France.

Delcroix, V., Piechowiak, S., and Rodriguez, J. (2002). Computing diagnosis with higher posterior probability using bayesian networks. In *International Conference on Information Processing and management of Uncertainty in Knowledge based-system*, pages 45–51.

Desinde, M. (2006). *Contribution à la mise au point d'une approche intégrée analyse diagnostique / analyse de risuqe*. PhD thesis, Institut National Polytechnique de Grenoble, France.

Dimitrova, E., Gadjeva, E., Van den Bossche, A., and Valchev, V. (2007). A model-based approach to automatic diagnosis using general purpose circuit simulators. In *IEEE ISIE 2006*, pages 2972–2977, Montreal, Quebec, Canada.

Dion, J.-M., Commault, C., and van der Woude. J (2003). Generic properties and control of linear structured systems : a survey. *Automatica*, 7 :1125–1144.

Dubuisson, B. (2001). *Diagnostic, Intelligence Artificielle et reconnaissance de formes*. Hermès Science.

Dulmage, A. and Mendelsohn, N. (1959). A structure theory of bi-partite graphs of finite exterior extension. *Transactions of the Royal Society of Canada*, 53(III) :1–13.

Dustegor, D., Cocquempot, V., and Staroswiecki, M. (2004). Structural analysis for residual generation : Towards implementation. In *the IEEE International Conference on Control Applications*, pages 1217–1222, Taipei, Taiwan.

Dustegor, D., Frisk, E., Cocquempot, V., Krysander, M., and Staroswiecki, M. (2006). Structural analysis of fault isolability in the DAMADICS benchmark. *Control Engineering Practice*, 14 :597–608.

Frank, P. (1996). Analytical and qualitative model-based fault diagnosis - a survey and some new results. *European Journal of Control*, 2 :6–28.

Frank, P. and Wunnenberg, J. (1989). *Fault diagnosis in dynamic systems-theory and applications*, chapter Robust fault diagnosis using unknown input observer schemes. Prentice Hall, London.

Frisk, E. (2000). Residual generator design for non-linear, polynomial systems-a grobner basis approach. In *SAFEPROCESS'2000*, Budapest, Hungary.

Frisk, E., Dustegor, D., Krysander, M., and Cocquempot, V. (2003). Improving fault isolability properties by structural analysis of faulty behavior models : application to the DAMADICS benchmark problem. In *SAFEPROCESS'03*, Washington, USA.

Frisk, E. and Krysander, M. (2007). Sensor placement for maximum fault isolability. In *the 18th International Workshop on Principles of Diagnosis, DX-07*, USA.

Gertler, J. (1987). *Fault detection and diagnosis in engineering systems*. Marcel Dekker.

Gertler, J. (1993). Analytical redundancy methods in fault detection and isolation. In *International Conference on fault diagnosis Tooldiag'93*.

hoblos, G., Staroswiecki, M., and Aitouche, A. (2000). Optimal design of fault tolerant sensor network. In *In proceeding of the 2000 IEEE*, Anchorage, Alaska, USA.

Isermann, R. (1993). Fault diagnosis of machines via parameter estimation and knowledge processing. *Automatica*, 29 :815–835.

Isermann, R. (1997). Supervision, fault detection and fault-diagnosis methods. *Control Engineering Practice*, 5(5) :639–652.

Isermann, R. (2004). Model-based fault detection and diagnosis-status and applications. In *16th symposium on Automatic Control in Aerospace (ACA'2004)*.

Iwasaki, Y. and Simon, H. (1986). Causality in device behaviour. *Artificial Intelligence*, 29(1) :3–33.

Izadi-Zamanabadi, R. (2002). Structural analysis approach to fault diagnosis with application to fixed-wing aircraft motion. In *the American Control Conference*, volume 5, pages 3949–3954, AK.

Izadi-Zamanabadi, R., Blanke, M., and Katebi, S. (2003). Cheap diagnosis using structural modelling and fuzzy-logic-based detection. *Control Engineering Prtaice*, pages 415–422.

Izadi-Zamanabadi, R. and Staroswiecki, M. (2000). A structural analysis method formulation for fault-tolerant control system design. In *the 39th IEEE, Conference on Decision and Control*, Sydney, Australia.

Khemliche, M., Ould Bouamama, B., and Haffaf, H. (2006). Sensor placement for component diagnosability using bond-graph. *Sensor and Actuators*, 132 :547–556.

Köing, D. (1989). *Theorie der endlichen und unendlichen Graphen*. Prentice Hall.

Krysander, M., Åslund, J., and Nyberg, M. . (2008). An efficient algorithm for finding minimal overconstrained subsystems for model-based-diagnosis. *IEEE Transactions on systems, Man, and Cybernetics-Part A : Systems and Humans*, 38(1) :197–206.

Krysander, M. and Nyberg, M. (2002). Structural analysis utilizing mss sets with application to a paper plant. In *the thirteenth International Workshop on Principles of Diagnosis*, Semmering, Austria.

Krysander, M. and Nyberg, M. (2005). An efficient algorithme for finding overconstrained sub-systems for construction of diagnostic tests. In *the 16th International Workshop on Principles of Diagnosis, DX-05*, California, USA.

Luong, M., Manquin, D., Huynh, C., and Ragot, J. (1994). Observability, redundancy, reliability and integrated design on measurement systems. In *IFAC SICICA'94*, budapest, Hungary.

Madron, F. and Veverka, V. (1992). Optimal selection of measuring points in complex plants by linear models. *AIChE journal*, 38(2) :227–236.

Maquin, D., Luong, M., and Ragot, J. (1997). Fault detection and isolation and sensor network design. *European Journal of Automation*, 31(2) :393–406.

Meinhardt, M. and Cramer, G. (2001). Multi-string-converter : the next step in evolution of string-converter technology. In *European Power Electronics Conference*, Austria.

Mishra, P. and Eich, M. (1992). Join processing in relational databases. *ACM Computing Surveys*, 24(1) :63–113.

Murota, K. (1987). *Systems analysis by graphs and matroids : structural solvability and controlability*. Springer-Verlag.

Myrzik, J. and Calais, M. (2003). String and module integrated inverters for single-phase grid connected photovoltaic system. *Power Tech Conference Proceedings, IEEE*.

Nyberg, M. and Frisk, E. (2006). Residual generation for fault diagnosis of systems described by linear differential-algebraic equations. *IEEE Transactions on Automatic Control*, 51 :1995–2000.

Nyberg, M. and Krysander, M. (2003). Combining AI, FDI, and statistical hypothesis-testing in a framework for diagnosis. In *SAFEPROCESS'2003*, Washington DC,USA.

Ould Bouamama, B., Samantaray, A., Medjaher, K., Staroswwieki, M., and Dauphin-Tanguy, G. (2005). Model builder using functional and bond graph tools for FDI design. *Control Engineering Practice*, 13 :875–891.

Patton, R. and Chen, J. (1991). A review of parity space approaches to fault diagnosis. In *SAFEPROCESS'1991*, Baden-Baden.

Paynter, H. (1961). *Analysis and design of engineering systems*. MIT Press.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems : network of plausible inference*. Morgan Kaufmann.

Ploix, S. and Adrot, O. (2006). Parity relations for linear uncertain dynamic systems. *Automatica*, 9 :1553–1562.

Ploix, S., Désinde, M., and Michau, F. (2004). Assessment and diagnosis for virtual reality training. In *CALLIE2004*, Grenoble, France.

Ploix, S., Desinde, M., and Touaf, S. (2005). Automatic design of detection tests in complex dynamic systems. In *The 16th IFAC World Congress*, Prague, Czech republic.

Ploix, S., Touaf, S., and Flaus, J. (2003). A logical framework for isolation in fault diagnosis. In *SAFEPROCESS'2003*, Washington DC,USA.

Ploix, S., Yassine, A., and Flaus, J. (2008). An improved algorithm for the design of testable subsystems. In *The 17th IFAC WORLD CONGRESS*, Seoul, Korea.

Pothen, A. and Chin-Ju, F. (1990). Computing the block triangular form of a sparse matrix. *ACM Transactions on Mathematical Software*, 16(4) :303–324.

Pulido, B. and Alonso, C. (2000). An alternative approach to dependency-recording engines in consistency-based diagnosis. In *9th International Conference, AIMSA 2000*, pages 115–126, Varna, Bulgaria.

Pulido, B. and Alonso, C. (2002). Possible conflicts, ARRs, and conflicts. In *the thirteenth International Workshop on Principles of Diagnosis, DX-2002*, pages 122–127, Semmering, Austria.

Pulido, B. and Alonso, C. (2004). Possible conflicts : a compilation technique for consistency-based diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics*, 34 :1083–4419.

Reiter, R. (1987). A theory of diagnosis from first principles. *Artificial Intelligence*, 32 :57–95.

Russell, S. and Norvig, P. (2003). *Artificial intelligence, a modern approach, 2nd ed.* Prentice Hall.

Samantaray, A., Medjaher, K., and Ould Bouamama, B. (2006). Diagnostic bond graphs for online fault detection and isolation. *Simulation Modelling Practice and Theory*, 14 :237–262.

Sen, S., Narasimhan, S., and Deb, K. (1998). Sensor network design of linear processes using genetic algorithms. *Computer and Chemical Engineering*, 22(3) :385–390.

Spanache, S., Escobet, T., and Travé-Massuyès, L. (2004). Sensor placement optimization using genetic algorithms. In *the 15th International Workshop on Principal of Diagnosis, DX'2004*.

Staroswiecki, M. (2002). *Fault diagnosis and fault tolerant control*, chapter Structural analysis for fault detection and isolation and for fault tolerant control. Encyclopedia of Lif Support Systems. Eolss Publishers, Oxford,UK.

Staroswiecki, M. (2003). On fault tolerant estimation in sensor networks. In *In proceeding of European Control Conference ECC'03*, Cambridge, UK.

Staroswiecki, M. and Comtet-varga, G. (2001). Analytical redundancy relations for fault detection and isolation in algebraic dynamic systems. *Automatica*, 37 :687–699.

Staroswiecki, M., hoblos, G., and Aitouche, A. (1999). Fault tolerant analysis of sensor system. In *In proceeding of the 38th Conference on Decision and control*, Phoenix, Arizona, USA.

Struss, P. (1992). What's in sd ? towards a theory of modeling for diagnosis. In *Readings in model-based diagnosis*. Morgan Kaufman.

Struss, P., Rehfus, B., Brignolo, R., Cascio, F., Console, L., Dague, P., Dubois, P., Dressler, O., and Millet, D. (2002). Model-based tools for the integration of design and diagnosis into a common process- a project report. In *the thirteenth International Workshop on Principles of Diagnosis, DX'02*, Semmering, Austria.

Thoma, J. (1975). *Introduction to Bond Graphs and Their Applications*. Pergamon Press, Oxford.

Travé-Massuyès, L., Dague, P., and Guerrin, F. (1997). *Le raisonnement qualitatif pour les sciences de l'ingénieur*. Edition Hermès.

Travé-Massuyès, L., Escobet, T., and Milne, R. (2001). Model-based diagnosability and sensor placement application to a frame 6 gas turbine subsystem. In *International joint Conference on Artificial intelligence IJCAI'01*, pages 551–556, USA.

Travé-Massuyès, L., Escobet, T., and Olive, X. (2006). Diagnosability analysis based on component supported analytical redundancy relations. *IEEE Transactions*, 36 :1146 – 1160.

Travé-Massuyès, L., Escobet, T., and Spanache, S. (2003). Diagnosability analysis based on component supported analytical redundancy relations. In *SAFEPROCESS'2003*, Washington D.C, U.S.A.

Vergé, M. and Jaume, D. (2004). *Modélisation structurée des systèmes avec les bond graphs*. TECHNIP.

Willsky, A. (1975). A survey of design methods for failure detection systems. *Automatica*, 12 :601–611.

Wuest, M., Toggweiler, P., and Riatsch, J. (1994). Single cell converter system (SCCS). *IEEE, First World Conference*, 1 :813–815.

Yassine, A., Ploix, S., and Flaus, J. (2007a). New results for sensor placement with diagnosability purpose. In *The 18th International Workshop on Principles of Diagnosis, DX-07*, USA.

Yassine, A., Ploix, S., and Flaus, J. (2007b). sensor placement and diagnosability. In *8th conference on diagnosis of processes and system, DPS07*, Poland.

Yassine, A., Ploix, S., and Flaus, J. (2008). Structural approach for sensor placement with diagnosticability purpose. In *The 17th IFAC WORLD CONGRESS*, Seoul, Korea.