N° d'Ordre : 3692

Université des Sciences et Technologies de Lille
Laboratoire d'Automatique, de Génie Informatique et Signal
LAGIS UMR CNRS 8146

THÈSE
Présentée en vue de l'obtention du grade de

DOCTEUR DE L'UNIVERSITÉ
Spécialité : Automatique et Informatique Industrielle

par
Dilek DÜŞTEGÖR

ASPECTS ALGORITHMIQUES DE L'ANALYSE STRUCTURELLE POUR LA SURVEILLANCE

Soutenue publiquement le 01/12/2005 devant la commission d'examen :

Présidente du Jury:	Geneviève DAUPHIN-TANGUY	Professeur, Ecole Centrale LILLE
Rapporteurs :	Jan LUNZE	Professeur, Ruhr-Universität Bochum
	Didier MAQUIN	Professeur, INPL Nancy
	Ron PATTON	Professeur, University of Hull
Co-directeurs :	Marcel STAROSWIECKI	Professeur, Ecole Polytech'Lille
	Vincent COCQUEMPOT	Maître de Conférences HDR, IUT A Université de Lille1

Avec la peur de l'artiste devant sa feuille vierge
et l'acte de création au bout de la plume

To myself

...à cette fillette de 12 ans
qui se demandait pourquoi...

Acknowledgements

This work has been carried out at *Laboratoire d'Automatique, Génie Informatique & Signal* formerly *Laboratoire d'Automatique et Informatique Industrielle de Lille*, Université des Sciences et Technologies de Lille, with professor Marcel Staroswiecki and Dr. Vincent Cocquempot as supervisors. I would like to thank them for leading me into the area of model based FDI, their support, and inspiring discussions. Special thanks to professor Jan Lunze, professor Didier Maquin, and professor Ron Patton to be my jury members, and kind regards to Professor Genevieve Dauphin-Tanguy to be the president of my jury. I would also like to thank all staff at the team *Sûreté de Fonctionnement des Systèmes Dynamiques* for creating a positive and friendly atmosphere.

I would like to thank the funding support under the EC FP5-RTN contract (HPRN-CT-2000-00110) through the DAMADICS Human Potential Research Training Network for the period of September 2001 to December 2002. I'm indebted to the *Computer Science Department* for offering me a lecturer position from February 2003 to August 2004.

My research colleague from Linköping university, Erik Frisk, is gratefully acknowledged for many insightful discussions during his postdoctoral stay at Lille, and for *boosting* me when I needed it the most. Other people who I have enjoyed fruitful discussions and priceless friendship with are Cyrille Christophe, Touria El-Mezyani, Mathieu Dulac and Malika Debuysschere.

Finally I would like to thank my family and A. Emin for their encouragement and support during the work.

Dilek Düştegör
December, 2005
Tallahassee - Florida

Contents

List of Tables

List of Figures

Chapter 1

Présentation générale

1.1 Introduction

Les systèmes automatisés de production sont caractérisés par une complexité croissante. Cette complexité les rend vulnérables aux défaillances, celles-ci étant à l'origine de coûts importants en termes de sécurité (risque d'accidents, de pollutions,...) et en termes de disponibilité (diminution de la productivité). Cette vulnérabilité justifie l'introduction de modules de supervision.

La supervision peut comporter de nombreuses fonctionnalités. De manière générale elle consiste à évaluer l'état d'une installation, à fournir ces informations à l'environnement (opérateurs humains en particulier) et à réagir (commande automatique par exemple) en fonction de cet état. Une des fonctions de la supervision est la surveillance. Celle-ci consiste à détecter et localiser les défaillances du système et quelquefois à en identifier le modèle.

La fonction de surveillance a pour premier objectif d'accroître la sécurité de l'installation. Ainsi, cette fonction a principalement été développée pour des systèmes critiques tels les installations chimiques, les centrales nucléaires, les plates-formes pétrolières et l'aéronautique.

Dans les phases initiales du cycle de vie d'un système technologique l'analyse des modes de défaillance et de leur criticité, permet d'avoir une idée des risques courus en exploitation. Un certain nombre d'indicateurs (fiabilité, disponibilité, sûreté, maintenabilité) en permettent une évaluation numérique. Si la sûreté de fonctionnement résulte bien sûr des choix de conception technologique (architecture matérielle, choix des composants, ...), une part significative peut reposer sur des algorithmes de surveillance qui détectent et localisent les défaillances, et sur des algorithmes de tolérance aux fautes, qui tentent de poursuivre ou restaurer un fonctionnement normal (ou dégradé) ou en tout cas qui visent à s'opposer à toute évolution catastrophique .

La surveillance (on dit aussi Diagnostic) regroupe l'ensemble des algorithmes de détection et de localisation des défaillances (Fault Detection and Isolation - FDI). La recherche en surveillance se développe depuis une trentaine d'année, essentiellement au sein des communautés Automatique, Traitement du Signal et Intelligence Artificielle. On trouvera de très bonnes introductions dans [71, 100, 42, 39, 65, 5].

Ce mémoire traite de l'analyse structurelle utilisée dans le contexte de la surveillance. L'analyse structurelle est un outil puissant qui permet de

déterminer de nombreuses propriétés intrinsèques d'un système dès la phase de conception. Ces propriétés sont obtenues à partir de la seule connaissance de l'existence de liens (contraintes) entre variables sans que les valeurs des paramètres soient nécessaires. Sur la base du modèle structurel, il est de plus possible de guider la recherche des indicateurs de défaillances.

1.2 Objectifs de la thèse - Problème considéré

L'analyse du modèle structurel, ou analyse structurelle, a été largement utilisée afin de déterminer les propriétés du système relatives à la détection et à la localisation des défaillances. Ces travaux ont donné lieu à de nombreuses publications: [28, 108, 11, 103, 83, 104, 23].

Ce mémoire traite le problème de l'automatisation de cette méthode. Plus précisément, nous nous intéressons aux aspects algorithmiques de l'analyse structurelle pour la surveillance afin d'améliorer l'implémentation de cette méthode et en assurer un meilleur transfert vers l'industrie.

En effet, bien que l'analyse structurelle pour la surveillance ait atteint une certaine maturité sur le plan théorique et ait montré son efficacité sur diverses applications, les aspects algorithmiques de la méthode n'ont été jusqu'à présent que très peu considérés. L'analyse structurelle est, dans le principe, particulièrement bien adaptée aux systèmes complexes, c'est-à-dire constitués de nombreux composants interconnectés, puisqu'elle ne s'intéresse qu'aux informations sur la structure du système. Cependant, lorsque des

informations structurelles nombreuses doivent être manipulées il est nécessaire de guider efficacement l'analyse sous peine de perdre tous les bénéfices de cette approche. De plus, les installations industrielles ne sont pas complètement figées. Elles évoluent au cours du cycle de vie en fonction des progrès technologiques, des opérations de maintenance, des objectifs de production. La méthode doit donc être facilement utilisable, rapide, efficace et adaptative.

1.3 Plan du mémoire

Le mémoire est structuré en quatre parties:

La première partie est un rappel des méthodes de surveillance en général et de l'analyse structurelle pour la surveillance. Cette partie permet d'établir le contexte de nos recherches, ainsi que la problématique. Cette partie ne comporte pas de résultats originaux mais synthétise l'ensemble des travaux sur l'analyse structurelle et permet de présenter l'outil, le vocabulaire, les techniques qui seront utilisées dans les parties suivantes

La partie II regroupe les résultats originaux de nos travaux, c'est-à-dire les différents algorithmes qui ont été élaborés.

La troisième partie est une illustration des algorithmes présentés sur une application industrielle. Cette étude a été réalisée dans le cadre du projet Européen DAMADICS[1] .

[1] *EC FP5 Research Training Network: Development and Application of Methods for*

Enfin, la dernière partie présente une conclusion suivie des perspectives de recherche. Les détails de chaque chapitre sont les suivants:

Partie I : L'analyse structurelle pour la surveillance

Chapitre 2 : **Préliminaires :** ce chapitre est une introduction générale à la surveillance des systèmes. Après avoir rappelé en quoi consiste la détection et la localisation de défaillances, les différentes techniques qui permettent de la mettre en oeuvre sont présentées.

Chapitre 3 : **L'état de l'art :** après un court rappel de l'évolution historique de la méthode, les différentes approches de l'analyse structurelle pour la surveillance sont présentées dans ce chapitre. Cela nous permettra de situer les différentes approches, de mettre en évidence leurs différences et similarités. L'analyse structurelle à partir des graphes bipartis est décrite en détail.

Cette partie, se basant sur le chapitre "Structural Analysis" de l'ouvrage "Fault Diagnosis and Fault Tolerant Control" [11], ne comporte aucun résultat nouveau. Cette partie a pour objectif de définir le vocabulaire et les notions d'analyse structurelle utilisées dans les chapitres suivants et qui constituent l'apport de cette thèse. L'apport de la thèse concerne seulement les aspects implémentation et algorithmiques qui suivront dans les parties II et III.

Actuator Diagnosis in Industrial Control Systems, 1 juillet 2000 - 30 juin 2003.

Partie II : Les aspects algorithmiques de l'analyse structurelle

Chapitre 4 : **Les propriétés d'un algorithme d'analyse structurelle pour la surveillance :** ce chapitre est consacré aux propriétés désirées pour un algorithme structurel pour la surveillance. Cet ensemble de propriétés nous permettra d'évaluer les algorithmes proposés dans la suite du mémoire.

Chapitre 5 : **La surveillabilité structurelle :** Nous proposons dans ce chapitre deux algorithmes permettant d'implémenter l'analyse de la structure d'un système afin d'évaluer ses propriétés de détection et de localisation des défaillances. La complétude et la correction des algorithmes seront prouvées et leur complexité sera étudiée. Ces resultats on été présentés au second symposium international IFAC "System, Structure and Control" [34].

Chapitre 6 : **Comment améliorer la localisabilité des défaillances ?** Ce chapitre est consacré à l'amélioration de la localisabilité des défaillances d'un système en se basant sur une analyse structurelle. Lorsque la localisabilité d'un système ne satisfait pas le cahier des charges, une solution possible est de rajouter des capteurs. Une autre possibilité consiste à utiliser des modèles de comportement défaillants sous certaines hypothèses de défaillances. Nous proposons dans ce chapitre d'utiliser la représentation structurelle pour décider sur quelles défaillances un effort de modélisation doit

être réalisé. Cette étude a été menée en collaboration avec Erik Frisk de l'université de Linköping (Suède) dans le cadre du projet européen DAMADICS. Les résultats obtenus ont été présentés à la conférence internationale IFAC/Safeprocess'2003 [51], puis aux Journées Doctorales d'Automatique 2003 [33]. Enfin, une version plus complète a été publiée dans le Journal Européen des Systèmes Automatisés [37].

Chapitre 7 : **La génération de résidus :** Le problème de génération de résidus est considéré dans ce chapitre. Un algorithme qui permet de générer la chaîne de calcul optimale par rapport aux contraintes pratiques du système est proposé. Pour cela, le problème du mariage stable connu en informatique ainsi que le problème du choix de couplage maximal à pondération minimale sont adaptés au contexte de l'analyse structurelle. Le premier algorithme a été présenté à la conférence internationale IEEE/Conference on Control Applications 2004 [36]. Le second a été présenté à la conférence internationale IEE/Control'04 [35].

Chapitre 8 : **Un algorithme adaptatif:** Dans ce chapitre, un algorithme adaptatif est proposé avec pour objectif de ne pas reprendre l'analyse à zéro en cas d'évolution du système lors de son cycle de vie. Cette évolution se traduit la plupart du temps par un ajout de contraintes. Les résultats qui y sont démontrés sont nouvaux.

Partie III : Application industrielle

Chapitre 9 : **La vanne DAMADICS :** Dans ce chapitre les algorithmes élaborés tout au long du mémoire seront appliqués à un modèle de vanne constituant le *benchmark* du réseau européen de recherche: DAMADICS[2]. Les résultats obtenus sont en cours de parution et vont être publiés dans le numéro spécial DAMADICS de la revue Control Engineering Practice [38]

Partie IV : Conclusions et perspectives

Chapitre 10 : **Conclusions :** Nos contributions au problème considéré seront résumées. Nous parlerons enfin de certaines perspectives de recherches intéressantes.

1.4 Notes sur les chapitres suivants

Pour chaque chapitre, un résumé assez détaillé en Français présente les problèmes qui sont abordés dans le chapitre et donne les principaux résultats qui y sont démontrés. Le reste du chapitre est en Anglais.

[2] *http://diag.mchtr.pw.edu.pl/damadics/*

Part I

Structural Analysis for FDI

Introduction to Part I

Technological systems are vulnerable to faults. Actuator faults reduce the performance of control systems and may even cause a complete break-down of the system. Erroneous sensor readings are the reason for operating points that are far from the optimal ones. Wear reduces the efficiency and quality of a production line. In most fault situations, the system operation has to be stopped to avoid damage to machinery and humans.

As a consequence, the detection and the handling of faults play an increasing role in modern technology, where many highly automated components interact in a complex way and where a fault in a single component may cause the malfunction of the whole system. Due to the simultaneously increasing economic demands and the numerous ecological and safety restrictions to be met, high dependability of technological systems has become a dominant goal in industry in the recent years.

To meet the increasing demand for safer and more reliable dynamic systems, early detection of faults using Fault Detection and Isolation (FDI) procedures is mandatory. In model-based FDI approaches, mathematical models are taken as the basis for diagnostic algorithms. Even when the system to diagnose is a well-known industrial plant, model building will require a major effort. Therefore, there is a recognized need for simple but efficient methods for overall analysis, before going to any detailed diagnostic algorithm design.

Structural analysis enables the evaluation of models with respect to fault

detectability and fault isolability properties by means of graph-based tools. Structural analysis has also proved useful in reconfigurability analysis for fault tolerant control. If the obtained isolability properties are not satisfactory, some ways to improve them by extra sensor placement can be proposed.

Despite extensive studies in this area, few results concern algorithmic issues. The major contribution of that thesis is to propose some algorithms that are easy to implement in order to automate the process. The necessity of automating structural analysis for FDI becomes more obvious when very large scale systems are considered. The design of algorithms that consider real-life practical constraints and that can be efficiently implemented will narrow the actual gap existing among the theory and practice of structural analysis for FDI.

Part I starts in chapter 2 giving an introductory background to model-based diagnosis. Fundamental concepts that will be used in this dissertation are introduced.

In chapter 3, structural analysis for FDI is recalled. For this purpose, the different approaches are presented with a special emphasis on the bipartite graph approach. Note that the bipartite graph-based structural analysis related sections are based on the corresponding chapter from the chapter "Structural Analysis" in the book "Fault Diagnosis and Fault Tolerant Control" [11]. Nothing new is presented in this part that aims to establish the current context of structural analysis for FDI.

The original contributions of our study are presented in parts II and III

that are devoted to implementation and algorithmic issues. The last section of this chapter explains the problem addressed in this thesis showing the necessity of such a research.

Chapter 2

Background information

Introduction Générale:

Dans ce premier chapitre, nous rappelons les concepts et terminologies liées au thème de la surveillance. Nous restons très général dans les notions abordées dont le but est d'établir les bases nécessaires pour comprendre le reste de ce mémoire. Le lecteur intéressé par plus de détails pourra se référer aux articles de synthèses [58, 48, 59, 49, 64] ou livres [100, 18, 101].

Après une brève introduction (section 2.3), nous abordons les approches qui s'intéressent à la détection et à la localisation des défaillances dans un système, plus particulièrement aux méthodes utilisant un modèle comportemental puisque notre travail s'inscrit dans ce cadre (section 2.4).

Le problème de génération d'indicateurs de défaillances (ou résidus) est présenté au paragraphe 2.5. Les résidus ainsi obtenus sont utilisés pour

détecter les défaillances (section 2.6), c'est-à-dire, déterminer si le système est en mode de fonctionnement normal ou défaillant. En cas de défaillances, l'étape suivante consiste à localiser les défaillances, c'est-à-dire à remonter aux composants (au sens large, il peut s'agir d'un ensemble de composants) défaillants. Cette étape est abordée au paragraphe 2.7.

2.1 Outline of the chapter

The aim of this chapter is not to explain in detail Fault Detection and Isolation (FDI) concepts and techniques. There are excellent survey papers [58, 48, 59, 49, 64] or books [100, 18, 101] written on this subject. Rather, our aim is to introduce the general principles and to define the related vocabulary (section 2.2) in order to position our research and innovative contributions in this field.

After a short introduction (section 2.3), model-based FDI methods are discussed in section 2.4. The residual generation problem is then discussed in section 2.5. Once residuals are generated, they are used in order to detect faults, that is to determine if the system is in normal operation mode or a faulty one (section 2.6). In case of faults, the next step is to isolate them, that is to say to determine the sources of the failure. This step is discussed in section 2.7.

2.2 Concepts and terminology

Being at the intersection of many fields such as industrial engineering, electrical and electronic engineering, computer science, different terminology and definitions are used in FDI.

As a step towards a unified terminology, the IFAC Technical Committee SAFEPROCESS has suggested preliminary definitions of some terms in the field of Fault Detection and Isolation. Some of these definitions are given

here:

- **Fault:** Unpermitted deviation of at least one characteristic property or variable of the system from acceptable/usual/standard behavior.
- **Failure:** Permanent interruption of a system's ability to perform a required function under specified operating conditions.
- **Fault Detection:** Determination of faults present in a system and time of detection.
- **Fault Isolation:** Determination of kind, location, and time of detection of a fault. Follows fault detection.
- **Fault Identification:** Determination of the size and time-variant behavior of a fault. Follows fault isolation.
- **Diagnosis:** Determination of kind, size, location, and time of detection of a fault. Follows fault detection. Includes fault isolation and identification.

Note that *perturbation* and *fault* are two different concepts not to be confused. In fact, perturbations are due to modelling errors, uncertainties or noise, and they might exist even when the system is working properly, whereas faults are due to some kind of malfunction in the system.

The sequel of this chapter is devoted to present the different general scheme of faut detection and identification methods.

2.3 Fault detection and identification

Fault detection and identification (FDI) has grown as an active research topic for more than twenty years. FDI algorithm development has been studied by both the Automatic Control and the Artificial Intelligence communities. Very good introductory information can be found in [71, 100, 42, 65].

Different points of view can be chosen to develop FDI algorithms, according to the level of knowledge one has about the system to be monitored. The two main points of view are related with model-based and non model-based techniques.

1. In non model-based techniques, past experimental records are analyzed in order to detect regularities which would link the observed data (the symptoms) with the final conclusions (the diagnosis).

2. Model-based techniques use some knowledge about the system normal behavior and analyze the data issued from the system to detect discrepancies between what actually is and what normally should be.

Both model and non model-based techniques lie on the redundant information inherent to the system. This redundancy can either be obtained through data collected during the system's previous runs in the normal mode, or through the a priori knowledge of the system (analytical model...). In other words, the redundancy may come from its inputs/outputs or its model.

The main idea of model-based FDI algorithms is to compare the real data obtained from the system with the expectations based on the system normal

behavior model (for the detection), or with the faulty behavior model (for isolation and diagnostic).

Our studies are mainly concerned with model-based methods, therefore only this approach is discussed in the next section.

2.4 Model-based methods

Model-based FDI method [94, 78, 21, 118] makes use of the system redundancy in order to check for inconsistency. There are two types of redundancies, hardware redundancy and analytical redundancy. The former requires redundant sensors, actuators, processors and software that enable to measure and/or control certain system variables. It has been utilized in the control of such safety-critical systems as aircraft space vehicles and nuclear power plants. However, its applicability is limited due to the extra cost and additional space required.

On the other hand, analytical redundancy (also termed functional, inherent or artificial redundancy) is achieved from the functional dependence among the process variables and is usually provided by a set of algebraic or temporal relationships among the states, inputs and the outputs of the system. These relationships are called analytical redundancy relations.

According to how the redundancy is accomplished, analytical redundancy can be further classified into two categories, direct and temporal [6, 19, 48]. A direct redundancy is accomplished from algebraic relationships among dif-

ferent sensor measurements. Such relationships are useful in computing the value of a sensor measurement from measurements of other sensors. The computed value is then compared with the measured value from that sensor. A discrepancy indicates that a sensor fault may have occurred. A temporal redundancy is obtained from differential or difference relationships among different sensor outputs and actuator inputs. With process input and output data, temporal redundancy is useful for sensor and actuator fault detection.

The whole process can roughly be decomposed in two phases: the detection phase (section 2.6), and the isolation phase (section 2.7). Figure 2.1 depicts the FDI process. First, the residual generation process is discussed in section 2.5.

2.5 Residual generation methods

Depending on the way the available knowledge is used, there are three main approaches to generate residuals.

1- **Parameter estimation based methods :** These kind of methods are based on the on-line identification of system parameters which are not measurable directly. Then, the estimated value is compared to the parameters nominal values. The estimation error is used as a residual. Interested readers are referred to [71, 72] and [31].

2- **Observer-based or filter-based approaches :** Observer-based or filter-based approaches are the more often used. The first studies in

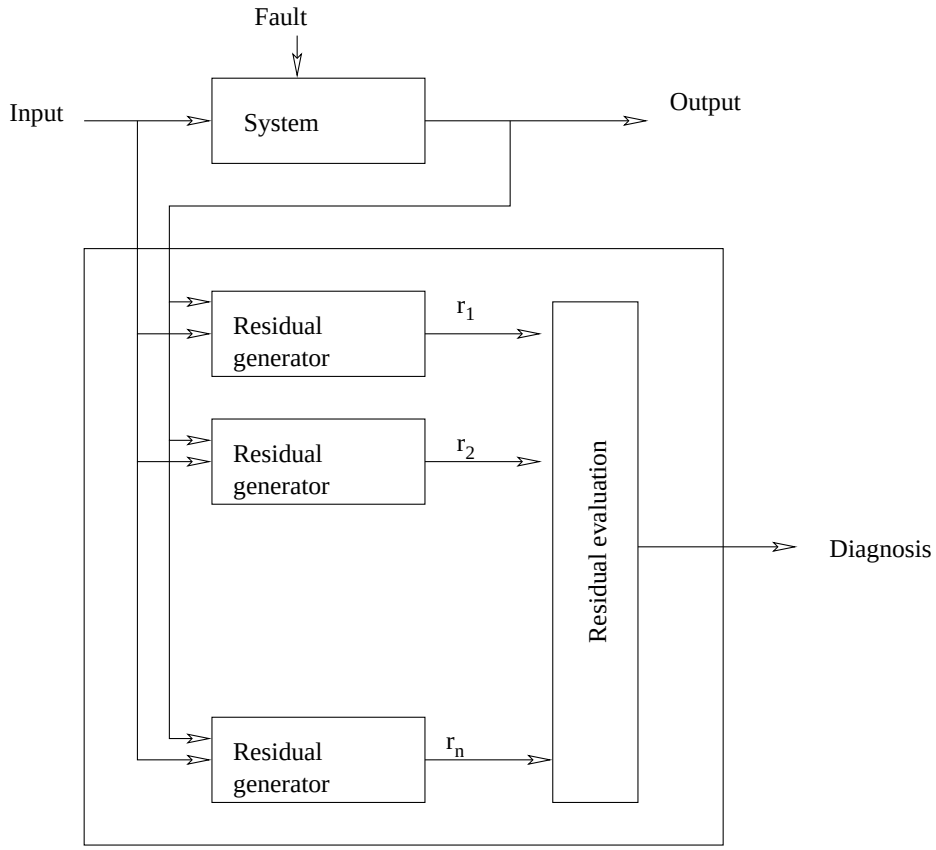

Figure 2.1: FDI general scheme

this domain began in the 70's [20]. These tools have been adapted for diagnostic purposes and there are many related studies [117, 50, 91, 79]. The main idea consists in comparing the estimated output functions with the measured output functions. The difference is used as a residual.

3- **Parity space based approaches :** Parity equations (also called Analytical Redundancy Relations or ARR) are rearranged and usually transformed variants of the input/output or state/space models of the plant [57, 59]. The essence is to check the parity (consistency) of the plant models with sensor outputs (measurements) and known process inputs. The off-line ARR generation problem can be stated as a general problem of variables elimination in a DAE system [19].

Once ARR are designed, the FD (Fault Detection) procedure checks at each time whether they are satisfied or not, and when not, the FI (Fault Isolation) procedure identifies the system component(s) which is (are) to be suspected. The existence of ARR is thus a prerequisite to the design of FDI procedures.

2.6 Fault detection

Fault detection procedure aims to determine if a fault occurs and when. This step only requires the normal behavior model. A fault is said to be detectable if and only if there is at least one residual that is sensitive to it [7].

In the perfect case (no modelling error, no unknown signals affecting the system,....), residuals equal zero when the system is running in normal operating conditions and are different from zero when a fault occurs. The fault detection process resumes then to raise an alarm when at least one of the residual becomes different from zero. In practical situation models are built based on hypothesis and are not perfect. As a consequence, residuals are not zero even in the normal situation and a decision procedure must be considered. The detection procedure efficiency depends on this decision procedure, but also on the "quality" of the used residuals. In order to minimize false alarms rate or missed-detection cases, residuals have to be optimized. In other words, they have to be the most sensitive to faults, and the less sensitive to perturbations or modelling errors. This is called robustness.

Robust residuals are insensitive to unknown inputs and unknown or uncertain parameters. Therefore, they are satisfied when no fault is present, whatever the value of the unknown inputs or uncertain parameters. When uncertain parameters are present, it may be wished not to use them in any ARR, because such ARR might generate false alarms or missed detections. The solution is simply to design the FDI system considering them as unknown variables (this boils down to use the subset of residuals in which no uncertain parameter intervenes).

2.7 Fault isolation

Once the detection phase detects a fault occurrence, a localization process is used in order to determine its origin. The localization process is based on the use of the set of residuals.

Two methods can be distinguished:

- Structured residuals [59].
- Directional residuals [56, 8].

2.7.1 Structured residuals

Definition 2.7.1. Let r_i be a residual, its structure with respect to a set of faults $\{\varphi\}$ of dimension η_φ is defined to be the binary word $S_{r,i}$ of size η_φ such that

- $S_{i,j} = 1$ if r_i is affected by the j^{th} element of $\{\varphi\}$.
- $S_{i,j} = 0$ if r_i is not affected by the j^{th} element of $\{\varphi\}$.

Structured residuals are conceived such that each residual is affected by a subset of faults φ_1 and is robust with respect to (not affected by) the remaining faults φ_2 (fig. 2.2). Thus, when a fault occurs, only a subset of residuals is different from zero.

Definition 2.7.2. A residual is said to be structured with respect to a fault vector φ_1 if it is only affected by faults from φ_1 and is robust with respect to other faults.

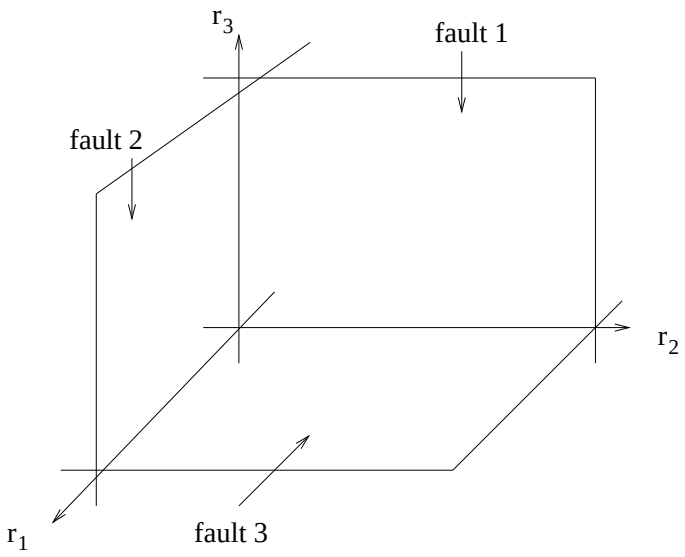

Figure 2.2: Structured residuals

Desired fault sensitivity and robustness information is put in a binary table called the *theoretical signature table*. When the i^{th} residual has to be affected by (respectively robust with respect to) the j^{th} fault, the corresponding entry (i^{th} line, j^{th} column) is 0 (respectively 1).

The detection procedure applied to each residual enables the generation of the real signature of each residual at a certain time. If this signature is zero, then none of the residuals are able to detect any faults. The system is then considered as healthy.

If the real signature is different from zero, the isolation process consists in matching the real signature in the theoretical signature table.

Definition 2.7.3. A fault may be structurally localized if all columns from

the theoretical signature table are different from each other.

The theoretical signature table has to be proposed such that the corresponding structured residual vector is computable and the isolation property is the most interesting, with respect to the isolability requirements definition.

2.7.2 Directional residuals

Directional residuals are built such that the residual vector points a certain direction in the residual space when a certain fault occurs (2.3).

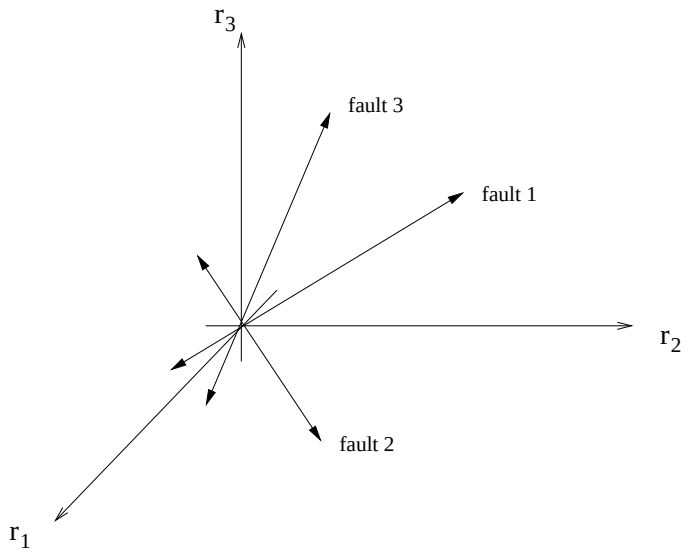

Figure 2.3: Directional residuals

The directional residuals' vector $\overrightarrow{r}(t)$, becomes as follows, when a fault $f_i(t)(i = 1, ..., \eta_f)$ occurs :

$$\overrightarrow{r}(t) = \alpha_i(t)\,\overrightarrow{l_i} \text{ with } i \in \{1, 2, .., \eta\} \tag{2.7.1}$$

with $\overrightarrow{l_i}$, a constant vector called the directional signature of fault i in the residuals space and α_i, a scalar function that depends on the faults's size and dynamic. The fault localization task is then to determine the theoretical directional signature the closest to the actual signature obtained by the residual computation.

2.8 Conclusion

In this chapter, the model-based FDI related vocabulary has been defined. The basis of these methods have been explained without going into details. In the following chapter, structural analysis for FDI is introduced.

Chapter 3

Structural analysis for FDI: review

L'état de l'art de l'analyse structurelle pour la détection et la localisation des défaillances:

La plupart des méthodes de surveillance utilisent un modèle du système considéré. Surveiller consiste à vérifier la consistance des données prélevées en ligne sur l'installation avec ce modèle. Une catégorie d'approche utilise des modèles comportementaux obtenus soit à partir des lois de la physique régissant le comportement de chaque composant dans son environnement, soit en utilisant des techniques d'identification. Dans la pratique, pour les systèmes industriels complexes (dans le sens ici où le système est constitué de nombreux composants) on ne dispose souvent que de descriptions graphiques représentant un certain nombre de sous-systèmes plus ou moins complexes interconnectés. Chaque sous-système reliant des variables d'entrée

et des variables de sortie, est décrit par un modèle plus ou moins détaillé qui se présente sous des formes très diverses: fonctions de transfert, tables ou abaques, règles, équations de bilan,....

La modélisation structurelle repose sur une connaissance relativement pauvre du système puisqu'elle ne s'intéresse qu'à l'existence des liens entre variables sans s'occuper (au moins dans une première phase) de la nature de ces liens.

L'analyse structurelle s'intéresse aux propriétés du modèle structurel. Les propriétés obtenues sont donc valides quelle que soit la nature des équations et la valeur des paramètres. Cette approche s'avère particulièrement intéressante dans la phase de conception d'un système ou lorsqu'il s'agit de modifier ou d'étendre les fonctionnalités d'un système existant.

Les propriétés structurelles qui nous intéressent à des fins de détection et de localisation des défaillances sont:

- Les possibilités de détection (détectabilité) et de localisation (localisabilité) des défaillances. L'analyse de ces propriétés conduit à la détermination du sous-système surveillable avec l'instrumentation donnée,

- les possibilités de générer des résidus robustes et structurés.

Dans une première partie de ce chapitre, nous rappelons l'évolution historique de l'analyse structurelle pour la détection et la localisation des défaillances (paragraphe 3.2). Cette approche a été introduite dans les années 70 pour analyser certaines propriétés structurelles telles que la commandabil-

ité et l'observabilité structurelle des systèmes [86]. La mise en oeuvre de la méthode pour la détection et la localisation des défaillances remonte entre autres aux travaux de Declerck [28, 30]. On peut constater plusieurs *écoles* différentes qui ont enrichi ce domaine de recherche. On peut classifier ces différentes *écoles* par rapport au modèle adopté pour représenter la structure du système. Nous verrons en effet qu'il existe trois modèles pour représenter la structure d'un système:

1- Les bond-graphs (paragraphe 3.3.1): L'outil de modélisation bond-graph a été défini par Henry Paynter en 1961 [102]. C'est une représentation graphique des systèmes physiques. Un bond-graph, littéralement "graphe à liens", montre explicitement les transferts énergétiques entre les différentes parties du système. La modélisation par bond-graph est en plein essor dans le monde industriel, notamment dans les domaines de l'automobile et de l'énergie. La causalité représentée par cette modélisation est de nature physique.

2- Le deuxième modèle (paragraphe 3.3.2) est issu directement de la modélisation du système par ses équations d'état. De cette représentation, nous pouvons extraire trois matrices structurées. Celle qui met en évidence la structure des interactions des variables d'état entre elles, celle qui met en évidence la structure des liens entre les commandes et les variables d'état et celle qui met en évidence la structure des liens entre les variables d'état et les mesures (sorties). Le système est alors

représenté par un di-graphe qui a initialement une certaine orientation. Dans cette représentation, la causalité intégrale est implicitement représentée. Notons aussi que cette modélisation nécessite de connaître les équations d'état.

3- Le troisième modèle (paragraphe 3.3.2) est un modèle général qui considère un système comme un ensemble de variables reliées entre elles par un ensemble de relations que nous appelons "contraintes". Cette manière de modéliser un système, permet de le représenter par un graphe biparti, non orienté initialement. Une matrice d'adjacence (variables - contraintes) structurelle peut lui être associée. Puisque ce type de graphe est non orienté, il permet de représenter tout type de causalité. La représentation graphe biparti, plus générale, sera adoptée dans la suite de ce mémoire.

Après avoir introduit l'historique et les différents modèles structurels, les propriétés structurelles sont définies au paragraphe. Nous présentons ensuite les outils qui permettent d'étudier les propriétés structurelles relatives à la surveillance d'un système (paragraphe 3.4). Nous verrons ainsi les notions de décomposition canonique d'un système et de couplage dans un graphe biparti ainsi que l'interprétation en termes de causalité calculatoire qui oriente le graphe. Le problème de génération de résidus robustes et structurés est explicité (paragraphe 3.5). Comme indiqué dans l'introduction générale, ce chapitre ne comporte aucun résultat original et se base sur [11].

Nous abordons ensuite les aspects algorithmiques pour l'implémentation de la méthode (paragraphe 3.6). Même si les concepts de base de cette méthode sont relativement simples, des algorithmes performants doivent être utilisés si l'on ne veut pas perdre tous les avantages de cette approche. Or peu de travaux ont été effectués dans cette voie. La deuxième partie de ce mémoire apporte une contribution dans cette direction et constitue l'apport de cette thèse.

3.1 Outline of the Chapter

The design of fault detection and isolation (FDI) algorithms is based on numerical behavior models. In model-based approaches, detailed behavior models are seldom available in the first phases of system design, and/or are very expensive to develop, especially when complex processes, with hundreds of variables, are considered, and simpler models have to be used. In such situations, structural models provide an interesting approach to the system analysis, since they only need a very primitive level of knowledge about the system behavior.

The structural model of a system is an abstraction of its behavior model, in the sense that only the structure of the constraints, i.e. the existence of links between variables and parameters is considered, and not the constraints themselves. The links are represented by a graph, which is independent of the nature of the constraints and variables (quantitative, qualitative, equations, rules, etc.) and of the value of the parameters. This indeed represents a qualitative, very low level, easy to obtain, model of the system behavior. Structural analysis is concerned with the properties of the system structure model, which resorts to the analysis of its graph. This graph being independent on the value of the system parameters, structural properties are true almost everywhere in the system parameter space. Here, almost means that there may be some isolated points where the set of constraints is pathological, with the result that structural properties are not valid in these part of

the system parameter space.

In spite of their simplicity, structural models can provide many useful information for FDI, since structural analysis is able to identify those components of the system which are - or are not - monitorable and to provide design approaches for analytic redundancy based residuals.

This chapter investigates how the analysis of system's structural model can be used for FDI purposes. For this, the historical evolution of structural analysis for FDI is rapidly presented in section 3.2. Then the different structural models, namely the bond-graph, the digraph, and the bipartite graph approaches, are explained in section 3.3.

In section 3.4, the tools that are required to analyze the bipartite graph and to determine structural properties are presented. Thus, the canonical decomposition of the system structural graph is first introduced. Then, matchings on a bipartite graph are explained, and their interpretation is given, introducing the idea of causality which provides the bipartite graph with an orientation. Finally, the robust and structured residuals generation problem is addressed in section 3.5. As previously mentioned, these sections are extracted from [11, 1] and aim at establishing the current context of structural analysis for FDI.

Finally, the existing algorithms that implement the bipartite graph-based structural analysis approaches are emphasized (section 3.6). Indeed, even though the basic concepts of structural analysis are relatively simple, high-performance algorithms have to be implemented in order not to lose the

advantages of the method. This investigation of algorithmic issues shows that some efforts still have to be spent in order to improve the implementation. This problem is addressed in part II.

3.2 Historical background

Structural concepts were first used in the 60/70's for the decomposition of large systems of equations in view of their hierarchical resolution [111, 66], and for the analysis of system structural properties, like observability and controllability, where most studies use a digraph representation and address linear systems [86, 87, 60, 98].

Especially, Lin's work on structural controllability is a contribution to the dynamical system's theory. Further works on the subject [107, 60, 47] have clearly shown that some structure properties such as controllability or observability are mainly determined by the system's structure, independently of parameters' exact values.

Those studies have also been extended to the design of multi-variable control systems, including considerations like disturbance rejection for example [106, 105], with numerous applications in chemical engineering [55, 88, 96] since complex large scale systems are often encountered in that field. An important issue is the solvability of large scale differential and algebraic equations systems, whose structural analysis is addressed in [115, 85].

The approach has been adopted by the FDI community after the studies of

Declerck in late 90's [28, 30]. Structural concepts have then been used for the analysis of system monitorability [109, 29] and for the design of structured residuals [57], which provide straightforward decision procedures for fault isolation [27]. Good descriptions can be found in [108, 11, 110].

Since then, many different *Structural Analysis for FDI Schools* have evolved and enabled this research area to reach its maturity. These approaches can be categorized according to their modelling approaches. This is important because according to the representation chosen, it is possible to obtain more or less accurate information regarding the system. The three main representation are the *bond-graph* (section 3.3.1), the *digraph* (section 3.3.2) and the *bipartite graph* (section 3.3.2).

Bond-graphists have developed specific tools for structural analysis. Applications of bond-graph to fault diagnosis can be found in [112, 12, 13].

The digraph representation has also been used for FDI purposes [23, 32, 25, 26]. Lately, this representation has been applied to sensor location [24].

The bipartite graph representation is widely used in structural analysis for FDI purposes. Besides its wide application to monitorability analysis [16, 73, 77, 89, 81, 82], it has recently been applied to the problem of sensor selection [14, 15, 95, 93, 114, 99, 2], and in FTC, for the analysis of system reconfigurability [54, 67, 68, 1].

Among all, the following applications to FDI are the most significant:

- A power plant PWR 900 MW [28, 30].

- The "Ship propulsion Benchmark" analysis [22, 74, 90].
- A paper plant analysis [83].
- The RØmer satellite analysis [75].
- A steam generator analysis [12, 13]

3.3 The different structural models

In this section, the different approaches of structural analysis for FDI are categorized according to their structural model, namely bond-graph, digraph and bipartite graph.

3.3.1 Bond-graph

Bond-graph is an explicit graphical tool for capturing the common energy structure of systems. In 1959, Prof. H.M.Paynter introduced the idea of portraying systems in terms of power bonds, connecting the elements of the physical system to the so-called junction structures which were manifestations of the constraints [102]. This power exchange portray of a system is called bond-graph, which can be both power and information oriented. By this approach, a physical system can be represented by symbols and lines, identifying the power flow paths. The lumped parameter elements of resistance, capacitance and inertia are interconnected in an energy conserving way by bonds and junctions resulting in a network structure. The bond-graph

language attempts to express general classes of physical systems through power interactions. The factors of power i.e., Effort and Flow, have different interpretations in different physical domains. Yet, power can always be used as a generalized coordinate to model coupled systems residing in several energy domains. One such system may be an electrical motor driving a hydraulic pump or a thermal engine connected with a muffler, where the form of energy varies within the system. Applications of bond-graph to fault diagnosis can be found in [112, 12, 13].

3.3.2 Digraph and bipartite graph

A controlled system is a set of interconnected components interacting with each other to achieve the system's goal. Each components behavior can be described by a set of algebraic and/or dynamic equations defining the trajectory of a set of variables.

Digraph and bipartite graph-based structural analysis only deals with the structural information contained in the model, i.e. which variables appear in which equation. This is a completely qualitative model, which does not consider the actual numerical-analytical form of the equations. Therefore, structural properties are valid for all models with the same structure [11].

Consider, for example, state space models like

$$\dot{x}(t) = f(x(t), d(t), u(t), \theta) \tag{3.3.1}$$

$$y(t) = g(x(t), d(t), u(t), \theta) \tag{3.3.2}$$

where $x(t) \in R^n$ stand for unknown internal states, $d(t) \in R^l$ stand for disturbances and unknown inputs that should not influence the residual, $u(t) \in R^m$ and $y(t) \in R^p$ are respectively the system inputs and outputs, and $\theta \in R^q$ is a parameter vector. Here, f stands for the set of differential constraints

$$\dot{x}_i(t) - f_i(x(t), d(t), u(t), \theta) = 0, \quad i = 1, ..r$$

and g stands for the measurement constraints

$$y_j(t) - g_j(x(t), d(t), u(t), \theta) = 0, \quad j = 1, ..q$$

Digraph

A popular structural representation of the model (3.3.1), (3.3.2) uses a directed graph (digraph).

Definition 3.3.1 (*Digraph*, [11])**.** The digraph associated with system (3.3.1), (3.3.2) is a graph where:

- The set of the input, output and state variables are respectively represented by the vertices $u_1, u_2, ..., u_m$, $x_1, x_2, ..., x_n$ and $y_1, y_2, ..., y_p$.
- The edges in the graph are defined by the following rules:
  - an edge exists from vertex x_i (resp. from vertex u_j) to vertex x_k if and only if the state variable x_i (resp. the input variable u_j)

really occurs in function f_k (i.e. $\frac{\partial f_k}{\partial x_i}$ - resp. $\frac{\partial f_k}{\partial u_j}$ - is not identically zero),

– an edge exists from vertex x_i to vertex y_j if and only if the state variable x_i really occurs in the function g_j.

In the digraph representation, an edge from x_k (resp. from u_l) to x_i means that the time evolution of the derivative $\dot{x}_i(t)$ depends on the time evolution of $x_k(t)$ (resp. $u_l(t)$). Similarly, an edge from x_k to y_j means that the time evolution of the output $y_j(t)$ depends on the time evolution of the state variable $x_k(t)$.

Example 3.3.1 (Digraph representation). Consider the system represented by the following equations:

$$\begin{cases} x_1 &= f_1(x_3) \\ x_2 &= f_2(x_3) \\ x_3 &= f_3(x_1, x_2, x_3, u) \\ y &= g_1(x_2) \end{cases} \tag{3.3.3}$$

The corresponding structured matrices are:

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

and the digraph associated with the system is given by figure 3.1 where

$$x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}. \qquad \square$$

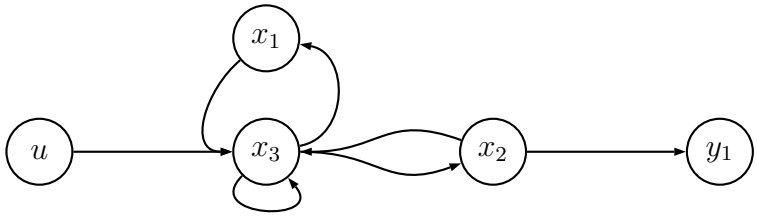

Figure 3.1: Example 3.3.1 : digraph representation

Applications to FDI of the digraph representation based structural analysis can be found in [23, 32, 25, 26], or in [24] for sensor location. In this kind of model, the integral causality among graph elements is implicitly represented.

Bipartite graph

A graph is bipartite if its vertices can be separated into two disjoint sets C and V in such a way that every edge has one endpoint in C and the other one in V.

Definition 3.3.2 (Bipartite graph)**.** The bipartite graph representing the system's structure is the graph $G = (V \bigcup C, \Gamma)$ where

$$V = \{x_1, x_2, ..., x_n, y_1, y_2, ...y_p, u_1, u_2, ..., u_m\}$$

is the set of nodes corresponding to the variables occurring in the system's constraints and

$$C = \{c_1, c_2, ..., c_s\}$$

is the set of nodes corresponding to the constraints,

$$\Gamma = \{(c_i, v_j) | v_j \text{ appears in } c_i, v_j \in V, i \in \mathbf{m}, j \in \mathbf{n}\}$$

is the set of edges. Here, $\mathbf{m} \equiv \{1, 2, ..., s\}$ and $\mathbf{n} \equiv \{1, 2, ..., n + p + m\}$. □

The corresponding adjacency matrix M of a bipartite graph G is a boolean matrix where rows correspond to C, columns to V and

$$M = \{m_{i,j} | m_{i,j} = 1 \text{ if } (c_i, v_j) \in \Gamma, 0 \text{ otherwise } \}$$

In a dynamic model, differentiated variables are used. That is to say, some of the variables (call them x_d) appear in the form $\dot{x}_d$ in the system representation. For simplicity purpose, algebraic variables (that are not differentiated in the system representation) are called x_a. There are at least three different ways to represent time differentiated variables in a structural model:

1. Consider x_d and $\dot{x}_d$ to be structurally the same variable and treat dynamic equations in the same way as static equations (see [30]).

2. Consider x_d and $\dot{x}_d$ to be structurally different. Two ways may be used to emphasize the relation between these two variables:

i) It is possible to define an extra set of variables $\dot{x}_d$ and an extra set of constraints

$$i = 1, ...n \quad \dot{x}_i(t) - \frac{d}{dt}x_i(t) = 0 \tag{3.3.4}$$

so that the system is

$$\begin{aligned} V &= x_a \cup x_d \cup \dot{x}_d \cup u \cup y \\ C &= f \cup g \cup \frac{d}{dt} \end{aligned} \tag{3.3.5}$$

where $\frac{d}{dt}$ stands for the differential constraints (3.3.4) and all the constraints f, and g are algebraic.

ii) Perform structural differentiations of the initial model and consider x_d and $\dot{x}_d$ (as well as x_a and $\dot{x}_a$ obtained from the performed structural differentiations) as completely independent variables (see [83]).

The first approach does not consider the dynamic aspect of the system. A dynamic system and a static system may have the same structural model. Indeed, consider the following first-order dynamic model:

$$c_1 : \dot{x}_1 = \alpha x_1 + x_2 + u \tag{3.3.6a}$$

$$c_2 : y = x_2 \tag{3.3.6b}$$

where x_1 and x_2 are unknown variables, u the control signal and y the measurement. The equation (3.3.6a) describes the internal behavior of the component, whereas (3.3.6b) describes the sensor.

	$\overline{x_1}$	x_2	y	u
c_1	1	1		1
c_2		1	1	

Table 3.1: Bi-adjacency matrix: approach 1

The structural representation of this component according to the first approach is depicted by table 3.1 and figure 3.2 ($\overline{x_1}$ stands for x_1 and $\dot{x}_1$).

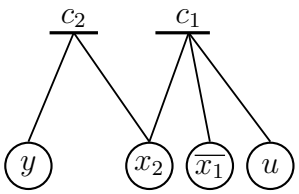

Figure 3.2: Bipartite graph: approach 1

The structural model of the following static system ((3.3.7a), (3.3.7b)) is exactly identical.

$$c_1 : x_1 = x_2 - u \tag{3.3.7a}$$

$$c_2 : y = x_2 \tag{3.3.7b}$$

When structural properties such as observability and controllability are investigated, the second approach seems to be preferable.

For instance, according to approach (2i), the differentiation relation (3.3.8) is added to the model.

$$c_3 : \dot{x}_1 = \frac{dx_1}{dt} \tag{3.3.8}$$

	x_1	$\dot{x}_1$	x_2	y	u
c_1	1	1	1		1
c_2			1	1	
c_3	1	1			

Table 3.2: Bi-adjacency matrix: approach 2i

	x_1	$\dot{x}_1$	x_2	y	u
c_1	1	1	1		1
c_2			1	1	

Table 3.3: Bi-adjacency matrix: approach 2ii

The dynamic aspect is explicitly considered thanks to $\dot{x_1}$. Table 3.2 and figure 3.3 show the obtained structural model.

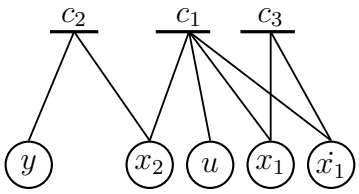

Figure 3.3: Bipartite graph: approach 2i

According to approach (2ii), the structure of the system is as shown in table 3.3. However, during the structural analysis of the system, c_1 and c_2 can be many times differentiated and added to the original system. Of course, this differentiation will add new variables x_1^i and x_2^j, with i and j being the ordered of differentiation of each constraint.

The modelling explained with the approach approach (2i) will be used in the sequel of this dissertation.

Faults representation. Analyzing the FDI possibilities of a system needs faults to be precisely defined. Generally speaking, a fault is defined as *a change in a subset of constraints*. In order to represent and study the structural sensitivity of faults in a set of constraints (i.e. a residual), structural model may be completed by considering the available information about faults. Three levels of knowledge may be identified:

1. A basic level is to only specify which component is influenced by the fault. No specific information on how and in which constraints the fault influences the process is included. A component is regarded as a subset of constraints.

2. The second level is more precise since it describes which equation(s) is not valid in case of a fault and how the nominal equations are affected by faults occurrence. For this purpose, variables describing faults ($f_i's$) should be added to the framework. Faults can be additive or multiplicative depending on how they affect the equations.

3. A third level includes a model of the fault signal. This could for example be that the fault size is slowly varying and can be assumed constant, or that the fault size is highly correlated with some other signals in the model.

	x_1	$\dot{x}_1$	x_2	f_1	f_2	y	u
c_1	1	1	1	1			1
c_2			1		1	1	
c_3	1	1					

Table 3.4: Bi-adjacency matrix: second level of knowledge

To exemplify the above, consider model ((3.3.6a), (3.3.6b)) and suppose it is subjected to two faults: an internal fault and a sensor fault.

The first level of knowledge is a direct consequence of faults description. The internal fault invalidates constraint (3.3.6a), while the sensor fault invalidates constraint (3.3.6b).

The second level explains how the two constraints are modified in case of faults. For example, the internal fault can be described by a change in parameter α, the sensor fault by a gap between x_2 and y. This second level of knowledge requires two additional variables f_1 and f_2 representing the faults. The analytical model becomes:

$$c_1 : \dot{x}_1 = -(\alpha + f_1)x_1 + x_2 + u \tag{3.3.9a}$$

$$c_2 : y = x_2 + f_2 \tag{3.3.9b}$$

The structural model becomes as shown in table 3.4.

In order to exemplify the third level of knowledge, assume that f_1 is known to be constant. The following equation is added to the model:

$$c_4 : \dot{f}_1 = 0 \tag{3.3.10}$$

	x_1	$\dot{x}_1$	x_2	f_1	$\dot{f}_1$	f_2	y	u
c_1	1	1	1	1				1
c_2			1			1	1	
c_3	1	1						
c_4					1			
c_5				1	1			

Table 3.5: Bi-adjacency matrix: third level of knowledge

The differentiation equation $c_5 : \dot{f}_1 = \frac{df_1}{dt}$ is also added to the structural model (table 3.5, figure 3.4).

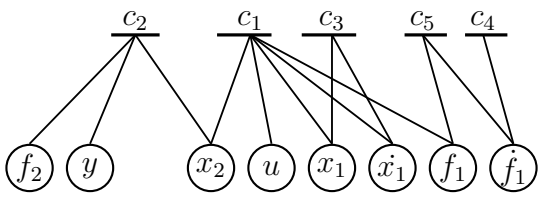

Figure 3.4: Bipartite graph with fault model

3.3.3 Which representation

The bond-graph representation is merely based on power, the causality among different components is physical. Meanwhile, both digraph and bipartite graph are an abstraction of the behavior model, because they merely describes which variables are connected by which constraints, but they do not say how these constraints look like. Hence, the structural model presents the

basic features and properties of a system.

The bipartite graph representation is the most widely used in structural analysis for FDI purposes. This is mainly due to the fact that it is a neutral representation in terms of causality. In other words, neither physical relationship (as in bond-graph) nor integral causality (as in digraph) is imposed to the system representation.

In the sequel, bipartite graphs will be used for the representation of the system structure.

3.4 Tools for structural analysis

The structural model of a system is an abstraction of its behavior model. Two systems which have the same structure are said to be structurally equivalent. Since structural properties are properties of the structural graph, they are obviously shared by all systems which have the same structure. In particular, systems which only differ by the value of their parameters are structurally equivalent, thus making structural properties independent of the values of the system parameters.

Of course, actual system properties may differ from structural ones. Actual properties are only potential when structural properties are satisfied. They can certainly not be true when structural properties are not satisfied. In other words, structural properties are necessary but not sufficient properties for the actual system.

Structural analysis makes use of the system structure (section 3.3.2) in order to investigate these structural properties. For this purpose, some specific tools are used and they are defined in this section.

3.4.1 Matching

The basic tool for structural analysis is the concept of matching on a bipartite graph, which is introduced in this section. A matching is a causal assignment which associates some system variables (that are considered to be unknown) with the system constraints from which they can be calculated. A matching $\mathcal{M}$ is then a subset of G such that no two edges $\gamma_i, \gamma_j \in \Gamma$ in $\mathcal{M}$ have common vertices.

The set $\mathbb{M}$ of all matchings is a subset of 2^{Γ}, which is partially ordered by the set-inclusion order relation. Thus, maximal elements can be defined.

Definition 3.4.1 (maximal matching, [29, 30])**.** A maximal matching is a matching $\mathcal{M}$ such that $\forall \mathcal{N} \in 2^{\Gamma}$ with $\mathcal{M} \subset \mathcal{N}$, $\mathcal{N}$ is not a matching.

Definition 3.4.2 (Complete matching, [29, 30])**.** Given a matching $\mathcal{M}$ between $C' \subset C$ and $V' \subset V \backslash \{u_1, ..., y_1, ...\}$. $\mathcal{M}$ is called complete with respect to C' if $|\mathcal{M}| = |C'|$ holds (resp. a matching is called complete with respect to V' if $|\mathcal{M}| = |V'|$ holds).

3.4.2 Oriented graph associated with a matching

The causality depicted by a matching introduces some orientations of the edges Γ. Constraints which appear in the initial graph have no direction, because all variables have the same status. For example, in the illustrative system 3.3.6a, the constraint

$$c_1 : \dot{x}_1 = \alpha x_1 + x_2 + u \tag{3.4.1}$$

can be used to compute any of the two variables x_1 or x_2 whenever the other variable is known. The non-oriented form emphasizes that the constraint itself has no preference for any of the three variables.

Once a matching is chosen, this symmetry is broken, since each matched constraint is now associated with one matched variable and some non-matched ones. Consider a matching $\mathcal{M}$ and choose an edge $(c, x) \in \mathcal{M}$. The variable x can be considered as the output of the constraint c while the other variables in c are the inputs. In other words, the matching represents some causality assignment by which the constraint c is used to compute the variable x assuming the other variables are known. This computational causality is depicted by orienting all the edges $(c_i, v_j) \in \Gamma$ in the graph G in the following way:

- if c_i and v_j are matched, $c_i \rightarrow v_j$,

- otherwise $v_j \rightarrow c_i$.

3.4.3 Matching: causal interpretation

The aim of this subsection is to discuss the causal interpretation of the oriented bipartite graph associated with a matching.

Indeed, selecting a pair (c, x) to belong to a matching, implies a causality assignment, by which the constraint c is used to compute the variable v, assuming the other variables are known. However, all (c, x) may not fulfill this implication. An obvious situation in which (c, x) cannot be matched is when c is not invertible with respect to x. In this case, in order to graphically represent that such a matching is *forbidden*, the symbol (Δ) is used to represent the non-calculable variables in the adjacency matrix.

As explained previously, in case of differential constraints, differentiation constraints as shown in eq. (3.4.2) are added to the system.

$$d : x_2(t) - \frac{d}{dt}x_1(t) = 0 \tag{3.4.2}$$

Here, $x_1(t)$ and $x_2(t)$ cannot be chosen independently of each other. Obviously, when the trajectory $x_1(t)$ is known, its derivative can always be computed. It follows that the constraint can always be matched for x_2 which is then uniquely defined. This is called *derivative causality.* When $x_2(t)$ is known, matching this constraint for x_1 is called *integral causality* and it does not determine $x_1(t)$ uniquely, unless the initial condition $x_1(0)$ is known. Derivative causality can be forced, when necessary, by using the same symbol (Δ) in the structure matrix in order to forbid integral causality.

	x_1	$\dot{x_1}$	x_2	y	u
c_1	(1)	1	1		1
c_2			(1)	1	
c_3	Δ	(1)			

Table 3.6: Maximal and complete matching

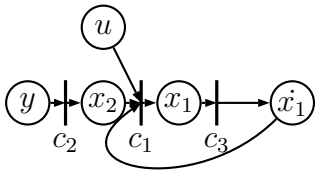

Figure 3.5: Oriented graph corresponding to matching in table 3.6

Consider the matching in table 3.6, it is a maximal matching that is complete with respect to $(x_1, \dot{x_1}, x_2)$.

The oriented graph induced for the matching shown in table 3.6 is shown in figure 3.5.

In the oriented graph associated with a matching, cycles are special subsets of constraints, which have to be solved simultaneously, because the outputs of some constraints in the cycle are the inputs of some others in the same cycle.

In structural analysis, an algebraic cycle is always supposed to have a unique solution. On the contrary, the uniqueness of the solution associated

with a differential cycle will depend on the context of the problem. A differential cycle represents a system of algebraic and differential equations that cannot be solved if initial conditions are not known.

When differential constraints are considered, matching all the variables in a subsystem guarantees that there is a unique solution under integral causality, i.e. when the initial conditions are known. Under derivative causality, the solution is unique if and only if there is a matching which avoids differential cycles. A matching that fulfills the causality assignment is called a *causal matching*.

3.4.4 System canonical decomposition

This section recalls a classical result from bipartite graph theory, which states that any finite-dimensional graph can be decomposed into three subgraphs with specific properties, respectively associated with an over-constrained, a just-constrained and an under-constrained subsystem [9, 40, 41]. This decomposition is unique for a given system.

Definition 3.4.3 (Over-constrained graph)**.** A graph $G_+ = (C_+ \bigcup X_+, \Gamma_+)$ is called over-constrained if there is a complete matching on the variables X_+ but not on the constraints C_+. It is denoted S^+.

Definition 3.4.4 (Just-constrained graph)**.** A graph $G_0 = (C_0 \bigcup X_0, \Gamma_0)$ is called just-constrained if there is a complete matching on the variables X_0 and on the constraints C_0. It is denoted S^0.

Definition 3.4.5 (Under-constrained graph)**.** A graph $G_- = (C_- \bigcup X_-, \Gamma_-)$ is called under-constrained if there is a complete matching on the constraints C_- but not on the variables X_-. It is denoted S^-.

Figure (5.3) illustrates the canonical decomposition (also called Dulmage-Mendelsohn decomposition) of any structure graph, showing the three canonical components on its bi-adjacency matrix.

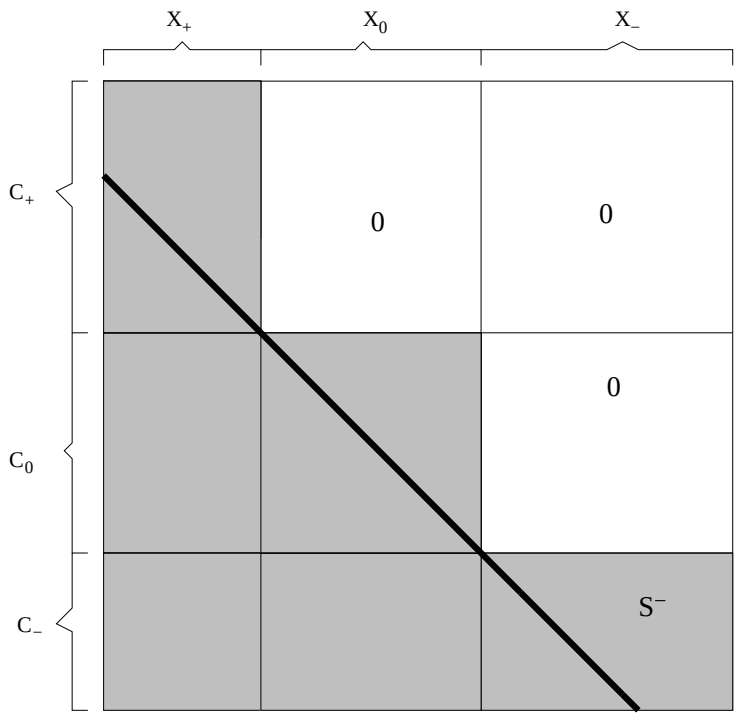

Figure 3.6: *The general scheme of any system canonical decomposition*

3.4.5 Interpretation of canonical decomposition

In the following, the canonical subsystems are analyzed from the point of view of the existence of solutions, thus providing a key for the analysis of the system monitorability property.

In the over-constrained subsystem: S^+, the variables X_+ (let n^+ be their number) have to satisfy more than n^+ constraints. Since thereare more constraints than variables, the system description is redundant [11, 110].This redundancy can be used to generate residuals.

In the just-constrained subsystem: S^0, the variables X_0 (let n^0 be their number) have to satisfy exactly n^0 constraints. A unique solution exists [11, 110].

In the under-constrained subsystem: S^-, the variables X_- (let n^- be their number) have to satisfy less than n^- constraints. Therefore, the under-constrained subsystem cannot exhibit any unique solution.

3.5 Structural monitorability analysis

This section explains how to use the tools just defined in order to analyze system's structural monitorability property. Also, the design of Fault Detection and Isolation (FDI) algorithms based on *Analytical Redundancy Relations* (ARR) issues is discussed in a structural perspective.

Indeed, based on chapter 2, redundancy is a requirement for monitorability and for residual generation. Structural analysis via the canonical decomposition is able to tell which part of the system is potentially redundant. Also, by using the *matching tool*, one might be able to know how to generate residuals on this redundant subsystem.

3.5.1 The structurally monitorable subsystem

Definition 3.5.1 (structurally monitorable subsystem)**.** The structurally monitorable part of the system is the subset of the constraints such that there exists ARR which are structurally sensitive to their change.

Then the following result holds:

Theorem 3.5.1 (Monitorability, [11])**.** *A necessary condition for a fault φ to be monitorable is:*
φ belongs to the over-constrained part of the system (C, V) such that there is at least one causal complete matching on the unknown variables.

Indeed, let S^+ be the over-constrained subsystem with at least one complete causal matching, then there exists a subset $C_+^i \subset C_+$ of $n = |X_+|$ constraints which (from a structural point of view) can be solved uniquely for the variables X_+. These variables can thus be computed as functions of the known variables. Putting the obtained values into the remaining constraint set $R_+^i = C_+ \backslash C_+^i$, one obtains $|C_+| - |X_+|$ relations which link only known variables and which are, therefore, redundancy relations.

Note that in general, several different complete matchings can be performed in a given causal over-constrained subsystem, thus providing different means of computing the unknown variables X_+ from the known ones.

3.5.2 The design of analytic redundancy relations

As explained previously, an oriented instance of the structure graph is associated with each matching $\mathcal{M}$. In a given oriented-graph, paths going from known variables $u_1, ..., u_m$ and $y_1, ..., y_p$ to each unknown variable $x_1, ..., x_n$ give the computation sequences in order to obtain each x_j's. Due to the graph-orientation explained in section 3.4.2, the non-matched constraints are always the *ending-nodes* or the *leaves* in such a tree-like graph. Hence, paths from known variables to a non-matched constraint c_k provide the computation sequences to follow in order to compute the unknown variables involved in c_k. It is then possible to know the signature (or the structure) of the residual that can be generated by using the redundant constraint c_k, based on the matching $\mathcal{M}$.

Therefore, designing a set of residuals calls for building maximal matchings on the given structural graph, under derivative causality, and identifying the residuals as the non matched constraints in which all the unknowns have been matched.

Note that the robustness requirements of residual is automatically fulfilled in structural analysis, using the strong decoupling approach, since it exhibits ARR which are, by definition, only dependent on known variables. Unknown

variables which affect the structurally monitorable subsystem are eliminated so that no residual can depend on them; when unknown variables cannot be eliminated, the part of the system they affect is not monitorable. When uncertain parameters are present, it may be wished not to use them in any ARR, because such ARR might generate false alarms or missed detections. The solution is simply to design the FDI system considering them as unknown variables.

3.6 Algorithmic issues

Algorithms which find maximal matchings have been previously investigated. An algorithm of complexity $O(N^3)$ for finding maximal matchings has been proposed by [44, 43], while [69, 70] have applied an algorithm of complexity $O(N^{2.5})$ to bipartite graphs where N is the number of vertices. Maximal matchings can also be found from the solutions to the assignment problem [84], or from the maximal flow problem [46, 45]. For details on the algorithms and more bibliographical notes, refer to [10, 61, 17]. Efficient algorithms exist ([40, 17] and therein references) to transform the system structure matrix into its Dulmage-Mendelsohn decomposition.

It is worth noticing that very few studies have been pursued in order to implement the structural analysis for FDI. The general trend till now was to refer to existing well-known graph-theoretical algorithms in order to perform the canonical decomposition, for instance, or to find any matching. In [76],

a tentative matching algorithm has been proposed, whereas a heuristic to generate the residual structure is proposed in [80].

Most of structural analysis studies are concerned with the formalization of the method in order to investigate how far it can be useful for FDI purposes. There are very few guides to *translate* the theory into efficient algorithms. Therefore, the next step to take is to consider algorithmic issues, and especially those that can arise during implementation phases.

For this purpose, first the properties that a structural FDI algorithm should possess have to be defined. Indeed, such a set of properties has never been defined previously for structural methods. This set would be very useful in order to assess any structural FDI algorithm. Then, the basic steps of structural analysis, that enables the analysis of a system's fault detectability and isolability properties, presented through this chapter, have to be translated into a dedicated algorithm that satisfies the defined criteria. Next, the residual generation problem should be addressed. That is to say, an algorithm that enables the generation of residuals based on structural information as accurately as possible has to be developed. Finally, since industrial systems are evolving over time, and their structures are subject to change, an algorithm that can efficiently deal with these possible modifications has to be developed.

3.7 Conclusion

Structural analysis is an important tool, which is of interest in the early stage of the control and supervision system design, when detailed models are not available. It enables the identification of the monitorable part of the system and provides some insight on the fault isolability property. Since detailed behavior models need only to be developed for those parts of the system, structural analysis is also a tool for deciding which modelling investments must be done for the design of the supervision system.

Undoubtedly, this approach has reached a certain theoretical maturity thanks to the rich bibliography reviewed in this chapter. Many real-life applications to various systems shows the method's usefulness and applicability.

However, as outlined previously, algorithms dedicated to the method have not yet been elaborated. We believe that this is an important obstacle to the practical use of the method. Next part is dedicated to our contribution to this problem.

Part II

Towards Implementation - Some Algorithmic Issues

Introduction to Part II

Part II gathers the original contributions of our studies, that is to say the algorithm that have been elaborated in order to efficiently automate the structural analysis for FDI method. Each chapter corresponds to a different problem identified.

First, in Chapter 4 the desirable characteristics of a structural analysis based FDI system are presented. It is useful to identify a set of desirable characteristics that a diagnostic system should possess. Then the algorithms may be evaluated against such a common set of requirements or standards. These characteristics might also be useful to benchmark various methods.

Then, in chapter 5, an algorithm that enables the analysis of system's structural monitorability property by generating the fault signature matrix is proposed. The algorithm's completeness and correctness proofs are performed, as well as its complexity analysis.

Chapter 6 proposes an algorithm that aims to improve the fault isolability property of a system. To be more precise, the algorithm enables the determination of the faults that require more detailed modelling in order to be able to isolate them from the others. Since the modelling process is expensive, it is valuable to get information of what level of knowledge about faults is necessary to obtain sufficient fault isolability properties of the diagnosis system.

The residual generation problem is addressed in chapter 7. The aim is to

choose among a set of different computation sequences that lead to equivalent residuals the most accurate with respect to practical constraints that might exist due to the systems specificity.

In chapter 8, an answer is sought to the question "what if the system structure slightly changes?". Indeed, the current answer is to re-iterate the whole analysis on the new structure. An algorithm that makes use of already-obtained results is proposed, thus avoiding unnecessary time consumption and computation.

Chapter 4

Algorithms: desirable properties

Algorithmes structurels de détection et localisation des défaillances: les propriétés souhaitées

Dans la partie précédente, nous avons tout d'abord présenté les principes généraux de la détection et de l'identification des défaillances. Ensuite, l'analyse structurelle a été introduite en soulignant ses avantages et la nécessité d'une telle approche. Enfin, bien que cette méthode existe depuis plusieurs dizaines d'années et bien qu'elle ait atteint une certaine maturité théorique grâce aux différents travaux, nous avons constaté qu'en pratique elle n'est pas fortement répandue dans le monde industriel du fait d'un manque de formalisation des algorithmes permettant une implantation simple et efficace.

Dans ce but, nous allons tout d'abord expliciter l'ensemble des propriétés

que l'analyse structurelle pour la surveillance doit posséder. Cet ensemble de propriétés permettra d'évaluer les algorithmes proposés dans les chapitres suivants dans le but d'automatiser cette méthode.

4.1 Outline of the chapter

In part I, the general problem of fault diagnosis was presented. Next, structural methods were emphasized by explaining their merits. Although structural methods have existed for decades and despite the large number of studies that made this method reach a certain theoretical maturity, there is a certain gap between the application and the theory. A discussion on the possible reason for this gap ended the previous part.

It is necessary to study algorithmic issues and to improve the method's implementation in order to make it more easily accessible. For this purpose, first a set of desirable characteristics that a diagnostic system should possess will be identified. The previous studies aiming to define such a set for diagnosis systems did not specifically target structural methods (see [116] for a good survey). However, some of the desirable properties previously defined in the literature are irrelevant for structural analysis. Indeed, some properties are inherent to structural analysis therefore it is unnecessary to check or to try to fulfill them.

Such a common set of requirements or standards will make it possible to evaluate the proposed algorithms that aim to automate the process in the next chapters. These characteristics may also be useful in order to benchmark various structural methods.

In the following, six properties are proposed which are believed to be the main properties of the ideal Structural FDI method, whereas one property

that is inherent to structural analysis is explained.

4.2 Soundness

The theory behind a structural FDI algorithm must be sound. Whenever an abnormality occurs in a process, the structural FDI algorithm would come up with a set of hypotheses or faults that explains the abnormality. Soundness of the structural FDI algorithm would require the proposed fault set to be a subset of the actual fault(s). This means that the solution proposed by the algorithm is correct.

4.3 Completeness

Another very important property is completeness and its definition is very similar to soundness definition. Whenever an abnormality occurs in a process, a structural FDI algorithm would come up with a set of hypotheses or faults that explains the abnormality. Completeness of a structural FDI algorithm would require the actual fault(s) to be a subset of the proposed fault set. This means that the algorithm finds at least all the possible solutions.

4.4 Efficiency

A structural FDI method should make it possible to quickly evaluate a large number of candidate solutions. Algorithms are commonly called efficient if

they solve problems in polynomial time in a measure of the problem size; if not, they are called inefficient. Usually, quick real-time solutions would require algorithms and implementations which are computationally less complex, but might entail high storage requirements. One would prefer a diagnostic system that is able to achieve a reasonable balance on these two competing requirements.

4.5 Effectiveness

For the purpose of efficiency, it is desired that structural FDI methods directs its search towards the ideal set of solution, instead of performing a candidate-by-candidate test for a particular criterion. The latter, brute-force approach is indirect and not solvable efficiently.

4.6 Adaptability

Processes in general change and evolve due to changes in external inputs or structural changes due to retrofitting and so on. Process operating conditions can change not only due to disturbances but also due to changing environmental conditions such as changes in production quantities with changing demands, changes in the quality of raw material etc. Thus the diagnostic system should be adaptable to changes. It should be possible to gradually develop the scope of the system as new cases and problems emerge, as more

information becomes available.

4.7 Usefulness

Usefulness is the property to obtain some useful results. This property is of great importance for structural analysis for FDI. Indeed, since the structural information of a system is a simplification, it is important to assure that the structural calculations made can result in something that can leads to useful result.

4.8 Ability to handle different model types

The ability to handle different kinds of models is a rather significant property for FDI systems in general since the information might be from various sources as well as from different types. This property however is inherent to structural analysis since this method does only take into account the structure of the system, rather than its nature.

4.9 Conclusion

A set of characteristics that any structural FDI algorithm should meet has been proposed. The rest of part II is devoted to the different algorithms that have been elaborated in order to efficiently automate the structural analysis for FDI.

Chapter 5

Monitorability analysis

Algorithme de surveillance

Dans ce chapitre, nous nous intéressons aux aspects algorithmiques de l'analyse structurelle pour la surveillance afin d'améliorer l'implémentation de cette méthode. Notre but est d'élaborer un algorithme qui permettra d'automatiser la génération de la table de signature des résidus. Comme nous l'avons montré dans la partie précédente, l'analyse de cette table nous permet d'évaluer les propriétés structurelle de détectabilité et de localisabilité des défaillances.

Dans la suite de ce chapitre, nous proposons dans un premier temps un algorithme dont nous prouvons les propriétés de correction et complétude. Nous élaborons alors une seconde version dont la complexité calculatoire est moins élevée. Nous démontrons l'équivalence du second algorithme au

premier, ce qui implique la correction et la complétude du second.

Les travaux décrits dans ce chapitre ont été présentés à [34].

5.1 Outline of the chapter

In this chapter, the structural analysis for FDI method previously introduced is translated in an easy to implement algorithm in order to automate the process of residual signature table generation. First, the problem is recalled in section 5.2. Then an exhaustive algorithm is proposed in section 5.3.1. This algorithm is evaluated with respect to the criteria defined in chapter 5.3.2. Finally, an improved version with less complexity is given in section 5.4.

5.2 The problem formulation

- **Input:** The structure matrix M (the corresponding bipartite graph G).

- **Aim:** To assess the system's monitorability property by analyzing its structural monitorability.

- **Output:** The residual signature matrix that enables the determination of which faults are definitely not detectable, and which are definitely not isolable from each other.

5.3 The exhaustive approach

5.3.1 The algorithm

A straightforward way to obtain the residual signature table is to generate all the complete matchings on the monitorable part S^+ of the system to diagnose, that is to say the matchings that cover all its unknown variables. The reason why all the complete matchings are generated is to gather all the possible residuals structure, without loss of information, in order to be able to distinguish different faults from each other. This exhaustive algorithm is given below in table 5.1.

The set of all complete matchings can be generated by performing a depth-first-traversal of the bipartite graph. This consists in starting at any unknown variable from G and continuing as far as possible into the graph by traversing alternatively constraints and unknown variables. Each new edge traversed must be one that has not already been covered by the algorithm. The path-building operation continues until it reaches a point at which there is no edge leading to a vertex that has not already been visited.

At such a point, if all unknown variables have been traversed, the obtained path corresponds to a complete matching, according to the orientation defined in section 3.4.2. If all unknown variables have not been traversed, we backtrack to the previous node, choose a new edge from that point, and build as long a path as possible from there. In backtracking, if all the edges from the previous node have been tried we backtrack again to the next-previous

	Algorithm 1**:**
1-	Compute the Dulmage-Mendelsohn decomposition of G
2-	Identify the over-constrained part S^+ of size $n \times m$ with $n > m$
3-	Compute all complete causal matchings on S^+
4-	**For** each complete matching on S^+:
	* Identify the set of constraints necessary to compute each unknown variable (also called computation sequence)
	* Generate the residuals' structure by substituting in the $(n - m)$ extra constraints the involved unknown variables' computation sequence
4-	**end for**
5-	Compute the *residual signature table*

Table 5.1: Residual signature table generation algorithm

node and so on.

The actual traversal is often accomplished using a stack data-structure. First, the starting node is pushed onto the stack. Then the following process repeats:

- Pop a node off the stack

- Traverse to this current node

- Push all nodes adjacent to the current node onto the stack

The process terminates when the stack is empty (see [69, 113] for further details).

Example 5.3.1. Consider the following system where u and y are known variables, x_1, x_2 and x_3 unknown variables.

$$c_1(x_1, x_2, x_3, y) = 0 \tag{5.3.1}$$

$$c_2(x_1, u) = 0 \tag{5.3.2}$$

$$c_3(x_1, x_2, x_3) = 0 \tag{5.3.3}$$

$$c_4(x_2) = 0 \tag{5.3.4}$$

$$c_5(x_1, x_2, x_3) = 0 \tag{5.3.5}$$

Regardless of the mathematical expression of the constraints, the system structure matrix is shown in table 5.2.

	x_1	x_2	x_3	y, u
c_1	1	1	1	1
c_2	1			1
c_3	1	1	1	
c_4		1		
c_5	1	1	1	

Table 5.2: Example 5.3.1 : structural model

The previous system is over-constrained. The structural analysis of this system for FDI purpose consists in finding first all the possible complete matchings on unknown variables $\{x_1$,x_2, $x_3\}$.

Consider the matching in table 5.3.1, the variable x_1 is matched to c_2. The variable x_2 is matched to c_4. By computing both variables and substituting them on c_1, the variable x_3 is computed. If c_3 is used in order to generate a residual, its signature is $\{c_1; c_2; c_3; c_4\}$.

If the other extra equation c_5 is used, a residual with signature $\{c_1; c_2; c_4; c_5\}$ is obtained.

There are 21 possible matchings on x_1, x_2 and x_3. The remaining 20 matchings are given below.

Matching 1:	x_1	x_2	x_3
c_1	1	1	(1)
c_2	(1)		
c_4		(1)	
c_3	1	1	1
c_5	1	1	1

Table 5.3: Example 5.3.1 : a possible matching

Matching 2:	x_1	x_2	x_3
c_1	1	1	1
c_2	(1)		
c_3	1	1	(1)
c_4		(1)	
c_5	1	1	1

Matching 3:	x_1	x_2	x_3
c_1	1	1	1
c_2	(1)		
c_3	1	1	1
c_4		(1)	
c_5	1	1	(1)

Matching 4:	x_1	x_2	x_3
c_1	(1)	1	1
c_2	1		
c_3	1	(1)	1
c_4		1	
c_5	1	1	(1)

Matching 5:	x_1	x_2	x_3
c_1	(1)	1	1
c_2	1		
c_3	1	1	(1)
c_4		1	
c_5	1	(1)	1

Matching 6:	x_1	x_2	x_3
c_1	1	(1)	1
c_2	1		
c_3	(1)	1	1
c_4		1	
c_5	1	1	(1)

Matching 7:	x_1	x_2	x_3
c_1	1	1	(1)
c_2	1		
c_3	(1)	1	1
c_4		1	
c_5	1	(1)	1

Matching 8:	x_1	x_2	x_3
c_1	1	1	(1)
c_2	1		
c_3	1	(1)	1
c_4		1	
c_5	(1)	1	1

Matching 9:	x_1	x_2	x_3
c_1	1	(1)	1
c_2	1		
c_3	1	1	(1)
c_4		1	
c_5	(1)	1	1

Matching 10:	x_1	x_2	x_3
c_1	1	(1)	1
c_2	(1)		
c_3	1	1	(1)
c_4		1	
c_5	1	1	1

Matching 11:	x_1	x_2	x_3
c_1	1	1	(1)
c_2	(1)		
c_3	1	1	1
c_4		(1)	
c_5	1	1	1

Matching 12:	x_1	x_2	x_3
c_1	1	1	1
c_2	(1)		
c_3	1	(1)	1
c_4		1	
c_5	1	1	(1)

Matching 13:	x_1	x_2	x_3
c_1	1	1	1
c_2	(1)		
c_3	1	1	(1)
c_4		1	
c_5	1	(1)	1

Matching 14:	x_1	x_2	x_3
c_1	1	(1)	1
c_2	(1)		
c_3	1	1	1
c_4		1	
c_5	1	1	(1)

Matching 15:	x_1	x_2	x_3
c_1	1	1	(1)
c_2	(1)		
c_3	1	1	1
c_4		1	
c_5	1	(1)	1

Matching 16:	x_1	x_2	x_3
c_1	(1)	1	1
c_2	1		
c_3	1	1	(1)
c_4		(1)	
c_5	1	1	1

Matching 17:	x_1	x_2	x_3
c_1	1	1	(1)
c_2	1		
c_3	(1)	1	1
c_4		(1)	
c_5	1	1	1

Matching 18:	x_1	x_2	x_3
c_1	1	1	1
c_2	1		
c_3	(1)	1	1
c_4		(1)	
c_5	1	1	(1)

Matching 19:	x_1	x_2	x_3
c_1	1	1	1
c_2	1		
c_3	1	1	(1)
c_4		(1)	
c_5	(1)	1	1

Matching 20:	x_1	x_2	x_3
c_1	(1)	1	1
c_2	1		
c_3	1	1	1
c_4		(1)	
c_5	1	1	(1)

Matching 21:	x_1	x_2	x_3
c_1	1	1	(1)
c_2	1		
c_3	1	1	1
c_4		(1)	
c_5	(1)	1	1

After analyzing these 21 matchings and performing the propagation into unused equations, residual structure are obtained. There are 5 different structures. The residual signature matrix depicted in table 5.3.1 is obtained.

□

5.3.2 Evaluation of *Algorithm-1*

In this section, the proposed algorithm is evaluated with respect to the criteria defined in the last chapter, that is to say the criteria that an ideal structural FDI algorithm should meet.

	c_1	c_2	c_3	c_4	c_5
r_1	1	1	1	1	
r_2	1	1	1		1
r_3	1	1		1	
r_4	1		1	1	1
r_5		1	1	1	1

Table 5.4: Example 5.3.1 : residual signature matrix

Soundness - Completeness of *Algorithm-1*:

Soundness and completeness of the algorithm can be more precisely stated as follows:

- if it is structurally possible to generate using variable elimination a residual whose signature is s_i then it is found by *Algorithm-1*.

- if a residual whose signature is s_i is found by *Algorithm-1*, it is structurally possible to generate s_i using variable elimination.

As explained at chapter 4, the *soundness* guarantees that all results found by *Algorithm-1* are valid. The second property, *completeness*, guarantees that *Algorithm-1* generates all the structurally possible residual signatures. This is especially important, since it assures that no residual structure will be omitted and the algorithm output can be used to evaluate the system's fault isolability property.

Theorem 5.3.2 (Soundness)**.** *Let S_{all} denote the signature set of all residuals that can be obtained by variable elimination and S_1 the set of residuals signature obtained by algorithm-1. Then, $\forall s_i \in S_1, s_i \in S_{all}$.*

Proof. This is a straightforward consequence of residual definition (a relation between inputs and outputs) and the fact that $\forall s_i \in S_1$, s_i is obtained by variable elimination. □

Theorem 5.3.3 (Completeness)**.** *Let S_{all} denote the signature set of all residuals that can be obtained by variable elimination and S_1 be the set of residuals signature obtained by algorithm-1. Then, $\forall s_i \in S_{all}, s_i \in S_1$.*

Proof. $\forall s_i \in S_{all}$, $s_i = \{c_i, c_j..c_p\}$ where $\{c_j, ..., c_p\}$ is the set of constraints required to compute all the unknown variables appearing in c_i. That is to say, the set of constraints used in order to generate s_i corresponds to an over-constrained block with one more constraint than variables (for instance of size $p+1 \times p$). We have to prove that *algorithm-1* considers all such possible blocks.

Assume that the over-constrained part of the whole system to be diagnosed S^+ is of size $(n \times m)$ with $n > m$.

Consider any over-constrained set of size $((p+1) \times p)$ (call it O_p). By adding $(m - p - 1)$ constraints to this set, a square block of size $(m \times m)$ is obtained. If the Dulmage-Mendelsohn decomposition of this square block is performed, O_p is part of the over-constrained subsystem. Suppose this over-constrained block is exactly O_p (figure (5.1)).

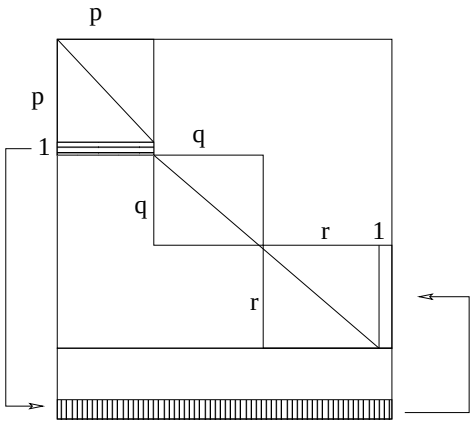

Figure 5.1: Line swapping

Since we performed the decomposition of a square block, there is also an under-constrained part that holds 1 more unknown variable than constraints. Since the whole system S^+ is over-constrained, there is at least one complete matching on the m unknown variables. This implies that there are among the remaining $(n-m)$ constraints at least 1 constraint to match the unknown variables of the under-constrained part (the vertically dashed constraint in table (5.1)). Therefore, by interchanging this constraint with the extra constraint from O_p (the horizontally dashed constraint in table (5.1)), we obtain a square block of size $m \times m$ that is just constrained (see figure (5.2)).

That is to say, the diagonal entries form a complete matching on the unknown variables. By substituting on the extra equation (horizontally dashed constraint) the computation sequence of occurring unknown variables, we will obtain the residual structure that can be obtained from O_p. This proves that for each over-constrained block of degree 1, there is a complete matching

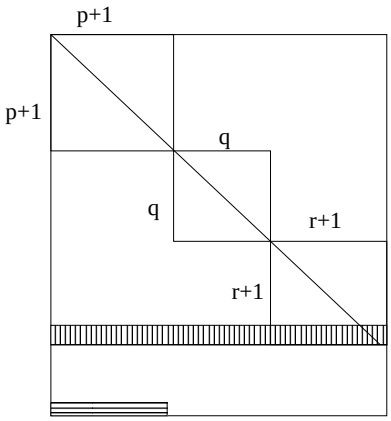

Figure 5.2: The complete matching after line swapping

that enables the generation of the corresponding residual's structure. □

Efficiency of *Algorithm-1*:

When S^+ is a full matrix of size $n \times m$, the worst-case occurs. The total number of matching to deal with, T_M, is then:

$$T_M = \frac{n!}{(n-m)!}$$

Effectiveness of *Algorithm-1*:

Although *algorithm-1* is sound and complete, it requires to generate all the possible complete matchings in order to generate residuals structure. The previous example shows that some different residuals may have the same structure. There is a certain redundancy in the process that makes *algorithm-1* time-consuming. In fact, one can think that the residual generation process

is performed off-line, therefore complexity issues are not so important. However, the diagnosis system is an evolutive system that may have to be rapidly updated, for instance in case of reconfiguration due to sensor fault or sensor addition. That is why it is important to keep time complexity as low as possible.

The next section shows that, by considering some graphical properties, the same structural result may be obtained with less computation.

5.4 How to improve *algorithm-1*?

In order to decrease the number of matchings to consider, we propose to exploit some graphical properties of the over-constrained subsystem, namely the connectedness property of the König-Hall blocks.

As we mentioned before, the first step of structural analysis is to permute the system structure matrix M so as to obtain its canonical decomposition. In general, it is not possible to transform the adjacency matrix to a strictly lower-triangular form. However it can be transformed into a unique block-lower triangular form i.e. a quasi lower triangular form in which square blocks of dimension ≥ 1 are present along the diagonal (see figure 5.3).

Each of the S_i^+ for $i \in 1...n$ stands for subsystems with as many equations as unknown variables. They are called *König-Hall components* [63] and they correspond to connected components of S^+. The connected components of a graph are the maximal connected subgraphs of this graph.

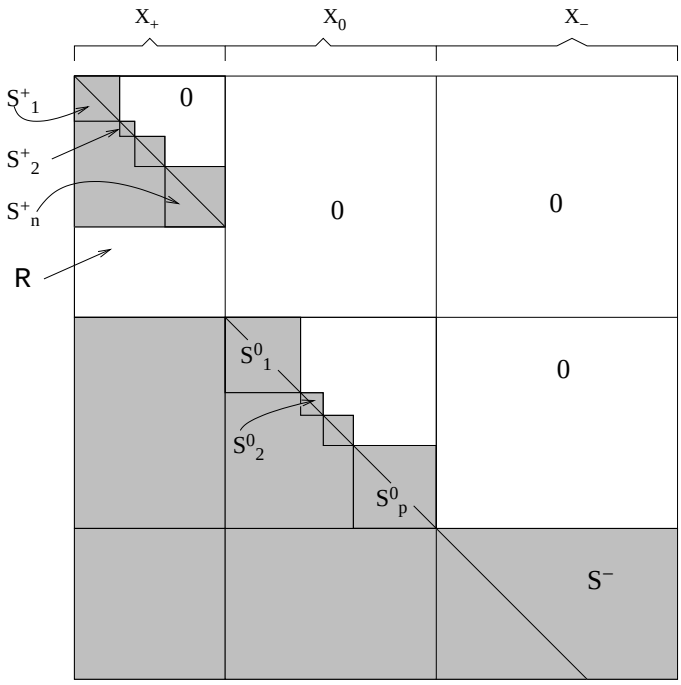

Figure 5.3: The canonical decomposition with KH-blocks

Indeed, since König-Hall components correspond to connected components present along the diagonalized form representing the computation to be performed sequentially, they stand for systems of equations that have to be solved simultaneously. That is to say, a König-Hall block S_i of size k corresponds to a minimal subsystems of k equations involving k unknowns, that are mutually dependant. Graphically, this corresponds to a cycle in the oriented graph induced by the matching.

The mutual dependence of variables implies that in order to compute an unknown variable of S_i, all the other variables of S_i have to be computed. For this purpose, all the constraints of S_i are used. Considering figure (5.3), suppose $c_r \in R$ such that it involves only one unknown variable x_i that is from S_1^+. In order to generate a residual from c_r, all the ways of computing x_i are investigated by *algorithm-1*. However, the connectedness property of König-Hall blocks tells us that all possible matchings involve every constraints in S_1^+. Therefore, all the different matchings are equivalent with respect to the set of constraints used to generate the residual from c_r. The main idea of the proposed revisited algorithm is to generate one matching per König-Hall block configuration instead of all possible matchings.

In the over-constrained subsystem S^+, the constraints can be arranged in different ways in order to obtain different blocks $\{S_1^+, S_2^+, ..., S_n^+\}$. Each such different arrangement is called *König-Hall configuration*. In fact, there are as many different König-Hall configurations as just-constrained part of size $m \times m$, with m being the number of variables in S^+. The proposed revisited

	Algorithm 2:
1-	Compute the Dulmage-Mendelsohn decomposition of G
2-	Identify the over-constrained part S^+ of size $n \times m$ with $n > m$
3-	**For** each block of size $m \times m$:
4-	**If** it is just-constrained:
	Identify König-Hall blocks
	Derive $(n - m)$ residuals' signature by the
	König-Hall blocks connectedness and order of resolution properties
5-	**end If**
6-	**end if**
7-	Compute the *residual signature matrix*

Table 5.5: Improved residual signature matrix generation algorithm

algorithm finds each just-constrained part and computes only one complete matching per König-Hall configuration.

5.4.1 The improved algorithm

The main steps of an improved algorithm, that takes advantage of the Köning-Hall blocks connectedness property are recapitulated in table 5.5.

5.4.2 Algorithm properties

The Improved Algorithm Soundness and completeness Proofs:

In order to show that *algorithm-2* is also complete and sound, we show that it is complete and sound with respect to *algorithm-1*, that is to say:

- if a residual signature s_i is found by *algorithm-2*, it is also found by *algorithm-1*.
- if a residual signature s_i is found by *algorithm-1*, it is also found by *algorithm-2*.

Theorem 5.4.1 (soundness)**.** *Let S_1 denote the set of residual signatures obtained by applying algorithm-1 to the system and S_2 the set of residuals signature obtained by algorithm-2. Then, $\forall s_i \in S_2, s_i \in S_1$.*

Proof. Denote by M_2 the set of matchings that permit to generate S_2, and M_1 the set of matchings to generate S_1. Since M_1 is by definition the set of all complete matchings, $M_2 \subseteq M_1$. Since, S_2 and S_1 are generated based on M_2 and M_1 respectively, $S_2 \subseteq S_1$. □

Theorem 5.4.2 (Completeness)**.** *Let S_1 denote the set of residual signatures obtained by applying algorithm-1 to the system and S_2 the set of residuals signature obtained by algorithm-2. Then, $\forall s_i \in S_1, s_i \in S_2$.*

Proof. This is a straightforward consequence of the connectedness property of König-Hall blocks. □

Corollary 5.4.3. *S_1 is equivalent to S_2 because $S_1 \subseteq S_2$ (by Theorem 5.4.1) and $S_2 \subseteq S_1$ (by Theorem 5.4.2).*

Efficiency of *algorithm-2*:

In *algorithm-2*, the number of possible König-Hall configuration determines the overall complexity. In case of fully connected systems, the total number of König-Hall configurations to work with, T_{KH}, would be:

$$T_{KH} = \frac{n!}{m! \times (n-m)!}$$

This is the best possible case for *algorithm-2*.

Effectiveness of *algorithm-2*:

Algorithm-2 considers the König-Hall components connectedness property in order to minimize the number of matchings to handle thus the same structural result may be obtained with less computation than *algorithm-1*.

Example 5.4.4 (continuing). When *algorithm-2* is applied to the previous example, only 10 blocks of size 3×3 are considered (see table 5.4.4). In this table, each line corresponds to different König-hall configuration considered by *algorithm-2*. The first column gives for each König-hall configuration the set of equivalent matchings considered by *algorithm-1*. For example, line 1 tells that though the matching 1 and the matching 11 are distinct case found by *algorithm-1*, they are equivalent according to *algorithm-2* since both have

Equivalent matching	c_1	c_2	c_3	c_4	c_5
$1 : \{1; 11\}$	x_3	x_1		x_2	
$2 : \{2\}$		x_1	x_3	x_2	
$3 : \{3\}$		x_1		x_2	x_3
$4 : \{4; 5; 6; 7; 8; 9\}$	x_1		x_2		x_3
$5 : \{10\}$	x_2	x_1	x_3		
$6 : \{12; 13\}$		x_1	x_2		x_3
$7 : \{14; 15\}$	x_3	x_1			x_2
$8 : \{16; 17\}$	x_1		x_3	x_2	
$9 : \{18; 19\}$			x_1	x_2	x_3
$10 : \{20; 21\}$	x_3			x_2	x_1 □

Table 5.6: Example 5.4.4 : matchings obtained with algorithm-2

the same König-hall configurations. Therefore, it is sufficient to generate only one of them.

□

5.5 Conclusion

The aim of this chapter is to give an easily implementable algorithm of structural analysis for FDI to automate the process. First, an exhaustive algorithm is proposed. The algorithm is shown to be complete and sound. Although structural analysis for FDI is an off-line method, complexity is-

sues are important. Therefore, a second algorithm with lower complexity is presented.

Chapter 6

Fault isolability

La localisabilité des défaillances

Ce chapitre est consacré à l'analyse des résultats obtenus à partir de l'algorithme proposé au chapitre précédent, afin d'évaluer les possibilités de localisabilité des défaillances du système.

L'analyse de la table de signature des résidus permet de déterminer les propriétés de détectabilité et de localisabilité structurelles des défaillances. Une défaillance est détectable si sa signature comporte au moins un "1". Deux défaillances sont localisables si leurs signatures sont différentes. Afin d'étudier et visualiser la localisabilité des défaillances, il est possible de construire une matrice appelée *table de localisabilité*. Par permutation de lignes et colonnes, la table de localisabilité peut être mise sous forme bloc-diagonale, où chaque bloc est une matrice carrée constituée uniquement de 1. Chaque bloc représente un sous-ensemble de défaillances non localisables entre elles.

La matrice de localisabilité permet d'indiquer les sous-ensembles de défaillances non localisables entre elles, c'est-à-dire ne pouvant être différenciées à l'aide des résidus.

Afin d'améliorer la localisabilité des défaillances, une connaissance supplémentaire sur le système doit être utilisée. Un moyen possible consiste à ajouter des capteurs [14, 15, 114, 93, 99, 2]. Un autre moyen consiste à utiliser les modèles des défaillances lorsque ceux-ci sont disponibles. L'utilisation de ces modèles peut permettre en effet d'améliorer les performances du système de surveillance en terme de localisabilité des défaillances. L'analyse de la table de localisabilité structurelle permet d'indiquer sur quelles défaillances un effort de modélisation doit être porté afin de respecter le cahier des charges de surveillance en terme de localisabilité des défaillances.

Le chapitre 6 est structuré de la manière suivante: le paragraphe 6.2 explique comment l'information structurelle peut être utilisée afin d'améliorer la localisabilité structurelle des défaillances du système. Au paragraphe 6.3, un exemple illustre l'idée de modéliser plus finement certaines défaillances.

6.1 Outline of the chapter

The residual structure matrix obtained as an output of last chapter's algorithm may be analyzed in order to assess the system's potential fault isolability and detectability properties. If the obtained isolability properties are not satisfactory, some ways to improve them by extra sensor placement [15, 14, 114, 93, 99, 2] can be proposed. In this chapter, the focus will be on improving fault isolability by further fault modelling [33, 37, 38, 51].

Indeed, obtaining accurate models is expensive and difficult and this is especially true when models describing faulty behavior of the plant are considered. To obtain such models, experimental data from a faulty process or deep engineering knowledge and experience of process components are needed, neither of which may be present in the early stages of the development process. Therefore it is interesting to analyze the level of knowledge about specific faults that is needed to meet diagnosis requirements and to get information of what level of knowledge about faults is necessary to obtain sufficient fault isolability properties of the diagnosis system. Structural analysis is an appropriate tool to get such information.

This chapter is organized as follows. Section 6.2 explains how structural information can be used to analyze the structural fault isolability property of a system. Then in section 6.3 an example illustrates that introducing additional fault models may improve this property.

6.2 Structural fault isolability

The algorithm presented at the previous chapter enables the generation of the residual signature matrix. As explained at 3.3.2, there may be three different way of modelling faults according to how much it is known about them. Consider the case where fault information from at least level 2 is available, that is to say we know which equation(s) within a component description is subjected to fault influences by means of variables describing faults ($f_i's$). Then, the residual signature matrix is used to generate the *fault signature table*.

Definition 6.2.1 (Fault signature table)**.** This is a matrix where each row corresponds to faults and each column to residuals. A *1* in position (i, j) indicates that fault i affects the residual j. Thus, in case $r_j \neq 0$, f_i may have occurred.

Faults with zero signature are obviously not detectable. Faults that have the same signature are not isolable from each other. Since the fault signature table size may be very important, visualizing faults isolability property may become difficult. In order to easily visualize the isolability property, the *isolability matrix* is computed.

Definition 6.2.2. The isolability matrix is a square matrix where each row and each column correspond to a fault. A *1* in position (i, j) indicates that fault i is not isolable from fault j.

	x	$\dot{x}$	f_1	f_2	u	y
c_1	1	1	1		1	
c_2	(1)			1		1
c_3	Δ	(1)				

Table 6.1: Illustrative example: structural model

The Dulmage-Mendelsohn permutation of the isolability matrix puts it in block lower triangular form where each diagonal block represents a subset of faults non-isolable from each other.

6.3 Improvement by adding faulty behavior model

The isolability analysis determines the subset of non-isolable faults. In order to improve fault isolability, deeper knowledge about the system is required. One way to achieve this is to add sensors [14, 93, 114, 99, 2]. Another way consists in incorporating fault models (the third level of knowledge described in section 3.3.2) when this is possible [33, 37, 38, 51]. In order to illustrate this idea, consider the following model describing the faulty behavior:

$$\dot{x} = -(\alpha + f_1)x + u \tag{6.3.1a}$$

$$y = x + f_2 \tag{6.3.1b}$$

The structural model is shown in table 6.1.

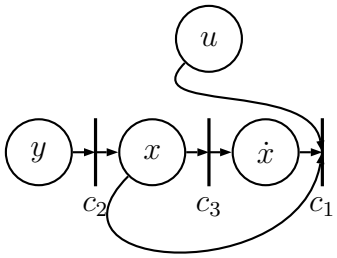

Figure 6.1: Oriented graph obtained from matching in table 6.1

	f_1	f_2
f_1	1	1
f_2	1	1

Table 6.2: Illustrative example: fault isolability matrix

$\dot{x}$ are x the unknown variables. The given complete causal matching leads to the oriented graph given in figure 6.3 and the fault isolability matrix in table 6.3.

The residual r_1 obtained based on this computation sequence is structurally sensitive to faults f_1 and f_2. Since there is no other residual with different fault signature, the faults are not isolable.

Now, assume it is known that fault f_1 is slowly varying or, for a given limited time window, approximately constant (cf. third level of knowledge in section 3.3.2). This information can be stated as

	x	$\dot{x}$	f_1	$\dot{f}_1$	f_2	u	y
c_1	1	1	(1)			1	
c_2	(1)				1		1
c_3	Δ	(1)					
c_4			Δ	(1)			
c_5				1			

Table 6.3: Illustrative example: structural model after further modelling

$$\dot{f}_1 = 0 \tag{6.3.2}$$

The structural model becomes as in table 6.3.

If we consider $\dot{f}_1$ and f_1 as two extra unknown variables to be decoupled, the system is still over-constrained. There is a complete causal matching on $\{x, \dot{x}, f_1, \dot{f}_1\}$. The residual r_2 structurally sensitive only to fault f_2 can be generated. The corresponding computation sequence is derived from the oriented graph induced by the following complete causal matching (fig. 6.3)

The fault signature matrix and the corresponding isolability matrix are given in table 6.4 and 6.5.

The isolability matrix shows which faults are not isolable from each other. The idea is to *break* each block by a further modelling effort in order to fulfill isolability requirements.

Note however, it is not guaranteed that fault isolability will be improved

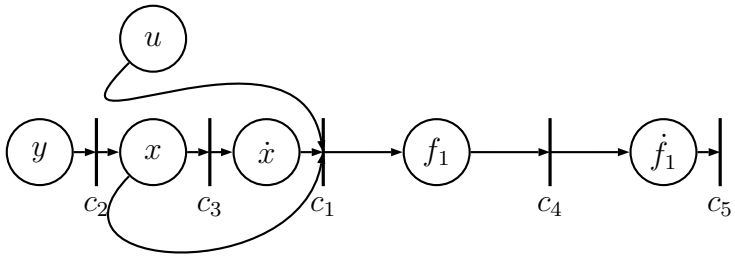

Figure 6.2: Illustrative example: oriented graph corresponding to matching in table 6.3

	f_1	f_2
r_1	1	1
r_2		1

Table 6.4: Illustrative example: fault signature matrix after further modelling

	f_1	f_2
f_1	1	
f_2		1

Table 6.5: Illustrative example: isolability matrix after further modelling

by further modelling. It may happen that even further modelling does not generate residuals with different structures.

Moreover, the method does not tell which fault(s) requires further modelling among a block of non-distinguishable faults. For instance, in the previous example, we supposed further modelling for f_1. However, further modelling on f_2 would not generate a new residual to isolate f_1 and f_2 from each other.

6.4 The problem formulation and proposed solution

- **Input:** The isolability matrix.
- **Aim:** Improve fault isolability.
- **Output:** The set of faults that requires finer modelling.

The straightforward idea consists in using the isolability matrix in order to first identify the set of faults not isolable from each other. Then, by supposing the existence of further knowledge about one of these faults, the analysis is performed from the beginning in order to check if such an extra information enables the improvement of fault isolability. By this way, it is possible to know which faults needs a further modelling.

Figure 6.3 summarizes the whole process of fault isolability improvement by combining both the sensor addition and the further modelling approaches.

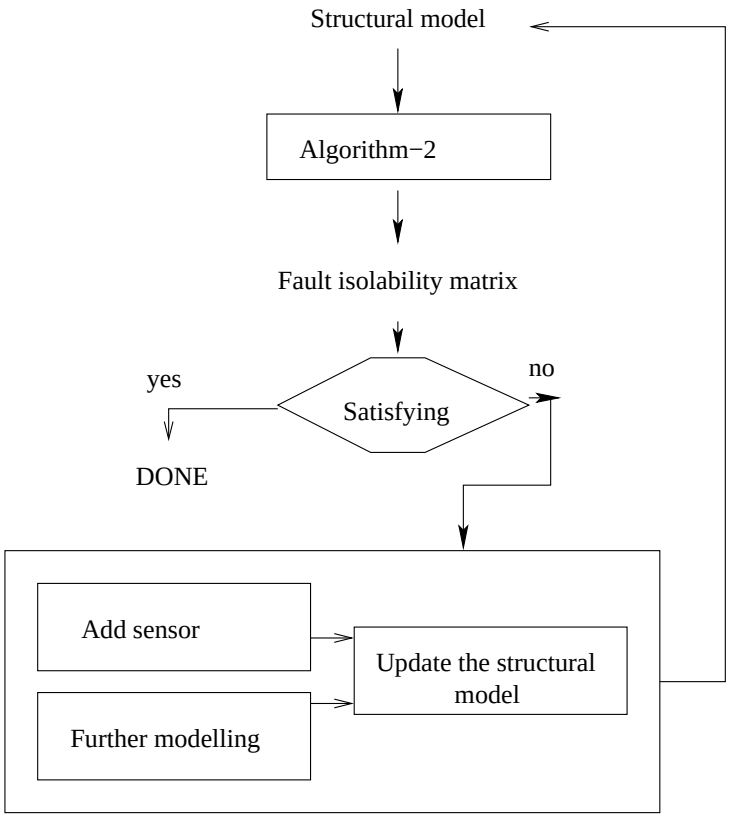

Figure 6.3: Flow-chart: algorithm of fault isolability improvement

With respect to algorithmic criteria defined at chapter 4, the soundness and completeness of the proposed approach is a natural consequence of *Algorithm*-2's soundness and completeness property. Regarding the effectiveness, *Algorithm*-2 is repeated as many times as there is faults in the target set to dissociate.

6.5 Conclusion

This chapter is concerned with structural fault isolability analysis. It is first shown how structural analysis can be used to assess fault isolability properties of a model, and then how it can help to improve this property. Since detailed knowledge about faulty systems is difficult to develop, it is essential to minimize this task. First, only information on which equations in the model that is influenced by the faults are provided. If the fault isolability objectives are not met, it is shown how structural analysis provides information on which faults need further modelling. It is also shown how additional fault models can be directly incorporated into the structural isolability analysis to ensure that the isolability objectives are met.

Chapter 7

Residual generation

Génération de résidus

Dans ce chapitre, nous considérons le problème du choix du couplage sur le graphe (matching) pour la génération de résidus.

Sur un bloc de König-Hall, il existe plusieurs façons de coupler les variables et relations de contraintes. Ceci entraîne des séquences de calculs différentes pour les variables donc pour les résidus. Étant donné que l'ensemble de contraintes est identique, la sensibilité structurelle des résidus est la même quel que soit le couplage choisi. Nous avons ainsi proposé au chapitre 5 un algorithme de couplage permettant de tenir compte de ces blocs de König-Hall. Nous utilisons le fait qu'en ne considérant qu'un couplage par bloc, il est possible de réduire considérablement le nombre de couplages nécessaires pour générer l'ensemble des résidus de structure différente.

D'un point de vue structurel tous les couplages sur des blocs de König-Hall sont équivalents. Ceci n'est plus vrai lorsque les résidus sont implantés en considérant les formes explicites des contraintes. Les raisons sont les suivantes :

- **Complexité de calcul** : certaines contraintes sont plus facilement calculables ou implantables *dans un sens plutôt que dans un autre*. Certaines non-linéarités ou tables par exemple sont difficilement inversibles.

- **Robustesse des calculs** : en présence de perturbations (incertitudes de modélisation, bruits), il est préférable d'utiliser certaines fonctions d'une manière plutôt que d'une autre afin de ne pas amplifier l'effet de ces incertitudes.

- **Sensibilité aux défaillances** : la sensibilité d'un résidu à une défaillance agissant sur une contrainte peut être différente suivant l'utilisation qui est faite de cette contrainte.

- **Précision (ou conditionnement)** : même en linéaire il vaut mieux diviser par un nombre important que par presque zéro (en variables réduites évidemment). Bien sûr, dans un cadre structurel, les valeurs numériques ne sont pas connues. Néammoins, ils se peut qu'on sache à l'avance l'ordre de grandeur d'une variable.

Nous avons proposé dans [36] et [35] de prendre en compte ces considéra-

tions pratiques (lorsque l'information est disponible) d'utilisation des contraintes dans l'analyse structurelle du système. L'idée originale est d'introduire des ordres partiels de préférence $\preccurlyeq$ sur les variables et sur les contraintes. Formellement, ces ordres partiels sont définis de la manière suivante:

- Par rapport aux variables : $c_j \preccurlyeq_{x_i} c_k$ lorsqu'il est préférable de coupler x_i à c_j plutôt qu'à c_k.

- Par rapport aux contraintes : $x_j \preccurlyeq_{c_i} x_k$ lorsqu'il est préférable de coupler c_i à x_j plutôt qu'à x_k

Dans [36], nous avons cherché à obtenir le "meilleur" couplage sur le graphe en tenant compte des ordres de préférences sur les variables et sur les contraintes. Nous avons rapproché notre problème au problème classique de combinatoire : le Stable Marriage Problem (SMP) [52] pour lequel de nombreux algorithmes ont été proposés dans la littérature [62, 92]. L'adaptation du SMP à notre problème a nécessité de définir les concepts de *paire (ou couple contrainte/variable) bloquante* et de *couplage stable*.

Dans [35], l'ordre partiel est transformé en pondération, ou coût, sur chaque arête du graphe structurel. Le coût global d'un couplage M sur le graphe est donné par la somme des pondérations de chaque couplage contrainte/variable impliqué dans M. Nous cherchons alors le "meilleur" couplage M^* au sens du minimum de coût global. Il s'agit d'un problème classique dans la théorie des graphes, pour lequel de nombreux algorithmes ont

été proposés [53].

7.1 Outline of the chapter

In this chapter an innovative way of dealing with the generation of residuals based on structural information is presented. The developed technique considers implementation issues, therefore it has a more realistic point of view compared to classical structural approaches. The major contribution of this chapter is the incorporation of extra knowledge - when available - into the structural analysis framework in order to choose among equivalent matchings the most suitable to generate residuals.

We here assemble in an original way a group of concepts from computer science, combinatorics, and graph theory which, taken together, make residual generation much easier to achieve.

In the following, first the problem is formulated in section 7.2. Then, practical issues that might be encountered such as computational complexity or implementation considerations are introduced in section 7.3. The way of incorporating them into the existing structural framework is explained in section 7.4. Finally, section7.5 shows how some known algorithms can be successfully adapted in order to choose the most suitable matching.

7.2 The problem formulation

When residual generation is considered, different matchings lead to different computation sequences. In order to choose the best suitable matching with respect to implementation issues, further information, if available, may be

	x_1	x_2	u, y
c_1	1	(1)	1
c_3	(1)	1	
c_2	1		1

Table 7.1: Example 7.2.1: matching 1

considered and incorporated into the current structural framework.

Example 7.2.1. Consider the system ((7.2.1),(7.2.2),(7.2.3)).

$$c_1(x_1, x_2, y) = 0 \tag{7.2.1}$$

$$c_2(x_1, u) = 0 \tag{7.2.2}$$

$$c_3(x_1, x_2) = 0 \tag{7.2.3}$$

Suppose that $c_1 : 7x_1 + x_2^2 = u$.

The matching in table 7.2.1 leads to computations that require to handle square root since x_2 is computed from c_1.

Whereas the other possible matching (table 7.2.1) leads to simpler computation:

Despite the two matchings are equivalent with respect to residual structure, they are not equivalent according to implementation issues. □

The residual generation problem considering implementation issues can be formalized as follows:

	x_1	x_2	u, y
c_1	(1)	1	1
c_3	1	(1)	
c_2	1		1

Table 7.2: Example 7.2.1: matching 2

- Decide which considerations have to be taken into account for implementation purposes (section 7.3)

- Incorporate them into the structural framework (section 7.4)

- Choose the best matching that fulfils the defined requirements (section 7.5)

7.3 Some practical considerations

There are many practical considerations that might affect accuracy and efficiency of the residual generation process and they mainly depend on the available knowledge about the system. The following can be mentioned for illustration purpose:

- *Computational complexity:* non-linear models are often subject to this kind of consideration. For instance, some functions are harder to compute if they have to be inverted (see previous example).

- *Uncertain relations due to perturbations, unknown, or varying parameters:* in complex systems, it can be difficult to obtain detailed and accurate models. Some parameters can vary or be uncertain. As a consequence, corresponding constraints should be used in a given way in order not to emphasize the effect of the uncertainty on the residual value.

- *Sensitivity to faults* : A residual sensitivity to a fault on a certain constraint may differ according to how this constraint is used.

- *Precision (or conditioning)* : even for linear systems, it is always preferable to divide a number by a much larger number than a much smaller number (almost zero). Although in structural analysis the numerical values of variables are unknown, experts may know the magnitude of a variable (its approximative range) based on previous experience or further engineering knowledge .

Based on the available knowledge, the matching that maximizes both accuracy and efficiency is desired among all the possible matchings giving the same signature residuals.

7.4 Incorporation into the structural framework

In the previous example, we intuitively sorted the variables of c_1 in order to minimize computational complexity. In a similar way, an ordered preference

list ranking its variables can be created for each constraint. Also, an ordered preference list ranking its occurring constraints can be created for each variable. If it is not possible to rank the variables (constraints) with respect to constraints (variables) because there is no information or it does not matter, they may have the same rank.

The preference lists can be formally represented by a partial order $\succcurlyeq$ such that $x_j \succcurlyeq_{c_i} x_k$ for $c_i \in C$, x_j and $x_k \in Z$, when x_j is more suited to be matched to c_i than x_k. In other words, when c_i appears before c_k in x_j's preference list.

Similarly, $c_j \succcurlyeq_{x_i} c_k$ for $x_i \in Z$, c_j, $c_k \in C$ will be written, when c_j is more suited to be matched to x_i than c_k.

Example 7.4.1 (continuing). Suppose system ((7.2.1),(7.2.2),(7.2.3)) is:

$$c_1 : 7x_1 + x_2^2 = y \tag{7.4.1}$$

$$c_2 : x_1^3 = u \tag{7.4.2}$$

$$c_3 : x_1^2 + x_2^2 = 0 \tag{7.4.3}$$

If the way of computing x_1 in order to decrease computational complexity is considered, first c_1 is preferable, then c_3 and c_2. Therefore, c_1 is more suited than c_3, and c_3 is more suited than c_2 to be matched to x_1. The preference orders in table 7.4.1 can then be obtained.

□

The current structural analysis consider every matching with equal rank.

x_1	:	c_1	$\succcurlyeq_{x_1}$	c_3	$\succcurlyeq_{x_1}$	c_2
x_2	:					$\{c_1, c_3\}$
c_1	:			x_1	$\succcurlyeq_{c_1}$	x_2
c_2	:					x_1
c_3	:					$\{x_1, x_2\}$

Table 7.3: Example 7.4.1 : preference order

The next section seeks an answer to the question *how to use this partial ordering to find the most suitable matching.*

7.5 How to find the best matching?

Given the structural representation of the system to be supervised, and the partial ordering information available, the goal can be set as *finding a complete matching M^* which gives the residual computation sequence that is the most suited to fulfil specifications* (here-after called *Residual Generation Problem* for simplicity). For this purpose, two methods are proposed. The first one is a method based on an interesting class of algorithms that solve the *Stable Marriage Problem* that is a well-studied combinatorics problem. It is presented and adapted to our concerns (section 7.5.1). In the second method, *the minimum weight maximum cardinality matching problem* is used; algorithms that solve this problem and their performance are considered in

section 7.5.2.

7.5.1 The stable marriage problem

The stable marriage problem (SMP) was first introduced in 1955 by Gale and Shapley [52] and has been studied intensively [62, 92].

The original problem is stated in terms of two equal-sized sets:

- a set A of n men,
- a set B of n women all of whom wish to get married to a member of the other set.

Each man independently creates a strictly ordered preference list, ranking each of the n women. Each woman does the same, ranking each of the n men.

Consider a_i and $a_j \in A$, b_k and $b_l \in B$ such that a_i is matched to b_k and a_j is matched to b_l. If there is a man and a woman, for example a_j and b_k such that a_j strictly prefers b_k to its actual partner (b_l) and b_k strictly prefers a_j to its actual partner (a_i), this matching is said to be unstable.

A stable matching is any matching that is not unstable. The aim is to find a stable matching between A and B so that each man is matched to a woman and each woman is matched to a man.

It is proven that every instance I of SMP admits at least one stable matching, that can be found with an algorithm, the *Gale-Shapley algorithm* (hereafter called GS-algorithm), in polynomial time.

A few extensions to the basic SMP above have been studied. The first one is to allow incomplete preference lists. The *Stable Marriage Problem with Incomplete List (SMPI)* is polynomially solvable by extending the GS-algorithm (see [62], Section 1.4.2).

The other extension is to allow preference lists including ties, i.e. equal ranking. By breaking the ties arbitrarily (or extending each partial order to a total order), an instance I of *Stable Marriage Problem with Ties (SMPT)* becomes an instance I' of *SMP*, and clearly the stable matching found for I' is also a stable matching for I. Thus a solution to SMPT can be found using the GS-algorithm.

Finally, the *Stable Marriage Problem with Incomplete List and Ties (SMPIT)* is NP-complete for the above given stability definition that requires a strict preference; however, there is a polynomial algorithm to solve it when equal preference is allowed (this is called *strong stability*, see [92]).

Instances of SMP and SMPT have complete stable matchings [52]. Although stable matchings for instances of SMPI and SMPIT may be incomplete, they all have the same cardinality and involve exactly the same men and women [92].

7.5.2 Adaptation of the SMP to the residual generation problem

The analogy with our problem becomes more explicit when the *variables set* and the *constraints set* of a König-Hall block are assigned to A and B respectively. The concepts given above can then be adapted to our concern and formalized as in the following definitions.

Definition 7.5.1 (Blocking pair)**.** Let M be a matching. Suppose $(c_i, x_k) \in M$ and $(c_l, x_j) \in M$. The couple (c_i, x_j) is a blocking pair if $x_j \succ_{c_i} x_k$ and $c_i \succ_{x_j} c_l$ □

Definition 7.5.2 (Stable matching)**.** A matching M^* is said to be stable if it admits no blocking pair □

Four different cases may be considered regarding the König-Hall block for which a matching is sought:

1- *All variables appear in all constraints*

a- **Case 1:** The available knowledge enables the establishment of a strict suitability ordering for each variable and constraint.
This is an instance of SMP. The GS-algorithm adapted to our concern (see figure 7.1) finds a stable matching on this König-Hall block. Since this matching is complete and takes into consideration suitability ordering, it is also a solution to our problem.

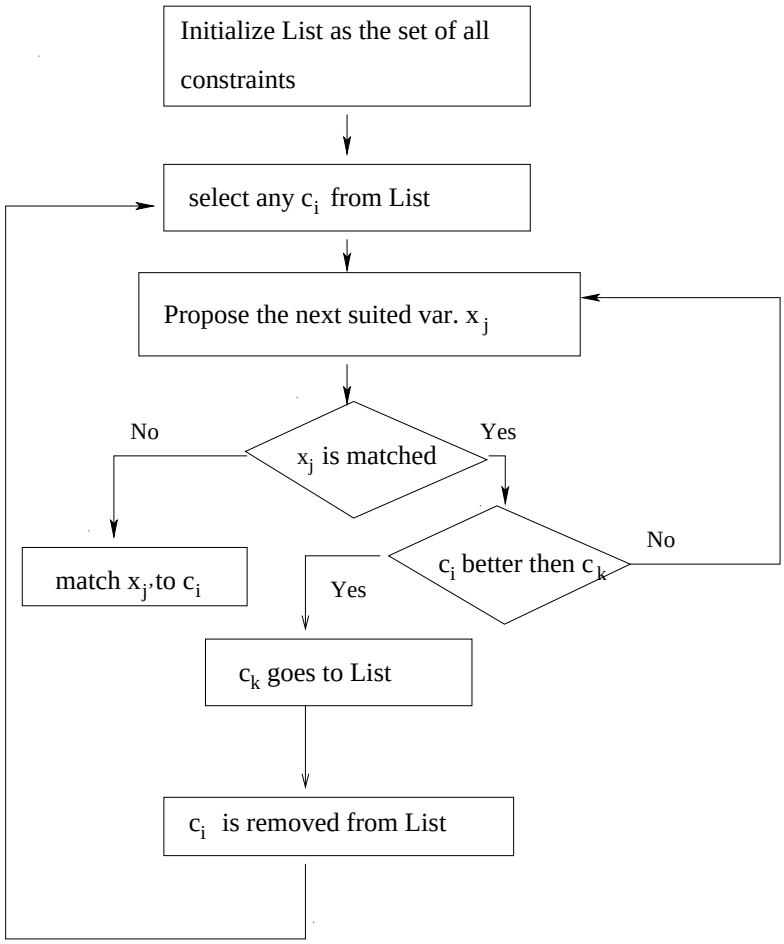

Figure 7.1: GS-algorithm flow-chart: adaptation to the residual generation problem

b- **Case 2:** A strict suitability ordering can not be defined.

This is an instance of SMPT. By arbitrarily breaking ties, the GS-algorithm can still be used to find a stable matching that is a solution to our problem.

2- *All variables do not appear in all constraints*

a- **Case 3:** It is possible to define a strict suitability ordering:

This is an instance of SMPI. The extended GS-algorithm cited in the previous subsection finds a stable matching but there is no guarantee that this is a complete matching.

b- **Case 4:** Strict suitability ordering can not be defined:

This is an instance of SMPIT. This problem requires a re-evaluation of the concept of blocking pair in order to apply the polynomial method that finds a strongly stable matching, cited in the previous subsection.

Definition 7.5.3 (Strongly blocking pair)**.** Let M be a matching. Suppose $(c_i, x_k) \in M$ and $(c_l, x_j) \in M$. The couple (c_i, x_j) is a strongly blocking pair either if $x_j \succ_{c_i} x_k$ and $c_i \succeq_{x_j} c_l$, or $x_j \succeq_{c_i} x_k$ and $c_i \succ_{x_j} c_l$ □

Definition 7.5.4 (Strongly stable matching)**.** A matching M^* is said to be strongly stable if it admits no strongly blocking pair □

A stable matching is also strongly stable but the reverse is not true. In fact, strong stability is a more appropriate criterion when there are ties in

the preference lists.

The strongly stable matching found in the case of SMPIT is again not guaranteed to be complete. Nevertheless, it is known in both cases that there exists at least one complete matching since the König-Hall block is strongly connected. It can then be concluded that the defined suitability ordering does not allow the finding of a (strongly) stable matching that is a solution to our problem, that is to say a matching that is complete. By modifying the partial order, a complete stable matching can be found.

Figure (7.2) recapitulates the main steps of searching the most suitable matching for residual generation in a König-Hall block.

Example 7.5.1 (Continuing:). Considering again the example, according to the partial order defined, it is an instance of SMPT. After arbitrarily breaking ties and applying algorithm (7.1), the first matching $\{(c_1, x_1); (c_3, x_2)\}$ is found to be stable. Indeed, c_1 is first in x_1's partial order and vice-versa, similarly for c_3 and x_2. Whereas in the other possible matching, (c_1, x_1) is a blocking pair (see figure 7.3).

□

Of course, in the general case there can be more than one stable matching. In fact, all stable matchings are equivalent according to our evaluation criteria.

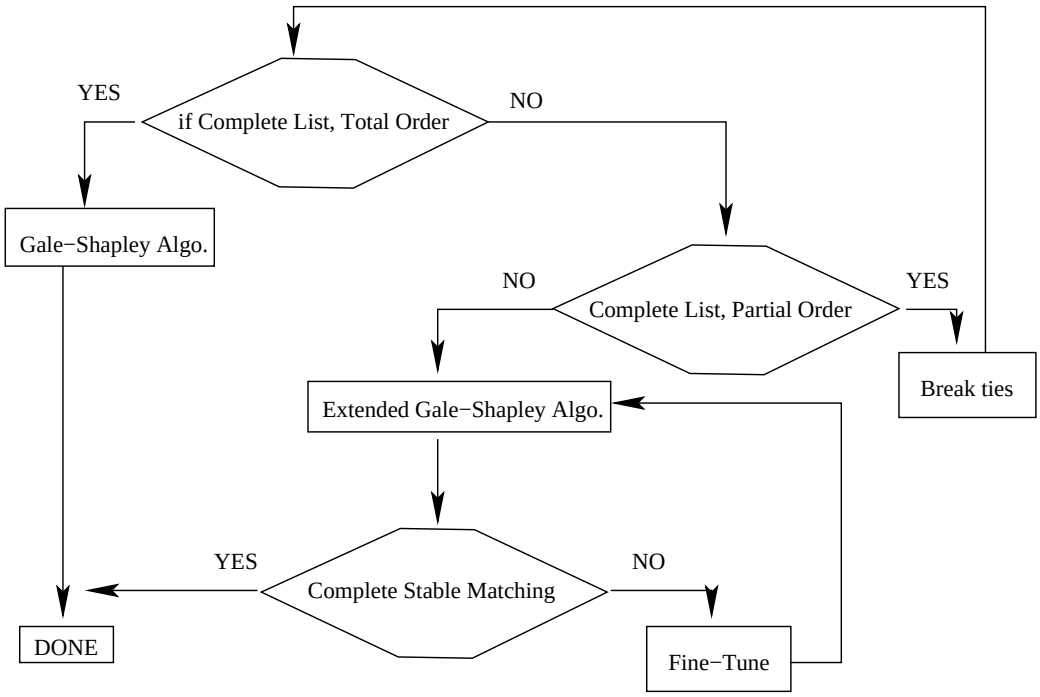

Figure 7.2: Matching research for residual generation: flow-chart

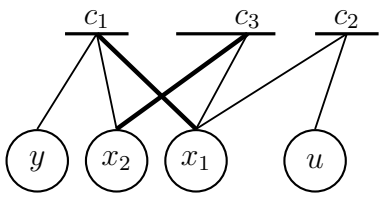

Figure 7.3: Example 7.5.1 : a stable pair

	c_1	c_2	c_3
x_1	1	3	2
x_2	1		1

Table 7.4: Example 7.5.2 : suitability matrix

7.5.3 Min. weight - max. cardinality matching

In this section, another approach is used to solve the residual generation problem. The partial order defined in section 7.4 can also be represented by a suitability matrix S, holding the rank of each constraint for the variables such that each row corresponds to unknown variables and each column to constraints in the over-constrained subsystem.

The suitability matrix entries can be also considered as a cost of matching on the edges $\gamma \in \Gamma$ between unknown variables and constraints from the over-constrained part. Suppose that for $(x_i, c_j) \in \Gamma$, c_j is at the k^{th} position in x_i's list, then $S(i,j) = k$ and the cost of the edge (x_i, c_j) is k. Of course, this is not the unique way to define the cost, any increasing function of the rank would be fine.

Example 7.5.2 (continuing). Based on the previously obtained preference lists, the suitability matrix of ((7.2.1),(7.2.2),(7.2.3)) is depicted in table 7.5.2.

The resulting weighted graph is shown in figure 7.4.

□

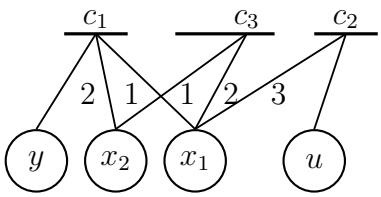

Figure 7.4: Example 7.5.2 : weighted graph

The overall cost of a matching M can then be defined as:

$$\sum_{(x_i,c_j)\in M} S(i,j). \tag{7.5.1}$$

We are then seeking the matching M^* that minimizes the overall cost. This is typically a *minimum weight maximum cardinality matching problem*.

Example 7.5.3 (Continuing). The previously found two matchings become as shown in figure 7.5 and 7.6.

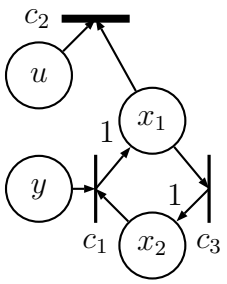

Figure 7.5: Example 7.5.3 : weighted matching (1)

The first matching gives an overall cost of 2, whereas the second cost is 3. Therefore, $M^* = \{(c_1, x_1); (c_3, x_2)\}$. □

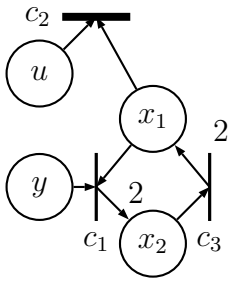

Figure 7.6: Example 7.5.3 : weighted matching (2)

Efficiency issues

The *minimum weight maximum cardinality matching problem* is a well-known problem in graph theory. The classical algorithm for the assignment problem is due to Edmonds [44, 43]. This algorithm is based on the works of Kuhn [84] and Munkres [97], which is also known as the Hungarian method. In addition, since it can be formulated as a network problem, the *Ford-Fulkerson maximum flow-minimum cut algorithm* may solve it [46]. Since then, a lot of work has been done to develop better solution procedures for the standard form of the assignment problem. For a survey, see [53].

Commercial computer code for the minimum weight maximum cardinality matching problem are readily available and able to solve large problems. In such problems, the number of edges (essentially) determines the solution time. The above mentioned algorithms run in $O(n^3)$ time with n being the number of edges in G.

7.5.4 Comparison of the two methods

In the previous section, two different existing algorithms have been adapted in order to solve our residual generation problem. Both methods have advantages and disadvantages.

The method proposed in figure 7.2 is sound, since in each possible case (SMP, SMPT, SMPI, SMPIT) the algorithms used are sound. Regarding the completeness of the method, the method might not find the complete matching in the case of SMPIT. In all the other three cases, the solution is found. In terms of efficiency, the algorithms applied in each possible case are individually polynomial in time as previously mentioned. However, considering the method in figure 7.2, the fine-tuning of the operation in the case of SMPIT might be time consuming.

As mentioned in the section 7.5.3, the *minimum weight maximum cardinality matching problem* is also solved in polynomial time.

The most important difference among the two methods is that *the stable marriage problem* has a qualitative nature due to the fact that it only needs a preference ordering, whereas the *minimum weight maximum cardinality matching problem* has a quantitative nature related to the numerical cost that needs to be assigned to the graph's edges. The cost assignment method proposed in section 7.5.3 is not unique, and any other combination of the ranking might function as well.

7.6 Conclusion

This chapter is concerned with structural analysis for residual generation by considering practical implementation issues. The aim is to show how available information can be incorporated into structural analysis in order to determine the most suitable matching to generate residuals. First, it is explained how to incorporate the available knowledge by defining partial orders on variables and constraints. Then, existing algorithms are adapted to solve the defined problem.

Chapter 8

Model evolution

Un algorithme adaptatif:

En cas de changements de structure du système, dûs par exemple à une perte ou un ajout de contraintes, l'approche actuelle nécessite de refaire l'analyse structurelle complète du système. Nous montrons dans ce chapitre que dans certains cas, il est possible d'utiliser les résultats de l'analyse structurelle obtenus pour le système initial en les actualisant suivant les contraintes concernées. Les algorithmes développés dans ce chapitre seront ainsi qualifiés d'*algorithmes adaptatifs*.

Le développement d'algorithmes adaptatifs se révèle aussi très important dans un contexte de conception de systèmes sûrs de fonctionnement. Un tel algorithme permet en effet de tester rapidement différentes configurations. Ainsi, on peut ajouter des capteurs, faire des hypothèses de défaillance,

modifier l'architecture matérielle du système.

La première étape d'un tel algorithme est de déterminer quelle partie du système initial est affectée par l'ajout ou la suppression de contraintes. Le paragraphe 8.2 étudie les différents cas de figures possibles. Au paragraphe 8.3, nous étudions le cas de suppression de contraintes. Le paragraphe 8.4 traite le cas d'ajout de capteurs, avec et sans ajout de nouvelles variables inconnues. Des algorithmes dédiés aux differents cas de figures qui adaptent de manière efficace les résultats de l'analyse précédente sont proposés.

8.1 Outline of the chapter

It is very likely that an industrial system's structure slightly changes or evolves during its life cycle. These modifications can be related to various causes:

- production objectives may change, it follows that some physical components may be added or suppressed,
- the technology of some equipments evolves that leads to replace them,
- component failure makes some subsystems temporarily unusable.

The current structural analysis approach consists in the duplication of the work from the beginning. Our idea is that this duplication is not always necessary because we can re-use structural results obtained from the previous run of the monitorability analysis algorithm (chapter 5). This may significantly improve efficiency of structural analysis with respect to model structure evolution. In this chapter, an adaptive/evolutive approach to the structural analysis method for FDI that enables the re-use of previously obtained structural results is proposed.

The benefit of such an adaptive algorithm is very important in a "safe system design" context. Indeed, such adaptive algorithms enable the rapid test of different configurations. The designer can add sensors, try different fault scenario, change the system's component architecture etc., efficiently and easily.

In order to propose such an adaptive algorithm, it is of prime importance to know how the initial system is affected by the structural modification. Constraint addition or removal does not have the same consequences and should be handled differently. Also, in both cases, the added (removed) constraint can add (remove) unknown variables.

In the following, section 8.2 analyzes the different cases that can occur. Then, section 8.3 treats the *constraint removal case* whereas 8.4 proposes an adaptive algorithm that makes efficient use of already-obtained structural information, in order to analyze the updated system structural FDI properties, for the *constraint addition case*. Finally, section 8.5 summarizes the whole analysis.

8.2 Possible cases

For any kind of modification that can affect the system's structure, we are looking for an efficient way to make use of previously obtained structural information, that is to say, the canonical decomposition and the already generated matchings. Any matching can be graphically represented as in figure 8.1.

The first type of modification that can occur is *constraint removal*. Clearly, such a modification will affect the system's FDI related properties only if the removed constraint is part of the over-constrained part S^+. For a given matching, constraint removal can occur in two different ways: either the re-

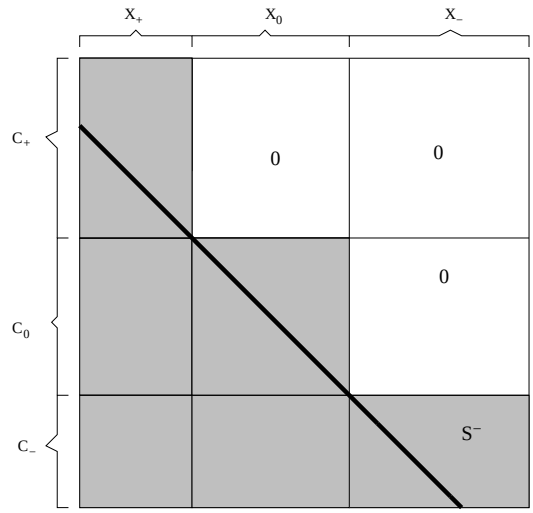

Figure 8.1: Matching: the general scheme

moved constraint is part of the matching (see figure 8.2), or it is a redundant constraint (see figure 8.3). Both cases have to be handled accurately and are explained in section 8.3.

The complementary case of *constraint removal* is *constraint addition*. This case is more subtle to handle because not only the over-constrained part, but also the just and under-constrained part of the system have to be considered. Moreover, added constraints can introduce new unknown variables in the system.

For a given matching, the added constraint c_a can have many different structure as shown in figures 8.4, 8.5, 8.6, 8.7, 8.8 and 8.9. All these cases are handled in section 8.4.

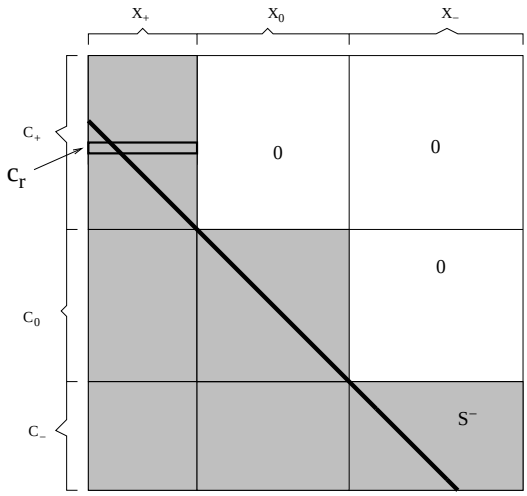

Figure 8.2: Constraint removal: c_r part of the matching

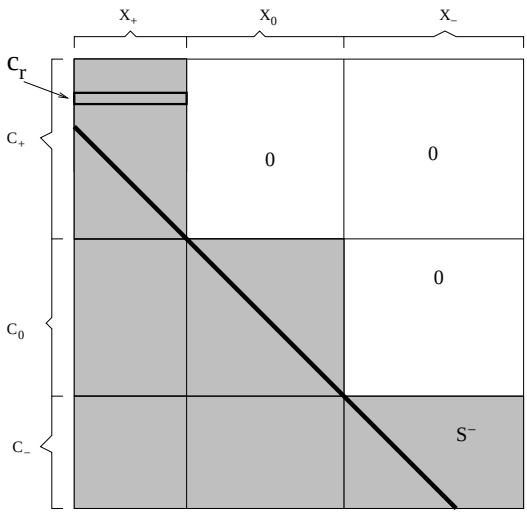

Figure 8.3: Constraint removal: c_r not matched

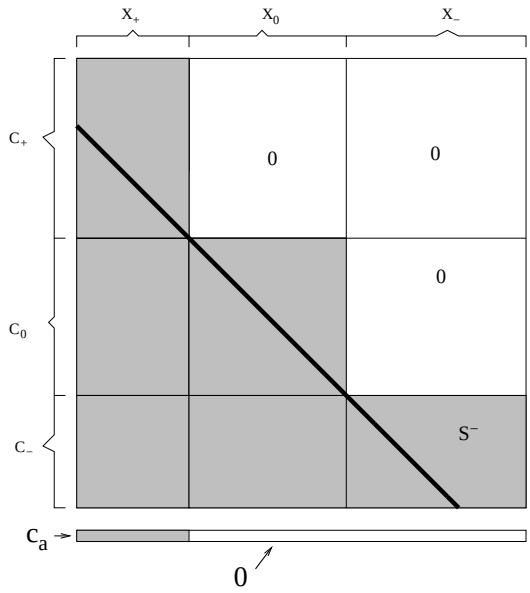

Figure 8.4: Constraint addition: c_a affects S^+

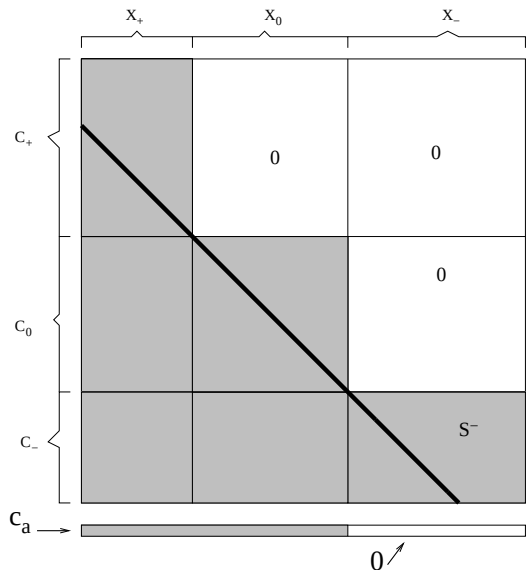

Figure 8.5: Constraint addition: c_a affects S^0

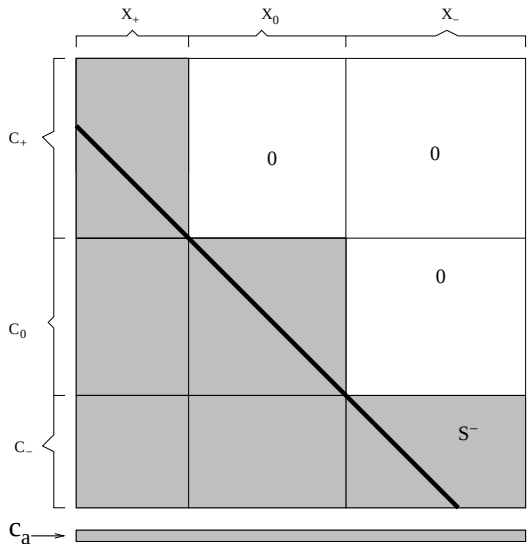

Figure 8.6: Constraint addition: c_a affects S^-

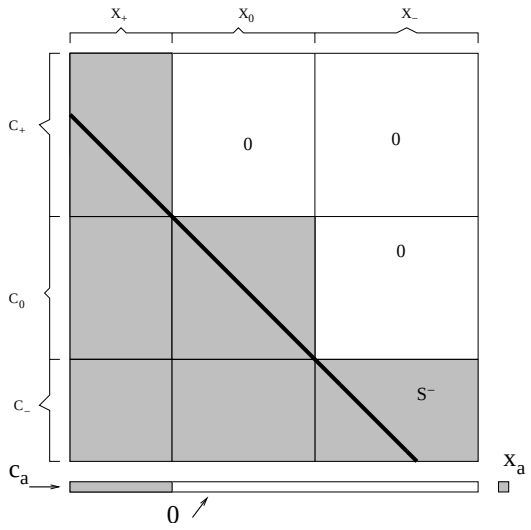

Figure 8.7: Constraint addition: c_a affects S^+ and introduces x_a

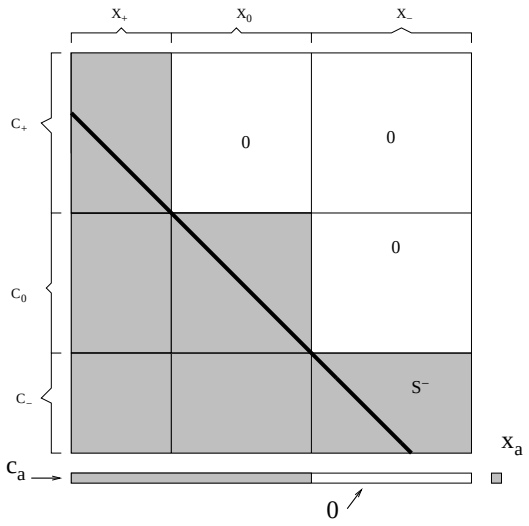

Figure 8.8: Constraint addition: c_a affects S^0 and introduces x_a

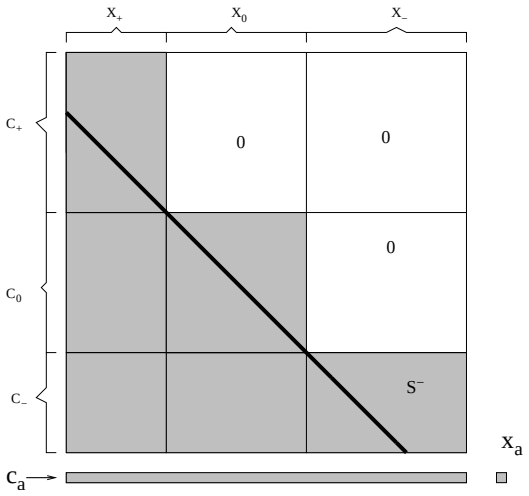

Figure 8.9: Constraint addition: c_a affects S^- and introduces x_a

8.3 Constraint removal

Let c_r be the removed constraint. As mentioned before, for any matching M_i obtained from the previous analysis, two situations are possible:

1- c_r is used in the matching (see figure 8.2),

2- c_r is not used in the matching, but is used as a redundant constraint to generate a residual (see figure 8.3).

Handling the second case is straightforward, and simply consists in discarding the residual generated upon c_r from the total set of residual signature obtained from the initial analysis of the system residual's. In practical words, this means removing the line c_r from M_i.

Regarding the case where c_r is part of the matching, obviously the matching M_i is not valid anymore after removing c_r. In order to decide to discard M_i or not, the following question has to be answered:

- *Are we losing any information by discarding M_i?*, in other words, , *is there any residual that can be generated only upon a matching where c_r is matched but is still valid after c_r is removed?*

The example 8.3.1 shows that the answer is positive.

Example 8.3.1. In table 8.1, a matching M_i where c_r is involved is shown. Based on M_i, two residuals can be generated by using the extra constraints c_5 and c_6. The computation sequence of the first residual (call it r_1) is

	x_1	x_2	x_3	x_4
c_1	(1)			
c_2		(1)	1	
c_r		1	(1)	
c_4	1			(1)
c_5		1		1
c_6	1			

Table 8.1: A matching M_i

$\{c_1; c_2; c_r; c_4; c_5\}$, whereas the computation sequence of the second one (call it r_2) is $\{c_1; c_6\}$.

After the removal of c_r, r_1 can not be generated. However, r_2 does not use c_r and can still be generated. Note also that the system's structure is such that all possible complete matchings on x_1, x_2, x_3 and x_4 use c_r. This example shows that discarding matchings where c_r is involved is not a solution in the sense that some still valid residual can be lost or ignored. □

This example shows that the brute force approach that consists in discarding any M_i of the type shown in figure 8.2 might ignore some residuals, and result in loss of some valuable information. Instead of discarding the type of matchings shown in figure 8.2, we propose to check each redundant constraint and to determine if it involves unknown variables that requires c_r in their computation sequences. If not, then the corresponding residual is still computable after c_r removal.

The method is explained step-by-step in table 8.2. In the following, we call M_{old} the set of matchings obtained by applying the monitorability analysis algorithm (table 5.5) to the initial system structure. The set of matchings obtained by applying the monitorability analysis algorithm to the modified system structure is called $M_{iterate}$. Here, note that, the removal of c_r can partition S^+ in a smaller over-constrained part and add some constraints to the just and under-constrained part. Finally, we call $M_{adaptive1}$, the set of matchings that is obtained by applying the adaptive algorithm in table 8.2 to M_{old}.

In order to prove that *Adaptive algorithm-1* is correct and complete, it is sufficient to show that it is correct and complete with respect to the monitorability analysis algorithm presented in table 5.5 (there-after called *Algorithm-2* for simplicity), since the later has been shown to be correct and complete at section 5.4.2.

The re-iteration of *algorithm-2* seeks all the complete matchings on the modified system, that is to say $S \setminus c_r$. Clearly, $M_{iterate} \subset M_{old}$. Besides, *adaptive algorithm-1* eliminates from M_{old} the matchings that are no more valid after c_r is removed.

8.4 Constraint addition

As depicted in figures 8.4, 8.5, 8.6, 8.7, 8.8 and 8.9, there are six different cases if constraint addition occurs. In the following, first the cases without

	Adaptive algorithm-1:
1-	Set $M_{adaptive1} := M_{old}$
2-	**For** each matching M_i in $M_{adaptive1}$,
	$M_{adaptive1} := M_{adaptive1} \setminus M_i$,
3-	**For** each removed constraint c_r,
	$M_i := M_i \setminus c_r$
4-	**If** c_r is matched in M_i,
5-	**For** each redundant constraint c_{red},
6-	**If** c_{red}'s variables use c_r,
	$M_i := M_i \setminus c_{red}$
7-	**End If**
8-	**end For**
9-	**End If**
10-	**end For**
11-	$M_{adaptive1} := M_{adaptive1} \bigcup \{M_i\}$
12-	**end For**
13-	Compute the modified *residual signature matrix* based on $M_{adaptive1}$

Table 8.2: Adaptive algorithm: constraint removal case

new unknown variable addition are handled (section 8.4.1). In this context, three different cases can be distinguished, depending on which part of the system is affected by the constraint addition. Finally, the new unknown variable addition case is addressed (section 8.4.2)

8.4.1 Constraint addition with no new unknown variable addition

As illustrated in figures 8.4, 8.5 and 8.6, it is crucial to determine which part of the system is affected by the addition of constraints in order to efficiently re-use already-obtained structural information. The following definitions are needed.

Suppose we have an initial system S whose Dulmage-Mendelsohn decomposition yields S^+, S^0 and S^- .

Definition 8.4.1. Let Q be a mapping between a set of constraints and a set of unknown variables:

$$\phi \mapsto Q(\phi) = \{x_i \in X : \exists c_j \in \phi \text{ s.t. } (x_i, c_j) \in \Gamma\}$$

Q associates with any subset of constraints ϕ, the subset of those unknown variables which intervene in at least one of them.

□

According to this definition and considering the sets of variables X_+, X_0 and X_- (see figure 5.3):

- $Q(S^+) = X_+$

- $Q(S^0) = X'_+ \cup X_0$ with $X'_+ \subseteq X_+$

- $Q(S^-) = X'_+ \cup X'_0 \cup X_-$ with $X'_+ \subseteq X_+$ and $X'_0 \subseteq X_-$

Definition 8.4.2. A system constraint c_a is said to belong to the over-constrained part S^+ if and only if:

$$(Q(\{c_a\}) \cap Q(S^+) \neq \emptyset) \wedge (Q(\{c_a\}) \cap Q(S^0 \cup S^-) = \emptyset) \tag{8.4.1}$$

□

In other words, a constraint c_a belonging to the over-constrained part involves only unknown variables from X_+.

Definition 8.4.3. A system constraint c_a is said to belong to the just-constrained part S^0 if and only if:

$$(Q(\{c_a\}) \cap Q(S^0) \neq \emptyset) \wedge (Q(\{c_a\}) \cap Q(S^-) = \emptyset) \tag{8.4.2}$$

□

In other words, a constraint c_a belonging to the just-constrained part involves at least one variable from X_0 and none from X_-.

Definition 8.4.4. A system constraint c_a is said to belong to the under-constrained part S^- if and only if:

$$Q(\{c_a\}) \cap Q(S^-) \neq \emptyset \tag{8.4.3}$$

□

A constraint c_a belonging to the under-constrained part involves at least one variable from X_-.

Based on these definitions, three cases can be distinguished:

- the added constraint(s) belongs to S^+,

- the added constraint(s) belongs to S^0,

- the added constraint(s) belongs to S^-.

The next section explains how these three cases differ and proposes dedicated algorithms to handle them.

Case 1 – Modification in S^+

The first case to consider is when modification occurs in S^+ (see figure 8.4). The added constraint c_a belongs to S^+.

We can easily see that S^0 or S^- are not affected at all by this upgrade. That is to say, no redundancy is added to S^0 and elements of X_- are still not computable. To be more precise, the monitorable part of the upgraded system is $S^+ \cup \{c_a\}$ and only this part is of further interest for FDI purposes.

The former run of our algorithm told us the ways to compute X_+'s by using S^+'s constraints. Therefore, we already know how to compute the unknown variables involved in c_a. Instead of applying back *algorithm-2* to

$S^+ \cup \{c_a\}$ and unnecessarily duplicate the whole work, we only have to propagate in c_a the computation sequences of $Q(\{c_a\})$ that have already been computed. By this way, the new residuals induced by the system structure modification are generated. In fact, the extra residuals are the added constraints.

Example 8.4.1 (Continuing - example 5.3.1, page 95). Considering our illustrative example, suppose that $c_6 : y = x_2$ is the new constraint added to the system structure during its life-evolution.

By considering c_6 as an extra equation in the 10 matchings yielded by the first run of *algorithm-2* (see page 111), and by propagating the computation sequence information of x_2 for each matching, 2 more residual structures are found: $\{c_1; c_2; c_3; c_6\}$ and $\{c_4; c_5; c_6\}$. □

In order to show the completeness of the proposed methodology, we have to answer the following question:

- *Does this approach generate all the possible extra residuals?*

The aim in asking this question is to ensure that no residual is missed, therefore the fault signature matrix generated is accurate and complete with respect to the system's structural FDI properties.

A negative answer to this question would imply that :

- *there exists a matching M_a where c_a is matched. This matching enables the generation of a residual r_a whose structure can not be found by the above explained approach.*

It is easier to visualize that this is not true by thinking in terms of bipartite graph instead of adjacency-matrix. As it has been mentioned previously, each matching induces some orientation in the bipartite graph G.

Suppose that such a matching M_a indeed exists. This means that c_a is used to compute one of its unknown variable, say x_a^i. This is graphically depicted by orienting all the edges from $(Q(\{c_a\}) - x_a^i)$ to c_a and by orienting the edge from c_a to x_a^i (see figure 8.10).

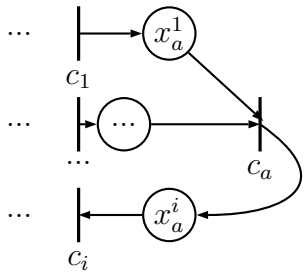

Figure 8.10: c_a used for a matching

By changing the orientation towards c_a, we will end with figure 8.11. That is to say, c_a is used as a redundant equation to generate a residual, say r_b. The set of nodes traversed before and after the edge direction inversion is the same. As a consequence, r_a's structure is equivalent to r_b's structure.

Since *algorithm-2* generates all the possible matchings, the matching corresponding to figure 8.11 has also been generated during the algorithm first run. By contradiction, it has been shown that M_a can not exist.

The algorithm to generate the updated fault signature matrix in case of constraint belonging to S^+ is summarized in table 8.3.

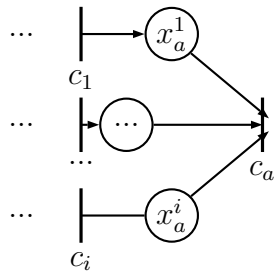

Figure 8.11: c_a used as a redundant equation

	Adaptive algorithm-2
1-	**For** each matching from the previous run: By making use of König-Hall blocks Derive the residual's signature based on c_a by the König-Hall blocks connectedness and order of resolution properties
2-	**end For**
3-	Update the *residual signature matrix*

Table 8.3: Adaptive algorithm: constraint addition affecting S^+

Case 2 – change in S^0

Suppose a set of constraints C_a belonging to S^0 is added to S (see figure 8.5). We can assert that S^- is not affected by this system upgrade. That is to say, elements of X_- are still not computable. However, redundancy is added to S^0 and this redundancy may be useful to generate new residuals.

In such a situation, the brute method would be to re-apply *algorithm-2* from the very beginning, that is to say application of the Dulmage-Mendelsohn decomposition to the upgraded system's structure $S \bigcup C_a$. Then seeking all matchings leading to different König-Hall configuration on the new over-constrained sub-system etc. (see table 5.5).

However, considering the computation ordering property induced by the block lower triangular form of the Dulmage-Mendelsohn decomposition, the first unknown variables to be computed in $S \bigcup C_a$ will be $Q(S^+)$, that is to say the unknown variables from the initial system's over-constrained part. This is a direct consequence of the fact that added constraints do not belong to S^+ but belong to S^0. In other words, the only set of equations needed to compute $Q(S^+)$ is the set of equations in S^+.

Consequently, without any loss of information, it is sufficient to only consider $S^0 \bigcup C_a$ by considering $Q(S^+)$ as known variables. Indeed, applying *algorithm-2* to $S \bigcup C_a$ is a time consuming operation that generates some information we already have.

Regarding $S^0 \bigcup C_a$, the addition of C_a to S^0 does not always make it

become over-constrained. In fact, it depends on C_a. However, some part of it has to become over-constrained due to the added redundancy (see example 8.4.2). By computing the canonical decomposition of $S^0 \bigcup C_a$, we can find this part (call it S^+_{new}). Then, the next step is to compute the different matching on S^+_{new} by applying *algorithm-2*. The main steps of handling this case are summarized in table 8.4. The algorithm is called with $S_{new} = S^0 \bigcup C_a$.

Example 8.4.2. In figure 8.12, the added constraint c_a does not make S^0 become over-constrained. However, we can see that $S^0_1 \bigcup c_a$ becomes over-constrained, whereas $S^0_2 \bigcup S^0_3 \bigcup ... \bigcup S^0_p$ remains just-constrained.

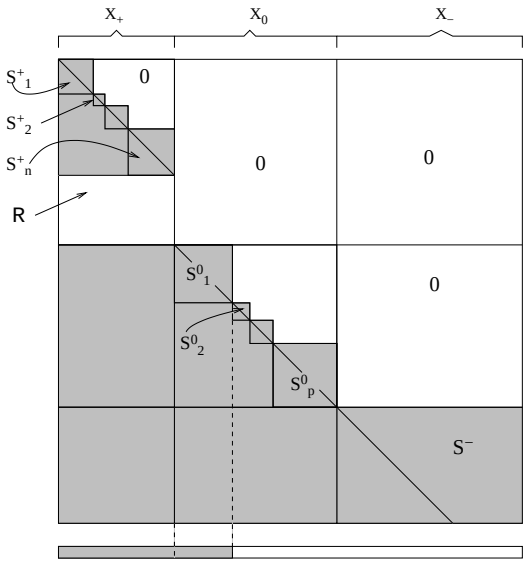

Figure 8.12: Constraint addition: S^0 does not become over-constrained by the added c_a

	Adaptive algorithm-3
1-	Compute the canonical decomposition of S_{new}
2-	Apply *algorithm-2* to S^+_{new}
	by considering $Q(S^+)$ as known
3-	**For** each row of the residual signature matrix:
4-	**If** any of $Q(S^+)$ variable is involved:
	* Add as many rows as different ways to compute the variable from $Q(S^+)$.
	* Derive the newly added residuals' signature by using information from initial run of *algorithm-2*
5-	**end If**
6-	**end For**
7-	Compute the *new residual signature matrix*

Table 8.4: Adaptive algorithm: constraint addition affecting S^0 or S^-

□

Case 3 – change in S^-

Suppose a set C_a belonging to S^- is added to the original system (see figure 8.6). Handling this case is very similar to the case shown in figure 8.5.

With the same reasoning on the computation ordering induced by the

block-lower triangular form of the canonical decomposition, dealing with only $S^- \bigcup C_a$ is sufficient, by supposing X_+ and X_0 known. The main difference is that adding any c_a belonging to S^- does not guarantee that any variable from X_- becomes computable and that any redundancy is added to the system. Therefore, the canonical decomposition of $S^- \bigcup C_a$ is performed to determine if the addition of constraint(s) generates an over-constrained sub-part. If the canonical decomposition of $S_{new} = S^- \bigcup C_a$ yields an over-constrained part (call it S^+_{new}), then the algorithm in table 8.4 is applied by defining $S_{new} = S^- \bigcup C_a$. Otherwise, this constraint addition does not yield any new residual. The same algorithm as for the previous case (table 8.4) is applied to $S_{new} = S^- \bigcup C_a$, and line 2 is applied only if $S^+_{new} \neq \emptyset$.

Constraint addition with no new unknown variable addition: The general case

By combining methods proposed throughout section 8.4.1, an adaptive / evolutionary algorithm that handles each possible case can be derived.

We have shown that the nature of the added constraint is important and there may be three cases:

- The added constraint belongs to S^+

- The added constraints belong to S^0

- The added constraints belong to S^-

All three cases have been analyzed with respect to their effects on the system's structure and on how to effectively make use of previously generated results. However, what if we have a set of added constraints, not all belonging to the same part? How can we combine *Algorithm-2* that generates all the possible matchings on the whole system, *Adaptive algorithm-3* (that applies *Algorithm-2* to a sub-part of the upgraded system only, in case of addition of a set of constraints belonging to S^0 or S^-) and *Adaptive algorithm-2* (that makes use of all already generated matchings in case of an added constraint belonging to S^+)?

Let C_a be the set of constraints added to the system. This set can be partitioned in three sub-sets C_a^+, C_a^0 and C_a^- as follows:

$$C_a^+ = \{c_a \in C_a \text{ s.t. } c_a \text{ belongs to } S^+\}$$
$$C_a^0 = \{c_a \in C_a \text{ s.t. } c_a \text{ belongs to } S^0\}$$
$$C_a^- = \{c_a \in C_a \text{ s.t. } c_a \text{ belongs to} S^-\}$$

C_a^+, C_a^0 and C_a^- have to be handled one by one in this order. The general case algorithm is summarized in figure 8.13 in terms of flow-chart.

In terms of complexity issues, the adaptive method that is proposed costs less than the "brute force" approach since it makes efficient use of already generated matchings instead of running *Algorithm-2* from the beginning and re-generating all the matchings. To be more precise, the worst case of the adaptive algorithm is equivalent to the complexity of *algorithm-2*. This would occur only if there are more than one added constraint belonging to S^+.

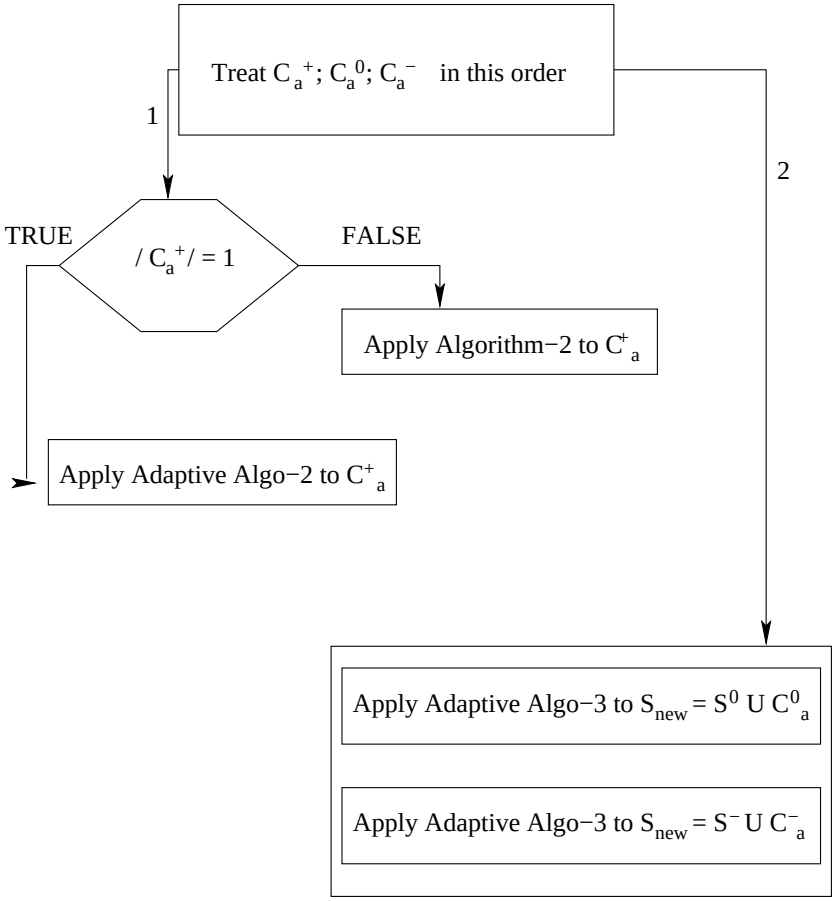

Figure 8.13: The adaptive algorithm: constraint addition - no new variable introduced

All the other possible cases are less complex since *Adaptive algorithm-3* has the same computational complexity as *Algorithm-2* but the input system is smaller. Finally, *Adaptive algorithm-2* is much less complex since the König-Hall configurations used in the *For-Loop* are those already generated by *Algorithm-2* whereas re-running *Algorithm-2* would imply to re-generate all of them.

8.4.2 Constraint addition with new unknown variable addition

Suppose the constraint c_{a1} is added to the system's model and suppose a new unknown variable x_a is introduced to the system, that is to say $Q(S \bigcup \{c_{a1}\}) = X \bigcup \{x_a\}$. In order to be able to use c_{a1} for residual generation purpose, the new unknown variable x_a has to be matched, that is to say we need to be able to compute it. That is why, we need at least one more constraint c_{a2} such that x_a occurs in c_{a2}.

In more general terms, let's call C_a the set of added constraints that introduces new unknown variables (call the set of introduced unknown variables X_a) to the system. In order to be able to use this set for residual generation purpose (hence generate new residuals), C_a has to have an over-constraint subpart when we decompose it canonically by supposing that $Q^+ \bigcup Q^0 \bigcup Q^-$ is known. This is a necessary but not a sufficient condition. In fact, it depends on the other unknown variables involved in C_a, that is to say $Q(C_a) \setminus X_a$,

that we supposed to be known. Two cases are possible:

- $Q(C_a) \setminus X_a$ belongs to S^+ (see figure 8.7) or S^0 (see figure 8.8), then $Q^+ \bigcup Q^0$ are computable independently of C_a. Therefore, we can suppose that $X_+ \bigcup X_0$ is known and generate new residuals on the over-constrained sub-part of C_a by applying *Algorithm-2* to the over-constrained part of C_a.

- $Q(C_a) \setminus X_a$ belongs to S^- (see figure 8.9), then X_- is not computable. New residuals can be generated if and only if the redundancy introduced by the the over-constrained part in C_a enables to match and compute X_-. Then, we can suppose that $X_+ \bigcup X_0$ is known and generate new residuals on $C_a \bigcup S^-$ by applying *Algorithm-2* to the over-constrained part of $C_a \bigcup S^-$.

8.5 Conclusion

In this chapter, the system's structure evolution problem during its life cycle has been emphasized. The actual approach is to apply the whole analysis from the very beginning to the modified system without making use of already-obtained results. The analysis of different possible cases accomplished throughout this chapter shows that:

- Constraint removal affects the FDI-related properties of the system if and only if the removed constraint belongs to S^+.

- In case of removal of constraint belonging to S^+, it is unnecessary to re-iterate the whole process of monitorability analysis from the beginning (*Algorithm-2*). By analyzing the previously generated matchings one-by-one, the already obtained information can efficiently be updated to fit the new system structure (*Adaptive algorithm-1*).

- The flow-chart in figure 8.13 shows how to handle constraint addition without new unknown variable addition. Only in case of constraint belonging to S^+, is it possible to fully make use of previously obtained matchings. However, in the other two cases (constraint belonging to S^0 or S^-), *Algorithm-2* is performed on only a sub-part of the system instead of the whole system. Considering the possible large size of complex systems, this may lead to an important decrease of complexity.

- The constraint addition with new unknown variables addition is very similar to the previous case, except the fact that added constraints have to provide means to redundantly compute the new variables (and, in some cases, variables from the under-constrained sub-part) in order to be able to generate new residuals.

Undoubtedly, these algorithms are less costly than performing the whole analysis from the very beginning.

Part III

A Case study

Introduction to Part III

Part III is the application of all the algorithms presented in part II to a new actuator benchmark for fault diagnosis studies: the DAMADICS valve benchmark model[1]. The benchmark is FDI method independent and based on an in-depth study of the phenomena that can lead to likely faults in valve actuator systems. The study uses a detailed consideration of the physics and electromechanical properties of the actuator together with typical engineering requirements of an actuator valve operating under challenging industrial conditions. More details on the benchmark can be found in the Control Engineering Practice special DAMADICS issue [3].

This real industrial application on the one hand illustrates the methods proposed all along the dissertation. On the other hand, this shows that the algorithms' implementation enables the simple treatement of real scale problems.

[1]EC FP5 European Research Training Network: Development and Application of Methods for Actuator Diagnosis in Industrial Control Systems, 1/7/2000 - 30/6/2005

Chapter 9

A Benchmark study

Une Application Industrielle - La vanne Damadics

On peut citer un certain nombres d'applications industrielles de l'analyse structurelle utilisant la représentation bipartie qui ont contribué à la diffusion de cette approche vers la communauté scientifique et vers l'industrie mais aussi révélé des points sur lesquels un travail théorique a été nécessaire.

- Une Centrale PWR 900 MW : l'analyse de la propriété structurelle de détection et isolation des défaillances [28, 30].
- Ship propulsion Benchmark : l'analyse de la propriété structurelle de détection et isolation des défaillances [22, 74, 90].
- L'unité de stock et de préparation d'une usine de papier en Australie : l'analyse de la propriété structurelle de détection et isolation des

défaillances [83].

- Le satellite RØmer : l'analyse de la propriété structurelle de détection et isolation des défaillances [90].

- La vanne du projet Damadics: l'analyse de la propriété structurelle de détection et isolation des défaillances [34], l'analyse de la propriété structurelle de l'isolabilité [51, 33, 37, 38], la génération de résidus "optimaux" [36, 35], et enfin l'application de l'algorithme adaptif.

- Le generateur à vapeur du projet CHEM: dans cette étude, l'analyse structurelle utilisant la représentation bond-graph et le graphe biparti ont été toutes deux utilisés dans le but d'étudier les propriétés structurelle de détection et isolation des défaillances [12, 13].

L'application au modèle de vanne constituant le *benchmark* d'un réseau européen de recherche - DAMADICS [1] - sera illustrée en détails dans ce chapitre. Cela permettra d'une part d'illustrer les résulats obtenus mais aussi de montrer que l'implémentation des algorithmes permet de traiter des problèmes à échelles réelles sans difficulté.

[1] *EC FP5 Research Training Network: Development and Application of Methods for Actuator Diagnosis in Industrial Control Systems*, 1 juillet 2000 - 30 juin 2003.

9.1 Outline of the chapter

First, in section 9.2, the model of the Damadics valve is presented. Then, in section 9.3, the structural analysis of the valve for the fault detectability and isolability assessment is performed. The algorithm presented in chapter 5 is applied and the results obtained are presented. Based on the obtained results, the fault isolability property of the valve is investigated in section 9.4. Moreover, the approach explained in chapter 6 is applied in order to determine which faults would need further modelling in order to improve the isolability. Finally, in section 9.5, the analysis is taken one step further in order to generate the optimal residual among the equivalent ones with respect to the system's practical constraints. For this purpose, both algorithms presented in chapter 7 are applied to the system.

9.2 Model of the Damadics valve

9.2.1 Description of the benchmark actuator

A schematic figure of the valve is shown in figure 9.1. The figure also shows an internal control loop that is used to increase the accuracy of the valve plug positioning.

The benchmark actuator belongs to the class of intelligent electro-pneumatic devices widespread in industrial environment. Furthermore, the actuator is considered as an assembly of devices consisting of:

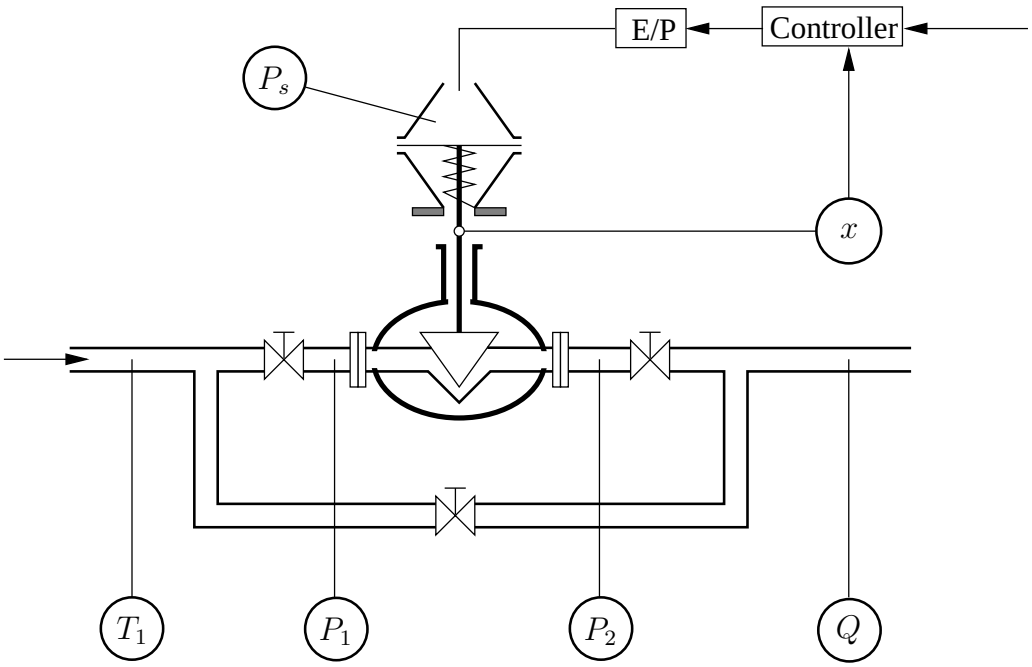

Figure 9.1: Schematic figure of the DAMADICS valve

- control valve,
- spring-and-diaphragm pneumatic servomotor,
- positioner.

The control valve acts on the flow of the fluid passing through the pipeline installation. A servomotor carries out a change in the position of the control valve plug, thus acting on fluid flow rate. A spring-and-diaphragm pneumatic servomotor is a compressible fluid powered device in which the fluid acts upon the flexible diaphragm, to provide linear motion of the servomotor stem. The positioner is a device applied to eliminate the control-valve-stem miss-positions produced by the external or internal sources such as: friction, clearance in mechanical assemblies, supply pressure variations, hydrodynamic forces, etc. Measured variables of the valve, indicated by circles, are the valve plug position x, the fluid flow Q, fluid temperature T_1, up- and downstream pressure of the valve P_1, P_2, and the transducer chamber pressure P_s.

Details of this model is not included in this chapter, only the structure of the model is described. Readers interested in details of this model are referred to [4, 3] and the references therein. The variables and the set of actuator faults definition, as well as the model equations are also given in appendix 10.2. Note that the constraint c_1 that describes the servomotor stem movement is derived from Newton's second law of dynamics, and involves $\ddot{x}$.

9.2.2 Spring-and-diaphragm pneumatic servo-motor

This component consists of an electro-pneumatic transducer providing linear motion to the valve-plug. Thus, the equations in this component describe the dynamics of the valve plug and the transducer chamber pressure P_s, which provides the main driving force of the plug.

The relative valve position x is a dynamic function of the pressure in the chamber and the opposing force F_{vc}, the vena-contracta force, i.e.

$$x = f(P_s, F_{vc})$$

This equation includes a model of the spring in the transducer and also friction components in the driving force. The pressure in the transducer chamber is a dynamic equation depending on the valve plug position (since this determines the effective volume of the chamber) and the net mass-flow of air Q_c into the chamber. The inlet flow is a dynamic function of the valve plug position controller CVI output and the chamber pressure. Thus, these models can be summarized as

$$P_s = f(x, Q_c)$$
$$Q_c = f(P_s, CVI)$$

9.2.3 Control and bypass valve

The valve equation describes the flow passing the valve, Q_v, and the vena-contracta force. Both entities are functions of the pressure upstream, P_1, and

downstream, P_2, of the valve, the fluid temperature T_1 and the valve-plug position.

$$Q_v = f(x, P_1, P_2, T_1)$$
$$F_{vc} = f(x, P_1, P_2, T_1)$$

The bypass valve is manually operated and only used when the flow passing the control valve becomes choked. The flow passing the bypass valve Q_{v3} obeys similar relations as the flow passing the control valve.

$$Q_{v3} = f(x_3, P_1, P_2)$$

where x_3 is the position of the manually operated by-pass valve.

9.2.4 Fault modelling

In [4], detailed fault models are described for 19 faults acting on the valve and its components. Typical faults in the valve are valve clogging, leakages, sensor faults, and different faults acting on the dynamics of the servo. In total, the control valve, the servo motor, and the positioner have 7, 4, and 4 modelled faults respectively. In addition 4 general/external faults are modelled. The faults are denoted $f_1, \ldots, f_{19}$ according to the order above.

9.2.5 Valve model summary

The number of model equations depends on what form and how many intermediate variables are used when forming the model. As a result of the

modelling, 19 equations are obtained (see appendix for the model equations). The equations c_1 and c_9 are dynamic since they involve time derivatives $\dot{x}$, $\ddot{x}$, $\dot{x_h}$ and $\dot{P}_s$. As explained previously, the derivation equation $\frac{d}{dt}$ is added for each time derivatives. As a consequence, the system structure is made of $19 + 4 = 23$ constraints. The variables in the equations include 15 unknown variables, 14 faults, and 9 known signals. The 9 known signals consist of 6 sensor signals, the valve-plug position controller reference value and output, and the position of the by-pass valve. The structure of the model is shown in table 9.1 for the unknown variables, table 9.2 for the faults, and table 9.3 for the known variables. The bipartite representation is not given here because the size of the system model makes it unreadable.

	x	$\dot{x}$	$\ddot{x}$	x_h	$\dot{x_h}$	P_s	$\dot{P_s}$	P_1	P_2	P_z	P_v	Δ_P	ΔP_a	Q	Q_v	Q_{v3}	Q_c	T_1	F_{vc}
c_1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
c_2	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c_3	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
c_4	0	0	0	1	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0
c_5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
c_6	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	0	0	0
c_7	0	0	0	1	0	0	0	1	1	0	1	0	1	0	0	0	0	0	1
c_8	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0
c_9	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0
c_{10}	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0
c_{11}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
c_{12}	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
c_{14}	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
c_{15}	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
c_{16}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
c_{17}	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
c_{18}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
c_{19}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c_{20}	Δ	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c_{21}	0	Δ	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c_{22}	0	0	0	Δ	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c_{23}	0	0	0	0	0	Δ	1	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.1: Damadics valve: structural model - unknown variables

f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}
0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	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
0	0	0	0	0	0	0	0	0	0	0	0	0	0	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	0	0	0	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	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	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	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	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	0	0	0
0	0	0	0	0	0	1	0	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	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.2: Damadics valve: structural model - faults

y_x	y_Q	y_{P1}	$y_P 2$	y_T	y_{ps}	C_{VI}	C_v	x_3
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	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	0	0	0	0	0	0	0
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	0
0	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	0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	1	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Table 9.3: Damadics valve: structural model - known variables

9.3 Fault detection and isolation assessment

In this section the structural fault detectability and isolability properties of the valve model are analyzed.

9.3.1 No faults decoupling

First, the lowest possible knowledge about faults is used in the analysis. In order to improve fault isolability, additional knowledge will be used, and faults isolation will be improved in Sections 9.4.

The DM-decomposition is applied to the structural model. An over-constrained block of dimension 19×15 and a just-constrained block of dimension 4×4 are obtained. The faults f_{16} and f_9 influence the just-constrained part, therefore no residual sensitive to them can be generated.

The next step consists in applying algorithm-2 from chapter 5, that is to say the different KH-configurations on the over-constrained block of the structural model are computed. Note that the application of algorithm-1 generates 532 possible matchings, whereas algorithm-2 considers only 205 of them because most of them are equivalent due to the connectedness property of KH-blocks.

No fault is considered as unknown disturbances to be decoupled. For each matching found, the fault sensitivity is computed. First the theoretical fault signature table is obtained (see table 9.4). Note that there are 23 different structures possible. To visualize fault isolability properties, the

fault isolability analysis matrix is computed based on table 9.4 and shown in table 9.5.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}
r_1	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
r_2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
r_3	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
r_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
r_7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
r_9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_{10}	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0
r_{11}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_{12}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
r_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
r_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_{16}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_{17}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
r_{18}	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
r_{19}	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
r_{20}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_{21}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_{22}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r_{23}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.4: Theoretical fault signature table

From the fault isolability matrix it is seen that without any fault decoupling:

1. Faults f_{16} and f_9 do not appear in table 9.5 because they are even not detectable.

2. The last four rows of the fault matrix show how faults f_7 (evaporation

	f_{19}	f_{18}	f_{17}	f_6	f_5	f_3	f_2	f_1	f_{14}	f_{11}	f_{10}	f_8	f_4	f_7	f_{12}	f_{13}	f_{15}
f_{19}	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_{18}	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_{17}	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_6	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_5	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_3	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_2	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_{14}	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
f_{11}	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
f_{10}	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
f_8	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
f_4	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
f_7	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
f_{12}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
f_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
f_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 9.5: Fault isolability matrix

in the control-valve), f_{12} (fault in the electro-pneumatic transducer), f_{13} (fault in the valve-plug displacement sensor) and f_{15} (positionner feedback fault) are isolable from all other faults without any further need of fault modelling or fault decoupling.

3. The two blocks of 8 and 5 faults respectively show groups of faults that are isolable from each other but where individual faults within each group is not isolable from the other faults in the group.

9.4 Fault isolability improvement

In this section, the results obtained in the last section are used in order to determine if fault isolability property can be increased. Further isolability analysis is concentrated on the second block of faults (f_4, f_8, f_{10}, f_{11}, f_{14}).

9.4.1 Faults decoupling

The next step in trying to increase isolability performance is to decouple faults in a group of non-isolable faults. If decoupling of one fault in the group of faults is possible without losing fault sensitivity to other faults in the group, isolability performance is increased. To analyze this, one fault at a time in the set $\{f_4, f_8, f_{10}, f_{11}, f_{14}\}$ is decoupled. That is to say one fault at a time is considered as an unknown variable and algorithm-2 is applied to the so-obtained modified system's structure. Then, all the possible KH-blocks configurations are generated in order to find if there is a new residual enabling to as in the previous section is performed.

Performing these operations reveals that, in this case, the structure of the system is such that decoupling of the faults does not provide any additional isolability properties. Indeed, when decoupling any one of the faults, the detectability of the other ones vanishes and therefore also the possibility to isolate these faults from each other.

9.4.2 Using additional fault models

Since faults decoupling cannot achieve the desired properties, additional information is needed. One way is of course to include additional sensors. If this is not possible, further modelling of the faults may be a solution.

According to figure 2 in [3], two kinds of faults are considered: abrupt and incipient faults. The following information is available related to the dynamics of faults considered in the benchmark:

Abrupt fault :

- $t < t_{from}$: $f = 0 \Rightarrow \dot{f} = 0$
- $t > t_{from}$: $f = \text{constant} \Rightarrow \dot{f} = 0$

Incipient fault :

- $t < t_{from} : f = 0 \Rightarrow \dot{f} = 0$
- $t \leq t_{from} < t_1 : \dot{f} = \text{constant} \Rightarrow \ddot{f} = 0$
- $t > t_1 : f = 1 \Rightarrow \dot{f} = 0$

Let us first consider the sensor fault f_{14} and assume this is an abrupt fault. We first introduce

$$\dot{f_{14}} = 0 \tag{9.4.1}$$

By considering f_{14} and $\dot{f_{14}}$ as two extra unknown variables, decoupling has been performed. A causal matching that enables the generation of a residual whose signature is sensitive to (f_4, f_8, f_{10}, f_{11}) but not sensitive to f_{14} has been found (see table (9.6)). It is then possible to isolate f_{14} from the other faults in the block by further modelling.

	T_1	P_2	x_h	P_v	Q	Δ_{Pa}	P_1	Q_{V3}	Q_V	Δ_P	F_{VC}	x	$\dot{x}$	$\ddot{x}$	P_s	f_{14}	$\dot{f_{14}}$
c_{16}	①																
c_{15}		①															
c_{12}			①														
c_5	1			①													
c_{13}					①												
c_4		1	1		1	①											
c_6		1				1	①			1							
c_8		1						①									
c_{11}								1	①								
c_3							1			①	1						
c_7			1							1	①						
c_2			1									①					
c_{21}												Δ	①				
c_{22}													Δ	①			
c_1											1	1	1	1	①		
c_{17}															1	①	
c_{25}																Δ	①

Table 9.6: A causal matching enabling to isolate f_{14}

By introducing model constraints of the form (9.4.1) for (f_4, f_8, f_{10}, f_{11}), and by performing the same decoupling procedure as in Section 9.4 a fault isolability matrix as in table 9.7 is obtained.

Therefore, it is seen that using additional fault models of f_4, f_8, f_{10}, f_{11}, and f_{14}, structural analysis shows that it is possible to isolate these faults from each other.

	f_{19}	f_{18}	f_{17}	f_6	f_5	f_3	f_2	f_1	f_{14}	f_{11}	f_{10}	f_8	f_4	f_7	f_{12}	f_{13}	f_{15}
f_{19}	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_{18}	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_{17}	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_6	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_5	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_3	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_2	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
f_{14}	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0
f_{11}	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0
f_{10}	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0
f_8	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0
f_4	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0
f_7	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
f_{12}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
f_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
f_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 9.7: Fault isolability analysis matrix after fault models introduction

9.5 Residual generation issues

The structural analysis, in the classical sense, of the Damadics valve that have been performed in the last few sections determines the monitorable part of the system, and allows the determination of the structure of residuals that can be generated. The fault signature matrix that is built based on this information enables the fault isolability analysis.

However, the important number of different matching possibilities in König-Hall components makes the residual generation task rather hard to perform. For example, consider the configuration in table (9.8), 3 residuals can be generated by replacing in c_{12}, c_{14} and c_{18} the computed values of x_h, P_1 and T_1 respectively. However, there exists 17 different matchings enabling to compute x_h and P_1 due to the König-Hall block of size 10. Its constraints are $\{c_1; c_2; c_3; c_4; c_6; c_7; c_8; c_{11}; c_{20}; c_{21}\}$. Actually, all the 17 matchings are giving residuals that are equivalent according to fault signature but with different computation sequences.

In order to apply the method proposed in section 7.5, preference lists of involved constraints and variables have to be defined first. Based on the system equations (table 9.9), Constraints Suitability Ordering (table 9.11) and Variables Suitability Ordering (table 9.10) have been obtained. The main ideas while defining the suitability partial order are:

- to avoid square root operations (c_3 and c_8)

- to avoid to have to inverse $hyst()$ since it is rather difficult (in c_1)

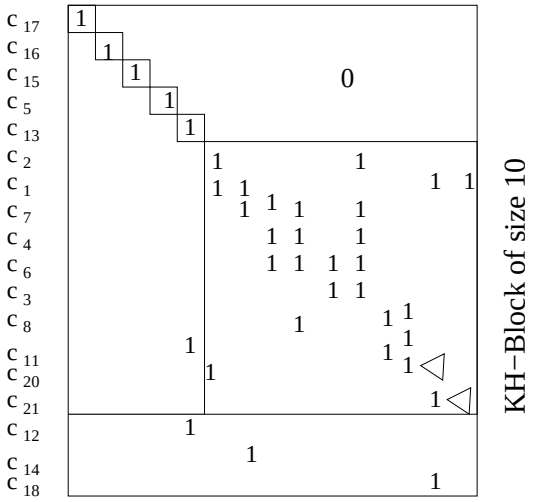

Table 9.8: Over-constrained subsystem of the Damadics valve

- to avoid computing $\Delta_{p-allow}$, P_1 and P_2 in constraints (c_6) and (c_7). Indeed, in (c_6) for example, the equation is $\Delta_p = P1 + P2$ if $P_1 - P_2 < \Delta_{P-allow}$ or it is $\Delta_{P-allow}$ otherwise. The dependance among $\Delta_{p-allow}$, P_1 and P_2 might make it difficult to compute any of these variables from (c_6) or (c_7)

9.5.1 The SMP algorithm

The over-constrained part of the Damadics valve structure is an instance of SMPIT. After applying the method explained in chapter 7.5, the matching given in figure (9.2) has been found. One can see that except P_1 that is the least suited variable to compute from c_4, all variables and constraints are matched to their most suited mate according to the defined partial suitability

$$
\begin{aligned}
P_s A_e(1-f_{10}) =& m\ddot{x} + k_v(1+f_4)\dot{x} + (c_1)(k_s(1+f_{11}) + k_d)x && (c_1)\\
&+ F_{vc} + (k_s(1+f_{11}) + k_d)x_0 - mg)\\
x_h =& hyst(x)[f_8 = 0] \quad \vee \quad hyst(x, f_8)[f_8 \neq 0] && (c_2)\\
Q_v =& 100(1-f_5)(1+f_1)K_v(x_h)\sqrt{\frac{\Delta_p}{\rho}} && (c_3)\\
\Delta_{p-allow} =& K_m(x_h)(P_1 - r_c(P_1)P_v) && (c_4)\\
\Delta_p =& (P_1+P_2)[P_1 - P_2 < \Delta_{p-allow}] \quad \vee && (c_6)\\
&\Delta_{p-allow}[P_1 - P_2 > \Delta_{p-allow}]\\
F_{vc} =& \pi r^2(P_1 - \frac{\Delta_p}{K_m(x_h)})[P_1 - P_2 < \Delta_{p-allow}] \quad \vee && (c_7)\\
&\pi r^2 P_v[P_1 - P_2 > \Delta_{p-allow}]\\
Q_{v3} =& K_{v3}(x_3(1-f_{18}))\sqrt{\frac{P_1 - P_2}{\rho}} && (c_8)\\
Q =& Q_v + Q_{v3} && (c_{11})\\
\ddot{x} =& \frac{d\dot{x}}{dt} && (c_{20})\\
\dot{x} =& \frac{dx}{dt} && (c_{21})
\end{aligned}
$$

Table 9.9: Damadics valve: over-constrained sub-system Equations

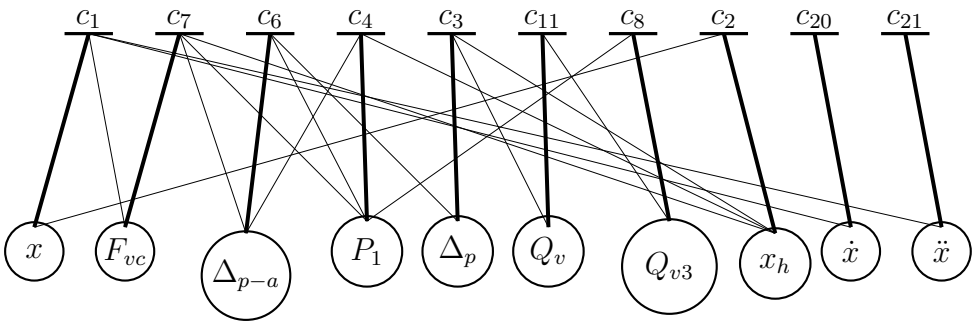

Figure 9.2: Damadics valve: a stable matching

orders.

9.5.2 Max. cardinality min. weight matching

In order to apply the second method proposed in section 7.5.3, the suitability matrix (table 9.12) has been obtained.

The second algorithm finds the same matching shown in figure 9.2. The total cost, based on the suitability matrix, of this matching is 12 (see table 9.13). Indeed, this is the minimal cost matching among all possible matchings. Note that the next optimal matching's cost is 13 (see table 9.14), whereas the worst matching's cost (maximal cost matching) is 17 (see table 9.15).

x :	$c_1 \succcurlyeq_x c_2$
F_{vc} :	$\{c_1, c_7\}$
$\Delta_{P-allow}$:	$c_6 \succcurlyeq_{\Delta_{P-allow}} c_4 \succcurlyeq_{\Delta_{P-allow}} c_7$
P_1 :	$\{c_6, c_7\} \succcurlyeq_{P_1} c_4 \succcurlyeq_{P_1} c_8$
Δ_p :	$c_6 \succcurlyeq_{\Delta_p} c_3$
Q_v :	$\{c_3, c_{11}\}$
Q_{v3} :	$\{c_8, c_{11}\}$
x_h :	$c_2 \succcurlyeq_{x_h} c_4 \succcurlyeq_{x_h} c_3 \succcurlyeq_{x_h} c_7$
$\dot{x}$:	$c_{20} \succcurlyeq_{\dot{x}} c_1$
$\ddot{x}$:	$c_{21} \succcurlyeq_{\ddot{x}} c_1$

Table 9.10: Damadics valve: variables suitability order

c_1 :	$\{x, F_{vc}, \dot{x}\}$
c_2 :	$x_h \succcurlyeq_{c_2} x$
c_3 :	$\Delta_p \succcurlyeq_{c_3} Q_v \succcurlyeq_{c_3} x_h$
c_4 :	$\Delta_{p-allow} \succcurlyeq_{c_4} x_h \succcurlyeq_{c_4} P_1$
c_6 :	$\{\Delta_{p-allow}, P_1, \Delta_p\}$
c_7 :	$F_{vc} \succcurlyeq_{c_7} P_1 \succcurlyeq_{c_7} \Delta_{p-allow} \succcurlyeq_{c_7} x_h$
c_8 :	$Q_{v3} \succcurlyeq_{c_8} P_1$
c_{11} :	$\{Q_{v3}, Q_v\}$
c_{20} :	$\dot{x}$
c_{21} :	$\ddot{x}$

Table 9.11: Damadics valve: constraints suitability order

	c_1	c_2	c_3	c_4	c_6	c_7	c_8	c_{11}	c_{20}	c_{21}
x	1	2								
F_{vc}	1					1				
$\Delta_{P-allow}$				2	1	3				
P_1				2	1	1	3			
Δ_p			2		1					
Q_v			1					1		
Q_{v3}							1	1		
x_h		1	3	2		4				
$\dot{x}$	2								1	
$\ddot{x}$	2									1

Table 9.12: Damadics valve: suitability matrix

	c_1	c_2	c_3	c_4	c_6	c_7	c_8	c_{11}	c_{20}	c_{21}
x	①	2								
F_{vc}	1					①				
$\Delta_{P-allow}$				②	1	3				
P_1				2	①	1	3			
Δ_p			②		1					
Q_v			1					①		
Q_{v3}							①	1		
x_h		①	3	2		4				
$\dot{x}$	2								①	
$\ddot{x}$	2									①

Table 9.13: Damadics valve: minimum cost matching of 12

	c_1	c_2	c_3	c_4	c_6	c_7	c_8	c_{11}	c_{20}	c_{21}
x	①	2								
F_{vc}	1					①				
$\Delta_{P-allow}$				②	1	3				
P_1				2	1	1	③			
Δ_p			2		①					
Q_v			①					1		
Q_{v3}							1	①		
x_h		①	3	2		4				
$\dot{x}$	2								①	
$\ddot{x}$	2									①

Table 9.14: Damadics valve: second minimal cost matching of 13

	c_1	c_2	c_3	c_4	c_6	c_7	c_8	c_{11}	c_{20}	c_{21}
x	1	②								
F_{vc}	①					1				
$\Delta_{P-allow}$				②	1	3				
P_1				2	1	1	③			
Δ_p			2		①					
Q_v			①					1		
Q_{v3}							1	①		
x_h		1	3	2		④				
$\dot{x}$	2								①	
$\ddot{x}$	2									①

Table 9.15: Damadics valve: maximum cost matching of 17

Part IV

Conclusions and perspectives

Chapter 10

Conclusions

10.1 Summary of contributions

Structural analysis is an important tool of interest in the early stage of the control and supervision system design, when detailed models are not available. This approach has reached a certain theoretical maturity and many real-life applications to various systems show its usefulness and applicability.

In this dissertation, algorithms dedicated to structural analysis for FDI purposes have been elaborated in view of easily and efficiently automating this method. For this purpose, first a set of characteristics that any structural FDI algorithm should meet has been proposed.

The algorithm that generates the residual signatures proposed in chapter 5 has been proved to meet all the desired requirements. It is sound, complete, and has polynomial time complexity.

The output of this first algorithm is further investigated in order to assess the system's fault isolability property. Moreover, the same output enables the determination of faults that require further modelling in order to meet the isolability requirements. The whole process is also shown to be sound and correct, though the complexity requirement might not be met.

The residual generation problem is addressed in chapter 7. First it is shown how available information can be incorporated into the current structural analysis framework. Then, two well-known algorithms are adapted in order to determine the most suitable matching. Although both proposed algorithms are sound, complete and polynomially complex, the stable marriage problem seems to be more adapted to the structural framework because of its qualitative nature.

Finally, the system's structure evolution problem during its life cycle is addressed. It is shown throughout the chapter that the duplication of the whole procedure is unnecessary. Depending on the type and number of added / removed constraints, the already-obtained results from the previous structural analysis of the system may be partially or fully used. Adaptive / evolutive algorithms that handle the different possible cases are presented. Undoubtedly, these algorithms are less costly than performing the whole analysis from the very beginning.

10.2 Further research perspectives

The system structure modification problem is important, especially with respect to its applicability to hybrid systems. Some further research should be done in order to determine how to efficiently apply adaptive / evolutive algorithms to the monitoring of hybrid systems. Indeed, hybrid systems are complex systems which have discrete event dynamics as well as continuous time dynamics. This broad class of systems includes continuous systems with phased operation, continuous systems controlled by discrete logic, and coordinating processes. In case of the existence of repeated structures and / or common pattern among different mode, we may consider each mode as a modified version of others. Thus, the adaptive / evolutive algorithm may take advantage of the structure of the problem and reduce the effort required in order to generate residuals for each mode.

Another interesting research area is to investigate how to combine the structural analysis framework with Artificial Intelligence (logic-based diagnostic) methods that have the advantage to handle multiple faults implicitly. Thus, no special care for isolation of multiple faults will be needed.

Finally, structural analysis can help during the system design phase to built an appropriate system architecture, by telling where to put sensors, in which part to introduce more redundancy, or which constraint to *prefer* for better fault detection and isolation. In other words, this method guides the design phase in order to achieve FDI requirements. Meanwhile, there are

other off-line methods that guide the construction of safe systems: fault trees, FMECA (Failure Mode, Effects and Criticality Analysis), etc. The system architecture obtained by this second type of methods is not always identical to the one obtained by structural analysis. An in-deep analysis of these system design methods can be very fruitful in order to integrate to the structural framework system evaluation criteria such as reliability, availability. Such a complementary approach would hopefully improve industrial systems safety thus optimizing maintainability without increasing costs.

Appendix A

DAMADICS Model equations

A.1 Variables

Variable	Description
x	Valve-rod displacement
P_s	Pneumatic servo-motor chamber pressure
F_{vc}	Vena-contracta force
P_1	valve upstream pressure
P_2	valve downstream pressure
T_1	valve upstream temperature
Q_v	Flow passing control valve
Q_{v3}	Flow passing bypass valve
Q	Total flow
CVI	Displacement controller output
P_z	Supply pressure for electro-pneumatic transducer
C_v	Commanded displacement (reference value)
y_x	Measured rod position
y_Q	Total flow measurement
y_{P1}	Valve upstream pressure measurement
y_{P2}	Valve downstream pressure measurement
y_T	Valve upstream temperature measurement

A.2 The set of actuator faults

	Control valve faults
f_1	Valve clogging
f_2	valve or valve-seat sedimentation
f_3	valve or valve-seat erosion
f_4	increase of valve or bushing friction
f_5	external leakage (bushing, covers, terminals)
f_6	internal leakage (valve tightness)
f_7	medium evaporation or critical flow
	Pneumatic servo-motor faults
f_8	twisted servo-motor piston rod
f_9	servo-motor housing or terminals tightness
f_{10}	servo-motor diaphragm perforation
f_{11}	servo-motor spring fault
	Positioner faults
f_{12}	electro-pneumatic transducer fault
f_{13}	rod displacement sensor fault
f_{14}	pressure sensor fault
f_{15}	positioner feedback fault
f_{16}	positioner supply pressure drop
	General/external faults
f_{17}	Unexpected pressure change across the valve
f_{18}	fully or partly opened bypass valves
f_{19}	flow rate sensor fault

A.3 Model Equations

Notation $[A]$ means 1 iff statement A is true, 0 otherwise. For example

$$[x > 0] = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$$

$$
\begin{aligned}
(c_1)\quad P_s A_e(1-f_{10}) &= m\ddot{x} + k_v(1+f_4)\dot{x} + (e_1)(k_s(1+f_{11})+k_d)x \\
&\quad + F_{vc} + ((k_s(1+f_{11})+k_d)x_0 - mg) \\
(c_2)\quad x_h &= hyst(x)[f_8 = 0] + hyst(x, f_8)[f_8 \neq 0] \\
(c_3)\quad Q_v &= 100(1-f_5)(1+f_1)K_v(x_h)\sqrt{\frac{\Delta_p}{\rho}} \\
(c_4)\quad \Delta_{p-allow} &= K_m(x_h)(P_1 - r_c(P_1)P_v) \\
(c_5)\quad \log(p_v) &= -\frac{a}{T_1} + b \\
(c_6)\quad \Delta_p &= (P_1+P_2)[P_1 - P_2 < \Delta_{p-allow}] + \\
&\quad \Delta_{p-allow}[P_1 - P_2 > \Delta_{p-allow}] \\
(c_7)\quad F_{vc} &= \pi r^2 (P_1 - \frac{\Delta_p}{K_m(x_h)})[P_1 - P_2 < \Delta_{p-allow}] \\
&\quad + \pi r^2 P_v [P_1 - P_2 > \Delta_{p-allow}] \\
(c_8)\quad Q_{v3} &= K_{v3}(x_3(1-f_{18}))\sqrt{\frac{P_1 - P_2}{\rho}} \\
(c_9)\quad Q_c K_c &= \dot{P}_s V(x_h) + P_s \frac{dV(x_h)}{dx}\dot{x}_h
\end{aligned}
$$

$$
\begin{aligned}
(c_{10}) \quad Q_c &= (1 - f_9 2.10^{-6}\sqrt{P_s})(1 - f_{10} 2.10^{-6}\sqrt{P_s})|CVI| \\
&\quad K_{c1}(P_z(1 - f_{16}) - P_8)[CVI \geq 0] - \\
&\quad |CVI| K_{c2} P_s [CVI < 0] \\
(c_{11}) \quad Q &= Q_v + Q_{v3} \\
(c_{12}) \quad y_x &= x_h(1 + 1.25 f_{13}) \\
(c_{13}) \quad y_Q &= Q(1 + f_{19}) \\
(c_{14}) \quad y_{p1} &= P_1 \\
(c_{15}) \quad y_{p2} &= P_2 \\
(c_{16}) \quad y_T &= T_1 \\
(c_{17}) \quad y_{ps} &= \max(0, \min(1, (1 + f_{14})P_8)) \\
(c_{18}) \quad T_1 &= T_{10}[f_7 = 0] + (T_{10} + 200 + 100 f_7)[f_7 \neq 0] \\
(c_{19}) \quad CVI &= (1 - f_{12}) K_p (C_v - f(y_x)) \\
(c_{20}) \quad \ddot{x} &= \frac{d\dot{x}}{dt} \\
(c_{21}) \quad \dot{x} &= \frac{dx}{dt} \\
(c_{22}) \quad \dot{x}_h &= \frac{dx_h}{dt} \\
(c_{23}) \quad \dot{P}_s &= \frac{dP_s}{dt}
\end{aligned}
$$

Bibliography

[1] Mohammed Assas, *Analyse de la tolérance aux fautes: approche fonctionelle et structurelle*, Ph.D. thesis, Université des sciences et technologies de Lille, 2002.

[2] M. J. Bagajewicz, *Process plant instrumentation: Design and upgrade*, CRC Press, 2000.

[3] M. Bartys, R. Patton, M. Syfert, S.de las Heras, and J. Quevedo, *Introduction to the damadics actuator* FDI *benchmark study*, Control Engineering Practice **In Press, Corrected proof available online**.

[4] M. Bartys and M. Syfert, *Using damadics actuator benchmark library (dablib)*, Final, v. 1.21, 2002.

[5] M. Basseville and I.V. Nikiforov, *Detection of abrupt changes - theory and application*, Information and System Sciences, Prentice-Hall, 1993.

[6] Michèle Basseville, *Detecting changes in signals and systems : A survey*, Automatica **24** (1988), no. 3, 309–326.

[7] Michèle Basseville, *An invariance property of some subspace-based detection algorithms*, IEEE Transactions on Signal Processing **47** (1999), no. 12, 3398–3400.

[8] R.V. Beard, *Failure accommodation in linear systems through self-reorganization*, Ph.D. thesis, MIT, 1971.

[9] Claude Berge, *Two theorems in graph theory*, Princeton University Press, 1957.

[10] ———, *Graphes et hypergraphes*, Dunod, 1973.

[11] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Fault diagnosis and fault tolerant control*, Springer-Verlag, 2003.

[12] B. Ould Bouamama, M. Staroswiecki, B. Riera, and E. Cherifi, *Multi-modelling of an industrial steam generator*, Control Engineering Practice **8** (2000), no. 11, 1249 – 1260.

[13] Belkacem Ould Bouamama, *Bond graph approach as analysis tool in thermofluid model library conception*, Journal of the Franklin Institute **340** (2003), no. 1, 1 – 23.

[14] T. Carpentier, R. Litwak, and J.Ph. Cassar, *Criteria evaluation of* FDI *systems application to sensors location*, Proceedings of IFAC/Safeprocess 1997, vol. 2, 1997, pp. 1083–1088.

[15] Thierry Carpentier, *Placement de capteurs pour la surveillance de processus complexes*, Ph.D. thesis, Université des sciences et technologies de Lille, 1999.

[16] J. Ph. Cassar and M. Staroswiecki, *A structural approach for the design of failure detection and identification systems*, Proceedings IFAC-IFIP-IMACS Conference on Control of Industrial Processes (Belfort, France), 1997.

[17] G. Chartrand and O. Oellermann, *Applied and algorithmic graph theory*, Pure and Applied Mathematics, 1992.

[18] J. Chen and R. J. Patton, *Robust model-based fault diagnosis for dynamic systems*, Kluwer Academic Publisher, 1999.

[19] E.Y. Chow and A.S. Willsky, *Analytical redundancy and the design of robust failure detection systems*, IEEE Transactions on Automatic Control **29** (1984), no. 7, 603 – 614.

[20] R.N. Clark, *The dedicated observer approach to instrument fault detection*, Proceedings of the 18th IEEE Conference on Decision and Control (CDC'79) (Ford Lauderdale, FL.), 1979, pp. 237 – 241.

[21] R.N. Clark, D.C. Fosth, and W.M. Walton, *Detecting instrument malfunctions in control systems*, IEEE Transaction on Aeronautical and Electrical Systems **11** (1975), no. 4, 465 – 473.

[22] V. Cocquempot, R. Izadi-Zamanabadi, M. Staroswiecki, and M. Blanke, *Residual generation for the ship benchmark using structural approach*, Proceedings of the International Conference on Control'98, 1998.

[23] C. Commault and J. M. Dion, *A system decomposition for failure detection and isolation*, Procedings of the 2nd IFAC Symposium on System,

Structure and Control (SSSC'04) (Oaxaca, Mexique), 8-10 december 2004.

[24] C. Commault, J. M. Dion, and S. Yacoub Agha, *A system decomposition for sensor location in fault detection and isolation*, Procedings of the 16th IFAC World Congress (Prague, Czech Republic), 4-8 july 2005.

[25] C. Commault, J.M. Dion, O. Sename, and R. Moteyian, *Observer-based fault detection and isolation for structured systems*, IEEE Transactions on Automatic Control **47** (2002), 2074–2079.

[26] C. Commault, J.M. Dion, O. Sename, and J.C. Avila Vilchis, *Fault detection and isolation : a graph approach*, Proceedings of the European Control Conference (ECC'99) (Karlsruhe, Germany), September 1999.

[27] M. O. Cordier, P. Dague, F. Lévy, J. Montmain, M. Staroswiecki, and L. Travé-Massuyès, *Conflicts versus analytical redundancy relations : A comparative analysis of the model-based diagnostic approach from the artificial intelligence and automatic control perspectives*, IEEE Transactions on Systems, Man and Cybernetics - Part B **34** (2004), 2163–2177.

[28] Ph. Declerck and M. Staroswiecki, *Characterization of the canonical components of a structural graph: Application to fault detection in large scale industrial plants*, Proceedings of European Control Conference (ECC'91) (Grenoble, France), 1991.

[29] Ph. Declerck and M. Staroswiecki, *Identification of structurally solvable subsystems for the design of fault detection and isolation schemes, using the embedding procedure*, Proceedings of the 9th IFAC/IFORS

Symposium on Identification and System Parameter Estimation (Budapest, Hungary), 1991.

[30] Philippe Declerck, *Analyse structurale et fonctionnelle des grands systèmes. application à une centrale* PWR 900 MW, Ph.D. thesis, Université des sciences et technologies de Lille, 1991.

[31] G. Delmaire, *Comparaison des méthodes d'identification paramétrique et de l'espace de parité pour la détection et la localisation de défaillances dans les systèmes automatisés*, Ph.D. thesis, Université des Sciences et Technologies de Lille 1, France, 1996.

[32] J.M. Dion, C. Commault, and J. van der Woude, *Generic properties and control of linear structured systems: a survey*, Automatica **39** (2003), no. 7, 1125 – 1144.

[33] D. Düstegör and V. Cocquempot, *Isolabilité structurelle des défaillances: application à un modèle de vanne*, Proceedings of Journées Doctorales d'Automatique 2003 (JDA'03) (Valenciennes FRANCE), 2003.

[34] D. Düstegör, V. Cocquempot, and M. Staroswiecki, *Structural analysis for fault detection and identification: an algorithmic study*, Proceedings of the 2^{nd} Symposium on System Structure and Control 2004 (SSSC'04) (Oaxaca, Mexico), 8-10 december 2004.

[35] ———, *Structural analysis for residual generation: Implementation issues*, Proceedings of IEE/Control 2004 (Hull UK), 2004.

[36] ———, *Structural analysis for residual generation: Towards implementation*, Proceedings of IEEE/Conference on Control Applications (CCA'04) (Taipei Taiwan), 2004.

[37] D. Düstegör, V. Cocquempot, M. Staroswiecki, and E. Frisk, *Isolabilité structurelle des défaillances: Application à un modèle de vanne*, Journal Européen des Systèmes Automatisés **38** (2004), no. 1-2, 103–124.

[38] D. Düstegör, E. Frisk, V. Cocquempot, M. Krysander, and M. Staroswiecki, *Structural analysis of fault isolability in the damadics benchmark*, Control Engineering Practice **in press, Corrected proof available online**.

[39] Bernard Dubuisson, *Diagnostic et reconnaissance des formes*, Hermes, Paris, 1990.

[40] A. L. Dulmage and N. S. Mendelsohn, *Covering of bipartide graphs*, Canadian Journal of Mathematics **10** (1958), 517–534.

[41] ———, *A structure theory of bi-partite graphs of finite exterior dimension*, vol. Section III, Trans. of Royal Soc. Canada, 1959.

[42] D. Dvorak and B. J. Kuipers, *Model-based monitoring of dynamic systems*, Proceedings of the 11th Joint Conf. on Artificial Intelligence (Detroit), 1989, pp. 1238–1243.

[43] J. Edmonds, *Maximum matching and a polyhedron with (0,1) vertices*, J. Res. Nat. Bur. Standards Sect. **8** (1965), 125–130.

[44] ———, *Paths, trees, and flowers*, Canadian Journal of Mathematics **17** (1965), 449–467.

[45] L. R. Ford, , and D. R. Fulkerson, *A simple algorithm for finding maximal network flows and an application to the hitchcock problem*, Canad. J. Math. **9** (1957), 210–218.

[46] L. R. Ford and D. R. Fulkerson, *Maximal flow through a network*, Canadian J. Math. **8** (1956), 399–404.

[47] O.I. Franfsen, P. Falster, and F.J. Evans, *Potential controllability and observability*, Tech. Report 7608, Electric Power Engineering Department - The Technical University of Denmark, 1976.

[48] P. M. Frank, *Fault diagnosis in dynamic systems using analytical and knowledgebased redundancy : A survey and some new results*, Automatica **26** (1990), no. 3, 459 – 474.

[49] P. M. Frank, *On-line fault detection in uncertain nonlinear systems using diagnostic observers : A survey*, Int. J. Systems SCI. **25** (1994), no. 12, 2129 – 2154.

[50] P.M. Frank and X. Ding, *Frequency domain approach to optimally robust residual generation and evaluation for model-based fault diagnosis.*, Automatica **30** (1994), no. 5, 789 – 804.

[51] E. Frisk, D. Düstegör, M. Krysander, and V. Cocquempot, *Improving fault isolability properties by structural analysis of faulty behavior models: application to the damadics benchmark problem*, Proceedings of IFAC/Safeprocess 2003 (Washington, USA), 2003.

[52] D. Gale and L.S. Shapley, *College admissions and the stability of marriage*, Naval Research Logistics **2** (1955), 83–97.

[53] Zvi Galil, *Efficient algorithms for finding maximum matching in graphs*, ACM Comp. Surv. **18** (1986), 23–38.

[54] A.L. Gehin, M. Assas, and M. Staroswiecki, *Structural analysis of system reconfigurability*, Proceedings of IFAC/Safeprocess 2000, vol. 1, 2000, pp. 292–297.

[55] A. Georgiou and C.A. Floudas, *Structural analysis and synthesis of feasible control systems : theory and applications*, Chemical Engineering Research and Design **67** (1989), 600–618.

[56] J. Gertler and R. Monajemy, *Generating directionnal residuals with dynamic parity relations*, Automatica **31** (1995), no. 4, 627–635.

[57] J. Gertler and D. Singer, *A new structural framework for parity space equation based failure detection and isolation*, Automatica **26** (1990), 381–388.

[58] Jonas Gertler, *Survey on model-based failure detection and isolation in complex plants*, IEEE Control System Magazine **8** (1988), no. 6, 3 – 11.

[59] ———, *Analytical redundancy methods in fault detection and isolation, survey and synthesis*, Proceedings of IFAC/Safeprocess 1991 (Baden Baden, Germany), vol. 1, 1991, pp. 9 – 21.

[60] K. Golver and L. M. Silverman, *Characterization of structural controllability*, IEEE Trans. Autom. Control **AC-21** (1976), no. 4, 534–537.

[61] M. Gondran and M. Minoux, *Graphes et algorithmes*, Eyrolles, 1995.

[62] D. Gusfield and R. Irving, *The stable marriage problem: Structure and algorithms*, The MIT Press, 1989.

[63] P. Hall, *On representatives of subsets*, J. of London Math.Soc. (1935), no. 10, 26–30.

[64] H. Hammouri, M. Kinnaert, and E.H. Yaagoubi, *Observer based approach to fault detection and isolation for nonlinear systems*, IEEE Trans. on Aut. Control **44** (1999), no. 10, 1879 – 1884.

[65] W. Hamsher, L. Console, and D. De Kleer, *Readings in model-based diagnosis*, Morgan Kaufman, 1991.

[66] F. Harary, *A graph theoretic approach to matrix inversion by partitioning*, Numer. Math. **4** (1962), 128–135.

[67] G. Hoblos, M. Staroswiecki, and A. Aitouche, *Optimal design of fault tolerant sensor networks*, Proceedings of the IEEE International Conference on Control Application (CCA'2000) (Anchorage, Alaska, USA), 2000, pp. 467–472.

[68] Ghaleb Hoblos, *Contribution à l'analyse de la tolérance aux fautes des systèmes d'instrumentation*, Ph.D. thesis, Université des sciences et technologies de Lille, 2001.

[69] J. Hopcroft and R. Tarjan, *Algorithm 447: Efficient algorithms for graph manipulation*, Comm. ACM (1973), no. 16, 372–378.

[70] J. E. Hopcroft and R. M. Karp, *An $n^{\frac{5}{2}}$ algorithm for maximum matchings in bi-partite graphs*, S.I.A.M. J. Comp. **2** (1973), 225–231.

[71] R. Iserman, *Process fault detection based on modeling and estimation methods: A survey*, Automatica **20-4** (1984), 387–404.

[72] ———, *Fault diagnosis of machines via parameter estimation and knowledge processing*, Automatica **29** (1993), 815–835.

[73] R. Izad-Zamanabadi and M. Staroswiecki, *A structural analysis method formulation for fault-tolerant control system design*, Proceedings of the 39th IEEE Conference on Decision and Control (CDC'00) (Sydney, Australia), December 2000.

[74] R. Izadi-Zamanabadi, P. Amann, M. Blanke, V. Cocquempot, G. Gissinger, E. Kerrigan, T. F. Lootsma, J. M. Perronne, and G. Schreier, *Control of complex systems*, ch. Ship Propulsion Control and Reconfiguration, Springer Verlag, 2001.

[75] R. Izadi-Zamanabadi and M. Blanke, *Ship propulsion system as a benchmark for fault tolerant control*, Control Engineering Practice **7** (1999), 227–239.

[76] ———, *Structural analysis for diagnosis - the matching problem revisited*, Proceedings of IFAC World Congress 2002 (Barcelona), 2002.

[77] Roozbeh Izadi-Zamanabadi, *Active fault-tolerant supervisory control - system analysis and logic design*, Ph.D. thesis, Dept. of Control Engineering, Aalborg University, Denmark, September 1999.

[78] H. L. Jones, *Failure detection in linear system*, Ph.D. thesis, MIT, Cambridge, MA, 1973.

[79] M. Kinnaert, *Robust fault detection based on observers for bilinear systems*, Automatica **35** (1999), 1829–1842.

[80] M. Krysander, J. Åslund, and M. Nyberg, *An efficient algorithm for finding over-constrained sub-systems for construction of diagnostic tests*, Proceedings of the 16th International Workshop on Principles of Diagnosis (DX'05), 2005.

[81] M. Krysander and M. Nyberg, *Fault diagnosis utilizing structural analysis*, CCSSE, October 2002.

[82] ———, *Structural analysis for fault diagnosis of DAE systems utilizing MSS sets*, Proceedings of the IFAC World Congress, July 2002.

[83] M. Krysander and M. Nyberg, *Structural analysis utilizing mss sets with application to a paper plant*, Proceedings of the 13th International Workshop on Principles of Diagnosis (DX'02), 2002.

[84] H. W. Kuhn, *The hungarian method for the assignment problem*, American Mathematical Monthly **69** (1962), no. 1, 9–15.

[85] A. Leitold and K.M. Hangos, *Structural solvability analysis of dynamic process models*, Computers and Chemical Engineering **25** (2001), 1633–1646.

[86] C.T. Lin, *Structural controllability*, IEEE Transactions on Automatic Control **19** (1974), no. 3, 201–208.

[87] ———, *System structure and minimal structuire controllability*, IEEE Trans. AC **22** (1977), no. 5, 855–862.

[88] X. Lin, M.O. Tade, and R.B. Newell, *Output structural controllability condition for the synthesis of control systems for chemical processes*, Int. J. Systems Sci. **22** (1991), no. 107-132.

[89] T. Lorentzen and M. Blanke, *Industrial use of structural analysis - a rapid prototyping tool in the public domain*, Proceedings of the ACD workshop 2004 (Karlsruhe), 17-18 November 2004 2004, pp. 166–171.

[90] T. Lorentzen, M. Blanke, and H. H. Niemann, *Structural analysis - a case study of the rømer satellite*, Proceedings of IFAC/Safeprocess 2003 (Washington DC), 2003.

[91] J.F. Magni and P. Mouyon, *On residual generation by observer and parity space approaches*, IEEE Trans. on Automatic Control **39** (1994), no. 2, 441 – 447.

[92] D. F. Manlove, *Stable marriage with ties and unacceptable partners*, Tech. Report TR-1999-29, Unversity of Glasgow - Computing Science Department, 1999.

[93] D. Maquin, M. Luong, and J. Ragot, *Fault detection and isolation and sensor network design*, European journal of automation **31** (1997), no. 2, 393–406.

[94] R. Mehra and J. Peschon, *An innovation approach to fault detection and diagnosis in dynamic system*, Automatica **7** (1971), 637 – 640.

[95] M. Meyer, J.M. Le Lann, B. Koehret, and M. Enjalbert, *Optimal selection of sensor location on a complex plant using a graph oriented approach*, Computer Chemical Eng. **18** (1994), S535–S540.

[96] M. Morari and G. Stephanopoulos, *Studies in the synthesis of control structures for chemical processes part ii : structural aspects and the synthesis of alternative feasible control schemes*, AIChE Journal **40** (1980), no. 2, 232–246.

[97] J. Munkres, *Algorithms for assignment and transportation problems*, J. Soc. Ind. Appl. Math. **5** (1957), 32–38.

[98] K. Murota, *Systems analysis by graphs and matroids, algorithm and combinatorics 3*, Springer-Verlag, 1987.

[99] S. Narasimhan and C. Jordache, *Data reconciliation and gross error detection: an intelligent use of process data*, Gulf publishing company, 1999.

[100] R. J. Patton, P. Frank, and R. Clark, *Fault diagnosis in dynamic systems*, Prentice Hall, 1989.

[101] R.J. Patton, P. Frank, and R. Clark, *Issues of fault diagnosis for dynamic systems*, Springer Verlag, 2000.

[102] H. Paynter, *Analysis and design of engineering systems*, MIT Press, 1961.

[103] B. Pulido and C. Alonso, *An alternative approach to dependency-recording engines in consistency-based diagnosis*, Lecture Notes in Artificial Intelligence, vol. 1904, 9th International Conference on Artificial Intelligence: Methodology, Systems, and Applications (AIMSA-00), Springer-Verlag, Berlin, Germany, 2000, pp. 111–121.

[104] J. Ragot, D. Maquin, and F. Kratz, *Analytical redundancy for systems with unknown inputs. application to faults detection*, Control Theory and Advanced Technology **9** (1993), no. 3, 775–788.

[105] K.J. Reinschke, *Multivariable control : a graph theoretic approach*, Vol. 108 of Lect. Notes in Control and Information Sciences, Springer-Verlag, 1988.

[106] C. Schizas and F.J. Evans, *A graph-theoretical approach to multivariable control systems design*, Automatica **17** (1981), 371–377.

[107] R.W. Shields and J.B. Pearson, *Structural controllability of multi-input linear systems*, IEEE Trans. Autom. Control **AC-21** (1976), no. 3.

[108] M. Staroswiecki, J. Ph. Cassar, and Ph. Declerck, *A structural framework for the design of* FDI *in large scale industrial plants*, Issues of fault diagnosis for Dynamic Systems, Springer Verlag, 2000.

[109] M. Staroswiecki and Ph. Declerck, *Analytical redundancy in non-linear interconnected systems by means of structural analysis*, Proceedings of IFAC/IMACS/IFORS Conf. AIPAC' 89 (Nancy, France), vol. 2, 1989, pp. 23–27.

[110] Marcel Staroswiecki, *Fault diagnosis and fault tolerant control*, ch. Structural Analysis for Fault Detection and Isolation and for Fault Tolerant Control, Encyclopedia of Life Support Systems (EOLSS), 2002, Developed under the auspices of the UNESCO, Eolss Publishers, Oxford, UK.

[111] D.V. Steward, *On an approach to techniques for the analysis of the structure of large systems of equations*, SIAM Review **4** (1962), 321–342.

[112] M. Tagina, J.Ph. Cassar, G. Dauphin-Tanguy, and M. Staroswiecki, *Bond-graph models for direct generation of formal fault detection systems*, Systems Analysis Modelling and Simulation J. **23** (1996), 1–17.

[113] R. E. Tarjan, *Depth-first search and linear graph algorithms*, SIAM J. Comput. (1972), no. 1, 146–160.

[114] L. Travé-Massuyès, T. Escobet, and R. Milne, *Model-based diagnosability and sensor placement. application to a frame 6 gas turbine subsystem*, Proceedings of the 12th international workshop on principles of diagnosis (DX'01) (D. T. Dupré S. Mcilraith, ed.), 2001, pp. 205–212.

[115] J. Unger, A. Kroner, and W. Marquardt, *Structural analysis of differential - algebraic equation systems - theory and applications*, Computers and Chemical Engineering **19** (1995), no. 8, 867–882.

[116] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, *A review of process fault detection and diagnosis part 1: Quantitative model-based methods*, Computers and Chemical Engineering **27** (2003), no. 5, 293 – 311.

[117] N. Viswanadham and R. Srichander, *Fault detection using unknown input observer*, Control Theory and Advanced Technology **3** (1987), 91 – 101.

[118] A.S. Willsky, *A survey of several failure detection method*, Automatica (1976), 601 – 611.

L'analyse structurelle est un outil puissant qui permet de déterminer de nombreuses propriétés intrinsèques d'un système dès la phase de conception. Ces propriétés sont obtenues à partir de la seule connaissance de l'existence de liens (contraintes) entre variables sans que les valeurs des paramètres soient nécessaires. L'analyse du modèle structurel, ou *analyse structurelle*, a été largement utilisée afin de déterminer les propriétés du système relatives à la surveillance : détection et localisation des défaillances. Ce mémoire traite des aspects algorithmiques de cette méthode afin d'améliorer son implémentation.
Dans une première partie, nous présentons les différentes approches de modélisation à caractère structurel rencontrées dans la littérature et justifions notre utilisation des graphes bipartis pour la surveillance. Nous recensons les propriétés désirées pour les algorithmes à implanter. La méthode d'analyse structurelle est ensuite détaillée et nous proposons un premier algorithme permettant d'évaluer les propriétés de détection et de localisation des défaillances. Nous abordons ensuite le problème de l'amélioration de la localisabilité des défaillances. L'algorithme proposé permet d'aboutir aux sous-ensembles de défaillances non localisables entre elles. Afin d'améliorer la localisabilité des défaillances, des capteurs supplémentaires peuvent être implantés. Nous proposons une autre solution qui consiste à utiliser des modèles d'évolution dynamique des défaillances lorsque ceux-ci sont disponibles. Nous considérons ensuite le problème de génération de signaux indicateurs de défaut, ou *résidus*. Un algorithme qui permet de générer la chaîne de calcul optimale par rapport à des contraintes de complexité d'implantation est alors proposé. Pour cela, le problème du mariage stable connu en informatique ainsi que le problème du choix de couplage maximal à pondération minimale sont adaptés au contexte de l'analyse structurelle. Enfin, un algorithme évolutif/adaptatif est décrit afin de tenir compte des évolutions de la structure du système lors de son cycle de vie. Tous les algorithmes développés sont évalués par rapport à l'ensemble de propriétés désirables et sont appliqués sur l'exemple d'un modèle de vanne.

Mots clés : Détection et localisation de défaillances, Surveillance, Relations de Redondance Analytiques, Résidus, Analyse structurelle, Graphe biparti.

Structural analysis is a powerful tool that allows to determine many system properties right during the design phase. Indeed, structural properties depend only on the existence of constraints between the system variables, and no knowledge of the exact model and of the values of the constraint parameters is necessary. Structural analysis has been widely used to exhibit the fault detection and isolation properties of a system. This work is concerned with the algorithmic point of view of structural analysis, i.e. it is concerned with implementation issues.
In the first part of the dissertation, the different structural models that are used in the literature are presented, and the choice of the bipartite graph representation is justified. The properties of the algorithms that are to be implemented are listed and explained. The structural analysis approach is then detailed. A first algorithm for the evaluation of the detection and isolation propertie of a system is proposed, and some improvements are then given. The algorithm creates a partition of the faults such that faults in the same class are not isolable from each other while faults in different classes are isolable. We show that fault isolability can be improved when dynamic models of the faults are available. The problem of residual computation is then addressed, and we propose an algorithm that allows to minimise a complexity index associated with the computation sequence that is chosen in the structural graph. This proposal follows from the adaptation to the structural analysis context of well known operations research problems, namely the «stable marriage» and the « minimal weight maximal matching » problems. Finally, the adaptation / evolution of our algorithms with respect to adaptations / evolutions of the system structure during its life cycle are addressed. All the developed algorithms are evaluated with respect to the set of desirable properties, and they are applied on the valve benchmark of the DAMADICS European Research and Training network.

Keywords : Fault detection and isolation, Diagnostics, Analytical redundancy relations, Residuals, Structural analysis, Bipartite graph.