Méthodologies et algorithmes pour l'analyse de la sûreté de fonctionnement des systèmes industriels complexes

E. Gascard

Université Grenoble Alpes

HABILITATION À DIRIGER DES RECHERCHES

Pour obtenir le diplôme

DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : **Automatique – Productique**

Arrêté ministériel : 25 mai 2016

Présentée par

Eric GASCARD

préparée au sein du **Laboratoire des Sciences pour la Conception, l'Optimisation et la Production de Grenoble (G-SCOP)**

dans **l'École Doctorale Electronique, Electrotechnique, Automatique et Traitement du Signal (EEATS)**

Méthodologies et algorithmes pour l'analyse de la sûreté de fonctionnement des systèmes industriels complexes

HDR soutenue publiquement le **14/12/2022**,
devant le jury composé de :

M. Frédéric KRATZ
Professeur des Universités, INSA Centre Val de Loire, Président

M. Benoît IUNG
Professeur des Universités, Université de Lorraine, Rapporteur

M. François PÉRÈS
Professeur des Universités, INP Toulouse - ENIT, Rapporteur

Mme Thao DANG
Directrice de recherche, CNRS délégation Alpes, Rapporteuse

M. Bruno CASTANIER
Professeur des Universités, Université d'Angers, Examinateur

M. Pierre DEHOMBREUX
Professeur des Universités, Université de Mons, Examinateur

Mme Zineb SIMEU-ABAZI
Professeure des Universités, Université Grenoble Alpes, Examinatrice

Remerciements

Je tiens tout d'abord à remercier les personnes qui ont permis la soutenance de cette HDR :

- *Benoît Iung*, *François Pérès* et *Thao Dang*, qui ont bien voulu être mes rapporteurs malgré leur charge de travail et qui ont apporté des visions complémentaires sur mon HDR : la sûreté de fonctionnement des systèmes industriels d'une part, les méthodes formelles appliquées aux systèmes cyber-physiques d'autre part.
- *Frédéric Kratz*, *Bruno Castanier* et *Pierre Dehombreux* qui ont accepté d'être membres de mon jury de soutenance et d'évaluer mon travail.
- *Zineb Simeu-Abazi* qui m'a accompagné en 2010 dans mon changement de thématique de recherche en me proposant le co-encadrement de stages sur la thématique de la sûreté de fonctionnement et avec qui j'ai depuis j'ai une collaboration fructueuse dans l'encadrement de stages de Master, de thèses et le développement de projet de recherche. Travailler en collaboration avec Zineb est très enrichissant scientifiquement, professionnellement et humainement. Notre complémentarité sur l'accompagnement des étudiants et nos compétences conjointes en automatique et en informatique nous ont permis de développer de très beaux projets de recherche et de former de nombreux étudiants au diagnostic, au pronostic et à la maintenance. Je remercie également très chaleureusement Zineb pour tous ses conseils, son soutien et la confiance qu'elle m'a accordée par nos nombreuses collaborations.

Je remercie tous les stagiaires et doctorants avec qui j'ai eu le plaisir de travailler et qui ont contribué aux résultats que je présente dans cette HDR.

Je remercie le laboratoire G-SCOP de m'avoir accueilli dès 2014 comme l'un de ses membres, ses chercheurs et les membres de l'équipe de recherche GCSP (renommée maintenant DOME2S) pour les échanges scientifiques, leurs conseils et les collaborations que nous avons pu menés ensemble.

Je remercie également tous mes collègues de Polytech Grenoble avec qui je travaille. Les équipes pédagogiques et administratives dans lesquelles j'interviens m'apportent beaucoup en pédagogie, plaisir d'enseigner et dans l'accompagnement de nos étudiants.

Enfin, je remercie ma famille pour son soutien indéfectible.

"La simplicité est la sophistication suprême."
Léonard de Vinci

Table des Matières

Avant-propos **1**

Introduction **3**

A) Méthodes de diagnostic basées sur les modèles **7**

I Construction automatique d'un diagnostiqueur de SED temporisés **11**

1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 11

1.1 Contexte 11

1.2 Etat de l'art, positionnement et originalité 11

2 Synthèse des travaux développés 14

2.1 Présentation synthétique d'une nouvelle méthode de diagnostic pour les systèmes à événements discrets temporisés 14

2.2 Illustration de notre méthodologie de construction d'un diagnostiqueur 16

3 Conclusion et perspectives sur la construction de diagnostiqueurs 24

II Etude de la diagnosticabilité d'un SED **27**

1 Présentation générale : contexte, état de l'art, positionnement 27

1.1 Contexte 27

1.2 Etat de l'art 27

1.3 Positionnement 31

2 Synthèse des travaux développés 31

2.1 Modélisation des systèmes manufacturiers non temporisés par automates à états finis 31

2.2 Présentation de l'étude de cas 32

2.3 Notations, définition de la diagnosticabilité 34

2.4 Test de diagnosticabilité 35

2.5 Illustration de l'étude de la diagnosticabilité 37

3 Conclusion et perspectives sur l'étude de la diagnosticabilité 38

III Génération de l'automate comportemental de SED temporisés **41**
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 41
2 Synthèse des travaux développés 42
2.1 Présentation de l'exemple illustratif 43
2.2 Modélisation du comportement dynamique du procédé par des règles de causalité 44
2.3 Construction de l'automate comportemental 45
3 Conclusion et perspectives sur la construction de l'automate comportemental 47

IV Diagnostic de systèmes dynamiques hybrides **49**
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 49
1.1 Contexte 49
1.2 Etat de l'art, positionnement, originalité 51
1.3 Positionnement, originalité 53
2 Synthèse des travaux développés 54
2.1 Diagnostic de systèmes dynamiques hybrides combinant bond graph, observateur et automate temporisé 54
2.2 Diagnostic de systèmes dynamiques hybrides combinant automate hybride et automate temporisé 61
3 Conclusion et perspectives sur le diagnostic de systèmes dynamiques hybrides 67

V Diagnostic de systèmes cyber-physiques **71**
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 71
1.1 Contexte, positionnement, originalité 71
1.2 Etat de l'art 75
2 Synthèse des travaux développés 76
2.1 Méthodologie de modélisation d'un système cyber-physique 76
2.2 Méthodologie de détection, localisation et identification des défaillances 79
2.3 Exemple applicatif de notre méthodologie de diagnostic de CPS : robot de téléprésence RobAIR 85
3 Conclusion et perspectives sur la modélisation et le diagnostic de systèmes cyber-physiques 96

VI Diagnostic de dérives temporelles dans un GET **99**
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 99
1.1 Contexte 99
1.2 Etat de l'art, positionnement, originalité 99
2 Synthèse des travaux développés 100
2.1 Modélisation d'un système de production par un graphe d'événement temporisé 100
2.2 Exemple illustratif 101
2.3 Présentation de notre méthodologie de diagnostic de fautes temporelles 102

3 Conclusion et perspectives sur l'identification de fautes temporelles 111

B) Outils pour le diagnostic et la maintenance 113

VII Usage des arbres de défaillances dynamiques pour la SdF 115

1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 115
1.1 Contexte 115
1.2 Présentation des arbres de défaillances statiques et dynamiques . . . 116
1.3 Etat de l'art sur l'analyse qualitative des arbres de défaillances dynamiques 120
1.4 Etat de l'art sur l'analyse quantitative des arbres de défaillances dynamiques 121
1.5 Positionnement et originalité de nos travaux 124
2 Analyse des causes racines des défaillances avec les arbres de défaillances dynamiques 126
2.1 Contexte, positionnement 126
2.2 Méthode d'analyse des causes racines avec les arbres de défaillances dynamiques 127
3 Analyse quantitative d'arbres de défaillances dynamiques 132
3.1 Contexte, positionnement 132
3.2 Analyse quantitative des arbres de défaillances dynamiques par simulation de Monte-Carlo 136
4 Simulation d'arbres de défaillances dynamiques : application au filtrage de fausses alarmes 144
4.1 Contexte, positionnement 144
4.2 Formalisation des portes temporelles et dynamiques avec des événements intermittents 145
4.3 Présentation de la boite à outils des arbres de défaillances temporels et dynamiques dans Matlab/Simulink 149
4.4 Exemple illustratif 150
5 Conclusion et perspectives sur l'usage des arbres de défaillances dynamiques pour la sûreté de fonctionnement 151

VIII Usage des files d'attente et des métaheuristiques pour la maintenance 155

1 Présentation générale : contexte, positionnement, originalité, état de l'art . . 155
1.1 Contexte, positionnement et originalité 155
1.2 Etat de l'art 157
2 Synthèse des travaux développés 158
2.1 La maintenance centralisée pour les systèmes de production multi-sites 158
2.2 Usage des réseaux de files d'attente pour le dimensionnement d'un atelier de maintenance 162

2.3 Usage des métaheuristiques pour l'optimisation des tournées de l'atelier de maintenance mobile . . . 170
3 Conclusion et perspectives sur la maintenance . . . 179

C) Pronostic de systèmes complexes par des méthodes basées sur les données 183

IX Pronostic de l'état de santé de systèmes complexes 185
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 185
1.1 Contexte . . . 185
1.2 Etat de l'art sur les méthodes de pronostic, positionnement . . . 187
2 Synthèse des travaux développés : pronostic de l'état de santé des batteries lithium-ion de véhicules électriques . . . 191
2.1 Données caractérisant l'état de santé des batteries lithium-ion . . . 191
2.2 Pronostic par modèle de Markov caché . . . 193
2.3 Pronostic par combinaison de modèle auto-regressif et régression à vecteur de support . . . 199
2.4 Cas d'études . . . 203
3 Conclusion et perspectives sur le pronostic de l'état de santé de systèmes complexes . . . 204

Perspectives de recherche 207

X Projets et ambitions de recherche 209
1 Projets de recherche . . . 210
1.1 Diagnostic de systèmes dynamiques complexes à base de modèles incomplets . . . 210
1.2 Optimisation de la maintenance distribuée par une approche guidée par les données . . . 211
1.3 Aide à la décision pour les systèmes cyber-physiques de production . 213
2 Ambitions de recherche . . . 214
2.1 Utilisation de la réalité virtuelle /réalité augmentée pour le diagnostic et l'aide à la maintenance d'équipement de production . . . 215
2.2 Prise en compte de l'humain dans la surveillance de production avec un robot collaboratif . . . 216

Conclusions générales 219

Références bibliographiques 221

Liste des figures

I.1 Diagnostic d'un système manufacturier 14
I.2 Principe de construction du diagnostiqueur 15
I.3 Exemple de procédé batch 19
I.4 Automate du contrôleur dans le procédé batch 20
I.5 Automates des vannes dans le procédé batch 21
I.6 Automates des capteurs de niveaux dans le procédé batch 21
I.7 Automates du comportement du procédé batch 22
I.8 Automate temporisé de l'esquisse du diagnostiqueur du procédé batch . 23
I.9 Automate temporisé du diagnostiqueur du procédé batch 24
I.10 Notre logiciel de génération automatique de diagnostiqueur 25
II.1 Exemple illustratif : système HVAC simplifié 33
II.2 Modélisation des composants du système HVAC simplifié 33
II.3 Système composite du système HVAC simplifié 33
II.4 Test de diagnosticabilité de la faute f_1 38
II.5 Test de diagnosticabilité de la faute f_2 38
III.1 Automates des capteurs de niveaux dans le procédé batch (sans fautes) 44
III.2 Esquisse de l'automate comportemental 45
III.3 Automate comportemental 47
IV.1 Structure d'un système dynamique hybride [100] 50
IV.2 Système dynamique hybride avec différents défauts [138] 54
IV.3 Diagnostic de SDH avec Bond graph, Observateur et Automate temporisé [142] 55
IV.4 Exemple d'un système hydraulique à deux réservoirs [142] 57
IV.5 Modèle Bond Graph du système hydraulique à deux réservoirs [142] . . 58
IV.6 Représentation d'état du système hydraulique à deux réservoirs [142] . . 58
IV.7 Définition de l'observateur du système hydraulique à deux réservoirs [142] 58
IV.8 Séquence de contrôle du système hydraulique à deux réservoirs [142] . . 59
IV.9 Diagnostiqueur par automate temporisé du système hydraulique à deux réservoirs [142] 60
IV.10 Exemple d'un automate hybride [139] 62
IV.11 Organigramme de la construction d'un diagnostiqueur [139] 64
IV.12 Construction du diagnostiqueur par un automate hybride global [143] . 65

IV.13 Exemple d'un système hydraulique à deux réservoirs [143] 66
IV.14 Automate hybride du système hydraulique [143] 66
IV.15 Modèle Stateflow du processus hydraulique [143] 67
IV.16 Liste des défauts pouvant affecter le système hydraulique [139] 67
IV.17 Automate hybride global du système hydraulique [143] 68
V.1 Vision holistique des systèmes cyber-physiques [153] 72
V.2 Principes des systèmes cyber-physiques [154], [155] 73
V.3 Principe général de réduction des ambiguïtés de localisation de défaillance [148] 80
V.4 Domaines d'application du RobAIR [148], [175] 86
V.5 Système de diagnostic appliqué au CPS RobAIR [148], [175] 86
V.6 Technologies et composants du robot RobAIR [148], [175] 87
V.7 Diagramme de relations entre RobAIR et le monde physique [148], [175] 87
V.8 Diagrammes FAST pour les fonctions de service de déplacement et de communication [148], [175] 88
V.9 Modélisation structurelle du RobAIR [148] 88
V.10 Modélisation fonctionnelle du RobAIR [148] 89
V.11 Modélisation topologique du RobAIR [148] 89
V.12 Extrait de l'analyse AMDEC du RobAIR [148], [175] 90
V.13 Cas d'usage du RobAIR [148], [175] 91
V.14 Phases de trajectoire du RobAIR sur le cas d'usage [148], [175] 91
V.15 Principe de modélisation du RobAIR [148], [175] 91
V.16 Modélisation sous Simulink de la partie Process du RobAIR [148], [175] 92
V.17 Extrait du modèle Stateflow de la commande du RobAIR [148], [175] . . 92
V.18 Caractérisation des modes de défaillances du RobAIR [148], [175] 93
V.19 Bloc Process avec prise en compte de l'injection de défaillances [148], [175] 93
V.20 Classe d'équivalence selon les variables dynamiques et les états des bits de la carte MD49 [148], [175] 94
V.21 Temporisation des phases de commande du RobAIR [148], [175] 95
V.22 Extrait du bloc Stateflow d'identification des défaillances [148], [175] . . 95
V.23 Modélisation de RobAIR avec injection de défaillances et son diagnostiqueur [148], [175] 96
VI.1 Représentation d'une chaîne de production par un GET 102
VI.2 Structure de la méthode de diagnostic de fautes temporelles 103
VII.1 Exemple d'un arbre de défaillances d'un ascenseur 117
VII.2 Portes dynamiques [198], [199] 118
VII.3 Exemples de chronogrammes de portes OR et AND 130
VII.4 Règles de remplacement de portes dynamiques 131
VII.5 Dépendance circulaire introduite par les portes FDEP 134
VII.6 Exemples de situations non-déterministes causées par la porte FDEP . . 135
VII.7 Exemple de deux portes SPARE partageant un composant de rechange . 136
VII.8 DFT of the CPS [232] 142

VII.9 DFT of the CAS [262] . . . 142
VII.10 DFT of the MDCS [232] . . . 142
VII.11 DFT of the SAP [288] . . . 142
VII.12 DFT of the FTPP [199] . . . 143
VII.13 Chronogramme de la porte PAND . . . 146
VII.14 Automate de Moore de la porte PAND . . . 146
VII.15 Chronogramme de la porte $DUR_{>}$ - scénario ① . . . 147
VII.16 Chronogramme de la porte $DUR_{>}$ - scénario ② . . . 147
VII.17 Automate de Moore de la porte $DUR_{>}$ . . . 148
VII.18 Chronogramme de la porte $DUR_{<}$ . . . 148
VII.19 Automate de Moore de la porte $DUR_{<}$ . . . 148
VII.20 Fenêtre de paramétrage de la porte DUR . . . 149
VII.21 Boite à outils des arbres de défaillances dynamiques et temporels développée dans Matlab/Simulink . . . 149
VII.22 Exemple de trace de simulation . . . 150
VII.23 Arbre de défaillance dynamique et temporel dans Matlab/Simulink . . . 151
VII.24 Résultat de la simulation de l'arbre de défaillance dynamique et temporel 152
VIII.1 Relations entre les différents sites de production et les ateliers de maintenance [298] . . . 156
VIII.2 Cycle de réparation d'un équipement défectueux dans un atelier de maintenance [327] . . . 159
VIII.3 Processus de maintenance corrective avec un atelier de maintenance central [327] . . . 160
VIII.4 Processus de maintenance corrective par réparation par remplacement [327] . . . 161
VIII.5 Représentation d'une station (file d'attente avec un serveur unique) [324] 162
VIII.6 Exemple d'organisation d'un atelier de maintenance [327] . . . 163
VIII.7 Réseau de files d'attente pour un atelier de maintenance [327] . . . 163
VIII.8 Notre toolbox de simulation de réseaux de file d'attente [326], [327] . . . 166
VIII.9 Synchronisation entre la sortie d'une station est une demande d'équipement . . . 167
VIII.10 Simulation des temps de séjours moyens des stations et de l'atelier de maintenance [327] . . . 167
VIII.11 Atelier de maintenance central avec station M_5 dupliquée [327] . . . 167
VIII.12 Simulation des temps de séjours moyens quand la station M_5 est dupliquée [327] . . . 168
VIII.13 Configuration de l'AdM central avec le niveau 1 de remplacement [327] . 169
VIII.14 Simulation des temps de séjours moyens avec le niveau 1 de remplacement [327] . . . 169
VIII.15 Configuration de l'AdM central avec le niveau 2 de remplacement [327] . 170
VIII.16 Simulation des temps de séjours moyens avec le niveau 2 de remplacement [327] . . . 170

IX.1 Chronologie de la détection, pronostic et diagnostic des défaillances [341] adaptée de [3] 186
IX.2 Principe du pronostic basé sur un modèle physique [349] 188
IX.3 Principe du pronostic guidé par les données [364] 189
IX.4 Structure d'un système de gestion des batteries [388] 192
IX.5 Modèles probabilistes et non probabilistes basés sur les données pour la prédiction de l'état de santé [387] 193
IX.6 Etape d'apprentissage des modèles de Markov cachés [382], [383] 198
IX.7 Etape d'identification du modèle de Markov caché et du diagnostic du SoH [382], [383] 199
IX.8 Organigramme de fonctionnement de la méthode de Box et Jenkins [418] 201
IX.9 Création de la base de données de modèles SVR [385] 202
IX.10 Identification du modèle meilleur modèle SVR [385] 203

Liste des algorithmes

I.1 Construction de l'esquisse du diagnostiqueur 17
I.2 Construction du diagnostiqueur 18
II.1 Test de diagnosticabilité 36
III.1 Construction de l'esquisse de l'automate comportemental du procédé . . 46
III.2 Construction de l'automate comportemental du procédé 48
V.1 Identification des ambiguïtés de localisation des composants pour le sous-système SP_i 81
V.2 Réduction des ambiguïtés de localisation des composants pour le sous-système SP_i 82
VI.1 Simulation conduite par les événements d'un GET 106
VI.2 Détection de dérives temporelles dans un GET 107
VI.3 Localisation de dérives temporelles dans un GET 108
VI.4 Identification de dérives temporelles dans un GET 110
VII.1 Simulation conduite par les événements d'un DFT 140
VII.2 Analyse quantitative d'un DFT par simulation de Monte Carlo 141
VIII.1 Calcul du coût d'une tournée de l'atelier de maintenance mobile 175
VIII.2 Algorithme génétique pour le problème de tournée de l'atelier de maintenance mobile 178
VIII.3 Algorithme du recuit simulé pour le problème de tournée de l'atelier de maintenance mobile 180

Avant-propos

Ce manuscrit a été rédigé en vue d'obtenir l'*Habilitation à Diriger des Recherches* (HDR). C'est un diplôme national de l'enseignement supérieur qu'il est possible d'obtenir après le doctorat. Il permet à son titulaire de diriger des thèses, d'être choisi comme rapporteur de thèse et également d'être candidat à l'accès au corps des professeurs des universités.

Le texte définissant l'HDR précise que ce diplôme sanctionne la reconnaissance du haut niveau scientifique du candidat, du caractère original de sa démarche dans un domaine de la science, de son aptitude à maîtriser une stratégie de recherche dans un domaine scientifique ou technologique suffisamment large et de sa capacité à encadrer de jeunes chercheurs.

Les règles de fond et de forme pour la présentation d'un manuscrit d'HDR ne sont pas explicitées formellement, cependant pour le fond il est attendu par l'école doctorale, les rapporteurs et les examinateurs que le manuscrit est bien aux standards requis, c'est-à-dire qu'il met en valeur tous les éléments cités précédemment.

Concernant le « fond », mon manuscrit présente donc les travaux de recherche que je mène au sein du laboratoire G-SCOP depuis 2010 lorsque j'ai effectué un changement de thématique de recherche. Dans cette partie « scientifique », je présente mon domaine de recherche, la sûreté de fonctionnement et le pronostic et management de la santé et sous la forme d'un fil conducteur j'explicite tous les axes de recherche (dénotés A), B) et C) dans le manuscrit) sur lesquels je travaille, mes contributions et l'encadrement doctoral que j'ai mené. Cette partie « scientifique » se poursuit avec la présentation de mes projets et ambitions de recherche (chapitre X).

Concernant la « forme » de la présentation du manuscrit de recherche, il n'y a pas de standard requis, elle est laissée au choix du candidat à la HDR et lui permet d'y exprimer « sa plume » et « sa personnalité ». J'ai ainsi choisi pour la forme de mon manuscrit de HDR :

— une présentation sous la forme d'une publication scientifique à vocation pédagogique pour de jeunes chercheurs ;

— appliquer à mon manuscrit le concept pédagogique que j'applique dans mes enseignements : rester dans la simplicité des explications tout en allant dans la profondeur des concepts ;

— de faire un document détaillé sur mon travail de recherche (le passé) afin d'expliciter les fondations sur lesquelles seront construits mes projets de recherche (le futur) ;

— de faire apparaitre les méthodologies et les algorithmes que j'ai développés, non pas pour rendre ardu la lecture de mon manuscrit, mais pour que le lecteur sache comment s'exprime la problématique de mettre en œuvre une méthodologie de diagnostic, de pronostic et d'aide à la décision pour la maintenance en concevant des algorithmes : problématique de modéliser les informations, définir les structures de données adéquates, trouver comment agréger ces données et les exploiter pour obtenir la finalité souhaitée.

Introduction

L'activité de recherche que je mène depuis mon doctorat consiste à comprendre le fonctionnement de systèmes complexes, à trouver une modélisation mathématique/informatique adéquate qui représente leur fonctionnement et à permettre de les simuler puis d'élaborer des méthodes, algorithmes et outils de raisonnement qui permettent d'extraire d'une part des propriétés qui caractérisent leur état de fonctionnement et de santé et d'autre part de fournir des indicateurs/métriques qui permettent d'évaluer leurs performances.

Cette démarche de recherche est présentée dans cette première partie de mon manuscrit par la présentation des travaux de recherche sur la thématique de la sûreté de fonctionnement des systèmes industriels complexes que je mène depuis une dizaine d'années au sein du laboratoire G-SCOP par des coopérations avec des chercheurs de G-SCOP et des encadrements de thèses ou de stages de niveaux Master 2 Recherche.

Un système industriel complexe est défini comme un système qui comporte un nombre important de composants ou de sous-systèmes ayant les caractéristiques suivantes :

— différents types de composants : électriques, mécaniques, chimiques, réseaux, etc. ayant chacun ses propres modes de dégradation ;

— de nombreuses interactions entre les composants ou sous-systèmes ;

— le système évolue dans un environnement dynamique et incertain et sera alors utilisé dans plusieurs conditions opérationnelles.

Mes travaux de recherches se sont orientés sur le développement de méthodes et d'outils pour contribuer à la sûreté de fonctionnement de tels systèmes par des approches complémentaires qui sont le diagnostic de défaillances, le pronostic de défaillances et l'aide à la décision dans la maintenance industrielle.

La thématique de recherche du diagnostic est partagée par plusieurs communautés : la communauté FDI (Fault Detection and Isolation), la communauté DX (Diagnosis eXpert system) et la communauté DES (Discrete Event System). Mes travaux relatifs au diagnostic se rattachent principalement à la communauté DES. Mes activités de recherche sur la maintenance et le pronostic m'inscrivent dans la communauté PHM (Prognostics and Health Management). Les systèmes complexes auxquels je m'intéresse sont des systèmes de production de biens (chaine d'assemblage de produits ou de remanufacturing, système de production automatisée, etc.), des systèmes cyber-physiques utilisés dans l'industrie (un cobot par exemple) ou dans les services (robot d'assistance à la personne) ou encore des

produits/équipements, comme des batteries lithium-ion, dont la fonctionnalité est critique pour les systèmes dans lesquels elles sont utilisées.

Mes activités de recherches durant ces dernières années ont été d'accompagner l'évolution de la sûreté de fonctionnement à l'industrie du futur et de considérer les verrous scientifiques suivants :

— prise en compte de systèmes industriels de complexité et d'hétérogénéité sans cesse croissant ;

— prise en compte des nouvelles évolutions technologiques telles que l'Internet of Things (IoT) et le Big data pour concevoir des méthodes de diagnostic, de pronostic, d'aide à la décision à la maintenance à des systèmes fonctionnant en réseau et générant un grand nombre de données ;

— prise en compte des grandes masses de données hétérogènes générées par les systèmes : extraction de données de qualités et leur exploitation par des outils issus de l'intelligence artificielle dans des contextes complexes, multi-sources, incertains, dégradés ;

— prise en compte de la transition écologique dans le développement de méthodes de diagnostic, de pronostic et d'aide à la décision à la maintenance : intégration de critères permettant d'obtenir une durée de vie plus longue des équipements, d'optimiser les stratégies de réparation et de maintenance pour réduire la surconsommation de produits et limitant ainsi les transports et les déchets.

Ce manuscrit a pour objectif de décrire les travaux que j'ai mené dans ce sens, leur état d'avancement, les principaux résultats obtenus et la présentation de mes projets de recherche.

Il est composé de neuf premiers chapitres répartis sur trois parties pour présenter une synthèse de mes principaux travaux de recherche et se poursuit par la présentation de mes perspectives de recherche dans un dixième et dernier chapitre. Les travaux de recherche présentés dans ce mémoire sont ainsi organisés autour de trois grands axes :

A) Méthodes de diagnostic basées sur des modèles.

B) Développement de nouveaux outils d'aide à la décision dans le diagnostic et la maintenance.

C) Méthodes de pronostic basées sur les données.

Les chapitres I à VI regroupés dans l'axe de recherche *Méthodes de diagnostic basées sur des modèles* présentent tous mes travaux relatifs au diagnostic de systèmes à événements discrets temporisés, de systèmes dynamiques hybrides et de systèmes cyber-physiques. Je me suis intéressé à développer de nouvelles méthodes de modélisation de ces systèmes complexes et à élaborer des algorithmes pour le diagnostic qui exploitent au mieux les outils de modélisation adaptés (automates temporisés, automates hybrides, réseaux de Petri temporisé).

Les chapitres VII et VIII regroupés dans l'axe de recherche *Développement de nouveaux outils d'aide à la décision dans le diagnostic et la maintenance* présentent l'usage

du formalisme des arbres de défaillances dynamiques pour la sûreté de fonctionnement et du formalisme des réseaux de files d'attente pour le dimensionnement d'un atelier de maintenance ainsi que l'usage de métaheuristiques pour la planification de la maintenance préventive systématique suivant les périodicités calculées. Sur ces problématiques, nous avons développé plusieurs outils d'aide à la décision.

Le chapitre IX contenu dans l'axe *Méthodes de pronostic basées sur les données* traite de la problématique d'exploiter des données de surveillance en vue du pronostic à base de méthodes d'apprentissage issues de l'intelligence artificielle. Ce chapitre propose une méthodologie de diagnostic et de pronostic de santé des batteries lithium-ion dans un contexte d'économie circulaire.

Le dernier chapitre X présente plusieurs projets de recherche que je souhaite animer et mes ambitions de recherche sur les prochaines années.

Axe A)

Méthodes de diagnostic basées sur les modèles

I Construction automatique d'un diagnostiqueur de SED temporisés 11
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 11
2 Synthèse des travaux développés 14
3 Conclusion et perspectives sur la construction de diagnostiqueurs 24

II Etude de la diagnosticabilité d'un SED 27
1 Présentation générale : contexte, état de l'art, positionnement 27
2 Synthèse des travaux développés 31
3 Conclusion et perspectives sur l'étude de la diagnosticabilité 38

III Génération de l'automate comportemental de SED temporisés 41
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 41
2 Synthèse des travaux développés 42
3 Conclusion et perspectives sur la construction de l'automate comportemental 47

IV Diagnostic de systèmes dynamiques hybrides 49
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 49
2 Synthèse des travaux développés 54
3 Conclusion et perspectives sur le diagnostic de systèmes dynamiques hybrides 67

V Diagnostic de systèmes cyber-physiques 71
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 71
2 Synthèse des travaux développés 76
3 Conclusion et perspectives sur la modélisation et le diagnostic de systèmes cyber-physiques 96

VI Diagnostic de dérives temporelles dans un GET 99
1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 99
2 Synthèse des travaux développés 100
3 Conclusion et perspectives sur l'identification de fautes temporelles 111

Collaborations et encadrements de travaux de recherche dans l'Axe A)

Les travaux de recherche relatés dans les chapitres suivants I à VI ont été réalisés par une collaboration avec Zineb Simeu-Abazi (laboratoire G-SCOP, Polytech Grenoble), Hassani Messaoud (laboratoire LARATSI, ENIM Monastir, Tunisie) et Hedi Dhouibi (laboratoire LARATSI, ISSAT Kairouan, Tunisie) au travers d'encadrement de thèses et de stages de niveau Master.

Les prochains chapitres présenteront une synthèse des travaux réalisés et citeront les publications issues de ces collaborations de recherche.

Publications issues de collaborations et d'encadrements de thèses et de stages de Master dans l'Axe A)

(RI : Revue Internationale, CI : Conférence Internationale, WI : Workshop International, CN : Conférence Nationale)

[RI-1] **E. Gascard** and Z. Simeu-Abazi, "Modular Modelling for the Diagnostic of Complex Discrete-Event Systems", *IEEE Transactions on Automation Science and Engineering*, vol. 10, no. 4, pp. 1101–1123, 2013.

[RI-2] **E. Gascard**, Z. Simeu-Abazi, and B. Suiphon, "A Polynomial Algorithm for Diagnosability Analysis of Discrete Event Systems", *International Journal of Adaptive, Resilient and Autonomic Systems (IJARAS)*, vol. 6, no. 2, pp. 1–22, 2015.

[RI-3] L. Belkacem, Z. Simeu-Abazi, H. Dhouibi, **E. Gascard**, and H. Messaoud, "Diagnostic and prognostic of hybrid dynamic systems: Modeling and RUL evaluation for two maintenance policies", *Reliability Engineering & System Safety*, vol. 164, pp. 98–109, 2017.

[CI-1] Z. Simeu-Abazi, **E. Gascard**, and F. Chalagiraud, "Diagnostic of Discrete Event Systems using Timed Automata in Matlab Simulink", in *Proceedings of the 20th European Safety and Reliability Conference, ESREL 2011*, CRC Press, 2011, pp. 402–409.

[CI-2] B. Suiphon, Z. Simeu-Abazi, and **E. Gascard**, "Implementation of a fault diagnosis method for timed discrete-event systems", in *Proceedings of the 5th International Conference on Industrial Engineering and Systems Management (IESM 2013)*, IEEE Press, 2013, pp. 870–877.

[CI-3] **E. Gascard**, Z. Simeu-Abazi, and B. Suiphon, "A Polynomial-Time Algorithm for Diagnosability Verification of Discrete Event Systems", in *Proceedings of the 2nd World Conference on Complex Systems (WCCS 2014)*, IEEE Press, 2014, pp. 286–291.

[CI-4] L. Belkacem, Z. Simeu-Abazi, L. Mhamdi, H. Messaoud, and **E. Gascard**, "Diagnosis of Hybrid Dynamical Systems through Hybrid Automata", in *Proceedings of the 8th IFAC Conference on Manufacturing Modelling, Management, and Control (MIM 2016)*, ser. IFAC-PaperOnLine, ELSEVIER, vol. 49, 2016, pp. 990–995.

[CI-5] B. Maaref, Z. Simeu-Abazi, H. Dhouibi, H. Messaoud, and **E. Gascard**, "Mixed approach for fault diagnosis and fault location of hybrid systems", in *Proceedings of the 8th IFAC Conference on Manufacturing Modelling, Management, and Control (MIM 2016)*, ser. IFAC-PaperOnLine, ELSEVIER, vol. 49, 2016, pp. 1002–1007.

[CI-6] M. A. Haj Kacem, Z. Simeu-Abazi, **E. Gascard**, G. Lemasson, and J. Maisonnasse, "Application of a modeling approach on a cyber-physical system RobAIR", in *Proceedings of the 20th World Congress of the International Federation of Automatic Control (IFAC WC 2017)*, ser. IFAC-PaperOnLine, ELSEVIER, vol. 50, 2017, pp. 14 230–14 235.

[CI-7] Z. Simeu-Abazi, **E. Gascard**, and M. A. Haj Kacem, "Implementation of start-up tests for system health assessment: Application to a telepresence robot RobAIR", in *Proceedings of the 3rd International Conference on Control, Automation and Diagnosis (ICCAD'19)*, IEEE Press, 2019, pp. 33–38.

[CI-8] Z. Simeu-Abazi and **E. Gascard**, "Fault diagnosis method for timed discrete-event systems: Application to autonomous electric vehicle", in *Proceedings of the 3rd International Conference on Control, Automation and Diagnosis (ICCAD'19)*, IEEE Press, 2019, pp. 374–379.

[CI-9] **E. Gascard**, Z. Simeu-Abazi, and N. Moussa, "Detection and Localization of Time Shift Failures in Timed Event Graphs: Application to a Remanufacturing Line", in *Proceedings of the 31st European Safety and Reliability Conference (ESREL 2021)*, Research Publishing, Singapore, 2021, pp. 1–8.

[CI-10] I. Hnayen, H. Dhouibi, C. Ben Njima, **E. Gascard**, and Z. Simeu-Abazi, "Fault diagnosis and modeling of hybrid systems through hybrid automata", in *Proceedings of the 6th International Conference on Control, Automation and Diagnosis (ICCAD'22)*, IEEE, 2022.

[WI-1] **E. Gascard** and Z. Simeu-Abazi, "Automatic Construction of Diagnoser for Complex Discrete Event Systems", in *Proceedings of the 3rd International Workshop on Dependable Control of Discrete Systems (DCDS 2011)*, IEEE Press, 2011, pp. 90–95.

[WI-2] M. A. Haj Kacem, Z. Simeu-Abazi, and **E. Gascard**, "Failure identification and propagation analysis of cyber-physical systems", in *3rd Workshop on Advanced Maintenance Engineering, Service and Technology (AMEST 2016)*, 2016.

[CN-1] **E. Gascard**, Z. Simeu-Abazi, and G. Mayol, "Elaboration du comportement dynamique d'un système pour le diagnostic des défaillances", in *10ème Congrès International de Génie Industriel (CIGI 2013)*, 2013.

[CN-2] B. Suiphon, Z. Simeu-Abazi, and **E. Gascard**, "Premiers pas vers le diagnostic de défaillances par exploitation d'un modèle SysML", in *19ème Congrès de Maîtrise des Risques et Sûreté de Fonctionnement (LambdaMu 19)*, 2014.

[CN-3] M. A. Haj Kacem, Z. Simeu-Abazi, and **E. Gascard**, "Diagnostic des défaillances des systèmes cyber-physiques par la modélisation des connaissances", in *11ème Congrès International de Génie Industriel (CIGI 2015)*, 2015.

Chapitre I

Construction automatique d'un diagnostiqueur de systèmes à événements discrets

1 Présentation générale : contexte, état de l'art, positionnement, originalité

1.1 Contexte

Le contexte de ce travail de recherche est la construction automatique d'un module de surveillance, appelé *diagnostiqueur*, pour détecter, localiser et identifier des fautes permanentes dans un système manufacturier. Le concept de diagnostiqueur a été introduit par Sampath et al. dans [1], [2].

Dans le cadre de cette recherche nous nous intéressons uniquement au diagnostic et non au pronostic de défaillances [3], nous supposons que nous avons comme données observables uniquement les effets d'une défaillance. Nous considérons que nous n'avons pas à notre disposition des données montrant la dégradation d'un équipement jusqu'à sa panne.

1.2 Etat de l'art, positionnement et originalité

Une classification usuelle des méthodes de diagnostic se fait selon la connaissance disponible du système. Cette classification répartit les méthodes de diagnostic en trois familles :

— Les méthodes basées sur les connaissances.

— Les méthodes basées sur les données.

— Les méthodes basées sur les modèles.

Les méthodes basées sur les connaissances ne disposent pas d'un modèle explicite d'un système ni de grands jeux de données montrant le comportement normal et anormal du système. Dans ces méthodes, le diagnostic est élaboré à partir de connaissances sur les relations causales entre les défauts, les symptômes et les défaillances. Dans les méthodes basées sur les connaissances, nous retrouvons les approches à bases de systèmes experts

[4], d'AMDEC (analyse des modes de défaillance et de leurs effets) [5], ou d'arbres de défaillances [6].

Dans les méthodes basées sur les données, les données proviennent de capteurs supposés fiables, ils correspondent à des signaux, des vidéos, des enregistrements sonores, etc. Les valeurs de ces données sont donc supposées correctes et suffisamment nombreuses pour représenter convenablement les différents états (normaux, dégradés et défaillants) du système. Ces méthodes ont pour finalité d'associer un ensemble de données historiques à un état de fonctionnement du système par des techniques d'apprentissage [7]. Ces méthodes d'apprentissage peuvent être classées en trois catégories selon le type d'apprentissage qu'elles emploient : apprentissage supervisé, apprentissage non supervisé et apprentissage par renforcement. Dans les techniques classiques d'apprentissage supervisé, on trouve l'algorithme des k plus proches voisins, la classification bayésienne et les réseaux de neurones. Les méthodes d'apprentissage non supervisé cherchent à découvrir les structures sous-jacentes à des données non étiquetées, parmi ces méthodes, on retrouve l'algorithme des k-moyennes et l'analyse en composantes principales. Le lecteur trouvera dans [8], [9] une présentation des différentes méthodes d'apprentissage supervisé et non supervisé. Avec les méthodes d'apprentissage par renforcement [10], les algorithmes d'apprentissage apprennent un comportement étant donnée une observation du système et produisent une valeur de retour qui guide l'algorithme d'apprentissage. La méthode la plus courante d'apprentissage par renforcement est le Q-learning.

Les méthodes basées sur les modèles reposent sur la connaissance d'un modèle du fonctionnement du système en mode nominal, et éventuellement dégradé et défaillant. Les modèles utilisés peuvent être quantitatifs [11], [12] ou qualitatifs [13] :

— Le diagnostic à base de modèles quantitatifs est basé sur un modèle physique du système : des équations algébro-différentielles permettent de représenter le comportement continu du système. La méthode de diagnostic compare des mesures effectuées sur des variables du système avec le comportement prédit à l'aide du modèle physique pour générer des indicateurs de défaut, appelés résidus. Ces comparaisons permettent la détection d'incohérence. Si le modèle physique de dysfonctionnement est connu, ces comparaisons peuvent permettre la localisation puis l'identification des fautes. Ces méthodes de diagnostic appartiennent à la communauté FDI (Fault Detection and Isolation) et utilisent des approches par espace de parité, d'estimation de paramètres ou d'estimations d'état.

— Le diagnostic à base de modèles qualitatifs est basé sur la structure du système et les interactions entre ses composants. On retrouve ainsi les approches issues de la communauté des Systèmes à Événements Discrets (SED) et celles issues de la communauté DX (Diagnosis eXpert system). Ces modèles qualitatifs permettent d'abstraire le comportement du système, ils décrivent d'une manière qualitative l'espace d'états du système. La modélisation du système n'est plus axée sur la physique du système, mais sur ses modes de fonctionnement et le niveau d'abstraction considéré du système : systèmes à événements discrets, systèmes continus, systèmes hybrides. Selon

le niveau d'abstraction du système considéré, différentes approches ont été définies. Pour les systèmes continus, les approches développées utilisent des graphes causaux ou des graphes causaux temporels. Pour les systèmes à événements discrets, les approches utilisent des automates d'états finis, des automates temporisés, des réseaux de Petri, etc. Pour les systèmes hybrides, les approches utilisent des automates hybrides à temps discrets, des graphes de liaisons (*bond graphs*), des réseaux de Petri hybrides, etc.

Nos travaux de recherche s'inscrivent dans la thématique du diagnostic à base de modèles analytiques en utilisant des systèmes à événements discrets temporisés modélisés par des automates d'états finis temporisés communicants.

L'utilisation du diagnostiqueur peut être faite en ligne ou hors ligne. Le diagnostic en ligne consiste à surveiller l'état du système pendant son fonctionnement sans connaissance a priori de la présence d'un défaut ou non et à calculer le diagnostic (localisation et identification) du système dès que la détection d'une défaillance est réalisée. Le diagnostic en ligne peut être passif ou actif. Un diagnostic passif en ligne analyse le flot des entrées et des sorties du système sans influencer le choix des entrées pour guider le diagnostic alors qu'un diagnostic actif en ligne peut influencer le choix des entrées pour guider le processus de diagnostic. Le diagnostic hors ligne est réalisé après l'apparition d'une panne, il consiste à utiliser l'historique des entrées-sorties du système pour localiser et identifier le composant défectueux. Lors du diagnostic hors ligne, des séquences de test peuvent être calculées et utilisées pour guider le diagnostic.

Dans nos travaux de recherche, nous utilisons une approche de diagnostic passif en ligne.

La structure du diagnostiqueur peut être centralisée, décentralisée ou distribuée [14]. Avec une structure centralisée, le diagnostiqueur est composé d'un seul module de diagnostic, sa méthode de construction nécessite d'avoir la description du modèle global du système. La limite de cette approche peut être l'explosion combinatoire du nombre d'états du diagnostiqueur. Les approches décentralisées et distribuées nécessitent la construction d'un diagnostiqueur local pour chaque composant du système. Chaque diagnostiqueur local a une observation partielle du système à monitorer. Dans l'approche distribuée, les diagnostiqueurs locaux communiquent ensemble pour résoudre les problèmes de conflits de localisation et d'identification. Dans l'approche décentralisée, chaque diagnostiqueur élabore sa propre décision de localisation et d'identification sans communiquer avec les autres diagnostiqueurs locaux. Les problèmes de conflits de localisation et d'identification sont résolus par un module de diagnostic supplémentaire, le coordinateur.

La méthode de construction du diagnostiqueur que nous avons utilisé dans notre travail de recherche s'apparente à une approche centralisée. Nous construisons un unique module de diagnostic, cependant notre méthodologie de construction du diagnostiqueur ne requiert pas d'avoir le modèle global du système : notre approche utilise une description modulaire du système à base d'automates temporisés communicants. Notre méthode de construction du diagnostiqueur utilise de manière maitrisée la composition synchrone des automates

afin de ne pas construire l'ensemble des états du système global, nous limitons le problème de l'explosion combinatoire des états en ne visitant pas les états inatteignables.

2 Synthèse des travaux développés

2.1 Présentation synthétique d'une nouvelle méthode de diagnostic pour les systèmes à événements discrets temporisés

Le système à diagnostiquer est décrit par un réseau d'automates temporisés communicants composés d'un contrôleur $\mathcal{C}$, d'un procédé $\mathcal{P}$, de n actionneurs $\mathcal{A}_1, \ldots, \mathcal{A}_n$ et de m capteurs $\mathcal{S}_1, \ldots, \mathcal{S}_m$.

Nous considérons seulement les fautes permanentes sur les actionneurs et capteurs. Nous sommes dans une approche de diagnostic de type DES et non FDI, aussi nous ne considérons pas des dégradations progressives dans la dynamique du procédé. Les observations sont les commandes issues du contrôleur aux actionneurs et les données fournies par les capteurs au contrôleur. Les données non observables sont l'état des actionneurs qui ont une influence sur le comportement du procédé et la dynamique du procédé qui a une influence sur l'état des capteurs. La figure I.1 présente le principe de diagnostic passif en ligne de notre approche.

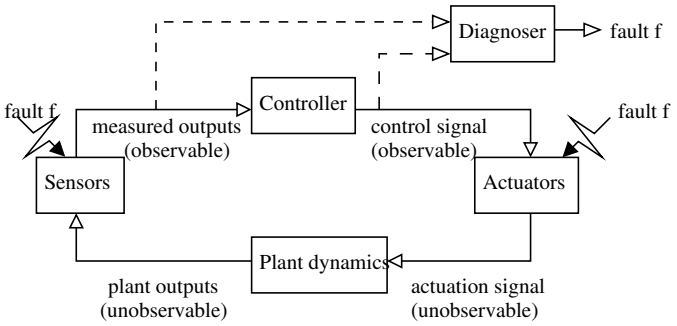


FIGURE I.1 – Diagnostic d'un système manufacturier

Une première différence dans notre approche par rapport à celle de Sampath et al. est sur la modélisation utilisée. Nous utilisons un réseau d'automates temporisés communicants et nous ne représentons pas une faute comme un événement inobservable, mais comme un état particulier dans l'automate temporisé. L'occurrence d'une faute est alors traduite par l'arrivée sur un état étiquetté comme fautif. Nous proposons ainsi une approche de modélisation compositionnelle d'un système manufacturier (contrôleur, actionneurs, système et capteurs) avec des automates temporisés communicants qui évite de construire un modèle global du système et réduit ainsi sa difficulté de modélisation. La figure I.2 présente le principe de construction du diagnostiqueur qui est décomposé en deux étapes.

La première étape est la *construction automatique* par notre algorithme I.1 d'un automate temporisé non déterministe étiqueté uniquement par des données observables que nous appellerons l'*esquisse du diagnostiqueur*. Sa construction se déroule comme suit :

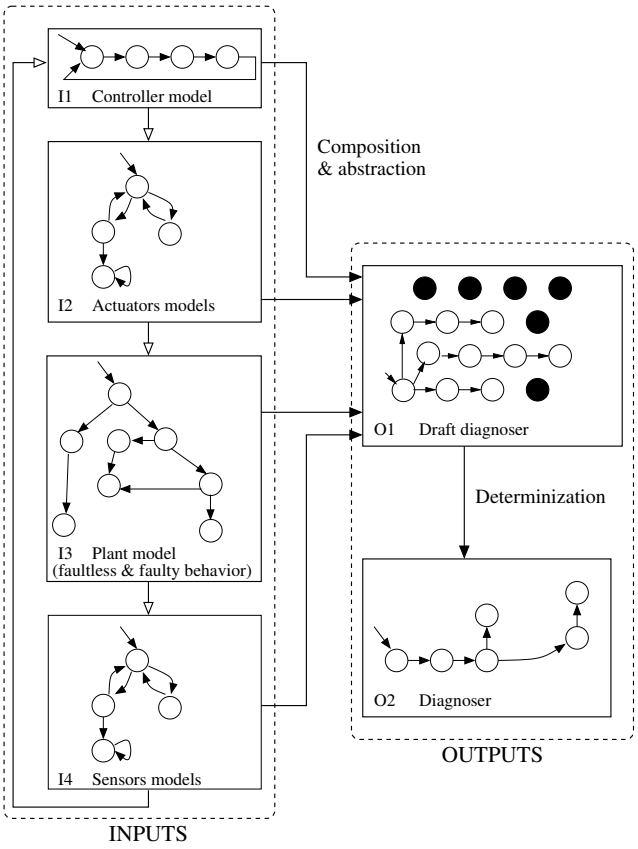


FIGURE I.2 – Principe de construction du diagnostiqueur

— Construction d'une partie restreinte de la composition parallèle des automates $\mathcal{C}\|\mathcal{A}_1\|\ldots\|\mathcal{A}_n\|\mathcal{S}_1\|\ldots\|\mathcal{S}_m\|\mathcal{P}$. Nous limitons l'exploration de l'ensemble des états atteignables à ceux guidés par le contrôleur : lorsqu'il y a le choix entre de multiples transitions, si certaines d'entre elles proviennent du contrôleur, nous choisissons uniquement celles-ci. En effet, l'état du procédé est déterminé par les actions du contrôleur, aussi il est inutile de prendre en compte des états qui ne peuvent pas être atteints par le fonctionnement du contrôleur.

— Simplification de l'étiquetage des transitions du modèle global restreint : nous ne faisons plus apparaitre les gardes et les mises à jour de variables en rapport avec des données non observables. Les transitions de notre esquisse du diagnostiqueur sont alors composées d'une garde manipulant des données observables temporisées (valeurs des capteurs ou commande sur les actionneurs par le contrôleur associées à une condition sur la valeur d'une horloge).

La seconde étape est la *construction automatique* du diagnostiqueur par notre algorithme I.2 à partir de son esquisse : nous le déterminisons et de nouvelles transitions sont générées pour la localisation et l'identification des défaillances.

Les algorithmes I.1 et I.2 présentent les étapes de construction de l'esquisse du diagnostiqueur (*draft diagnoser*) et du diagnostiqueur. Le lecteur trouvera dans [15] les explications détaillées sur les structures de données utilisées et le fonctionnement de ces algorithmes.

Une seconde différence avec l'approche de Sampath et al. est notre capacité à identifier un plus grand nombre de défaillances grâce aux informations temporelles que nous prenons en compte. Nous pouvons résoudre des ambiguïtés de localisation et d'identification qui peuvent apparaitre dans une approche non temporisée en utilisant les temps d'apparition de données observables.

Une dernière étape est la *vérification formelle du diagnostiqueur* : nous devons nous assurer de sa correction. Pour cela nous devons démontrer que le diagnostiqueur est correct dans le sens qu'il identifie une défaillance si et seulement si une faute correspondante est apparue. Nous proposons d'utiliser des techniques de vérification de modèles (*model checking*) pour obtenir la vérification formelle des diagnostiqueurs. Nous vérifions qu'il n'existe pas d'absence de diagnostic si une défaillance est survenue et qu'il n'y a pas de faux diagnostic : toute identification d'une défaillance est causée par sa faute correspondante.

2.2 Illustration de notre méthodologie de construction d'un diagnostiqueur

2.2.1 Présentation des automates temporisés communicants

Un automate temporisé est un automate à états finis étendu avec des horloges (variables à valeurs réelles, positives ou nulles) qui permettent d'exprimer des contraintes temporelles.

Un automate communicant est un automate à états finis qui peut envoyer et recevoir des messages sur un ensemble fini de canaux.

Nous représentons les automates temporisés communicants avec la syntaxe de l'outil UPPAAL [16]. UPPAAL est un outil qui permet de définir des réseaux d'automates temporisés communicants, de simuler leur comportement, et de vérifier des propriétés logico-temporelles par des techniques de model checking. La communication dans UPPAAL se réalise par une synchronisation binaire utilisant des canaux avec une syntaxe de type émission/réception. Ainsi, sur le canal c, un émetteur envoie le signal $c!$ et un récepteur se synchronise avec lui par le signal complémentaire $c?$. Uppaal permet, outre l'utilisation d'horloges, de manipuler des variables entières afin d'étendre l'espace d'état de l'automate au-delà de ses localités.

Avec la syntaxe UPPAAL, un automate temporisé communicant se définit par $\mathcal{A} = (L, \ell_0, Sync, Clk, V, E, Init, I)$ où :

- L est un ensemble fini d'états appelés localités, $\ell_0 \in L$ représente la localité initiale.
- $Sync$ contient l'ensemble des actions de synchronisation (émission de type $c!$ et réception de type $c?$) ainsi que l'absence d'étiquette de synchronisation qui indique une action interne (τ-action) de l'automate.
- Clk est l'ensemble des horloges.
- V est l'ensemble des variables entières.

Algorithme I.1 Construction de l'esquisse du diagnostiqueur

procedure BUILDDRAFTDIAGNOSER
Input: $\mathcal{C}, \mathcal{P}, \mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{S}_1, \ldots, \mathcal{S}_m$
Output: draft diagnoser $\mathcal{G} = (L^g, \ell_0^g, Sync, Clk, V, E^g, Init, I^g)$

Initialization : $\ell_0^g = (\ell_0^c, \ell_0^{a1}, \ldots, \ell_0^{an}, \ell_0^{s1}, \ldots, \ell_0^{sm}, \ell_0^p)$; $L^g = \{\ell_0^g\}$; $E^g = \emptyset$; $Waiting = \{\ell_0^g\}$;
while $Waiting \neq \emptyset$ **do**
  $\ell^g = Waiting.pop()$;
  $lst\_trans = Available\_Transitions(\ell^g)$;
  $(lst\_trans_{Ctrl}, lst\_trans_{Plant}) = Partition\_Controller(lst\_trans)$;
  {we partition the set of available transitions according to the role of the controller or the plant in the transition}
  if $lst\_trans_{Ctrl} \neq \emptyset$ **then** {we use a transition directed by the controller}
    for each transition $e \in lst\_trans_{Ctrl}$ **do**
      if $e.destination \notin L^g$ **then**
        $Waiting.push(e.destination)$;
        $L^g = L^g \cup \{e.destination\}$;
      end if
      if $e.sync = \emptyset$ **then**
        {case $\ell_u^c \xrightarrow{g} \ell_v^c$ where g is a Boolean expression on the status of the sensors}
        $E^g = E^g \cup \{\ell^g \xrightarrow{g} e.destination\}$;
      else
        {case $\ell_u^c \xrightarrow[sentactuator_i:=1]{actuator_i!} \ell_v^c$ synchronized with $\ell_u^{ai} \xrightarrow[v:=value]{actuator_i?} \ell_v^{ai}$}
        $E^g = E^g \cup \{\ell^g \xrightarrow[sentactuator_i:=0]{sentactuator_i==1} e.destination\}$;
      end if
    end for
  else{we use a transition directed by the plant}
    for each transition $e \in lst\_trans_{Plant}$ **do**
      if $e.destination \notin Q$ **then**
        $Waiting.push(e.destination)$;
        $L^g = L^g \cup \{e.destination\}$;
      end if
      if $e.sync = \emptyset$ **then**
        {case $\ell_u^p \xrightarrow[clk_i:=0]{g_1} \ell_v^p$ where g_1 is a Boolean expression on the status of the actuators}
        $E^g = E^g \cup \{\ell^g \xrightarrow{clk_i==0} e.destination\}$;
      else
        {case $\ell_v^p \xrightarrow{g_2, sensor_j!} \ell_w^p$ synchronized with $\ell_u^{sj} \xrightarrow[v:=value]{sensor_j?} \ell_v^{sj}$ where g_2 is a Boolean expression on the clock clk_i}
        $E^g = E^g \cup \{\ell^g \xrightarrow{g_2 \&\& v==value} e.destination\}$;
      end if
    end for
  end if
end while

Algorithme I.2 Construction du diagnostiqueur

procedure BUILDDIAGNOSER

Input: draft diagnoser $\mathcal{G} = (L^g, \ell_0^g, Sync, Clk, V, E^g, Init, I^g)$

Output: diagnoser $\mathcal{D} = (L^d, \ell_0^d, Sync, Clk, V, E^d, Init, I^d)$

1: Initialization : $\ell_0^d = \{\ell_0^g\}; L^d = \{\ell_0^d\}; E^d = \emptyset; Waiting = \{\ell_0^d\}$;
2: **while** $Waiting \neq \emptyset$ **do**
3: $\ell^d = Waiting.pop()$;
4: $(states\_with\_transitions, states\_without\_transitions) = Partition\_SourceOfTransitions(\ell^d)$;
5: {we partition the set of states ℓ^d according to they are or not source of transitions in $\mathcal{G}$}
6: **for each** $\ell^g \in states\_with\_transitions$ **do**
7: $\{\ell^d = \{\ldots, \ell^g, \ldots\}$ and ℓ^g has some transitions in $\mathcal{G}\}$
8: $lst\_trans = Transitions(\ell^g)$;
9: **for each** $e \in lst\_trans$ **do**
10: **if** $\exists \ell'^d \in L^d$ such that $e.destination \in \ell'^d$ **then**
11: {$e.destination$ is present in of another state of the diagnoser, so it is unnecessary to create a new state}
12: $E^d = E^d \cup \{\ell^d \xrightarrow[e.assignment]{e.guard} \ell'^d\}$;
13: **else if** $\exists\{\ell^d \xrightarrow[e.assignment]{e.guard} \ell'^d\} \in E^d$ **then**
14: {there exists already a transition in the diagnoser starting from ℓ^d with the same guard/assignment of transition e, so we complete the subset of states ℓ'^d with $e.destination$}
15: $\ell'^d = \ell'^d \cup \{e.destination\}$;
16: **else**
17: Create new state $\ell'^d = \{e.destination\}$;
18: $L^d = L^d \cup \{\ell'^d\}$;
19: $E^d = E^d \cup \{\ell^d \xrightarrow[e.assignment]{e.guard} \ell'^d\}$;
20: $Waiting.push(\ell'^d)$;
21: **end if**
22: **end for**
23: **end for**
24: **for each** $\ell^g \in states\_without\_arcs$ **do**
25: $\{\ell^d = \{\ldots, \ell^g, \ldots\}$ and ℓ^g has no transition in $\mathcal{G}\}$
26: **if** $\exists\{\ell^d \xrightarrow{clk_i==0} \ell'^d\} \in E^d$ **then**
27: $\ell'^d = \ell'^d \cup \{\ell^g\}$
28: **else if** $\exists\{\ell^d \xrightarrow{clk_i==constant \&\& v==value} \ell'^d\} \in E^d$ such that the Boolean expression $v == value$ is satisfied in ℓ^g **then**
29: $\ell'^d = \ell'^d \cup \{\ell^g\}$
30: **else if** $\exists\{\ell^d \xrightarrow{TO\_BE\_DEFINED} \ell'^d\} \in E^d$ **then**
31: $\ell'^d = \ell'^d \cup \{\ell^g\}$
32: **else**
33: Create new state $\ell'^d = \{\ell^g\}$;
34: $L^d = L^d \cup \{\ell'^d\}$;
35: $E^d = E^d \cup \{\ell^d \xrightarrow{TO\_BE\_DEFINED} \ell'^d\}$;
36: **end if**
37: **end for**
38: **end while**

— E est l'ensemble des transitions entre les localités. Une transition est étiquetée avec une action de synchronisation (de type émission, réception ou interne) conditionnée par une garde et une remise à zéro de certaines horloges et une mise à jour de certaines variables internes.

— $Init$ est l'ensemble des initialisations des variables.

— I associe à chaque localité un invariant : l'automate temporisé peut rester dans une localité tant que son invariant est satisfait, sinon il doit prendre une transition dont la garde devra être satisfaite pour changer de localité.

2.2.2 Exemple de modélisation d'un procédé batch dans UPPAAL

Nous présentons un exemple très simple de procédé batch qui servira à illustrer notre méthode de construction de diagnostiqueur dans le prochain paragraphe, figure I.3. Ce procédé de production représente le mélange de deux ingrédients dans une cuve pour obtenir un produit final. La cuve est équipée par deux senseurs/capteurs de niveaux S_1 et S_2 pour mesurer les niveaux $L1$ et $L2$ représentés par des variables et de deux actionneurs/vannes de remplissage A_1 et A_2, dont l'état de fonctionnement est représenté par les variables $V1$ et $V2$. Le remplissage de la cuve doit suivre la séquence de contrôle suivante : la vanne A_1 est ouverte, le premier ingrédient commence à remplir la cuve. Lorsque le niveau $L1$ est atteint, détecté par le capteur S_1, la vanne A_1 est refermée et la vanne A_2 est ouverte. Lorsque le niveau $L2$ est atteint, détecté par le capteur S_2, la vanne A_2 est refermée.

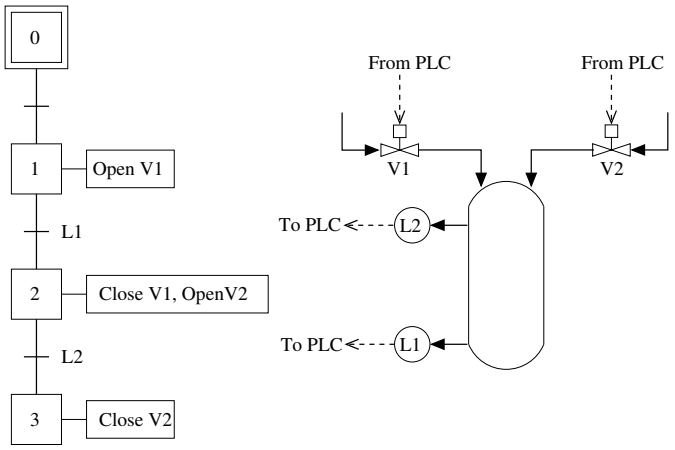


FIGURE I.3 – Exemple de procédé batch

Nous modélisons chaque équipement (capteurs et vannes), le contrôleur et le comportement du procédé par un automate temporisé. Nous présentons leur modélisation au format UPPAAL. Les états initiaux sont représentés par un double cercle. Les transitions sont étiquetées par un triplet : garde, action de synchronisation et affectations. Si l'action de synchronisation est interne, elle n'est pas représentée.

2.2.2.1 Automate temporisé du contrôleur

Les commandes d'ouverture et de fermeture des vannes sont modélisées par des actions de synchronisations $OpenV1!$, $CloseV1!$, $OpenV2!$ et $CloseV2!$. Les états des niveaux $L1$ et $L2$ sont exprimés par des variables entières initialisées à 0 et mises à jour à 1 lorsque le niveau est atteint. La figure I.4 décrit l'automate temporisé du contrôleur. De manière générique, les transitions d'un contrôleur sont de deux formats : $\ell_u^c \xrightarrow{g} \ell_v^c$ ou $\ell_u^c \xrightarrow[sentactuator_i:=1]{actuator_i!} \ell_v^c$ avec g est une expression booléenne sur le statut des capteurs et $actuator_i$ est un canal de synchronisation entre le contrôleur et l'actionneur A_i, $sentactuator_i$ est une variable entière mémorisant l'action de synchronisation sur le canal $actuator_i$.

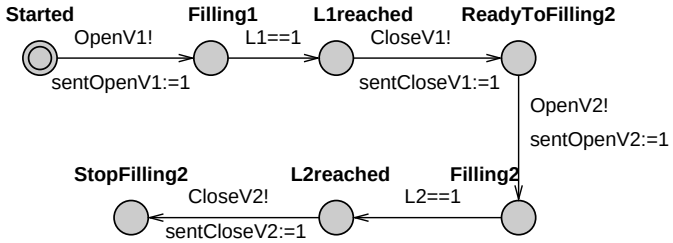


FIGURE I.4 – Automate du contrôleur dans le procédé batch

2.2.2.2 Automates temporisés des vannes

Nous modélisons l'état réel des vannes A_1 et A_2 par des variables entières $V1$ et $V2$ initialisées à 0. lorsqu'une vanne est ouverte, sa variable associée est mise à jour à 1. Elle est mise à jour à 0 lorsqu'elle est refermée. Pour la modélisation de la vanne A_1, nous considérons les défauts suivants :

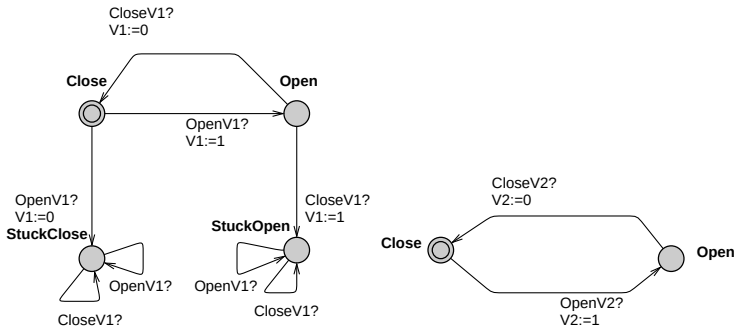
— *Stuck Close* : la vanne reste fermée lors d'une demande d'ouverture.

— *Stuck Open* : la vanne reste ouverte lors d'une demande de fermeture.

La figure I.5(a) décrit l'automate temporisé de la vanne A_1 en prenant en compte les deux défauts ci-dessus. La vanne A_1 dans son mode nominal peut être dans deux états : *Close* ou *Open*. Son état initial est *Close*. L'état de la vanne peut être changé par l'une des actions de synchronisation $CloseV1!$ ou $OpenV1!$. La description du comportement défaillant de A_1 correspond à la mise à jour de la variable d'état $V1$ avec une valeur incorrecte et menant à l'un des états fautifs *StuckClose* ou *StuckOpen*. La figure I.5(b) décrit l'automate temporisé de la vanne A_2 sans prise en compte de défauts pour cette vanne. De manière générique, les transitions d'un actionneur A_i sont de la forme $\ell_u^{ai} \xrightarrow[v_i:=value]{actuator_i?} \ell_v^{ai}$ où $actuator_i$ est un canal de synchronisation entre le contrôleur et l'actionneur A_i et v_i est la variable associée à l'état de cet actionneur.

2.2.2.3 Automates temporisés des capteurs

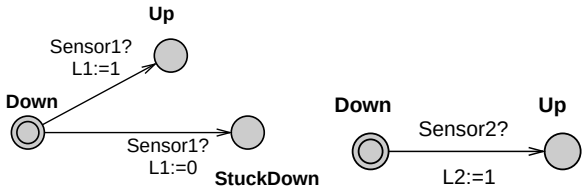
La modélisation des capteurs est réalisée d'une manière similaire à celle des actionneurs. L'état réel des capteurs S_1 et S_2 est modélisé par les variables entières $L1$ et $L2$ initialisées



(a) Valve $V1$ (b) Valve $V2$

FIGURE I.5 – Automates des vannes dans le procédé batch

à 0 (niveau non atteint). Lorsque le niveau $L1$ (respectivement $L2$) est atteint, sa variable d'état $L1$ (respectivement $L2$) est mise à 1. Pour la modélisation du capteur S_1, nous considérons le défaut *Stuck Down* : le capteur reste toujours à l'état 0, il ne détecte pas que le niveau est atteint. La figure I.6(a) décrit l'automate temporisé du capteur S_1 en prenant en compte le défaut ci-dessus. Le capteur S_1 dans son mode nominal peut être dans deux états : *Down* et *Up*. L'état initial est *Down*. L'état peut être changé par une action de synchronisation $Sensor1?$. L'arrivée dans l'état fautif de S_1 correspond à mettre à jour la variable $L1$ avec une valeur incorrecte. La figure I.6(b) décrit l'automate temporisé du capteur S_2 sans faute. De manière générique, les transitions d'un capteur S_j sont de la forme $\ell_u^{sj} \xrightarrow[v_j:=value]{sensor_j?} \ell_v^{sj}$ où $sensor_j$ est un canal de synchronisation entre le comportement du procédé et le capteur S_j et v_j est la variable associée à l'état de ce capteur.



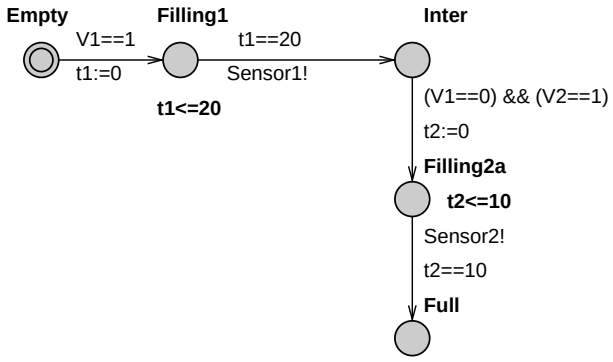
(a) Sensor $L1$ (b) Sensor $L2$

FIGURE I.6 – Automates des capteurs de niveaux dans le procédé batch

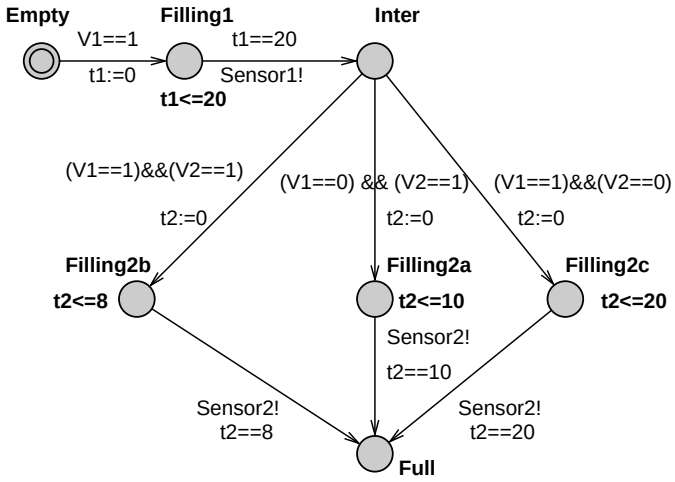
2.2.2.4 Automate temporisé du comportement du procédé

Les interactions du comportement physique du procédé dans la cuve avec les capteurs est représenté par un automate temporisé. Tout d'abord, nous définissons son comportement nominal (figure I.7(a)), puis nous le complétons avec ses comportements en présence de fautes (figure I.7(b)). Pour notre méthodologie de construction du diagnostiqueur, il est nécessaire que la modélisation du comportement du procédé couvre tous ses modes de fonctionnement : nominal et en présence de fautes. Nous utilisons deux horloges $t1$ et $t2$ pour observer le remplissage de la cuve jusqu'aux niveaux $L1$ et $L2$. L'évolution de la

dynamique du procédé est représentée par des transitions utilisant l'état des actionneurs, les horloges et les canaux de synchronisation des capteurs. Par exemple, l'état initial *Empty* mène à l'état de remplissage *Filling1* lorsque la vanne A_1 est ouverte ($V1 == 1$) puis passe de l'état *Filling1* à l'état *Inter* après 20 unités de temps de remplissage ($t1 == 20$). A cet instant $t1 = 20$, le niveau $L1$ doit normalement être atteint, et le capteur associé doit se déclencher, ceci est représenté par l'action de synchronisation $Sensor1!$.



(a) Comportement nominal du procédé batch



(b) Comportement avec et sans défauts du procédé batch

FIGURE I.7 – Automates du comportement du procédé batch

2.2.3 Application de notre méthodologie de construction du diagnostiqueur

2.2.3.1 Construction de l'esquisse du diagnostiqueur

A partir de la description des automates temporisés communicants du contrôleur, des actionneurs, des capteurs et du comportement du procédé, nous calculons tout d'abord par un algorithme d'atteignabilité un automate temporisé centralisé qui permet le suivi de l'évolution du système à diagnostiquer en utilisant uniquement des données observables. Nous avons défini une version adaptée de l'algorithme de composition d'automates pour

ne calculer qu'une partie restreinte de la composition de $\mathcal{C}\|\mathcal{A}_1\|\ldots\|\mathcal{A}_n\|\mathcal{S}_1\|\ldots\|\mathcal{S}_m\|\mathcal{P}$: lorsque de multiples transitions sont possibles, si certaines d'entre elles impliquent des transitions du contrôleur, nous gardons uniquement ces transitions. Ainsi cet automate temporisé décrit tous les états atteignables du système guidé par le fonctionnement du contrôleur. De plus, nous réalisons une abstraction sur les événements non observables : les transitions de l'esquisse du diagnostiqueur ne sont étiquetées que par des données observables. Son algorithme de construction est décrit dans [15], [17].

La figure I.8 décrit l'esquisse du diagnostiqueur $\mathcal{G}$ sur notre exemple de procédé batch. Les états de $\mathcal{G}$ sont définis comme des tuples d'états des composants du système. Ainsi, son état initial ℓ_0 est un tuple composé des états initiaux des actionneurs (vannes) A_1 et A_2, du contrôleur C, du comportement du procédé P et des capteurs de niveaux S_1 et S_2. Cet automate temporisé ne peut pas être utilisé directement comme un diagnostiqueur à cause de son non-déterminisme (plusieurs transitions étiquetées par la même garde partent d'un même état source vers différents états destination). Aussi, il est nécessaire de le déterminiser. C'est la deuxième étape de notre méthodologie de construction du diagnostiqueur présenté au paragraphe suivant.

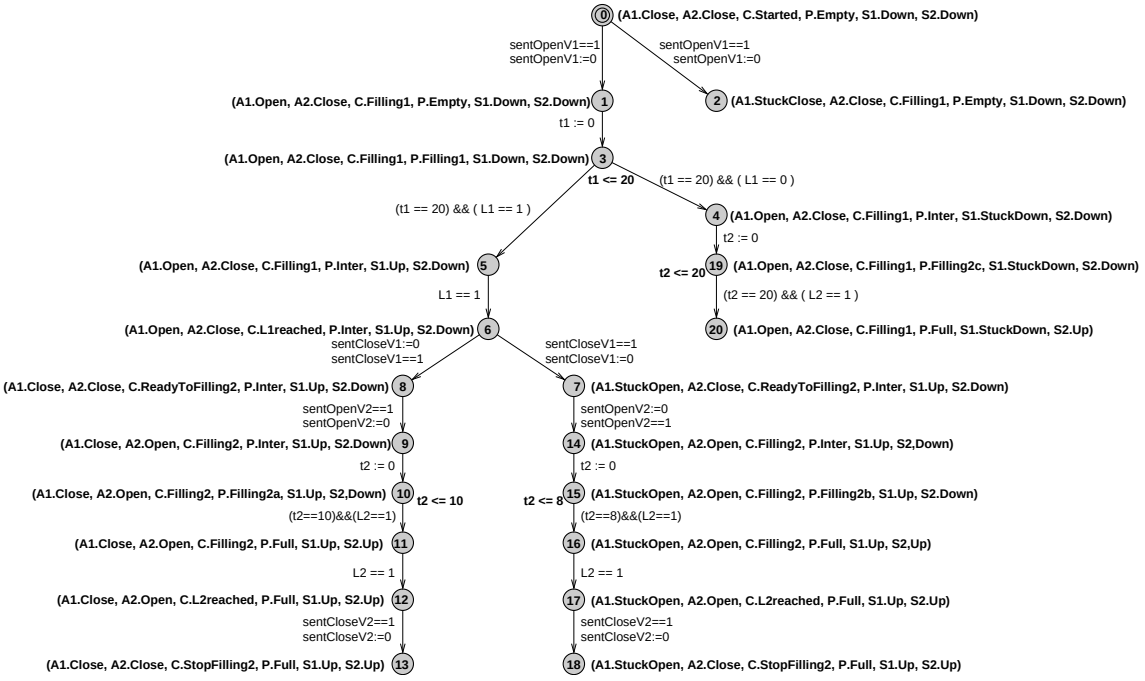


FIGURE I.8 – Automate temporisé de l'esquisse du diagnostiqueur du procédé batch

2.2.3.2 Construction du diagnostiqueur

Nous avons défini dans [15], [17] un algorithme de déterminisation adapté à notre problème. Il est basé sur l'algorithme de construction de sous-ensemble. La figure I.9 montre

le résultat de notre méthode et l'automate obtenu du diagnostiqueur. Pour déterminiser, notre algorithme construit de nouveaux états puits et ajoute de nouvelles transitions. Le diagnostiqueur que nous avons obtenu permet bien de détecter et d'identifier tous les défauts que nous avons considérés, nous avons trois états puits pour identifier les défaillances *Stuck Close* et *Stuck Open* sur la vanne A_1 et la défaillance *Stuck Down* sur le capteur S_1.

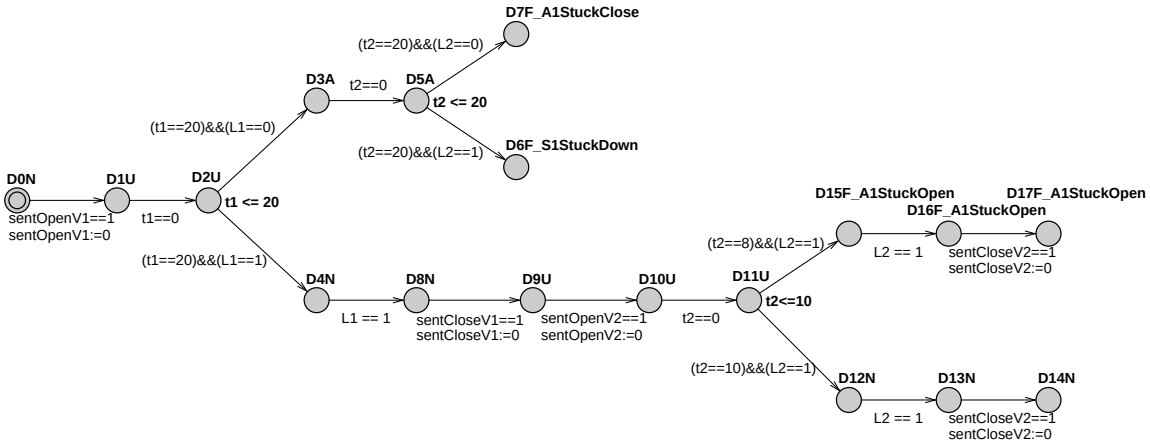


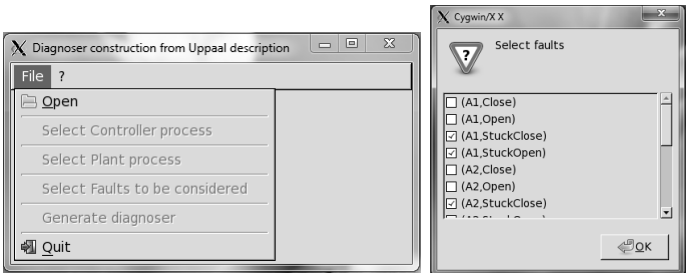
FIGURE I.9 – Automate temporisé du diagnostiqueur du procédé batch

2.2.3.3 Vérification du la correction du diagnostiqueur

Nous nous assurons de la correction diagnostiqueur par model cheking avec UPPAAL. Nous considérons la composition synchrone du modèle global du système avec le modèle du diagnostiqueur et nous vérifions que pour chaque défaillance : il n'y a pas de détection / identification manquante et que le diagnostiqueur ne produit pas de fausse détection.

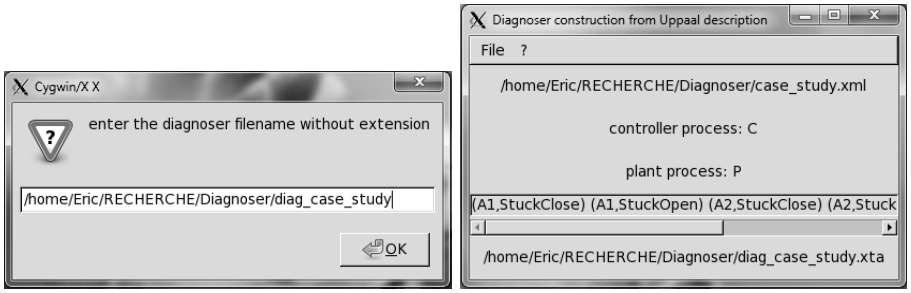
3 Conclusion et perspectives sur la construction de diagnostiqueurs

Nous avons proposé une méthodologie de construction automatique de diagnostiqueurs pour des systèmes dynamiques complexes modélisés par des automates temporisés communicants. L'étape de construction du diagnostiqueur se fait *hors ligne*, c'est-à-dire avant la mise en fonctionnement du système à surveiller. Le diagnostiqueur est utilisé de manière *passive en ligne*, c'est-à-dire qu'on réalise la surveillance du système pendant son fonctionnement (en ligne) sans connaissance a priori de la présence d'un défaut ou non et le diagnostic est obtenu par l'analyse le flot des entrées et des sorties du système sans influencer le choix des entrées pour guider le diagnostic (analyse passive). Nous avons considéré des fautes permanentes sur les actionneurs et les capteurs et exprimé les contraintes temporelles de la dynamique des procédés sous forme de tests d'égalités sur des durées de temporisation.



(a) Sélection du système dynamique complexe

(b) Sélection des défaillances à prendre en compte



(c) Génération du diagnostiqueur

(d) Résultats de diagnostiqueur généré

FIGURE I.10 – Notre logiciel de génération automatique de diagnostiqueur

Nous avons élaboré les algorithmes de construction de l'esquisse du diagnostiqueur et du diagnostiqueur. Pour valider le diagnostiqueur construit par notre algorithme, nous utilisons la technique de vérification formelle par model checking. Notre méthodologie a été appliqué sur deux études de cas complètes en implémentant nos algorithmes et en développant un logiciel. Ce logiciel possède une IHM (figure I.10) par laquelle l'utilisateur décrit son système de production industriel à diagnostiquer en fournissant des automates temporisés communicants au format UPPAAL. Le logiciel fournit en sortie un fichier décrivant le diagnostiqueur au format UPPAAL.

Les perspectives de ce travail sont de prendre en compte les incertitudes sur les durées des tâches qui peuvent apparaitre dans certains procédés (utilisation d'intervalles de validité pour les tâches). Une seconde perspective est l'étude de la diagnosticabilité des défaillances pour de tels systèmes temporisés. Dans la prochaine section, nous expliquerons cette thématique de recherche et les résultats que nous avons obtenus dans le cadre des systèmes à événements discrets non temporisés.

Chapitre II

Etude de la diagnosticabilité de systèmes à événements discrets

1 Présentation générale : contexte, état de l'art, positionnement

1.1 Contexte

Pour réaliser le diagnostic d'un système industriel complexe, il est nécessaire d'avoir suffisamment de données observables qui permettent de détecter l'occurrence d'une défaillance et de pourvoir discriminer quelle défaillance est survenue et ainsi la localiser et l'identifier. Ces propriétés sur la *qualité* des données observables nécessaires au diagnostic se définissent par la propriété de diagnosticabilité. Si un système possède la propriété de diagnosticabilité, alors nous avons la certitude qu'il est possible de construire un diagnostiqueur qui sera toujours capable de détecter une défaillance et qu'il localisera et identifiera toujours la source de cette défaillance. L'étude de la diagnosticabilité doit être un préalable à la construction du diagnostiqueur. Si cette propriété n'est pas obtenue, il est alors nécessaire d'avoir des données observables supplémentaires qui permettront de discriminer chacune des défaillances possibles en ajoutant de nouveaux capteurs dans le système ou en repositionnant les capteurs existants ou en modifiant le système pour permettre d'obtenir la propriété de diagnosticabilité.

1.2 Etat de l'art

Dans les méthodes de diagnostic à bases de modèles, les systèmes dynamiques peuvent être classés selon leur modélisation en systèmes continus, systèmes à événements discrets (SED), systèmes à événements discrets temporisés, systèmes hybrides ou systèmes stochastiques. Pour chacune de ces modélisations, des approches différentes d'études de la diagnosticabilité ont été développées. Dans le cadre des systèmes continus, le système est décrit par plusieurs modes de fonctionnement, chaque mode de fonctionnement est défini par un ensemble de variables continues auxquelles sont reliées des contraintes. De nombreux travaux ont étudié la diagnosticabilité des systèmes continus, ceux-ci sont principalement

issus de la communauté de l'intelligence artificielle DX (Diagnosis eXpert system) ou de la communauté automatique FDI (Fault Detection and Isolation).

L'approche DX exprime explicitement le lien entre un composant et les formules logiques du premier ordre décrivant son comportement. Les travaux fondateurs de l'approche DX sont ceux de Reiter [18] et De Kleer et al. [19], [20]. Pour analyser la diagnosticabilité des systèmes, l'approche DX utilise le concept de *conflit*. Un conflit est ensemble de composants d'un système dont les hypothèses de comportement correct ne sont pas consistantes avec les observations réelles.

L'approche FDI étudie la diagnosticabilité avec les *relations de redondance analytique* pour construire une matrice de signature des fautes puis de vérifier si toute faute est détectable (discriminable de l'état nominal) et que toute paire de fautes est discriminable. Deux fautes sont discriminables si leurs signatures sont complètement différentes. Le lecteur trouvera dans les travaux de Cordier et al. [21] une analyse comparative des approches DX et FDI sur la diagnosticabilité.

Dans le cadre des systèmes à événements discrets, les premiers travaux sur la diagnosticabilité ont été publiés par Lin dans [22] pour les automates à états finis, et étendus par Sampath et al. dans [1], [2]. Ushio et al. ont étudié la diagnosticabilité dans [23] pour les réseaux de Petri. La diagnosticabilité des SED se définit informellement ainsi : un SED est diagnosticable si et seulement après toute occurrence d'une faute, il existe une séquence finie d'observables qui ne se produit pas en l'absence de cette faute (cette séquence finie d'observables permet alors de discriminer la faute). Comme nous l'avons relaté dans le chapitre I, la modélisation du système peut-être centralisée, décentralisée ou distribuée. Des méthodes d'étude de la diagnosticabilité ont été développées pour chacune de ces structures.

1.2.1 Etude de la diagnosticabilité des SED par une approche centralisée

Dans le cadre des systèmes à structure centralisée, les travaux de références sont ceux de Sampath et al. [1], [2]. Les auteurs ont développé une méthodologie de construction d'un diagnostiqueur et un test de diagnosticabilité qui est basé est sur l'étude du diagnostiqueur construit. L'inconvénient de leur approche est sa complexité exponentielle par rapport au nombre d'états du système et doublement exponentielle par rapport au nombre de types de fautes. En effet, le nombre d'états du diagnostiqueur peut être exponentiel par rapport au nombre d'états du système. Dans l'approche utilisée par Sampath et al., le système est modélisé par un automate à états finis et l'occurrence d'une faute est modélisée par un événement. Il existe également une autre modélisation à base d'états des défauts, c'est l'approche utilisée par Zad et al. dans [24]. Dans cette approche qui modélise le système également par un automate à états finis, l'occurrence d'une faute est modélisée par l'arrivée sur un état fautif particulier, aussi il est requis que l'espace d'état de l'automate puisse être partitionné selon l'état du système (nominal ou bien un mode de défaillance particulier). Leur approche s'inspire de celle de Sampath et al., le test de la diagnosticabilité repose également sur le diagnostiqueur qui doit être construit préalablement. Leur complexité est

également exponentielle par rapport au nombre d'états du système et doublement exponentielle par rapport au nombre de types de fautes A la suite des travaux de Sampath et al., de nouvelles approches pour tester la diagnosticabilité de systèmes modélisés par des automates à états finis ont été développées pour améliorer sa complexité : ce sont des approches qui ont une complexité polynomiale par rapport au nombre d'états du système. Ce gain en complexité est obtenu par des algorithmes qui cherchent à détecter des cycles de séquences d'événements observables ambigües entre une séquence se produisant après l'occurrence de la faute et une séquence normale et cela sans avoir à construire préalablement le diagnostiqueur. Les travaux de Jiang et al. dans [25] utilisent un modèle intermédiaire appelé *twin plant*, les travaux de Yoo et Lafortune dans [26] manipulent un modèle intermédiaire appelé *verifier* et l'approche de Cassez et Tripakis dans [27] utilise un modèle intermédiaire appelé *observer*, leur test de diagnosticabilité est basé sur le test du vide des automates de Büchi, sa complexité est également polynomiale. Dans [28], le lecteur trouvera une présentation et une comparaison entre différentes techniques de test de diagnosticabilité sur les automates et les réseaux de Petri. Il existe d'autres approches pour l'étude de la diagnosticabilité lorsque le système centralisé est modélisé par un automate à états finis. Dans [29], l'analyse de la diagnosticabilité se fait à la volée par des techniques énumératives et symboliques. Cette approche construit une variante du diagnostiqueur appelé *hybrid diagnoser* en utilisant le graphe d'observation symbolique. Les techniques de model checking ont également été utilisées pour vérifier la diagnosticabilité de systèmes modélisés par des automates à états finis [30]-[32]. De nombreux travaux ont été menés sur l'étude de la diagnosticabilié lorsque le système centralisé est modélisé par un réseau de Petri étiqueté [33]-[39]. En effet, la modélisation par réseaux de Petri offre une représentation du système à événements discrets plus compacte permettant ainsi d'obtenir des algorithmes avec une complexité moindre que ceux basés sur les automates. Dans ces travaux, différentes hypothèses ou outils d'analyse ont été pris en compte : places partiellement observables, transitions partiellement observables, réseau de Petri sans blocage borné ou non borné, utilisation de la notion d'explication minimale, résolution par programmation linéaire en nombre entier, technique de dépliage, adaptation de l'approche *twin-plant* aux réseaux de Petri, calcul de la diagnosticabilité à la volée, utilisation du graphe d'atteignabilité appelé *verifier net*, etc.

1.2.2 Etude de la diagnosticabilité des SED ayant une architecture décentralisée

Il existe de très nombreux travaux sur la diagnosticabilité de systèmes locaux appelé *codiagnosticabilité*. Le lecteur pourra se référer à [40] pour une présentation de ce domaine de recherche. Nous ne présentons dans ce paragraphe que quelques exemples de travaux. Debouk et al. présentent dans [41] l'un de premiers travaux sur la diagnosticabilité décentralisée. Les auteurs proposent plusieurs protocoles pour la mise en oeuvre d'une architecture de diagnostic distribuée avec l'étude de leur diagnosticabilité. Contant et al proposent dans [42] une généralisation de l'algorithme de vérification de la diagnosticabilité proposé par

Sampath et al. au cas de systèmes ayant une architecture décentralisée modulaire. [43]-[45] présentent différentes approches d'analyse de la codiagnosticabilité basées sur l'approche du vérificateur (*verifier,twin plant*) en utilisant le formalisme des automates à états finis. Il existe également des travaux sur l'analyse de la diagnosticabilité de réseaux de Petri dans une architecture décentralisée [46], [47].

1.2.3 Etude de la diagnosticabilité des SED ayant une architecture distribuée

Dans le cadre des systèmes à architecture distribuée, plusieurs approches sur l'étude de leurs diagnosticabilités ont été développées, par exemple [48]-[51]. Pencolé dans [48] utilise la diagnosticabilité locale pour vérifier la diagnosticabilité d'une faute localisée à un sous-système puis définit un test de diagnosticabilité incrémentale qui évite la construction d'un vérificateur pour le système dans sa globalité si cela n'est pas nécessaire dans le cas où l'on détecte à une étape du test la non-diagnosticabilité. Dans [49], Schumann et Pencolé présentent une amélioration des résultats publiés dans [48], l'hypothèse d'observabilité vivante n'est plus requise, la taille des vérificateurs locaux est réduite et l'analyse de la diagnosticabilité d'un système distribué est réalisée par agrégation distribuée. Les travaux de Ye et Dague dans [50] présentent une optimisation de la construction des vérificateurs locaux de [48]. Dans [51], les auteurs présentent une approche à base de model checking pour l'étude de la diagnosticabilité de systèmes à architecture distribuée.

1.2.4 Etude de la diagnosticabilité des systèmes à événements discrets temporisés

L'étude de la diagnosticabilité dans le cadre de systèmes à événements discrets temporisés est plus récente, et a donc jusqu'à présent une production scientifique moins développée que celle sur la diagnosticabilité dans les systèmes non temporisés. Les travaux publiés ont pris en considération essentiellement les systèmes à architecture centralisée modélisés par des automates temporisés [52]-[55] ou par des réseaux de Pétri temporels [56]-[59]. Des travaux ont été menés également sur l'étude de la diagnosticabilité de systèmes temporisés à architecture décentralisée ou distribuée [60]-[62].

1.2.5 Etude de la diagnosticabilité des systèmes hybrides

L'étude de la diagnosticabilité s'est également porté sur les systèmes hybrides [63], [64]. Différentes approches ont été développés : combinaison des méthodes d'analyse de la diagnosticabilité issues des communautés FDI et DES, extension de la technique de *twin plan* ou *verifier* aux systèmes hybrides, utilisation des techniques d'abstraction. Voici quelques exemples de travaux sur la diagnosticabilité de systèmes hybrides [65]-[72].

1.2.6 Etude de la diagnosticabilité des systèmes stochastiques

Pour terminer ce tour d'horizon des méthodes d'analyses de la diagnosticabilité, celles-ci ont également porté sur les systèmes stochastiques, comme [73]-[76].

Le lecteur intéressé trouvera dans les manuscrits de thèses [77]-[82] des présentations plus détaillées sur toutes les méthodes d'analyse de diagnosticabilité.

1.3 Positionnement

Le travail que nous avons mené sur la diagnosticabilité était d'une part de se former à ce domaine en réalisant un état de l'art exhaustif et de développer une nouvelle méthode de test de diagnosticabilité dans le cadre d'un système manufacturier non temporisé avec une approche centralisée. La modélisation que nous utilisons est un mixte entre la modélisation utilisée par Sampath et al. [1], [2], où l'occurrence d'une faute est modélisée par un événement, et celle utilisée par Zad et al. [24] à base d'états des défauts où l'occurrence d'une faute est modélisée par l'arrivée sur un état fautif particulier, l'espace d'état de l'automate est alors partitionné selon l'état du système. Cette modélisation nous permet de considérer des fautes permanentes (l'état fautif est un état puit ou absorbant) de caractère aléatoire (l'occurence de la faute est un événement). En plus des deux hypothèses classiques considérées dans le domaine de la diagnosticabilité (le modèle ne contient pas de cycle non observable et le langage généré par le modèle est vivant) , nous prenons en compte une troisième hypothèse : nous considérons que des fautes imprédictibles : nous supposons ne pas avoir à notre disposition des données montrant la dégradation d'un équipement jusqu'à sa panne. L'idée de notre test de diagnosticabilité consiste pour chaque événement de faute f apparaissant sur une transition du modèle composite G, nous construisons deux automates observateurs $G_o(x_f)$ et $G_o(x_{\overline{f}})$ puis nous testons s'il existe un cycle dans $G_o(x_f)$ qui peut être plongé dans $G_o(x_{\overline{f}})$. Le premier avantage de cette approche est qu'elle ne nécessite pas la construction du diagnostiqueur comme dans [1], [2], [24]. Le second avantage est qu'elle permet de réduire l'espace d'états à explorer pour l'analyse de la diagnosticabilité. En effet, du à la troisième hypothèse, les états initiaux de $G_o(x_f)$ et $G_o(x_{\overline{f}})$ ne sont pas nécessairement l'état initial du modèle composite G comme dans [25], [26], [54], mais l'état source de la transition étiquetée par l'événement de faute f. Notre méthode de test a une complexité polynomiale dans le nombre d'états du système composite. Notre complexité est la même que celle de Yoo et Lafortune dans [26] dans le pire des cas, mais est plus petite dans le cas moyen. Nous considérons un scénario de défaillance unique, cependant nos résultats peuvent être étendus à des scénarios de défaillances multiples comme expliqué dans [1].

2 Synthèse des travaux développés

2.1 Modélisation des systèmes manufacturiers non temporisés par automates à états finis

Nous modélisons le comportement fonctionnel d'un système manufacturier par un système à événements discrets représenté par un automate à états finis $G = \langle X, \Sigma, \delta, x_0 \rangle$ où

X est l'ensemble fini des états, Σ est l'ensemble des événements, $\delta \subseteq X \times \Sigma \times 2^X$ est la fonction de transition et x_0 est l'état initial.

Nous partitionnons l'ensemble des événements Σ en événements observables Σ_o et événements non observables Σ_{uo} ($\Sigma = \Sigma_o \dot{\cup} \Sigma_{uo}$). Les événements observables sont les commandes issues du contrôleur aux actionneurs et les données fournies par les capteurs au contrôleur. Les événements non observables contiennent les occurrences des fautes sur les actionneurs ou les capteurs. L'ensemble des événements de fautes est représenté par $\Sigma_f \subseteq \Sigma_{uo}$ et il est partitionné en p ensembles disjoints correspondant aux différents p types de fautes, $\Sigma_f = \Sigma_{f_1} \dot{\cup} \Sigma_{f_2} \dot{\cup} \ldots \dot{\cup} \Sigma_{f_p}$. Nous représentons cette partition par Π_f.

Nous supposons un scénario à défaillance unique où au plus une défaillance peut survenir à un instant donné. Aussi le système peut être seulement dans un des $p+1$ modes du système : N (comportement nominal), $F_1, \ldots, F_p$. Nous représentons par $\mathcal{K} = \{N, F_1, \ldots, F_p\}$ l'ensemble des modes de fonctionnement du système.

Nous partitionnons l'ensemble des états de l'automate selons les $p+1$ modes du système, $X = X_N \dot{\cup} X_{F_1} \dot{\cup} \ldots \dot{\cup} X_{F_p}$. Les effets des défaillances sont supposés permanents, cela signifie qu'après apparition de la défaillance, le système reste dans le même mode de défaillance indéfiniment. Dans l'automate G, il y a donc aucune transition d'un état dans X_{F_i} à un état dans $X_N \cup (\bigcup_{j \neq i} X_{F_j})$. Nous définissons la fonction $\kappa : X \to \mathcal{K}$ telle que pour tout état $x \in X$, $\kappa(x)$ est le mode du système à l'état x.

2.2 Présentation de l'étude de cas

Nous illustrerons notre méthode d'analyse de la diagnosticabilité sur un exemple académique présenté dans [24]. Il s'agit d'une version simplifiée d'un système de chauffage, ventilation et climatisation (HVAC) composé d'une pompe, d'une vanne, d'un capteur de débit et d'un contrôleur. Le contrôleur ouvre la vanne (commande *ve – valve enable*) puis active la pompe (commande *pe – pump enable*). Après un moment, le contrôleur désactive la pompe (commande *pd – pump disable*) puis ferme la vanne (commande *vd – valve disable*), et cette séquence est rebouclée sur elle-même. La figure II.1 présente ce système HVAC simplifié et la séquence du contrôleur sous la forme d'un Grafcet. Pour la modélisation de la vanne, nous considérons les fautes suivantes :

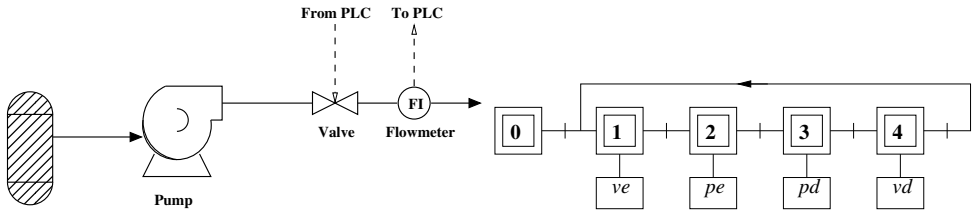
— f_1 : la vanne reste fermée lors d'une demande d'ouverture, état SC (*stuck closed*).

— f_2 : la vanne reste ouverte lors d'une demande de fermeture, état SO (*stuck open*).

La pompe et le capteur de débit sont supposés fiables, non sujets à des défaillances. La sortie du capteur de débit est soit fl (*flow*) quand la vanne est ouverte et que la pompe fonctionne en même temps, ou $nofl$ (*no flow*) autrement.

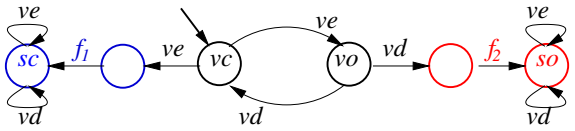
La figure II.2 représente les modèles de la vanne, de la pompe et du comportement fonctionnel du capteur de débit selon l'état de la pompe et de la vanne.

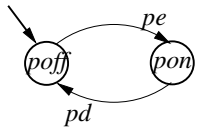
La figure II.3 représente sous la forme d'un automate d'états finis le système composite résultant du fonctionnement en parallèle du contrôleur, de la pompe, de la vanne et du capteur de débit. L'ensemble des événements observable est $\Sigma_o = \{ve, vd, pe, pd, fl, nofl\}$



(a) Schéma du procédé (PDF) (b) Contrôleur (Grafcet)

FIGURE II.1 – Exemple illustratif : système HVAC simplifié





Valve	Pump	Flowmeter
vc	*poff*	*nofl*
vc	*pon*	*nofl*
vo	*poff*	*nofl*
vo	*pon*	*fl*
sc	*poff*	*nofl*
sc	*pon*	*nofl*
so	*poff*	*nofl*
so	*pon*	*fl*

FIGURE II.2 – Modélisation des composants du système HVAC simplifié

et l'ensemble des événements de fautes inobservables est $\Sigma_f = \{f_1, f_2\}$ qui correspond à l'ensemble des événements inobservables Σ_{uo}.

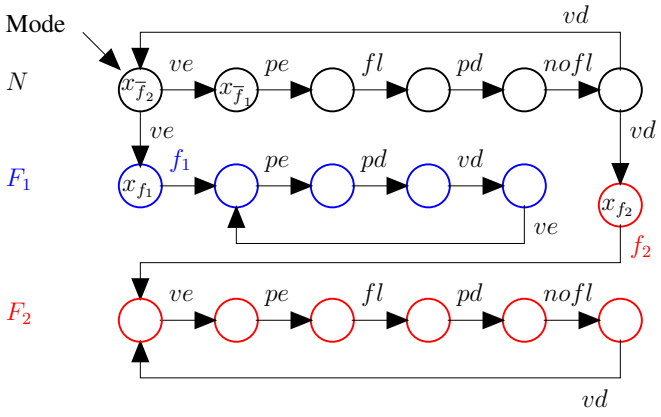


FIGURE II.3 – Système composite du système HVAC simplifié

2.3 Notations, définition de la diagnosticabilité

2.3.1 Définitions

Un *chemin* dans G est une séquence de transitions $\gamma = x_0\,\sigma_0\,x_1\,\ldots\,x_n\,\sigma_n\,x_{n+1}$ telle que pour tout i, $0 \leq i \leq n$, $(x_i, \sigma_i, x_{i+1}) \in \delta$. Ce chemin est un cycle si $x_{n+1} = x_0$. Nous représentons par $paths(x)$ l'ensemble de tous les chemins qui démarrent de l'état $x \in X$ et par $paths(G)$ l'ensemble de tous les chemins dans G, c.-à-d. $paths(G) = paths(x_0)$. La fonction κ est étendue aux chemins ainsi : $\kappa(x_0\,\sigma_0\,x_1\,\ldots\,x_n\,\sigma_n\,x_{n+1}) = \kappa(x_{n+1})$.

L'ensemble de toutes les séquences d'événements, appelées *traces* dans la suite de ce paragraphe, de longueur finie sur l'alphabet Σ est représenté par Σ^*. La séquence d'événements vide est dénotée $\epsilon \in \Sigma^*$. La trace d'un chemin $\gamma = x_0\,\sigma_0\,x_1\,\ldots\,x_n\,\sigma_n\,x_{n+1}$, dénotée $trace(\gamma)$, est la séquence des événements $\sigma_0\,\sigma_1\,\ldots\,\sigma_n$ apparaissant dans γ. La longueur d'une trace $s \in \Sigma^*$ est représentée par $|s|$.

La fonction de transition δ est récursivement étendue aux traces, telle que pour $x \in X$, $s \in \Sigma^*$, et $\sigma \in \Sigma$, $\delta(x, \epsilon) = x$ et $\delta(x, s\sigma) = \delta(\delta(x, s), \sigma)$ si $\delta(x, s)$ existe.

Un langage sur Σ est un sous-ensemble de Σ^*. Le langage généré par G est défini comme $L(G) = \{s \in \Sigma^* | \delta(x_0, s) \text{ existe}\}$. Pour simplifier les notations, dans la suite nous notons le langage $L(G)$ par L.

2.3.2 Définition de la diagnosticabilité

Nous présentons dans ce paragraphe la définition de la diagnosticabilité pour les systèmes à événements discrets telle qu'elle a été présentée par Sampath et al. dans [1]. Nous introduisons les notations suivantes utilisées dans cette définition. Nous définissons $\Psi(\Sigma_{f_i}) = \{s\,\sigma_f | \sigma_f \in \Sigma_{f_i}\}$, c.-à-d. $\Psi(\Sigma_{f_i})$ représente l'ensemble de toutes les traces dans L terminées par une faute de l'ensemble Σ_{f_i}. Nous représentons par L/s l'ensemble des suffixes de L qui commencent par s, c.-à-d. $L/s = \{t \in \Sigma^* | s\,t \in L\}$.

Le langage L peut être partiellement observé par l'opérateur de projection $P_{\Sigma_o} : \Sigma^* \to \Sigma_o^*$ qui est défini inductivement ainsi : $P_{\Sigma_o}(\epsilon) = \epsilon, P_{\Sigma_o}(\sigma) = \sigma$ si $\sigma \in \Sigma_o$, $P_{\Sigma_o}(\sigma) = \epsilon$ si $\sigma \in \Sigma_{uo}$ et pour chaque $s \in \Sigma^*$, $\sigma \in \Sigma$: $P_{\Sigma_o}(s\,\sigma) = P_{\Sigma_o}(s)\,P_{\Sigma_o}(\sigma)$. L'opérateur de projection P_{Σ_o} permet de filtrer les événements non observables.

Soit $\overline{s}$ la fermeture par préfixe d'une trace $s \in \Sigma^*$. Pour simplifier les notations, dans la suite nous notons $\Sigma_{f_i} \in s$ pour représenter que $\overline{s} \cap \Psi(\Sigma_{f_i}) \neq \emptyset$, c.-à-d. la trace s contient un événement de faute $\sigma_f \in \Sigma_{f_i}$.

Définition 1 ([1]) *Soit G la modélisation d'un système à événements discrets par un automate d'états finis. Les hypothèses suivantes sont considérées :*

(H1) Le langage $L(G)$ est un langage vivant, c.-à-d. il existe toujours une transition pour n'importe quel état de G.

(H2) Il n'existe pas dans G de cycle de transitions étiquetées par des événements non observables.

Le langage $L(G)$ (ou par abus de notation G) est dit diagnosticable par rapport à une partition des fautes Π_f et des événements observables Σ_o, des événements non observables Σ_{uo} et des événements de fautes $\Sigma_f \subseteq \Sigma_{uo}$ si et seulement si :

$$(\forall i \in \Pi_f)\ (\exists n_i \in \mathbb{N})\ (\forall s \in \Psi(\Sigma_{f_i}))\ (\forall t \in L/s)\ [|t| \geq n_i \Rightarrow D]$$

où la condition de diagnosticabilité D est définie par : $\forall w \in P^{-1}_{\Sigma_o}(P_{\Sigma_o}(st)) \cap L \Rightarrow \Sigma_{f_i} \in w$.

Cette définition signifie qu'un système à événements discrets G est diagnosticable si et seulement si toute trace st qui contient une faute de l'ensemble Σ_{f_i} peut être identifiée de manière unique de toute autre trace w ne comportant pas de fautes de l'ensemble Σ_{f_i} après un délai fini de détection, spécifiquement en au moins n_i événements après la trace s. Par conséquent, un système à événements discrets n'est pas diagnosticable si et seulement il existe deux traces de longueur non bornée qui possède la même séquence d'événements observables et l'une d'entre elles contient un événement de faute et l'autre pas.

2.4 Test de diagnosticabilité

Notre méthode de test de diagnosticabilité assume les hypothèses suivantes :

(H1) Le langage $L(G)$ est un langage vivant.

(H2) Il n'existe pas dans G de cycle de transitions étiquetées par des événements non observables.

(H3) Les fautes sont imprédictibles : tout événement de faute f_i ne peut pas être prédit avant son occurrence en se basant sur les événements observables.

Les hypothèses (H1) et (H2) sont communes aux différentes méthodes d'analyse de la diagnosticabilité existantes. L'hypothèse (H3) suppose l'absence de données montrant la dégradation d'un équipement jusqu'à sa panne. Pour présenter notre méthode, nous commençons par introduire quelques notations :

- x_{f_i} représente l'état source d'une transition étiquetée par un événement de faute de type Σ_{f_i}.
- γ_{f_i} représente le chemin débutant par l'état initial x_0 de G et se finissant par l'état x_{f_i}. c.-à-d. $\gamma_{f_i} = x_0 \dots x_{f_i}$.
- $\gamma_{\overline{f}_i}$ représente un chemin dont l'état de départ est x_0 et ayant la même trace d'événements observables que γ_{f_i}, c.-à-d. $P_{\Sigma_o}(trace(\gamma_{f_i})) = P_{\Sigma_o}(trace(\gamma_{\overline{f}_i}))$, mais dont les extensions ne contiennent pas un événement de faute de type Σ_{f_i} (tous les états accessibles à partir de $x_{\overline{f}_i}$ sont donc dans $X_N \cup (\bigcup_{j \neq i} X_{F_j})$).
- $x_{\overline{f}_i}$ représente l'état destination du chemin $\gamma_{\overline{f}_i}$, c.-à-d. $\gamma_{\overline{f}_i} = x_0 \dots x_{\overline{f}_i}$.
- $G(x_{f_i})$ représente le sous-automate de G dont l'état initial est x_{f_i} et dont tous les états appartiennent à X_{F_i} seulement.
- $G(x_{\overline{f}_i})$ représente le sous-automate de G dont l'état initial est $x_{\overline{f}_i}$ et dont tous les états appartiennent à $X_N \cup (\bigcup_{j \neq i} X_{F_j})$ seulement.

— $L(G, x)$ représente le langage généré par G à partir de l'état x, c.-à-d. le langage du sous-automate $G(x)$.

Nous définissons le théorème suivant sur lequel se base notre algorithme de test de diagnosticabilité.

Théoreme 1 (propriété de diagnosticabilité) *Sous les hypothèses (H1), (H2) et (H3) ainsi qu'un scénario de défaillance unique, G est dit diagnosticable par rapport à une partition des fautes Π_f et des événements observables Σ_o, des événements non observables Σ_{uo} et des événements de fautes $\Sigma_f \subseteq \Sigma_{uo}$ si et seulement si :*

$$(\forall i \in \Pi_f)\ (\forall k \in \mathbb{N})\ (\forall u\, v^k \in P_{\Sigma_o}(L(G, x_{f_i})))[u\, v^k \notin P_{\Sigma_o}(L(G, x_{\overline{f}_i}))]$$

La preuve de ce théorème est dans nos publications [83], [84].

Nous utilisons ce théorème pour construire notre test de diagnosticabilité. L'algorithme II.1 présente de manière synthétique les étapes pour tester la diagnosticabilité.

Algorithme II.1 Test de diagnosticabilité

procedure TESTDIAGNOSTICABILITÉ

Pour chaque transition étiquettée par une événement de faute f_i :

1. Construire les deux automates $G(x_{f_i})$ et $G(x_{\overline{f}_i})$.
2. Construire les deux automates non-déterministes $G_o(x_{f_i})$ et $G_o(x_{\overline{f}_i})$ dont les langages sont respectivement $L(G_o(x_{f_i})) = P_{\Sigma_o}(L(G(x_{f_i}))$ et $L(G_o(x_{\overline{f}_i})) = P_{\Sigma_o}(L(G(x_{\overline{f}_i}))$.
3. Construire un automate non-déterministe $G_o^{\times}(x_{f_i}, x_{\overline{f}_i})$ dont le langage associé est $L(G_o^{\times}(x_{f_i}, x_{\overline{f}_i}))$ défini par $P_{\Sigma_o}(L(G(x_{f_i}))) \cap P_{\Sigma_o}(L(G(x_{\overline{f}_i})))$ et tester s'il existe dans $G_o^{\times}(x_{f_i}, x_{\overline{f}_i})$ un cycle. Si un cycle est présent alors G n'est pas diagnosticable, autrement G est diagnosticable.

La construction de l'automate non-déterministe $G_o(x_{f_i}) = \langle X_o^{f_i}, \Sigma_o, \delta_o^{f_i}, x_{f_i}\rangle$ à partir de l'automate $G(x_{f_i}) = \langle X_{F_i}, \Sigma, \delta, x_{f_i}\rangle$ est réalisée ainsi :

— $X_o^{f_i} = \{x \in X_{F_i} \,|\, \exists \gamma \in paths(x_{f_i})$ avec $P_{\Sigma_o}(trace(\gamma)) \neq \epsilon\}$.

— $\delta^{f_i} \subseteq X_o^{f_i} \times \Sigma_o \times X_o^{f_i}$ est l'ensemble des transitions : $(x, \sigma_o, x') \in \delta_o^{f_i}$ si et seulement si il existe un chemin $\gamma = x\, \sigma_1\, x_1\, \ldots\, \sigma_n x_n \sigma_o\, x'$ dans G tel que $\forall i \in [1 \ldots n]\, \sigma_i \in \Sigma_{uo}$ et $\sigma_o \in \Sigma_o$.

La langage de $G_o(x_{f_i}) = \langle X_o^{f_i}, \Sigma_o, \delta_o^{f_i}, x_{f_i}\rangle$ est alors $L(G_o(x_{f_i})) = P_{\Sigma_o}(L(G(x_{f_i}))$.
La construction du second automate non-déterministe $G_o(x_{\overline{f}_i})$ est faite de manière similaire à partir de $G(x_{\overline{f}_i})$.

La construction de l'automate produit $G_o^{\times}(x_{f_i}, x_{\overline{f}_i}) = \langle X_o^{\times}, \Sigma_o, \delta_o^{\times}, (x_{f_i}, x_{\overline{f}_i})\rangle$ est obtenue comme suit :

— $X_o^{\times} = \{\, (x_1, x_2) \in X_{F_i} \times X_N \cup (\bigcup_{j \neq i} X_{F_j}) \,|\, \exists \gamma_1 \in paths(x_{f_i})$ avec $P_{\Sigma_o}(trace(\gamma_1)) \neq \epsilon$ et $\exists \gamma_2 \in paths(x_{\overline{f}_i})$ avec $P_{\Sigma_o}(trace(\gamma_2)) \neq \epsilon\,\}$.

— $\delta_o^{\times} \subseteq X_o^{\times} \times \Sigma_o \times X_o^{\times}$ est l'ensemble des transitions : $((x_1, x_2), \sigma_o, (x_1', x_2')) \in \delta_o^{\times}$ si et seulement si il existe un chemin $\gamma_1 = x_1\, \sigma_1^1\, x_2^1\, \ldots\, \sigma_n^1 x_n^1 \sigma_o\, x_1'$ dans G tel que

$\forall i \in [1 \dots n]\, \sigma_i^1 \in \Sigma_{uo}$ et $\sigma_o \in \Sigma_o$ et il existe un chemin $\gamma_2 = x_2\, \sigma_1^2\, x_2^2\, \dots\, \sigma_m^2 x_m^2 \sigma_o\, x_2'$ dans G tel que $\forall i \in [1 \dots m]\, \sigma_i^2 \in \Sigma_{uo}$.

L'étude de la complexité de cet algorithme est fournie dans [83], [84]. La complexité de notre algorithme est en $\mathcal{O}(|X|^2 \times |\Sigma_o| \times |\Sigma_f|)$. Comparée aux tests polynomiaux de [25], [54] qui ont une complexité polynomiale du quatrième ordre dans le nombre d'états du modèle composite, notre méthode a une complexité plus faible, elle a en effet une complexité polynomiale du deuxième ordre dans le nombre d'états de G.

2.5 Illustration de l'étude de la diagnosticabilité

Nous illustrons notre test de diagnosticabilité sur l'exemple académique d'un système simplifié de chauffage, ventilation et climatisation présenté précédemment. L'ensemble des événements observables est $\Sigma_o = \{ve, vd, pe, pd, fl, nofl\}$ et l'ensemble des événements de fautes non observables est $\Sigma_f = \{f_1, f_2\}$.

2.5.1 Analyse de la diagnosticabilité de la faute f_1

Les étapes du test de diagnosticabilité de f_1 sont illustrées sur la figure II.4.

1. Nous étiquetons les états correspondants du système composite avec x_{f_1} et $x_{\overline{f}_1}$ (voir figure II.3).
2. Nous construisons les sous-automates $G_o(x_{f_1})$ (respectivement $G_o(x_{\overline{f}_1})$) dont l'état initial est x_{f_1} et consistant d'états appartenant uniquement à X_{F_1} (respectivement avec l'état initial $x_{\overline{f}_1}$ et consistant d'états appartenant uniquement à $X_N \cup X_{F_2}$).
3. Nous construisons $G_o^{\times}(x_{f_1}, x_{\overline{f}_1})$, puis nous testons la présence d'un cycle dans cet automate produit. Dans notre cas, il n'y a pas de cycle, par conséquent la faute f_1 est diagnosticable.

2.5.2 Analyse de la diagnosticabilité de la faute f_2

Les étapes du test de diagnosticabilité de f_2 sont illustrées sur la figure II.5.

1. Nous étiquetons les états correspondants du système composite avec x_{f_2} et $x_{\overline{f}_2}$ (voir figure II.3).
2. Nous construisons les sous-automates $G_o(x_{f_2})$ (respectivement $G_o(x_{\overline{f}_2})$) dont l'état initial est x_{f_2} et consistant d'états appartenant uniquement à X_{F_2} (respectivement avec l'état initial $x_{\overline{f}_2}$ et consistant d'états appartenant uniquement à $X_N \cup X_{F_1}$).
3. Nous construisons $G_o^{\times}(x_{f_2}, x_{\overline{f}_2})$, puis nous testons la présence d'un cycle dans ce automate produit. Dans notre cas, il y a un cycle, par conséquent la faute f_2 n'est pas diagnosticable. En effet dans le mode de défaillance F_2 où la vanne reste ouverte lors d'une demande de fermeture, les événements observables sont les mêmes que dans le mode nominal.

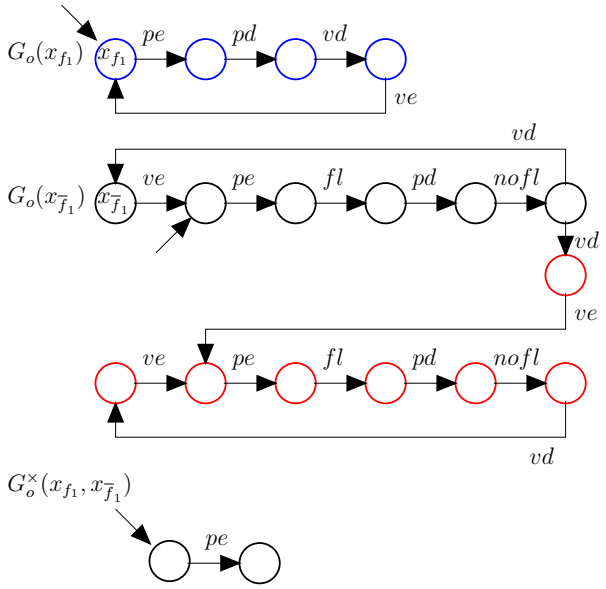


FIGURE II.4 – Test de diagnosticabilité de la faute f_1

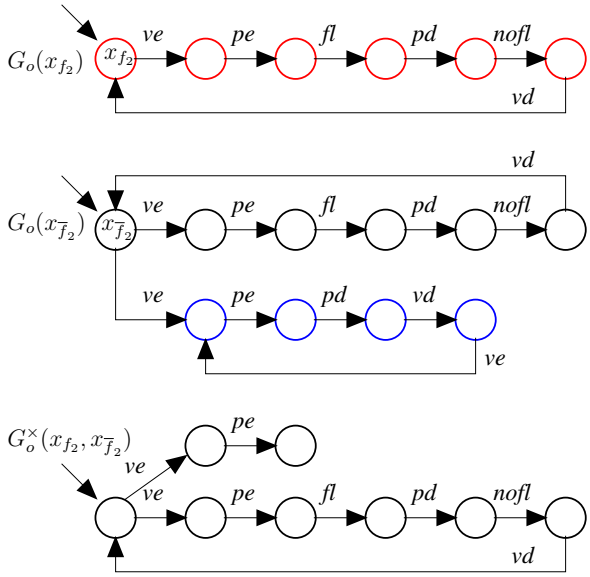


FIGURE II.5 – Test de diagnosticabilité de la faute f_2

3 Conclusion et perspectives sur l'étude de la diagnosticabilité

Notre étude sur la diagnosticabilité nous a permis de faire un état de l'art exhaustif sur ce domaine et de développer un test de diagnosticabilité pour les systèmes à événements discrets non temporisés ayant une complexité polynomiale en nombre d'états du système composite.

Les perspectives à moyen terme de ce travail sont d'étendre notre approche d'une part aux systèmes décrits de manière modulaire sous la forme d'un réseau d'automates temporisés communicants tel que ceux décrits dans le chapitre I. Les objectifs seront alors de développer une méthode d'analyse de diagnosticabilité prenant en compte des informations temporelles et d'étudier la codiagnosticabilité afin de ne pas construire le modèle composite. Une perspective à plus long terme est d'étudier dans le cadre des systèmes hétérogènes qui combinent différents domaines (électronique, mécanique, réseau et logiciel) comme les systèmes cyber-physiques ou systèmes embarqués, les relations entre les informations de testabilité de tels systèmes avec la propriété de diagnosticabilité et des propriétés voisines telles que la prédictibilité (qui est plus forte que la diagnosticabilité) et la manifestabilité (qui est plus faible que la diagnosticabilité).

Chapitre III

Génération de l'automate comportemental d'un système manufacturier

1 Présentation générale : contexte, état de l'art, positionnement, originalité

La détection d'une défaillance par l'approche *diagnostiqueur* nécessite un modèle du comportement nominal (sans faute) du système pour détecter une déviation de son comportement attendu. Cependant, les étapes de localisation et d'identification nécessitent une connaissance du comportement du système dans ses modes défaillants. Une grande partie des travaux sur le développement de méthodes de diagnostic des systèmes à événements discrets présupposent avoir une modélisation complète du système à diagnostiquer, c'est-à-dire son comportement nominal et son comportement avec fautes dans un même formalisme de système à événements discrets (automates à états finis, réseaux de Petri, réseaux de files d'attente, etc.). Toutefois, les systèmes actuels sont de plus en plus complexes et ils font intervenir de très nombreux composants, il est alors difficile (ou dans certains cas impossible) de modéliser sous la forme de systèmes à événements discrets le comportement d'un composant dans tous ses modes de défaillances ainsi que toutes les interactions/conséquences d'un composant défaillant sur les autres composants et sur la dynamique du système.

Des travaux de recherche ont été menés sur le développement de méthodes de diagnostic où seule la description du comportement nominal sous la forme d'un système à événements discrets (SED) est connue (*diagnosis using fault-free models* [85]). Pour permettre les étapes de localisation et d'identification, le comportement du système dans ses modes de défaillances ou les conséquences des défaillances sont alors exprimés dans un autre formalisme que les SED. Parmi ces travaux, nous pouvons citer [86]-[94]. Dans [86]-[89], différentes méthodologies de modélisation de motifs d'événements temporels sont utilisées pour décrire la prévision des dates d'occurrences des événements : *les signatures temporelles causales*, *les motifs de conditions*, *les chroniques*. Ces formalismes permettent de décrire les séquences d'événements attendus ou fautifs et les relations temporelles entre les événements. La surveillance de ces motifs temporels permet de détecter une défaillance et aide à

la localiser. Dans [90], [91], la détection d'une défaillance se fait par la comparaison entre les signaux observés et les signaux attendus, les résidus générés sont alors utilisés pour la localisation et l'identification de la défaillance. Cette méthodologie est étendue dans [92] aux automates temporisés. Sayed-Mouchaweh [93] utilise une représentation décentralisée du système à base de systèmes à événements discrets booléens. Chaque composant est représenté uniquement par son comportement nominal. Des contraintes temporelles représentées par un ensemble de conséquences attendues positives ou négatives sont utilisées pour modéliser la dynamique du système. Un monitoring de ces contraintes temporelles permet la localisation et l'identification des défaillances. Dans [94], les auteurs présentent une approche de diagnostic à base de consistance (*consistency-based diagnosis*) de la communauté DX [18]-[20] appliquée à des systèmes dynamiques représentés par des systèmes à événements discrets où seul le comportement nominal est décrit.

Le contexte de ce travail s'inscrit en partie dans ce cadre du diagnostic où seul le comportement nominal du système dynamique sous la forme d'un système à événements discrets est connu. Notre travail s'est porté sur l'aide à la construction d'un système à événements discrets temporisé décrivant la totalité des comportements possibles d'un procédé industriel, nominal et avec fautes. Les systèmes étant de plus en plus complexes et faisant appel à de nombreux composants, l'élaboration du comportement dynamique d'un système est compliquée et sujette à erreurs. Cette thématique de recherche fait suite à nos travaux sur la construction de diagnostiqueur (chapitre I), nous nous sommes donc intéressés aux systèmes manufacturiers en utilisant une description modulaire du procédé selon le paradigme capteur-contrôleur-actionneur. Nous modélisons alors par des automates communicants le contrôleur, les actionneurs et les capteurs. La modélisation des actionneurs et des capteurs prend en compte les fautes permanentes du type collage à l'état passant (stuck open) ou collage à l'état bloqué (stuck closed). Nous devons également modéliser l'automate comportemental du procédé industriel : c'est un automate temporisé communicant décrivant l'ensemble des comportements possibles de la spécification en présence ou non de fautes. Cependant la définition de cet automate comportemental est complexe dû à la manipulation d'un grand nombre de variables représentant les états des actionneurs et des capteurs et à leurs multiples interactions. *Nous proposons donc une méthode de construction automatique de cet automate comportemental à l'aide d'un ensemble de règles de causalité décrivant les interactions entre les actionneurs et les capteurs.* Cet ensemble de règles est plus simple à énoncer. Notre démarche gagnera alors en simplicité d'utilisation et en automatisation. Notre approche prend en compte actuellement uniquement les défaillances sur les actionneurs.

2 Synthèse des travaux développés

Nous présentons dans cette section notre méthode de génération de l'automate comportemental à partir d'un ensemble de règles de causalité.

2.1 Présentation de l'exemple illustratif

Nous illustrons cette méthode avec l'exemple du procédé batch présenté à la section 2.2.2, page 19. Ce système industriel est un procédé de neutralisation par remplissage d'une cuve. La cuve est équipée de deux capteurs de niveaux S_1 et S_2 (les variables $L1$ et $L2$ représentent l'état des capteurs) ainsi que de deux vannes A_1 et A_2 (les variables $V1$ et $V2$ représentent l'état des actionneurs). Le remplissage de la cuve doit suivre la séquence de contrôle suivante :

(E0) Quand le procédé est initialisé, la cuve doit être vide.

(E1) Ouverture de la vanne A_1, un ingrédient 1 remplit la cuve.

(E2) Lorsque le niveau $L1$ est atteint, fermeture de la vanne A_1 puis ouverture de la vanne A_2, un ingrédient 2 remplit alors la cuve.

(E3) Lorsque le niveau $L2$ est atteint, fermeture de la vanne A_2.

Le contrôleur gère selon une suite logique le déroulement ordonné des opérations à réaliser décrites par le GRAFCET (figure I.3, page 19). Dans le cas d'un GRAFCET sans branche parallèle, comme sur notre exemple, le contrôleur peut se modéliser par un unique automate communicant. La structure des étapes et transitions est reproduite par des localités et des transitions (figure I.4, page 20). Les commandes pour ouvrir et fermer les vannes sont modélisées par les signaux d'émission $OpenV1!$, $CloseV1!$, $OpenV2!$ et $CloseV2!$. Dans ce travail de recherche, nous avons considéré uniquement des GRAFCET sans branche parallèle, mais notre travail peut être étendu aux GRAFCET avec parallélisme en utilisant les schémas de modélisation développés dans [95].

Les vannes A_1 et A_2 sont représentées par des automates communicants (figure I.5, page 21). Les réceptions des commandes du contrôleur sur les vannes sont modélisées par les signaux de réception $OpenV1?$, $CloseV1?$, $OpenV2?$ et $CloseV2?$. Les variables d'état des vannes $V1$ et $V2$ sont initialisées à 0 (vannes fermées au départ du procédé) et mises à jour à 1 lorsque les vannes seront ouvertes. Pour notre exemple, nous considérons uniquement les fautes suivantes sur la vanne A_1 :

- f_1 : la vanne A_1 est bloquée en fermeture (stuck closed) ;
- f_2 : la vanne A_1 est bloquée en ouverture (stuck open).

Les capteurs S_1 et S_2 sont représentés par des automates communicants (figure III.1). Ils réagissent selon la dynamique du procédé. Les variables d'état des capteurs $L1$ et $L2$ sont initialisées à zéro, et mise à jour à 1 lorsque le niveau est atteint. Les transitions d'un capteur S_j sont de la forme $\ell_u^{sj} \xrightarrow[l_j:=value]{Sensor_j?} \ell_v^{sj}$ où $Sensor_j$ est un canal de synchronisation entre le comportement du procédé et le capteur S_j et l_j est la variable associée à l'état de ce capteur.

Nous ne prenons pas en compte les fautes sur les capteurs. Sur notre exemple, le contrôleur ne fait que remplir la cuve, aussi nous ne modélisons que les passages des états bas aux états hauts des capteurs. Si nous devions prendre en compte une vidange de la cuve, nous aurions alors placé des transitions allant des états hauts aux états bas.

(a) Capteur $L1$ (b) Capteur $L2$

FIGURE III.1 – Automates des capteurs de niveaux dans le procédé batch (sans fautes)

2.2 Modélisation du comportement dynamique du procédé par des règles de causalité

Le comportement dynamique du système est caractérisé par un ensemble $\mathcal{R} = \{r_1, \ldots, r_k\}$ de k règles de causalité entre les actionneurs et les capteurs. Ces règles ont la forme suivante : $r_x \stackrel{def}{=} g\{clk_i := O\} \Rightarrow (clk_i == K)Sensor_j!$.

La règle de causalité se comprend alors ainsi : quand la garde g est vérifiée, après K unités de temps le capteur S_j se déclenche. La garde g est une expression booléenne sur les variables entières associées aux actionneurs et aux capteurs, par exemple $(\overline{L1} \wedge V1 \wedge \overline{V2})$ exprime que le capteur S_1 est en position basse et que la vanne A_1 est ouverte et que la vanne A_2 est fermée. clk_i est une variable d'horloge utilisée pour déclencher le capteur S_j via l'action de synchronisation $Sensor_j!$ après K unités de temps (K une constante). L'horloge mesure le temps écoulé depuis que la garde est vérifiée, l'horloge est mise à zéro quand g est satisfaite.

Sur notre exemple applicatif, l'ensemble des règles R est défini par :

$$\begin{cases} r_1 \stackrel{def}{=} (\overline{L1} \wedge V1 \wedge \overline{V2})\{t_1 := O\} \Rightarrow (t_1 == 20)Sensor_1! \\ r_2 \stackrel{def}{=} (L1 \wedge \overline{V1} \wedge V2)\{t_2 := O\} \Rightarrow (t_2 == 10)Sensor_2! \\ r_3 \stackrel{def}{=} (L1 \wedge V1 \wedge V2)\{t_2 := O\} \Rightarrow (t_2 == 8)Sensor_2! \end{cases}$$

Notre méthodologie requiert que les règles reflètent toutes les situations de comportement possible et soient déterministes (une seule règle peut être appliquée à la fois). Dans notre exemple, la vanne A_1 pouvant être sujette aux fautes f_1 ou f_2, le procédé peut se trouver dans trois situations possibles :

1. Comportement sans fautes :
   - après ouverture de la vanne A_1, la cuve se remplit et atteint le niveau $L1$: la règle r_1 décrit cette dynamique ;
   - on procède alors à la fermeture de la vanne A_1 et à l'ouverture de la vanne A_2, le niveau L_2 est atteint : la règle r_2 décrit cette dynamique.
2. Comportement avec faute f_1 : la vanne A_1 étant bloquée en fermeture, le niveau L_1 reste en position basse, il n'y a donc aucune dynamique de comportement du procédé. Il n'y a donc pas de règles de causalités pour cette situation.
3. Comportement avec faute f_2 : la vanne A_1 restant bloquée en ouverture après que le niveau L_1 soit atteint, l'ouverture de la vanne A_2 provoque un remplissage plus rapide de la cuve : la règle r_2 décrit cette dynamique.

2.3 Construction de l'automate comportemental

Nous cherchons à définir par un automate temporisé communicant $\mathcal{P}$ le comportement dynamique d'un procédé décrit par son contrôleur $\mathcal{C}$, ses n actionneur $\mathcal{A}_1, \ldots, \mathcal{A}_n$, ses m capteurs $\mathcal{S}_1, \ldots, \mathcal{S}_m$, et ses k règles de causalités $r_1, \ldots, r_k$.

La construction de l'automate comportemental du procédé $\mathcal{P}$, se décompose en deux étapes. La première étape construit une esquisse de cet automate : son comportement est celui recherché, mais n'est pas minimal en nombre d'états. Dans une deuxième étape, nous réduirons son nombre d'états. Le lecteur trouvera dans [96] les explications détaillées sur les algorithmes pour construire l'automate temporisé communicant du procédé.

2.3.1 Construction de l'esquisse de l'automate comportemental

Nous simulons la composition synchrone $\mathcal{C}\|\mathcal{A}_1\| \ldots \|\mathcal{A}_n\|\mathcal{S}_1\| \ldots \|\mathcal{S}_m$ pour obtenir un automate décrivant le comportement du procédé. Tous ces automates partagent les mêmes ensembles d'actions $Sync$, d'horloges Clk et de variables entières V. Nous explorons l'espace d'états atteignables par une analyse d'accessibilité. Lors de cette analyse d'accessibilité, l'automate du contrôleur se retrouvera bloqué sur certains états : il ne pourra pas prendre des transitions exprimant une garde sur l'état des capteurs. Pour « débloquer » ces transitions, nous utiliserons les règles de causalités : celles-ci nous permettront de simuler le comportement dynamique des capteurs, faisant ainsi évoluer l'état des capteurs, ce qui permettra en fin de compte de satisfaire les gardes bloquant le contrôleur.

L'algorithme III.1 décrit la première partie du calcul de l'automate temporisé communicant $\mathcal{P}$, son esquisse.

La figure III.2 présente l'esquisse de l'automate comportemental du procédé batch obtenu par l'algorithme III.1.

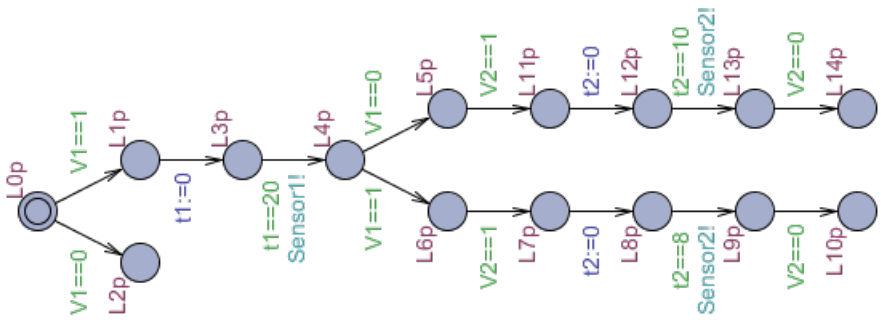


FIGURE III.2 – Esquisse de l'automate comportemental

2.3.2 Construction de l'automate comportemental du procédé

La version finalisée de l'automate comportemental se fait à partir de son esquisse. Nous appliquons des règles de simplification qui permettent de diminuer le nombre d'états de

Algorithme III.1 Construction de l'esquisse de l'automate comportemental du procédé

procedure BUILDDRAFTBEHAVIOURAUTOMATON

Input: $\mathcal{C}, \mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{S}_1, \ldots, \mathcal{S}_m, r_1, \ldots, r_k$

Output: draft plant $\mathcal{P} = (L^p, \ell_0^p, Sync, Clk, V, E^p, Init, I^p)$

Initialization : $\ell_0^p = (\ell_0^c, \ell_0^{a1}, \ldots, \ell_0^{an}, \ell_0^{s1}, \ldots, \ell_0^{sm})$; $L^p = \{\ell_0^p\}$; $E^p = \emptyset$; $Waiting = \{\ell_0^p\}$;
while $Waiting \neq \emptyset$ **do**
  $\ell^p = Waiting.pop()$;
  $\{\ell^p = (\ell^c, \ell^{a1}, \ldots, \ell^{an}, \ell^{s1}, \ldots, \ell^{sm})\}$
  $lst\_trans_{Ctrl} = Available\_Transitions\_Controller(\ell^p)$;
  {compute the available transitions directed by the controller}
  if $lst\_trans_{Ctrl} \neq \emptyset$ **then** {we use a transition directed by the controller}
    for each transition $e \in lst\_trans_{Ctrl}$ **do**
      if $e.destination \notin L^p$ **then**
        $Waiting.push(e.destination)$;
        $L^p = L^p \cup \{e.destination\}$;
      end if
      if $e.sync = \emptyset$ **then**
        {case $\ell_u^c \xrightarrow{g} \ell_v^c$ where g is a Boolean expression satisfied on the status of the sensors}
        $E^p = E^p \cup \{\ell^p \rightarrow e.destination\}$;
      else
        {case $\ell_u^c \xrightarrow{actuator_i!} \ell_v^c$ synchronized with $\ell_u^{ai} \xrightarrow[v_i:=value]{actuator_i?} \ell_v^{ai}$}
        $E^p = E^p \cup \{\ell^p \xrightarrow{v_i==value} e.destination\}$;
      end if
    end for
  else{there is no available transitions from ℓ^p, the controller is blocked at state ℓ_u^c by a transition $\ell_u^c \xrightarrow{g} \ell_v^c$ with a guard g unsatisfied}
    $lst\_sensors = SensorsInvolve(g)$
    {$lst\_sensors(g)$ contains the name of the sensor whose variable l_j is used in guard g}
    if $\exists r \in AvailableRule(lst\_sensors)$ **then**
      {there exists an available rule r : its premise is satisfied and its conclusion involves a sensor which is present in $lst\_sensors$}
      Create new state $\ell'^p = (\ell_v^c, \ell^{a1}, \ldots, \ell^{an}, \ell^{s1}, \ldots, \ell^{sm})$
      {In ℓ'^p, the location of the controller is updated to the destination of the unsatisfied controller's transition}
      $L^p = L^p \cup \{\ell'^p\}$;
      $E^p = E^p \cup \{\ell^p \xrightarrow[clk_i:=0]{} \ell'^p\}$;
      {$clk_i := 0$ is the reset of the clock used in rule r}
      Create new state $\ell''^p = (\ell_v^c, \ell^{a1}, \ldots, \ell^{an}, \ell^{s1}, \ldots, \ell_v^{sj}, \ldots, \ell^{sm})$
      {in ℓ''^p, the location of the sensor S_j used in rule r is updated}
      $L^p = L^p \cup \{\ell''^p\}$;
      $E^p = E^p \cup \{\ell'^p \xrightarrow{clk_i==K, Sensor_i!} \ell''^p\}$;
      $Waiting.push(\ell''^p)$;
    end if
  end if
end while

l'automate sans changer son comportement. L'algorithme III.2 décrit l'application de ces règles de simplification.

La figure III.3 présente l'automate comportemental du procédé batch obtenu par l'algorithme III.2. La 4ième règle de simplification permet d'éliminer les états l_2^p, l_{10}^p et l_{14}^p de l'automate de la figure III.2.

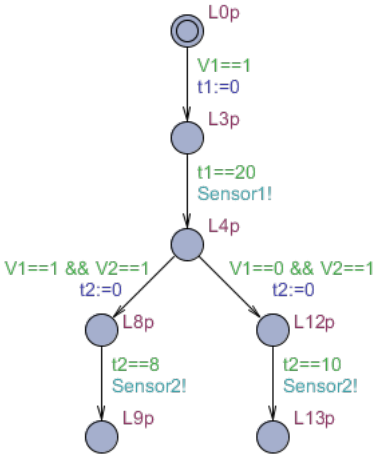


FIGURE III.3 – Automate comportemental

3 Conclusion et perspectives sur la construction de l'automate comportemental

La majorité des méthodes de diagnostic des systèmes à événements discrets requièrent d'avoir la description complète du comportement du système dynamique : son comportement nominal et son comportement en présence de défaillances. Cependant, les systèmes étant de plus en plus complexes, la modélisation par systèmes à événements discrets est de plus en plus difficile à obtenir. Nous avons proposé une méthode d'aide à la construction du système à événements discrets du comportement d'un système manufacturier en utilisant un ensemble de règles de causalités décrivant la dynamique du procédé : les interactions entre les actionneurs et les capteurs. Nous avons défini des algorithmes qui permettent de construire automatiquement un automate temporisé communiquant qui décrit l'ensemble des comportements possibles du système manufacturier en présence ou non de fautes sur les actionneurs.

Les perspectives de ce travail sont nombreuses : tout d'abord l'extension de notre approche aux défaillances sur les capteurs, puis la prise en compte du parallélisme dans les systèmes de commande décrits par des GRAFCET et enfin l'intégration dans nos modèles des incertitudes sur la durée des tâches qui peuvent apparaître dans certains procédés manufacturiers.

Algorithme III.2 Construction de l'automate comportemental du procédé

procedure ConstructionAutomateComportemental

1. Suppression des transitions sans étiquettes : la ligne 15 de l'algorithme III.1 crée des transitions sans étiquettes de la forme $\ell_u^p \rightarrow \ell_v^p$, nous fusionnons alors les états l_u^p et l_v^p en un seul état l_v^p.
2. Fusion de transitions étiquetées par des gardes sur l'état des actionneurs : La ligne 18 de l'Algorithme III.1 crée des transitions étiquetées par une garde de la forme $v_i == value$. Lorsque l'automate $\mathcal{P}$ comporte plusieurs transitions consécutives de ce type, nous définissons dans l'automate $\mathcal{P}$ une seule transition dont l'étiquette est une garde formée par la conjonction de toutes ces gardes.
3. Fusion de transitions étiquetées par des remises à zéro d'horloges : La ligne 29 de l'Algorithme III.1 crée des transitions de la forme $\ell_v^p \xrightarrow[clk_i:=0]{} \ell_w^p$. Celles-ci sont précédées par des transitions de la forme $\ell_u^p \xrightarrow{g} \ell_v^p$ où g est une garde sur l'état des actionneurs. Nous fusionnons ces deux transitions en une seule : $\ell_u^p \xrightarrow[clk_i:=0]{g} \ell_w^p$.
4. Elimination des états ne conduisant pas à une action de synchronisation sur un capteur : L'automate comportemental doit refléter par ses transitions les conditions qui mènent au déclenchement des capteurs. Nous pouvons donc éliminer dans l'esquisse de l'automate les états à partir desquels il n'y a aucune action de synchronisation sur les capteurs.

Chapitre IV

Modélisation et diagnostic de systèmes dynamiques hybrides

1 Présentation générale : contexte, état de l'art, positionnement, originalité

1.1 Contexte

Dans ce chapitre, nous nous intéressons au diagnostic de systèmes complexes dont la modélisation de la dynamique exprime à la fois une dynamique continue et une dynamique discrète, ce sont les systèmes dynamiques hybrides (SDH) [97], [98]. Ce mode de fonctionnement dynamique hybride se trouve par exemple dans les systèmes automatisés de production comme les procédés batch. Le procédé de fabrication est décomposé en étapes ou phases qui sont modélisées par des états d'un système à événements discrets. A chaque étape correspond une instrumentation particulière du système, le système est alors caractérisé dans l'état correspondant par une dynamique continue qui lui est propre.

Un système dynamique hybride peut ainsi être décrit comme la combinaison d'un système à événements discrets, de systèmes dynamiques continus et d'une interface qui gère les interactions entre les dynamiques continues et discrètes [99]. La figure IV.1 illustre la structure d'un système dynamique hybride.

Le système hybride est alors caractérisé par un ensemble de variables continues et discrètes. Les variables continues (variables internes, variables externes d'entrées et sorties du système) permettent de représenter le comportement d'un phénomène physique, par exemple une température, une position, un débit, une tension, un niveau de remplissage d'une cuve, etc. ou des horloges pour décrire des mécanismes de temporisation. Les variables discrètes permettent de représenter les états des actionneurs et des capteurs et l'état discret dans lequel le système se trouve. A chaque état de l'automate hybride sont associés des invariants sur les variables du système. Un invariant décrit des contraintes que doivent satisfaire des variables pour que le système reste dans ce mode.

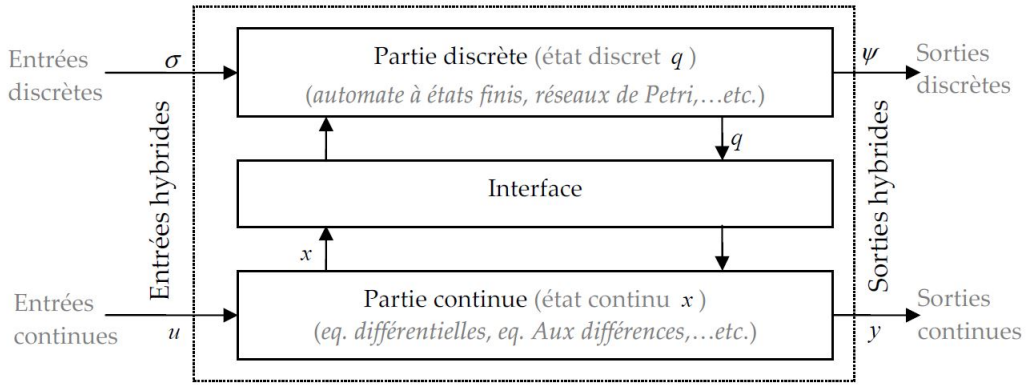


FIGURE IV.1 – Structure d'un système dynamique hybride [100]

L'automate change d'état par le franchissement d'une transition lorsque certaines conditions appelées gardes sont vérifiées. Le changement d'état peut être provoqué par un événement extérieur au système modélisé, par exemple par une commande du contrôleur qui provoque un changement de valeur sur une variable discrète, ou par un événement intérieur à l'état de l'automate, par exemple une variable continue ne satisfait plus une contrainte d'appartenance à un intervalle de valeurs ou une horloge a atteint une certaine temporisation.

Le système dynamique hybride peut être soumis à des perturbations (bruits, entrées inconnues) et à des défauts modifiant le comportement nominal des actionneurs, des capteurs et des processus physiques. On peut alors associer aux états de l'automate hybride des modes de fonctionnement. Ceux-ci sont partitionnés en modes nominaux, dégradés et défaillants. Le mode dégradé représente un état du SDH où il y a une baisse des performances du système causée par une perturbation. Des variables du système ne sont plus alors dans le domaine des valeurs nominales, elles sont dans une marge de tolérance. Le système peut basculer d'un mode nominal à un mode dégradé lorsqu'une perturbation est présente et réciproquement lorsque la perturbation disparait ou que l'asservissement du système ramène le système à un mode nominal. Lorsque le système est soumis à un défaut, les performances du système sont en dehors des marges de tolérance, le système bascule d'un mode nominal ou dégradé à un mode défaillant. Dans le cas de défaillances permanentes, le système ne peut plus quitter le mode défaillant dans lequel il se trouve.

Pour mettre en œuvre des méthodes de diagnostic pour un système dynamique hybride, nous devons tout d'abord disposer d'un modèle de celui-ci. Différentes approches ont été étudiées pour modéliser les SDH :

a) Extension des approches discrètes : basée sur un système à événements discrets, une approximation de la dynamique continue y est intégrée par l'utilisation d'événements qui caractérisent la dynamique continue. Dans cette approche, on trouve les formalismes des automates temporisés [101], des extensions des réseaux de Petri tels que RdP temporisé, RdP temporel, RdP continu [102].

b) Extension des approches continues : Dans ces approches, on trouve par exemple une modélisation par équations différentielles pour la dynamique continue et par équations aux différences pour la dynamique discrète. Il y a également la modélisation par bond graphs commutés [103].

c) Approche mixte : Dans les extensions des approches discrètes ou continues, l'interaction entre la partie continue et événementielle n'est pas représentée explicitement. L'approche mixte permet l'interaction entre des sous-modèles discrets ou continus. Dans cette approche mixte, on trouve les automates hybrides [104], [105], les statecharts hybrides [106], les réseaux de Petri hybrides [107] et les bond graphs hybrides [108].

1.2 Etat de l'art, positionnement, originalité

Dans cette section, nous nous interressons aux méthodes de diagnostic à base de modèles pour les systèmes dynamiques hybrides. Le lecteur intéressé par des états de l'art plus complet sur le diagnostic de systèmes dynamiques hybrides peut se référer à [100], [109], [110]. Les travaux relatifs au diagnostic de SDH par une approche à base de modèles peuvent être classés en trois catégories : les méthodes issues des systèmes continus, les méthodes issues des systèmes à événements discrets et les méthodes mixtes.

Méthodes issues des systèmes continus

Les méthodes de diagnostic pour les systèmes continus reposent sur la génération et l'analyse de résidus. Un résidu est un indicateur de défaut, il représente l'écart entre la sortie réelle du système et sa sortie estimée par le modèle du système. Trois approches sont principalement utilisées pour générer des résidus : les approches d'identification ou estimation paramétrique, les approches à base de relations de redondance analytique ou espace de parité et les approches à base d'observateurs ou de filtres.

La détection d'une défaillance est obtenue par la surveillance des résidus générés. Lorsqu'un ou des résidus ont des valeurs différentes de zéros, un écart est détecté entre le modèle et le système. La première problématique de l'approche de diagnostic à base de résidus est la qualité des résidus utilisés : ils doivent être sensibles aux défaillances, mais insensibles aux perturbations. La seconde problématique du diagnostic à base de résidus est de lever les ambiguïtés de localisation des défauts : il est alors nécessaire de construire un ensemble de résidus qui permettent d'identifier individuellement chacun des défauts pouvant affecter le système, on construit alors des résidus structurés [111] ou directionnels [112]. Des exemples de travaux pour le diagnostic de SDH en utilisant une approche à base de résidus peuvent être consultés dans [113]-[119].

Les limitations de cette approche sont d'une part qu'on doit disposer du mode de fonctionnement avant et après l'occurrence d'un défaut pour savoir quels sont les résidus à utiliser, en effet à chaque mode du système correspond un ensemble de résidus. D'autre part, les méthodes de constructions des résidus sont limitées à des systèmes ayant un modèle mathématique linéaire ou un modèle non linéaire très spécifique.

Méthodes issues des systèmes à événements discrets

Les méthodes de diagnostic de SDH qui sont basées sur l'utilisation de systèmes à événements discrets font appel à une abstraction des dynamiques continues et les représentent sous forme d'états ou de gardes sur les transitions d'états. Nous citons dans ce paragraphe quelques travaux dans cette thématique à base d'extension d'automates ou de réseaux de Petri.

Dans [120], [121], les auteurs proposent une abstraction des dynamiques continues basée sur un raisonnement temporel et événementiel et utilisent les réseaux de Petri temporels flous. Dans [122] les auteurs étendent l'approche du diagnostiqueur de Sampath et al. aux systèmes dynamiques hybrides en utilisant un modèle d'automate hybride à temps discret, l'évolution des variables continues est abstraite par discrétisation du temps, il y a une synchronisation sur l'occurrence d'un événement spécial, appelé *tick*, qui se produit périodiquement. Lunze propose également une adaptation de l'approche de Sampath et al. pour les systèmes continus contrôlés discrètement [123], [124]. Une abstraction qualitative des variables continues du système est effectuée à travers l'utilisation de quantificateurs. Dans [125], le système dynamique hybride est décrit comme un réseau de Petri temporisé, l'abstraction de la dynamique continue correspond alors à l'évolution temporelle des événements discrets du système.

Méthodes de diagnostic mixtes

Les méthodes issues des systèmes continus ou des SED privilégient l'une des deux dynamiques du système dynamique discret au détriment de l'autre. Les méthodes de diagnostic mixtes des systèmes dynamiques hybrides tiennent compte des deux dynamiques, continues et discrètes, à la fois. Nous relatons dans ce paragraphe quelques travaux de recherche dans ce domaine de recherche.

Karsai et al. dans [126] modélise le SDH par un bond graph hybride, le diagnostic est obtenu par la combinaison d'un observateur hybride et de relations causales et temporelles entre les différents types de défauts et les observations associées. Dans [127], les auteurs utilisent un automate hybride et proposent une méthode de sélection de générateurs de résidus pour réaliser le diagnostic.

Derbel et al. dans [128] propose une méthodologie de diagnostic des SDH avec une sous-classe des automates hybrides : les automates hybrides rectangulaires. Les auteurs proposent la construction d'un diagnostiqueur en ligne qui estime l'état courant du système et permet l'identification de défauts.

Dans [129] Bayoudh et al. proposent une méthodologie de diagnostic actif des systèmes dynamiques hybrides. Le diagnostic actif est un processus qui consiste à agir sur le système dans le but d'affiner le diagnostic. Ils proposent dans [130] de combiner les techniques de diagnostic issues des domaines continus et discrets. Dans leur approche, la partie continue du système est diagnostiquée en utilisant l'approche d'espace de parité.

Olivier-Maget et al. dans [131], [132] propose une méthode de diagnostic pour les procédés chimiques qui sont des systèmes dynamiques hybrides. Leur méthode combine les réseaux de Petri différentiel objet [133] pour la modélisation du système avec l'utilisation

de filtres de Kalman étendus pour l'estimation de l'état du système réel.

Belkhiat et al. proposent dans [134] une méthode de diagnostic à base de modèle d'une classe de systèmes linéaires à commutations. Leur approche combine les outils initialement dédiés au diagnostic des systèmes continus et d'autres spécifiques aux systèmes événements discrets et utilise trois modules : deux types de générateurs de résidus et un estimateur en ligne de l'état discret.

Louajri et al. [135] proposent également une approche mixte pour le diagnostic de SDH qui est basée sur un diagnostiqueur composée de trois parties : un diagnostiqueur discret, un diagnostiqueur continu et un coordinateur. Le diagnostiqueur discret est modélisé par un automate hybride à temps discret afin de détecter et d'isoler les défauts discrets. Le diagnostiqueur continu est basé sur un ensemble de résidus afin de diagnostiquer les défauts paramétriques. Le coordinateur combine les décisions discrètes et continues des deux diagnostiqueurs pour diagnostiquer les défauts nécessitant l'interaction des deux diagnostiqueurs.

Une méthode de diagnostic des systèmes dynamiques hybrides linéaires modélisés par des automates hybrides est proposée par Asma et al. [136]. Leur méthode propose d'utiliser deux types observateurs : le premier est un observateur discret qui est utilisé pour estimer l'état discret du système et détecter les défauts discrets. Le deuxième est un observateur continu qui est utilisé pour générer des résidus structurés et isoler des défauts de capteurs.

Rahal et al. [137] propose une méthodologie de détection et de localisation de fautes robuste aux incertitudes paramétriques pour les systèmes hybrides à base de modèle bond graph hybride et de relations de redondances analytiques globales et développe l'outil de bond graph hybride de diagnostic pour la modélisation, mais aussi la surveillance en ligne.

1.3 Positionnement, originalité

Nous nous intéressons au diagnostic de défauts brusques et permanents sur les actionneurs, les capteurs et les processus comme l'illustre la figure IV.2. Nos études ont porté principalement sur le diagnostic de défauts uniques. Notre axe de recherche est le développement de méthodes de diagnostic qui prennent en compte les aspects temporels pour pouvoir lever des ambiguïtés de localisation de défauts lorsqu'il existe une même signature pour des défauts différents.

Nos travaux sur le diagnostic de systèmes dynamiques hybrides ont été réalisés lors d'une collaboration internationale avec le laboratoire tunisien LARATSI pendant les thèses de Bochra Maaref et Lobna Belkacem ainsi que lors de l'encadrement de nombreux stages de Master 2 d'étudiants de l'ENIM de Monastir et de l'ISSAT de Kairouan.

Dans la thèse de Bochra Maaref [138], nous avons développé une méthodologie de diagnostic de systèmes dynamiques hybrides par une approche mixte combinant le bond graph, l'observateur et l'automate temporisé. La méthode de synthèse de l'observateur à partir de la description du système par un bond graph permet une robustesse par rapport aux entrées inconnues (perturbations) et fournit une sensibilité à l'apparition de défauts. La détection est obtenue par l'analyse des résidus. La localisation des défauts est réalisée

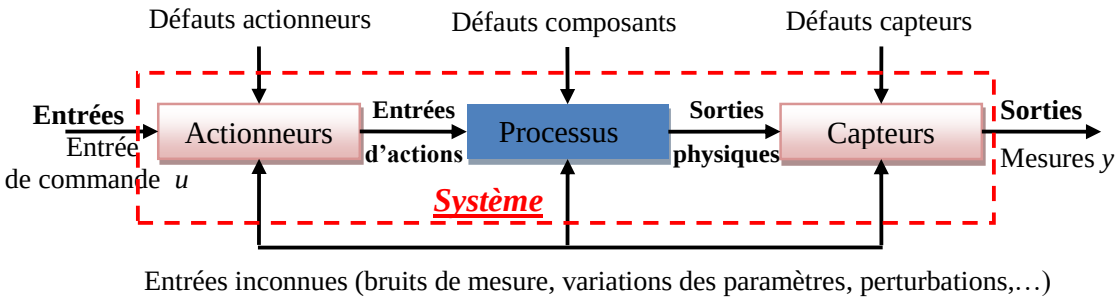


FIGURE IV.2 – Système dynamique hybride avec différents défauts [138]

par l'analyse des signatures des défauts (matrice de signature) et s'il y a ambiguïté de localisation, une analyse temporelle est réalisée par un automate temporisé.

Dans la thèse de Lobna Belkacem [139], nous avons également développé une méthodologie de diagnostic de systèmes dynamiques hybrides. La méthode de diagnostic est issue des systèmes à événements discrets, elle est basée sur la construction d'un automate hybride qui représente le comportement global du système, tous les modes y sont décrits (nominal, dégradé et défaillant) en leur associant des limites de temps de séjour. A partir du modèle sans fautes du système sous la forme d'un automate hybride et d'une analyse AMDEC, le modèle est enrichi par l'ajout des modes dégradés et défaillants. La simulation du modèle avec ou sans fautes permet l'identification temporelle des différents modes de fonctionnement du système. Par le suivi de l'état du système dynamique hybride, cet automate hybride permet de déterminer le mode courant du système et s'il y a eu défaillance de connaitre la trajectoire menant à un mode défaillant ainsi que l'ensemble des défauts ayant eu lieu.

2 Synthèse des travaux développés

2.1 Diagnostic de systèmes dynamiques hybrides combinant bond graph, observateur et automate temporisé

L'approche de diagnostic de système dynamique hybride que nous proposons est basée sur l'utilisation d'un bond graph (graphe à liens ou graphe de liaisons) [140], [141] pour la modélisation du SDH, d'un observateur pour estimer la sortie du système et générer des résidus afin de détecter des défauts permanents et les localiser si ceux qui ont une signature de défaut unique. Si plusieurs défauts ont la même signature, un automate temporisé sera utilisé pour localiser les défauts. Notre approche de diagnostic est robuste aux perturbations, l'observateur généré à partir du bond graph est insensible aux entrées inconnues (bruits, perturbations), mais sensible aux défauts.

2.1.1 Méthodologie de diagnostic

Nos travaux de recherche s'inscrivent donc dans les approches de diagnostic mixtes. Les différentes étapes de notre méthodologie sont décrites sur la figure IV.3.

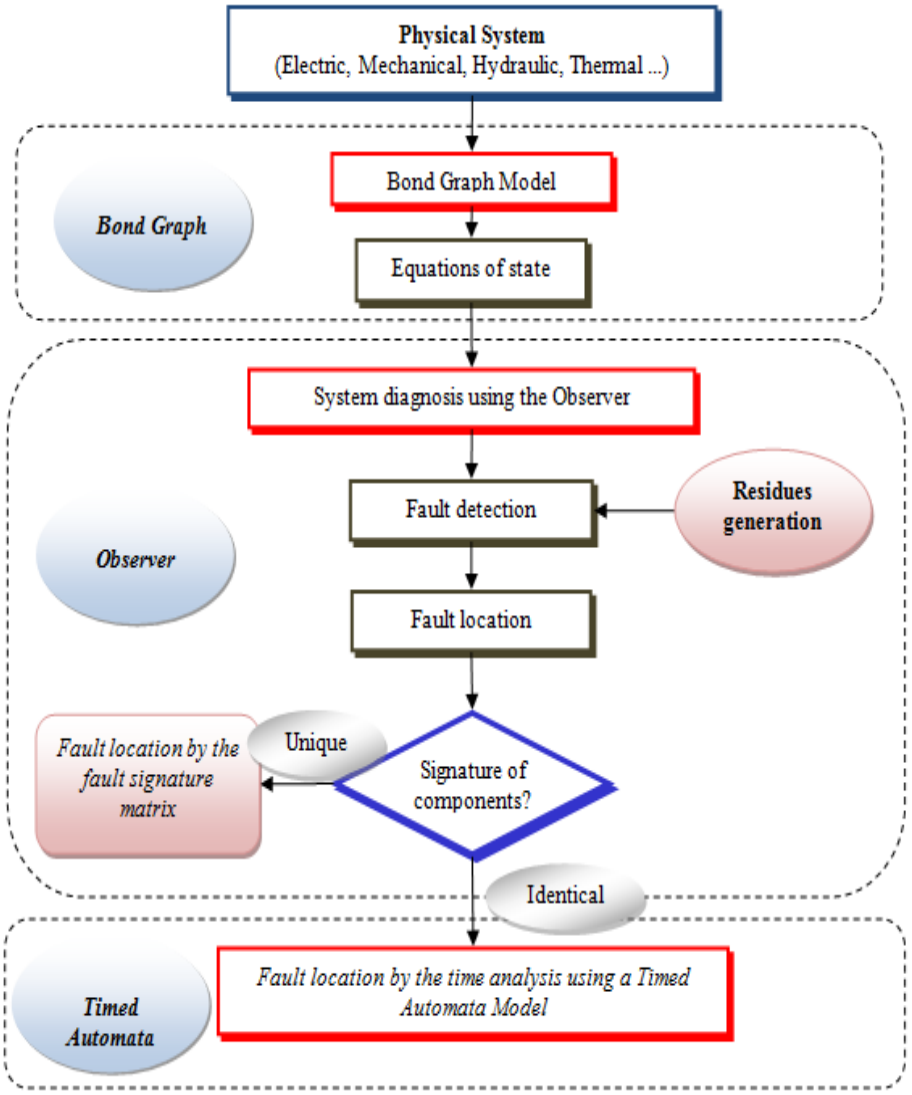


FIGURE IV.3 – Diagnostic de SDH avec Bond graph, Observateur et Automate temporisé [142]

La mise en oeuvre de notre méthode de diagnostic est la suivante :

1) A partir de la description bond graph, nous générons les équations d'états du système.
2) Nous définissions un observateur du système à partir de sa représentation sous forme d'équations d'états à laquelle nous avons intégré des entrées inconnues. Nous obtenons ainsi le vecteur $\tilde{y}(t)$ des sorties estimées du système.

3) A partir d'une analyse fonctionnelle par un diagramme FAST (Function Analysis System Technique) et d'une analyse des défauts par une AMDEC (Analyse des modes de défaillance, de leurs effets et de leur criticité), nous construisons le modèle global (comportement normal et défectueux) du système. Nous modélisons le système global sous Matlab / Simulink / Stateflow.
4) Nous définissons le vecteur $r(t)$ des résidus comme l'écart entre la sortie réelle $y(t)$ et la sortie estimée $\tilde{y}(t)$ du système, $r(t) = y(t) - \tilde{y}(t)$.
5) Nous simulons le modèle du système sous Matlab / Simulink / Stateflow avec l'injection de chacun des défauts identifiés par l'AMDEC et identifions la valeur des résidus en présence de chacun des défauts séparément. A partir des valeurs des résidus en présence des défauts, nous construisons la matrice de signatures des défauts.

Si dans la matrice de signatures des défauts il existe plusieurs défauts ayant la même signature, pour aider à lever les ambiguïtés de localisation du défaut, nous réalisons une analyse temporelle à l'aide d'un automate temporisé. Les étapes suivantes détaillent la construction de l'automate temporisé, le diagnostiqueur.

6) Nous simulons le modèle du système sous Matlab / Simulink / Stateflow en fonctionnement normal (sans défaillance) pour identifier les temporisations de chacun des modes/états nominaux : durées pendant lesquelles le système se trouve dans chacun des états nominaux. Nous identifions les durées T_{min} et T_{max} qui décrivent respectivement le temps minimum nécessaire et le temps maximum toléré pour l'exécution correcte de l'état.
7) Nous simulons le modèle du système sous Matlab / Simulink / Stateflow avec l'injection de chacun des défauts identifiés par l'AMDEC : nous identifions T_c le temps critique au-delà duquel il n'y a aucun doute de la présence d'une défaillance.
8) A partir de la séquence de commande du système et des données T_{min}, T_{max} et T_c pour chaque étape du fonctionnement du système nous construisons un diagnostiqueur sous la forme d'un automate temporisé qui permet la détection et la localisation des défauts

La mise en œuvre du diagnostic est réalisée par les étapes suivantes :

9) Nous utilisons l'observateur pour la surveillance du système. Si pendant le fonctionnement du système les valeurs de un ou plusieurs résidus diffèrent de zéro, une défaillance est détectée. Si la signature des résidus est unique dans la matrice de signatures des défauts, le défaut est alors identifié.
10) Si la signature des résidus obtenus lors de la surveillance apparait plusieurs fois dans la matrice de signatures, il y a alors une ambiguïté d'identification du défaut. Pour résoudre cette ambiguïté, nous utilisons l'automate temporisé qui nous permet de suivre l'évolution des modes de fonctionnement du système par analyse temporelle et d'identifier à quelle étape de la séquence de commande le système se trouve au moment de la détection d'un défaut par l'observateur et pour quel défaut le système se trouve dans un mode défaillant.

2.1.2 Exemple illustratif de notre méthodologie de diagnostic de SDH à partir d'une description par bond graph

Nous présentons dans ce paragraphe sur un exemple académique les principales étapes de notre méthodologie de diagnostic de systèmes dynamiques hybrides. Cet exemple est un système hydraulique à deux réservoirs illustré par la figure IV.4

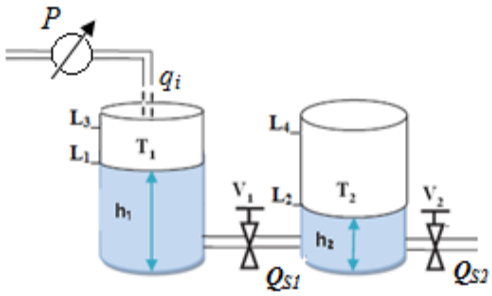


FIGURE IV.4 – Exemple d'un système hydraulique à deux réservoirs [142]

Le système se compose de deux réservoirs cylindriques : le premier réservoir T_1 de section S_1 ($S_1 = 0.0154\ m^2$) et de hauteur h_1 et le second réservoir T_2, de section S_2 ($S_2 = 0.0154\ m^2$) et de hauteur h_2. Les réservoirs communiquent par l'intermédiaire une vanne V_1 (de résistance hydraulique RV_1) avec un débit QS_1, toujours ouverte. Le système comporte une seule entrée : débit volumique q_i ($q_i = 10^{-4}\ m^3/s$). Le débit de sortie de second réservoir QS_2 est autorisé par une vanne V_2 de résistance hydraulique RV_2. Le système dispose de quatre capteurs de valeurs 0 ou 1 : deux capteurs de niveau L_1 et L_2 et deux capteurs de débordement L_3 et L_4.

L'objectif global du système est de maintenir un niveau de liquide dans les deux réservoirs à un niveau bien défini :

$$\begin{cases} h_1 \leq 0.15m \\ h_2 \leq 0.10m \end{cases} \text{ avec } h_1 \geq 0.15m \Rightarrow L_1 = 1 \text{ et } h_2 \geq 0.10m \Rightarrow L_2 = 1.$$

Le modèle bond graph représentant ce système est donné par la figure IV.5.

Les équations d'état du système peuvent être déduites directement du modèle bond graph, figure IV.6. Les niveaux dans les deux réservoirs h_1 et h_2 représentent les variables d'états $x = [h_1 h_2]$, l'entrée u est le débit entrant : $u = [q_i]$ et les sorties y sont les mesures des capteurs L_1 et L_2 : $y = [L_1 L_2]$.

Après la linéarisation autour du régime de fonctionnement stationnaire, la partie continue du système peut être modélisée par un système d'équations différentielles à partir desquelles on peut construire un observateur, figure IV.7.

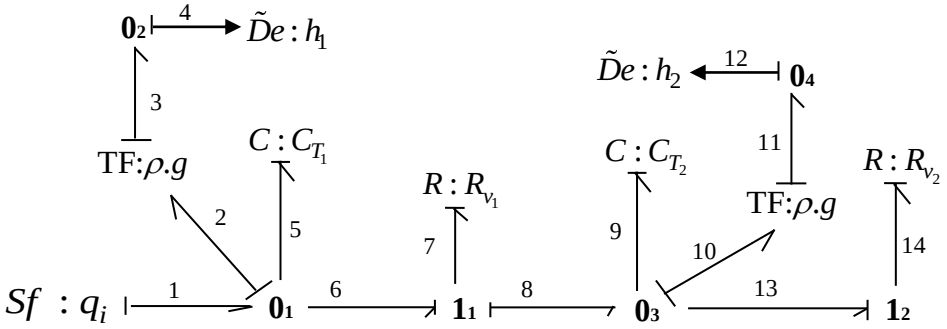


FIGURE IV.5 – Modèle Bond Graph du système hydraulique à deux réservoirs [142]

$$\begin{cases} S_1 \dfrac{dL_1}{dt} - q_i + \dfrac{\rho g (L_1 - L_2)}{R_{v1}} = 0 \\ S_2 \dfrac{dL_2}{dt} - \dfrac{\rho g (L_1 - L_2)}{R_{v1}} + \dfrac{\rho g L_2}{R_{v2}} = 0 \end{cases}$$

State representation

$$\begin{cases} \dfrac{dh_1}{dt} = \dfrac{q_i}{S_1} - \dfrac{(h_1 - h_2)}{S_1 R_1}; \quad R_1 = \dfrac{R_{v1}}{\rho g} \text{ and } L_1 = h_1 \\ \dfrac{dh_2}{dt} = \dfrac{h_1 - h_2}{S_2 R_1} - \dfrac{h_2}{S_2 R_2}; \quad R_2 = \dfrac{R_{v2}}{\rho g} \text{ and } L_2 = h_2 \end{cases}$$

FIGURE IV.6 – Représentation d'état du système hydraulique à deux réservoirs [142]

$$\begin{cases} \dot{x}(t) = A.x(t) + B.u(t) \\ avec\ u(t) = -K.x(t) + v(t) \\ y(t) = C.x(t) \end{cases}$$

with

$$A = \begin{pmatrix} \dfrac{-R_1}{S_1} & \dfrac{R_1}{S_1} \\ \dfrac{R_1}{S_2} & \dfrac{-(R_1 + R_2)}{S_2} \end{pmatrix}; \ B = \begin{pmatrix} \dfrac{1}{S_1} \\ 0 \end{pmatrix}; \ C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}; K = [-4.7 \ 6.06]$$

and R_1=0.5; R_2=1

FIGURE IV.7 – Définition de l'observateur du système hydraulique à deux réservoirs [142]

A partir de l'observateur, nous définissons le vecteur de résidus $r(t) = y(t) - \tilde{y}(t)$.

Nous définissons sous Matlab / Simulink / Stateflow un modèle global du système et nous le simulons avec l'injection de défauts sur les vannes V_1 et V_2 et les capteurs L_1 et L_2, nous construisons la matrice de signatures des défauts illustrée par la table 1.

TABLE 1 – Matrice de signatures des défauts du système hydraulique à deux réservoirs

Failure mode	Components	r_1	r_2
$V_1\_Stuck\_Close$	Valve V_1	1	0
$V_2\_Stuck\_Close$	Valve V_2	1	1
$V_2\_Stuck\_Open$			
$L_1\_Stuck\_Down$	Level sensor L_1	1	0
$L_1\_Stuck\_Up$			
$L_2\_Stuck\_Down$	Level sensor L_2	0	1
$L_2\_Stuck\_Up$			

En analysant cette matrice, on voit que les signatures de la vanne V_2 et du capteur de niveau L_2 sont uniques, ce qui signifie que les défaillances sur ces composants sont localisables. Par contre, les signatures de la vanne V_1 et du capteur de niveau L_1 sont identiques, ce qui signifie que les pannes affectant ces composants ne peuvent pas être isolées. Pour résoudre ce problème d'ambiguïté, nous utilisons un outil supplémentaire, l'automate temporisé afin de différencier les pannes sur ces deux composants.

Par simulation sans fautes et avec fautes du système nous identifions les temporisations de chaque mode de fonctionnement du système, la séquence de commande du système est décrite par la figure IV.8 et l'automate temporisé qui permet de diagnostiquer les défauts est représenté sur la figure IV.9.

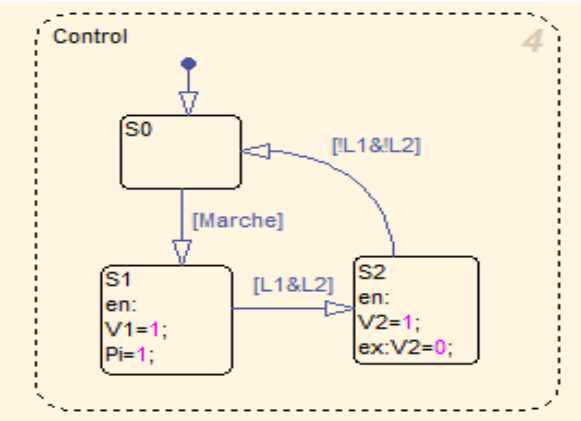


FIGURE IV.8 – Séquence de contrôle du système hydraulique à deux réservoirs [142]

Les deux méthodes de diagnostic (observateur et automate temporisé) fonctionnent en parallèle. En effet, la méthode basée sur l'observateur consiste à détecter et localiser un

défaut sur un système étudié à tout instant et sans délai. Et la méthode basée sur les automates temporisés permet de localiser les défauts qui ne peuvent pas être isolés par la première méthode (méthode basée sur l'observateur) et aussi d'identifier plus précisément les causes de défaillance, mais avec un certain retard.

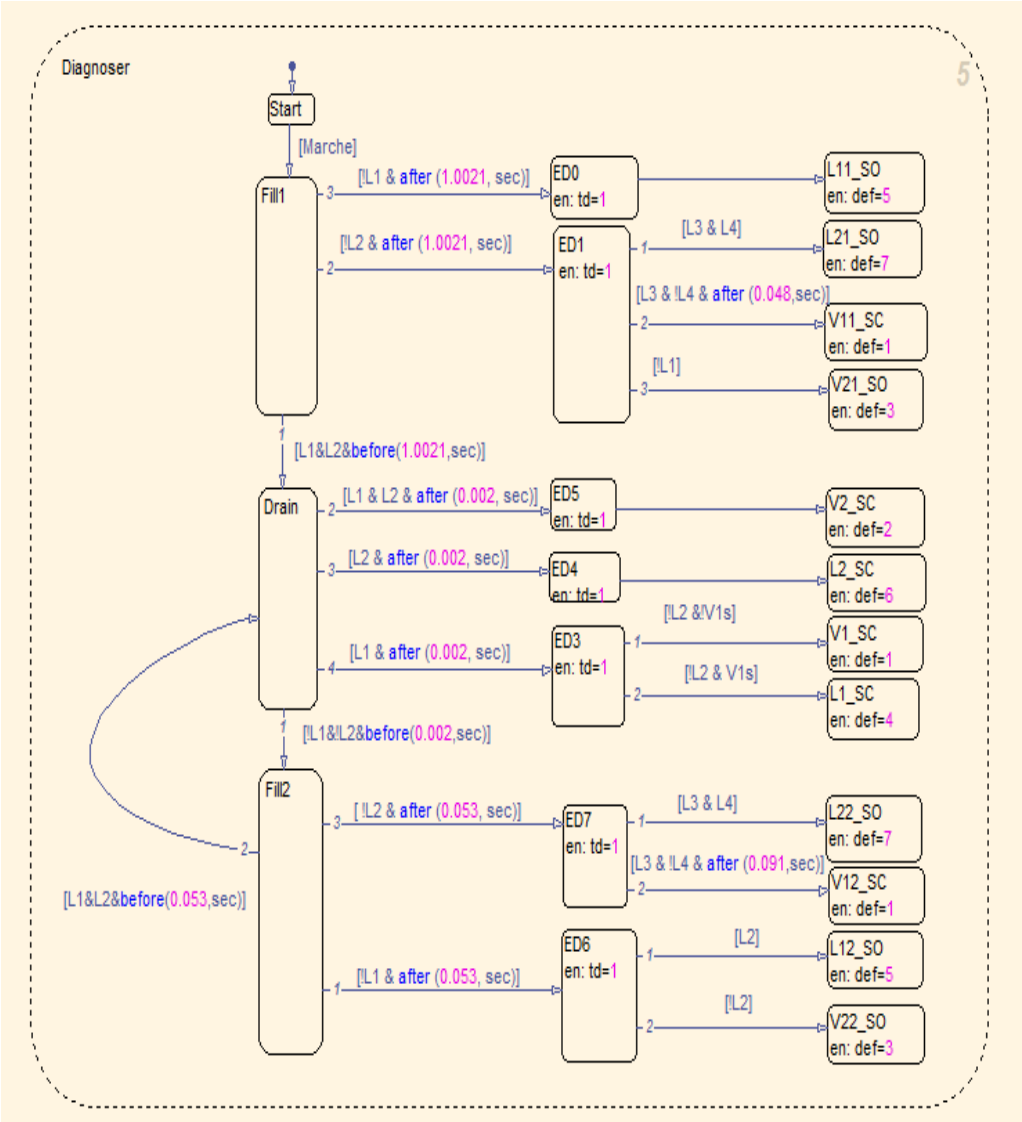


FIGURE IV.9 – Diagnostiqueur par automate temporisé du système hydraulique à deux réservoirs [142]

2.2 Diagnostic de systèmes dynamiques hybrides combinant automate hybride et automate temporisé

Dans cette deuxième approche de diagnostic d'un système dynamique hybride, nous modélisons le SDH par un automate hybride. A partir du modèle automate hybride, nous construisons un diagnostiqueur sous la forme d'un automate temporisé. Nous prenons en compte des défauts sur les actionneurs, capteurs et procédés. Nous considérons des défauts permanents et notre méthode s'applique pour la détection et la localisation d'un défaut unique.

Les automates hybrides [104], [105] sont une extension des automates à états finis. Ils intègrent deux dynamiques : la dynamique discrète modélisée par un automate à états finis et la dynamique continue modélisée par un système d'équations différentielles qui est associé à chaque état de l'automate.

Le modèle automate hybride est constitué par un ensemble de variables continues $X = \{x_1, x_2, \ldots\}$. Cet ensemble de variables est partitionné en variables d'entrée, variables de sorties et variables locales. Les variables d'entrée et de sortie représentent les entrées et sorties du système dynamique hybride.

A chaque état q d'un automate hybride, on associe à des variables locales x une équation différentielle de la forme $\dot{x}(t) = f(t)$ et un prédicat sur la valeur des variables continues appelé invariant. L'automate hybride peut également contenir des horloges pour restreindre la durée d'activation des états à une durée.

L'état actif d'un automate hybride peut changer suite au franchissement d'une transition entre deux états. A chaque transition peuvent être associées une condition de franchissement, appelée garde, et des affections sur les variables.

Un état particulier q_1 est l'état initial du système, une transition étiquetée $init(q1)$ décrit par un prédicat les conditions initiales du système.

L'évolution d'un automate hybride est caractérisée par une alternance de pas continus, où les variables continues et le temps progressent de façon continue, et de pas discrets où des transitions discrètes peuvent être franchies. La figure IV.10 présente un exemple d'automate hybride.

Nous supposons que nous disposons d'un automate hybride qui représente le modèle nominal du système dynamique hybride. Rappelons qu'à chaque état de l'automate hybride correspond une étape / tâche de fonctionnement du système industriel, c'est pour cela qu'à chaque état correspond une instrumentation particulière du système se caractérisant par une dynamique continue qui lui est propre. Ainsi à chaque état correspond un système d'équations différentielles particulier.

2.2.1 Méthodologie de diagnostic

Notre méthodologie de construction d'un module de surveillance, appelé diagnostiqueur, est la suivante :

1) Nous créons un modèle Matlab / Simulink / Stateflow du SDH à partir de son modèle

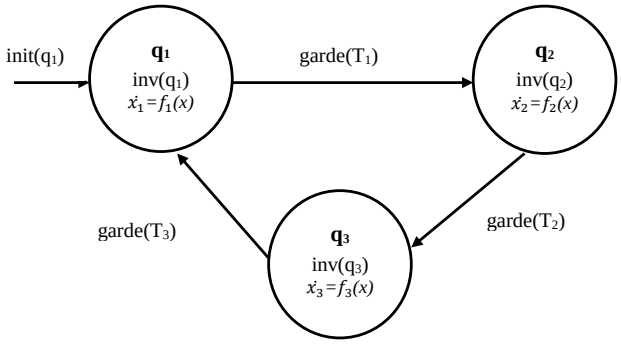


FIGURE IV.10 – Exemple d'un automate hybride [139]

automate hybride.

2) Nous simulons notre modèle et identifions pour chaque état q_i nominal la durée minimale $T_{min}^{N_i}$ et maximale $T_{max}^{N_i}$ d'exécution de cet état.

Nous notons par $I_n = [T_{min}^{N_i}, T_{max}^{N_i}]$ l'intervalle de temps nécessaire à la réalisation de la tâche associée à l'état q_i dans un mode nominal.

3) Nous procédons à une analyse fonctionnelle FAST et AMDEC pour identifier les différents modes de défaillances et identifions pour chaque défaut quelles variables continues il impacte. Pour chaque défaut, nous recensons les états dont la dynamique continue est impactée par celui-ci.

4) Nous transformons notre modèle Matlab / Simulink / Stateflow pour permettre l'injection aléatoire et contrôlée de ces défauts et nous réalisons un ensemble de simulation de notre système en injectant chaque défaut individuellement avant et pendant que l'automate est dans un état dont la dynamique est impactée par le défaut (nous les avons identifiés à l'étape 3 de notre méthodologie). Pour chaque état q_i nous identifions ainsi les durées critiques $T_c^{D_i}$ à partir desquelles le système tombe dans un mode défaillant.

5) Nous construisons alors l'automate hybride global qui contient les trois modes de fonctionnement nominal, dégradé et défaillant pour chaque état-étape du système. Le mode normal lorsque le fonctionnement du système est conforme aux exigences ; le mode de fonctionnement dégradé lorsqu'une dérive temporelle est signalée ou bien que le système réalise partiellement ses objectifs et le mode défaillant lorsqu'un défaut est présent. Pour distinguer les différents modes de fonctionnement du système, nous utilisons le temps d'activité d'un mode associé à un état. A partir des trois valeurs temporelles $T_{min}^{N_i}$, $T_{max}^{N_i}$ et $T_c^{D_i}$ associées à chaque état-étape du système, nous définissons deux nouveaux intervalles de fonctionnement J_m et F_m avec $J_n =]T_{max}^{N_i}, T_c^{D_i}[$ l'intervalle de temps du mode dégradé et $F_n =]T_c^{D_i}, +\infty[$ l'intervalle de temps du mode défaillant.

L'organigramme de construction du diagnostiqueur est présenté sur la figure IV.11. La figure IV.12 présente l'automate hybride global, le diagnostiqueur, obtenu par notre méthodologie.

La mise en œuvre de notre méthode de diagnostic est réalisée par l'étape suivante :

6) Nous réalisons le suivi de l'état du système réel à partir des capteurs et de la commande du système, nous utilisons le diagnostiqueur pour déterminer le mode courant du système et si il y a eu défaillance de connaitre la trajectoire menant à un mode défaillant ainsi que le défaut ayant eu lieu.

2.2.2 Exemple illustratif de notre méthodologie de diagnostic de SDH à partir d'une description par automate hybride

Nous illustrons notre méthodologie de construction d'un diagnostiqueur par automate hybride sur un exemple académique d'un système hydraulique schématisé par la figure IV.13.

Ce système est constitué de deux réservoirs R_1 et R_2 de hauteurs h_1 et h_2 respectivement, de section S, reliés par les conduites C_2 et C_3 équipées par les vannes V_2 et V_3 respectivement. Deux autres vannes V_1 et V_4 peuvent être utilisées pour évacuer le liquide par l'opérateur et de trois capteurs : deux capteurs de niveau Sh_1 et Sh_2, respectivement pour mesurer le niveau dans les réservoirs R_1 et R_2 et un capteur de débordement Dh_1. Le réservoir R_1 est alimenté par une pompe P de débit Q_p.

Les quatre vannes V_1, V_2, V_3 et V_4 sont commandées manuellement. Elles peuvent être ouvertes ou fermées par l'opérateur à tout moment. Nous considérons que le système est utilisé dans un mode de fonctionnement dans lequel les vannes V_1 et V_2 sont toujours ouvertes. Seules les vannes V_3 et V_4 sont contrôlées. Le comportement normal du système est alors représenté par 4 modes :

— Mode 1 : On remplit le réservoir R_1 jusqu'à ce que le niveau h_1 atteint 0.5.

— Mode 2 : le niveau h_1 atteint 0.5, les deux vannes V_2 et V_3 sont ouvertes. Le réservoir R_2 doit être rempli jusqu'à ce que le niveau h_2 atteigne 0.2.

— Mode 3 : A cette étape les deux niveaux h_1 et h_2 de réservoir dépassent les limites, on ouvre la vanne V_4.

— Mode 4 : le niveau h_1 dans le réservoir R_1 devient inférieur à 0.5, on vidange le réservoir R_2.

Chaque mode de fonctionnement du système est caractérisé par une instrumentation particulière sur les vannes V_3 et V_4. A partir de ces quatre modes, on définit l'automate hybride qui représente le système en fonctionnement normal. La figure IV.14 présente l'automate hybride de fonctionnement du système hydraulique. Les sommets Mode_1, Mode_2, Mode_3, Mode_4 représentent les états discrets du système où l'évolution continue a lieu.

Le modèle Stateflow de l'automate hybride est présenté sur la figure IV.15.

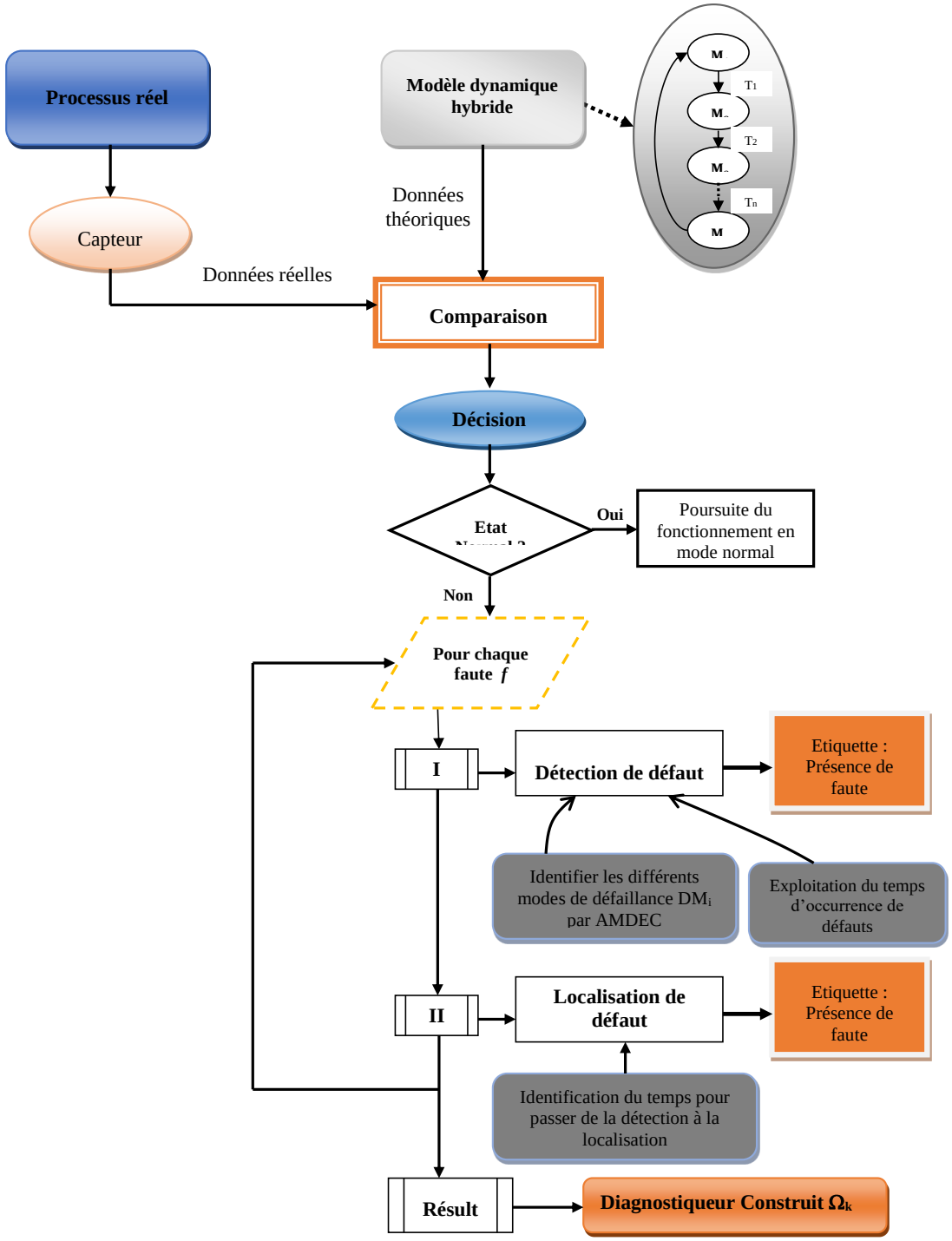


FIGURE IV.11 – Organigramme de la construction d'un diagnostiqueur [139]

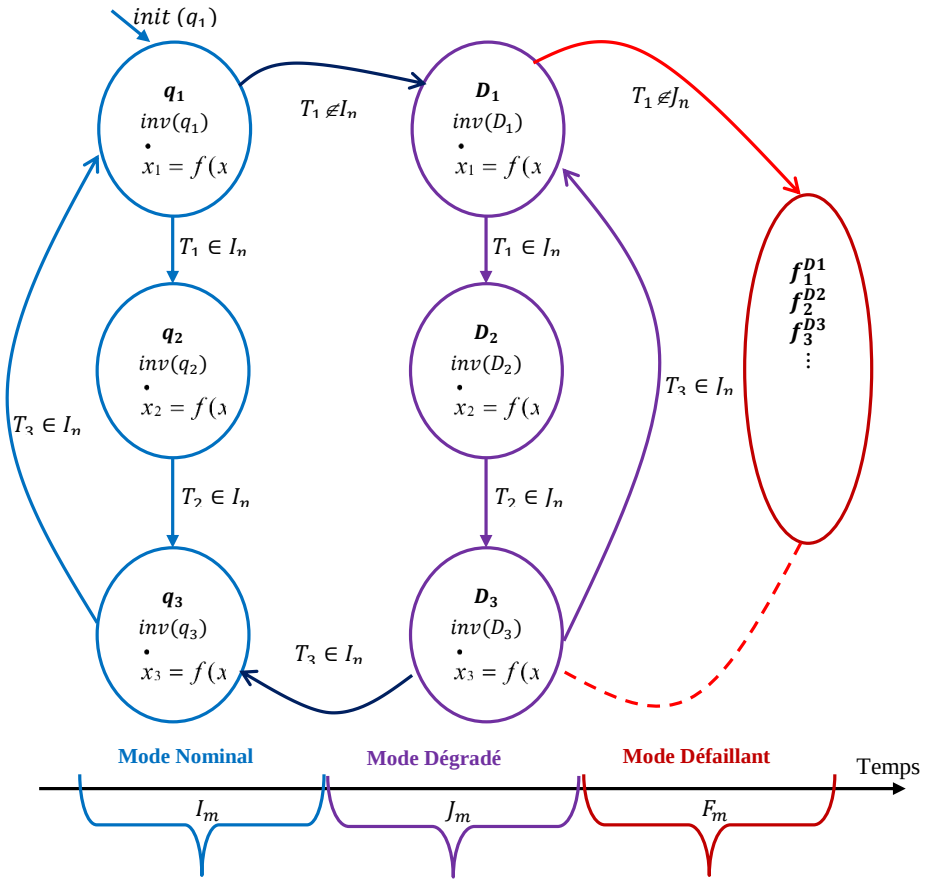


FIGURE IV.12 – Construction du diagnostiqueur par un automate hybride global [143]

Nous simulons notre modèle Matlab / Simulink / Stateflow pour un cycle de fonctionnement normal du système hydraulique et identifions les durées d'activité de chaque mode à partir des dates d'instrumentation des vannes ou des dates de changement de valeur des capteurs (table 2).

Après l'analyse FAST et AMDEC, nous identifions sept défauts possibles sur les capteurs Sh_1, Sh_2 et les vannes V_3 et V_4. La figure IV.16 présente la liste des défauts possibles.

Nous simulons le système en injectant chacun de ces défauts séparément et identifions les modes dégradés et défaillants du système hydraulique. Les trois modes de fonctionnement du système sont alors définis par :

— $N_i = \{\text{Mode}_1, \text{Mode}_2, \text{Mode}_3, \text{Mode}_4\}$

— $D_i = \{DM_1, DM_2, DM_3, DM_4\}$

— $F_i = \{Sh_1\_SO, Sh_1\_SC, Sh_2\_SO, Sh_2\_SC, V_3\_SO, V_4\_SC, V_4\_SO\}$

L'automate hybride global du système construit à partir de nos simulations et qui nous permet la détection et l'isolation des défauts est présenté sur la figure IV.17.

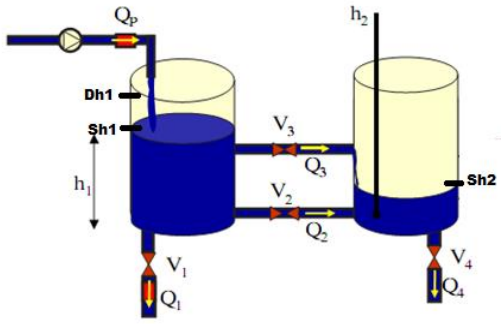


FIGURE IV.13 – Exemple d'un système hydraulique à deux réservoirs [143]

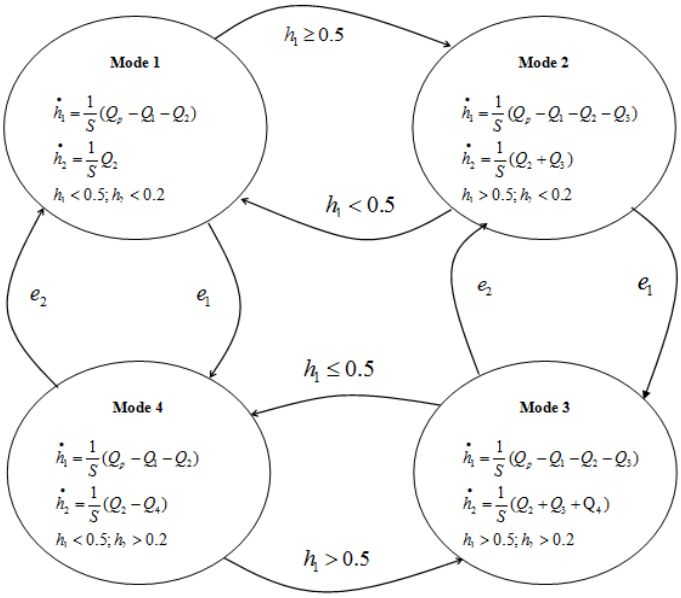


FIGURE IV.14 – Automate hybride du système hydraulique [143]

TABLE 2 – Identification des durées des modes de fonctionnement du système hydraulique [144]

Actions	Time in sec	Interval of time
Opening the pump Q_p Closing the pump Q_p	0 20	$[0, 20]$
Opening the valve V_4 Closing the valve V_4	240 380	$[240, 380]$
Activating the sensor Sh_1 Deactivating the sensor Sh_1	7.7 60	$[7.7, 60]$
Activating the sensor Sh_2 Deactivating the sensor Sh_2	20 180	$[20, 180]$

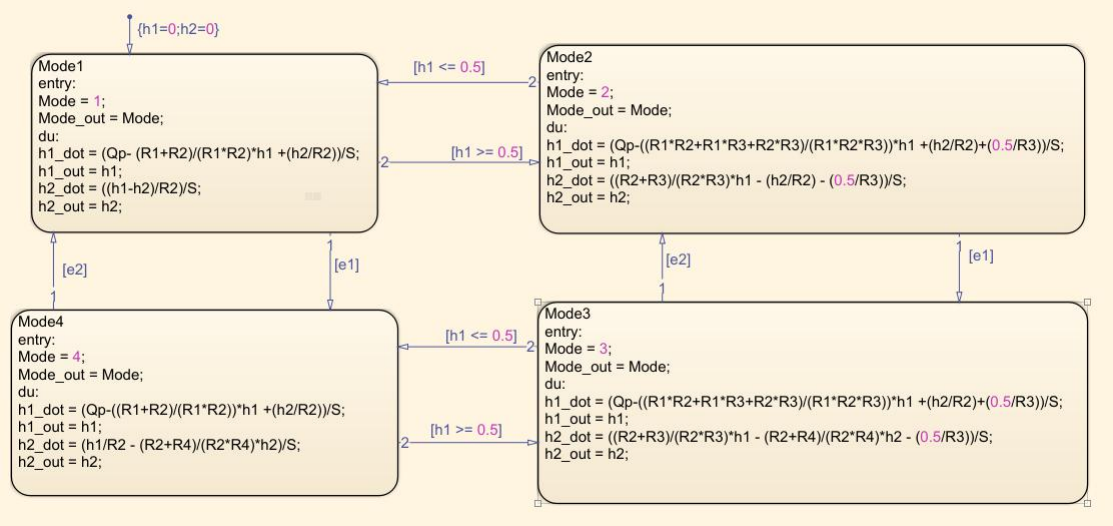


FIGURE IV.15 – Modèle Stateflow du processus hydraulique [143]

N°	Mode défaillance (Capteur)	Réf	N°	Mode de défaillance (Vannes)	Réf
1	Ne détecte pas la hausse du niveau (reste toujours à l'état 0)	**Sh1_SO**	**5**	Reste ouverte lors d'une demande de fermeture (V4 reste à l'état 1)	**V4_SO**
2	Ne détecte pas la baisse du niveau (reste toujours à l'état 1)	**Sh1_SC**	**6**	Reste fermé lors d'une demande d'ouverture (V4 reste à l'état 0)	**V4_SC**
3	Ne détecte pas la hausse du niveau (reste toujours à l'état 0)	**Sh2_SO**	**7**	Reste fermé lors (C3 reste à l'état 0)	**V3_SC**
4	Ne détecte pas la baisse du niveau (reste toujours à l'état 1)	**Sh2_SC**			

FIGURE IV.16 – Liste des défauts pouvant affecter le système hydraulique [139]

3 Conclusion et perspectives sur le diagnostic de systèmes dynamiques hybrides

Dans ce chapitre, nous avons proposé deux *approches basées modèle* pour le diagnostic de systèmes dynamiques hybrides. Nous avons considéré des défauts permanents sur les actionneurs, les capteurs et les composants du système. Les méthodes de diagnostic développées prennent en compte des défauts uniques.

La première méthodologie de diagnostic que nous avons développé est une méthode de diagnostic *mixte* qui dispose au départ d'une description du SDH sous la forme d'un bond graph. La méthode de détection et de localisation de défauts est basée sur l'utilisation d'un observateur (méthode issue des systèmes continus) et d'un automate temporisé (méthode issue des systèmes à événements discrets). Dans cette approche, nous avons considéré le problème du diagnostic pour des systèmes hybrides soumis à des effets indésirables dus

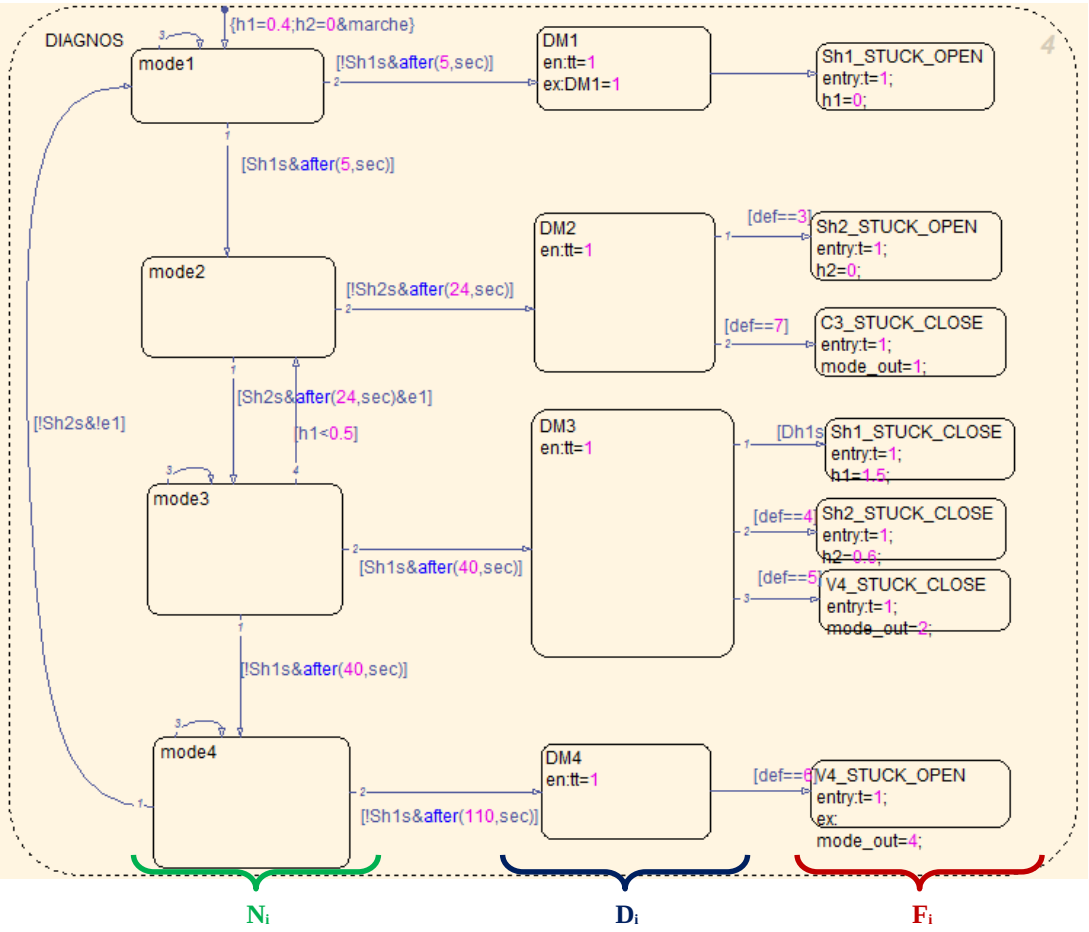


FIGURE IV.17 – Automate hybride global du système hydraulique [143]

à des entrées inconnues (bruit de mesure, perturbations externes, etc.). L'approche de diagnostic à base d'observateurs que nous proposons permet alors d'obtenir à la fois une robustesse par rapport aux entrées inconnues et une sensibilité à l'apparition de défauts. Notre méthode de diagnostic combine l'analyse de signatures avec l'analyse temporelle pour détecter et localiser les défaillances dans les systèmes dynamiques hybrides.

La deuxième méthodologie de diagnostic pour les systèmes dynamiques hybrides que nous avons proposés est également une approche mixte pour la modélisation du SDH à base d'automates hybrides. A partir de la description du fonctionnement nominal du SDH, des analyses FAST et AMDEC sont réalisées pour étendre la modélisation du SDH aux modes de fonctionnement dégradé et défaillant. La simulation du modèle en présence de défauts permet l'identification temporelle des différents modes de fonctionnement du système et la construction d'un automate hybride global du système qui nous sert de diagnostiqueur pour détecteur et isoler les défauts.

Les travaux que nous avons menés sur le diagnostic de systèmes dynamiques hybrides sont inscrit dans une collaboration internationale avec le laboratoire tunisien LARATSI lors de deux thèses [138], [139] et de plusieurs stages de Master 2. Cette collaboration sur le diagnostic de systèmes dynamiques hybrides a fait l'objet de plusieurs publications [142]-[147].

La première perspective de ces travaux est d'étendre nos méthodologies de diagnostic aux défauts multiples. La construction d'un modèle du système complexe en présence de défauts multiples est un problème difficile (voir chapitre III), il y a en effet une explosion combinatoire des modes de défaillances possibles. Notre perspective de recherche sur ce sujet est d'étudier les combinaisons de défaillances multiples ayant les probabilités les plus significatives et d'utiliser une méthode de diagnostic combinant une matrice de signatures de défauts et un automate temporisé pour distinguer les différents scénarios de défaillances uniques et multiples.

Une seconde perspective est de continuer le travail initié dans la thèse de Lobna Belkacem sur la construction d'un module de pronostic basé sur l'automate hybride et utilisant une politique de maintenance. Nous avons débutée des travaux de recherche sur le pronostic selon deux types politiques de maintenance [139], [143] : la première est une politique de réparation par remise en état (ABAO), la seconde est une politique de réparation par remplacement (AGAN).

Chapitre V

Modélisation et diagnostic de systèmes cyber-physiques

1 Présentation générale : contexte, état de l'art, positionnement, originalité

1.1 Contexte, positionnement, originalité

Ce chapitre présente l'extension de mes travaux de recherches sur les systèmes à événements discrets temporisés et les systèmes dynamiques hybrides aux systèmes cyber-physiques. Les travaux relatés dans ce chapitre sont dans le cadre de la thèse de Mohamed Amine Haj Kacem [148] que j'ai co-encadré.

Les systèmes cyber-physiques ont été définis pas la communauté scientifique selon différentes perspectives. Tout d'abord par Lee [149], [150] : « *Cyber-Physical Systems (CPS) are integrations of computation with physical processes. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa* ». D'autres chercheurs ont proposé d'autres définitions des CPS comme Rajkumar et al. [151] : « *Cyber-physical systems (CPS) are physical and engineered systems whose operations are monitored, coordinated, controlled and integrated by a computing and communication core* ». Monostori en 2014 [152] propose une synthèse des différentes définitions des systèmes cyber-physiques existantes dans la littérature ainsi : « *Cyber-Physical Systems (CPS) are systems of collaborating computational entities which are in intensive connection with the surrounding physical world and its on-going processes, providing and using, at the same time, data-accessing and data-processing services available on the internet* ». Les systèmes cyber-physiques sont ainsi des systèmes embarqués complexes dont la finalité est d'interagir avec l'environnement physique de manière continue et dynamique. Un CPS est composé d'un monde physique, de systèmes cyber et d'interfaces [153].

Le monde physique contient les phénomènes physiques que l'on souhaite surveiller et contrôler. Dans le monde physique, nous proposons outre les phénomènes physiques d'y incorporer également les composants physiques du CPS qui peuvent être surveillés et/ou

contrôlés, par exemple les moteurs, les réservoirs, les roues, etc.

Les systèmes cyber contiennent les capacités de stockage et de traitement de l'information qui peuvent être embarquées sur le CPS ou accessibles sur le réseau (cloud computing), les communications sur les réseaux, les services accessibles à partir du réseau internet, etc.

Les interfaces contiennent les composants intermédiaires entre le monde physique et les systèmes cyber : les capteurs, les actionneurs, les convertisseurs de valeurs physiques en données numériques, les systèmes de commande (e.g. PLC), les systèmes de contrôle et d'acquisition de données (e.g. SCADA), les éléments physiques pour les réseaux de communication, etc.

La figure V.1 représente cette vision globale d'un système cyber-physique.

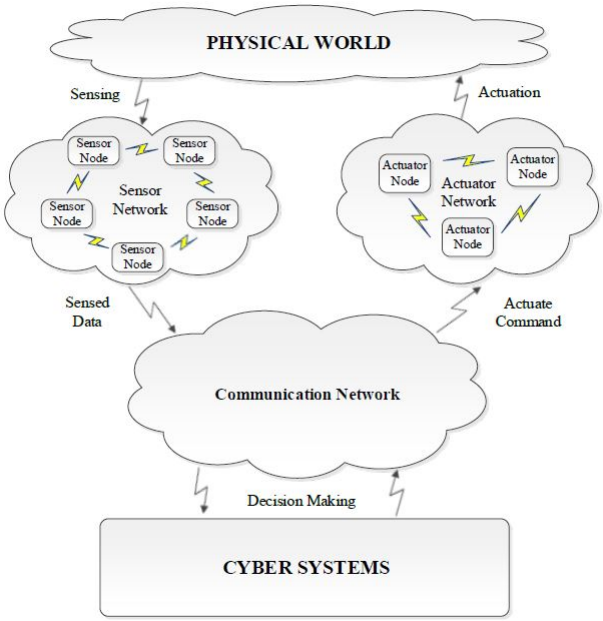


FIGURE V.1 – Vision holistique des systèmes cyber-physiques [153]

Les systèmes cyber-physiques offrent ainsi plusieurs capacités : le calcul, la communication, le contrôle/commande, la collaboration à distance entre différents composants du CPS ou entre plusieurs CPS, et l'autonomie [154]. La figure V.2 montre les interactions entre trois concepts essentiels des CPS : le calcul, la commande et la communication.

Les systèmes cyber-physiques ont des applications dans de nombreux domaines [153], [156] comme :

— industriel : dans le cadre de l'Industrie 4.0 avec les systèmes cyber-physiques de production et les usines intelligentes (smart factory, smart manufacturing) ;

— énergétique : avec les réseaux électriques intelligents (smart grid) ;

— infrastructure : dans le cadre du bâtiment intelligent (smart building) et des travaux publics ;

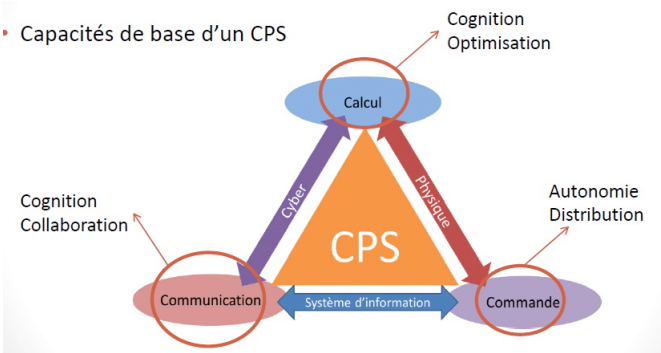


FIGURE V.2 – Principes des systèmes cyber-physiques [154], [155]

— transport : individuel ou collectif de personnes ou de marchandises (smart transport) ;

— santé : avec l'essor de la e-santé, télémédecine, téléchirurgie, des dispositifs médicaux invovants et intelligents (smart health) ;

— social : les systèmes socio-cyber physiques pour intégrer des processus physiques et sociaux ;

— défense, robotique, etc.

Les systèmes cyber-physiques sont des technologies hétérogènes par nature : ils combinent des sous-systèmes mécaniques, électroniques, du logiciel embarqué et du réseau. Ils associent à la fois du calcul, de la communication, du contrôle ainsi que de la dynamique physique. Cette hétérogénéité les rend alors plus difficiles à modéliser, à concevoir et à analyser que les systèmes homogènes [157]. Des solutions partielles existent pour chacune de ces typologies de sous-systèmes, mais ne sont pas adaptées à un système cyber-physique dans sa globalité.

Les thématiques de recherches sur les systèmes cyber-physiques sont diversifiées, nous pouvons mentionner parmi celles-ci :

— Le développement d'outils et de méthodologies de conception des systèmes cyber-physiques intégrant les concepts de sûreté de fonctionnement et robustesse.

— L'analyse et l'évaluation des performances et de la robustesse des CPS.

— La sécurité des systèmes cyber-physiques : sécurisation des CPS dès leur conception en intégrant des contre-mesures, développement de méthodes de détection d'intrusion et de protection contre les cyber-attaques.

— La sûreté de fonctionnement des systèmes cyber-physiques : diagnostic des défaillances des CPS, l'évaluation de l'état de santé (health monitoring) des CPS, pronostic des défaillances et aide à la décision dans la maintenance des CPS.

Nos travaux de recherche ont porté sur le développement de méthodes d'aide à la détection et au diagnostic de défaillances dans un système cyber-physique. Nous avons délimité notre étude aux défaillances fonctionnelles [158] causées par des défauts sur les composants.

Nous n'avons pas étudié les défaillances des CPS causées par des cyber-attaques provoquant des déviations du comportement des composants.

Les systèmes cyber-physiques sont présents de plus en plus dans notre société par leur usage dans l'industrie 4.0, les réseaux électriques intelligents, la domotique, les voitures autonomes, etc., et également dans des produits grand public. Une défaillance dans un système cyber-physique peut avoir des impacts économiques, humains, matériels ou environnementaux. Il est nécessaire de disposer de méthodes de diagnostic permettant de détecter une défaillance dans un CPS le plus tôt possible et d'isoler le composant défaillant afin de procéder à la maintenance corrective du CPS . Cependant la complexité croissante de ces systèmes à technologies hétérogènes, ne permet plus d'utiliser les méthodes de diagnostic spécifiques à chacune des technologies utilisées dans le CPS, mais nécessite des techniques de diagnostic de plus en plus élaborées.

Nous considérons que le CPS est composé de composants réparables ou remplaçables. Les composants disposant de fonctions d'auto-diagnostic sont qualifiés d'*observables*, ceux n'en disposant pas sont qualifiés de *non observables*. Un composant observable fournit son statut, par exemple OK pour un fonctionnement normal du composant, DC (Défaillance Connue) pour une défaillance connue du composant, HS (Hors Service) pour un comportement anormal causé par la défaillance d'un autre composant dont il dépend structurellement, ou DI (Défaillance Inconnue) lorsque la fonction d'auto-diagnostic du composant ne permet pas de définir la nature de la défaillance du composant [159]. Lorsqu'une défaillance est détectée par le module de surveillance, la phase de localisation du diagnostic fournit à l'équipe de maintenance la liste des composants observables et non observables susceptibles d'être à l'origine d'une défaillance. L'équipe de maintenance peut rapidement diagnostiquer les composants observables grâce aux données fournies par ceux-ci. Concernant la liste des composants non observables, l'équipe de maintenance doit procéder sur ceux-ci à des inspections et tests pour savoir s'ils sont à l'origine de la défaillance. Pour optimiser cette étape d'inspection et de test, il est alors nécessaire d'affiner la liste des composants non observables susceptibles d'être à l'origine de la défaillance.

Notre problématique de recherche dans le diagnostic des CPS est alors de proposer à l'équipe de maintenance corrective :

- *des outils d'aide à la décision pour affiner la localisation des composants non observables susceptibles d'être à l'origine d'une défaillance ;*
- *une méthodologie de construction d'un diagnostiqueur pour une signature de défaut donné qui permet de réduire ou résoudre les ambiguïtés de localisation des composants non observables à l'origine d'une défaillance.*

Dans nos travaux de recherche, nous considérons le CPS dans sa globalité et nous le modélisons avec plusieurs catégories de connaissances : fonctionnelle, structurelle, topologique et comportementale [160]. A partir de ces connaissances, nous proposons tout d'abord une aide à la localisation des composants susceptibles d'être à l'origine d'une défaillance. Pour lever les ambiguïtés de localisation, notre démarche est d'élaborer un fonctionnement du

CPS qui permette de solliciter les composants sur lesquels il y a une ambiguïté de localisation de la défaillance, et d'utiliser un diagnostiqueur à base d'automates temporisés pour affiner la localisation du ou des composants défaillants et identifier la cause de la défaillance.

1.2 Etat de l'art

Les travaux de recherches sur le diagnostic de défaillances de systèmes cyber-physiques sont de plus en plus nombreux cette dernière décennie [161]. La majorité de ces travaux appliquent des méthodologies de diagnostic utilisées dans les communautés DES (Discrete Event System), FDI (Fault Detection and Isolation) ou de l'intelligence artificielle (qui contient la communauté DX (Diagnosis eXpert system)) de manière indépendante à des sous-systèmes de CPS, mais il y a très peu de travaux de recherche qui ont porté sur le développement de méthodes de diagnostic de défaillances adaptées spécifiquement aux systèmes cyber-physiques et en les considérant dans leur globalité.

Le diagnostic des défaillances fonctionnelles des systèmes cyber-physiques peut être réalisé principalement par trois approches [161] : (1) basée sur la physique, (2) pilotée par les données et (3) basée sur la connaissance. Le lecteur peut se référer à l'état de l'art du chapitre I qui présente plus en détail la classification des méthodes de diagnostic et à l'état de l'art du chapitre IV qui présente plus spécifiquement les travaux de recherche sur le diagnostic des systèmes hybrides.

Avec les approches basées sur la physique qui sont des méthodes avec modèles, la phase de modélisation du CPS prend en compte la dynamique physique de ses composants. Des équations basées sur la physique, comme des équations de dépendance, des équations différentielles ordinaires ou des équations différentielles partielles, sont utilisées pour modéliser les caractéristiques physiques des composants du CPS. La méthode de diagnostic consiste à vérifier la cohérence entre les valeurs détectées et les valeurs prédites par les modèles basés sur la physique. Les filtres de Kalman [162], les modèles de Markov cachés [163] peuvent être utilisés comme observateurs et permettent de détecter et de localiser le composant défectueux.

Avec les approches basées sur les données qui sont des méthodes sans modèle, le comportement du CPS est obtenu par l'entraînement et l'apprentissage à partir d'un grand ensemble de valeurs détectées (approches de l'intelligence artificielle). Ces données sont étiquetées comme provenant d'un mode normal ou d'un mode dégradé ou défectueux. La méthode de diagnostic consiste en deux étapes : premièrement, une représentation du comportement du CPS en mode normal et en mode dégradé/défaillant est construite à partir des données étiquetées. Deuxièmement, une théorie logique de diagnostic est utilisée pour analyser les valeurs actuelles détectées. Des techniques d'intelligence artificielle comme les réseaux neuronaux [164], [165] ou l'apprentissage automatique [166] sont utilisés dans ces approches basées sur les données.

Avec les approches basées sur la connaissance, à partir d'une grande quantité de données (pas nécessairement étiquetées) et de la connaissance d'experts, une représentation de la

dynamique du CPS est construite. Ce modèle dynamique peut fournir à la fois un diagnostic et un pronostic. Dans les approches basées sur la connaissance appliquées aux CPS, nous pouvons mentionner l'utilisation de réseaux bayésiens [167], [168], de réseaux de Petri [169] et de systèmes experts [170]. Dans [171], les auteurs proposent une méthode de détection et de prévention des défaillances dans les systèmes cyber-physique en utilisant une base de connaissance construite à partir d'une ontologie obtenue par l'Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité (FMECA).

Il existe également des approches de diagnostic hybrides qui mélangent les approches ci-dessus telles que [172], [173]. Dans [172], la modélisation du système cyber-physique fusionne des modèles basés sur la physique et sur les données. La détection des anomalies et des défaillances est réalisée par de résidus générés. Le diagnostic est obtenu par une méthode de classification telle que les machines à vecteurs de support. Dans [173], le CPS est représenté par un automate hybride utilisant des prédicats logiques et des résidus. La méthode de diagnostic utilise les résidus et de l'apprentissage pour construire un automate de surveillance qui est utilisé pour réaliser le diagnostic en ligne du CPS.

2 Synthèse des travaux développés

2.1 Méthodologie de modélisation d'un système cyber-physique

Un système cyber-physique est composé de plusieurs sous-systèmes, dont la conception et le fonctionnement font intervenir différents domaines tels que domaines l'électronique, l'informatique, la mécanique, le réseau, la gestion de données, etc. Ces sous-systèmes collaborent entre eux pour réaliser différentes fonctionnalités ou services. Chaque sous-système délivre donc un ensemble de fonctions. La réalisation d'une fonction est assurée par un ou plusieurs composants. Il est à noter qu'un composant peut être sollicité par plusieurs fonctions.

Nous nous intéressons au diagnostic de défaillances sur les systèmes cyber-physique, nous cherchons donc à détecter et à localiser le composant source d'une défaillance afin de procéder à sa maintenance corrective. Nous avons par conséquent besoin d'une modélisation du système cyber-physique qui exprime l'organisation hiérarchique du CPS, les interactions et dépendances entre les composants, les fonctionnalités assurées par les composants et les modes de fonctionnement des composants. Notre méthodologie de modélisation utilise ainsi une représentation multi points de vue des connaissances sur le CPS : structurelle, fonctionnelle, topologique et comportementale [159].

2.1.1 Démarche de modélisation

Pour construire un modèle d'un système cyber-physique, nous procédons à deux études de ce système : une étude statique et une étude dynamique. L'étude statique utilise des méthodes qualitatives telles que :

— Une analyse fonctionnelle qui consiste à rechercher et à caractériser les fonctions

offertes par un produit pour satisfaire les besoins d'un utilisateur (norme AFNOR X50-151 puis NF EN 16271). Celle-ci est décomposée en analyse fonctionnelle externe pour mettre en évidence les fonctions du produit et en analyse fonctionnelle interne pour comprendre l'architecture, la combinaison des composants et les fonctions techniques. Les diagrammes FAST et pieuvre peuvent être utilisés pour décrire respectivement l'analyse fonctionnelle externe et interne.

— Une analyse de l'architecture du système qui consiste à définir les éléments de la structure du système en sous-systèmes et de la définition des sous-systèmes à base de composants. Des diagrammes de définitions de blocs et de bloc interne expriment l'architecture du système et présentent les liaisons entre les différents sous-systèmes et entre différents composants.

— Une analyse de défaillances qui permet une analyse systématique des défaillances des sous-systèmes et de leurs composants. Cette analyse peur se dérouler en deux étapes : tout d'abord une analyse AMDEC pour déterminer les causes des différents modes de défaillance et leurs criticités, puis la construction d'arbres de défaillances pour déterminer quelles combinaisons d'événements peuvent mener à des défaillances.

L'étude dynamique consiste à modéliser le comportement dynamique du système afin de pouvoir simuler celui-ci dans un comportement normal. Différents outils de modélisation peuvent être utilisés : Bond graphs, équations différentielles, les systèmes à événements discrets, les automates temporisés, les réseaux de Petri, etc. Par partir de la modélisation de la dynamique du système, un module de surveillance peut/être construit pour détecter un écart entre le comportement réel du système et son comportement attendu.

2.1.2 Notations

Nous utilisons une décomposition conceptuelle des systèmes cyber-physiques en sous-systèmes, fonctions et composants. Un système cyber-physique est représenté par la composition de n sous-systèmes :

$$CPS = \{SP_1, \ldots, SP_n\}$$

Un sous-système SP_i fournit m fonctionnalités/fonctions :

$$SP_i = \{F_{i_1}, \ldots, F_{i_m}\}$$

. La réalisation d'une fonction F_{i_j} repose sur p composants (CR pour composant remplaçable et/ou réparable) :

$$F_{i_j} = \{CR^1_{i_j}, \ldots, CR^p_{i_j}\}$$

2.1.3 Représentation de la connaissance structurelle

La connaissance structurelle vise à représenter l'ensemble des composants du système implémentant une fonction ainsi que leurs interconnexions.

Pour modéliser ces interconnexions, nous utilisons la notation $O_x(CR^k_{i_j})$ pour représenter la sortie numéro x du composant $CR^k_{i_j}$ et $I_y(CR^l_{i_j})$ pour représenter l'entrée numéro y du composant $CR^l_{i_j}$.

La connaissance structurelle de la fonction F_{i_j} se définit alors par un ensemble de relations $M_{S_{i_j}} = \{O_x(CR^k_{i_j}) \rightarrow I_y(CR^l_{i_j}) | CR^k_{i_j}, CR^l_{i_j} \in F_{i_j}\}$ exprimant que la sortie x du composant $CR^k_{i_j}$ est reliée à l'entrée y du composant $CR^l_{i_j}$. Cette connaissance structurelle permet de connaitre les dépendances entre les composants et d'affiner le diagnostic pour tenir compte des effets de propagation de défaillances entre composants interconnectés.

2.1.4 Représentation de la connaissance fonctionnelle

La connaissance fonctionnelle vise à représenter les composants concourants à l'implémentation des différentes fonctions d'un sous-système. La connaissance fonctionnelle du sous-système SP_i est représentée par la matrice M_{F_i} dont les colonnes sont indexées par tous les composants intervenant dans le sous-système SP_i et les lignes sont indexées par toutes les fonctions fournies par le sous-système. Un coefficient $\alpha_{u,v}$ de la matrice vaut 1 si le composant associé à sa colonne v ($CR^v_{i_j}$) intervient dans la réalisation de la fonction associée à sa ligne u (F_{i_u}), 0 sinon.

$$M_{F_i} = \begin{array}{c} \\ F_{i_1} \\ F_{i_2} \\ \dots \\ F_{i_m} \end{array} \begin{array}{c} \begin{array}{cccc} CR^1_{i_j} & CR^2_{i_j} & \dots & CR^p_{i_j} \end{array} \\ \begin{pmatrix} \alpha_{1,1} & \alpha_{1,2} & \dots & \alpha_{1,p} \\ \alpha_{2,1} & \alpha_{2,2} & \dots & \alpha_{2,p} \\ \dots & \dots & \alpha_{u,v} & \dots \\ \alpha_{m,1} & \alpha_{m,2} & \dots & \alpha_{m,p} \end{pmatrix} \end{array}$$

Cette connaissance fonctionnelle est utilisée après la détection d'une défaillance : la temporalité de la défaillance nous permet de connaitre les fonctions qui sont sollicitées au moment de la détection de la défaillance. Avec l'identification des fonctions sollicitées, nous pouvons connaitre la liste des composants observables et non observables pouvant être à l'origine de la défaillance. La vérification de l'état de santé des composants observables par l'équipe de maintenance permet d'affiner la localisation des composants défaillants, cependant concernant les composants non observables, il reste des ambiguïtés de localisation. L'équipe de maintenance corrective devra tester et inspecter ces composants non observables afin de résoudre ces ambiguïtés, ce qui augmente le coût de la maintenance. Nos travaux de recherche visent à diminuer le coût de cette maintenance en affinant la localisation des composants non observables susceptibles d'être à l'origine de la défaillance.

2.1.5 Représentation de la connaissance topologique

La connaissance topologique vise à représenter les influences par effet de proximité (électrique, magnétique, thermique, physique, etc.) que les composants peuvent avoir entre eux. Nous ne représentons dans la connaissance topologique que les influences indirectes

où les composants ne sont pas structurellement liés (la connaissance structurelle apporte déjà une connaissance sur les composants liés).

La connaissance topologique entre les composants du sous-système SP_i se définit alors par un ensemble de relations $M_{T_i} = \{CR^k_{i_u} \Rightarrow CR^l_{i_v} | CR^k_{i_u}, CR^l_{i_v} \in SP_i\}$ exprimant que la défaillance du composant $CR^k_{i_u}$ peut entraîner une défaillance sur le composant $CR^l_{i_v}$. La dépendance est unidirectionnelle. Cette connaissance topologique permet d'affiner le diagnostic pour tenir compte des effets de proximité de défaillances entre composants.

2.1.6 Représentation de la connaissance comportementale

La connaissance comportementale vise à représenter le comportement global du système. Cette modélisation décrit les différents modes de fonctionnement des composants et des fonctions du CPS ainsi que leurs modes de défaillance. La connaissance comportementale contient la représentation du comportement, de la dynamique de chaque composant dans un formalisme adéquat (diagramme, équations différentielles, automates, etc.).

Cette connaissance comportementale est utilisée pour la construction du module de surveillance afin de détecter une défaillance et pour la construction du diagnostiqueur.

2.2 Méthodologie de détection, localisation et identification des défaillances

La méthodologie de diagnostic que nous proposons se déroule en deux étapes :

— Dans la première étape, à partir des différentes connaissances du CPS que nous avons modélisé, nous calculons les coupes minimales et identifions les composants qui peuvent avoir un problème d'ambiguïté de localisation en cas de défaillance. Pour lever des ambiguïtés de localisation, des automates temporisés sont utilisés. Ces automates utilisent les temps de sollicitation des composants pour les fonctions proposées par le CPS pour affiner l'identification des composants non observables. A l'issue de cette étape, il peut rester des groupes de composants non observables sur lesquels un problème d'ambiguïté d'identification existe, il sera alors nécessaire de construire un diagnostiqueur pour affiner ou résoudre ces ambiguïtés.

— Dans la deuxième étape, s'il reste des ambiguïtés d'identification sur des groupes de composants non observables, nous construisons des séquences de fonctionnement du CPS qui sollicitent ces composants non observables ambigus pouvant être à l'origine d'une défaillance et nous construisons des diagnostiqueurs pour affiner ou lever les ambiguïtés sur l'origine d'une défaillance.

Ces deux étapes se déroulent avant l'usage en condition réelle du CPS auprès de ses utilisateurs, elles visent à construire des diagnostiqueurs associés à des usages du CPS qui permettront lorsqu'une défaillance apparaitra sur le CPS d'aider l'équipe de maintenance à localiser et à identifier le composant à l'origine de la défaillance.

2.2.1 Aide à la localisation des composants non observables défaillants

2.2.1.1 Construction des chemins de succès des fonctions et coupes minimales

Chemins de succès

Un chemin de succès est associé à chacune des fonctions F_{i_j} que fournit le CPS. A partir de la représentation fonctionnelle M_{F_i}, on définit le chemin de succès CS_{i_j} comme l'ensemble des composants qui participe à la réalisation de la fonction F_{i_j}. On distingue dans le chemin de succès les composants observables des composants non observables.

Coupes minimales

Une coupe CM_{i_j} correspond à la liste des composants sollicités par une fonction F_{i_j} dont la défaillance entraîne la défaillance du sous-système SP_i. Une coupe est dite minimale si en retirant n'importe quel composant de la liste, le sous-système SP_i n'est plus défaillant.

La connaissance par l'équipe de maintenance des chemins de succès, des coupes minimales et des composants observables qui les composent leur permet de construire la séquence de tests et d'inspections des composants. Ils pourront prioriser les composants observables puis élaborer la stratégie de test optimale pour les composants non observables [174].

2.2.1.2 Démarche générale de la détection à la localisation

Plusieurs étapes constituent l'approche que nous proposons pour l'aide à la localisation de composants défaillants. Le principe général est donné sur la figure V.3.

La première étape identifie les ambiguïtés de localisation qu'il peut exister sur des composants dans le cas où l'une des fonctions qu'ils assurent est défaillante. Afin de lever les ambiguïtés, une deuxième étape basée sur l'utilisation d'un automate temporisé pour identifier lors du fonctionnement du CPS les fonctionnalités défaillantes et de réduire le nombre de composants potentiellement source de la défaillance. Pour chacune des étapes, un algorithme est développé.

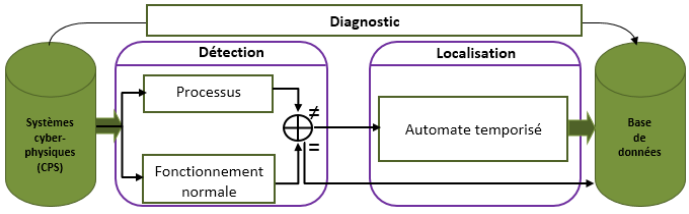


FIGURE V.3 – Principe général de réduction des ambiguïtés de localisation de défaillance [148]

2.2.1.2.1 Identification des ambiguïtés de localisation des composants

Lorsque le module de surveillance détectera une défaillance, l'équipe de maintenance

connaitra l'ensemble des fonctionnalités sollicitées au moment de la détection. Les composants intervenants dans les fonctions sollicitées au moment de la détection de la défaillance sont tous potentiellement défaillants. L'équipe de maintenance doit de manière optimale choisir quels sont les composants à tester et/ou inspecter. Nous nous intéressons ici au test et à l'inspection des composants non observables, les composants observables fournissent déjà à l'équipe de maintenance leur statut de fonctionnement.

Si un composant non observable n'intervient que dans une fonction, l'équipe de maintenance devra nécessairement tester et inspecter ce composant, car on ne peut pas discriminer l'état de santé de ce composant par rapport à d'autres fonctions où il interviendrait.

Si un composant non observable intervient dans plusieurs fonctionnalités, il y a un risque d'ambiguïté de localisation, il sera nécessaire de vérifier si les fonctions où ce composant intervient sont défaillantes.

A partir de la connaissance fonctionnelle de chaque sous-système SP_i décrite par la matrice M_{F_i}, on définit pour chaque composant $CR_{i_j}^k$ le vecteur colonne $V_{F_{i_j}}^k$ qui décrit les fonctions auxquelles il participe.

Pour vérifier le bon fonctionnement du système et détecter la source de défaillance des composants remplaçables, nous utilisons l'algorithme V.1 qui permet à partir des vecteurs $V_{F_{i_j}}^k$ d'identifier la liste des composants ayant un problème d'ambiguïté de localisation en cas de défaillance détectée sur le système SP_i.

Algorithme V.1 Identification des ambiguïtés de localisation des composants pour le sous-système SP_i

procedure DetectionComposantsAmbigus
Input: $F_i = \{F_{i_1}, \ldots, F_{i_m}\}$: Ensemble des fonctions du sous-système SP_i
Input: $CR_i = \{\ldots, CR_{i_j}^1, \ldots, CR_{i_j}^p, \ldots\}$: Ensemble des composants intervenant dans SP_i
Input: $V_i = \{\ldots, V_{F_{i_j}}^1, \ldots, V_{F_{i_j}}^p, \ldots\}$: Ensemble des vecteurs colonnes du sous-système SP_i
Output: ℓ_{CRA}^i : Ensemble des composants ambigus
1: $m := |F_i|$ {nombre de fonctions intervenant dans SP_i}
2: $\ell_{CRA}^i := \emptyset$
3: **for** $k := 1$ **to** $|CR_i|$ **do** {$|CR_i|$: nombre de composants dans SP_i}
4: $\quad CR_{i_j}^k :=$ k-ième élément de l'ensemble C_i
5: $\quad V_{F_{i_j}}^k :=$ k-ième élément de l'ensemble V_i
6: $\quad s := 0$
7: $\quad$ **for** $n := 1$ **to** m **do**
8: $\quad\quad s := s + V_{F_{i_j}}^k(n)$ {$V_{F_{i_j}}^k(n)$ est la valeur de la n-ième ligne du vecteur colonne $V_{F_{i_j}}^k$ }
9: $\quad$ **end for**
10: $\quad$ **if** $(s \geq 2)$ **then**
11: $\quad\quad \ell_{CRA}^i := \ell_{CRA}^i \cup \{CR_{i_j}^k\}$
12: $\quad$ **end if**
13: **end for**
14: **return** ℓ_{CRA}^i
end procedure

2.2.1.2.2 Réduction des ambiguïtés de localisation des composants défaillants

Pour résoudre le problème d'ambiguïté, on utilise la connaissance comportementale du CPS. Celle-ci nous fournit un automate temporisé qui exprime les temps de démarrage $T^d_{i_j}$ et de fin $T^f_{i_j}$ de chaque fonction F_{i_j}. Nous pouvons alors comparer le temps de détection de la défaillance T_{defect} avec le temps de sollicitation de chaque fonction.

L'algorithme V.2 utilise le résultat de l'algorithme V.1 et réalise une analyse temporelle au niveau des fonctions. L'algorithme détermine si chaque composant sollicité au moment de la défaillance peut être discriminé par rapport aux autres composants. Si un composant intervient dans une fonction sollicitée au moment de la défaillance et n'utilise qu'un seul composant ou si le composant est observable, alors ce composant devra être testé par l'équipe de maintenance (statut *inspection*). S'il reste des problèmes d'ambiguïtés d'identification, l'algorithme permet de réduire la liste des composants ambigus et susceptibles d'être défaillants.

Algorithme V.2 Réduction des ambiguïtés de localisation des composants pour le sous-système SP_i

procedure IDENTIFICATIONCOMPOSANTSAMBIGUS
Input: t_{defect} : Temps de détection de la défaillance
Input: $F_i = \{F_{i_1}, \ldots, F_{i_m}\}$: Ensemble des fonctions du sous-système SP_i
Input: T^d : Vecteur des temps de démarrage des fonctions F_i
Input: T^f : Vecteur des temps de fin des fonctions F_i
Input: ℓ^i_{CRA} : Ensemble des composants ambigus
Input: $O_i = \{O_{i_1}, \ldots, O_{i_m}\}$: Ensemble des vecteur d'observations des composants intervenant dans SP_i
Output: ℓ^i_{CR} : Simplification de l'ensemble des composants ambigus et non observables

```
m := |F_i| {nombre de fonctions intervenant dans SP_i}
ℓ^i_F := ∅
for j := 1 to m do
    if (T^d(j) ≤ T_defect ∧ T_defect ≤ T^f(j)) then
        ℓ^i_F := ℓ^i_F ∪ {F_ij}
    end if
end for
ℓ^i_CR := ∅
for each F_ij ∈ ℓ^i_F do
    for each CR^k_ij ∈ F_ij do
        if (CR^k_ij ∈ ℓ^i_CRA ∧ O_ij(CR^k_ij) = 0) then
            ℓ^i_CR := ℓ^i_CR ∪ {CR^k_ij}
        else
            Statut(CR^k_ij) := inspection
        end if
    end for
end for
return ℓ^i_CR
end procedure
```

2.2.2 Méthodologie de construction du module de surveillance et du diagnostiqueur

Lorsque le module de surveillance a détecté l'apparition d'une défaillance, nous utilisons les algorithmes V.1 et V.2 pour lister tous les composants pouvant être à l'origine de la défaillance. Pour certaines défaillances, à l'issue de l'algorithme V.2 il reste des ambiguïtés de localisation sur l'identification des composants défectueux, il est alors nécessaire d'utiliser un diagnostiqueur.

Cette section présente la méthodologie générale de construction d'un diagnostiqueur pour les systèmes cyber-physiques. La particularité d'un système cyber-physique est qu'il interagit avec son environnement dans lequel il capte des données, les traite et adapte son fonctionnement dans une boucle de rétroaction pour contrôler ou interagir avec les processus physiques avec lesquels il est en interaction. Le comportement d'un CPS n'est donc pas prédéfini, l'incertitude est intrinsèque aux CPS.

Pour construire un diagnostiqueur, notre démarche est d'élaborer un usage particulier du CPS qui permet de solliciter ses différentes fonctions, en analysant le comportement normal (sans faute) et le comportement défaillant (prise en compte des défauts), nous pouvons ainsi construire le diagnostiqueur.

Nous présentons ci-dessous notre démarche qui est décomposée en trois étapes. Nous utilisons Matlab / Simulink / Stateflow pour modéliser et simuler le comportement du CPS et du diagnostiqueur.

1re étape : Modélisation du CPS sous Matlab / Simulink / Stateflow

1) Etude préliminaire du système :
   - Analyse générale :
     - Analyse fonctionnelle du besoin : construction du diagramme pieuvre.
     - Analyse fonctionnelle interne : construction du diagramme FAST.
     - Analyse architecturale : construction des diagrammes de définition des blocs.
   - Analyse des défaillances :
     - Analyse AMDEC : identification des causes des défaillances et de leurs criticités.
     - Analyse par arbres de défaillances : identification des combinaisons d'événements menant à la défaillance redoutée.
     - Calcul des coupes minimales : identification des composants les plus critiques.
2) Modélisation du comportement normal sous Matlab / Simulink / Stateflow :
   - Tout d'abord, un choix d'utilisation du système cyber-physique doit être construit : il doit permettre de solliciter toutes les fonctions dont des composants restent ambigus à l'issue de l'algorithme V.2. Ce cas d'usage doit permettre à l'équipe

de maintenance de réduire des ambiguïtés d'identification des composants défaillants.

- Puis il faut choisir jusqu'à quel niveau de détail la modélisation doit être effectuée. Par exemple pour un robot de téléprésence, pour le bloc de déplacement, on peut se limiter pour chacune de ses roues à sa vitesse de déplacement et à son angle de rotation. On ne modélise pas l'accélération, le rayon des roues, le moment d'inertie du robot, la masse du robot, etc. La modélisation informatique du CPS ne sera pas complète, ce n'est pas un jumeau numérique qui est construit, mais un modèle de simulation des principales variables continues et discrètes qui caractérisent les différents modes de fonctionnement du CPS.
- Représentation de la dynamique des variables du CPS : par des systèmes d'équations différentielles, des automates temporisés,ou des automates hybrides, etc., ainsi que par des variables continues et discrètes. Celles-ci sont manipulées dans des fonctions Matlab, des blocs Simulink et Stateflow.
- Représentation des capteurs et des actionneurs du CPS par des fonctions Matlab et des modèles Simulink et Stateflow.
- Représentation de la commande du CPS par exemple par un modèle Stateflow pour représenter le GRAFCET d'une commande.

3) Modélisation du comportement défaillant sous Matlab / Simulink / Stateflow.
   A partir des modes de défaillances résultants de l'analyse AMDEC, nous complétons les modèles des actionneurs, capteurs et procédés pour la prise en compte de défaillances :

   - Des blocs d'injection de défaut sont ajoutés dans nos modèles Matlab / Simulink / Statelow.
   - Nous modifions nos modèles sur les procédés, actionneurs et capteurs pour qu'ils prennent en compte les défauts injectés.

2e étape : Construction des matrices de signatures des défauts et des classes des modes de défaillance

Cette étape a pour finalité de construire des matrices de signatures des défauts et des classes d'équivalence de défaillances. Pour les construire, on procède par :

- Sélection de l'ensemble des variables caractérisant l'état de fonctionnement du CPS.
- Réalisation d'un grand jeu de simulation des différents modes de fonctionnement du CPS lorsqu'il est en fonctionnement normal (sans présence de défaut) et recueil des valeurs des variables caractéristiques.
- Réalisation d'un grand jeu de simulation du CPS avec injections de défauts pour les différents modes de fonctionnement du CPS et recueil des valeurs des variables caractéristiques.
- A partir des valeurs des variables caractéristiques en comportement normal et en

comportement défaillant pour tous les modes de fonctionnement du CPS, utilisation de méthodes d'apprentissage et de classification pour construire des matrices de signatures de défauts et des classes d'équivalence de défaillances.

- Si on constate que des classes de défaillances regroupent plusieurs défauts, on recherche de nouvelles variables observables qui permettent de les discriminer.
- S'il reste encore des ambiguïtés, construction d'un automate temporisé exprimant les temps caractéristiques de sollicitations des composants selon le scénario d'utilisation du CPS pour différencier des défauts ayant la même signature.

3e étape : Construction du diagnostiqueur

Pour construire le diagnostiqueur, on applique les méthodologies développées aux chapitres I et IV qui présentent les méthodes de diagnostic des systèmes à événements discrets temporisés et des systèmes dynamiques hybrides. On procède en trois phases.

1) Phase de détection :
   On définit un automate qui réalise le suivi de la commande du CPS et détecte l'apparition d'une défaillance à partir des variables observables en utilisant les résidus.
2) Phase de localisation :
   On complète l'automate de surveillance en utilisant les classes d'équivalence des défauts selon les valeurs des variables caractéristiques. Après l'apparition de la déviation du comportement courant par rapport au comportement nominal, on utilise les classes d'équivalence des défauts pour localiser un ensemble de défauts susceptibles d'être à l'origine de la défaillance.
3) Phase d'identification :
   On procède à l'isolation des défauts en utilisant les matrices de signatures et s'il y a besoin de lever des ambiguïtés (les variables observables conduisent à une classe d'équivalence contenant plusieurs défauts possibles), utilisation d'un automate temporisé pour comparer les dates de sollicitation des composants et leurs durées de fonctionnement avec celles attendues dans le fonctionnement normal du CPS.

2.3 Exemple applicatif de notre méthodologie de diagnostic de CPS : robot de téléprésence RobAIR

Le Robot RobAIR est un projet de l'Université Grenoble Alpes destiné à l'enseignement et à l'expérimentation de la robotique de service dans des environnements réels. Il a été conçu au sein d'une Fablab en collaboration avec plusieurs écoles d'ingénieurs du bassin grenoblois.

Ce robot est composé de différents sous-systèmes mécaniques, automatiques, électroniques, capteurs, embarqués et réseau qui lui permettent d'assurer des fonctionnalités de communication, de calcul et de contrôle. Il répond à la définition des systèmes cyber-physiques.

Le robot RobAIR est un robot de téléprésence qui peut être utilisé pour faire des visites à distance de musées, pour rompre la solitude des enfants malades ou pour leur permettre de participer à des cours à distance, pour permettre aux personnes âgées une immersion à distance dans leur famille, etc. Ce robot nécessite la présence humaine pour fonctionner, il n'est donc pas autonome. La figure V.4 résume les principales utilisations du RobAIR.

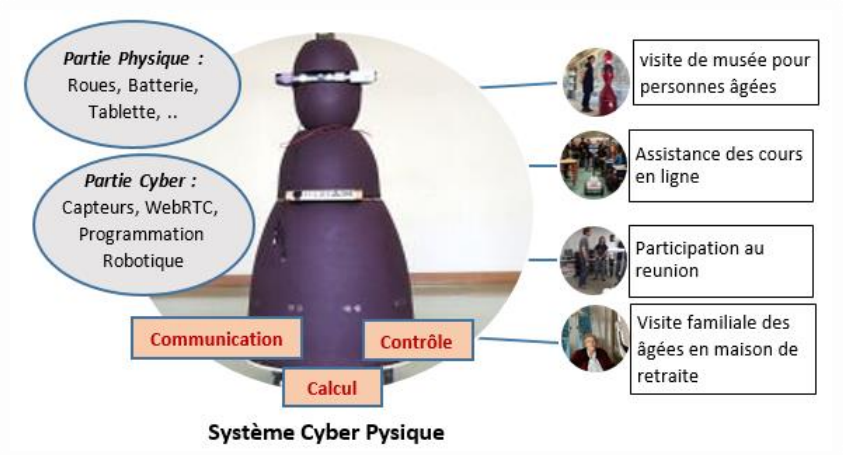


FIGURE V.4 – Domaines d'application du RobAIR [148], [175]

Le principe général de notre méthode de diagnostic de CPS appliquée au robot RobAIR est représenté sur la figure V.5.

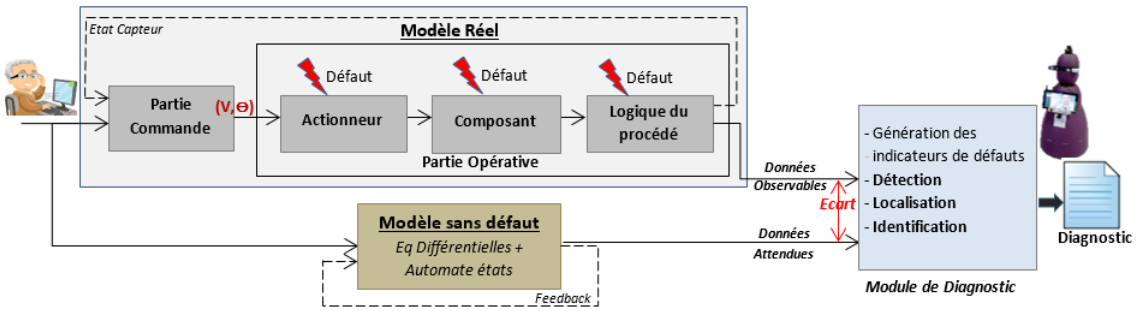


FIGURE V.5 – Système de diagnostic appliqué au CPS RobAIR [148], [175]

Nous présentons dans le reste de cette section seulement quelques étapes de notre modélisation du RobAIR et de la construction du module de diagnostic. Le lecteur peut se référer au rapport de stage de F. Ketchaou [175], au manuscrit de thèse de M. A. Haj Kacem [148] et aux différentes publications associées [176]-[179] pour avoir une présentation en détail de notre méthodologie de modélisation et de diagnostic de CPS et de son application au robot RobAIR.

2.3.1 Modélisation de RobAIR

2.3.1.1 Analyse du RobAIR

Le robot RobAIR comprend une partie contrôle et une partie opérationnelle. L'opérateur est l'entrée du système et la sortie du système correspond à l'exécution de la téléprésence. Les différentes technologies, leurs composants et leurs interactions avec le monde physique sont représentés sur la figure V.6.

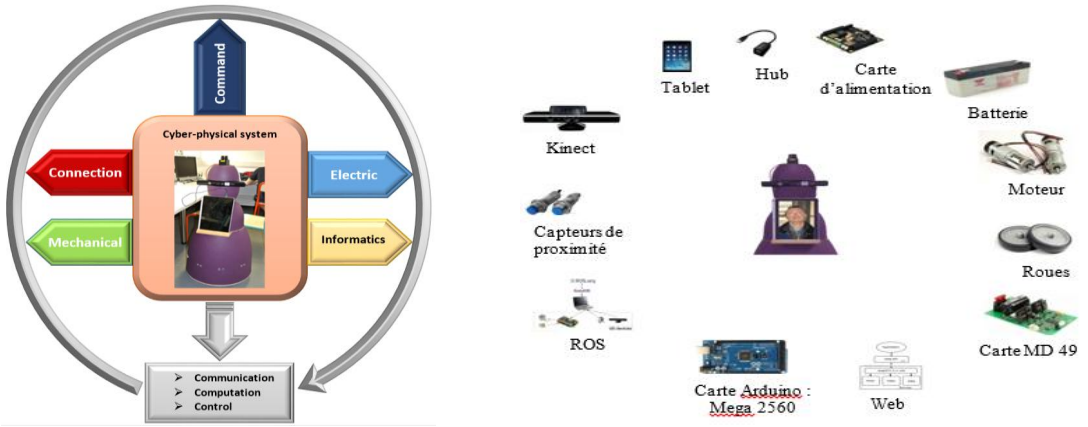


FIGURE V.6 – Technologies et composants du robot RobAIR [148], [175]

Une partie de l'analyse fonctionnelle du RobAIR est illustrée sur les figures V.7 et V.8.

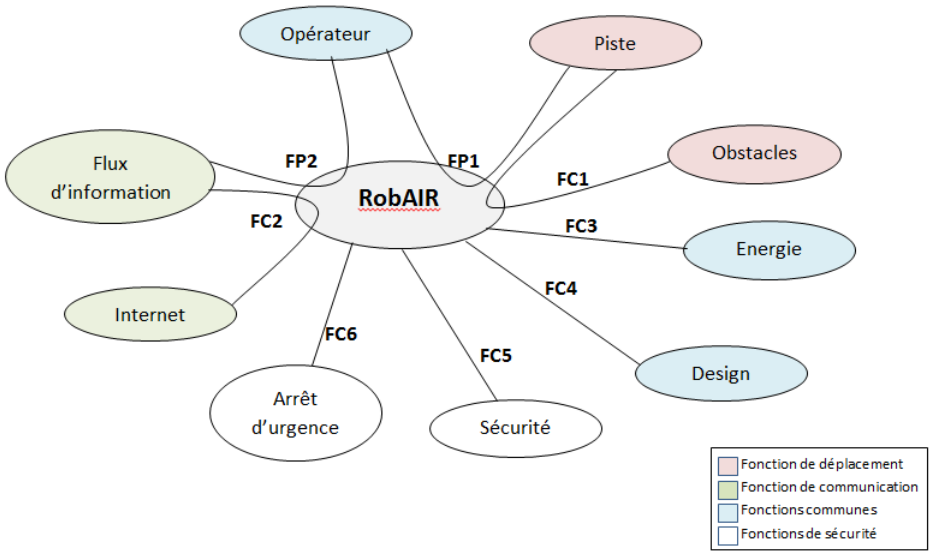


FIGURE V.7 – Diagramme de relations entre RobAIR et le monde physique [148], [175]

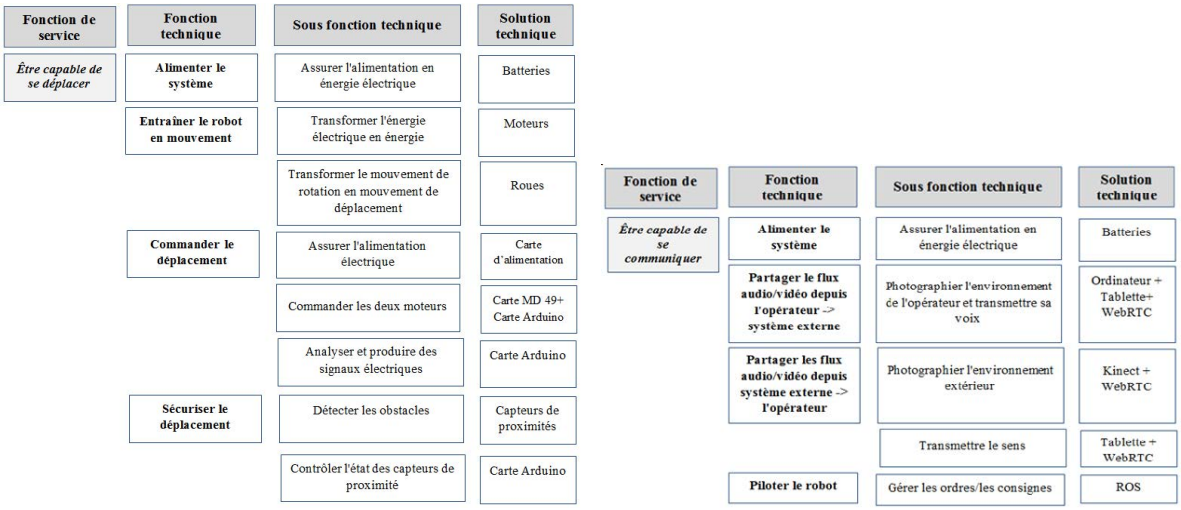


FIGURE V.8 – Diagrammes FAST pour les fonctions de service de déplacement et de communication [148], [175]

Une partie de la modélisation du RobAIR (structurelle, fonctionnelle et topologique) est illustrée sur les figures V.9, V.10 et V.11.

Une partie de l'analyse qualitative (AMDEC) est représentée par la figure V.12.

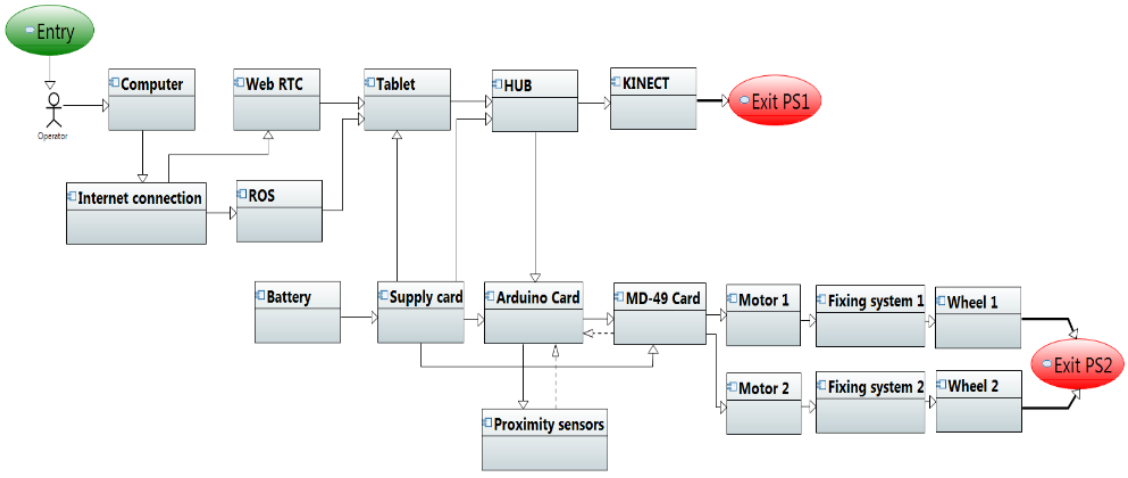


FIGURE V.9 – Modélisation structurelle du RobAIR [148]

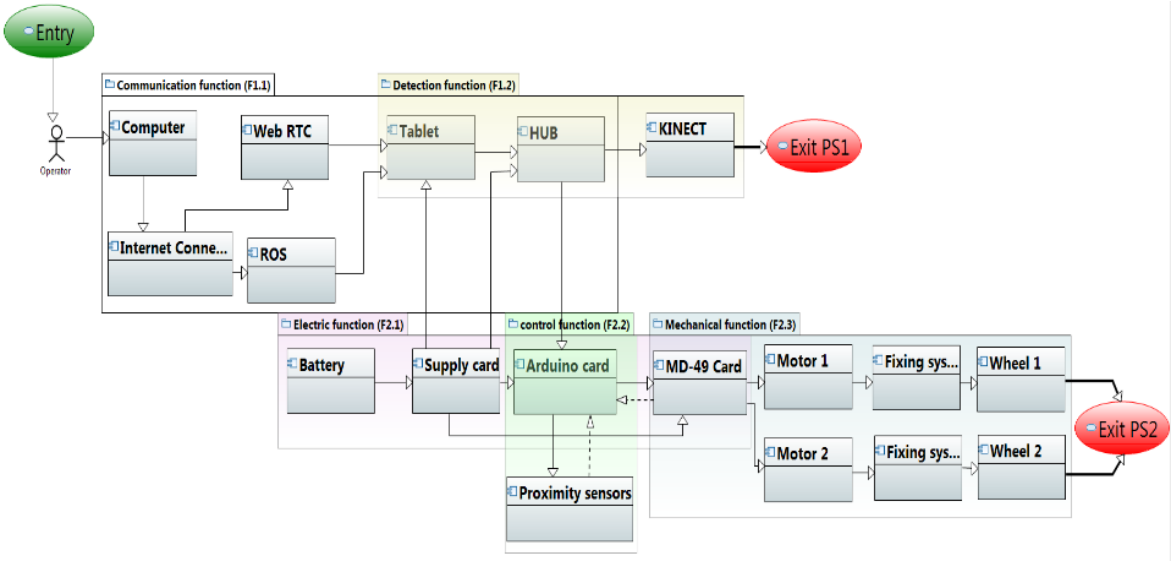


FIGURE V.10 – Modélisation fonctionnelle du RobAIR [148]

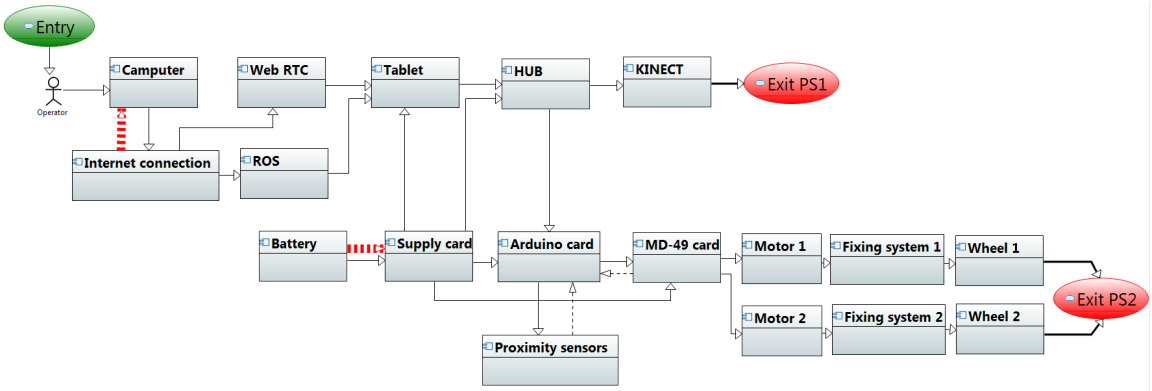


FIGURE V.11 – Modélisation topologique du RobAIR [148]

n°	Bloc organique	URL	Mode de défaillance	Fonction interne (fonction du bloc)	Cause de la défaillance	Effet local (interne au bloc)	Effet local (interne à l'URL)	Effet système	Lambda (10-5p/h)	Probabilité (P)	Gravité (G)	Détectabilité (D)	Criticité (RPN)
1	système de déplacement	roue	bande en caoutchouc endommagée	Transformer le mouvement de rotation en mouvement de déplacement	usure (caillou, clou, ,,)	énergie mécanique transmise ne correspond pas à la demande	Mauvais mouvement de la roue	Mauvaise direction du robot vitesse réduite voir blocage du robot Consommation énergétique importante	7,6	1	3	1	3
2		moyeu	Roue libre	Transformer le mouvement de rotation en mouvement de déplacement	siège du moyeu fortement usé (sous choc / fatigue encrassement corrosion)				2,5	1	5	2	10
		roue			Vis entièrement desserrée				3	1	5	1	5
		moteur			arbre cassée				3	1	5	2	10
3		moyeu	roue non centrée	Transformer le mouvement de rotation en mouvement de déplacement	siège du moyeu faiblement usé (sous choc / fatigue encrassement corrosion)				5	2	3	2	12
		roue			Vis partiellement desserrée			Robot en arrêt	5	2	3	1	6
4		moteur	Surchauffage du moteur	Transformer l'énergie électrique en énergie mécanique	blocage des roues --> Capteur de proximité non fonctionnel				2,28	3	3	1	9
5			moteur abimé	Transformer l'énergie électrique en énergie mécanique	court-circuit				4,56	2	4	2	16
6			Moteur non alimenté	Transformer l'énergie électrique en énergie mécanique	Détérioration du câble électrique			Robot en arrêt	1,14	1	4	2	8

FIGURE V.12 – Extrait de l'analyse AMDEC du RobAIR [148], [175]

2.3.1.2 Modélisation du comportement nominal de RobAIR

Nous définissons tout d'abord un cas d'usage du RobAIR, il est présenté par la figure V.13. La trajectoire du RobAIR se compose alors en différentes phases décrites par la figure V.14.

Notre principe de modélisation du RobAIR sous Matab /Simulink/Stateflow est décrit par la figure V.15.

La partie Process qui contient la dynamique de déplacement du robot et ses capteurs sont modélisés sous Simulink (figure V.16).

La partie Commande qui contient la séquence de contrôle du robot est décrite sous StateFlow (figure V.17).

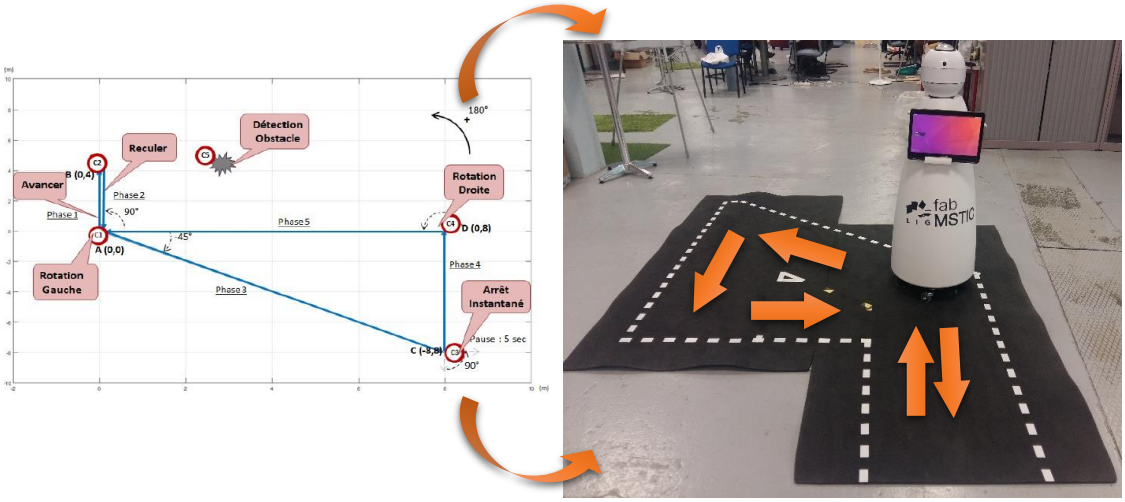


FIGURE V.13 – Cas d'usage du RobAIR [148], [175]

Phases	Distance (m)	Temps (s)	Mouvement Testé
Phase 1	4	2,67	Avance
Phase 2	4	2,67	Recule
Phase 3	11,31	7,54	Rotation sens indirecte
Pause	_	5,00	Arrêt instantané
Phase 4	8	5,33	Rotation sens directe
Phase 5	8	5,33	Retour position initial
Total	*35,31*	*28,54*	

FIGURE V.14 – Phases de trajectoire du RobAIR sur le cas d'usage [148], [175]

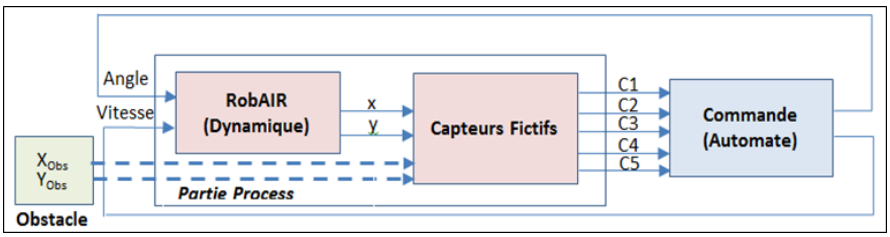


FIGURE V.15 – Principe de modélisation du RobAIR [148], [175]

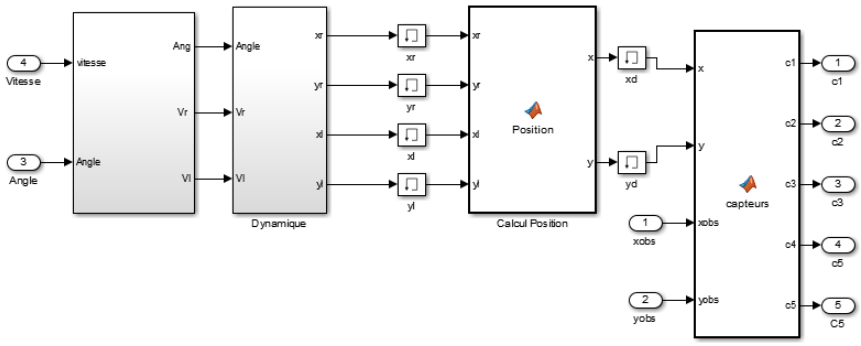


FIGURE V.16 – Modélisation sous Simulink de la partie Process du RobAIR [148], [175]

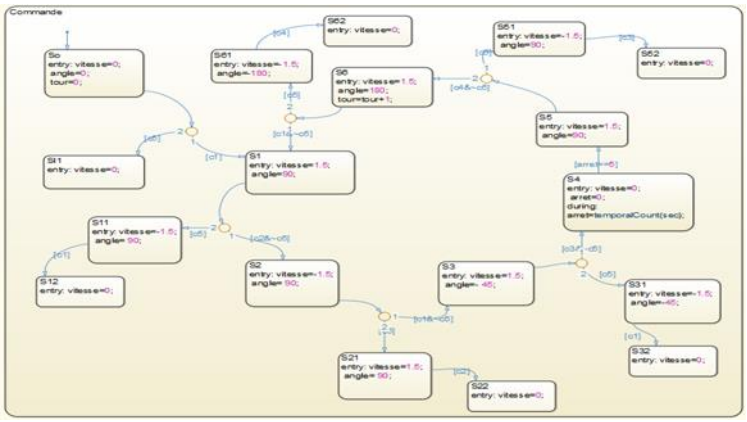


FIGURE V.17 – Extrait du modèle Stateflow de la commande du RobAIR [148], [175]

2.3.1.3 Modélisation du comportement en présence de défauts de RobAIR

La modélisation sous Matlab de l'injection d'un défaut sur la dynamique du mouvement du robot se fait par la définition d'un bloc Simulink. Ce bloc prend en entrée le numéro de défaillance et les variables de déplacement attendues (celles du fonctionnement normal) et retourne les variables de déplacement (Vd, Vg, Θ) obtenues en injectant le défaut.

Nous avons considéré des défaillances sur le système de déplacement, le bloc d'alimentation, le bloc de communication, le bloc de contrôle et sur les capteurs équipant le robot. La figure V.18 présente tous les modes de défaillance que nous avons considérés et leurs conséquences sur les variables observables qui sont la vitesse de la roue droite (Vd), la vitesse de la roue gauche (Vg) et l'angle de déplacement (Θ).

Le bloc Process du comportement nominal du RobAIR est alors modifié pour prendre en compte l'apparition aléatoire de défauts (figure V.19).

N°	Bloc	Mode de défaillance	Vd	Vg	Angle
1	système de déplacement	Bande droite en caoutchouc endommagée	diminue	Normal	faible
2		Bande gauche en caoutchouc endommagée	Normal	diminue	faible
3		Roue droite libre	augmente	Normal	faible
4		Roue gauche libre	Normal	augmente	faible
5		Roue droite non centrée	diminue	Normal	faible+
6		Roue gauche non centrée	Normal	diminue	faible+
7		Sur-chauffage du moteur droit	Nulle	Normal	fort
8		Sur-chauffage du moteur gauche	Normal	Nulle	fort
9		Moteur droit abimé	Nulle	Normal	fort
10		Moteur gauche abimé	Normal	Nulle	fort
11		Moteurs non alimentés (câble)	Nulle	Nulle	inchangé
12		Commande aléatoire (MD49/Arduino)	diminue	diminue	faible
13		Connexion perdue MD49	Nulle	Nulle	inchangé
14		Carte MD49 abimée	Nulle	Nulle	inchangé
15		Carte Arduino abimée	Nulle	Nulle	inchangé
16		Connexion perdue Hub-Arduino	Nulle	Nulle	inchangé
17		Connexion alimentation perdue Arduino-supply card	Nulle	Nulle	inchangé
18		Problème système d'exploitation	Nulle	Nulle	inchangé
19	Bloc d'alimentation	Batterie vide	Nulle	Nulle	Normal
20		Batterie faible	diminue	diminue	Normal
21		Carte d'alimentation abimée	Nulle	Nulle	inchangé
22		Connexion perdue : batterie-carte d'alimentation	Nulle	Nulle	inchangé
23	Bloc communication	Problème WebRTC	Normal	Normal	Normal
24		Tablette défaillante	Nulle	Nulle	inchangé
25		Hub Défaillant	Nulle	Nulle	inchangé
26	Bloc de contrôle	Kinect défaillant	Nulle	Nulle	inchangé
27	**Capteurs**	C1 reste éteint	Normal	Normal	Inchangé
28		C1 reste allumé	Normal	Normal	Normal
29		C2 reste éteint	Normal	Normal	Inchangé
30		C2 reste allumé	Normal	Normal	Normal
31		C3 reste éteint	Normal	Normal	Inchangé
32		C3 reste allumé	Normal	Normal	Normal
33		C4 reste éteint	Normal	Normal	Inchangé
34		C4 reste allumé	Normal	Normal	Normal
35		C5 reste allumé	Normal	Normal	Sens inverse

FIGURE V.18 – Caractérisation des modes de défaillances du RobAIR [148], [175]

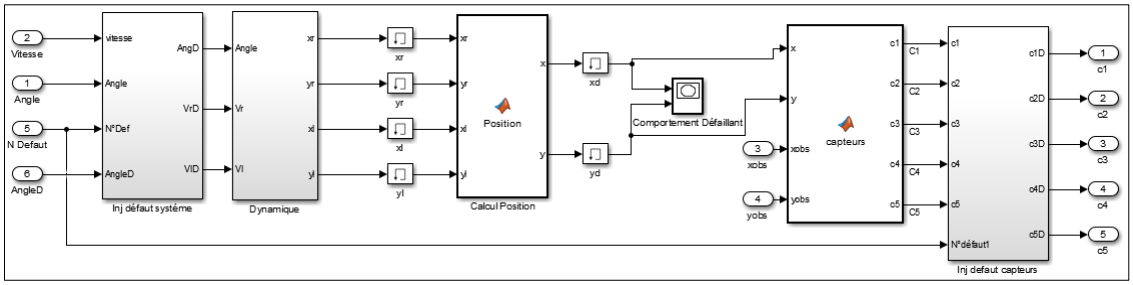


FIGURE V.19 – Bloc Process avec prise en compte de l'injection de défaillances [148], [175]

2.3.2 Construction du diagnostiqueur

La construction du diagnostiqueur se déroule en trois étapes : tout d'abord la construction des matrices de signatures de défaillances et des classes d'équivalence des défaillances (figure V.20), puis pour lever des ambiguïtés, la construction d'un automate temporisé à base des états de la partie commande qui avance en parallèle dans le temps avec la partie commande (figure V.21) et finalement la construction d'un bloc de diagnostic sous Stateflow qui permet d'identifier la défaillance (figure V.22).

La représentation sous Matlab / Simulink / Stateflow de RobAIR avec des injections de défaillances et le module de diagnostic est présentée sur la figure V.23.

N° Classe	1er filtrage (Vd, Vg, Θ)	2éme filtrage (MD49, Alim, K, T)	Défauts
1	(D, N, f)	(7, 1, 1, 1)	1
2	(N, D, f)	(7, 1, 1, 1)	2
3	(D, N, f+)	(7, 1, 1, 1)	5
4	(N, D, f+)	(7, 1, 1, 1)	6
5	(0, N, f)	(2, 1, 1, 1)	3
6	(N, 0, f)	(3, 1, 1, 1)	4
7	(0, N, F)	(2, 1, 1, 1)	7
8		(4, 1, 1, 1)	9
9	(N, 0, F)	(3, 1, 1, 1)	8
10		(4, 1, 1, 1)	10
11	(D, D, N)	(0, 1, 1, 1)	20
12	(D, D, f)	(0, 1, 1, 1)	12
13	(0, 0, I)	(0, 1, 1, 1)	11/17/15/18
14		(7, 1, 1, 1)	13
15		(2, 1, 1, 1)	14
16		(0, 1, 1, 1)	16
17		(0, 0, 0, 0)	19/21/22
18		(0, 1, 1, 0)	24
19		(0, 1, 0, 1)	25/26
20	(N, N)	(0, 1, 0, 1)	23
21	(N, N)	(0, 1, 1, 1)	27/29/31/33 30/28/32/34 35

FIGURE V.20 – Classe d'équivalence selon les variables dynamiques et les états des bits de la carte MD49 [148], [175]

Phase	Durée (sec)	Durée cumulée (sec) (depuis le début du cycle)	Intervalle de Tolérance (sec)
Phase 1	2,67	2,67	2 – 3
Phase 2	2,67	5,34	2 – 3
Phase 3	7,54	12,88	12 – 13
Pause	5,00	17,88	17 – 18
Phase 4	5,33	23,21	23 – 24
Phase 5	5,33	28,54	28 – 29

FIGURE V.21 – Temporisation des phases de commande du RobAIR [148], [175]

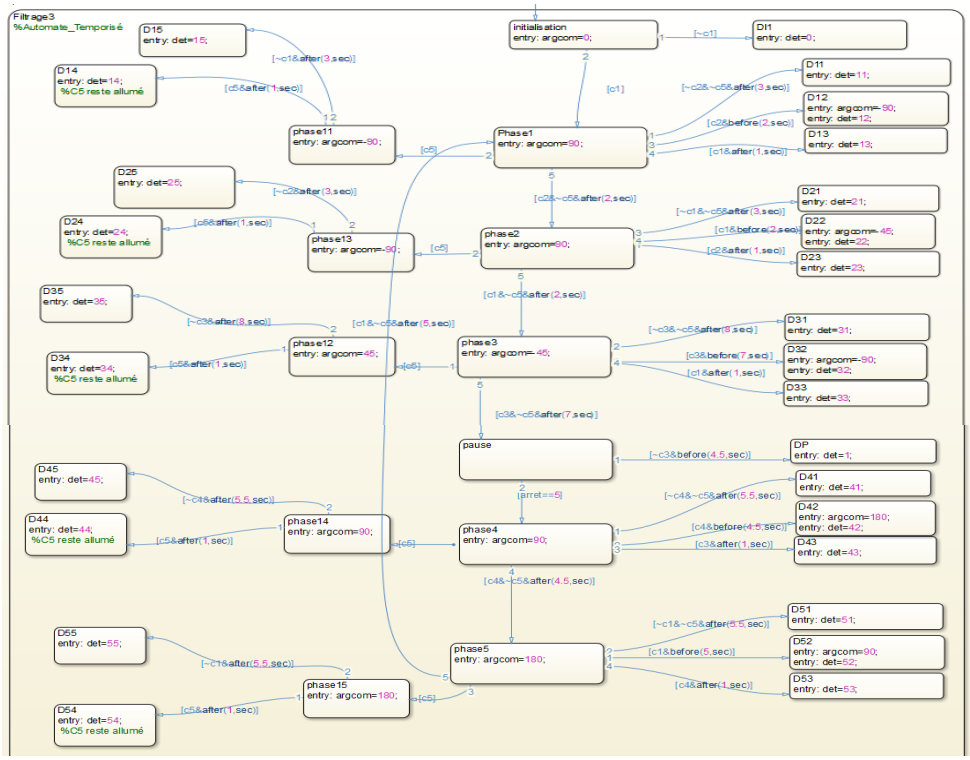


FIGURE V.22 – Extrait du bloc Stateflow d'identification des défaillances [148], [175]

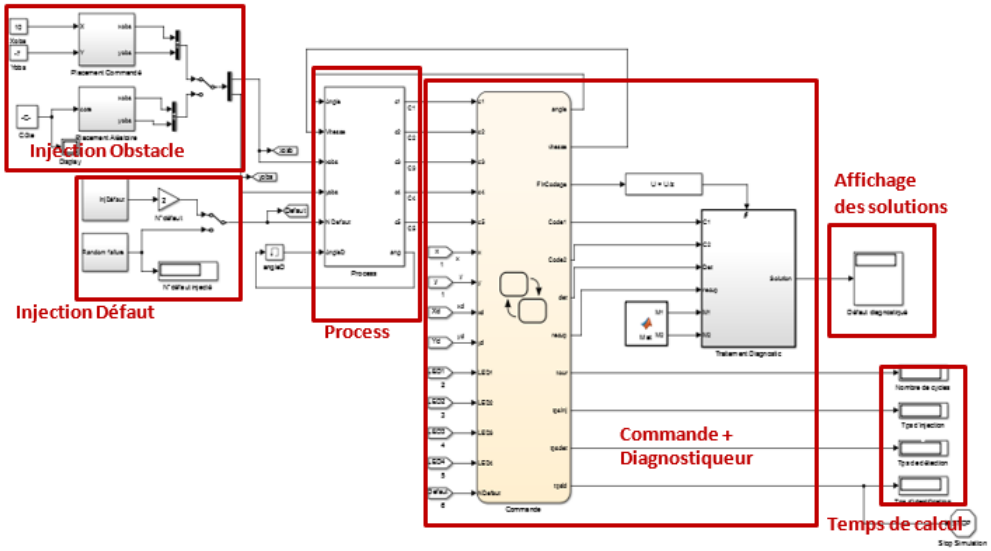


FIGURE V.23 – Modélisation de RobAIR avec injection de défaillances et son diagnostiqueur [148], [175]

3 Conclusion et perspectives sur la modélisation et le diagnostic de systèmes cyber-physiques

Dans une première partie de ce chapitre, nous avons proposé une méthodologie de modélisation des systèmes cyber-physiques à base de connaissances structurelle, fonctionnelle, topologique et comportementale à partir desquelles nous pouvons élaborer un modèle de simulation de ce système cyber-physique en fonctionnement nominal. Puis par des analyses de défaillances (AMDEC, arbres de défaillances, etc.), nous complétons notre modèle de simulation du CPS afin de prendre en compte le comportement du CPS en présence de défauts.

Dans la seconde partie de ce chapitre, nous avons proposé une méthodologie de diagnostic des défaillances. Nous avons tout d'abord proposé des algorithmes pour aider l'équipe de maintenance corrective à identifier le ou les composants sources d'une défaillance lorsque l'une des fonctionnalités du CPS n'est plus assurée : nous utilisons à bon escient les connaissances structurelles, fonctionnelles, topologiques et comportementales pour identifier le sous-système et ses composants sollicités au moment de la défaillance. Dans le cas où dans la liste des composants susceptible à l'origine de la défaillance il y a des composants non observables, nous proposons l'usage d'un automate temporisé pour lever des ambiguïtés de localisation de composants non observables. Dans une seconde partie de nos travaux de recherche sur le diagnostic de défaillances dans les CPS, nous avons développé une méthodologie de construction de diagnostiqueurs pour les CPS. Tout d'abord, nous élaborons un cas d'usage du CPS qui permet de solliciter l'ensemble des composants non observables

sur lesquels apparaissent des ambiguïtés de localisation de défaillances. Puis par une approche mixte qui se base d'une part sur une méthode à base de modèle physique pour la modélisation du comportement dynamique du système et sur une méthode à base de données par de l'apprentissage et de la classification d'autre part, nous construisons les classes d'équivalences des défaillances et un automate temporisé de suivi du CPS, qui composent notre diagnostiqueur.

Ce travail de recherche peut être poursuivi par plusieurs perspectives de recherche. Tout d'abord une étude en amont de la diagnosticabilité des systèmes cyber-physiques [180]. Nous avons en effet constaté la problématique d'ambiguïté d'identification de composants non observables. Une étude de la diagnosticabilité d'un CPS nous permettrait de connaitre l'ensemble des composants non observables dont on ne peut lever l'ambiguïté d'identification de défauts. Pour résoudre au mieux ce problème d'ambiguïté, l'ajout de points d'observation à travers des capteurs supplémentaires permettrait de distinguer des composants qui appartiennent aux classes d'équivalence renfermant une ambiguïté d'identification et de retrouver en conséquence exactement le défaut concerné.

Une seconde perspective est le développement d'une méthode d'aide à la maintenance corrective de CPS qui n'est plus associée à un cas d'usage particulier du CPS. En effet le comportement d'un CPS n'est pas prédéfini puisqu'il s'adapte à l'environnement physique avec lequel il interagit. Pour cela, un axe de travail serait de développer une méthode de génération en temps réel d'indicateurs de santé des composants du CPS et d'utiliser des méthodes d'apprentissage, de classification et de pronostic pour réaliser l'étude de la fiabilité des composants du CPS et du système dans sa globalité. Le pronostic de défaillances pourrait être réalisé par le jumeau numérique (digital twin) du CPS. Son modèle serait évolutif : la base de connaissance de l'état de santé de ses composants serait mise à jour régulièrement par la communication en temps réel des indicateurs d'états de santé des composants du CPS permettant le suivi de leurs dégradations.

Une perspective à plus long terme de nos travaux de recherche est la détection d'anomalie d'usage du CPS [181], [182]. Les systèmes cyber-physiques étant connectés au réseau sont vulnérables à des cyberattaques qui peuvent détourner le fonctionnement « classique » du CPS pour provoquer une défaillance. Il est alors nécessaire de développer des méthodes qui permettent de détecter des violations du comportement standard réalisé par le CPS dans ses différents cas d'usage.

Chapitre VI

Diagnostic de dérives temporelles dans des graphes d'événements temporisés

1 Présentation générale : contexte, état de l'art, positionnement, originalité

1.1 Contexte

Le contexte de ce travail de recherche est le diagnostic de fautes temporelles pour des systèmes dynamiques non autonomes, c'est-à-dire dont l'évolution est conditionnée par des événements externes et par le temps. On modélise de tels systèmes par des réseaux de Petri P-temporisés où une temporisation est associée aux places. Nous nous intéressons plus particulièrement à une sous-classe des réseaux de Petri P-temporisé : les graphes d'événements temporisés (GET) (*Timed Event Graphs*) [183], [184].

Des systèmes de production tels que des lignes d'assemblage, des ateliers flexibles, des chaînes d'approvisionnement, peuvent en effet être modélisés par de tels réseaux de Petri P-temporisés non autonomes où des contraintes de temps doivent impérativement être respectées. Les réseaux de Petri temporisés permettent également de représenter des synchronisations entre ressources, des durées de traitement et des temps de transport.

On s'intéresse à diagnostiquer des violations des contraintes temporelles pour ces systèmes qui conduisent alors à des dysfonctionnements qui peuvent entrainer des pannes ou à une détérioration de la qualité des biens ou des services. Les fautes temporelles auxquelles nous nous intéressons correspondent à des retards ou dérives sur les temporisations associées aux places du réseau de Petri. Il est nécessaire de détecter au plus tôt l'apparition d'une dérive temporelle, puis de localiser la place source de cette dérive temporelle, et finalement identifier le décalage temporel.

1.2 Etat de l'art, positionnement, originalité

Dans le cadre du diagnostic de dérives temporelles dans les systèmes à événements discrets temporisés, seul le modèle nominal est connu, et l'observation du système réel est partielle (seulement un sous-ensemble des événements est observable) et fourni des

séquences d'événements datés. A partir de la connaissance du modèle nominal (comportement attendu) et de la surveillance du système réel (séquences d'événements observables datés), le module de diagnostic doit réaliser les tâches de détection, localisation et identification au plus tôt d'une dérive temporelle si elle se produit. Les approches existantes du diagnostic temporel se classent essentiellement en deux catégories :

— La première catégorie regroupe toutes les méthodes de diagnostic qui utilise la reconnaissance de motifs temporels pour la détection de fautes temporelles, puis analyse les motifs temporels reconnus pour la localisation et l'identification des fautes temporelles. Ces méthodes ne s'appuient pas sur la formalisation du système nominal sous la forme d'un SED temporisé, mais utilisent le formalisme de signature temporelle causale (*Causal Temporal Signature*) [86], ou de motif de condition (*condition template*) [87] ou de chronique (*chronicle*) [88], [89]. Ces méthodes n'ont pas de restriction spécifique sur le placement de capteurs dans le système à diagnostiquer, cela dépend de la surveillance appliquée au système à diagnostiquer. Ces capteurs fournissent les événements datés observables utilisés dans les motifs temporels.

— La seconde catégorie utilise des méthodes algébriques. Le système de production modélisé par un graphe d'événements temporisé peut être représenté dans une algèbre des dioïdes particulière, l'algèbre (max,+). La théorie de l'algèbre (max,+) est alors combinée avec des méthodes d'estimation paramétrique [185], [186] ou avec la théorie de la résiduation appliquée aux dioïdes [187], [188] pour réaliser le diagnostic de fautes temporelles. Ces méthodes algébriques considèrent comme observables uniquement les transitions en entrées et en sorties du réseau de Petri.

Nos travaux se différencient de ceux de la littérature par une approche algorithmique. Nous proposons la simulation conduite par les événements à la place d'une approche algébrique pour obtenir le comportement attendu du système à partir des événements d'entrées datés et nous avons défini des algorithmes qui utilisent les séquences d'événements observables datés provenant de la surveillance du système réel et notre simulateur de graphes d'événements temporisés pour réaliser la détection, la localisation et l'identification de fautes temporelles. Dans notre approche, nous supposons qu'un sous-ensemble des transitions internes du GET est observable. La diagnosticabilité du système dépend en effet de l'observabilité du système. Notre approche considère des durées fixes pour les temps de séjour dans les places.

2 Synthèse des travaux développés

2.1 Modélisation d'un système de production par un graphe d'événement temporisé

Nous considérons des systèmes de production qui sont modélisables par des *graphes d'événements temporisés* (GET). Un graphe d'événements temporisés est un réseau de Petri temporisé qui associe à chaque place une constance positive (RdP P-temporisé),

toute place a exactement une transition en amont et une transition en aval [189] et le poids de chaque arc est 1.

Nous modélisons un GET dans notre méthodologie de diagnostic de fautes temporelles comme un réseau de Petri P-temporisé partiellement observable décrit par le triplet $\mathcal{T} = \langle N, \delta, X^o \rangle$ où $N = (P, T, I, O, M_o)$ est un réseau de Petri ordinaire, $\delta : P \to \mathbb{N}$, la fonction de durée associe un entier positif ou nul $\delta(p_i)$ à chaque place p_i. Dans un GET partiellement observable, nous considérons que les transitions en entrées et en sorties sont observables et qu'un sous-ensemble, possiblement vide, des transitions internes peuvent être observées. X^o représente l'ensemble des transitions internes observables. Le réseau de Petri N est défini par :

- $P = \{p_1, p_2, \ldots\}$, un ensemble fini de places.
- $T = U \cup X \cup Y$, un ensemble fini de transitions où $U = \{u_1, u_2, \ldots\}$ est le sous-ensemble des transitions en entrées, $X = \{x_1, x_2, \ldots\}$ est le sous-ensemble des transitions internes avec $X^o \subseteq X$ et $Y = \{y_1, y_2, \ldots\}$ est le sous-ensemble des transitions en sorties.
- I est l'application des places en amont d'une transition qui fournit toutes les places connectées aux arcs en entrée d'une transition.
- O est l'application des places en aval d'une transition qui fournit toutes les places connectées aux arcs en sorties d'une transition.
- M_0 est le marquage initial des jetons dans le réseau de Petri.

Notre représentation d'un système de production par un graphe d'événements temporisé est la suivante :

- Un *jeton* correspond à un produit / composant ou à une ressource disponible.
- Une *place* p_i correspond à une opération ou un traitement particulier dans le système de production ou à la disponibilité d'une ressource requise pour accomplir une opération.
- Chaque place p_i est associée avec un entier positif ou nul $\delta(p_i)$ qui représente le temps de séjour ou temporisation qu'un jeton doit passer dans cette place p_i, avant qu'il devienne disponible pour son unique transition en sortie.
- Dans les GETs, le franchissement d'une transition a une durée nulle. Si une transition est franchie au temps t, le jeton déposé sur la place en sortie p_i sera indisponible jusqu'à l'instant $t + \delta(p_i)$.

Dans notre travail, nous supposons que chaque place contient au plus un jeton dans le marquage initial, cependant nos résultats peuvent être étendus pour prendre plus qu'un jeton par place dans le marquage initial.

2.2 Exemple illustratif

Nous illustrerons notre méthodologie de diagnostic de fautes temporelles sur le graphe d'événements temporisés représenté par la figure VI.1. Ce GET peut modéliser une ligne

de production de lots de wafer (plaque ronde de matériau semi-conducteur regroupant des circuits intégrés) dans l'industrie microélectronique [187] ou une ligne de remanufacturing de toners d'imprimantes [190].

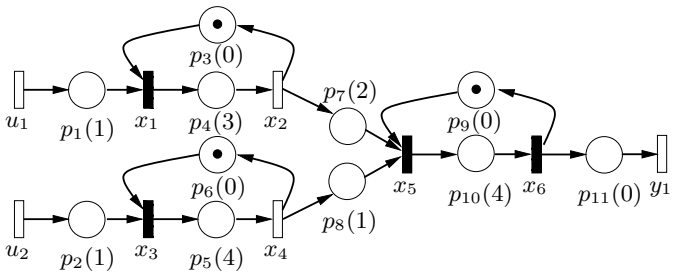


FIGURE VI.1 – Représentation d'une chaîne de production par un GET

Sur notre représentation graphique du GET, une transition observable est représentée par un rectangle non rempli, une transition non observable est représentée par un rectangle plein. A coté de chaque place, nous avons mentionné son temps de séjour entre parenthèses.

2.3 Présentation de notre méthodologie de diagnostic de fautes temporelles

Nous considérons qu'il n'y a qu'une seule faute temporelle permanente dans le système à événements temporisés qui modifie la durée de séjour d'une place. La figure VI.2 présente notre méthodologie de diagnostic. Nous utilisons les notations suivantes :

— Ω^o_U représente l'entrée du système, c'est la collection des séquences de franchissements des transitions en entrées U du GET.

— $\Omega^o_{X^o}$ et Ω^o_Y représentent les collections des séquences de franchissements des transitions internes observables X^o et de sorties Y du GET mesurées lors du fonctionnement du système réel.

— $\Omega^s_{X^o}$ et Ω^s_Y représentent les collections des séquences de franchissements des transitions internes observables X^o et de sorties Y du GET qui sont attendues, ces données proviennent de la simulation du GET.

Notre méthodologie de diagnostic se compose en trois étapes :

1. Tout d'abord, nous comparons les séquences de franchissement des transitions mesurées ($\Omega^o_{X^o}$, Ω^o_Y) avec les séquences de franchissement attendues ($\Omega^s_{X^o}$, Ω^s_Y) : la comparaison entre les dates de franchissement des transitions mesurées et attendues permet la détection d'une dérive temporelle. Les paragraphes 2.3.1 et 2.3.2 présentent respectivement l'algorithme VI.1 de simulation d'un GET et l'algorithme VI.2 de détection d'une dérive temporelle qui retourne l'ensemble des transitions T^f dont les dates de franchissements ne sont celles attendues.
2. Puis, en utilisant la topologie du réseau de Petri, nous obtenons une première approximation de la place source de la dérive temporelle. Le paragraphe 2.3.3 présente

l'algorithme VI.3 qui retourne l'ensemble des places candidates P^f pour être la source de la dérive temporelle.

3. Finalement, nous procédons à un raffinement des places candidates en appliquant des règles de réduction et en résolvant des équations symboliques dans Matlab. Le paragraphe 2.3.4 présente l'algorithme VI.4 d'identification qui raffine la localisation de la place source de la dérive temporelle, P^r.

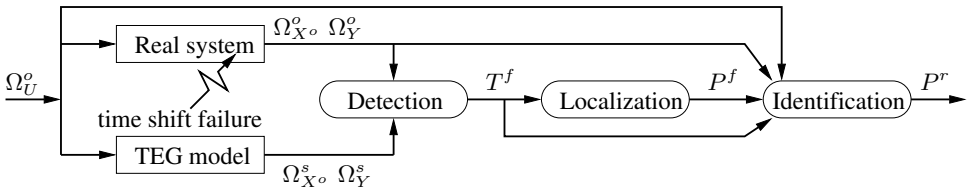


FIGURE VI.2 – Structure de la méthode de diagnostic de fautes temporelles

2.3.1 Simulation d'un graphe d'événements temporisés

Il existe différents logiciels pour la simulation de réseaux de Petri temporisé, nous avons choisi de développer notre propre algorithme de simulation de GET car celui-ci nous permet de paramétrer l'injection d'une faute temporelle dans le GET, nous utiliserons cette fonctionnalité dans notre algorithme d'identification. De plus, en ayant notre propre logiciel de simulation de GET, nous pourrons plus facilement faire évoluer nos travaux pour la prise en compte de l'incertain (le temps de séjour d'une place sera décrit par un intervalle). Le principe de simulation du graphe d'événements temporisés que nous avons utilisé est la simulation conduite par les événements. L'algorithme VI.1 présente notre méthode de simulation des GET.

Nous présentons ci-dessous les structures de données que nous utilisons dans cet algorithme. Dans notre article [190], nous détaillons ces structures de données et l'algorithme de simulation de GET.

Définition 2 (File d'attente des temps d'arrivées des jetons dans les places) *Nous associons à chaque place p une file d'attente, notée $p.queueTokens$, qui mémorise dans un ordre chronologique les temps d'arrivées des jetons dans la place p. La file d'attente vide est représentée par $\langle\rangle$. Soit θ la date d'arrivée la plus récente d'un jeton dans une place p. Nous présentons par l'instruction $p.queueTokens.push(\theta)$ l'ajout du temps d'arrivée θ en queue de la file d'attente. L'instruction $\theta := p.queueTokens.pop()$ extrait le jeton le plus ancien de la tête de la file d'attente et retourne dans θ la date d'arrivée du jeton qui était le plus ancien de la file. Le prédicat $p.queueToken.isEmpty()$ teste si la file d'attente est vide.*

2.3.1.1 Franchissement d'une transition

Nous avons défini deux fonctions ISENABLED(t) et FIRE(t) qui permettent la simulation

du franchissement d'une transition t. isEnabled(t) permet de savoir si la transition est franchissable étant donné l'état actuel du marquage du GET. Fire(t) retourne le temps du franchissement de la transition et met à jour le marquage du GET. Nos algorithmes de détection, localisation et identification de fautes temporelles vont analyser les franchissements des transitions observables afin de fournir un diagnostic, nous avons donc besoin de mémoriser l'historique des franchissements de chaque transition t dans une séquence temporisée ω_t. La définition 3 présente cette structure de données.

Définition 3 (Séquence des temps de franchissements d'une transition)
Une séquence des temps de franchissement d'une transition t (de type d'entrée ou interne ou de sortie d'un GET) est représentée par une séquence temporisée $\omega_t = [\theta_t^1, \theta_t^2, \ldots]$ *où* θ_t^k *est la date du k-ième franchissement de la transition t. La notation* $\omega_t(k)$ *représente la date du k-ième franchissement de la transition t (i.e.* θ_t^k*). Une séquence vide des temps de franchissement d'une transition est représentée par* []. $|\omega_t|$ *représente le nombre d'élements de la séquence* ω_t.

2.3.1.2 Collection de séquences de franchissements de transitions.

Une *collection* est une structure de données qui regroupe plusieurs données de même type. Dans nos algorithmes, nous manipulons des séquences de temps de franchissements de transitions. Nous représentons respectivement par Ω_U, Ω_X et Ω_Y les collections des séquences de temps de franchissements des transitions en entrées U, des transitions internes X et des transitions en sorties Y.

Nous représenterons une collection vide par $\emptyset$. La taille d'une collection est représentée par $|\Omega|$. A la structure de donnée de *collection* sont associées plusieurs opérations que nous utilisons dans nos algorithmes telles que :

— *add(ω_t)* pour ajouter la séquence de temps de franchissements ω_t d'une transition t à la collection ;

— *get(t)* pour retourner la séquence de temps de franchissements de la transition t présente dans la collection.

2.3.1.3 File d'attente de transitions franchissables

La structure de donnée principale que nous utilisons dans l'algorithme VI.1 de simulation de GET est une file d'attente Q de transitions franchissables. Cette file d'attente mémorise chronologiquement les transitions franchissables à traiter par l'algorithme de simulation et a la forme suivante :

$$Q = \langle [\theta^1, \langle t_i, t_j, \ldots \rangle], \ldots, [\theta^k, \langle t_l, t_m, \ldots \rangle], \ldots \rangle$$

Dans Q, à chaque date θ^k est associée une file de transitions franchissables $R = \langle t_l, t_m, \ldots \rangle$ dont le temps de franchissement est θ^k. L'instruction $Q.push(\theta^k, t)$ utilisée dans notre algorithme est définie ainsi : S'il existe une paire $[\theta^k, R]$ dans Q alors cette instruction exécute $R.push(t)$ sinon elle insère une nouvelle paire $[\theta^k, \langle t \rangle]$ dans la file Q à sa position

correcte d'insertion. L'instruction $[\theta, R] := Q.pop()$ dans notre algorithme retire la première séquence de transitions franchissables de Q.

Le fonctionnement de l'algorithme de simulation de GET est le suivant :

— Tout d'abord, il y a l'initialisation de la file d'attente Q à partir des séquences de transitions en entrées Ω^o_U qui sont connues et du marquage initial du GET. Cette phase d'initialisation permet d'insérer dans Q toutes les séquences de franchissements des transitions en entrée U et d'identifier toutes les autres transitions qui sont franchissables en raison du marquage initial du GET.

— Le corps de l'algorithme est une boucle de traitement des transitions franchissables : dans un ordre chronologique, toutes les transitions de Q sont traitées, les jetons transitent ainsi dans les places en respectant leurs durées de séjour, et de nouvelles transitions franchissables sont identifiées et insérées dans Q.

— L'algorithme de simulation s'arrête lorsque les derniers jetons en entrées du GET auront été traités et transités jusqu'à des transitions de sorties. Pendant toute la boucle de traitement des transitions franchissables, les franchissements des transitions internes observables X^O et de sorties Y simulés par notre algorithme sont mémorisés dans les collections Ω^s_X, Ω^s_Y.

2.3.2 Détection de dérives temporelles dans un GET

L'algorithme VI.2 effectue la détection d'une dérive temporelle comme suit : les données en entrée de l'algorithme sont $\Omega^o_{X^o}$, Ω^o_Y (données observées), $\Omega^s_{X^o}$ et Ω^s_Y (données simulées) qui représentent les collections des séquences de franchissements des transitions internes et de sorties. Les données observées sont issues des capteurs placés sur le système réel et les données simulées sont obtenues par simulation (Algorithme VI.1) du modèle du GET.

La détection d'une dérive temporelle est obtenue en comparant la séquence observée des temps de franchissements de chaque transition $t_i \in X^o \cup Y$, représentée par $\omega^o_{t_i}$ avec la séquence simulée des temps des franchissements de t_i représentée par $\omega^s_{t_i}$ (LIGNES 2-13). Une dérive temporelle est détectée dès qu'il existe une transition t_i telle que pour son k-ième franchissement, son temps de franchissement observé est différent de son temps de franchissement simulé (LINE 6). L'algorithme VI.2 retourne T^f, l'ensemble des transitions pour lesquelles il existe des différences entre les séquences de temps de franchissements observées et simulées.

2.3.2.1 Exemple de détection d'une dérive temporelle

Nous considérons le GET représenté par la figure VI.1. Nous supposons que les séquences de franchissements des transitions en entrées sont les suivantes : $\omega^o_{u_1} = \omega^o_{u_2} = [1, 2, 3, 4, 5, 6, 7]$.

Les séquences de franchissements des transitions internes observables $X^o = \{x_2, x_4\}$ et

Algorithme VI.1 Simulation conduite par les événements d'un GET

procedure SIMULATIONTEG
Input: $\mathcal{T}$: a TEG
Input: Ω_U^o : collection of *observed* sequences of firing times of the input transitions U
Outputs Ω_X^s, Ω_Y^s : collections of *expected* sequences of firing times of the internal transitions X and output transitions Y
Ensure: simulation of the TEG

```
1: Ω^s_X :=InitializationCollection(X) {Init. of the coll. of seq. of firing times of internal trans.}
2: Ω^s_Y :=InitializationCollection(Y) {Init. of the coll. of seq. of firing times of output trans.}
3: Q := ⟨⟩ {Initialization of the priority queue of the enabled transitions Q}
4: for each u ∈ U do
5:     ω^o_u := Ω^o_U.get(u)
6:     for k ← 1 to |ω^o_u| do
7:         θ^k_u := ω^o_u(k)
8:         Q.push(θ^k_u,u)
9:     end for
10: end for
11: for each p ∈ P do {Initialization of the priority queues of tokens associated to places}
12:     p.queueTokens := ⟨⟩
13:     if (p.isMarked = True) then
14:         p.queueTokens.push(p.duration)
15:         t := p.outputTransition
16:         if (isEnabled(t)) then
17:             θ := Fire(t)
18:             Q.push(θ,t) {Update of Q with the initial enabled transitions}
19:         end if
20:     end if
21: end for
22: while (not Q.isEmpty()) do {Main loop of simulation}
23:     [θ, R] := Q.pop()
24:     {θ is the current simulation time,
25:      R the queue of enabled transitions to process}
26:     while (not R.isEmpty()) do
27:         v := R.pop()
28:         if ((v.type = INTERNAL) ∨ (v.type = OUTPUT)) then
29:             ω^s_v.add(θ)
30:         end if
31:         if ((v.type = INPUT ∨ (v.type = INTERNAL)) then
32:             for each p ∈ v.successors do
33:                 p.queueTokens.push(θ + p.duration)
34:                 t := p.outputTransition
35:                 if (isEnabled(t)) then
36:                     θ_f := Fire(t)
37:                     Q.push(θ_f,t)
38:                 end if
39:             end for
40:         end if
41:     end while
42: end while
43: return Ω^s_X, Ω^s_Y
end procedure
```

Algorithme VI.2 Détection de dérives temporelles dans un GET

procedure FAILUREDETECTION
Input: $\mathcal{T}$: a TEG
Inputs: $\Omega^o_{X^o}$, Ω^o_Y : collection of observed sequences of firings times of sensored internal transitions X^o and output transitions Y
Inputs: $\Omega^s_{X^o}$, Ω^s_Y : collection of expected sequences of firings times of sensored internal transitions X^o and output transitions Y
Output: $failureDetected$: Boolean value expressing whether a failure is detected
Output: T^f : collection of internal or output transitions in which failure detection is performed
Ensure: Failure detection

```
1: T^f := ∅
2: for each t_i ∈ X^o ∪ Y do
3:     failureDetected := false
4:     k := 1
5:     while (not failureDetected ∧ (k ≤ |ω^o_{t_i}|)) do
6:         if (ω^o_{t_i}(k) ≠ ω^s_{t_i}(k)) then
7:             failureDetected := true
8:             T^f := T^f ∪ {t_i}
9:         else
10:            k++
11:        end if
12:    end while
13: end for
14: return T^f
end procedure
```

de sorties $Y = \{y_1\}$ simulées par l'algorithme VI.1 sont alors les suivantes :

$$\begin{cases} \omega^s_{x_2} = [5, 8, 11, 14, 17, 20, 23] \\ \omega^s_{x_4} = [6, 10, 14, 18, 22, 26, 30] \\ \omega^s_{y_1} = [11, 15, 19, 23, 27, 31, 35] \end{cases}$$

Nous considérons le scénario d'une faute temporelle permanente et présente dès le début de la phase de fonctionnement. Sur notre exemple illustratif, nous considérons qu'une dérive temporelle est présente sur la place p_5 et est caractérisée par un délai supplémentaire $d = 1$.

Les séquences observées des temps de franchissements des transitions x_2, x_4 et y_1 sont les suivantes :

$$\begin{cases} \omega^o_{x_2} = [5, 8, 11, 14, 17, 20, 23] \\ \omega^o_{x_4} = [7, 12, 17, 22, 27, 32, 37] \\ \omega^o_{y_1} = [12, 17, 22, 27, 32, 37, 42] \end{cases}$$

L'application de l'algorithme VI.2 nous fournit alors $T^f = \{x_4, y_1\}$. En effet, $\omega^o_{x_4}(1) \neq \omega^s_{x_4}(1)$ et $\omega^o_{y_1}(1) \neq \omega^s_{y_1}(1)$.

2.3.3 Localisation de dérives temporelles dans un GET

Si une dérive temporelle a été détectée, sa source doit être identifiée parmi les places du GET. L'étape de localisation, réalisée par l'algorithme VI.3, fournit l'ensemble des places candidates à être la source de la dérive temporelle. Par *place candidate*, nous signifions une place qui est en amont d'une des transitions $t_i \in T^f$ où a été détectée une différence entre la séquence observée des temps de franchissements et la séquence simulée des temps de franchissements (cf. étape de détection).

Pour chaque transition $t_i \in T^f$ (T^f est calculée par l'algorithme VI.2), nous calculons par une fonction PrePlaces(t_i), l'ensemble des places participant directement ou indirectement à cette transition. L'ensemble de toutes les places candidates contient la place source de la dérive temporelle. L'étape suivante sera pour le raffinement de cet ensemble de places par l'identification .

Algorithme VI.3 Localisation de dérives temporelles dans un GET

procedure FailureLocalization
Input: $\mathcal{T}$: a TEG
Input: T^f : collection of transitions in which failure detection is performed
Output: P^f : collection of candidate places of the source of the time shift
Ensure: Failure localization
1: $P^f := \emptyset$
2: **for each** $t_i \in T^f$ **do**
3: $P_i := prePlaces(t_i)$
4: $P^f := P^f \cup P_i$
5: **end for**
6: **return** P^f
end procedure

Exemple de localisation d'une dérive temporelle
Sur notre exemple illustratif, à la fin de l'étape de détection, nous avons obtenu $T^f = \{x_4, y_1\}$. La fonction PrePlaces calcule l'ensemble des places participant directement ou indirectement à une transition. Lorsque nous appliquons cette fonction sur les transitions $x_4, y_1 \in T^f$, nous obtenons : PrePlaces$(x_4) = \{p_2, p_5, p_6\}$ et PrePlaces$(y_1) = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}\}$. Aussi, l'algorithme VI.3 retourne l'ensemble des places candidates à être la source de la dérive temporelle $P^f = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}\}$.

2.3.4 Identification de dérives temporelles dans un GET

Il est nécessaire de réduire la liste des places candidates pouvant être à l'origine de la dérive temporelle. L'algorithme VI.4 réalise les deux étapes de raffinement suivantes :

1. Tout d'abord, nous réduisons l'ensemble de places P^f obtenu par l'algorithme de localisation en appliquant des *règles de simplification* (propositions 1 and 2).

2. Puis, pour chaque place restante dans la liste des places candidates, nous testons si cette place peut être la source de la dérive temporelle par *résolution d'un ensemble d'équations symboliques.*

2.3.4.1 Régles de simplification

L'algorithme de localisation (Algorithme VI.3) retourne P^f un ensemble de places dont on sait que l'une est à l'origine de la dérive temporelle. Il est possible de simplifier / raffiner cet ensemble de places candidates en appliquant les règles de simplification issues des deux propositions suivantes. L'ensemble des places candidates *raffinées* sera dénoté P^r par la suite.

Proposition 1 *Dans l'hypothèse d'une unique faute temporelle, si deux transitions (interne ou de sortie)* t_i *et* t_j *présentent la présence d'une dérive temporelle* ($t_i, t_j \in T^f$) *et que* $\text{PREPLACES}(t_i) \subseteq \text{PREPLACES}(t_j)$*, nous pouvons retirer de* P^f *toutes les places appartenant à* $\text{PREPLACES}(t_j)$ *qui ne sont pas présentes dans* $\text{PREPLACES}(t_i)$*.*

Proposition 2 *Dans l'hypothèse d'une unique faute temporelle, si une transition (interne ou de sortie)* t_k *ne présente pas la présence d'une dérive temporelle* ($t_k \notin T^f$)*, nous pouvons retirer de* P^f *toutes les places apparaissant dans un chemin sans synchronisation menant à la transition* t_k *([191]). Comme la transition* t_k *ne présente pas de dérives temporelles, ces places sont nécessairement sans faute à cause de l'absence d'une transition de synchronisation sur le chemin menant à* t_k *(il faut prendre en compte qu'une transition de synchronisation peut masquer la présence d'une dérive temporelle). Cet ensemble de places non fautives est calculé par la fonction* DIRECTPREPLACES*.*

2.3.4.2 Résolution d'un ensemble d'équations symboliques

La deuxième étape de raffinement des places candidates dans l'algorithme VI.4 d'identification procède comme suit : Après application des règles de simplification sur l'ensemble des places candidates P^f, nous obtenons un ensemble simplifié de places candidates P^r (LIGNE 1). Il est encore possible de raffiner cet ensemble de places candidates en calculant pour chaque place candidate $p \in P^r$ (LIGNE 2) quelle valeur de décalage temporel d expliquerait la première détection de dérive temporelle sur les transitions observées du système réel. La valeur de d est obtenue par résolution d'un système d'équations symboliques (LIGNES 3-4). Pour valider ou infirmer que la place p est bien la source de la dérive temporelle de délai d, notre simulateur de GET est utilisé pour vérifier qu'avec cette injection de faute temporelle, les séquences simulées des temps de franchissement des transitions observables du GET correspondent aux séquences observées des temps de franchissement des transitions observables du système réel (LIGNES 5-8).

Algorithme VI.4 Identification de dérives temporelles dans un GET

procedure FAILUREIDENTIFICATION
Input: $\mathcal{T}$: a TEG
Input: T^f : collection of transitions in which failure detection is performed
Input: P^f : collection of candidate places of the source of the time shift
Inputs: $\Omega^o_U, \Omega^o_{X^o}, \Omega^o_Y$: collection of observed sequences of firings times of transitions
Output: P^r : refinement of P^f after reduction rules and solving symbolic equations
Ensure: Failure identification
1: *Initizalize P^r with P^f and apply reduction rules (Propositions 1 and 2)*
2: **for each** $p \in P^r$ **do**
3: *Select a transition $t_i \in T^f$*
4: *Express and solve a system of symbolic equations where $\delta(p)$ is remplaced by $\delta(p) + d$ by using the sequences of firing times of the observed transitions : use of the first firing time where the failure is detected*
5: *Check if the solution d is valid with the next firing times where the failure is detected*
6: **if** (*d is not a valid solution*) **then**
7: *Remove p from P^r*
8: **end if**
9: **end for**
10: **return** P^r
end procedure

2.3.4.3 Exemple d'identification d'une dérive temporelle

Sur notre exemple illustratif, à la fin des étapes de détection et de localisation, nous avons obtenu $T^f = \{x_4, y_1\}$ et $P^f = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}\}$. Les étapes de raffinement de places candidates sont les suivantes :

— Nous appliquons la proposition 1 pour réduire l'ensemble des places candidates : $x_4, y_1 \in T^f$ et PREPLACES$(x_4) \subseteq$ PREPLACES(y_1). Aussi, l'ensemble des places candidates est simplifié à $P^r = \{p_2, p_5, p_6\}$. Sur notre exemple, la proposition 2 ne peut pas être utilisée pour réduire P^r.

— Nous supposons que la place p_5 est la source de la dérive temporelle et nous sélectionons la transition $x_4 \in T^f$ dont le comportement dépend de p_5. L'étape de détection a détecté la dérive temporelle lors du premier franchissement de la transition x_4 : $\omega^o_{x_4}(1) \neq \omega^s_{x_4}(1)$. Nous exprimons alors par un ensemble d'équations symboliques les relations entre la transition x_4 pour son premier franchissement et les entrées u_1 et u_2. Le temps de séjour de la place p_5 est alors modélisé par l'expression $\delta(p_5) + d$ où d est la variable symbolique à résoudre. Le système d'équations symbolique est le suivant : $\begin{cases} \omega_{x_4}(1) = \omega_{x_3}(1) + \delta(p_5) + d \\ \omega_{x_3}(1) = \max(\omega_{x_4}(0) + \delta(p_6), \omega_{u_2}(1) + \delta(p_2)) \end{cases}$
En utilisant les séquences des temps de franchissement de transitions en entrées ($\omega^o_{u_1} = \omega^o_{u_2} = [1, 2, 3, 4, 5, 6, 7]$), le système d'équations symboliques se simplifie ainsi :
$\begin{cases} \omega_{x_4}(1) = \omega_{x_3}(1) + 4 + d \\ \omega_{x_3}(1) = \max(0 + 0, 1 + 1) = 2 \end{cases}$

Comme $\omega^o_{x_4}(1) = 7$, nous obtenons l'équation symbolique $7 = 2 + 4 + d$. La solution pour la dérive temporelle est $d = 1$. Nous testons si cette solution $d = 1$ est valide pour les prochains temps de franchissements : nous utilisons l'algorithme VI.1 comme suit : nous remplaçons le temps de séjour de la place p_5 par $\delta(p_5) = 4 + d = 5$. Les séquences simulées des temps de franchissement des transitions x_2, x_4, y_1 avec cette injection de faute temporelle sont identiques aux séquences observées des temps de transitions. En conséquence, nous conservons p_5 dans P^r.

— Nous supposons que la place p_6 est la source de la dérive temporelle et nous sélectionnons la transition $x_4 \in T^f$. En appliquant le même raisonnement que pour la place p_5, nous obtenons le système d'équations symboliques :

$$\begin{cases} \omega_{x_4}(1) = \omega_{x_3}(1) + \delta(p_5) \\ \omega_{x_3}(1) = \max(\omega_{x_4}(0) + \delta(p_6) + d, \omega_{u_2}(1) + \delta(p_2)) \end{cases}$$

La solution à ce système est $d = 3$ et n'est pas valide avec les prochains temps de franchissement observés $\omega^o_{x_4}$. En conséquence, nous retirons p_6 de P^r.

— De manière similaire, nous retirons $p2$ de P.

— L'algorithme d'identification retourne alors $P^r = \{p_5\}$, la dérive temporelle est sur la place p_5 avec un délai supplémentaire $d = 1$.

3 Conclusion et perspectives sur l'identification de fautes temporelles

Nous avons proposé une approche algorithmique au diagnostic de fautes temporelles pour des systèmes de production modélisables par des graphes d'événements temporisés (GET). Tout d'abord, nous avons développé un ensemble de structures de données pour modéliser les GET et réaliser leurs simulations. Puis nous avons développé plusieurs algorithmes pour la détection, la localisation et l'identification de fautes temporelles.

Plusieurs perspectives peuvent être envisagées pour poursuivre ce travail de recherche :

— Prendre en compte des incertitudes sur les temps de séjour dans les places en modélisant la temporalité par un intervalle au lieu d'une durée fixe.

— Considérer que la dérive temporelle peut survenir à n'importe quel moment. Nous avons en effet fait l'hypothèse que la faute temporelle est présente dès le début de la phase de fonctionnement du système.

— Etendre notre méthodologie de diagnostic aux fautes multiples, nous avons considéré la présence d'une unique faute temporelle.

— Elargir notre méthode de diagnostic aux GET valués qui permet d'associer une valuation aux arcs afin de modéliser des opérations de type *batch* ou *split* pour regrouper ou diviser des ressources (dans les GET, le poids de chaque arc est 1).

— Développer une méthodologie d'analyse de la diagnosticabilité des dérives temporelles dans les GET.

Axe B)

Outils pour le diagnostic et la maintenance

VII Usage des arbres de défaillances dynamiques pour la SdF **115**

1 Présentation générale : contexte, état de l'art, positionnement, originalité . . 115

2 Analyse des causes racines des défaillances avec les arbres de défaillances dynamiques 126

3 Analyse quantitative d'arbres de défaillances dynamiques 132

4 Simulation d'arbres de défaillances dynamiques : application au filtrage de fausses alarmes 144

5 Conclusion et perspectives sur l'usage des arbres de défaillances dynamiques pour la sûreté de fonctionnement 151

VIII Usage des files d'attente et des métaheuristiques pour la maintenance **155**

1 Présentation générale : contexte, positionnement, originalité, état de l'art . . 155

2 Synthèse des travaux développés 158

3 Conclusion et perspectives sur la maintenance 179

Collaborations et encadrements de travaux de recherche dans l'Axe B)

Les travaux de recherche relatés dans les chapitres suivants VII et VIII ont été réalisés par une collaboration avec Zineb Simeu-Abazi (laboratoire G-SCOP, Polytech Grenoble) et Maria Di Mascolo (laboratoire G-SCOP, CNRS) et par le co-encadrement de stages de niveau Master. Les prochains chapitres présenteront une synthèse des travaux réalisés et citeront les publications issues de ces collaborations de recherche.

Publications issues de collaborations et d'encadrements de thèses et de stages de Master dans l'Axe B)

(RI : Revue Internationale, CI : Conférence Internationale, WI : Workshop International)

[RI-1] Z. Simeu-Abazi, M. Di Mascolo, and **E. Gascard**, "Queuing network-based methodology for designing and assessing performance of centralized maintenance workshops", *Journal of Manufacturing Technology Management*, vol. 25, no. 4, pp. 510–527, 2014.

[RI-2] **E. Gascard** and Z. Simeu-Abazi, "Quantitative Analysis of Dynamic Fault Trees by means of Monte Carlo Simulations: Event-Driven Simulation Approach", *Reliability Engineering & System Safety*, vol. 180, pp. 487–504, 2018.

[CI-1] **E. Gascard**, Z. Simeu-Abazi, and J. Younes, "Exploitation of Built in test for diagnosis by using Dynamic Fault Trees: Implementation in Matlab Simulink", in *Proceedings of the 20th European Safety and Reliability Conference, ESREL 2011*, CRC Press, 2011, pp. 436–444.

[CI-2] **E. Gascard** and Z. Simeu-Abazi, "Failure Root Causes Analysis of Complex Systems – Dynamic Fault Tree Approach", in *Proceedings of the 25th European Safety and Reliability Conference (ESREL 2015)*, CRC Press, 2015, pp. 695–703.

[CI-3] **E. Gascard**, Z. Simeu-Abazi, and Y. Sidqui, "Quantitative analysis of Dynamic Fault Tree by probabilistic approach", in *Proceedings of the 25th European Safety and Reliability Conference (ESREL 2015)*, CRC Press, 2015, pp. 735–743.

[CI-4] Z. Simeu-Abazi and **E. Gascard**, "Implementation of a cost optimization algorithm in a context of distributed maintenance", in *Proceedings of the 4th International Conference on Control, Automation and Diagnosis (ICCAD'20)*, IEEE, 2020, pp. 1–6.

[WI-1] Z. Simeu-Abazi, M. Di Mascolo, and **E. Gascard**, "Performance Evaluation of Centralized Maintenance Workshop by using Queuing Networks", in *Proceedings of the 2nd Workshop on Advanced Maintenance Engineering, Service and Technology (AMEST 2012)*, 2012.

Chapitre VII

Usage des arbres de défaillances dynamiques pour la sûreté de fonctionnement des systèmes dynamiques

1 Présentation générale : contexte, état de l'art, positionnement, originalité

1.1 Contexte

La discipline de la sûreté de fonctionnement (SdF) est un ensemble de démarches, d'outils et de méthodes qui visent à apporter la fiabilité (Reliability), la disponibilité (Availability), la maintenabilité (Maintainability) et la sécurité (Safety) de systèmes ou d'activités industrielles. Ces attributs de la sûreté de fonctionnement sont connus sous l'acronyme RAMS ([158], [192], [193]). Comme moyen pour contribuer aux attributs de fiabilité, disponibilité et sécurité, la prévision des fautes permet d'évaluer en amont le comportement d'un système par rapport à l'occurrence des fautes, d'anticiper les fautes et leur impact sur le système. Plusieurs méthodes d'analyse de prévision des fautes pour les systèmes industriels ont été développées , par exemple : l'Analyse des Modes de Défaillance et de leurs Effets (Failure Mode and Effects Analysis, FMEA), l'Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité (Failure Modes, Effects and Criticality Analysis, FMECA), l'Arbre de Défaillances (Fault Tree Analysis, FTA). Ces méthodes permettent des études ou des analyses qualitatives et quantitatives des défaillances. Comme axe de recherche, nous nous intéressons à l'outil de l'arbre de défaillances [194]-[197] et son extension l'arbre de défaillances dynamique (Dynamic Fault Tree, DFT) [198]-[200] pour la prévision des fautes et l'aide au diagnostic. Plus précisément, je me suis intéressé à l'utilisation des arbres de défaillances dynamiques pour l'aide à la décision dans le diagnostic et la maintenance.

1.2 Présentation des arbres de défaillances statiques et dynamiques

1.2.1 Arbres de défaillances statiques

Les entraves qui peuvent affecter un système et dégrader la sûreté de fonctionnement sont réparties en 3 notions : les fautes, les défauts et les défaillances [192]. La défaillance est la conséquence d'un défaut, dont la cause est une faute. La méthode de l'arbre de défaillances a pour objectif de déterminer les combinaisons possibles d'événements qui entraînent l'occurrence d'un événement indésirable (ou redouté). Un événement dans un arbre de défaillances correspond à la notion de défaillance. L'analyse par arbre de défaillances est une méthode de type déductive. A partir de l'événement redouté, il faut déterminer les combinaisons possibles d'événements conduisant à la réalisation de cet événement redouté. L'arbre de défaillances est composé des portes OR, AND et k-out-of-n qui sont des portes combinatoires ou statiques, on parle alors d'arbre de défaillances statique. L'arbre de défaillances est représenté graphiquement de haut en bas. La ligne la plus haute représente l'événement redouté, appelé événement sommet (top event), chaque ligne intermédiaire détaille comment chaque événement de la ligne qui lui est supérieure peut se produire à partir de combinaisons d'événements.

Chaque événement concerne la défaillance d'un composant, d'un sous-système ou du système dans son intégralité. Les événements apparaissant dans l'arbre de défaillances peuvent être divisé en événements basiques, événements conditionnels, événements non développés, événements externes et événements intermédiaires [194]. Différentes formes sont utilisées pour représenter ces événements dans la représentation graphique de l'arbre de défaillances.

Un *événement basique* ou *élémentaire*, représenté par un cercle, est un événement qui ne peut pas être détaillé davantage, il concerne la défaillance d'un composant du système. Un *événement intermédiaire*, représenté par un rectangle, est un événement résultant d'un ou plusieurs autres événements. Il est généré par la sortie d'une porte logique dont les opérandes sont les événements qui contribuent à cet événement intermédiaire (l'événement sommet est un événement intermédiaire). Un *événement de conditionnement*, représenté par un ovale, est une condition spécifique qui peut être appliquée à certaines portes logiques pour restreindre son usage. Un *événement non développé*, représenté par un losange, est un événement qui ne peut être considéré comme basique, mais qui ne sera cependant pas développé par choix ou par manque d'informations. Un *événement externe*, représenté par une maison, est une défaillance externe au système normalement prévu dans l'analyse des risques (et qui n'est pas en soi la conséquence d'un défaut). La figure VII.1 (source : `https://dtk.tankonyvtar.hu/xmlui/handle/123456789/3828`) montre un exemple possible d'analyse déductive par arbre des défaillances de la panne d'un ascenseur. Dans le reste de cette section, nous ne considérerons que des arbres de défaillances composés uniquement d'*événements intermédiaires* et d'*événements basiques*.

A partir de l'arbre de défaillances, une *analyse qualitative* et une *analyse quantitative* peuvent être menées [201]. L'analyse qualitative vise à déterminer toutes les combinaisons d'événements basiques qui peuvent produire l'événement sommet, ce sont les coupes

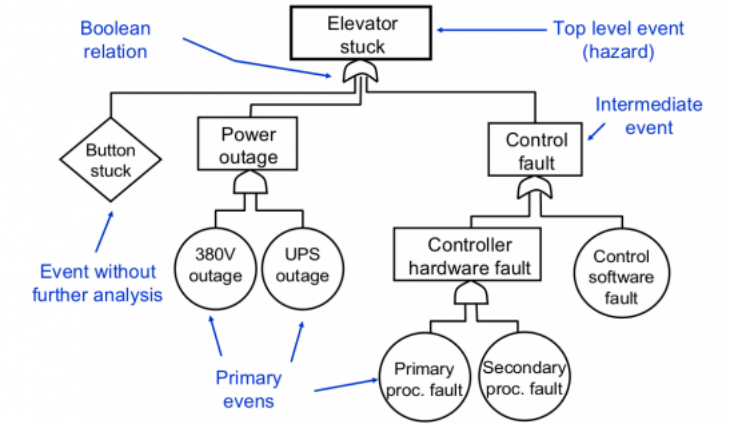
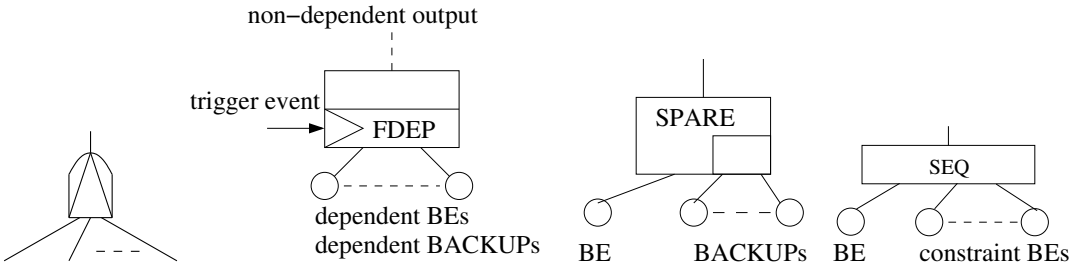


FIGURE VII.1 – Exemple d'un arbre de défaillances d'un ascenseur

minimales. La connaissance des coupes minimales permet « d'évaluer l'effet d'une modification du système, de comparer les conséquences des mesures qui peuvent être envisagées pour réduire l'occurrence de l'événement redouté étudié. » [196]. L'analyse quantitative vise à évaluer à partir des probabilités d'occurrence des événements basiques, la probabilité d'occurrence de l'événement redouté. Pour mener cette analyse quantitative, il est présumé que les événements basiques sont indépendants entre eux. Plusieurs mesures de fiabilité peuvent être déduites de l'analyse quantitative : $Q(t)$ la probabilité que l'événement sommet se réalise à l'instant t, $R(t)$ la probabilité que l'événement sommet ne se réalise pas dans l'intervalle $[0, t)$, le MTTF le temps moyen avant la première défaillance de l'événement sommet, etc.

1.2.2 Arbres de défaillances dynamiques

Le formalisme des arbres de défaillances statiques, qui n'utilise que des portes booléennes, souffre d'un manque d'expressivité. D'une part, comme les événements basiques sont assumés indépendants entre eux, les portes booléennes sont insuffisantes pour représenter une dépendance fonctionnelle entre les événements. D'autre part, la logique booléenne ne permet pas de représenter un comportement dynamique : l'ordre d'arrivée des événements. Finalement, les arbres de défaillances statiques ne permettent pas de modéliser la redondance de composants : l'utilisation de composant de rechange. Pour prendre en compte les comportements dynamiques tels que la dépendance fonctionnelle, l'ordre d'arrivée des événements et la gestion de la redondance, de nouvelles portes, dites dynamiques, ont été proposées par Dugan et al. [198], [199]. Les arbres de défaillances dynamiques étendent les arbres de défaillances statiques par l'usage de portes dynamiques. La méthodologie des arbres de défaillances dynamiques est présentée dans [200]. La figure VII.2 présente les quatre portes dynamiques proposées par Dugan et aL



(a) Porte PAND (b) Porte FDEP (c) Porte SPARE (d) Porte SEQ

FIGURE VII.2 – Portes dynamiques [198], [199]

La porte PAND

La porte PAND (figure VII.2(a)) a été introduite par Fussel et al. [202] pour ajouter une contrainte d'ordre d'occurrence aux entrées d'une porte AND. Elle étend ainsi la sémantique de la porte booléenne AND en prenant en compte l'ordre d'arrivée des événements qui lui sont connectés. La sortie de la porte PAND, modélisant une défaillance, est vraie (représentée par la valeur 1) si ses entrées se produisent dans un ordre spécifique : de la gauche vers la droite ; faux sinon (représentée par la valeur 0).

La porte FDEP

La porte Functional Dependency (figure VII.2(b)) définie par Dugan et al. [198] modélise une dépendance fonctionnelle entre une gâchette (trigger) et des composants élémentaires dépendants. Lorsque la gâchette est active, les composants élémentaires dépendants deviennent inutilisables. Dans une porte FDEP, la défaillance de la gâchette, c.-à-d. son activation, entraîne la défaillance des événements basiques dépendants qui lui sont connectés.

La porte SPARE

La porte SPARE (figure VII.2(c)) définie par Dugan et al. [198] modélise l'utilisation des composants de rechange (spares). Elle possède une entrée primaire qui correspond à un composant principal initialement actif et une ou plusieurs entrées secondaires qui correspondent aux composants de rechange qui sont initialement inactifs (dormants). Lorsque le composant principal défaille, le composant de rechange situé le plus à gauche dans les entrées secondaires est sollicité en premier à condition qu'il ne soit pas déjà sollicité par une autre porte SPARE (situation d'un composant de rechange partagé entre différentes portes SPARE), auquel cas, le deuxième composant de rechange est sollicité. Quand un composant de rechange sollicité défaille, le composant de rechange à sa droite est alors sollicité. La sortie de la porte SPARE, modélisant une défaillance, est vraie (1) si son composant principal a défailli et que tous ses composants de rechange sont soit défaillants, soit sollicités par une autre porte SPARE dans le cas d'un composant de rechange partagé. La porte SPARE a trois déclinaisons (*températures*) [203] : Cold (CSP), Warm (WSP) ou Hot

(HSP) qui décrivent l'atténuation du taux de défaillance (dormancy factor α) des composants de rechange avant leur sollicitation. Avec la porte CSP, les composants de rechange ne défaillent pas avant leur sollicitation ($\alpha = 0$). Avec la porte HSP, les composants de rechange défaillent de la même façon avant et après leur sollicitation ($\alpha = 1$). Avec la porte WSP, les composants de rechange défaillent différemment avant et après la sollicitation ($0 < \alpha < 1$).

La porte SEQ

La porte Sequence Enforcing (figure VII.2(d)) définie par Dugan et al. [199] impose que ses entrées, des événements basiques, se produisent dans un ordre spécifique : de la gauche vers la droite. La sortie de la porte SEQ est vraie si toutes ses entrées deviennent défaillantes dans l'ordre de leurs connexions. La porte SEQ se différencie de la porte PAND : la porte PAND détecte si ses événements en entrée apparaissent dans un ordre spécifique (les événements peuvent survenir dans un autre ordre, dans ce cas là la porte PAND ne s'active pas), alors que la porte SEQ ne permet aux événements en entrée de n'apparaitre que dans un ordre spécifique.

1.2.3 Autres extensions des arbres de défaillances

Il existe d'autres extensions possibles des arbres de défaillances comme [204]-[209]. Čepin et Mavko présentent dans [204] une approche d'arbres de défaillances dynamiques en introduisant les concepts de *house events* et *house event matrix* qui permettent de décrire les dépendances temporelles entre les événements et le fonctionnement du système à un moment donné.

Palshikar dans [205] définie la méthodologie *Temporal Fault Tree (TFT)* qui étend l'arbre de défaillances statique en introduisant de nouvelles portes temporelle exprimant d'une part des dépendances temporelles entre différents événements (portes SOMETIME, ALLPAST, PREV) et des contraintes temporelles sur les n instants précédents (portes WITHIN n, FORPAST n, PREV n).

Bouissou et Bon proposent dans [206] proposent les BDMPs (*Boolean logic Driven Markov Processes*) pour modéliser des systèmes dynamiques stochastiques réparables. Les BDMPs reprennent la représentation graphique des arbres de défaillances et ajoutent de nouveaux liens appelés gâchettes. Les événements ne sont plus associés à une variable booléenne, mais à des processus de Markov.

Dans [207], Lefebvre et al. proposent de nouvelles portes temporelles dans le cadre d'événements temporaires qui apparaissent dans l'avionique. Ces nouvelles portes permettent de prendre en compte la dépendance temporelle entre différents événements (porte PANDW - PAND fenêtre temporelle *Window*), la comparaison du temps de défaillance d'un événement par rapport à un seuil T (porte DUR - Durée), la comparaison du nombre de défaillances d'un événement à un nombre d'occurrences N (porte COUNT - compteur).

Walker et Papadopoulos proposent dans [208], [209] le formalisme Pandora pour des arbres de défaillances temporels. Les auteurs utilisent les portes temporelles Priority-AND

(PAND), Simultaneous-AND (SAND) et Priority-OR (POR).

Nos travaux de recherche sont axés sur les arbres de défaillances dynamiques (DFT) proposées par Dugan et al. Dans le reste de cette section, notre état de l'art sera donc principalement dédié aux travaux relatifs aux DFT.

1.3 Etat de l'art sur l'analyse qualitative des arbres de défaillances dynamiques

L'analyse qualitative d'un arbre de défaillances statique/dynamique vise à identifier les combinaisons/séquences des événements qui amènent à l'activation de l'événement sommet de l'arbre de défaillances statique/dynamique.

Dans [210], [211], Xiang et al. proposent une méthode d'analyse combinatoire d'arbres de défaillances dynamiques n'utilisant que des portes PAND. Dans leur méthodologie, ils transforment la porte PAND en une porte AND avec un ajout de condition d'événement (porte CAND) puis expriment l'analyse qualitative comme un problème combinatoire sans utiliser le concept de séquences de coupes minimales.

Il est possible de mener une analyse qualitative basée sur l'ensemble des coupes minimales à un arbre de défaillance dynamique si les portes dynamiques du DFT sont remplacées par des portes statiques. Par exemple, en remplaçant une porte FDEP par une porte OR et en remplaçant les portes PAND et SPARE par des portes AND. Cependant, les dépendances dynamiques, temporelles entre les événements sont alors perdues, elles n'apparaissent pas dans les ensembles de coupes minimales.

Tang et Dugan [212] ont alors étendu le concept de coupe minimale des arbres de défaillances statiques aux séquences de coupes minimales des arbres de défaillances dynamiques pour réaliser leur analyse qualitative. Les auteurs proposent une méthodologie à base de *Zero-suppressed Binary Decision Diagram* [213] pour générer les ensembles de coupes minimales des arbres de défaillances statiques et les ensembles de séquences de coupes minimales pour les arbres de défaillances dynamiques.

Liu et al. [214] définissent un algorithme pour la génération de l'ensemble des séquences de coupes en utilisant un mécanisme pour décrire la défaillance séquentielle entre deux événements indépendants. Leur méthode utilise un symbole particulier, le *Sequential Failure Symbol*, pour décrire l'ordre de défaillance entre deux événements.

Zhang et al. [215] proposent le concept de séquence de coupes étendues (Extended Cut Sequence) qui est basée sur les séquences de coupes : c'est l'ensemble des événements basiques associé à un ensemble de contraintes temporelles entre eux qui causent l'événement sommet de l'arbre de défaillances dynamique.

Merle dans sa thèse [216] utilise une méthode algébrique pour exprimer et calculer la séquence de coupes de DFT. Ses travaux s'appuient sur une extension de la fonction de structure utilisée classiquement pour les arbres de défaillances statiques afin de l'appliquer aux arbres de défaillances dynamiques. Pour cela, il introduit de nouveaux opérateurs temporels (non-inclusive before, inclusive before et simultaneous) qui permettent de modéliser les portes PAND, FDEP et SPARE.

Rauzy dans [217] définit une algèbre séquentielle et utilise le diagramme de décision séquentielle qui est une variante du Zero-Suppressed Binary Decision Diagram permettant le calcul d'opérations d'algèbre séquentielle de manière efficiente. L'utilisation des diagrammes de décision séquentielle permet le calcul des séquences de coupes.

Il existe d'autres travaux sur l'analyse qualitative de systèmes dynamiques avec une autre représentation que les arbres de défaillances dynamiques [205], [218], [219]. Dans [205], Palshikar propose une méthode d'analyse qualitative des arbres de défaillances temporels construits à partir de nouvelles portes temporelles (SOMETIME, PREV, WITHIN, etc.). Dans [218], Walker et Papadopoulos proposent une méthode d'analyse qualitative des arbres de défaillances temporels Pandora. Piriou et al. [219] s'intéressent à l'analyse qualitative de systèmes dynamiques stochastiques réparables modélisés par des BDMP (Boolean logic Driven Markov Processes [206]). Les auteurs redéfinissent la notion de séquence de coupe dans le cadre des systèmes dynamiques réparables et proposent un algorithme pour calculer les ensembles de séquences de coupe minimale.

1.4 Etat de l'art sur l'analyse quantitative des arbres de défaillances dynamiques

L'analyse quantitative des arbres de défaillances vise à évaluer la probabilité de l'événement sommet de l'arbre de défaillances à partir des probabilités des événements basiques. Le résultat de cette analyse quantitative peut être exact ou approché. Les méthodes d'analyse quantitative des arbres de défaillances dynamiques peuvent être classées en trois catégories : (1) les méthodes algébriques ; (2) les méthodes basées sur des systèmes à espaces d'états ; (3) les méthodes basées sur la simulation.

1.4.1 Méthodes algébriques

Les premiers travaux sur l'analyse quantitative par une approche algébrique furent ceux de Fussel et al. [202] qui proposent une méthode de calcul exacte ou approchée de la quantification de la porte PAND.

De nombreux travaux proposent une approche algébrique pour la quantification de DFTs, ils utilisent l'un de ces concepts : ensemble de séquences de coupes minimales (MCSS) [214], [220], [221], ensemble de séquences de coupes minimales étendues (MECSS) [215], ensemble de séquences de coupes minimales généralisées (GMCS) [222], forme canonique minimale de la fonction de structure [223] ou chemins disjoints.

L'approche à base de chemins disjoints est utilisée quand le DFT est converti en un *Sequential Binary Decision Diagram (SBDD)* [224]-[227] ou en un *Zero-suppressed Binary Decision Diagram (ZSDD)* [212].

Dans le cas des MCSS, MECSS, GMCS et de la fonction de structure l'analyse quantitative du DFT est menée en utilisant le principe d'inclusion-exclusion. [228]-[231] sont quelques exemples de travaux de recherche menés avec ces approches algébriques.

1.4.2 Méthodes basées sur des systèmes à espaces d'états

Avec les méthodes basées sur des systèmes à espaces d'états, l'arbre de défaillances dynamique est transformé en une chaîne de Markov, un réseau de Petri stochastique ou un réseau bayésien ou l'une de leurs extensions. Ces modèles stochastiques permettent de représenter le comportement dynamique du DFT en termes d'états atteignables et de transitions d'états. Une fois que le modèle stochastique est construit, l'analyse quantitative est obtenue par analyse numérique (équations différentielles stochastiques de Kolmogorov, transformée de Laplace, méthodes numériques implicites ou explicites).

Modèles de Markov pour quantifier les DFTs

La première méthode a été proposée par Dugan et al. dans [198], [199], elle convertit le DFT en une chaîne de Markov à temps continu.

Boudali et al. ont proposé dans [232], [233] une sémantique compositionnelle pour les DFTs où chaque élément du DFT est interprété comme une chaîne de Markov interactive et la sémantique du DFT est obtenue par composition parallèle de ses éléments. L'analyse quantitative du DFT est obtenue après transformation du DFT en une chaîne de Markov interactive. Le logiciel DFTCALC [234] qui permet l'analyse quantitative de DFT a été développé en utilisant cette approche.

Yevkin [235] propose une solution approchée à l'analyse quantitative des DFT. Cette approximation est basée sur la troncature, l'agrégation et l'élimination d'états de la chaîne de Markov pendant l'étape de transformation du DFT en chaîne de Markov.

Manian et al. [236] proposent également un algorithme de transformation de DFTs en chaînes de Markov à temps continu.

Avec la transformation du DFT en chaîne de Markov, le problème d'explosion combinatoire des états apparait. Pour réduire la taille de l'espace d'états lors de la conversion du DFT en chaîne de Markov, des techniques de modularisation ont été développées [237]-[242]. Le principe de la modularisation est basé sur le paradigme diviser-pour-régner. Le DFT est décomposé en plusieurs sous-arbres indépendants. Les sous-arbres contenant uniquement des portes booléennes sont quantifiés par exemple par une méthode à base de BDD et sont substitués par un nouvel événement basique. Pour les sous-arbres contenant des portes dynamiques, la méthodologie de conversion en chaîne de Markov est utilisée. Les résultats des sous-arbres indépendants sont combinés pour produire le résultat du DFT entier.

Réseaux de Petri pour quantifier les DFTs

Les réseaux de Petri et leur extension (généralisé, stochastique, coloré, temporel, etc.) sont largement utilisés comme outils de modélisation des systèmes dynamiques pour l'étude de leurs sécurités et de leurs fiabilités. Les réseaux de Petri ont été utilisés pour l'évaluation quantitative des arbres de défaillances statiques, par exemple [243], [244], puis pour l'évaluation quantitative d'arbres de défaillances dynamiques [245]-[247].

Bobbio et al. [245] proposent une méthode d'analyse quantitative pour les *Parametric*

Fault Trees qui sont une représentation plus compacte des DFT. Le Parametric Fault Tree est alors modularisé et chaque module est transformé en une extension d'un réseau de Petri coloré appelé *Stochastic Well-formerd Net (SWN)*. Ce réseau stochastique est utilisé pour générer une chaîne de Markov. Un outil DrawNet permet de réaliser ainsi l'analyse quantitative des Parametric Fault Trees.

Codetta-Raireti dans [246] propose des règles de transformation pour traduire les DFTs en *Generalized Stochastic Petri Net (GSPN)*. Un solveur a été développé dans l'outil DrawNet pour réaliser l'analyse quantitative des Generalized Stochastic Petri Net qui sont produits par cette approche.

Zhang et al. [247] proposent également une méthode de transformation des DFTs en réseau de Petri : ils proposent pour chaque type de porte dynamique un réseau de Petri équivalent. L'analyse quantitative est obtenue par simulation du réseau de Petri modélisant tout le DFT avec Matlab.

D'autres travaux définissent également des règles de transformation pour les portes dynamiques vers des réseaux de Petri stochastique [248], [249] mais ne proposent pas d'analyse quantitative.

On peut remarquer que les réseaux de Petri sont également utilisés pour l'analyse quantitative de système dynamique modélisé dans le formalisme Pandora [250].

Réseaux bayésiens pour quantifier les DFTs

De nombreux travaux ont été développés pour l'analyse quantitative d'arbres de défaillances dynamiques en utilisant les réseaux bayésiens.

Il existe des approches utilisant des réseaux bayésiens temporels discrets pour l'analyse quantitative des DFTs, par exemple [251], [252]. Ils produisent une solution rapide, mais pas forcément avec une grande précision, cela dépend de la discrétisation du temps utilisé.

Les approches utilisant les réseaux bayésiens temporels continus pour l'analyse quantitative de DFTs, par exemple [253]-[256], fournissent des résultats exacts, mais souffrent du problème d'explosion combinatoire d'états comme pour les approches à base de chaînes de Markov.

On peut noter que les réseaux bayésiens ont également été utilisés pour l'analyse quantitative d'arbres de défaillances temporels dans le formalisme Pandora [257].

1.4.3 Méthodes basées sur la simulation

Les approches algébriques ou basées sur des systèmes à espaces d'états connaissent des limitations. Pour les approches algébriques, leurs limitations sont principalement les lois de probabilités qui peuvent être utilisées (principalement la loi exponentielle) et les restrictions sur la représentation des DFTs prise en compte pour la génération des séquences de coupes ou des Sequential Binary Decision Diagram (événements répétés, spares partagés, placement des portes dynamiques dans le DFT). Pour les approches basées sur des systèmes à espaces d'états, leurs limitations portent également sur les distributions de défaillances prises en compte ainsi que le problème de l'explosion combinatoire du nombre d'états.

Pour combler ces difficultés, les approches à base de simulation de Monte-Carlo [258], [259] peuvent être utilisées dans l'étude de la fiabilité des systèmes complexes dynamiques. De nombreux travaux ont porté sur l'analyse quantitative de DFTs en utilisant la simulation de Monte-Carlo [260]-[273]. Nous proposons de classer ces travaux en cinq catégories.

Dans la première catégorie [260], [261], se trouvent les travaux les plus anciens qui transforment un DFT en un modèle (non) markovien suivi d'une simulation de Monte-Carlo sur le modèle produit.

Dans la seconde catégorie [262], [263], nous trouvons les méthodes qui génèrent aléatoirement les événements basiques selon leurs distributions de défaillances et propagent les défaillances des feuilles du DFT à son sommet. La propagation des défaillances dans le DFT est réalisée ainsi : chaque porte statique ou dynamique est associée avec une fonction qui implémente son comportement. Chaque fonction associée à une typologie de porte reçoit comme entrées le comportement temporel des opérandes de la porte durant le temps de simulation (c.-à-d. la description temporelle des états nominaux et défaillants) et retourne le comportement temporel de la sortie de la porte. Dans cette seconde catégorie, la simulation du DFT n'est pas dirigée par les événements qui se produisent pendant le temps de simulation, mais par la simulation de toutes les portes du DFT.

La troisième catégorie [264], [265] définit des bibliothèques de blocs dans Matlab/Simulink qui permettent de modéliser des DFT et de les simuler. Les portes dynamiques sont définies soit comme des fonctions [264], soit comme des systèmes de transitions adaptatifs [265], [274]. La simulation du DFT conduite par le simulateur Matlab/Simulink est de type simulation par pas de temps (time-driven simulation).

La quatrième catégorie [266]-[270] propose des méthodes hybrides pour l'analyse quantitative de DFTs. Ces méthodes combinent des approches algébriques ou des approches basées sur des systèmes à espaces d'états avec la simulation de Monte-Carlo.

Finalement, la cinquième catégorie [271]-[273] relate les travaux où la simulation de Monte-Carlo des DFTs est optimisée grâce à une simulation sur FPGA ou sur processeurs GPU. Dans ces travaux, les DFTs sont tout d'abord modélisés comme des arbres de défaillances temporels (*Time-To-Failure Tree* [275]) avant d'être simulés.

Le lecteur intéressé trouvera dans [276]-[278] un état de l'art complet sur tous les travaux relatifs aux arbres de défaillances dynamiques.

1.5 Positionnement et originalité de nos travaux

Sur la thématique de l'analyse qualitative d'arbres de défaillances dynamiques, nos travaux [279] ont porté sur la recherche des causes racines des défaillances de systèmes dynamiques complexes dans le cadre d'un diagnostic du système après sa défaillance. Notre méthode utilise la modélisation des causes possibles menant à la défaillance par un arbre de défaillances dynamiques et les traces d'exécution de tous les composants intervenant dans le système, ces traces d'exécution fournissent les dates de défaillances des composants. Nous nous positionnons dans le cadre d'un système dans lequel les événements des défaillances

des composants peuvent apparaitre puis disparaitre : nous prenons en compte la réparation des composants ou la présence de fausses alarmes générées par les composants. Nous avons proposé une méthode de recherche des causes racines qui utilise le concept de sensibilité des portes statiques et dynamiques pour identifier les opérandes des portes qui causent la modification de sa sortie et un algorithme de parcours en profondeur postfixe qui permet d'identifier les composants qui sont les causes racines de la défaillance de l'événement sommet de l'arbre de défaillances dynamiques.

Sur la thématique de l'analyse quantitative d'arbres de défaillances dynamiques, nos travaux [280], [281] se sont axés sur une approche à base de simulation de Monte-Carlo afin de prendre en compte tout type de distribution de défaillances. Les approches algébriques et à systèmes à espaces d'états sont souvent limitées à la loi exponentielle.

Nos premiers travaux ont porté sur le développement de programmes en Matlab et en Java ([280]) pour réaliser l'analyse quantitative des DFTs par simulation de Monte-Carlo. Nos programmes utilisaient la topologie de l'arbre de défaillances dynamique pour propager les événements générés au niveau des feuilles vers le sommet de l'arbre. Cependant, notre approche avait plusieurs limites : notre algorithme de propagation des événements dans le DFT ne permettait pas de prendre en compte les événements répétés (événement apparaissant comme opérande dans plusieurs portes dynamiques), les événements multi-contraint (événements apparaissant dans plusieurs portes SEQ ou FDEP) et les composants partagés (événement associé à des composants de rechange apparaissant dans plusieurs portes SPARE). De plus, les temps de calcul n'étaient pas efficients, toutes les portes de l'arbre de défaillances dynamiques étaient évaluées même s'il n'y avait pas l'arrivée d'événements de défaillances à leurs entrées.

Nous avons alors élaboré une nouvelle approche en utilisant la simulation conduite par les événements (event-driven simulation). Ce type de simulation est très efficient par rapport à la simulation par pas de temps (time-driven simulation) ou par rapport à l'approche où toutes les portes du DFT sont évaluées à chacun des cycles de simulation de Monte-Carlo. En effet, avec la simulation conduite par les événements, seules les portes dont des entrées sont actives sont évaluées. Des dizaines de millions de cycles de simulation de DFTs peuvent être réalisées en quelques secondes avec ce type de simulation. Dans [281], nous avons défini un algorithme de simulation conduite par les événements des arbres de défaillances dynamiques pour leur analyse quantitative. Notre approche permet de plus de prendre en considération toute représentation des arbres de défaillances dynamiques. Nous considérons les événements basiques répétés, les événements multi-contraints et les événements spare partagés. Nous nous sommes positionnés dans le cadre d'un système dont les composants sont non réparables, mais dans lequel la redondance de composants peut être possible (porte SPARE).

Dans le cadre de la sûreté de fonctionnement, nous nous sommes également intéressés à l'usage des arbres de défaillances dynamiques pour le filtrage de fausses alarmes [282]. Un arbre de défaillance dynamique représentera alors un ensemble de scénarios où des

alarmes sur des composants de base du système peuvent être levées. Dans le domaine de l'aéronautique, un grand nombre d'alarmes apparaissent lors de la phase de décollage et d'atterrissage d'un hélicoptère. Le diagnostic de ces alarmes n'est traité qu'une fois que l'appareil est au sol. L'équipe de maintenance a besoin à partir des enregistrements des alarmes dans les autotests (BIT : Built In Test) de détecter les équipements défaillants. Il est alors nécessaire d'enlever les ambiguïtés d'identification qui retardent les actions de maintenance en filtrant les fausses alarmes. Nous avons défini une sémantique de traces des portes dynamiques et de portes temporelles (PANDW, DUR, COUNT [207]) pour ensuite implémenter sous Matlab/Simulink une librairie permettant de créer un arbre de défaillances dynamique et de le simuler. La méthodologie de filtrages de fausses alarmes consiste alors à décrire par un arbre de défaillances dynamique les combinaisons des causes provoquant une fausse alarme, puis à le simuler en fournissant les chronogrammes décrivant le comportement des éléments de base de l'arbre. Le résultat des simulations permet d'identifier les composants susceptibles de produire de fausses alarmes et de réduire les coûts de maintenance.

2 Analyse des causes racines des défaillances avec les arbres de défaillances dynamiques

2.1 Contexte, positionnement

La maintenance industrielle joue un rôle primordial dans la fiabilité et la disponibilité des équipements. Elle permet de maintenir en état de fonctionnement la chaîne de production et réduit les coûts de production. La maintenance intervient à tous les stades du cycle de vie d'un équipement : lors de sa conception, par l'intégration de sa maintenabilité qui contribue à l'optimisation de son dépannage et permet ainsi un gain sur les coûts de maintenance ; lors de son usage quand l'équipement est en bon état de fonctionnement : c'est la maintenance préventive qui minimise le risque de défaillances (elle peut être systématique ou conditionnelle) ; lorsqu'une défaillance sur l'équipement apparait et entraîne son dysfonctionnement : c'est la maintenance corrective qui consiste en la remise en état de l'équipement défectueux (elle peut être palliative ou curative).

Nous nous positionnons dans le cadre de la maintenance corrective. Avant de procéder au dépannage de l'équipement, l'une des problématiques est d'identifier les causes racines de la défaillance pour identifier précisément les composants à remplacer ou réparer et ainsi éviter que la défaillance se reproduise. C'est le processus de l'*analyse des causes racines* (root causes analysis). La cause racine est la cause la plus fondamentale qui peut raisonnablement être identifiée et qui, lorsqu'elle sera réparée, empêchera la récurrence du problème [283]. Les causes de la défaillance peuvent être humaines, organisationnelles ou techniques [284]. Dans notre étude d'analyse des causes racines, nous nous sommes focalisés sur l'identification des causes techniques des défaillances dans les systèmes industriels. Dans ce cadre, il existe de nombreuses approches [285] :

— Les méthodes d'analyse technique et fonctionnelle telle que la méthode FAST (*Function Analysis System Technique*) ou la méthode SADT (*Structured Analysis and Design Technique*).

— Les outils du domaine de la qualité tels que la méthode QQOQCP (*quoi, qui, où, quand, comment, combien, pourquoi*) ou la méthode des cinq pourquoi ou le diagramme d'Ishikawa.

— Les méthodes de la sûreté de fonctionnement telles que l'analyse des modes de défaillances et de leurs effets (FMEA), l'analyse des modes de défaillance, de leurs effets et de leur criticité (AMDEC) ou l'analyse par arbre de défaillances (Fault Tree Analysis).

Nos travaux ont porté sur le développement d'une méthode d'identification des causes racines d'une défaillance modélisée par un arbre de défaillances dynamique à partir des traces d'exécutions de tous les composants. La section suivante présente notre méthodologie d'analyse des causes racines des défaillances en utilisant les arbres de défaillances dynamiques.

2.2 Méthode d'analyse des causes racines avec les arbres de défaillances dynamiques

Notre méthode a pour prérequis la définition de l'arbre de défaillances dynamique qui génère la défaillance redoutée dont il faut identifier ses causes racines et les traces de fonctionnement de tous les composants. Nous considérons qu'il peut y avoir plusieurs événements de défaillances des composants si il y a eu réparation des composants ou que les défaillances des composants peuvent être intermittentes dans le cas de fausses alarmes.

2.2.1 Définitions préliminaires

Notre méthodologie associe à chaque nœud de l'arbre de défaillances dynamique (événement intermédiaire correspondant à la sortie d'une porte ou événement basique correspondant à l'état de fonctionnement d'un composant) une trace décrivant son évolution. Nous considérons deux types de traces : les traces d'exécution et les traces de sensibilité.

Définition 4 (Trace d'exécution) *La trace d'exécution sera utilisée avec les composants (feuilles de l'arbre de défaillances dynamique). Une trace d'exécution ω_C d'un composant C est représentée par sa séquence de sortie d'impulsion. La séquence d'impulsion est une séquence alternée de $(0, t_i)$ et $(1, t_j)$ avec t_i, t_j des instants discrets.* 0 *représente l'absence de détection de défaillance,* 1 *représente la détection de défaillance. Une séquence d'impulsion peut être vue comme un chronogramme : une paire $(1, t)$ dans la trace représente un front montant dans le chronogramme, une paire $(0, t)$ représente un front descendant. Nous prenons comme hypothèses que la première paire de toute trace de simulation associée à un composant est $(0, 0)$ (le composant est en bon état de fonctionnement au début), que les défaillances sont « réparables » (prise en compte des réparations sur le composant ou présence de fausses alarmes) et que le temps est discrétisé.*

Définition 5 (Trace de sensibilité) *La trace de sensibilité sera utilisée avec les portes statiques et dynamiques (événements intermédiaires ou événement, sommet). Une trace de sensibilité est similaire à une trace d'exécution excepté que la contrainte de séquence alternée de* $(0, t_i)$ *et* $(1, t_j)$ *est allégée : des occurrences successives de* $(1, t_k)$ *sont permises. Les occurrences successives de* $(1, t_k)$ *après le premier front montant indiquent que bien que la porte continue à détecter une défaillance, les causes de cette défaillance ont changé.*

La définition des causes racines d'un nœud d'un arbre de défaillances dynamique basées sur sa trace (d'exécution ou de sensibilité) et sur les causes racines de ses opérandes est définie ci-dessous.

Définition 6 (Causes racines de la défaillance d'un événement basique) *A partir de la trace d'exécution* ω_C *associée à un événement basique* C *du DFT, les causes racines de toutes les défaillances de* C *sont définies par* $\mathcal{F}_C = \{(t_k, C) | \exists (1, t_k) \in \omega\}$.

Définition 7 (Causes racines de la défaillance d'un événement intermédiaire) *A partir de la trace de sensibilité* ω *d'un événement intermédiaire représentant la sortie d'une porte (statique ou dynamique), les causes racines de la défaillance à l'instant* t *dans* ω *(c.-à-d.* $(1, t) \in \omega$*), dénoté* $\mathcal{F}(t)$*, sont représentées par un ensemble de paires* (t_k, C_i) *où* C_i *est le composant défaillant qui prend part à la défaillance du noeud intermédiaire,* $t_k \leq t$ *est l'instant de défaillance du composant* C_i*. Les causes racines de toutes les défaillances dans* ω *sont représentées par* $\mathcal{F} = \{(t_k, \mathcal{F}(t_k)) | \exists (1, t_k) \in \omega\}$.

Définition 8 (Etat d'un nœud de l'arbre de défaillances dynamique) *Dans la suite pour décrire l'état d'un nœud/événement de l'arbre de défaillances dynamique, nous utiliserons le qualificatif de désactivé ou activé. L'état d'un nœud/événement du DFT à l'instant* t *selon sa trace (d'exécution ou de sensibilité) est désactivé s'il existe une paire* $(0, t)$ *dans la trace ou qu'il n'existe pas de paire* $(0, t)$ *dans la trace et la plus proche paire dans le passé est une paire* $(0, t_k)$ *avec* $t_k < t$ *; sinon, son état est activé.*

La suite de cette section présente la méthode de calcul des traces de sensibilité et des causes racines des portes statiques et dynamiques.

2.2.2 Analyse des causes racines des portes booléennes

Nous présentons dans ce manuscrit uniquement l'étude des portes OR et AND, la porte majorité k-out-of-n est présentée dans notre article [280]. Pour simplifier la présentation de nos travaux, nous ne considérerons que des portes OR et AND à deux entrées.

Porte OR

L'analyse des causes racines d'une porte OR nécessite de construire préalablement la trace de sensibilité de la sortie de la porte OR à partir des traces de ses opérandes.

La trace de sensibilité w d'une porte OR est définie comme suit à partir des traces (de simulation ou de sensibilité) de ses opérandes :

— Une paire $(1, t)$ apparait dans ω si et seulement si :

- $(1,t)$ apparait dans la trace d'une de ses opérandes et l'état de l'autre opérande à l'instant t est désactivé ;
- ou $(0,t)$ apparait dans la trace d'une de ses opérandes et l'état de l'autre opérande à l'instant t est activé ;
- ou $(1,t)$ apparait dans chacune des traces des deux opérandes.

— Une paire $(0,t)$ apparait dans ω si et seulement si :

- $(0,t)$ apparait dans la trace d'une de ses opérandes et l'état de l'autre opérande à l'instant t est désactivé ;
- ou $(0,t)$ apparait dans chacune des traces des deux opérandes.

A partir de la trace de sensibilité ω d'une porte OR, de la trace de ses opérandes et des causes racines des défaillances de ses opérandes, les causes racines d'une défaillance dans ω à l'instant t (c.-à-d. $(1,t) \in \omega$) sont obtenues comme suit :

— par les causes racines de la paire $(1,t)$ dans la trace d'une de ses opérandes si l'état de l'autre opérande à l'instant t est désactivé ;

— ou par les causes racines de la défaillance la plus proche dans le passé de l'instant t de l'opérande dont le statut est activé si $(0,t)$ apparait dans la trace de l'autre opérande ;

— ou par l'union des causes racines des défaillances $(1,t)$ apparaissant dans chacune des traces des deux opérandes.

Porte AND

L'analyse des causes racines d'une porte AND nécessite, comme pour la porte OR, de construire préalablement la trace de sensibilité de la sortie de la porte AND à partir des traces de ses opérandes.

La trace de sensibilité w d'une porte AND est définie comme suit à partir des traces (d'exécution ou de sensibilité) de ses opérandes :

— Une paire $(1,t)$ apparait dans ω si et seulement si :

- $(1,t)$ apparait dans la trace d'une de ses opérandes et l'état de l'autre opérande à l'instant t est activé ;
- ou $(1,t)$ apparait dans chacune des traces des deux opérandes.

— Une paire $(0,t)$ apparait dans ω si et seulement si $(0,t)$ apparait dans la trace d'au moins une de ses opérandes.

A partir de la trace de sensibilité ω d'une porte AND, de la trace de ses opérandes et des causes racines des défaillances de ses opérandes, les causes racines d'une défaillance dans ω à l'instant t (c.-à-d. $(1,t) \in \omega$) sont obtenues comme suit :

— par l'union des causes racines de la paire $(1,t)$ dans la trace d'une de ses opérandes avec les causes racines de la défaillance la plus proche dans le passé de l'instant t de l'autre opérande dont le statut est activé à l'instant t ;

— ou par l'union des causes racines des défaillances $(1,t)$ apparaissant dans chacune des traces des deux opérandes.

Exemple

Soit A, B, C trois événements basiques. La figure VII.3 décrit les comportements possibles de ces trois événements basiques et des événements intermédiaires $OR(A, B)$ et $AND(OR(A, B), C)$. Les flèches indiquent les causes racines de chaque défaillance (front montant) dans les événements intermédiaires de l'arbre de défaillances dynamique.

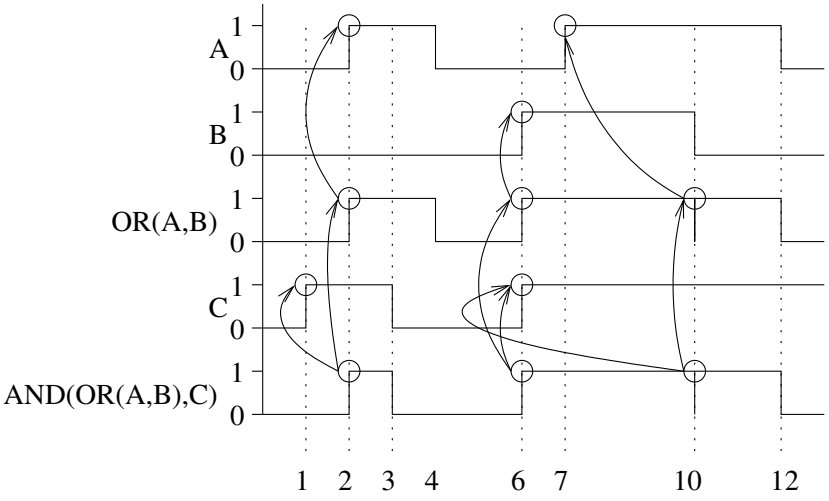


FIGURE VII.3 – Exemples de chronogrammes de portes OR et AND

Les traces d'exécution pour les trois événements basiques A, B, C sont les suivants : $\omega_A = \langle(0,0)(1,2)(0,4)(1,7)(0,12)\rangle$, $\omega_B = \langle(0,0)(1,6)(0,10)\rangle$ et $\omega_C = \langle(0,0)(1,1)(0,3)(1,6)\rangle$.

Les causes racines des défaillances de chaque événement basique sont déduites de leurs traces d'exécution :

— $\mathcal{F}_A = \{(2, A)(7, A)\}$
— $\mathcal{F}_B = \{(6, B)\}$
— $\mathcal{F}_C = \{(1, C)(6, C)\}$

La trace de sensibilité de l'expression booléenne $OR(A, B)$, notée ω_{O_1}, est calculée à partir des traces d'exécutions de ses opérandes ω_A et ω_B : $\omega_{O_1} = \langle(0,0)(1,2)(0,4)(1,6)(1,10)(0,12)\rangle$.

Les causes racines des défaillances de l'événement intermédiaire $OR(A, B)$, notées $\mathcal{F}_{O_1}$, sont calculées à partir de sa trace de sensibilité ω_{O_1} et des causes racines des défaillances de ses opérandes $\mathcal{F}_A$ et $\mathcal{F}_B$: $\mathcal{F}_{O_1} = \{(2, \mathcal{F}_{O_1}(2))(6, \mathcal{F}_{O_1}(6))(10, \mathcal{F}_{O_1}(10))\}$ avec : $\mathcal{F}_{O_1}(2) = \{(2, A)\}$, $\mathcal{F}_{O_1}(6) = \{(6, B)\}$ and $\mathcal{F}_{O_1}(10) = \{(7, A)\}$.

La trace de sensibilité de l'événement intermédiaire $AND(OR(A, B), C)$, notée ω_{A_1}, est calculée à partir de la trace de sensibilité de sa première opérande ω_{O_1} et de la trace de simulation de sa deuxième opérande ω_C : $\omega_{A_1} = \langle(0,0)(1,2)(0,3)(1,6)(1,10)(0,12)\rangle$.

Les causes racines des défaillances de l'événement intermédiaire $AND(OR(A, B), C)$, représentées par $\mathcal{F}_{A_1}$, sont calculées à partir de sa trace de sensibilité ω_{A_1} et des causes racines des défaillances de ses opérandes $\mathcal{F}_{O_1}$ et $\mathcal{F}_C$: $\mathcal{F}_{A_1} = \{(2, \mathcal{F}_{A_1}(2))(6, \mathcal{F}_{A_1}(6))(10, \mathcal{F}_{A_1}(10))\}$

avec : $\mathcal{F}_{A_1}(2) = \mathcal{F}_{O_1}(2) \cup \mathcal{F}_C(1) = \{(2, A)(1, C)\}$, $\mathcal{F}_{A_1}(6) = \mathcal{F}_{O_1}(6) \cup \mathcal{F}_C(6) = \{(6, B)(6, C)\}$ et $\mathcal{F}_{A_1}(10) = \mathcal{F}_{O_1}(10) \cup \mathcal{F}_C(6) = \{(7, A)(6, C)\}$.

2.2.3 Analyse des causes racines des portes dynamiques

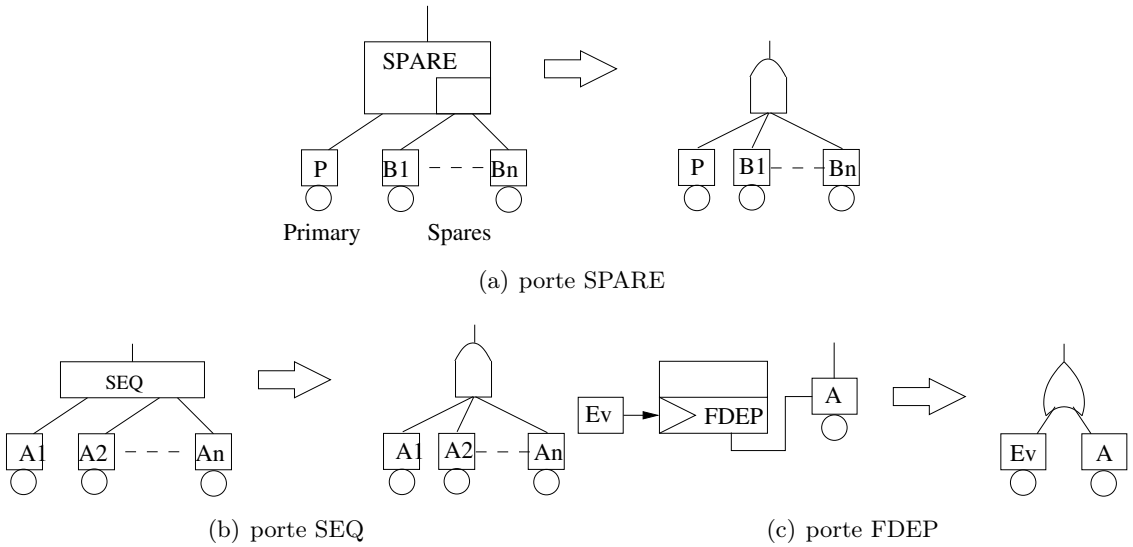
Portes SPARE et SEQ

Une porte SPARE défaille lorsque son entrée principale ainsi que tous ses composants de rechange sont défaillants. Les causes d'un front montant d'une porte SPARE à l'instant t proviennent des causes des fronts montants les plus proches dans le passé de l'instant t de toutes ses opérandes. Les principes de calcul de la trace de sensibilité d'une porte SPARE et de ses causes racines sont donc les mêmes que ceux de la porte AND. Aussi, dans notre méthode d'analyse des causes racines de la porte SPARE, celle-ci est alors substituée par une porte AND. Nous appliquons le même raisonnement pour la porte SEQ, celle-ci est substituée par une porte AND lors de l'analyse de ses causes racines.

Porte FDEP

La porte FDEP modélise une dépendance fonctionnelle entre une gâchette et des événements basiques dépendants. La défaillance de la gâchette, c.-à-d. son activation, entraîne la défaillance des événements basiques dépendants. Dans le cadre de l'analyse des causes racines des défaillances, la porte FDEP peut être substituée par plusieurs portes OR de la manière suivante : chaque événement basique dépendant est remplacé par une porte OR dont les opérandes sont la gâchette et l'événement basique dépendant.

La figure VII.4 présente les règles de substitution des portes SPARE, SEQ et FDEP.



(a) porte SPARE

(b) porte SEQ

(c) porte FDEP

FIGURE VII.4 – Règles de remplacement de portes dynamiques

Porte PAND

Une porte PAND défaille si toutes ses entrées défaillent dans un ordre spécifique : de la gauche vers la droite. Pour simplifier la définition de la trace de sensibilité et des causes racines de la porte PAND, nous considérons une porte PAND à deux entrées : l'entrée de gauche nommée A et l'entrée de droite nommée B.

La trace de sensibilité ω d'une porte PAND est définie comme suit à partir des traces (d'exécution ou de sensibilité) de ses opérandes ω_A et ω_B :

- Une paire $(1, t)$ apparait dans ω si et seulement si :
  - $(1, t)$ apparait dans la trace ω_B de son opérande droite et l'état de son opérande gauche à l'instant t est activé ;
  - ou $(1, t)$ apparait dans chacune des traces des deux opérandes.
- Une paire $(0, t)$ apparait dans ω si et seulement si $(0, t)$ apparait dans la trace d'au moins une de ses opérandes.

A partir de la trace de sensibilité ω d'une porte PAND, de la trace de ses opérandes ω_A et ω_B et des causes racines des défaillances de ses opérandes $\mathcal{F}_A$ et $\mathcal{F}_B$, les causes racines d'une défaillance dans ω à l'instant t sont obtenues comme suit :

- par l'union des causes racines de la paire $(1, t)$ présente dans la trace ω_B de son opérande de droite avec les causes racines de la défaillance la plus proche dans le passé de l'instant t de l'opérande gauche A dont le statut est activé à l'instant t ;
- ou par l'union des causes racines des défaillances $(1, t)$ apparaissant dans chacune des traces des deux opérandes.

2.2.4 Analyse des causes racines d'un arbre de défaillances dynamique

L'analyse des causes racines de l'événement sommet d'un arbre de défaillances dynamique est obtenue récursivement durant un parcours en profondeur postfixe de l'arbre de défaillances dynamique : cela consiste à parcourir son sous-arbre gauche, puis son sous-arbre droit, puis sa racine.

L'algorithme que nous avons développé génère les traces de sensibilité et les causes racines des défaillances pour tous les nœuds de l'arbre de défaillances dynamiques à partir des méthodes d'analyse des causes racines des portes booléennes et dynamiques que nous avons présenté précédemment.

Le lecteur intéressé trouvera dans notre article [280] la présentation détaillée d'un exemple applicatif.

3 Analyse quantitative d'arbres de défaillances dynamiques

3.1 Contexte, positionnement

L'analyse quantitative d'un système permet l'évaluation probabiliste de sa fiabilité. Pour chaque défaillance redoutée représentée par un arbre de défaillances dynamique, son analyse quantitative permettra d'estimer la probabilité de cet événement indésirable.

Nos travaux de recherche ont porté sur le développement d'une méthode d'analyse quantitative basée sur la simulation de Monte-Carlo. L'approche par simulation permet en effet de prendre en considération différentes fonctions de distribution de défaillances et n'est pas limitée aux lois exponentielles comme les approches markovienne. Pour réaliser l'analyse quantitative d'un DFT, les informations nécessaires sont les fonctions de répartition de défaillances $F(t)$ ou les fonctions de densité de probabilité de défaillances $f(t)$ de chaque composant et la valeur de leurs paramètres ainsi que la durée de la mission du système.

Cependant l'approche par simulation de Monte-Carlo fournit des résultats approchés, pour avoir un résultat dans un niveau de confiance élevé (90% ou 95%) avec une marge d'erreur faible ($< 2\%$), il est nécessaire d'avoir un très grand nombre de simulations, ce qui peut prendre un temps de simulation important. *Nos travaux proposent d'utiliser la simulation conduite par les événements pour accélérer le processus de simulation de Monte-Carlo.*

Avec notre méthode de simulation basée sur les événements, nous considérons toutes les portes dynamiques (PAND, SEQ, FDEP et SPARE) et nous n'avons aucune restriction sur leurs usages :

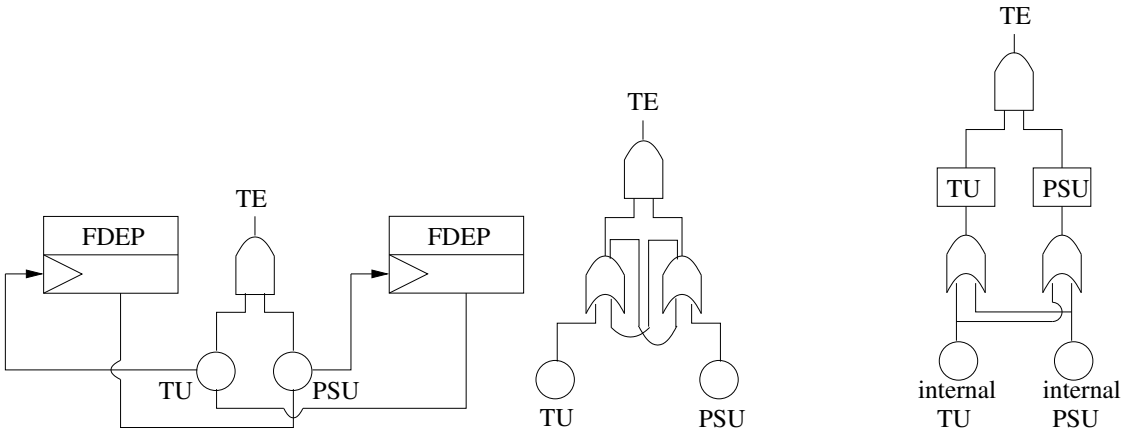
- Nous prenons en compte les composants connectés à plusieurs portes PAND ou SEQ ou FDEP ou SPARE. Nous appelons dans la suite les événements associés à ces composants, les événements répétés.
- Nous prenons en compte les composants de rechange partagés entre plusieurs portes SPARE. Nous appellerons dans la suite les événements associés à ces composants, les événements de rechange partagés.
- Nous prenons en compte les dépendances circulaires de défaillances entre composants : défaillances en cascade circulaire causée par des portes FDEP.

Nos travaux ont porté sur des systèmes ne comportant que des composants non réparables. Nous nous sommes intéressés à l'évaluation de la fiabilité avant l'apparition de la première défaillance. Cependant, les arbres de défaillances dynamiques que notre méthode prend en compte peuvent utiliser des portes SPARE pour modéliser des systèmes avec redondance de composants.

Dans le reste de cette section, nous illustrons différentes problématiques d'usage des portes dynamiques et comment notre méthodologie les ont prises en compte.

Dépendances circulaires avec les portes FDEP

Les défaillances en cascade peuvent être exprimées dans un arbre de défaillances dynamique en utilisant de multiples portes FDEP. Selon les interconnexions entre les portes FDEP, des dépendances circulaires entre les événements de défaillances de plusieurs composants peuvent alors être introduites. Considérons l'exemple proposé dans [286] : un bloc d'alimentation PSU qui doit être refroidi par un bloc thermique TU. Une défaillance du bloc thermique TU entraîne la défaillance du bloc d'alimentation PSU. Comme le bloc d'alimentation PSU alimente le bloc thermique TU, une défaillance du bloc d'alimentation PSU entraîne une défaillance du bloc thermique TU. Cette interdépendance est



(a) DFT original

(b) DFT avec dépendance circulaire

(c) DFT sans dépendance circulaire

FIGURE VII.5 – Dépendance circulaire introduite par les portes FDEP

représentée sur la figure VII.5(a).

S'il n'y a pas de dépendance circulaire dans l'arbre de défaillances dynamique causée par des portes FDEP, chaque porte FDEP peut être représentée par une porte équivalente OR à deux entrées : à la première entrée est connecté l'événement dépendant, à la seconde entrée est connectée la gâchette ([215]).

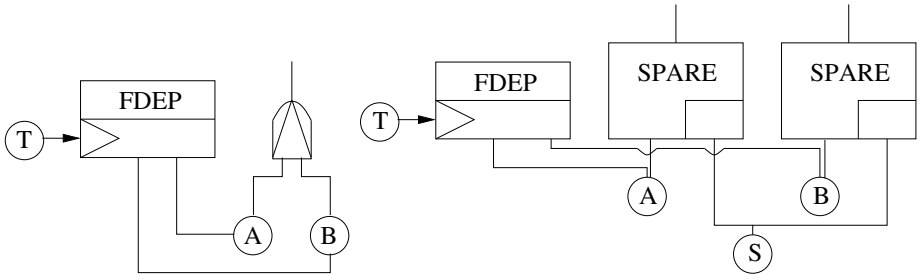
Cependant, cette règle de transformation ne peut pas s'appliquer pour des portes FDEP interdépendantes, des dépendances circulaires resteront présentes dans l'arbre de défaillances dynamique réécrit comme le montre la figure VII.5(b).

Pour supprimer les dépendances circulaires, l'astuce suivante est proposée ([200]) : les défaillances du bloc thermique TU et du bloc d'alimentation PSU sont divisées en défaillances internes et défaillances causées par la gâchette qui deviennent des événements intermédiaires (figure VII.5(c)).

Les méthodes de simulation des DFT dont le principe de propagation des défaillances dépend de la topologie du DFT ne permettent pas la présence de dépendances circulaires causées par des portes FDEP sans appliquer cette astuce de réécriture du DFT comme cela est expliqué dans [262]. Avec des simulations par pas de temps (time-driven) ou conduites par les événements (event-driven) qui considèrent les événements dans leur ordre d'apparition et non selon leurs positions dans le DFT, de telles dépendances circulaires peuvent prises en compte sans que le DFT soit réécrit.

Simultanéité et non-déterminisme introduits par les portes FDEP

Dans [203], [232], la problématique des portes FDEP avec de multiples composants dépendants est abordée. Comment gérer l'ordre d'apparition des défaillances sur les composants : en séquence ou simultanément ? Les exemples fournis dans [203], [232] et illustrés sur la figure VII.6 montrent les cas limites. Dans les deux arbres de défaillances dynamiques, une porte FDEP modélise que l'arrivée de la défaillance de la gâchette T entraîne la défaillance



(a) Portes FDEP et PAND (b) Portes FDEP et SPARE

FIGURE VII.6 – Exemples de situations non-déterministes causées par la porte FDEP

des composants A et B. Les problématiques soulevées par ces deux exemples sont :

— Quel sera l'état de la porte PAND dans la figure VII.6(a) ?

— Quelle porte SPARE activera le composant de rechange partagé S dans la figure VII.6(b) ?

Comme dans ces travaux, nous considérons que la gâchette T déclenche simultanément les défaillances sur les composants A et B et qu'elle introduit du non-déterminisme. Dans notre méthodologie de simulation pour l'analyse quantitative des arbres de défaillances dynamiques, nous avons pris en compte ce non-déterminisme en le traduisant par un choix équiprobable.

Portes SPARE avec un composant de rechange partagé

L'utilisation de composants de rechange partagés est usuelle dans les systèmes tolérants aux pannes qui utilisent la redondance des composants. La figure VII.7 montre un exemple d'un composant de rechange S avec deux composants principaux A et B. Si un composant principal partage un composant de rechange avec d'autres composants principaux, il sera remplacé par le composant de rechange seulement s'il défaille avant les autres composants principaux. En conséquence, une porte SPARE composée d'un composant principal et d'un unique composant de rechange partagé S défaille si son composant principal est défaillant et que son composant de rechange ne peut pas ou plus être utilisé pour le remplacer. Sur notre exemple, la porte SPARE avec le composant principal A est défaillant lorsque le composant A est défaillant et que 1) le composant de rechange S est également défaillant ou 2) le composant principal B de la seconde porte SPARE a défailli avant A et utilise alors le composant de rechange S qui n'est plus disponible pour remplacer A.

Notre méthode d'analyse quantitative de DFT prend en compte les composants de rechange partagés.

Modélisation de la porte SEQ

Différents travaux de recherche (e.g. [221], [252], [277]) considèrent que la porte SEQ est équivalente à une porte CSP (porte SPARE avec $\alpha = 0$). Cette équivalence est valide

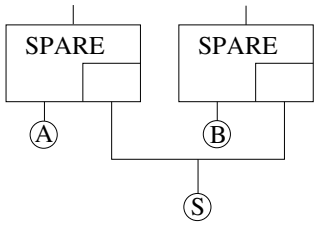


FIGURE VII.7 – Exemple de deux portes SPARE partageant un composant de rechange

quand les composants contraints par la porte SEQ ne sont pas connectés à d'autres portes ([276]).

Les deux scénarios suivants montrent la nécessité de modéliser la porte SEQ séparément de la porte CSP quand des composants sont partagés.

Dans un premier scénario, supposons qu'un composant A est à la fois l'entrée primaire d'une porte SEQ et la seconde entrée d'une autre porte SEQ. Le temps de défaillance de A sera alors contraint par le temps de défaillance du composant connecté à l'entrée primaire de la porte SEQ où A apparait comme entrée secondaire. Cependant, cette situation ne peut pas être prise en compte si les deux portes SEQ ont été remplacées par des portes CSP : A ne peut pas simultanément être un composant principal dans la première porte CSP et un composant de rechange dans la seconde porte CSP. La défaillance d'un composant principal peut arriver à n'importe quel moment (et selon sa distribution de défaillances) alors que la défaillance d'un composant de rechange ne peut arriver qu'à partir de son activation (et également en respectant sa distribution de défaillances).

Dans un second scénario, supposons qu'un composant A apparait comme entrée secondaire de deux portes SEQ. Avec le remplacement des deux portes SEQ par des portes CSP, A deviendrait un composant de rechange partagé. Comme conséquence, lorsque le composant principal d'une porte CSP serait défaillant, il serait remplacé par le composant de rechange partagé A. Aussi, le composant de rechange A deviendrait actif et pourrait rencontrer une défaillance avant le composant principal relié à l'autre porte CSP, cependant cette situation n'est pas permise lorsqu'on considère les portes SEQ.

Notre méthode d'analyse quantitative prend en compte des arbres de défaillances dynamiques avec des composants connectés à plusieurs portes SEQ. Pour cela nous avons formalisé le comportement de la porte SEQ séparément, et nous ne l'avons pas considéré comme une poste CSP.

3.2 Analyse quantitative des arbres de défaillances dynamiques par simulation de Monte-Carlo

3.2.1 Principe de l'analyse quantitative par simulation de Monte Carlo

La simulation de Monte-Carlo est une méthode d'estimation d'une valeur numérique en utilisant des tirages aléatoires. La simulation de Monte-Carlo est utilisée pour fournir des solutions approchées à des problèmes d'ingénierie complexes dont il n'est pas toujours

possible d'avoir une solution exacte. Elle est utilisée dans un grand nombre de domaines (industrie automobile, aéronautique, énergie, nucléaire, logistique, santé, etc.).

La simulation de Monte-Carlo appliquée à l'analyse quantitative d'un arbre de défaillances dynamique est décrite par les algorithmes VII.1 et VII.2 (c.f. section 3.2.3). Son principe est le suivant :

1. Soit N et T_m le nombre de simulations et le temps de mission du système (temps pendant lequel l'occurrence de la défaillance peut se produire). Initialiser n_{TE} le nombre d'occurrence de l'événement sommet (événement redouté) (*Top Event*) du DFT à zéro. Pour N simulations, exécuter les étapes suivantes :
   (a) Calculer les temps de défaillance des composants du DFT selon leurs lois de défaillances (exprimées par la fonction cumulative de distribution $F(t)$ et la fonction de densité de probabilité $f(t)$) et leurs rôles dans les portes dynamiques telles que SEQ, SPARE et FDEP. Si le temps de défaillance est ultérieur à T_m le temps de mission du système, le composant associé à l'événement basique est considéré comme opérationnel et n'est pas défaillant.
   (b) Calculer les temps de défaillances des événements intermédiaires (sorties des portes) selon leurs sémantiques et les temps de défaillances de leurs opérandes.
   (c) Calculer le temps de défaillance de l'événement sommet. Si ce temps de défaillance est inférieur ou égal au temps de mission du système, incrémenter n_{TE}.
2. Calculer l'estimation de la probabilité de l'événement sommet : $p_{TE} = \frac{n_{TE}}{N}$.
3. Estimer le taux d'erreur : Soit LC le niveau de confiance désiré (habituellement entre 90% et 95%). Le taux d'erreur de la probabilité p_{TE} est défini par :

$$\varepsilon\% = 100 \times \mathcal{U}_{1-\alpha/2} \times \sqrt{\frac{1 - p_{TE}}{N \times p_{TE}}}$$

où $\alpha = 1 - LC/100$ et $\mathcal{U}_{1-\alpha/2}$ est le $(1 - \alpha/2)$ quantile de la loi de distribution normale. Cela signifie qu'avec une vraisemblance de $LC\%$, la probabilité de la défaillance de l'événement redouté est dans l'intervalle $[p_{TE} - \frac{p_{TE} \times \varepsilon}{100}, p_{TE} + \frac{p_{TE} \times \varepsilon}{100}]$ avec N simulations de Monte-Carlo.

3.2.2 Génération des temps de défaillances des composants

Dans cette section, nous relatons comment sont générés les temps de défaillances des composants. Nous distinguerons trois situations :

— Les composants dont les temps de défaillances ne dépendent pas des temps de défaillances d'autres composants. Les événements relatifs à ces composants sont appelés dans la suite *événements basiques actifs*.

— Les composants de rechange connectés aux entrées secondaires des portes SPARE. Les événements relatifs à ces composants de rechange sont appelés dans la suite *événements de rechange*.

— Les composants qui sont connectés à des entrées secondaires de portes SEQ. Les événements relatifs à ces composants contraints par des portes SEQ sont appelés dans la suite *événements contraints*

Nous supposons que nous avons pour chaque composant sa fonction de répartition de défaillances $F(t)$ et qu'elle est inversible.

Génération des temps de défaillances des événements basiques actifs
Nous considérons un composant X qui n'est pas utilisé comme composant de rechange dans une poste SPARE, ni contraint par une porte SEQ. Nous représentons sa fonction de distribution de défaillance par $F_i(t) = P(X \leq t)$, c.-à-d. la probabilité que le composant X soit défaillant avant l'instant t. L'indice i indique l'indépendance statistique sur le composant X. La fonction de densité de probabilité est notée $f_i(t)$. Pour générer aléatoirement le temps de défaillance du composant X à partir de $F_i(t)$, nous générons un réel aléatoire U entre 0 et 1 associé à $F_i(t)$. Puis, nous utilisons la fonction inverse $F_i^{-1}(U)$ qui fournit le temps de défaillance t_U pour le composant X. Si t_U est antérieur au temps de mission T_m, alors le composant X est considéré comme défaillant à partir de l'instant t_U. Sinon, le composant n'a pas rencontré de défaillance durant son temps de mission.

Génération des temps de défaillances des événements de rechange
Nous considérons un composant de rechange B qui sert à remplacer un composant principal ou un autre composant de rechange A. Nous représentons par t_a le temps de défaillance du composant A, c.-à-d. le temps d'activation du composant de rechange B. Le temps de défaillance d'un composant de rechange est calculé lorsqu'il est appelé à remplacer un autre composant, nous supposons donc que nous avons connaissance du temps d'activation t_a pour calculer le temps de défaillance du composant de rechange B.

La fonction de répartition de défaillances de B est représentée dans différentes conditions d'utilisation de B. Lorsque B n'est pas utilisé comme un composant de rechange, sa fonction de répartition est notée $F_i(t)$. Lorsque B est un composant de rechange en attente (dormant), sa fonction de répartition est notée $F_d(t)$ (fonction de répartition dormante) et est définie par $F_d(t) = 1 - (1 - F_i(t))^\alpha$ où α est la température de la porte SPARE. Lorsque B est un composant de rechange actif sa fonction de répartition est notée $F_a(t)$ (fonction de répartition active) et est définie par $F_a(t) = F_i(t) \times (1 - F_i(t_a))^{\alpha-1}$. Finalement, la fonction de répartition conditionnelle $F(t|t_a)$ représente la fonction de répartition de défaillances du composant de rechange B (en mode actif ou dormant) quand t_a est son temps d'activation. La fonction de répartition conditionnelle $F(t|t_a)$ est définie par $F(t|t_a) = \mathrm{u}(t_a - t) \times F_d(t) + \mathrm{u}(t - t_a) \times (F_d(t_a) + F_a(t) - F_a(t_a))$ avec u la fonction Step définie par :

$$\mathrm{u}(t-\tau) = \begin{cases} 0 \text{ , if } t < \tau \\ \frac{1}{2} \text{ , if } t = \tau \\ 1 \text{ , if } t > \tau \end{cases}$$

Les étapes de calcul du temps de défaillance t_U du composant de rechange B sont les

suivantes :

1. Nous générons un réel aléatoire U entre 0 et 1 associé à $F(t_U|t_a)$.
2. Si $F_d^{-1}(U) \leq t_a$, alors le composant de rechange a rencontré une défaillance lorsqu'il était en attente et nous prenons $t_U = F_d^{-1}(U)$ comme temps de défaillance.
3. Sinon ($F_d^{-1}(U) > t_a$), le composant de rechange rencontre une défaillance lorsqu'il est actif. Nous définissons $U = F_d(t_a) + F_a(t_U) - F_a(t_a)$ à partir de la définition de $F(t|t_a)$ et nous calculons $t_U = F_a^{-1}(U - F_d(t_a) + F_a(t_a))$.
4. Si t_U est antérieur au temps de mission T_m, alors le composant de rechange B est considéré comme défaillant à partir à partir de l'instant t_U. Autrement, le composant de rechange n'a pas rencontré de défaillance pendant le temps de mission.

Génération des temps de défaillances des événements contraints

Nous considérons un composant B relié à une ou plusieurs portes SEQ et apparaissant comme entrée secondaire d'au moins l'une d'entre elles. Soit n le nombre de contraintes appliquées au composant B par les différentes portes SEQ auxquelles il est connecté. Nous représentons par $C_1, C_2, \ldots, C_n$ les composants qui précèdent immédiatement B dans les différentes portes SEQ auxquelles B est reliée en entrée secondaire. Avec la simulation conduite par les événements que nous utilisons, les événements de défaillances sont traités dans l'ordre chronologique. Nous représentons par t_c le temps de défaillance de la dernière (au sens temporel) contrainte appliquée sur B à être satisfaite. A cause de la sémantique de la porte SEQ, le composant B ne peut pas rencontrer de défaillance avant le temps t_c.

Nous représentons par $F(t|t_c)$ la fonction de répartition conditionnelle de B quand t_c est la dernière contrainte temporelle à s'appliquer sur B. $F(t|t_c)$ est définie par : $F(t|t_c) = \mathrm{u}(t - t_c) \times F_i(t - t_c)$ avec u la fonction Step définie précédemment.

Nous calculons le temps de défaillance d'un composant contraint B à partir de sa dernière contrainte temporelle t_c ainsi : nous générons un réel aléatoire U compris entre 0 et 1 et nous l'associons à $F_i(t_U - t_c)$. Ainsi $F_i^{-1}(U) + t_c$ nous fournit le temps de défaillance de B. Si $t_U = F_i^{-1}(U) + t_c$ est antérieur au temps de mission T_m, alors B est considéré comme défaillant à partir de t_U. Sinon, le composant contraint B n'a pas rencontré de défaillance pendant le temps de mission.

3.2.3 Analyse quantitative de DFT par simulation conduite par les événements

Dans cette section, nous présentons le principe de la simulation conduite par les événements appliquée à l'analyse quantitative d'un arbre de défaillances dynamique. A chaque itération de la simulation de Monte-Carlo, nous commençons par générer aléatoirement les temps de défaillances des composants actifs. Puis nous générons aléatoirement les temps de défaillances des composants de rechange et des composants contraints et nous calculons les temps de défaillances des événements intermédiaires (sorties des portes) et propageons ceux-ci afin de calculer le temps de défaillance de l'événement sommet.

Avec la simulation guidée par les événements, un ordonnanceur (*scheduler*) est utilisé pour traiter dans l'ordre chronologique les temps de défaillances des différents événements. Tous les événements de défaillances sont mémorisés dans une structure de données de type file d'attente Q. La progression du temps courant de simulation t_{sim} dépend du temps du prochain événement de défaillance e à traiter qui se trouve en tête de la file d'attente Q. On défile la file file d'attente Q ($e = Pop(Q)$) en défilant l'événement e de celle-ci. Le temps courant de simulation t_{sim} est alors avancé à ce temps de défaillance. L'événement de défaillance est alors traité : on étudie toutes les portes auxquelles cet événement est connecté et s'il déclenche un événement de défaillance sur la sortie de certaines portes, on calcule leurs temps de défaillance, et on enfile ces nouveaux événements de défaillances dans la file Q. Lorsqu'on traite un événement de défaillance, si celui-ci ne génère pas sur les portes auxquelles il est connecté de défaillance, alors la file Q n'est pas modifiée et par conséquent les portes reliées à celles-ci ne seront pas évaluées, ce qui apporte un gain de temps de calcul.

Algorithme VII.1 Simulation conduite par les événements d'un DFT

procedure OneStepMonteCarloDFT
Input: S : collection of nodes of the DFT
Input: A : collection of active BE
Input: T_m : mission time
Ensure: One-step Monte Carlo simulation of the DFT
1: Initialize the event queue Q with failure events of active basic events
2: $t_{sim} := 0$
3: **while** Q not empty and $t_{sim} \leq T_m$ **do**
4: $e := Pop(Q)$ (get unsimulated event with earliest time in the queue)
5: $t_{sim} := time(e)$
6: simulate event e
7: **for each** new event e' generated **do**
8: $Push(Q, e')$ (insert new event adequately to the event queue)
9: **end for**
10: **end while**

Dans notre article [281], nous détaillions toutes les structures de données et les algorithmes que nous avons développés pour l'analyse quantitative d'un arbre de défaillances dynamique.

3.2.4 Expérimentations

Nous avons implémenté nos algorithmes en langage de programmation Java. Nous avons comparé les fonctionnalités et les performances de notre logiciel appelé DFTEDS (DFT Event Driven Simulation) à trois autres outils : DFTCalc, DFTSim et MatCarloRE en utilisant cinq études de cas.

DFTCalc [234] utilise une approche basée sur des systèmes à espaces d'états : il convertit

Algorithme VII.2 Analyse quantitative d'un DFT par simulation de Monte Carlo

procedure QUANTITATIVEANALYSISDFT
Input: S : collection of nodes of the DFT
Input: TE : root node (Top Event) of the DFT
Input: T_m : mission time
Input: N : number of Monte Carlo simulation runs
Input: LC : desired level of confidence
Ensure: Monte Carlo simulation for quantitative analysis of the DFT
1: Computation of the active basic events A
2: **for** $i := 1$ **to** N **do**
3: ONESTEPMONTECARLODFT(S, A, T_m) {cf. **Algorithme VII.1**}
4: **end for**
5: **for each** $s \in S$ **do**
6: $p_s := \frac{s.\#failures}{N}$ {approx. failure probability of s}
7: **end for**
8: $\alpha := 1 - LC/100$ {half-width confidence interval}
9: $\varepsilon\% = 100 \times \mathcal{U}_{1-\alpha/2} \times \sqrt{\frac{1-p_{TE}}{N \times p_{TE}}}$ {error analysis of p_{TE}}
end procedure

le DFT en une chaîne de Markov interactive (Input/Output interactive Markov chain [233]) et utilise des techniques de vérification de modèle stochastique. Cet outil prend en considération des distributions exponentielles, il ne prend pas en compte les événements réparables. Nous avons utilisé l'interface web de cet outil, il prend en entrée la description du DFT au format Galileo [287].

DFTSim [262] utilise la simulation de Monte-Carlo pour réaliser l'analyse quantitative du DFT. Cet outil transforme le DFT décrit au format Galileo en modèle Matlab. L'approche utilisée par cet outil est la suivante : il calcule les temps de défaillance des événements basiques puis les propage dans le DFT, par des fonctions récursives, pour calculer les temps de défaillances des événements intermédiaires et l'événement sommet (les sorties des portes). DFTSim n'a pas de limitation sur les lois de distribution des défaillances si leurs fonctions inverses sont définies dans Matlab. Cet outil ne prend pas en compte les événements réparables.

MatCarloRE [264] est un outil qui utilise également la simulation de Monte-Carlo et est implémenté dans Matlab/Simulink. Leurs auteurs ont développé une librairie Simulink pour les DFT qui permet de créer un modèle Simulink d'un DFT et de réaliser son analyse quantitative. Cet outil également n'a pas de limitation sur les lois de distribution des défaillances si leurs fonctions inverses sont définies dans Matlab et ne prend pas en compte les événements réparables.

La table 1 présente les performances de notre outil DFTEDS sur six études de cas et les compare aux performances des trois autres outils présentés ci-dessus.

Les études de cas sont le *cascaded PAND system (CPS)* [232], le *cardiac assist system*

(CAS) [262], le *multi-processor distributed computer system (MDCS)* [232], le *section of an Alkylation plant (SAP)* [288], le *fault tolerant parallel processors (FTPP)* [199] et le *Weibull FTPP*.

Les figures VII.8, VII.9, VII.10, VII.11 et VII.12 présentent les arbres de défaillances dynamiques de ces études de cas.

Les études de cas CPS, CAS, MDCS, SAP et FTPP utilisent des lois de distributions exponentielles ou des taux de défaillances constants. L'étude de cas Weibull FTPP est le même que l'étude de cas FTPP excepté que les lois de défaillances sont des lois de Weibull.

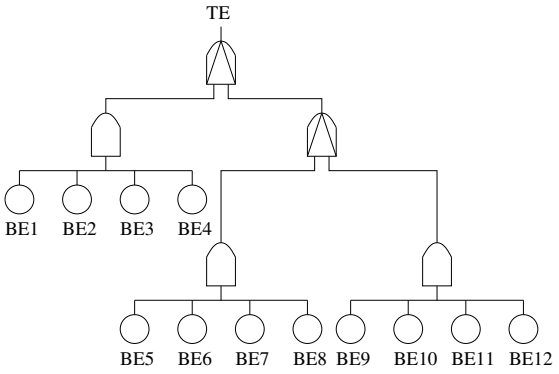


FIGURE VII.8 – DFT of the CPS [232]

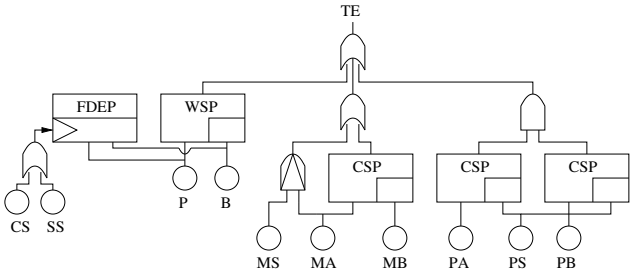


FIGURE VII.9 – DFT of the CAS [262]

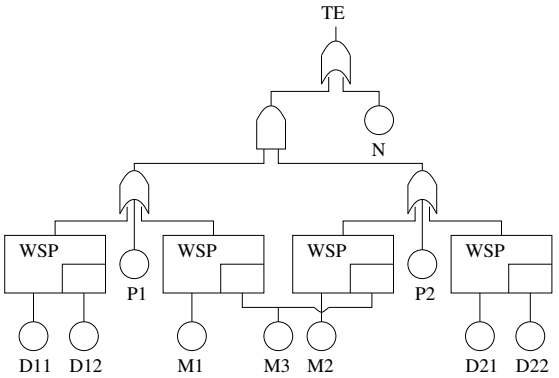


FIGURE VII.10 – DFT of the MDCS [232]

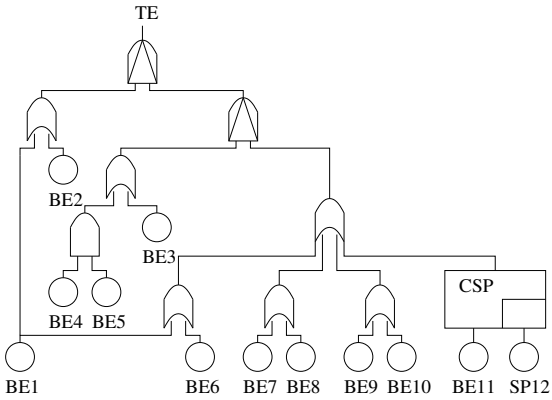


FIGURE VII.11 – DFT of the SAP [288]

Les résultats présentés dans la table 1 montrent l'efficacité de notre approche et ses performances en temps de temps de calcul. Nos résultats de simulation qui sont égaux ou supérieurs aux autres travaux existants s'expliquent ainsi :

— Dans notre approche, nous n'utilisons pas une transformation en systèmes à espaces d'états, aussi notre approche n'est pas confrontée au problème de l'explosion du

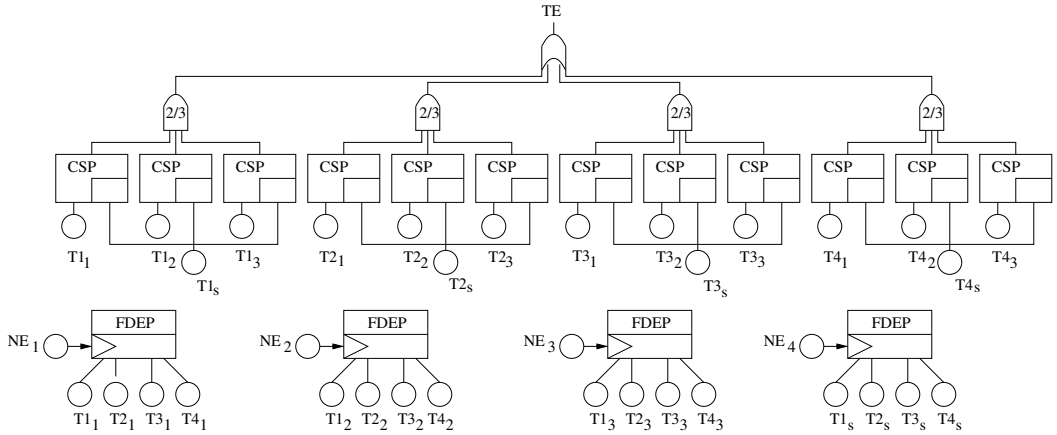


FIGURE VII.12 – DFT of the FTPP [199]

TABLE 1 – Analyse quantitative de DFT : Résultats des études de cas

Case study	Tool	#iterations	Unreliability	Relative error with 95% confidence	Comp. time (sec)
CPS ($T_m = 1h$)	DFTSim	10^5	0.00142	16%	40
	MatCarloRE	10^5	0.00140	16.5%	17
	DFTCalc web-tool	-	0.00135	-	54.95
	DFTEDS (our tool)	$\mathbf{10^5}$	**0.00136**	**16.7%**	**0.252**
CAS ($T_m = 1h$)	DFTSim	10^5	0.65651	0.44%	43
	MatCarloRE	10^5	0.65770	0.44%	64
	DFTCalc web-tool	-	0.6579	-	85.98
	DFTEDS (our tool)	$\mathbf{10^5}$	**0.65767**	**0.44%**	**0.291**
MDCS ($T_m = 1h$)	DFTSim	10^5	0.06737	2.30%	39
	MatCarloRE	10^5	0.06680	2.31%	52
	DFTCalc web-tool	-	0.06664	-	63.46
	DFTEDS (our tool)	$\mathbf{10^5}$	**0.06663**	**2.31%**	**0.163**
SAP ($T_m = 8760h$)	DFTSim	not tested			
	MatCarloRE	10^6	0.000185	14.40%	278
	DFTCalc web-tool	not accepted due to fixed probability			
	DFTEDS (our tool)	$\mathbf{10^6}$	**0.00017665**	**14.74%**	**3.48**
FTPP ($T_m = 1h$)	DFTSim	10^5	0.01981	4.35%	98
	MatCarloRE	not tested			
	DFTCalc web-tool	-	0.0192186	-	277.16
	DFTEDS (our tool)	$\mathbf{10^5}$	**0.0191557**	**4.43%**	**0.269**
Weibull FTPP ($T_m = 1h$)	DFTSim	10^5	0.01292	5.41%	97
	MatCarloRE	not tested			
	DFTCalc web-tool	not accepted due to Weibull distribution			
	DFTEDS (our tool)	$\mathbf{10^5}$	**0.0128746**	**5.42%**	**0.305**

nombre d'états à explorer comme l'outil DFTCalc.

— Notre méthode qui utilise la simulation conduite par les événements évalue le fonctionnement d'une porte seulement quand l'une de ses entrées reçoit un événement de défaillance. Ce n'est pas le cas pour l'outil DFTSim qui évalue chaque porte du DFT à chaque itération de la simulation de Monte-Carlo même si des portes ne reçoivent pas d'événements de défaillances en entrées.

— Avec notre méthode de simulation conduite par les événements, les périodes d'inactivités entre les événements de défaillances ne sont pas considérées. Cela augmente la vitesse de calcul de notre outil, ce qui n'est pas le cas de l'outil MatCarloRE qui utilise une simulation par pas de temps.

4 Simulation d'arbres de défaillances dynamiques : application au filtrage de fausses alarmes

4.1 Contexte, positionnement

Les systèmes ou équipements modernes, composés de composants électroniques, ont la capacité de s'autotester grâce au Built-In-Test (BIT). Les informations issues des BIT d'un système dynamique complexe peuvent être exploitées pour réaliser son diagnostic, mais nécessitent un traitement préalable pour filtrer les fausses alarmes. C'est par exemple la situation dans le secteur aéronautique où les équipements peuvent générer un grand nombre de fausses alarmes à des phases particulières du vol (décollage, atterrissage). Ces fausses alarmes provoquent alors aux équipes de maintenance des problèmes d'ambiguïtés, de localisation qui retardent les actions de maintenance ou provoquent des opérations de maintenance inutiles, ou demandent des procédures de test additionnelles pour localiser une défaillance.

Pour améliorer les opérations de localisation des défaillances, nous proposons d'utiliser les arbres de défaillances temporels et dynamiques pour exprimer des règles de filtrage de fausses alarmes. En effet, les équipes de maintenance exploitent des corrélations entre différentes alarmes pour filtrer les fausses alarmes et lever des ambiguïtés de localisation. Les arbres de défaillances dynamiques avec leurs portes PAND, SEQ, FDEP et SPARE permettent d'exprimer des comportements séquentiels et des dépendances fonctionnelles entre différents composants, cependant ces portes ne permettent pas de capturer des dépendances temporelles ou des événements intermittents, Lefebvre et al. proposent dans [207] de nouvelles portes temporelles à cet effet (PANDW, DUR, COUNT).

Il existe différents outils d'analyse des arbres de défaillances temporels ou dynamiques (DIFTree [289], Galileo [287], Radyban [252], DFTSim [262], Pandora [250], [290], MatCarlo [264], DFTCalc [234], RAATSS [265], Altarica [291], [292]), cependant ceux-ci sont axés sur l'analyse qualitative et quantitative des arbres de défaillances temporels ou dynamiques. Pour mettre en œuvre ces règles de filtrage de fausses alarmes, exprimées par des arbres de défaillances temporels et dynamiques, nous proposons dans [282] un outil sous Matlab/Simulink/Stateflow qui permet de modéliser les arbres de défaillances temporels et dynamiques et de simuler leurs comportements en fonction d'un historique des événements basiques. Notre outil permettra ainsi d'implémenter des règles de filtrages en utilisant les arbres de défaillances. Nous présentons dans la suite de cette section, la démarche de formalisation de portes dynamiques et temporelles dans le cadre d'événements intermittents (pour la prise en compte de fausses alarmes) que nous avons suivie, la toolbox sous Matlab/Simulink.Stateflow que nous avons développée permettant de simuler des arbres de défaillances temporels et dynamiques et un exemple de filtrage de fausses alarmes.

4.2 Formalisation des portes temporelles et dynamiques avec des événements intermittents

Les portes dynamiques et temporelles lorsqu'elles ont été introduites, ont été définies de manière informelle en langage naturel. Leurs implémentations dans des outils logiciels nécessitent une formalisation de leurs sémantiques. Dans cette section, tout d'abord nous définissons leurs sémantiques en termes de traces qui satisfont leurs comportements en utilisant le prédicat de satisfaction $\vdash$. Puis, à chaque porte dynamique ou temporelle est associé un automate de Moore qui respecte sa sémantique. Les automates de Moore qui sont des machines d'états dont la fonction de génération ne dépend que de l'état courant peuvent être facilement implémentés en utilisant la bibliothèque Stateflow de Matlab/Simulink.

Considérons un ensemble fini de n événements de défaillance $\mathcal{F} = \{x_1, \ldots, x_n\}$ qui prennent des valeurs dans le domaine booléen $\mathbb{B}$. L'occurrence d'une défaillance est représentée par 1 et sa non-occurrence par 0. Une trace ω sur $\mathcal{F}$ est définie comme une séquence de valuation pour tous les événements de défaillances appartenant à $\mathcal{F}$. Nous supposons que les événements de défaillances peuvent être intermittents afin de considérer les fausses alarmes et que le temps est discretisé. Une discrétisation du temps est représentée par une séquence numérotée de *ticks*. Une trace sur un ensemble de n événements de défaillances est alors une fonction de $\mathbb{N}$ vers $\mathbb{B}^n$. L'occurrence de l'événement de défaillance x_i dans la trace ω au temps t est représentée par $\omega^t \vdash x_i$ (la trace ω au temps t satisfait x_i).

Dans la suite de cette section nous expliquons comment nous avons défini les sémantiques de la porte dynamique PAND et de la porte temporelle DUR et leurs représentations sous forme d'automates de Moore. Le lecteur trouvera dans [282] la présentation des sémantiques des autres portes dynamiques (SEQ et FDEP) et temporelles (PANDW).

4.2.1 Porte PAND

Nous considérons une porte PAND à deux entrées. La modélisation d'une porte PAND avec un plus grand nombre d'entrées est obtenue par combinaison de portes PAND à deux entrées. Dugan et al. [199] définissent la porte PAND comme équivalente à la porte AND avec la condition additionnelle que les événements de défaillance doivent apparaitre sur les entrées de la porte PAND dans un ordre spécifique : de gauche à droite. La sortie de la porte PAND est ainsi vraie si tous les événements sur ses entrées sont présents et que l'événement de l'entrée gauche apparait avant l'événement de l'entrée droite. Si aucun événement n'apparait ou si l'événement d'entrée droite apparait avant l'événement d'entrée gauche, alors la sortie de la porte PAND est fausse. La figure VII.13 illustre le comportement de la porte PAND.

Soit a et b les entrées gauche et droite de la porte PAND. La sortie de la porte PAND pour une trace ω sur les événements a et b au temps t est définie comme suit :

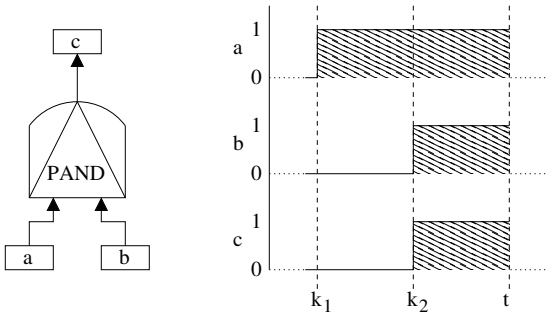


FIGURE VII.13 – Chronogramme de la porte PAND

$$\omega^t \vdash PAND(a,b) \Longleftrightarrow \exists k_1 < k_2 < t \cdot \left(\begin{array}{ll} & \omega^{k_1} \nvdash a \wedge \omega^{k_1} \nvdash b \\ \wedge & \forall j \in]k_1 \ldots k_2] \cdot (\omega^j \vdash a \wedge \omega^j \nvdash b) \\ \wedge & \forall j \in]k_2 \ldots t] \cdot (\omega^j \vdash a \wedge \omega^j \vdash b) \end{array} \right)$$

L'automate de Moore associé à la porte PAND et implémentant cette sémantique de trace est représenté sur la figure VII.14.

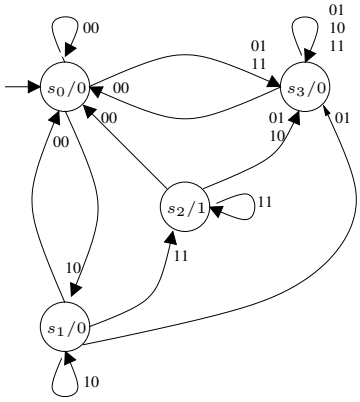


FIGURE VII.14 – Automate de Moore de la porte PAND

4.2.2 Porte DUR

Les événements de défaillances ne sont pas forcément persistants et peuvent être intermittents. Il est alors utile de définir une porte temporelle pour détecter le caractère intermittent d'une défaillance. La porte DUR compare la durée d'une défaillance (temps entre le front montant où la défaillance apparait et le front descendant où la défaillance disparait) à un seuil T, si la durée est supérieure ou inférieure au seuil T, la sortie de la porte DUR sera alors vraie. Deux situations peuvent être représentées :

— La porte $DUR_>$ génère une défaillance si son entrée a une durée de défaillance supérieure à T unités de temps.

— La porte $DUR_<$ génère une défaillance si son entrée à une durée de défaillance inférieure à T unités de temps.

La sortie de la porte $DUR_>$ avec le seuil T pour une trace ω sur les événements a et b au temps t est définie par les deux scénarios suivants :

$$\omega^t \vdash DUR_>(T,a) \overset{①}{\Longleftrightarrow}$$
$$\exists k_1 < k_2 < t \ \cdot \ \left(\begin{array}{ll} & \omega^{k_1} \nvdash a \\ \wedge & \forall j \in]k_1 \ldots k_2] \cdot (\omega^j \vdash a) \\ \wedge & k_2 - k_1 = T \\ \wedge & \forall j \in]k_2 \ldots t] \cdot (\omega^j \vdash a) \end{array} \right)$$
$$\omega^t \vdash DUR_>(T,a) \overset{②}{\Longleftrightarrow}$$
$$\exists k_1 < k_2 < t \ \cdot \ \left(\begin{array}{ll} & \omega^{k_1} \nvdash a \\ \wedge & \forall j \in]k_1 \ldots k_2] \cdot (\omega^j \vdash a) \\ \wedge & k_2 - k_1 \geq T \\ \wedge & \forall j \in]k_2 \ldots t] \cdot (\omega^j \nvdash a) \end{array} \right)$$

Le comportement de la porte $DUR_>$ est représenté par les figures VII.15 et VII.16.

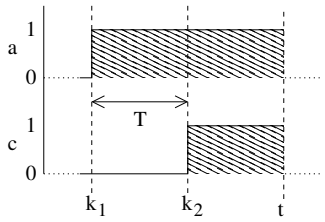


FIGURE VII.15 – Chronogramme de la porte $DUR_>$ - scénario ①

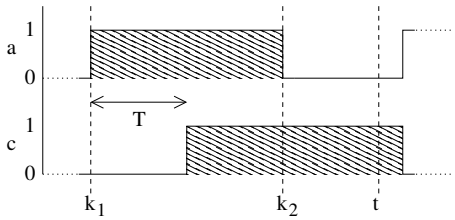


FIGURE VII.16 – Chronogramme de la porte $DUR_>$ - scénario ②

L'automate de Moore de la porte $DUR_>$ est représenté par la figure VII.17.

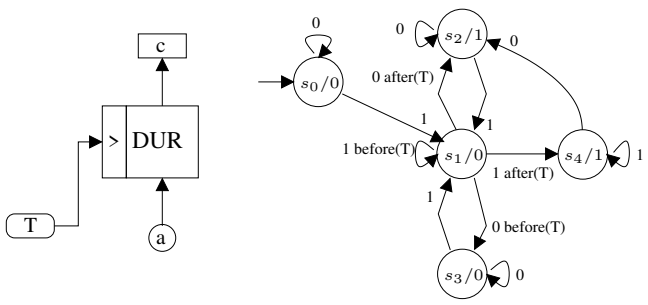


FIGURE VII.17 – Automate de Moore de la porte $DUR_{>}$

La sortie de la porte $DUR_{<}$ avec le seuil T pour une trace ω sur les événements a et b au temps t est définie comme suit :

$$\omega^t \vdash DUR_{<}(T,a) \iff$$

$$\exists k_1 < k_2 < t \ \cdot \ \left(\begin{array}{ll} & \omega^{k_1} \nvdash a \\ \wedge & \forall j \in]k_1 \ldots k_2] \cdot (\omega^j \vdash a) \\ \wedge & k_2 - k_1 \leq T \\ \wedge & \forall j \in]k_2 \ldots t] \cdot (\omega^j \nvdash a) \end{array} \right)$$

Le comportement de la porte $DUR_{<}$ est représenté par la figure VII.18.

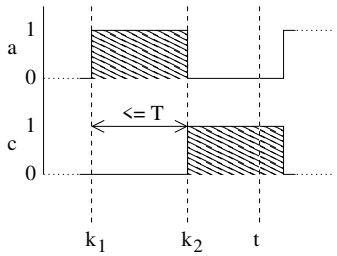


FIGURE VII.18 – Chronogramme de la porte $DUR_{<}$

L'automate de Moore associé à la porte $DUR_{<}$ est représenté par la figure VII.19.

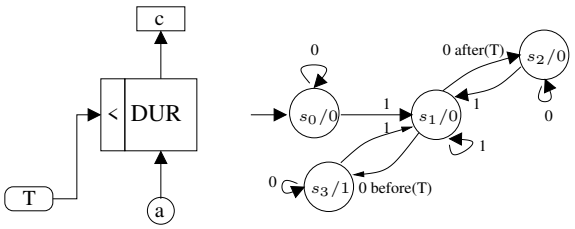


FIGURE VII.19 – Automate de Moore de la porte $DUR_{<}$

4.3 Présentation de la boite à outils des arbres de défaillances temporels et dynamiques dans Matlab/Simulink

Nous avons développé dans Matlab/Simulink une boite à outils (toolbox) proposant une bibliothèque de blocs pour la simulation d'arbres de défaillances dynamiques et temporels dans Matab/Simulink.

Pour les blocs des portes temporelles (PANDW, DUR, COUNT), nous avons défini des blocs paramétrables. L'utilisateur pourra choisir le critère de comparaison à utiliser pour la porte temporelle : $PANDW_<$ ou $PAND_>$, $DUR_<$ ou $DUR_>$, la valeur du seuil T pour les portes PANDW pu DUR ou du nombre d'occurrences N pour la porte COUNT. La figure VII.20 présente le paramétrage proposé pour la porte temporelle DUR. La figure VII.21 présente notre toolbox des arbres de défaillances dynamiques et temporels intégré dans Matlab/Simulink.

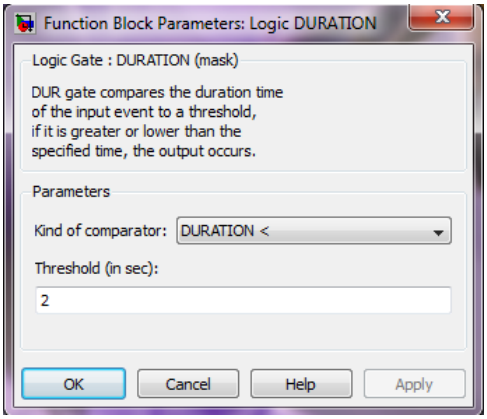
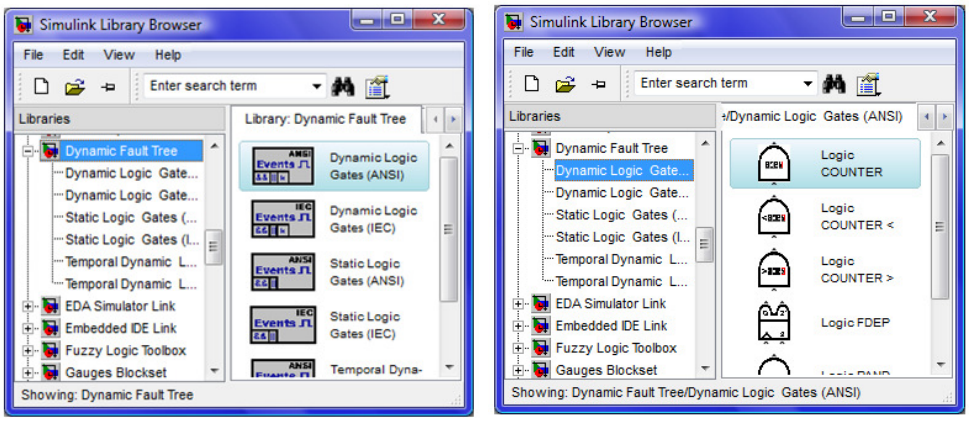


FIGURE VII.20 – Fenêtre de paramétrage de la porte DUR



(a) Librairie parent des DFT et TFT (b) Portes DFT et TFT proposées

FIGURE VII.21 – Boite à outils des arbres de défaillances dynamiques et temporels développée dans Matlab/Simulink

4.4 Exemple illustratif

Nous illustrons l'usage de notre toolbox des DFT et TFT dans Matlab/simulink par un exemple proposé dans [293]. Le système dont il faut filtrer les alarmes est constitué des composants A, B, C, D, E, F et il rencontre une défaillance si l'une de ces trois situations se produit :

— Les composants B et C défaillent et la défaillance de C se produit au moins 5 secondes après la défaillance de B et la défaillance du composant A n'est pas arrivée avant les défaillances de B et C.

— La défaillance du composant D apparait au moins de 10 secondes.

— Les défaillances des composants E et F apparaissent simultanément et la défaillance de E est apparue au moins 2 fois.

La défaillance du système est représentée par un arbre de défaillance dynamique et temporel, l'événement sommet est la sortie d'une porte OR dont les entrées correspondent aux trois situations décrites ci-dessus :

— La première situation de défaillance peut être décrite par une porte $PANDW_{>}$ qui connecte les événements B et C et par une porte SEQ qui connecte la sortie de la porte $PANDW_{>}$ à l'événement A.

— La deuxième situation de défaillance peut être modélisée par une porte $DUR_{>}$ sur l'événement D avec un seuil $T = 5$.

— La troisième situation de défaillance se traduit par une porte $COUNT_{>}$ sur l'événement E avec un nombre d'occurrences $N = 2$ et une porte AND qui connecte la sortie de la porte $COUNT_{>}$ à l'événement F.

Pour simuler l'arbre de défaillance dynamique et temporel, une trace de simulation doit être fournie pour les événements en entrées. La figure VII.22 présente une trace de simulation possible pour les événements A, B, C, D, E, F. Cette trace de simulation est générée par le bloc Signal Builder de Matlab/Simulink.

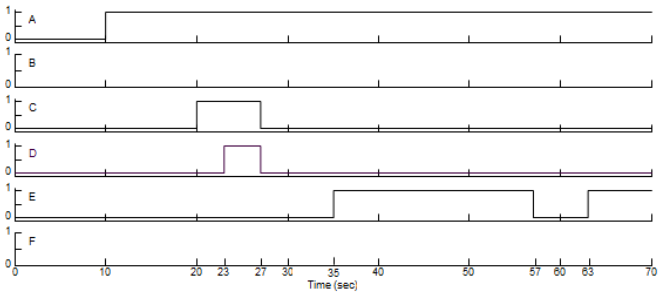


FIGURE VII.22 – Exemple de trace de simulation

Le modèle Simulink de la simulation de l'arbre de défaillance dynamique et temporel est représenté sur la figure VII.23.

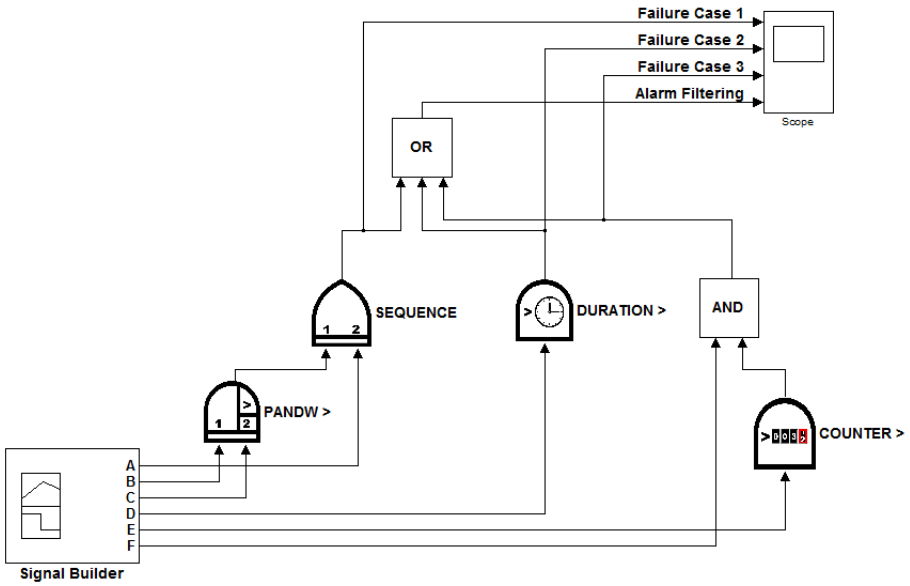


FIGURE VII.23 – Arbre de défaillance dynamique et temporel dans Matlab/Simulink

Le résultat de la simulation est présenté sur la figure VII.24. Le bloc Scope reliée à l'événement sommet de l'arbre et aux trois sous-arbres montre :

- La sortie de la porte $PANDW_{>}$ qui est la première entrée de la porte SEQ n'est pas générée. En conséquence, la sortie de la porte SEQ (première situation de défaillance) n'est jamais générée. Cela signifie que toutes les alarmes de la première situation de défaillance ont été filtrées.
- La sortie de la porte $DUR_{>}$ (deuxième situation de défaillance) est générée au temps de simlation $t = 27$ car la durée de la défaillance de l'événement D a durée au moins 4 unités de temps. L'alarme de la seconde situation de défaillance n'a pas été filtrée.
- La sortie de la porte AND (troisième situation de défaillance) n'est pas générée, car la défaillance sur l'événement F n'est jamais présente. Concernant la porte $COUNT_{>}$, sa sortie est générée, car la défaillance sur E est apparue au moins deux fois. Toutes les alarmes de la troisième situation de défaillance ont été filtrées.
- Comme résultat, la sortie de la porte OR (événement sommet *Alarm Filtering*) est générée lorsque la deuxième situation de défaillance est rencontrée, en conséquence le système est défaillant.

5 Conclusion et perspectives sur l'usage des arbres de défaillances dynamiques pour la sûreté de fonctionnement

Dans ce chapitre, nous avons étudié l'usage des arbres de défaillances dynamiques pour la sûreté de fonctionnement des systèmes complexes.

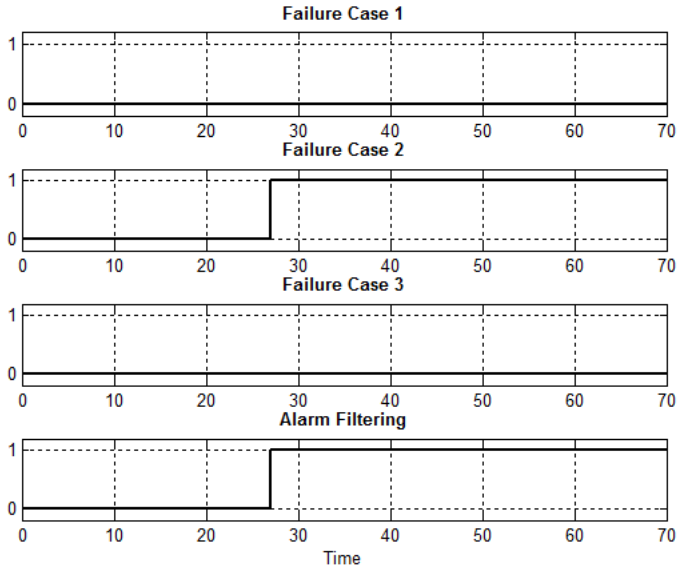


FIGURE VII.24 – Résultat de la simulation de l'arbre de défaillance dynamique et temporel

Le formalisme des arbres de défaillances dynamiques est bien adapté à une démarche déductive de la recherche des causes racines d'une défaillance. Nous avons proposé une méthode de recherche des causes racines d'une défaillance en utilisant un arbre de défaillances dynamique et les traces d'exécution des composants. Ce travail facilite ainsi le processus de maintenance corrective en limitant la recherche des composants défaillants. Une première perspective est d'étendre notre méthodologie aux arbres de défaillances temporels. Une seconde perspective est de prendre en compte un grand jeu de données sur le monitoring des composants (existant ou qu'on générerait à partir des lois de distributions des défaillances) et l'utiliser pour calculer des indicateurs pour chaque composant indiquant sa part de causalité dans les défaillances du système et son temps moyen de bon fonctionnement (MTTF). Cela fournirait aux équipes de maintenance des outils d'aide à la décision pour la maintenance préventive et corrective.

Notre travail sur l'analyse quantitative des DFT par une simulation guidée par les événements permet de prendre en compte tout arbre de défaillances dynamiques sans restriction sur les événements (événements répétés, événements partagés, événements multi-contraints) et sur les lois de distributions des défaillances et à montrer des résultats très satisfaisants en temps de calcul. Nous avons de nombreuses perspectives pour améliorer notre outil. Une première perspective est de prendre en compte les événements réparables en indiquant pour chaque événement son taux de réparation. Une seconde perspective est d'améliorer les performances et l'usabilité de notre solution en prenant en compte les événements rares (p. ex. un taux de défaillance inférieur à 10^{-9}) par des techniques d'échantillonnage préférentiel (*importance sampling*) et des méthodes multi-niveaux (*multilevel splitting*) [294]. Une troisième perspective est de prendre en compte des événements dont il est difficile de formuler leurs lois de défaillances (par manque de données, dues à la

complexité des comportements menant aux défaillances) en utilisant la logique flou pour représenter de tels systèmes incertains. Finalement, notre dernière perspective est d'incorporer dans notre solution des facteurs d'importances telles que le facteur d'importance de Birnbaum.

Nous avons également implémenté dans Matlab/Simulink une bibliothèque de blocs modélisant des portes dynamiques et temporelles pour simuler des arbres de défaillances dynamiques et temporels. Nous avons utilisé notre outil de simulation pour le filtrage de fausses alarmes dans le cadre de l'aéronautique. Cette méthodologie de filtrage de fausses alarmes est ainsi un outil d'aide à la décision pour les équipes de maintenances correctives.

Chapitre VIII

Usage des réseaux de files d'attente et des métaheuristiques pour la maintenance

1 Présentation générale : contexte, positionnement, originalité, état de l'art

1.1 Contexte, positionnement et originalité

La maintenance industrielle a pris au fil du temps de plus en plus d'importance dans les activités d'une entreprise de production. Elle est devenue un enjeu économique majeur par son rôle de plus en plus important dans la productivité de l'entreprise en assurant le bon fonctionnement des outils de production et en s'intégrant dans un contexte de développement durable par le recyclage et l'optimisation des ressources permettant ainsi d'économiser l'achat de nouveaux équipements de production. Les coûts de maintenance ont été estimés entre 15% et 70% du budget total de production [295]. De plus, environ 70% des entreprises considèrent la maintenance comme un coût plutôt que comme un investissement ou une source de profit [296].

Les méthodologies de maintenance industrielle peuvent être regroupées en deux grandes familles : la *maintenance corrective* et la *maintenance préventive* [297]. La maintenance corrective intervient lorsqu'un équipement d'un système industriel est défaillant. Cette défaillance provoque alors une indisponibilité de l'équipement qui diminue la productivité de l'entreprise. La maintenance corrective a pour objectif de rétablir la productivité de l'équipement par des interventions palliatives ou curatives. Une intervention palliative remet l'équipement en état de fonctionnement provisoirement par un dépannage et devra être suivie d'une action curative. Une intervention curative remet l'équipement en état de fonctionnement définitivement. Cela peut être obtenu par une réparation ou un remplacement. La maintenance préventive est la maintenance qui est effectuée avant la défaillance afin de maintenir l'équipement dans son état nominal en fournissant des inspections qui permettent la détection et la prévention de défaillances naissantes, elle vise à réduire la probabilité d'occurrence d'une défaillance. Selon la temporalité et les actions des interventions de maintenance préventive, on y distingue la maintenance systématique, la maintenance

conditionnelle et la maintenance prédictive. La maintenance systématique est établie selon un échéancier établi et prend en charge un ensemble d'opérations de maintenance (entretien, restauration, remplacement) effectuées systématiquement. La maintenance conditionnelle est réalisée suite à des contrôles sur l'état des équipements qui révèlent un état d'usure ou de dégradation nécessitant une opération de maintenance. La maintenance prédictive est décidée suite à l'analyse de l'évolution surveillée de l'usure, la dégradation d'un équipement. Cette analyse permet d'optimiser, en retardant ou en avançant, la planification d'une opération de maintenance.

Nous nous intéressons aux industries à haute technicité (industrie du transport aérien ou automobile, industrie de la défense, industrie de produits électriques et électroniques, etc.) ayant plusieurs sites de production. Dans ces domaines d'activités où la disponibilité des équipements de production est primordiale, les activités de maintenance sont divisées en deux structures : une structure qui réalise le processus de réparation : l'atelier de maintenance (AdM) central (*Central Maintenance Workshop*, CMW) et une structure qui effectue des inspections et des remplacements sur les différents sites de production : l'atelier de maintenance mobile (*Mobile Maintenance Workshop*, MMW). La figure VIII.1 représente le contexte de notre travail de recherche et met en évidence les relations entre l'AdM central et l'AdM mobile et les différents sites.

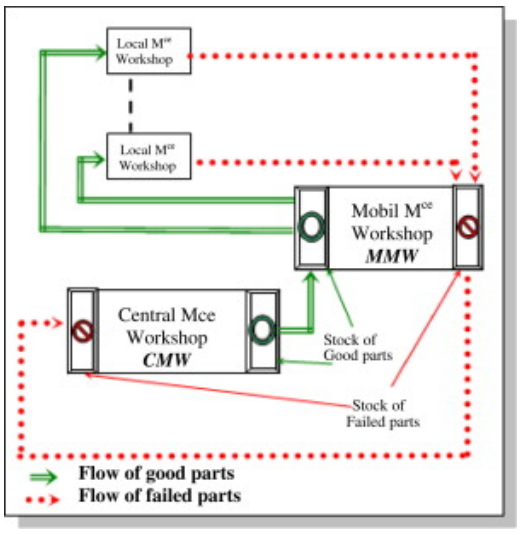


FIGURE VIII.1 – Relations entre les différents sites de production et les ateliers de maintenance [298]

Nos travaux de recherche proposent l'optimisation de cette maintenance multi-sites en combinant deux stratégies d'optimisation. La première vise à trouver le dimensionnement optimal (nombre de stations, valeur initiale des stocks d'équipements et de composants de rechange) de l'atelier de maintenance central, la seconde à trouver la planification optimale des trajets de l'atelier de maintenance mobile.

1.2 Etat de l'art

Le lecteur trouvera dans les ouvrages [299]-[301] des présentations didactiques et générales sur la maintenance industrielle. Les travaux de recherche sur la maintenance adressent différentes problématiques : la politique de maintenance, l'ordonnancement et le choix des opérations de maintenance, l'organisation de l'atelier de maintenance, le flux et la gestion des pièces de rechange.

L'état de l'art présenté dans cette section porte principalement sur l'étude de la maintenance lorsqu'il n'y a qu'un seul atelier de maintenance (l'atelier de maintenance central du cadre de nos travaux de recherche sur la maintenance multi-sites), nous ne considérons pas les situations où il serait préférable d'externaliser la maintenance sur chaque site de production.

Afin de choisir la maintenance la plus appropriée, des travaux de recherche ont proposé des évaluations comparatives de différentes politiques de maintenance [297], [302]-[306]. La problématique du dimensionnement d'un atelier de maintenance a fait l'objet de nombreuses publications. Cette problématique vise à trouver un compromis entre la quantité des stocks d'équipements et de composants de rechange (*spare part inventory*) et la capacité de réparation (*repair capacity*) [307]. La capacité de réparation décrit le nombre de postes et ses services associés (diagnostic, désassemblage, réparation, réassemblage, test). L'augmentation de la disponibilité des équipements de production peut être obtenue d'une part par l'augmentation des stocks des équipements de rechange pour ne pas attendre qu'un équipement défaillant passe par tout le cycle de réparation, d'autre part par l'augmentation du nombre de postes dans l'atelier de maintenance afin de diminuer son temps de séjour dans l'atelier. Différents travaux de recherche ont porté sur l'optimisation de la capacité de réparation [308], [309]. La capacité de réparation d'un atelier de maintenance peut être également augmentée par une optimisation de l'ordonnancement des travaux sur les postes de l'atelier [310]. Des travaux de recherche étudient également l'optimisation de la gestion des stocks des pièces de rechange selon la politique de maintenance utilisée : dimensionnement du stock initial, capacités et organisation des lieux de stockage (multi-echelon warehouse management) [311]-[315].

Les analyses de performance des ateliers de maintenance peuvent être conduites par différentes approches : de manière analytique ou par simulation. Les outils de modélisation principaux pour modéliser un atelier de maintenance sont les réseaux de files d'attente et les réseaux de Petri stochastiques. Par exemple dans [298], une démarche de modélisation par réseaux de Petri stochastiques généralisés à synchronisations internes (RdPSGSyI) d'un système de production multi-sites avec un atelier de maintenance central et un atelier de maintenance mobile est proposée. La simulation par synchronisation des sites de production, de l'ADM central et de l'AdM mobile permet d'analyser et d'évaluer les performances de maintenance.

Pour optimiser la maintenance, différents travaux utilisent les métaheuristiques, en particulier pour la maintenance préventive [316]-[320]. Dans [316], Lapa et al. ont étudié l'utilisation des algorithmes génétiques pour l'optimisation de la maintenance préventive

des réacteurs à eau pressurisée dans les centrales nucléaires. Dans [317], les auteurs utilisent la métaheuristique des algorithmes génétiques pour optimiser les intervalles d'interventions entre deux actions de maintenance planifiées pour chaque composant du système, et ce afin de minimiser le coût de fonctionnement et de maintenance du système. Dans [318], [319], les auteurs utilisent les métaheuristiques des algorithmes génétiques et des colonies de fourmis pour la détermination des conditions de remplacement des composants (dates et composants concernés) dans les systèmes série-parallèle minimisant le coût de la politique de maintenance. Dans [320] la métaheuristique du recuit simulé est utilisée pour optimiser l'opération de maintenance à effectuer (inspection, réparation, remplacement) lors des visites planifiées de maintenance préventive.

La problématique de l'optimisation de la capacité de stockage et de la tournée de l'atelier de maintenance mobile est un cas particulier des problèmes du voyageur de commerce (*traveling salesman problem*), tournées de véhicules (*vehicle routing problem*) et de capacité du véhicule de transport (*capacitated vehicle routing problem*). Les méthodes d'optimisation les plus efficaces sont l'algorithme génétique, le recuit simulé, la colonie de fourmis. En prenant en compte l'insertion – le retour à l'atelier de maintenance central dans la tournée de l'atelier de maintenance mobile pour prendre en compte sa capacité de stockage de pièces réparées et défaillantes, cette problématique peut être reliée à des travaux de recherche relatifs à la tournée de véhicules avec contraintes de fenêtre temporelle [321], de présence de dépôts intermédiaires [322] et de ramassages et livraisons [323].

2 Synthèse des travaux développés

2.1 La maintenance centralisée pour les systèmes de production multi-sites

Nos travaux de recherche se sont portés sur la maintenance centralisée pour des systèmes de production multi-sites. Plus spécifiquement, nos activités de recherchent visent à l'amélioration de la disponibilité des équipements de production et la réduction des coûts de maintenance par des outils d'aide à la décision sur :

- la politique de maintenance à choisir ;
- le dimensionnement de l'AdM central ;
- le choix de la position de l'AdM central relatif aux différents sites de production ;
- le dimensionnement de l'AdM mobile et son parcours entre l'AdM central et les différents sites de production.

Les travaux que nous avons menés sont dans la continuité des thèses de Rosa Abbou [324] et Ahmad Alhouaij Alali [325]. Nos travaux de recherche ont porté d'une part sur le dimensionnement de l'atelier de maintenance central lorsqu'on utilise une politique de maintenance corrective de type réparation par remplacement [326], [327], et d'autre part sur l'optimisation de la planification de la tournée de l'atelier de maintenance mobile [328].

2.1.1 Organisation de l'atelier de maintenance central

L'organisation de l'atelier de maintenance central est caractérisée par sa capacité de réparation (nombre de postes de diagnostic, de démontage, de réparation, de réglage, d'assemblage et de test) et son stock de pièces de rechange. La figure VIII.2 présente le cycle de réparation d'un équipement défectueux dans un atelier de maintenance sur différents postes d'intervention selon le type de défaillance et la probabilité d'occurrence de cette défaillance. Le processus de réparation est le suivant : après une phase de diagnostic et démontage où la cause de la défaillance est identifiée, l'équipement ou le composant cause de la défaillance est transféré vers l'un des postes de réparation selon une certaine probabilité liée à l'occurrence de la défaillance, puis s'effectue les opérations de remontage, de réglage et de test.

Dans nos travaux de recherche, nous nous sommes intéressés à la maintenance corrective des équipements des sites de production distribués.

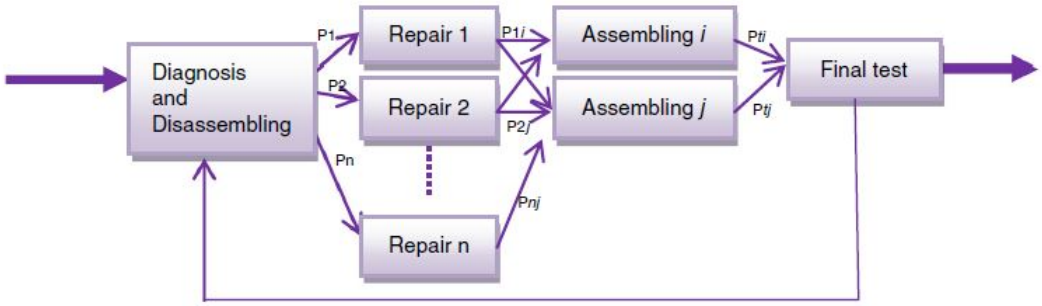


FIGURE VIII.2 – Cycle de réparation d'un équipement défectueux dans un atelier de maintenance [327]

2.1.2 Processus de maintenance corrective dans un atelier de maintenance centralisé,

les différentes étapes du processus de réparation des équipements défectueux avec un atelier de maintenance centralisé sont détaillées dans l'organigramme de la figure VIII.3.

Suite à une panne constatée d'un équipement de production, l'équipement défectueux est transféré à l'atelier de maintenance central et placé dans un stock d'entrées des équipements à réparer. Après une phase de diagnostic, l'équipement ou l'un de ses composants est acheminé vers l'un des postes de réparation dédiés au type de défaillance que rencontre l'équipement. Après la réparation, une étape de test permet de requalifier cet équipement pour être à nouveau utilisé dans le système de production et il est placé dans un stock de sorties d'équipements prêt à être réinstallé. Si la phase de requalification échoue, un nouvel équipement ou un nouveau composant est commandé. Lorsque l'équipement est réparé, il est placé dans le stock de sorties. Il sera pris en charge par l'atelier de maintenance mobile pour sa réinstallation sur le site de production. Si l'équipement ne peut pas être déplacé,

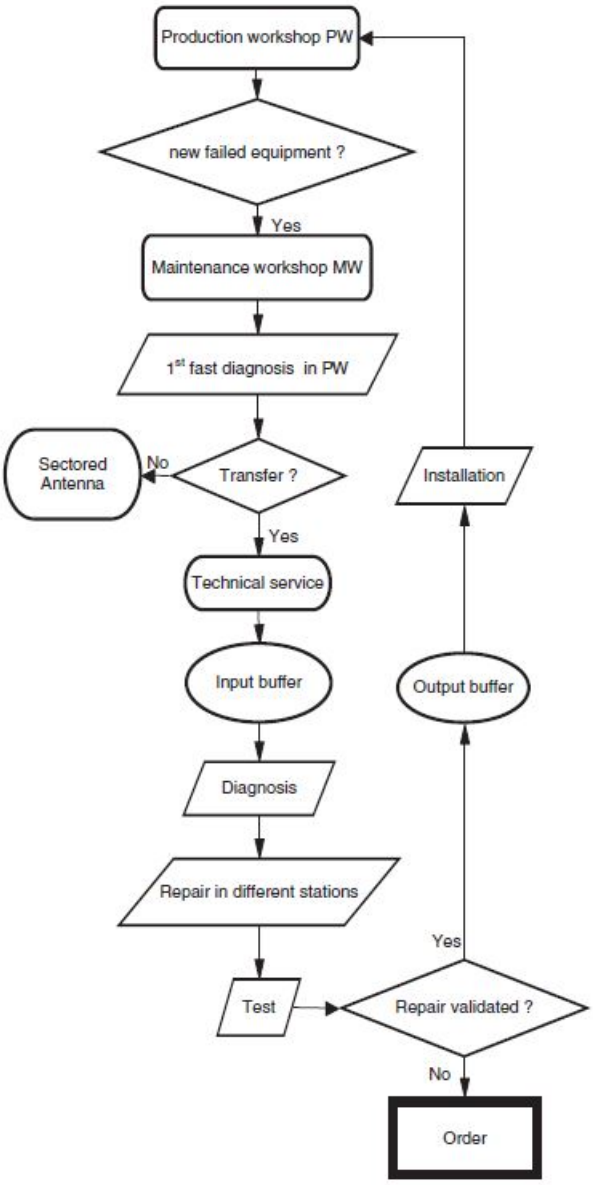


FIGURE VIII.3 – Processus de maintenance corrective avec un atelier de maintenance central [327]

l'atelier de maintenance mobile se déplace pour effectuer la maintenance corrective sur le site de production.

Pour améliorer la disponibilité des équipements de production, la maintenance corrective de type *réparation par remplacement* peut être utilisée. Cette technique de réparation signifie qu'un équipement ou un composant défaillant est immédiatement remplacé par un équipement ou un composant identique ou équivalent, prêt à l'emploi (réparé ou neuf).

En parallèle du remplacement, le processus de réparation est démarré. Cette technique de réparation nécessite que le stock de l'atelier de maintenance centralisé soit approvisionné en équipements et de composants de rechange de toutes sortes. Les différentes étapes du processus de réparation par remplacement dans un atelier de maintenance centralisé sont détaillées dans l'organigramme de la figure VIII.4.

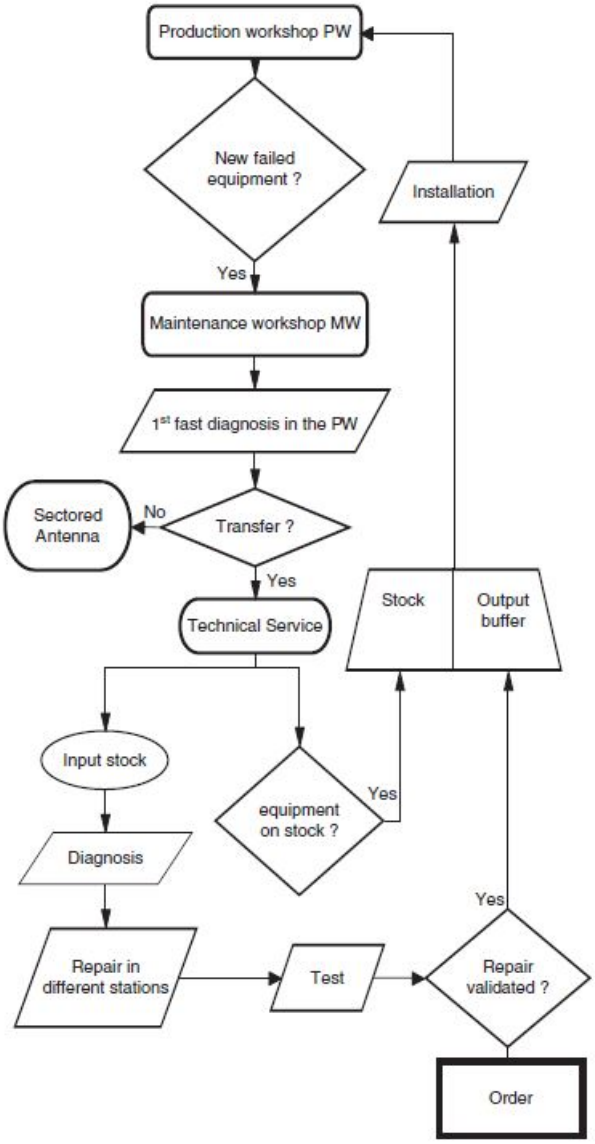


FIGURE VIII.4 – Processus de maintenance corrective par réparation par remplacement [327]

2.2 Usage des réseaux de files d'attente pour le dimensionnement d'un atelier de maintenance

La problématique de la conception de l'atelier de maintenance central, à laquelle nous nous sommes intéressés, est de déterminer le dimensionnement optimal en nombre de postes d'intervention et en équipements à stocker en fonction des caractéristiques des équipements de productions (taux de défaillance) pour minimiser les coûts de gestion des stocks et des temps de séjour des équipements défaillants dans l'atelier de maintenance centralisé.

2.2.1 Présentation des réseaux de files d'attente

La théorie des files d'attente est une théorie mathématique relevant de la théorie des probabilités et de la modélisation stochastique. Son origine provient des travaux du mathématicien A. K. Erlang [329], [330] sur les réseaux téléphoniques au début du XXième siècle à l'aide de processus de Poisson. Plus tard, ces résultats furent étendus en considérant d'autres lois de distribution et en énonçant de nouveaux résultats analytiques.

La théorie des files d'attente permet de modéliser des systèmes où un flux d'événements qu'on appelle *clients* arrive séquentiellement en réclamant des *services*. Le dispositif délivrant le service est appelé *serveur*. Lorsqu'un client utilise le serveur, celui-ci est indisponible pendant une durée de service. Quand un client arrivant ne peut pas accéder au serveur s'il est occupé, celui-ci est placé dans une *file d'attente*. L'ensemble du serveur et de sa file d'attente associée est appelé *station*. La figure VIII.5 représente une station. Une station se caractérise par :

— Les temps d'inter-arrivées des clients décrits à l'aide d'un processus stochastique.

— La discipline de service du serveur qui décrit dans quel ordre sont traités les clients. Les disciplines les plus connues sont en respectant l'ordre d'arrivée des clients (discipline FIFO) ou en servant en premier le dernier client arrivé (discipline LIFO).

— La capacité de la file d'attente : à capacité limitée ou illimitée. Dans le cas d'une capacité limitée, il y aura refoulement (perte) de clients à l'entrée du système si la file d'attente est pleine.

— Les temps de service du serveur décrit à l'aide d'un processus stochastique. La loi de distribution utilisée permet d'exprimer le temps passé par un client dans le serveur.

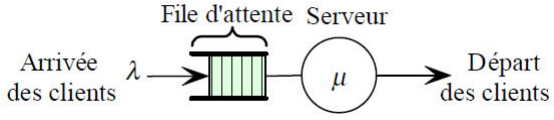


FIGURE VIII.5 – Représentation d'une station (file d'attente avec un serveur unique) [324]

2.2.2 Modélisation et évaluation des performances d'un atelier de maintenance central par simulation

2.2.2.1 Modélisation par réseaux de files d'attente

Nous évaluons les performances d'un atelier de maintenance à l'aide de réseaux de files d'attente. Sur la figure VIII.6, nous présentons un atelier de maintenance. Son organisation est constituée de plusieurs étapes : diagnostic, réparation, réglages, assemblage, etc. Chaque étape est réalisée par un ou plusieurs postes d'intervention. En modélisant chaque poste d'intervention par une station, nous représentons l'atelier de maintenance comme un réseau de files d'attente, voir figure VIII.7.

Pour chaque station, des clients arrivent séquentiellement et attendent dans la file d'attente si nécessaire, puis reçoivent le service fourni par le serveur et quittent la station. La station est alors caractérisée par son taux d'arrivée (λ) et le temps moyen de service du serveur (μ). Ces paramètres correspondent au taux d'arrivée des équipements défaillants à chaque poste d'intervention et au temps de service moyen dans celui-ci.

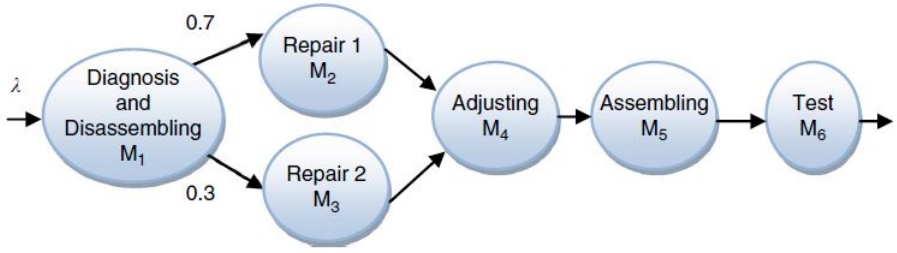


FIGURE VIII.6 – Exemple d'organisation d'un atelier de maintenance [327]

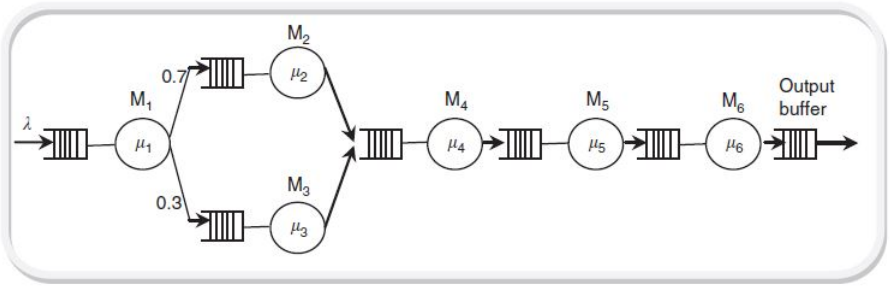


FIGURE VIII.7 – Réseau de files d'attente pour un atelier de maintenance [327]

2.2.2.2 Evaluation des performances d'un réseau de files d'attente

Des mesures de performance peuvent être associées à chaque station ainsi que pour le réseau : taux d'utilisation des serveurs, taille moyenne des files d'attente, temps moyen d'attente d'un client, temps moyen de séjour des stations, etc. Dans nos travaux de recherche, nous nous sommes concentrés uniquement sur le temps de séjour qui caractérise

le temps pendant lequel l'équipement à réparer n'est pas disponible. Cela nous aide à déterminer le nombre de stations dans l'atelier de maintenance central, à définir le niveau de stock de pièces de rechange et les capacités de stockage (capacité des files d'attente). Notre objectif est de minimiser le temps de séjour des équipements défaillants afin d'augmenter la disponibilité des équipements de production. Pour une classe particulière de réseaux de files d'attente connue sous le nom de réseaux de Jackson [331], on peut évaluer les performances exactes du système par des méthodes analytiques. Les réseaux de Jackson correspondent à des réseaux de files d'attente où la distribution des temps de service est exponentielle, les arrivées extérieures sont des processus de Poisson, les files d'attente ont des capacités illimitées de sorte qu'il n'y ait pas de blocage et la discipline de service dans toutes les files d'attente est le premier arrivé, premier servi. Tous les réseaux de files d'attente ne sont pas des réseaux de Jackson lorsqu'on introduit de nouvelles lois de distribution ou de contraintes sur les capacités des files d'attente. Pour évaluer les performances de ces réseaux, des méthodes mathématiques approximatives doivent être développées [332], [333] ou faire appel à la simulation stochastique [334], [335]. Dans nos travaux de recherche, nous avons utilisé une approche à base de simulation pour analyser et dimensionner les réseaux de files d'attente.

2.2.2.3 Simulation de réseaux de files d'attente sous Matlab/Simulink

Pour évaluer les performances d'un atelier de maintenance centralisé, nous avons développé une toolbox sous Matlab/Simunlink [326], [327] qui permet à l'utilisateur de créer un réseau de files d'attente puis de le simuler. La figure VIII.8 présente cette toolbox. Cette toolbox propose quatre types de blocs.

Le bloc *Inter-arrival generator* (figure VIII.8(a)) modélise l'arrivée d'équipements défaillants à l'entrée de l'atelier de maintenance centralisée (flèche étiquetée λ sur la figure VIII.7). La loi de distribution des arrivées et son espérance λ sont paramétrables dans le bloc. Le bloc génère l'arrivée d'équipements par le calcul des temps d'inter-arrivées.

Le bloc *Single server queue* modélise une station. Ce bloc est paramétrable par :

— la capacité de la file d'attente ;

— le temps de service moyen du serveur μ ;

— le nombre de sources d'arrivées sur la file d'attente.

Si le nombre de sources d'arrivée de la station est égal à 1 (figure VIII.8(b)), ce bloc permet de représenter les stations M_1, M_2, M_3, M_5 et M_6 avec leurs files d'attente et leur temps de services respectifs.

Si le nombre d'arrivée est supérieur à 1 (figure VIII.8(c)), ce bloc permet de représenter plusieurs sources d'arrivées (*traffic merging*) sur la file d'attente. Cette situation est illustrée par la station M_4 sur la figure VIII.7 où il existe deux sources d'arrivées d'équipements sur la station M_4, l'une provenant de la sortie de la station M_2, l'autre de la station M_3.

Le bloc *Distribution* (figure VIII.8(d)) distribue la sortie d'une station vers différentes destinations. Ce bloc est paramétrable avec un vecteur de probabilités. Si la sortie d'une station est distribuée sur n destinations différentes avec les probabilités respectives $P_1, P_2, \ldots, P_n$ telle que $P_1 + P_2 + \ldots + P_n = 1$, le bloc Distribution utilisera comme paramètre le vecteur $[P_1 P_2 \ldots P_n]$. Cette situation est illustrée sur la figure VIII.7 par la distribution de la sortie de la station M_1 vers les deux stations M_2 et M_3 avec les probabilités respectives 0.7 et 0.3.

Le bloc *Synchronization* (figure VIII.8(e)) permet de modéliser la synchronisation entre la sortie d'une station et une demande d'équipements (figure VIII.9). Le bloc est paramétrable par la taille des files d'attente et la demande d'équipements initiale. Une sortie ne sera générée par le bloc Synchronization qu'en présence de l'arrivée d'un équipement issue d'une station et de la d'une demande d'équipement.

2.2.3 Méthodologies d'optimisation de l'atelier de maintenance central

Dans nos articles [326], [327] nous proposons différentes méthodologies d'optimisation d'un atelier de maintenance centralisée. Notre critère est de minimiser le temps de séjour de l'équipement défaillant dans l'atelier. Nous avons étudié deux cadres d'études pour lesquelles nous proposons des méthodologies d'optimisation.

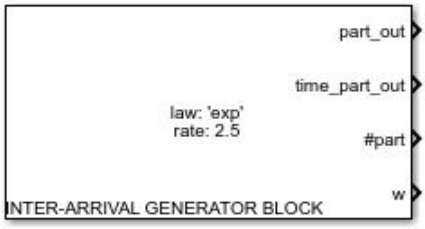
2.2.3.1 Optimisation du temps de séjour par duplication de stations

Dans la première étude, nous nous plaçons dans la situation d'une maintenance corrective avec réparation dans un atelier de maintenance centralisé (organigramme figure VIII.3) : un équipement dans un site de production local est défaillant, celui-ci est alors déplacé à l'atelier de maintenance centralisé. Lorsque l'équipement défaillant sera réparé, il sera réinstallé dans son site de production par l'atelier de maintenance. Pendant toute la durée de sa réparation, l'équipement n'est donc pas disponible pour le site de production. Pour trouver le dimensionnement de l'atelier de maintenance procurant les temps de séjours minimaux, nous analysons différentes configurations de l'atelier de maintenance en dupliquant des stations.

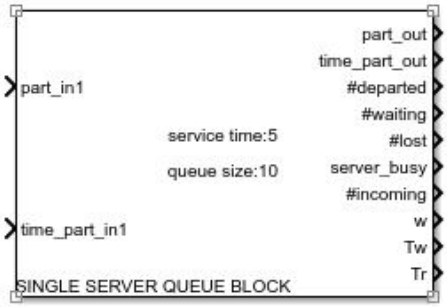
La figure VIII.10 présente les résultats de la simulation dans Matlab/Simulink de l'atelier de maintenance centralisée (figure VIII.7) pour des valeurs fixées du taux d'arrivée λ des équipements défaillants et des temps de séjours moyens μ des stations.

Les résultats de la simulation montent que les temps de séjour moyens les plus importants sont dans les stations M_5 et M_2, suivi par la station M_3. Une solution possible pour diminuer le temps moyen de réparation d'un équipement défaillant dans l'atelier de maintenance, ici égal à 11.94, est de dupliquer les stations avec les temps de séjours les plus longs. Nous avons étudié plusieurs configurations alternatives de l'atelier de maintenance centralisé : la configuration où la station M2 est dupliquée, la configuration où la station M3 est dupliquée et la configuration où la station M5 est dupliquée.

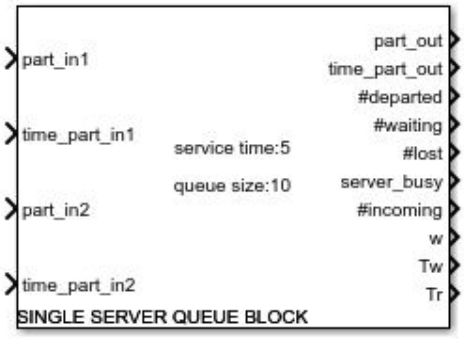
Les simulations de ces configurations montent que c'est l'atelier de maintenance avec



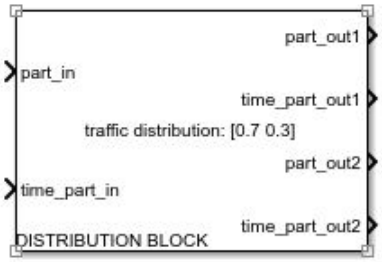
(a) Inter-arrival generator block



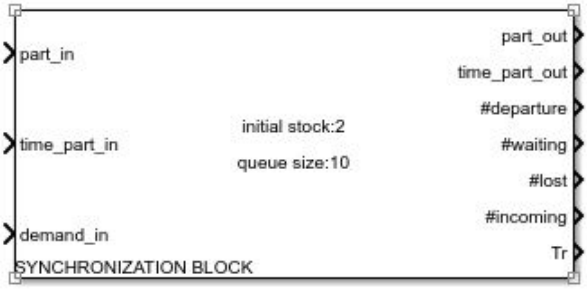
(b) Single server queue block without traffic merging



(c) Single server queue block with traffic merging



(d) Distribution block



(e) Synchronization block

FIGURE VIII.8 – Notre toolbox de simulation de réseaux de file d'attente [326], [327]

la station M_5 dupliquée qui offre le temps de réparation moyen minimal (figures VIII.11 et VIII.12). Le temps de séjour moyen d'un équipement dans l'AdM central est réduit de 11.94 à 10.28.

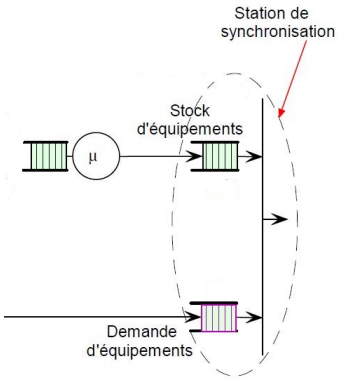


FIGURE VIII.9 – Synchronisation entre la sortie d'une station est une demande d'équipement

10,017 parts ($<\lambda>=0.2005$)		M_1	M_2	M_3	M_4	M_5	M_6
Service rate	μ	0.65	0.40	0.48	2.00	0.40	1.80
Average sojourn time at station i (hour)	wi	2.22	*4.05*	2.34	0.54	*4.99*	0.62
Average sojourn time at workshop (hour)	W_T			*11.94*			

FIGURE VIII.10 – Simulation des temps de séjours moyens des stations et de l'atelier de maintenance [327]

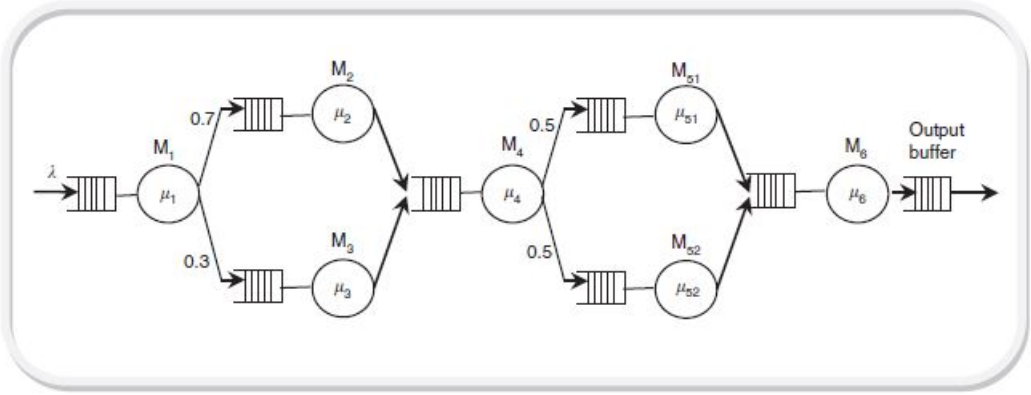


FIGURE VIII.11 – Atelier de maintenance central avec station M_5 dupliquée [327]

2.2.3.2 Optimisation du temps de séjour par technique de réparation par remplacement

Dans la deuxième étude, nous nous plaçons dans la situation d'une maintenance corrective de type *réparation par remplacement* (organigramme figure VIII.4). Nous avons étudié deux niveaux de réparation par remplacement :

10,043 parts ($<\lambda> = 0.2010$)		M_1	M_2	M_3	M_4	M_{51}	M_{52}	M_6
Service rate	μ_i	0.65	0.40	0.48	2.00	0.40	0.40	1.80
Average sojourn time at station i (hour)	w_i	2.29	3.88	2.35	0.56	3.36	3.37	0.61
Average sojourn time at workshop (hour)	W_T				10.28			

FIGURE VIII.12 – Simulation des temps de séjours moyens quand la station M_5 est dupliquée [327]

— Niveau 1 : le remplacement consiste à remplacer l'équipement défaillant par un équipement de rechange provenant des stocks de l'AdM central. Une synchronisation entre la demande d'un équipement de rechange et le stock d'équipements de rechange sera nécessaire.

— Niveau 2 : Si un équipement de remplacement n'est pas disponible, le remplacement consiste à remplacer le composant provoquant la défaillance de l'équipement par un composant de rechange provenant des stocks de l'AdM central. Dans cette situation, la synchronisation entre la demande d'un composant de rechange avec le stock de composants de rechange sera également nécessaire.

Pour trouver le dimensionnement de l'atelier de maintenance procurant les temps de séjours minimaux, nous analysons différentes configurations de l'atelier de maintenance en fonction de l'état des stocks initiaux des équipements ou des composants de rechange.

2.2.3.2.1 Niveau 1 de remplacement

Le principe de cette technique est le suivant : dès qu'un équipement défaillant est présent dans l'AdM central, sa réparation commence. En même temps, si un équipement opérationnel est disponible en stock, il est utilisé pour son remplacement dans le site de production. Ainsi, le temps d'indisponibilité est réduit au temps de transfert du nouvel équipement et à son l'installation dans le site de production. Le modèle de réseau de file d'attente correspondant à cet atelier avec remplacement de niveau 1 est illustré à la figure VIII.13.

Lorsqu'un équipement défaillant arrive à l'AdM central, la demande de remplacement D est mise à jour. Si un équipement est disponible dans le stock S, il est retiré du stock (synchronisation entre l'état du stock S et l'état des demandes D). Dans l'atelier de maintenance, l'équipement défaillant transite entre les différentes stations et lorsqu'il est réparé, il devient un équipement disponible dans le stock S. La figure VIII.14 présente les différentes performances de l'AdM central avec le niveau 1 de remplacement selon le dimensionnement du stock initial d'équipement disponible S. S'il n'y a pas de stock initial (capacité du stock 0), le temps de séjour d'un équipement dans l'AdM central est inchangé (11.94). Les simulations de l'AdM central avec le niveau 1 de remplacement montrent que le niveau du stock d'équipement de rechange fait diminuer la durée du temps de séjour moyen de l'équipement dans l'AdM central.

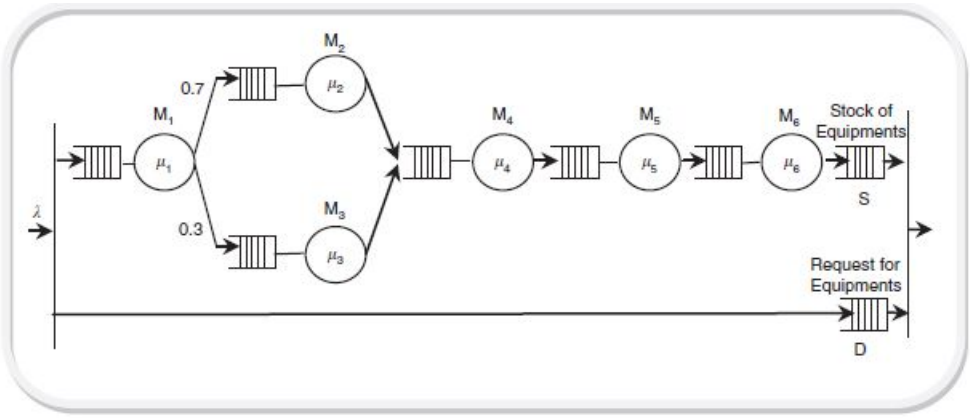


FIGURE VIII.13 – Configuration de l'AdM central avec le niveau 1 de remplacement [327]

Stock size	0	1	2	3	4
Average sojourn time at workshop (hours)	11.94	7.25	3.44	1.82	0.93

FIGURE VIII.14 – Simulation des temps de séjours moyens avec le niveau 1 de remplacement [327]

2.2.3.2.2 Niveau 2 de remplacement

Dans le cas de la politique de remplacement de niveau 2, nous considérons que la défaillance des équipements est principalement due à un seul composant démontable et réparable. Ainsi, lorsqu'un équipement défaillant est transféré à l'AdM central trois cas se présentent :

1. Si un équipement de rechange est disponible en stock, il est alors remplacé par celui dans le site de production et l'équipement défaillant suit toutes les phases de réparation dans l'AdM central pour compléter le stock d'équipement de rechange.
2. S'il n'y a pas un équipement de rechange dans le stock, mais qu'un composant de rechange est disponible en stock, le composant est remplacé. Ainsi, il y a un passage direct aux stations d'assemblage et de test (stations M_5 et M_6). Le composant défaillant suit les étapes de réparation dans l'AdM pour compléter le stock de composants de rechange.
3. S'il n'y a ni équipement de rechange ni composant de rechange dans les stocks, la demande de remplacement attend pour que toutes les phases du processus de réparation soient complétées.

Le modèle de réseau de files d'attente correspondant à cette politique de remplacement est donné sur la figure VIII.15.

La figure VIII.16 présente les différentes performances de l'AdM central avec le niveau 2 de remplacement selon le dimensionnement du stock initial d'équipement de rechange S_2

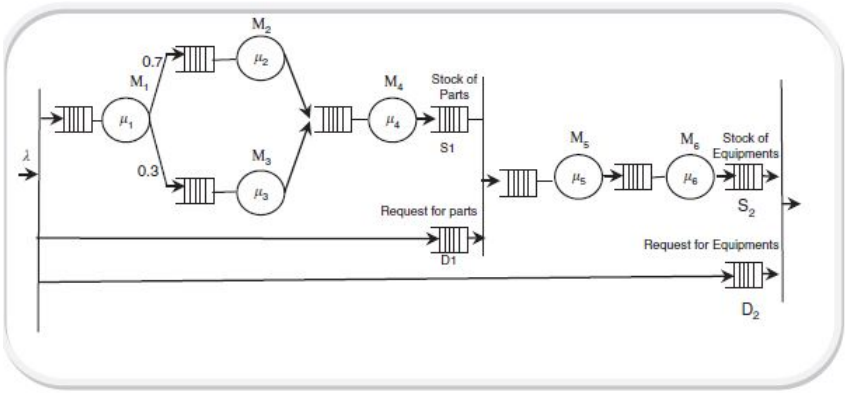


FIGURE VIII.15 – Configuration de l'AdM central avec le niveau 2 de remplacement [327]

et le dimensionnement du stock initial de composants de rechange S_1.

Stocks size of S_1	0				1				2				3			
Stocks size of S_2	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
Sojourn time (hours/week)	10.1	7.01	5.37	4.59	6.14	3.80	2.62	2.07	3.44	1.97	1.25	0.9	1.82	0.99	0.59	0.41

FIGURE VIII.16 – Simulation des temps de séjours moyens avec le niveau 2 de remplacement [327]

Nos résultats de simulations montrent bien évidemment que les équipements de rechange réduisent de manière plus importante les temps de séjours que les composants de rechange et que l'augmentation des stocks initiaux permet aussi de diminuer de manière plus importante les temps de séjour dans l'atelier de maintenance.

Cependant, pour choisir le dimensionnement de l'atelier de maintenance, il faut prendre en compte également les coûts d'investissements, les pertes et les gains. Pour dimensionner d'une manière optimale les stocks initiaux des équipements et des composants de rechange, il faut alors trouver un compromis entre la minimisation du temps de séjour, l'investissement dans le stockage des équipements et des composants ainsi que l'augmentation du bénéfice.

2.3 Usage des métaheuristiques pour l'optimisation des tournées de l'atelier de maintenance mobile

Dans le cadre de notre étude sur la maintenance multi-sites avec un atelier de maintenance central et un atelier de maintenance mobile, nous sommes également intéressés aux trajets de l'atelier de maintenance mobile, celui-ci joue en effet un rôle prépondérant dans le calcul du coût de maintenance d'où la nécessité d'optimiser son parcours.

La difficulté de trouver le trajet optimal réside dans la nature combinatoire du problème. En effet pour trouver la solution optimale il faudrait explorer l'ensemble des solutions puis choisir celle qui génère le plus faible coût. Cette méthode de calcul « méthode exacte » demande beaucoup de temps si nous avons un nombre de sites de production relativement important. Pour répondre à cette contrainte de temps de calcul inhérente à ce type de problème à forte combinatoire, des méthodes approchées (*métaheuristiques*) ont été développées. Les métaheuristiques s'inspirent généralement de phénomènes physiques, biologiques, etc. Nous avons utilisé deux métaheuristiques dans notre travail à savoir l'*algorithme génétique* et le *recuit simulé*. Ce choix est justifié par le fait que ces algorithmes sont efficaces sur les problèmes de tournées.

2.3.1 Présentation du problème d'optimisation des tournées de l'atelier de maintenance mobile

Le problème que nous considérons est de construire le trajet de l'atelier de maintenance mobile pour délivrer des équipements de rechange passant par tous les sites de production ayant le coût le plus faible. Le trajet de l'atelier de maintenance mobile doit prendre en compte un ou plusieurs retours à l'atelier de maintenance central pour le réapprovisionner en équipements de rechange. Cette contrainte de retour à l'AdM central pendant la tournée est liée à la capacité de stockage de l'atelier de maintenance mobile et aux demandes des sites de production.

Nous avons développé des algorithmes utilisant des métaheuristiques et prenant en compte ces points de retour à l'atelier de maintenance central pour trouver la tournée optimale de l'atelier de maintenance mobile . Notre méthodologie est décomposée en 3 étapes : 1) L'acquisistion et la modélisation des données, 2) l'expression de la fonction coût de maintenance mobile, 3) Application de l'algorithme d'optimisation du trajet de l'AdM mobile.

2.3.1.1 Principales hypothèses

L'étude de notre problème prend en compte les hypothèses suivantes :

(H1) Hypothèses sur les sites de production : tous les sites de production sont identiques, ils utilisent des équipements de caractéristiques identiques, les équipements de rechange transportés par l'AdM mobile sont donc similaires et ont tous la même durée d'installation sur un site de production.

(H2) Hypothèses sur l'atelier de maintenance central : dans l'AdM central, il y a toujours un stock suffisant d'équipements de rechange prêts à être chargées. La durée de chargement des équipements de rechange est considérée comme négligeable.

(H3) Hypothèses sur l'atelier de maintenance mobile : Un remplacement standard d'un seul type d'équipement est effectué au niveau de chaque SdP. Le temps de séjour dans un SdP est assimilé à la durée de remplacement d'un équipement.

2.3.1.2 Modélisation des données

Dans un contexte de production multi-sites avec une maintenance centralisée, le coût d'une tournée de l'atelier de maintenance mobile va dépendre :

— des distances séparant les sites de production entre eux et à l'atelier de maintenance central ;

— des temps de parcours des sites de production entre eux et avec l'AdM central ;

— de la durée d'intervention pour installer un équipement ;

— des demandes en équipements de rechange de chaque site de production.

La première étape de notre méthodologie d'optimisation du parcours de l'atelier de maintenance mobile est l'acquisition puis la modélisation des données.

2.3.1.2.1 Données géographiques

Les données géographiques concernent la situation géographique des différents sites de production (SdP) ainsi que la position de l'atelier de maintenance central (AdM central). On modélise les données géographiques par :

— n : le nombre de sites de production ;

— $d_{c,i}$: la distance entre l'AdM central et le SdP_i ;

— $d_{i,j}$: la distance entre le SdP_i et le SdP_j.

Ces données sont mémorisées dans la matrice des distances M_D de $n+1$ lignes et $n+1$ colonnes. Les index de numéros de lignes et de colonnes de la matrice M_D démarrent à partir de la valeur 0. L'index 0 correspond à l'AdM central, les index $1, \ldots, n$ correspondent aux SdP. On considère que la distance entre le SdP_i et le SdP_j est identique à la distance entre le SdP_j et le SdP_i ($d_{i,j} = d_{j,i}$).

$$M_D = \begin{bmatrix} 0 & d_{c,1} & d_{c,2} & \ldots & d_{c,n} \\ d_{c,1} & 0 & d_{1,2} & \ldots & d_{1,n} \\ d_{c,2} & d_{1,2} & 0 & \ldots & d_{2,n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ d_{c,n} & d_{1,n} & d_{n,2} & \ldots & 0 \end{bmatrix}$$

2.3.1.2.2 Données de demande en équipements de rechange

Le besoin en pièces de rechange pour chaque SdP est connu avant le départ de l'AdM mobile (demande déterministe). La demande en équipements du SdP_i est représentée par D_i. L'ensemble des données de demande en équipements de rechange est modélisée par la matrice ligne de demande d'équipements $M_R = [0 \quad D_1 \quad \ldots \quad D_n]$.

2.3.1.2.3 Données de temps de trajet

Les données temporelles correspondent d'une part au temps mis par l'AdM mobile pour se déplacer vers un SdP ou vers l'Adm central (matrice des durées de trajet) et d'autre part au temps passé par le personnel de maintenance sur un SdP afin d'assurer les opérations

de maintenance (vecteur de durées de remplacement). On modélise les données relatives aux temps de trajet par :

— les temps de trajet entre l'AdM central et les SdP_i : $T_{c,i}$;

— le temps de trajet entre le SdP_i et le SdP_j : $T_{i,j}$.

On considère que le temps de trajet entre le SdP_i et le SdP_j est identique au temps de trajet entre le SdP_j et le SdP_i ($T_{i,j} = T_{j,i}$). Ces données sont mémorisées dans la matrice des durées de trajet M_{TT}.

$$M_{TT} = \begin{bmatrix} 0 & T_{c,1} & T_{c,2} & \dots & T_{c,n} \\ T_{c,1} & 0 & T_{1,2} & \dots & T_{1,n} \\ T_{c,2} & T_{1,2} & 0 & \dots & T_{2,n} \\ \dots & \dots & \dots & \dots & \dots \\ T_{c,n} & T_{1,n} & T_{n,2} & \dots & 0 \end{bmatrix}$$

2.3.1.2.4 Données des durées d'intervention de maintenance

La politique de maintenance utilisée est une politique de remplacement. On considère donc que les durées d'interventions au niveau des différents sites sont proportionnelles à la demande en équipements de rechange pour chaque SdP (matrice ligne M_R). On a fait l'hypothèse que tous les sites de production ont les mêmes types d'équipements, la durée de remplacement d'un équipement est supposée identique sur tous les sites de production.

La durée de remplacement des équipements sur le site de production SdP_i est représentée par DMP_i : demande en pièces de rechange pour le site i ($DMP_i = M_R[i] \times$ durée d'intervention unitaire). La matrice ligne des durées d'interventions se définit par $M_{RT} = [0 \quad DMP_1 \quad \dots \quad DMP_n]$.

2.3.2 Modélisation de la fonction de coût de la maintenance

Le calcul du coût de maintenance total doit prendre en compte les coûts induits par :

— les équipements de rechange ;

— les équipements de maintenance, réparation et de test ;

— le personnel de maintenance(salaires, formations, etc.) ;

— le transport et la manutention ;

— le réapprovisionnement ;

— le système d'information et de support des activités.

Dans notre contexte de la maintenance centralisée, nous exprimons alors le coût de maintenance (CM) par une partie fixe (K) et une partie variable (CTM). Cette dernière correspond au coût de transport et de manutention. Le coût de maintenance total est exprimé par $CM = CTM + K$.

Pour optimiser le coût de maintenance total (CM), il nous suffit d'optimiser le parcours de l'atelier de maintenance mobile puis ajouter au coût de transport associé à ce parcours (CTM), une constante K qui représente tous les autres coûts de maintenance.

Nous avons exprimé le calcul du coût de transport CTM en fonction de trois paramètres :

— un paramètre kilométrique (CK) : somme des coûts directs de carburant, de pneumatiques, de péages et d'entretien-réparations avec comme unité des euros par kilomètre ;

— un paramètre horaire du personnel de maintenance (CC) : somme des coûts du personnel de maintenance affecté à l'atelier de maintenance mobile ;

— un paramètre journalier fixe de véhicule (CJ) : sommes des coûts fixes de l'atelier de maintenance mobile.

Le coût de transport et de manutention CTM est alors une fonction dépendante des éléments CK, CC et CJ. Ainsi la fonction coût de maintenance que nous cherchons à optimiser s'écrit selon l'équation : $CM = F(CK, CC, CJ) + K$.

2.3.3 Algorithme d'évaluation du coût d'une tournée de l'atelier de maintenance mobile

Nous représentons les positions de l'atelier de maintenance central et des sites de production par des entiers entre 0 et n : 0 représente l'atelier de maintenance centrale, $1 \leq i \leq n$ représente le i-ième site de production SdP_i.

Nous représentons un ordre de visite des sites de production par une permutation sur les entiers 1 à n mémorisée dans une matrice ligne $visit$. Le i-ième site de production à visiter est alors défini par $visit[i]$. La matrice ligne $visit$ à $n + 1$ éléments, $visit[0] = 0$ représente la position de départ qui est l'atelier de maintenance central.

Pour une capacité de chargement q d'équipements de rechange dans l'atelier de maintenance mobile et une planification de l'ordre de visite des n sites de production représentée par $visit$, nous cherchons à calculer le coût de maintenance de cette tournée de l'atelier de maintenance mobile et à calculer les points de retour de l'AdM mobile à l'AdM central pour le réapprovisionner en équipements de rechange. Le chemin calculé sera mémorisé dans une matrice ligne $route$.

L'algorithme VIII.1 présente le calcul du coût de maintenance associée à une tournée prévisionnelle $visit$ avec le calcul des points de retour de l'AdM mobile à l'AdM central.

2.3.4 Usage des métaheuristiques pour l'optimisation des tournées de l'atelier de maintenance mobile

Une métaheuristique est une méthode de résolution approchée permettant de résoudre en un temps de calcul raisonnable des problèmes d'optimisation difficiles. Par exemple des problèmes d'optimisation discrète ayant un nombre exponentiel de combinaisons à explorer pour pouvoir choisir la solution optimale ou des problèmes d'optimisation à variables

Algorithme VIII.1 Calcul du coût d'une tournée de l'atelier de maintenance mobile

procedure CostMMWtour
Input: n : number of production workshop (PW)
Input: q : maximal capacity of parts in the mobile maintenance workshop (MMW)
Input: M_D, M_R, M_{TT}, M_{RT} : demand, geographic and temporal data
Input: CK, CC, CJ : parameters of the transportation cost function F
Input: $visit$: a permutation of the n PW, i.e. a MMW tour
Output: $cost$: transportation cost (CTM)
Output: $route$: a route of the MMW with returns to the Central Maintenance Workshop (CMW)

$route := [0]$ {initialize the route with the CMW position}
$curr\_q := q$ {current capacity of parts in stock in the MMW}
$curr\_pos := 0$ {at the beginning of the tour, the MMW is at the CMW}
$\#PWvisited := 0$ {number of PW visited}
$cost := 0$ {initialize the transportation cost of the MMW}
while $(\#PWvisited < n)$ **do**
    if $(curr\_pos = 0)$ **then** {the MMW goes to the next PW planned}
        $\#PWvisited := \#PWvisited + 1$
        $next\_pos := visit[\#PWvisited]$
        $demand := M_R[next\_pos]$
        $distance := M_D[0, next\_pos]$
        $duration := M_{TT}[0, next\_pos] + M_{RT}[0, next\_pos]$
        $cost := cost + F(distance \times CK, duration \times CC, duration/8 \times CJ)$
        $curr\_q := curr\_q - demand$
        $curr\_pos := next\_pos$
        $route.append(curr\_pos)$
    else
        $next\_pos := visit[\#PWvisited]$
        $demand := M_R[next\_pos]$
        {test if the MMW has sufficient parts in stock to go to the next PW in his tour}
        if $demand > curr\_q$ **then** {the MMW returns to the CMW}
            $distance := M_D[curr\_pos, 0]$
            $duration := M_{TT}[curr\_pos, 0]$
            $cost := cost + F(distance \times CK, duration \times CC, duration/8 \times CJ)$
            $current\_pos := O$
            $current\_q := q$
            $route.append(curr\_pos)$
        else{the MMW goes to the next PW}
            $\#PWvisited := \#PWvisited + 1$
            $distance := M_D[curr\_pos, next\_pos]$
            $duration := M_{TT}[curr\_pos, next\_pos] + M_{RT}[curr\_pos, next\_pos]$
            $cost := cost + F(distance \times CK, duration \times CC, duration/8 \times CJ)$
            $curr\_q := curr\_q - demand$
            $curr\_pos := next\_pos$
            $route.append(curr\_pos)$
        end if
    end if
end while
return $cost, route$

continues, pour lesquels il n'existe pas d'algorithmes permettant d'identifier la meilleure solution possible. Le lecteur trouvera dans [336] les principales métaheuristiques.

L'encyclopédie gratuite Wikipédia définit ainsi les métaheuristiques : « Les métaheuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global (c'est-à-dire l'extremum global d'une fonction), par échantillonnage d'une fonction objectif. Elles se comportent comme des algorithmes de recherche, tentant d'apprendre les caractéristiques d'un problème afin d'en trouver une approximation de la meilleure solution. »

Les algorithmes des métaheuristiques reproduisent les concepts de différents phénomènes physiques, biologiques et naturels et procèdent à des tirages au hasard pour effectuer certains choix ou appliquer certaines règles.

Dans le cas où l'atelier de maintenance mobile aurait une capacité de stockage des équipements de rechange infinie, avec n sites de production on a $(n-1)!/2$ tournées possibles. Hormis des petites instances du problème, l'exploration de toutes les tournées possibles est impraticable : si l'on considère que le temps de calcul par un ordinateur pour évaluer le coût d'une tournée est de une demi-microseconde, pour $n = 20$, le temps d'évaluation de toutes les tournées possibles est de presque 1000 ans. L'utilisation d'une métaheuristique est alors nécessaire.

Dans cette section, nous présentons l'application de deux métaheuristiques, l'*algorithme génétique* et le *recuit simulé*, à notre problème de tournée de l'atelier de maintenance mobile. Nous cherchons à optimiser le coût du parcours de l'atelier de maintenance mobile.

2.3.4.1 Algorithme génétique

Les algorithmes génétiques [337] sont des algorithmes d'optimisation basés sur des techniques dérivées de la génétique et de l'évolution naturelle. L'algorithme démarre avec un ensemble de solutions potentielles du problème, appelées *individus*, constituant une *population*. Un individu est décrit par un ensemble d'informations appelées *chromosomes*. La population évolue au cours des itérations de l'algorithme, appelées *générations*, par trois opérateurs d'évolution :

- la *sélection* : choix des individus les mieux adaptés ;
- le *croisement* : création de nouveaux individus (enfants) par le mélange des chromosomes contenus dans les individus actuels (parents) ;
- la *mutation* : application de modifications aléatoires sur les chromosomes d'un individu.

Pour évaluer les individus au problème au fil des générations, on associe à un individu une mesure de performance calculée selon ses chromosomes. C'est le critère à optimiser.

Application de l'algorithme génétique à la maintenance multi-sites

Un individu correspond à une tournée qui a comme point de départ et d'arrivée l'AdM central après une visite de l'ensemble des sites de production. Un chromosome correspond à une portion d'une tournée. La population sur laquelle on travaille correspond à une

collection de tournées constituée au départ d'une manière aléatoire puis évoluant pour ne garder que les meilleures tournées au fil des itérations. La sélection des meilleures tournées se fait à travers l'évaluation du coût de maintenance de chaque tournée de la population (Algorithme VIII.1). L'évolution de la population se fait par application de différents opérateurs de mutation sur un ensemble de tournées sélectionnées.

Pour la création de nouveaux individus, il est nécessaire de choisir une stratégie de sélection des individus les plus prometteurs sur lesquels seront appliqués les opérateurs de croisement et de mutation. Plusieurs stratégies de sélection existent : la méthode de la *loterie biaisée*, la méthode *élitiste*, la sélection *par tournois* et la sélection *universelle stochastique*. Pour ne pas perdre les meilleurs individus au fil des générations, nous avons choisi la stratégie *élitiste* : nous conservons les m meilleurs individus de la génération i pour les intégrer dans la population de la génération $i+1$. Le reste de la population de la génération $i+1$ est créé en appliquant des opérateurs génétiques de mutation tels que *Flip*, *Swap* et *Slide*.

L'algorithme VIII.2 présente l'adaptation de l'algorithme génétique au problème d'optimisation de la tournée de l'atelier de maintenance mobile.

2.3.4.2 Recuit simulé

La méthode du recuit simulé [338] s'inspire du recuit des métaux en métallurgie et est basée sur la méthode de simulation de Metropolis [339].

Le but est de trouver une *configuration optimale*. Une fonction associée à une configuration permet de calculer son *énergie* E. Une configuration optimale est atteinte lorsque celle-ci arrive sur un *état stable* avec une énergie minimale. Cet état stable est atteint par *paliers*. Une *température* T est associée à chaque palier, un changement de palier correspond à un refroidissement de la température par un *facteur de refroidissement* K.

Pour un palier fixé, un nombre $MaxIter$ de tentatives d'amélioration de la configuration est effectué. On part d'une configuration courante, et l'on choisit aléatoirement une nouvelle configuration issue de son *voisinage*. Si la nouvelle configuration fait diminuer son énergie, celle-ci est acceptée. Sinon, elle n'est acceptée qu'avec une probabilité égale à $exp(-\Delta(E)/T)$ où $\Delta(E)$ est le différentiel entre l'énergie de la configuration courante et celle de son voisinage. Si la nouvelle configuration est acceptée, on l'utilisera comme configuration courante pour la prochaine tentative d'amélioration.

Pendant les $MaxIter$ tentatives d'amélioration de la configuration, on tient à jour le nombre de nouvelles configurations acceptées. Un *taux d'acceptation par palier* est alors calculé. Un état stable sera atteint lorsque le taux d'acceptation calculé pour le palier courant deviendra inférieur à un taux d'acceptation minimal AR_{min} fixé. Ce *critère d'arrêt* exprime qu'il n'est plus possible d'améliorer la configuration minimale atteinte.

Application du recuit simulé à la maintenance multi-sites

Une configuration correspond à une tournée de l'atelier de maintenance mobile. L'énergie correspond au coût de la maintenance induit par cette tournée. Un voisin d'une tournée correspond à la permutation aléatoire de deux sites de production dans la tournée.

Algorithme VIII.2 Algorithme génétique pour le problème de tournée de l'atelier de maintenance mobile

procedure GeneticAlgorithmMMW
Input: n : number of production workshop (PW)
Input: q : maximal capacity of parts in the mobile maintenance workshop (MMW)
Input: M_D, M_R, M_{TT}, M_{RT} : demand, geographic and temporal data
Input: CK, CC, CJ : parameters of the transportation cost function F
Input: $Maxiter$: number of iterations of the genetic algorithm
Input: m : number of the best individuals used for the next generation of the population
Input: $size$: size of the population
Output: $OptRoute$: optimal route of the MMW

1: $curr\_generation :=$ CreateInitialGeneration$(size, n)$ {create a population of size individuals}
2: **for** $i := 1$ **to** $MaxIter$ **do**
3: **for** $p := 1$ **to** $size$ **do**
4: $cost[i] :=$ CostMMWTour$(\ldots, curr_generation[i])$ {Algorithm VIII.1 applied to $curr\_generation[i]$}
5: **end for**
6: *Select the m best individuals of curr_generation and insert then in next_generation*
7: *Complete next_generation by applying the genetic mutation operator on the other individuals of curr_generation*
8: $curr\_generation := next\_generation$
9: **end for**
10: *OptRoute := the individual of the last generation which has the minimal cost*
11: **return** $OptRoute$

L'algorithme VIII.3 présente l'adaptation de l'algorithme du recuit simulé au problème d'optimisation de la tournée de l'atelier de maintenance mobile.

2.3.5 Analyse des différentes métaheuristiques pour l'optimisation des tournées de l'atelier de maintenance mobile

Nous présentons dans [328] un exemple illustratif avec onze localisations géographiques : une localisation pour le site de l'atelier de maintenance central et dix localisations pour les sites de production.

Pour chaque métaheuristique, nous avons étudié l'influence de ses paramètres en les faisant varier sur la qualité du coût de la maintenance produit par cette métaheuristique. Pour l'algorithme génétique, nous avons étudié par exemple l'influence de la taille de la population sur la qualité du coût de la maintenance et pour le recuit simulé, nous avons étudié par exemple l'influence de la température initiale. Pour obtenir les meilleurs résultats de chaque métaheuristique, nous les avons simulés en faisant varier les différentes caractéristiques du problème. Nous avons ainsi recherché la position optimale de l'AdM central et la capacité de stockage optimale de l'AdM mobile.

En faisant une comparaison en termes de coût de maintenance calculé par les deux métaheuristiques, sur notre exemple applicatif l'algorithme génétique fournit une solution avec un coût de maintenance inférieur à celui du recuit simulé. Une comparaison en termes de temps de calcul entre ces deux métaheuristiques est plus complexe à faire, car les deux algorithmes ne fonctionnent pas de la même façon. Avec l'algorithme génétique, la taille de la population ainsi que le nombre d'itérations influencent fortement le temps de calcul tandis que pour le recuit simulé c'est la température initiale ainsi que le coefficient de refroidissement qui influencent le temps de calcul.

3 Conclusion et perspectives sur la maintenance

Nous avons étudié dans ce chapitre la mise en œuvre d'une maintenance corrective centralisée pour un système de production multi-sites. Nous nous sommes intéressés à la conception d'un atelier de maintenance central pour optimiser le temps de séjour des équipements à réparer dans l'atelier afin d'augmenter la productivité des sites de production, et à la planification optimale des tournées de l'atelier de maintenance mobile afin de réduire les coûts de l'équipe de maintenance mobile.

Dans la première partie de nos travaux, nous avons proposé une organisation de l'atelier de maintenance central selon différents services (diagnostic, démontage, réparation, réassemblage, test) et nous avons évalué deux techniques pour diminuer le temps de séjour des équipements : la duplication des stations et la réparation par remplacement. Afin d'étudier les performances de l'AdM central selon ces différentes techniques et en prenant en compte le dimensionnement de l'AdM central (nombre de stations dupliquées, quantité initiale des équipements de rechange), nous avons modélisé l'AdM central par un réseau de files d'attente et avons développé un outil de simulation de réseaux de files d'attente dans

Algorithme VIII.3 Algorithme du recuit simulé pour le problème de tournée de l'atelier de maintenance mobile

procedure SIMULATEDANNEALINGALGORITHMMMW
Input: n : number of production workshop (PW)
Input: q : maximal capacity of parts in the mobile maintenance workshop (MMW)
Input: M_D, M_R, M_{TT}, M_{RT} : demand, geographic and temporal data
Input: CK, CC, CJ : parameters of the transportation cost function F
Input: $MaxIter$: number of iterations of the genetic algorithm
Input: T_0 : initial temperature
Input: AR_{min} : minimal acceptation rate
Input: K : cooling factor (rate at which the temperature is reduced)
Output: $OptRoute$: optimal route of the MMW

1: $curr\_T := T_0$
2: $curr\_route :=$ *InitialSolution(n)* {we generate an initial solution randomly or using heuristic}
3: $curr\_cost :=$ *CostMMWTour(..., curr_route)* {Algorithm VIII1 applied to $curr\_route$}
4: $curr\_rate := 100$
5: **while** $(curr\_rate \geq AR_{min})$ **do**
6: $\quad \#accepted\_route := 0$
7: $\quad$ **for** $i := 1$ **to** $MaxIter$ **do**
8: $\quad\quad next\_route :=$ *Neighnour(curr_route)* {choose randomly a neighbouring solution}
9: $\quad\quad next\_cost :=$ *CostMMWTour(..., next_route)*
10: $\quad\quad \Delta(cost) := next\_cost - curr\_cost$
11: $\quad\quad$ **if** $(\Delta(cost) < 0$ **then**
12: $\quad\quad\quad curr\_cost := next\_cost$
13: $\quad\quad\quad curr\_route := next\_route$
14: $\quad\quad\quad \#accepted\_route := \#accepted\_route + 1$
15: $\quad\quad$ **else**
16: $\quad\quad\quad p := Random(0, 1)$
17: $\quad\quad\quad \alpha := exp(-\Delta(cost)/current\_T)$
18: $\quad\quad\quad$ **if** $(p < \alpha)$ **then**
19: $\quad\quad\quad\quad curr\_cost := next\_cost$
20: $\quad\quad\quad\quad curr\_route := next\_route$
21: $\quad\quad\quad\quad \#accepted\_route := \#accepted\_route + 1$
22: $\quad\quad\quad$ **end if**
23: $\quad\quad$ **end if**
24: $\quad$ **end for**
25: $\quad curr\_rate := (\#accepted\_route/MaxIter) \times 100$
26: $\quad curr\_T := curr\_T \times K$ {the temperature is reduced}
27: **end while**
28: $OptRoute := curr\_route$
29: **return** $OptRoute$

Matlab/Simulink. Nous avons pu ainsi développer une méthodologie de dimensionnement de l'AdM central qui vise à réduire le temps de séjour des équipements améliorant ainsi la productivité des sites de production. La conception de l'AdM central sera obtenue par le meilleur compromis entre la disponibilité résultant de l'équipement réparé, la taille de l'AdM (nombre de stations dupliquées) et les coûts de maintenance (prenant en compte la quantité initiale des équipements de rechange).

Une première perspective de ce travail est de développer un algorithme de recherche du dimensionnement de l'AdM central selon des paramètres de coût et de performance souhaités : déterminer les stations à dupliquer et choisir le niveau de stock initial. Une seconde perspective est d'étendre notre méthodologie pour prendre en compte l'obsolescence des équipements (planification progressive des équipements, reconstitution du stock d'équipements et de pièces de rechange). Une dernière perspective est d'intégrer cette méthodologie de dimensionnement d'un atelier de maintenance dans un contexte distribué : prise en compte des demandes des différents sites avec des politiques de réparation différentes et auxquelles sont associées différents niveaux de priorités.

Dans la deuxième partie de nos travaux, nous avons proposé une méthodologie d'optimisation des coûts de l'atelier de maintenance mobile basée sur l'utilisation des métaheuristiques plus particulièrement de l'algorithme génétique et du recuit simulé. Notre méthodologie s'appuie sur trois étapes : la première étant celle de l'acquisition des données (distances, temps de transport, demandes des sites de production,etc.), la seconde définit la fonction du coût de maintenance à optimiser et la dernière étape consiste à appliquer les algorithmes d'optimisation en faisant varier ses différents paramètres et réaliser une analyse comparative des résultats de simulation afin de choisir la solution optimale. En appliquant cette méthodologie sur un exemple applicatif, nous avons constaté la prédominance du coût de transport dans le calcul du coût de maintenance. Nous avons comparé l'algorithme génétique et le recuit simulé sur notre problème particulier et avons constaté suite aux simulations que l'algorithme génétique reste en général plus efficace en proposant un coût de maintenance le plus faible. La comparaison des temps de calcul donne aussi l'avantage à l'algorithme génétique.

La première perspective de ce travail est d'étudier de nouvelles méthodes de création du voisinage d'une tournée pour le recuit simulé (autres méthodes de permutation des SdP en vue de la création de nouvelles tournées à évaluer) et d'utiliser de nouveaux opérateurs de mutation et de croisement pour l'algorithme génétique pour créer une nouvelle génération de tournées de l'AdM mobile. Une seconde perspective est la prise en compte de l'incertain dans la planification de la tournée de l'AdM mobile (retards, pannes, priorité donnée à certains sites, etc.).

Axe C)

Pronostic de systèmes complexes par des méthodes basées sur les données

IX Pronostic de l'état de santé de systèmes complexes **185**

1 Présentation générale : contexte, état de l'art, positionnement, originalité . . . 185

2 Synthèse des travaux développés : pronostic de l'état de santé des batteries lithium-ion de véhicules électriques 191

3 Conclusion et perspectives sur le pronostic de l'état de santé de systèmes complexes 204

Collaborations et encadrements de travaux de recherche dans l'Axe C)

Les travaux de recherche relatés dans le chapitre suivant IX ont été réalisés par une collaboration avec Zineb Simeu-Abazi (laboratoire G-SCOP, Polytech Grenoble) et Peggy Zwolinski (laboratoire G-SCOP, Grenoble INP – Génie industriel) lors du co-encadrement d'une thèse. Le chapitre IX présentera une synthèse des travaux réalisés et citera les publications issues de cette collaboration et encadrement de thèse.

Publications issues de collaborations et d'encadrements de thèses et de stages de Master dans l'Axe C)

(RI : Revue Internationale, CI : Conférence Internationale)

[RI-1] A. Basia, Z. Simeu-Abazi, **E. Gascard**, and P. Zwolinski, "Review on State of Health estimation methodologies for lithium-ion batteries in the context of circular economy ", *CIRP Journal of Manufacturing Science and Technology*, vol. 32, pp. 517–528, 2021.

[CI-1] A. Basia, **E. Gascard**, Z. Simeu-Abazi, and P. Zwolinski, "Overcoming the Barriers in Diagnostics and Prognostics of the Circular Industrial System by Hidden Markov Model", in *Proceedings of the 3rd International Conference on Control, Automation and Diagnosis (ICCAD'19)*, IEEE Press, 2019, pp. 314–319.

[CI-2] A. Basia, Z. Simeu-Abazi, **E. Gascard**, and P. Zwolinski, "First step towards the development of a Prognosis Health Management (PHM) System for Li-ion batteries: An FMMEA based approach", in *Proceedings of the 29th European Safety and Reliability Conference (ESREL 2019)*, Research Publishing, Singapore, 2019, pp. 1194–1200.

[CI-3] A. Basia, Z. Simeu-Abazi, **E. Gascard**, and P. Zwolinski, "State of Health Estimation for Lithium-ion Battery by Incremental Capacity Based ARIMA - SVR Model", in *Proceedings of the 31st European Safety and Reliability Conference (ESREL 2021)*, Research Publishing, Singapore, 2021, pp. 1–6.

[CI-4] A. Basia, Z. Simeu-Abazi, **E. Gascard**, and P. Zwolinski, "Comparison of data driven algorithms for SoH estimation of Lithium-ion batteries", in *Proceedings of the 5th International Conference on Control, Automation and Diagnosis (ICCAD'21)*, IEEE, 2021, pp. 1–6.

Chapitre IX

Pronostic de l'état de santé de systèmes complexes : Application aux batteries au lithium-ion

1 Présentation générale : contexte, état de l'art, positionnement, originalité

1.1 Contexte

Ce chapitre présente le dernier thème de mes travaux de recherche sur la sûreté de fonctionnement, le pronostic de défaillances. *Plus précisément, je me suis intéressé au développement de méthodologies de pronostic de l'état de santé de systèmes complexes par des approches basées sur les données.* Il existe dans la littérature de nombreuses formulations pour définir le pronostic et le résultat attendu du pronostic, nous considérons celles issues de la norme ISO 13381-1 [340] : « Le pronostic est une estimation de la durée de fonctionnement avant défaillance et de la probabilité d'existence ou d'apparition ultérieure d'un ou de plusieurs modes de défaillance . Il repose sur la connaissance détaillée du processus de propagation des défauts et sur l'expérience acquise. Le but du pronostic est de fournir à l'utilisateur un moyen pour prédire la durée de vie utile restante avec un niveau de confiance satisfaisant. »

Cette définition associe le pronostic de défaillances d'un système à un processus de prédiction de la durée de fonctionnement avant défaillance (appelée RUL : Remaining Useful Life) en supposant connu l'état de santé courant du système.

La figure IX.1 présente la chronologie des étapes de détection, pronostic et diagnostic. Le diagnostic est postérieur à l'apparition d'une défaillance sur le système alors que le pronostic est antérieur à la survenue de la défaillance.

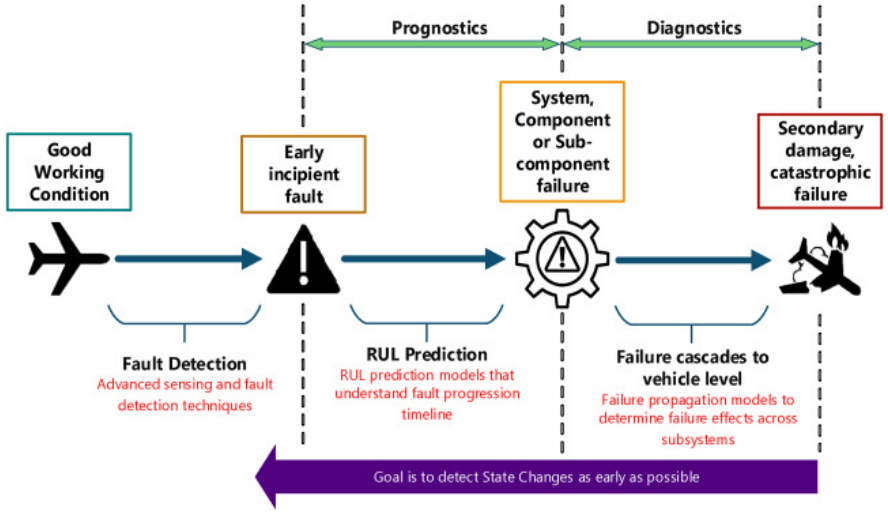


FIGURE IX.1 – Chronologie de la détection, pronostic et diagnostic des défaillances [341] adaptée de [3]

Notre étude du pronostic porte un système complexe qui est défini comme un système qui comporte un nombre important de composants ou de sous-systèmes ayant les caractéristiques suivantes :

— différents types de composants : électriques, mécaniques, chimiques, etc. ayant chacun ses propres modes de dégradation ;

— de nombreuses interactions entre les composants ou sous-systèmes ;

— le système évolue dans un environnement dynamique et incertain et sera alors utilisé dans plusieurs conditions opérationnelles.

Depuis plus de deux décennies avec le progrès et l'essor de l'intelligence artificielle et de la science des données, de nouveaux travaux en sûreté de fonctionnement ont porté sur la prédiction des défaillances. Ainsi les disciplines du pronostic et management de la santé (appellé PHM : Prognostics and Health Management) [342] et de la maintenance conditionnelle (appelée CBM : Condition-based maintenance) [343] se sont développées pour anticiper les défaillances, proposer de nouvelles solutions pour réduire les coûts de maintenance et minimiser les risques de défaillances ainsi que leurs conséquences.

La maintenance conditionnelle diffère des maintenances traditionnelles telles que la maintenance préventive ou corrective : ses interventions sont réalisées avant une défaillance et ne sont pas planifiées indépendamment de l'état du système. La maintenance conditionnelle utilise les connaissances de l'état courant du système (estimé ou réel) et les projette dans le futur pour prévoir les futures actions de maintenance. La CBM permet ainsi d'augmenter la disponibilité, la fiabilité et la sécurité du système tout en réduisant les coûts de maintenance.

La mise en œuvre de la maintenance conditionnelle requiert des méthodes, outils et algorithmes pour le diagnostic de l'état courant des composants et du système complexe,

l'étude de leurs dégradations, le pronostic de leurs états de santé et l'estimation de leurs durées de vie résiduelle. Ces tâches font partie du pronostic et de la gestion de l'état de santé de systèmes industriels (PHM).

Le développement d'un modèle de la dégradation d'un système complexe et l'estimation de sa durée de vie résiduelle sont des verrous scientifiques actuels sur lesquels la communauté PHM travaille. Les problématiques à prendre en compte sont multiples :

— la dégradation d'un système est un processus dynamique complexe, il est non mesurable directement par un capteur et nécessite un arrêt du système, le démontage de ses composants et des tests sur ceux-ci pour pouvoir être observé . Cependant, pendant l'exploitation du système, nous ne disposons que de mesures sur le système et ses composants du système par différentes métriques (nombre d'utilisation, températures, vibrations, etc.) qui permettent de caractériser leurs niveaux de dégradation et d'estimer leurs RUL ;

— la dynamique de dégradation du système dépend de facteurs internes, il y a des interactions entre les composants, et de facteurs externes, les conditions opérationnelles dans lequel le système évolue ;

— le système peut évoluer dans différentes conditions opérationnelles, il faut tenir compte des effets de chaque condition de fonctionnement séparément, mais également leur combinaison qui peut produire un vieillissement accéléré du système.

1.2 Etat de l'art sur les méthodes de pronostic, positionnement

De nombreuses méthodes de pronostic de défaillances ont été proposés durant les deux dernières décennies. Le lecteur peut se référer aux articles ou thèses suivantes pour avoir un état de l'art exhaustif sur les méthodes de pronostic et leurs utilisations dans la maintenance conditionnelle [3], [7], [344]-[356].

Différentes classes ont été définies pour regrouper des méthodes de pronostic similaires afin de proposer une classification : pronostic basé sur un modèle, pronostic avec modèles physiques, approche basée sur des modèles de durée de vie, pronostic guidé par des données, pronostic utilisant des outils de l'intelligence artificielle, approche utilisant des réseaux de neurones, pronostic basé sur des outils statistiques, pronostic basé sur l'expérience, approche basée sur la connaissance, approche combinant pronostic guidé par les données avec le pronostic basé sur l'expérience, approche basée sur la fusion de modèles, approche hybride.

La communauté PHM utilise depuis plusieurs années une classification des méthodes de pronostic en trois classes [357] composées de :

— approches basées sur des modèles physiques ;

— approches guidées par les données ;

— approches hybrides.

1.2.1 Approches de pronostic basées sur des modèles physiques

Les méthodes de pronostic basées sur des modèles physiques modélisent mathématiquement les processus de dégradation des composants (usure, fissures, corrosion, etc.) pour prédire l'évolution future de leurs états de dégradation et en déduire le moment où la dégradation atteindra le seuil de défaillance. Ces approches utilisent des modèles issus de lois de la physique (hydraulique, électricité, chimie, mécanique, etc.), les lois de fatigue et de résistance des matériaux. Les outils utilisés dans cette approche sont essentiellement les équations algébro-différentielles.

Cette approche nécessite des connaissances spécifiques et une forte maitrise sur les mécanismes de dégradation des composants. Cette approche applicable au niveau composant est difficilement généralisable à un système dû à la complexité de modélisation des interactions entre les différents composants et avec les facteurs extérieurs.

La mise en œuvre du pronostic par une approche basée sur la physique de défaillance nécessite de suivre le comportement du système et sa dégradation. Les techniques / outils appartenant à cette approche sont l'espace de parité, les observateurs et l'estimation paramétrique. La figure IX.2 présente le principe du diagnostic et du pronostic basé sur un modèle physique.

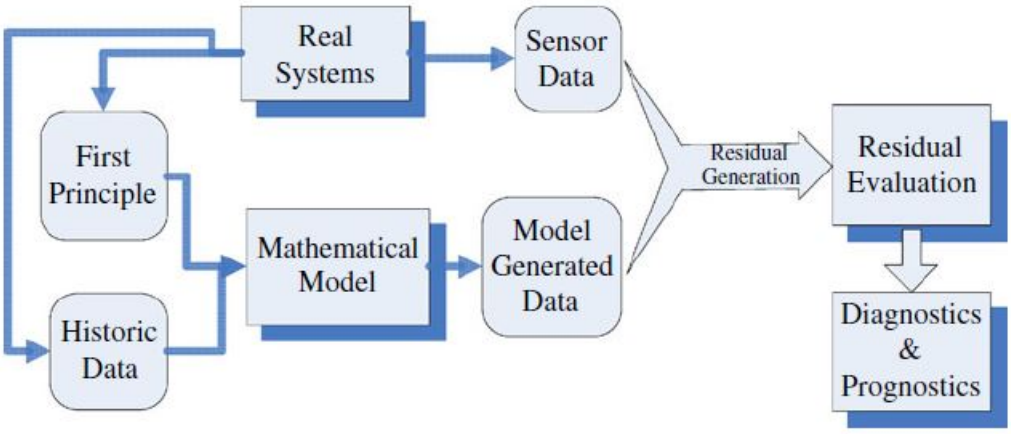


FIGURE IX.2 – Principe du pronostic basé sur un modèle physique [349]

Des exemples de travaux sur le pronostic en utilisant une approche basée sur des modèles physiques peuvent être consultés dans [358]-[363].

1.2.2 Approches de pronostic guidées par les données

Les méthodes guidées par les données cherchent à extraire d'un historique de données de surveillance sur les composants d'un système (données obtenues par l'observation du système analysé ou issues de mesures passées de composants similaires) des modèles de l'évolution de l'état de santé des composants de leurs mises en fonctionnement jusqu'à leurs

défaillances. Cette approche est utile lorsqu'on dispose d'un grand nombre de données, mais qu'on n'a pas une connaissance physique de la nature des dégradations que peut subir le système. Elle suppose cependant que les caractéristiques statistiques des données issues de plusieurs composants sont similaires.

La mise en œuvre des approches guidées par les données comporte deux phases. La première phase hors ligne (c'est-à-dire avant la mise en fonctionnement du système à surveiller) porte sur la construction d'une base de modèles de prédiction de l'état de santé du système à partir d'un historique de données sur des systèmes similaires. La seconde phase en ligne (c'est-à-dire pendant le fonctionnement du système analysé) estime l'état de santé courant du système et choisit le modèle de prédiction le plus adapté contenu dans sa base de modèles pour prédire la durée de fonctionnement avant défaillance. La figure IX.3 présente le principe du pronostic guidé par les données.

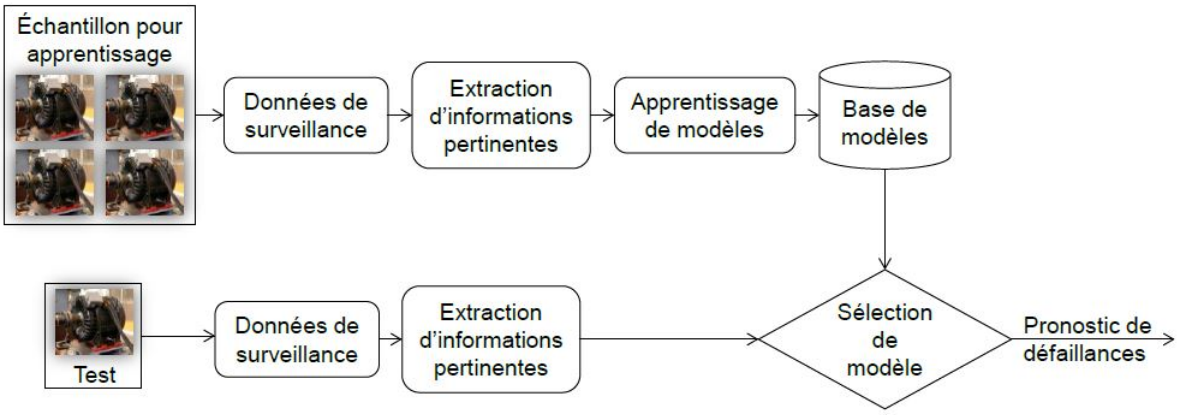


FIGURE IX.3 – Principe du pronostic guidé par les données [364]

Les méthodes de pronostic guidé par les données peuvent être regroupées en trois catégories selon la nature des techniques employées :

— le pronostic par analyse statistique ou par tendance ;

— le pronostic par apprentissage automatique issu de l'intelligence artificielle ;

— le pronostic par estimateur d'état.

Pronostic par analyse statistique ou par tendance

Ces méthodes de pronostic font appel à des approches statistiques telles la régression linéaire (Linear Regression), le lissage exponentiel (Exponential Smoothing), l'analyse en composante principale (Principal Component Analysis) ainsi qu'à des modèles à espaces d'états (modèle de Markov caché (Hidden Markov Model) , réseaux bayésiens dynamiques (Dynamic Bayesian Network)) et des outils de prévision de séries temporelles (modèle auto-régressif de type ARMA, ARIMA).

Pronostic par apprentissage automatique

Ces méthodes issues de l'intelligence artificielle permettent de capturer les relations subtiles ou cachées entre les données. Elles sont de type « boite noire », c'est-à-dire qu'il est difficile d'expliquer le lien logique entre les données fournies en entrée et les sorties retournées. Les principales techniques utilisées sont le réseau de neurones artificiels (Artificial Neural Network) et ses variantes (réseau neuronal convolutif, réseau de neurones récurrents, etc.), le système neuro-flou (neuro-fuzzy system), la machine à vecteur de support (Support Vector Machine).

Pronostic par estimateur d'état

Le pronostic par estimateur d'état est utilisé lorsqu'un diagnostic est réalisé par une technique de reconnaissance de formes. Il est adapté à des mécanismes de dégradation lents. Le pronostic consiste alors à prédire l'évolution de la trajectoire de la forme par exemple par l'intermédiaire d'un filtre de Kalman.

Le lecteur trouvera dans [365]-[373] des exemples de travaux de recherche sur le pronostic guidé par les données.

1.2.3 Approches de pronostic hybride

Les approches de pronostic hybrides (appelée également fusion de méthodes de pronostic) visent à intégrer les avantages de différentes approches et à minimiser leurs inconvénients pour fournir une meilleure estimation de l'état de santé du système et de ses composants.

Une méthode de pronostic hybride fait appel à un modèle de pronostic basé sur la physique des défaillances et à une approche de pronostic guidée par les données, on parle alors de méthodes fusionnées de pronostic.

Le lecteur peut se référer à [374]-[381] pour avoir différents exemples de méthodes de pronostic hybrides.

1.2.4 Synthèse des approches de pronostic

Dans les approches basées sur des modèles physiques, la connaissance du modèle mathématique du comportement dynamique de la dégradation rend leur utilisation très aisée. Cependant, le développement d'un tel modèle est coûteux, car il nécessite un haut niveau de qualification pour maitriser les mécanismes de dégradation des composants.

Les approches basées sur les données sont recommandées pour les systèmes dont la connaissance de leur physique de dégradation est limitée ou inexistante, mais dont on possède des données en grande quantité. Cependant, pour appliquer les approches basées sur les données, celles-ci doivent être de qualité et doivent provenir de toute la période d'exploitation du système. Ainsi, cette approche ne peut pas être appliquée dans le cas de nouveaux systèmes pour lesquels il y a peu de données ou de systèmes ayant un nombre faible d'exemplaires, les données ne seraient pas alors assez représentatives. Un autre inconvénient de cette approche est qu'il est nécessaire de filtrer et de prétraiter les données

pour en conserver que les données utiles, car les données fournies par les capteurs ne sont pas toujours directement exploitables.

Les approches hybrides présentent de bonnes performances cependant elles présentent les inconvénients des deux approches précédentes : elles peuvent être coûteuse en ressources de calcul pour les étapes de traitement des données et d'apprentissage et sont limitées dans leur usage par la nécessité de la connaissance d'un modèle physique de phénomènes de dégradation.

1.2.5 Positionnement, originalité

Nos travaux de recherche sur le développement de méthodes de pronostic se sont axés sur les méthodes de pronostic basées sur les données.

Dans le cadre de la thèse de Akash Basia [382] nous avons développé des modèles de dégradation des batteries lithium-ion utilisées dans les véhicules électriques en utilisant des approches de pronostic basées sur les données comme les modèles de Markov cachés, les modèles auto-régressif de types ARIMA, la régression à vecteur de support, la régression linéaire et les réseaux de neurones à propagation avant.

Nos travaux ont plus spécifiquement porté sur l'évaluation de santé des batteries lithium-ion dans le contexte de l'économie circulaire en proposant un indicateur de santé pour faciliter une meilleure prise de décision lors de la réutilisation des batteries de véhicules électriques en fin de vie.

2 Synthèse des travaux développés : pronostic de l'état de santé des batteries lithium-ion de véhicules électriques

Nous exposons dans cette section les principes généraux des méthodologies de pronostic de l'état de santé de batteries lithium-ion que nous avons développé pendant la thèse de A. Basia, notre présentation s'axera sur l'usage de modèles d'apprentissage et de prédiction à base de données.

Le lecteur peut se référer à la thèse de A. Basia [382] et à nos publications [383]-[387] pour une présentation des sujets connexes comme le principe de fonctionnement des batteries lithium-ion et leurs processus de dégradation, l'économie circulaire, la définition d'un indicateur de santé d'une batterie dans le cadre de son reconditionnement et de sa réutilisation dans un contexte industriel.

2.1 Données caractérisant l'état de santé des batteries lithium-ion

Les packs de batteries lithium-ion utilisées dans les véhicules électriques disposent d'un système électronique qui supervise la gestion des batteries, c'est le *Battery Management System* (BMS). Le BMS est en charge de mesurer différents éléments sur les cellules composant la batterie et sur la batterie dans sa globalité tels que : la tension, le courant, la température, etc. au fil des cycles de charge / décharge, voir figure IX.4.

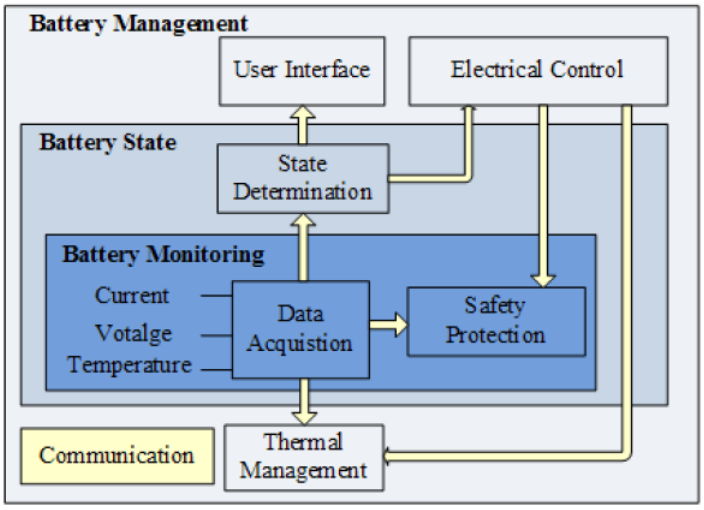


FIGURE IX.4 – Structure d'un système de gestion des batteries [388]

A partir des mesures de tension et courant mesurées lors des cycles de charge / décharge d'une batterie, une analyse incrémentale de la capacité (incremental capacity analysis ICA) peut être réalisée. De cet indicateur, l'état de santé (SoH) de la batterie peut être estimé [389], [390].

D'autres indicateurs comme la tension différentielle ou la résistance interne peuvent être également utilisés pour estimer l'état de santé d'une batterie [391]-[393].

A partir des indicateurs d'analyse incrémentale de la capacité, de tension différentielle et de résistance interne, des méthodes basées sur les données peuvent être utilisées pour prédire l'état de santé de la batterie. Ces méthodes d'apprentissage peuvent être classifiées en deux familles : les méthodes d'apprentissage non probabilistes et les méthodes d'apprentissage probabilistes, voir figure IX.5 (les termes abrégés pour les méthodes sont : Fuzzy Logic (FL), Support Vector Machine (SVM), Auto regressive Model (ARMA), Least Square (LS), Artificial Neural Network (ANN), Bayesian Network (BN), Hidden Markov Model (HMM), Gaussian Process Regression (GPR), Particle Filter (PF), Sample Entropy (SE) et Relevance Vector Machine (RVM)).

Nous présentons dans le reste de cette section les deux méthodologies de pronostic du SoH de batteries lithium-ion que nous avons développées :

— la première utilise un modèle probabiliste, le modèle de Markov caché (HMM) ;

— la seconde combine deux méthodes non probabilistes : la régression à vecteur de support (SVR) et un modèle auto-régressif (ARIMA).

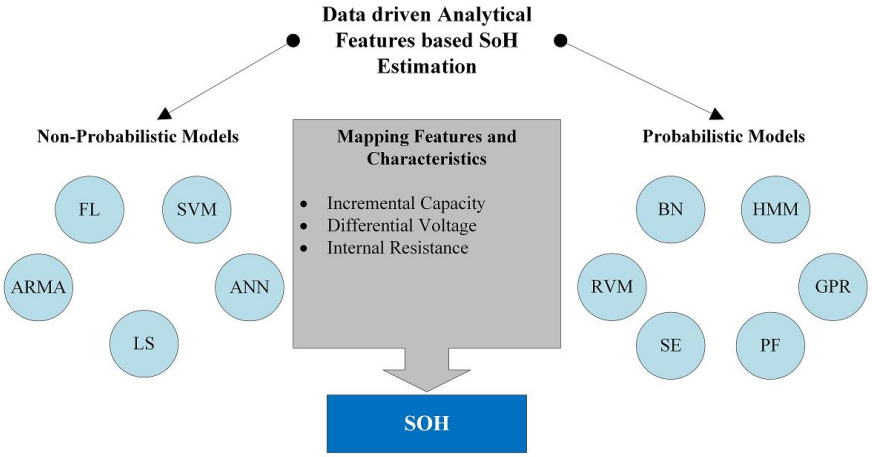


FIGURE IX.5 – Modèles probabilistes et non probabilistes basés sur les données pour la prédiction de l'état de santé [387]

2.2 Pronostic par modèle de Markov caché

2.2.1 Présentation des modèles de Markov cachés

2.2.1.1 Définition d'un HMM

Un modèle de Markov caché (appelé HMM : Hidden Markov Model) est basé sur un modèle de Markov dont on ne peut pas observer directement la séquence d'états, les états sont *cachés*. Cependant, on peut observer des symboles émis par les états selon une loi de probabilité.

Un modèle de Markov caché à temps discret $\mathcal{H} = \langle S, V, A, B, \Pi \rangle$ est défini par les éléments suivants [394] :

— $S = \{S_1, S_2, S_3, \ldots, S_N\}$ un ensemble de N états. L'état où se trouve le HMM à l'instant t est noté q_t ($q_t \in S$). Nous considérons une discrétisation du temps avec $t \geq 1$.

— $V = \{v_1, v_2, v_3, \ldots, v_M\}$ un alphabet de M symboles observables. Lorsque le HMM est dans l'état q_t, une seule observation est émise et notée o_t.

— $A = [a_{ij}]$ la matrice de probabilités de transition entre les états : a_{ij} représente la probabilité que le modèle évolue de l'état S_i vers l'état S_j et se défini par $a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i)$ pour $i, j \in [1\ N]$ et $t \geq 1$. La matrice A vérifie : $\forall i, j \in [1, N]\ a_{ij} \geq 0 \wedge \sum_{j=1}^{N} a_{ij} = 1$.

— $B = [b_{jm}]$ la matrice de probabilités d'observation des symboles dans chacun des états du modèle : b_{jm} représente la probabilité que l'on observe le symbole v_m alors que le modèle se trouve dans l'état S_j et se définit par $b_{jm} = P(o_t = v_m \mid q_t = S_j)$ pour $j \in [1, N]$ et $m \in [1, M]$. La matrice B vérifie : $\forall j \in [1, N]\ \forall m \in [1, M]\ b_{jm} \geq 0 \ \wedge \sum_{k=1}^{M} b_{jk} = 1$.

— $\Pi = [\pi_i]$ le vecteur de probabilités initiales. Pour tout état S_i, π_i est la probabilité

que l'état de départ du HMM soit l'état S_i et se définit par $\pi_i = P(q_1 = S_i)$ pour $i \in [1, N]$. La matrice Π vérifie : $\forall i \in [1, N]\ \pi_i \geq 0\ \wedge\ \sum_{i=1}^{N} \pi_i = 1$.

Les modèles de Markov cachés supposent les deux hypothèses suivantes [394] :

(I) Hypothèse Markovienne : absence de mémoire à l'intérieur du processus stochastique :

$$P(q_t = S_i \mid q_{t-1} = S_j, q_{t-2} = S_k, \ldots) = P(q_t = S_i \mid q_{t-1} = S_j) \tag{1}$$

(II) Hypothèse d'indépendance des observations : les observations générées par le modèle Markov caché sont supposées indépendantes entre elles. La probabilité d'une observation o_t ne dépend que de l'état q_t qui a produit l'observation et non de tout autre état ou de toute autre observation :

$$P(o_t \mid q_t, q_{t-1}, q_{t-2}, \ldots, o_{t-1}, o_{t-2}, \ldots) = P(o_t \mid q_t) \tag{2}$$

2.2.1.2 Simulation d'un HMM

La simulation d'un HMM pour générer T observations est défini par la procédure suivante [394] :

1. Choisir aléatoirement un état initial $q_1 = S_i$ selon la distribution initiale Π.
2. Définir $t = 1$.
3. Choisir $O_t = v_k$ selon la distribution des observations dans l'état S_i, c'est-à-dire selon b_{ik}.
4. Choisir le nouvel état $q_{t+1} = S_j$ selon les probabilités de transitions de l'état S_i, c'est-à-dire selon a_{ij}.
5. Définir $t = t + 1$; si $t \leq T$ alors retour à l'étape (3). Sinon fin de la procédure de génération de la séquence d'observations.

2.2.1.3 Problèmes fondamentaux liés aux HMM

Il existe trois problèmes fondamentaux qu'il est possible de résoudre grâce à un modèle de Markov caché [394] :

i. Problème d'évaluation : évaluation de la probabilité de générer une observation. Etant donné une séquence d'observations $\mathcal{O} = \langle o_1, \ldots, o_T \rangle$ et une HMM $\mathcal{H} = \langle S, V, A, B, \Pi \rangle$, il s'agit de déterminer la probabilité avec laquelle la HMM $\mathcal{H}$ peut générer l'observation $\mathcal{O}$ notée $P(\mathcal{O} \mid \mathcal{H})$. La solution à ce problème est obtenue par l'*algorithme forward* ou par l'*algorithme forward-backward* [394]-[397].

ii. Problème de décodage : Etant donné une séquence d'observations $\mathcal{O} = \langle o_1, \ldots, o_T \rangle$ et une HMM $\mathcal{H} = \langle S, V, A, B, \Pi \rangle$, il s'agit de déterminer la séquence d'états cachés $\mathcal{Q} = \langle q_1, \ldots, q_T \rangle$ la plus probablement suivie lors de la génération de l'observation $\mathcal{O}$ par $\mathcal{H}$ (c'est-à-dire optimale selon le critère de vraisemblance de génération des observations par la HMM). La solution à ce problème est obtenue par l'*algorithme de Viterbi* [397]-[399].

iii. Problème d'apprentissage : Etant donné une séquence d'observations $\mathcal{O} = \langle o_1, \ldots, o_T \rangle$ et les dimensions d'une HMM (nombre d'états N et taille M de l'alphabet de symboles observables), il s'agit de déterminer le modèle $\mathcal{H} = \langle S, V, A, B, \Pi \rangle$ tel que sa probabilité de générer l'observation $\mathcal{O}$ soit maximale. La solution à ce problème peut être obtenue soit par l'*algorithme de Baum-Welch* [397], [400], [401] qui est une instanciation particulière de l'*algorithme d'espérance-maximisation* (expectation-maximization) adapté aux HMM, soit par l'algorithme dit de *segmental k-means* [402].

2.2.1.4 Application des HMM

Les HMM sont des outils d'apprentissage très efficaces s'ils disposent de suffisamment de données d'apprentissage. Ils sont largement utilisés [403] en particulier dans la reconnaissance de la parole ou de l'écriture manuscrite, ainsi que dans la bio-informatique.

Les modèles de Markov cachés sont également utilisés pour le développement de stratégies de maintenance prédictive ou pour l'estimation de la durée de vie résiduelle de systèmes [369], [404]-[409].

2.2.2 Méthodologie de pronostic de l'état de santé de batteries lithium-ion par modèle de Markov caché

2.2.2.1 Modélisation de l'état de santé par une HMM

On modélise par une HMM le processus de dégradation d'un système lorsqu'il n'est pas possible d'observer directement son état de santé (les états du système sont cachés), mais qu'on dispose d'observations sur celui-ci qui nous permettent de révéler ses états.

C'est la situation que nous rencontrons avec les batteries lithium-ion : l'état de santé global (SoH) d'une batterie ne peut pas être directement évalué, cependant nous pouvons collecter sur la batterie des indicateurs comme l'analyse incrémentale de la capacité (ICA), la tension différentielle et la résistance interne obtenues par les mesures de tension, courant, impédance, température, etc. [410].

Nous pouvons alors modéliser la dynamique de la dégradation de la batterie par une HMM :

— Les états correspondront à différents niveaux d'états de santé : chaque état caché correspondra à un intervalle du SoH (par exemple les différents états cachés sont associés à différents intervalles de valeurs du SoH : état 1 pour un SoH entre 90% et 100%, état 2 pour un SoH entre 80% et 90%, etc.).

— Les observations correspondront à un codage des indicateurs indirect du SoH (par exemple les symboles observables sont associés à différents intervalles de valeurs de l'ICA).

Les méthodes de résolution des trois problèmes fondamentaux associés aux HMM nous permettent de réaliser le pronostic de l'état de santé de batteries lithium-ion :

— Pour construire un modèle de Markov caché qui correspond aux différentes observations de batteries soumises aux mêmes conditions environnementales et opérationnelles (appelée *région* par la suite), le problème d'apprentissage est considéré. Cela nous permettra de construire pour chaque type de région un modèle de Markov caché qui lui est spécialisé.

— Lors de la réception d'une batterie dont on ne connait pas a priori sa région d'origine, le problème d'évaluation est considéré. On pourra sélectionner dans la base de données des modèles régionaux de HMM, le modèle de HMM qui a la plus grande vraisemblance avec la batterie reçue.

— Pour connaitre la durée de vie résiduelle (RUL) d'une batterie dont on a identifié avec la plus grande vraisemblance sa région d'origine, le problème de décodage est considéré. A partir des observations fournies par le BMS de la batterie, nous pourrons identifier l'état caché du HMM qui correspond à son état de santé puis estimer son RUL.

2.2.2.2 Méthodologie de pronostic de batteries lithium-ion basée sur les HMM

Le processus de dégradation des batteries lithium-ion des véhicules électriques est très dynamique et varie en fonction de leurs usages dans différentes régions géographiques en raison de variations environnementales et de conditions opérationnelles. Pour prendre en compte ce dynamisme et cette hétérogénéité d'usages, nous proposons une méthode de pronostic de l'état de santé des batteries lithium-ion basée sur un modèle de Markov caché.

Une étape préalable est le choix du modèle de Markov caché qui sera utilisé : choix du nombre d'états cachés, choix du nombre de symboles observables, choix de la topologie du HMM. Pour cela, il est nécessaire d'identifier les paramètres qui doivent être surveillés dans les batteries. Dans les batteries Lithium-ion, l'état de santé se dépréciant avec la perte de capacité ou l'augmentation de la résistance interne, il sera utile de choisir parmi ces indicateurs au lieu de sélectionner des paramètres dont l'état de santé est indépendant. Dans notre étude, nous avons choisi d'utiliser l'indicateur d'analyse incrémentale de capacité (ICA) qui est un outil efficace pour déterminer l'état de santé (SoH) des cellules lithium-ion [389], [390].

La méthodologie de pronostic de la durée de vie résiduelle des batteries lithium-ion est décomposée en trois étapes :

1. A partir d'un ensemble de batteries reçues dont les origines régionales sont connues, en utilisant les mesures fournies par leurs *Battery Management System* et par des tests en laboratoire, on construit à l'aide de l'algorithme de Baum-Welch une base de données de modèles régionaux de HMM. A chaque région i est associé un modèle de Markov caché $\mathcal{H}_i$ du processus de dégradation des batteries lithium-ion utilisées dans cette région.
2. A la réception d'une batterie dont on ne connait pas sa provenance (sa région d'utilisation), on sélectionne tout d'abord le modèle de HMM régional $\mathcal{H}_{rel}$ qui correspond le plus précisément à la batterie en utilisant l'*algorithme forward* ou l'*algorithme*

forward-backward [394]-[397]. Puis on identifie l'état de santé actuel (SoH) de la batterie en utilisant l'algorithme de Viterbi [397], [399].

3. On réalise le pronostic de la durée de vie utile restante (RUL), à l'aide du modèle $\mathcal{H}_{rel}$ sélectionné à l'étape 2 et de la connaissance de son SoH.

1ère étape : Création de la base de données des modèles HMM

La dégradation des batteries lithium-ion varie différemment selon les régions géographiques, un seul modèle de Markov caché ne suffit pas à modéliser l'ensemble des dynamiques de dégradation. Il est alors judicieux de créer une base de données de modèles, où chaque modèle représente une région géographique choisie. Nous supposons avoir X région géographique distincte. Dans un premier temps, nous définissons X modèles de Markov cachés $(\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_X)$ correspondant aux régions $R_1, R_2, \ldots, R_X$.

Le modèle de Markov caché $\mathcal{H}_i$ peut être exprimé par le tuple $\langle S, V, A_i, B_i, \Pi_i \rangle$. L'ensemble d'états S et l'ensemble de symboles d'observation distincts V sont communs à tous les HMM. La matrice de transition d'état A_i, la matrice de symboles d'observation B_i et le vecteur de probabilité initial Π_i sont spécialisés à chaque région R_i. Les valeurs initiales de A_i, B_i et Π_i sont calculées à partir d'une partie des données d'entrainement associées à la région.

Les modèles de chaque région sont mis à jour à partir des ensembles de données d'apprentissage des batteries appartenant à la région concernée en utilisant la méthode de Baum-Welch.

La figure IX.6 illustre la construction de la base de données de modèles de Markov cachés régionaux.

2ème étape : Sélection du meilleur modèle HMM et diagnostic du SoH

Le processus de reconditionnement d'une batterie lithium-ion d'un véhicule électrique nécessite de connaitre son état de santé (SoH) actuel afin de calculer sa durée de vie résiduelle (RUL). A partir de ces informations, les cellules de la batterie peuvent être reconditionnées dans un nouveau pack de batterie prêt pour d'autres usages, moins exigeants en performance qu'un véhicule électrique.

Etant donné une séquence d'observation O issue d'une batterie à reconditionner, nous identifions le modèle de Markov caché le plus fiable parmi la base de données de modèles créée dans la première étape. Le processus d'identification du modèle est basé sur le calcul pour chaque région R_i de la probabilité que le modèle $\mathcal{H}_i$ ait généré la séquence O $(P(O \mid \mathcal{H}_i))$. L'*algorithme forward* est utilisé pour calculer la probabilité $P(O \mid \mathcal{H}_i)$. Nous définissons la région R_{rel}, comme la région qui nous donne le modèle le plus fiable associé à la séquence d'observation. R_{rel} satisfait la propriété suivante : $P(O \mid \mathcal{H}_i \leq P(O \mid \mathcal{H}_{rel})$, $\forall i \in [1, X]$.

On associe à la région sélectionnée R_{rel}, sa valeur de confiance C, définie comme $C = P(O \mid \mathcal{H}_{rel})$.

Le modèle choisi $\mathcal{H}_{rel}$ est utilisé pour identifier le SoH de la batterie par l'*algorithme*

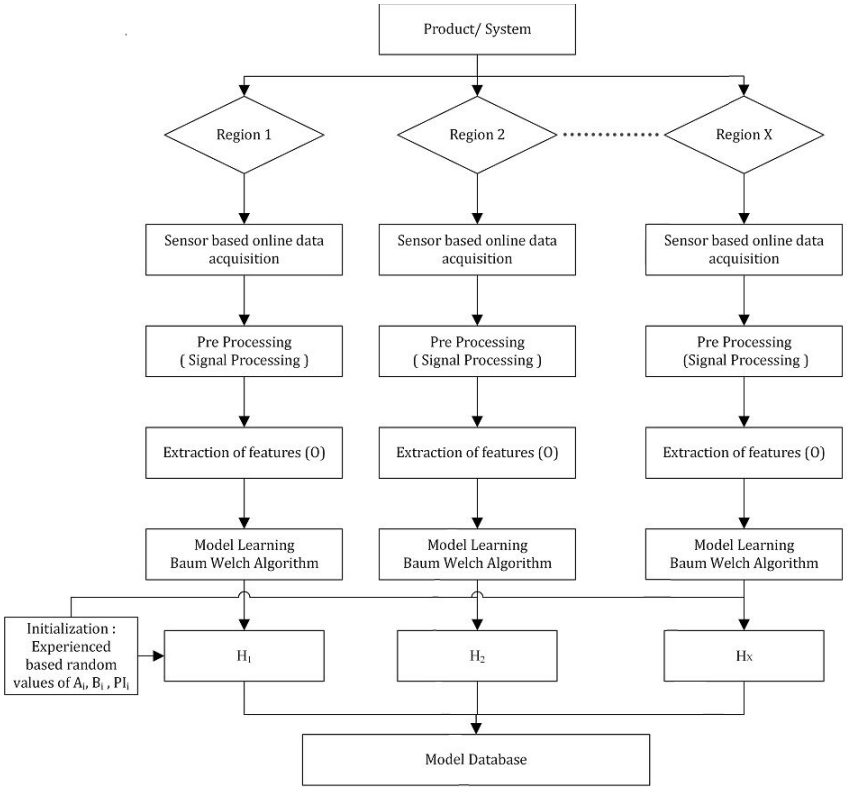


FIGURE IX.6 – Etape d'apprentissage des modèles de Markov cachés [382], [383]

de Viterbi : à partir des observations de la batterie, nous pouvons estimer l'état caché représentant l'état de santé de la batterie.

La figure IX.7 illustre le processus d'identification du modèle $\mathcal{H}_{rel}$ et du diagnostic du SoH.

3ème étape : Pronostic du RUL

Pour proposer une aide à la décision sur le reconditionnement des cellules d'un pack de batterie, il est important de connaitre l'état de santé futur de la batterie. Une étape de pronostic de la durée de vie résiduelle (RUL) de la batterie doit être réalisée.

Le principe de calcul du RUL est d'utiliser l'état actuel de la batterie qui a été identifié, l'état final (l'état de défaillance) et la matrice de probabilités de transition entre états A_{rel} du modèle de Markov caché sélectionné $\mathcal{H}_{rel}$ pour calculer le chemin critique qui part de l'état actuel à l'état final. Les transitions potentielles sont identifiées par toutes les probabilités non nulles dans la matrice A_{rel}. En additionnant la durée de séjour de chaque état du chemin critique, la durée de vie résiduelle est calculée. Les références [355], [411], [412] présentent différentes méthodes de calcul du RUL dans les modèles de Markov cachés.

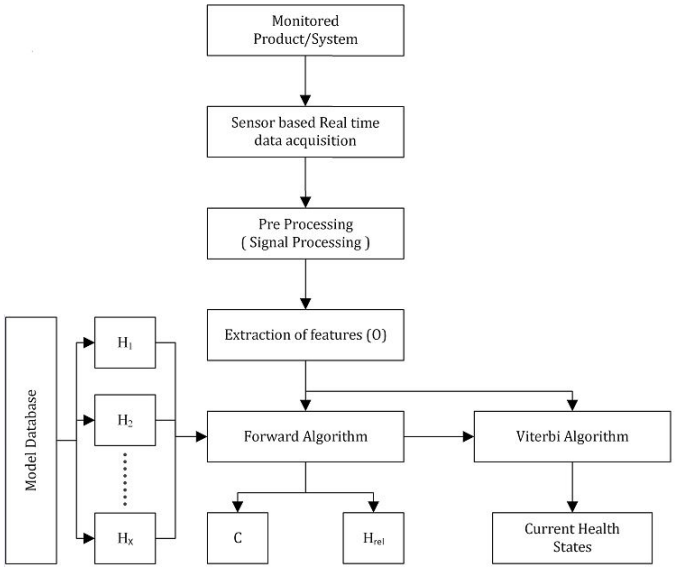


FIGURE IX.7 – Etape d'identification du modèle de Markov caché et du diagnostic du SoH [382], [383]

2.3 Pronostic par combinaison de modèle auto-regressif et régression à vecteur de support

2.3.1 Présentation des outils utilisés

2.3.1.1 Analyse incrémentale de la capacité, capacité régionale

L'analyse incrémentale de la capacité (ICA) consiste à calculer la dérivée de la capacité par rapport à la tension sur un petit intervalle de tension. Puisque l'état de santé (SoH) est défini à partir de la capacité, l'analyse incrémentale de la capacité permet une estimation du SoH. Cependant, les méthodes ICA ne sont pas parfaites, leurs implémentations reposent fortement sur un courant constant stable et à faible bruit de mesure.

Tang et al. [413] définissent l'indicateur de *capacité régionale* (regional capacity) extraite de l'analyse incrémentale de capacité qui a une sensibilité au bruit fortement réduite. Cette capacité régionale peut être corrélée au SoH d'une batterie à l'aide d'algorithmes probabilistes ou non probabilistes.

L'adjectif *régionale* n'a ici pas de lien avec la région d'usage de la batterie, les auteurs utilisent le terme de *regional capacity* pour signifier que son calcul est réalisé au niveau du pic de la courbe de capacité incrémentielle (IC curve).

Dans [413], les auteurs présentent une procédure pour calculer la capacité régionale $\hat{C}_k$ pour chaque cycle k de charge / décharge de la batterie. L'état de santé d'une batterie peut être modélisé à partir des valeurs de sa capacité régionale pour chaque cycle de charge / décharge [413].

Nous utiliserons cet indicateur de capacité régionale pour l'apprentissage de nos modèles de prédiction de l'état de santé des batteries.

2.3.1.2 Régression à vecteur de support

La régression à vecteurs de support (SVR) est une généralisation de la méthode de classification de machine à vecteur de support (SVM). Le modèle SVR prédit une sortie à valeur continue [414], [415].

SVR est un algorithme très efficace pour décrire la relation non linéaire entre des entrées x_i et des sorties y_i. Il est ainsi parfaitement adapté à l'estimation du SOH à partir de la capacité régionale.

La régression à vecteur de support consiste à trouver la fonction $f(x)$ la plus « lisse » qui passe par les (ou à proximité des) données d'apprentissage, c'est-à-dire qui a au plus une déviation ε par rapport aux exemples d'apprentissage (x_i, y_i), pour $i \in [1, N]$ (ne pas considérer les erreurs inférieures à ε et à interdire celles supérieures à ε).

2.3.1.3 Modèle de prédiction de série temporelle ARIMA

Un modèle ARIMA (Auto Regressive Integrated Moving Average) est un modèle de série temporelle qui exprime les valeurs futures d'une série chronologique sous la forme d'une combinaison linéaire de ses valeurs passées. Le modèle ARIMA est une combinaison de trois processus : un processus autorégressif (AR), une différenciation pour rendre les données ou les séries stationnaires et un processus de moyenne mobile (MA).

Un modèle ARIMA est étiqueté comme modèle $ARIMA(p, d, q)$ dans lequel p est le nombre de termes auto-régressifs, d est le nombre de différences et q est le nombre de moyennes mobiles.

Box et Jenkins [416], [417] ont développé une méthodologie qui permet de construire le modèle $ARIMA(p, d, q)$ restituant le mieux possible le comportement d'une série temporelle selon une procédure en trois étapes : identification, estimation, diagnostic :

- L'identification permet le choix des ordres p, d et q. L'observation de la représentation graphique de la série temporelle permet une analyse de sa tendance, sa saisonnalité. Si la série n'est pas stationnaire, un processus de différenciation est appliqué pour rendre la série temporelle stationnaire. L'analyse de l'autocorrélogramme (ACF) et de l'autocorrélogramme partielle (PACF) permet de déterminer les ordres du modèle.
- Plusieurs techniques sont possibles pour l'estimation des paramètres p, q et d : la méthode du maximum de vraisemblance, la méthode par moindres carrés ou la méthode des moments.
- Le diagnostic est réalisé par un examen des paramètres par divers tests statistiques tels que les tests de significativité des paramètres et le test sur le bruit blanc.

La figure IX.8 schématise la méthodologie de Box et Jenkins pour construire un modèle ARIMA.

Lorsque la série temporelle des capacités régionales est représentée par un modèle ARIMA (à l'aide de la méthodologie de Box et Jenkins), il peut être facilement utilisé pour la prédiction.

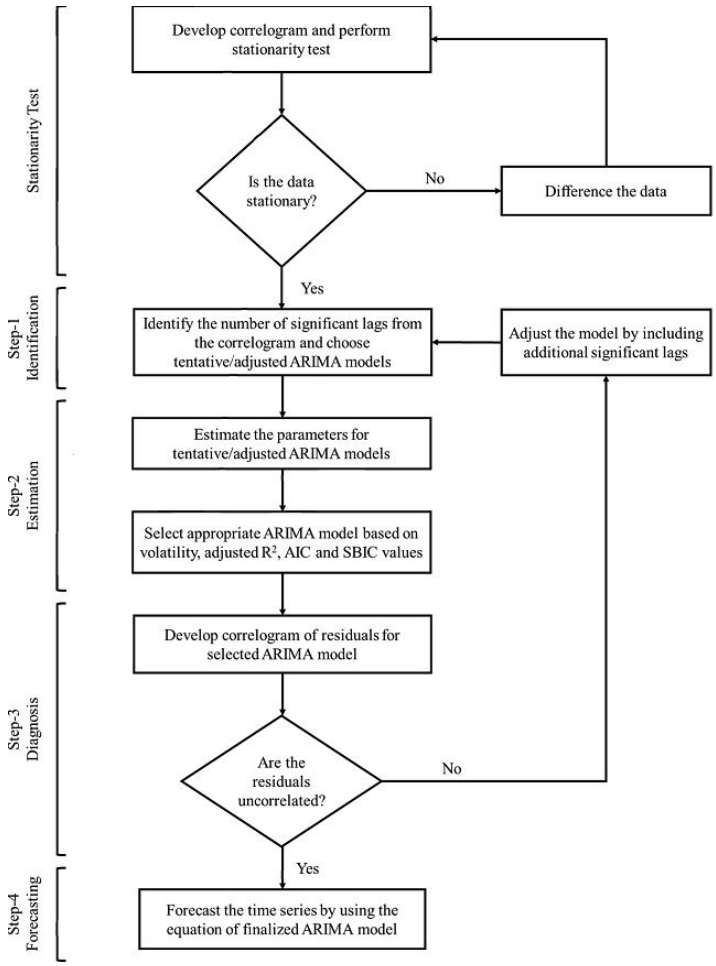


FIGURE IX.8 – Organigramme de fonctionnement de la méthode de Box et Jenkins [418]

Nous utiliserons ainsi un modèle de prédiction de série temporelle ARIMA pour prédire les valeurs de la capacité régionale pour de futurs cycles de charge / décharge de la batterie.

2.3.2 Méthodologie de pronostic de l'état de santé de batterie lithium-ion par combinaison de modèle auto-regressif et régression à vecteur de support

Nous supposons disposer d'un ensemble de batteries lithium-ion d'origine géographique connue et dont chaque cycle de charge / décharge est étiqueté par le SoH de la batterie. Nous proposons une méthodologie de pronostic de l'état de santé des batteries lithium-ion à partir de leurs mesures de capacité régionale, celle-ci est constituée de trois étapes décrites ci-dessous.

1ère étape : Création de la base de données des modèles SVR

Pour chaque batterie provenant d'une région R_k dont on connait l'état de santé, en utilisant les données issues de son BMS, nous construisons une série chronologique $(\hat{C}_i^k, SoH_i^k)$

exprimant pour chaque cycle i la valeur de la capacité régionale [413] et le SoH de la batterie.

A partir des séries chronologiques $(\hat{C}_i^k, SoH_i^k)$ des batteries issues d'une même région géographique R_k, un modèle SVR M_k est défini par apprentissage supervisé.

La figure IX.9 schématise la démarche de construction de cette base de données de modèles SVR.

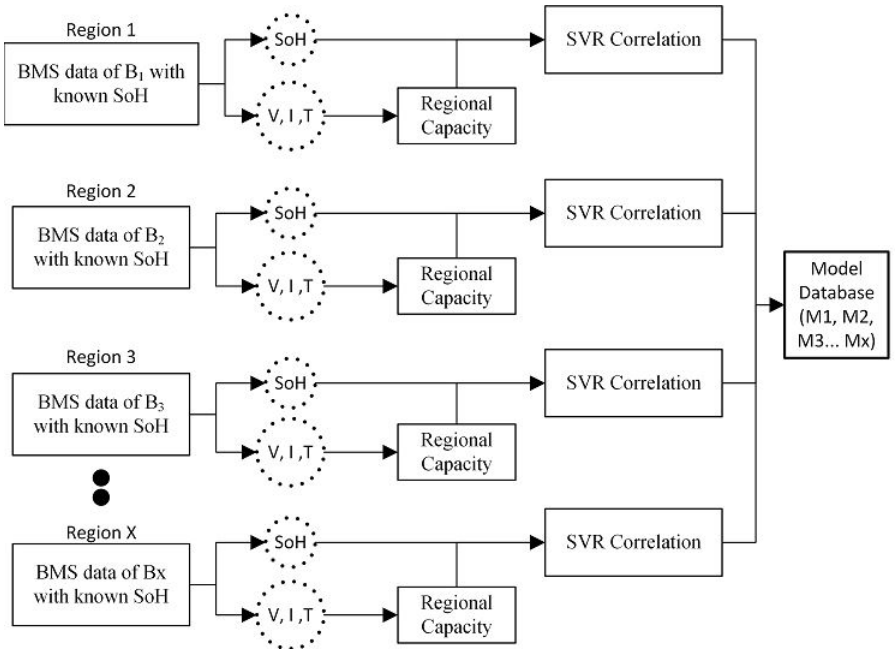


FIGURE IX.9 – Création de la base de données de modèles SVR [385]

2ème étape : Sélection du meilleur modèle SVR

A la réception d'une batterie dont on ne connait ni l'origine géographique ni l'état de santé, en utilisant les mesures fournies par son *Battery Management System* on réalise pour chaque cycle de charge / décharge l'analyse incrémentale de la capacité et le calcul de la capacité régionale. Ces données sont ensuite modélisées sous la forme d'une série temporelle.

La série temporelle de la batterie à tester est ensuite comparée aux séries temporelles qui ont utilisé lors de l'étape de construction des modèles SVR. La comparaison est réalisée par le critère RMSE (Root Mean Square Error) :

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}} \tag{3}$$

avec x_i la capacité régionale au cycle i de la batterie à tester et $\hat{x}_i$ la capacité régionale au cycle i associée à une région particulière.

On sélectionne le modèle SVR M_{best} associé à la région dont la série temporelle de capacité est la plus « proche » selon le critère RMSE de la batterie à tester.

La figure IX.10 illustre le processus de sélection du meilleur modèle SVR.

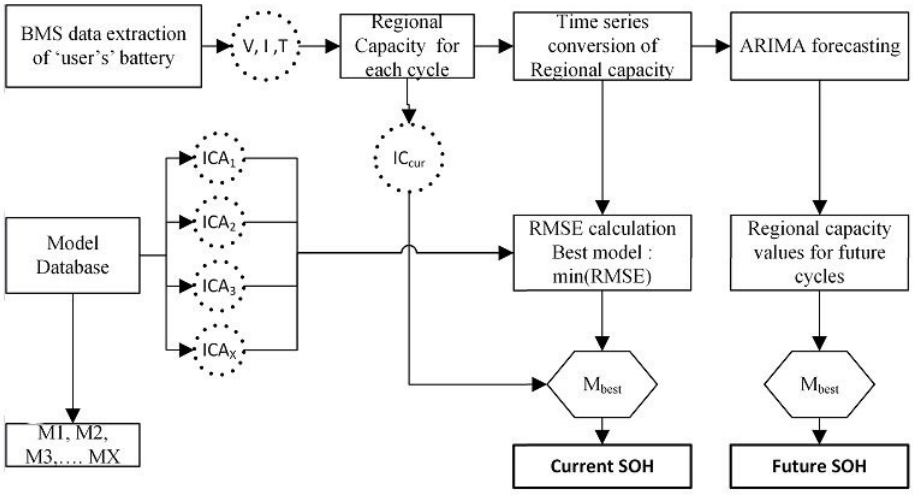


FIGURE IX.10 – Identification du modèle meilleur modèle SVR [385]

3ème étape : Diagnostic et pronostic

En utilisant les mesures du dernier cycle de charge / décharge contenues dans le BMS de la batterie à tester, nous calculons sa capacité régionale actuelle IC_{cur}. Le modèle SVR M_{best} obtenu à l'étape précédente permet par corrélation de connaitre l'état de santé actuel de la batterie SoH_{cur}.

La série temporelle de la capacité régionale de la batterie est ensuite modélisée en un modèle ARIMA par la méthode de Box et Jenkins. Des modèles ARIMA et SVR M_{best} les futures capacités régionales et SoHs peuvent être prédites.

2.4 Cas d'études

Nous avons validé nos deux méthodologies de pronostic de l'état de santé de batteries lithium-ion en utilisant les données fournies par le « Prognostic Center of Excellence » de la NASA [419].

Pour plus de détails sur la mise en œuvre de nos méthodologies de pronostic, le lecteur peut se référer au manuscrit de thèse de Akash Basia [382] et à nos publications conjointes [383]-[387].

3 Conclusion et perspectives sur le pronostic de l'état de santé de systèmes complexes

Nous avons étudié dans ce chapitre la problématique de la prédiction de l'état de santé de systèmes complexes. Nos travaux de recherche se sont axés sur le développement de méthodes à base de données à la différence de méthodes basées sur la physique des défaillances (PoF) ou de méthodes hybrides combinant des approches PoF avec des approches basées sur les données.

Ce travail de recherche a été réalisé pendant l'encadrement de la thèse de A. Basia, le sujet de sa thèse était l'évaluation de santé des batteries lithium-ion dans le contexte de l'économie circulaire

Les principales contributions de ce travail de recherche ont été le développement de deux méthodologies de pronostic de l'état de santé de batteries en utilisant des méthodes basées sur les données telles que les modèles de Markov cachés, la régression à vecteur de support et les modèles ARIMA. Les méthodes que nous avons développées permettent de prendre en compte l'hétérogénéité technologique des batteries lithium-ion et la pluralité de leurs conditions d'usages qui accentuent la difficulté d'établir un pronostic sur leur état de santé et leur durée de vie résiduelle. Nous avons testé et validé nos méthodes de pronostic du SoH et du RUL avec un cas d'utilisation des données de batteries en open source fournies par la NASA.

Pour considérer la diversité de conditions opérationnelles dans lesquelles une batterie lithium-ion peut être utilisée, nous avons considéré leur usage dans différentes régions géographiques, en effet les conditions environnementales influencent la dégradation des batteries. Nous avons donc discriminé les différentes conditions opérationnelles des batteries lithium-ion sur un seul critère, la région d'usage. Il serait intéressant d'étendre nos travaux de recherche pour prendre en compte plusieurs critères de discrimination des conditions opérationnelles. Par exemple en considérant l'hétérogénéité technique que l'on peut rencontrer dans les packs de batteries : les packs de batteries lithium-ion sont fabriqués par différentes entreprises qui utilisent différents types de composants lors de la fabrication, ces composants ont des processus de dégradation qui diffèrent entre eux. Le fabricant du pack de batterie lithium-ion pourrait alors être utilisé comme second critère.

Une première perspective de ce travail est d'appliquer nos méthodes de pronostic sur un cas réel afin de prendre en compte les problématiques de données insuffisantes, de données incomplètes.

L'usage des méthodes de pronostic basées sur les données telles que les outils d'apprentissage automatique issus de l'intelligence artificielle nous ont montré la difficulté à calibrer / paramétrer ces outils afin d'identifier le « meilleur » modèle de prédiction. Une seconde perspective de nos travaux est ainsi le développement d'une méthodologie d'usage de critères et d'indicateurs de performances pour proposer une analyse comparative entre différents paramétrages d'outils d'apprentissage automatique.

Une troisième perspective au développement de méthodes de pronostic du SoH et du RUL est de prendre en compte les opérations de maintenance en considérant la politique

de maintenance déployée comme la politique de réparation parfaite (AGAN : As Good As New) ou minimale (ABAO : As Bad As Old).

Comme dernières perspectives à nos travaux sur les méthodes de pronostic guidées par les données, nous étudierons de nouveaux outils / méthodes de l'IA tels que les réseaux neuronaux et leurs variantes afin d'identifier dans quel cas d'usage un outil issu de l'IA est le plus adapté. Nous avons initié cette analyse dans la publication [386] en comparant la régression linéaire, la régression à vecteur de support et le réseau de neurones à propagation avant (feed forward neural network) pour l'estimation de l'état de santé des batteries lithium-ion.

Perspectives de recherche

Chapitre X

Projets et ambitions de recherche

Les travaux de recherche que j'ai menés et co-encadrés par des thèses et des stages de Master 2 sur le diagnostic, le pronostic et l'aide à la décision pour la maintenance ont apporté des résultats satisfaisants, de nouveaux outils et méthodes sur ces thématiques ont été développés et validés. Cependant, comme il a été relaté dans la conclusion et les perspectives de chaque chapitre / thème de recherche, de nombreuses pistes de recherche restent à explorer.

Ce chapitre présente une vision globale de mes intérêts de recherche en proposant des projets et ambitions de recherche sur lequels je souhaite travailler dans les années futures. Dans une première partie je vais tout d'abord présenter plusieurs projets de recherche qui correspondent aux thématiques de recherche que je souhaite poursuivre et renforcer : mener une activité de recherche « théorique » validée par des cas d'études faisant intervenir différents domaines scientifiques et technologiques et participant à une transition réussie vers l'Industrie 4.0. Dans une seconde partie, je vais présenter mes ambitions de recherche pour répondre à l'un des défis majeurs de la révolution industrielle 4.0, la prise en compte de l'humain dans le développement des nouvelles méthodes et technologies pour la fiabilité, le diagnostic et la maintenance des usines intelligentes ou « smart factory ». L'industrie du futur fait usage de robotique autonome et collaborative (cobots) et autres systèmes cyber-physiques qui impliquent l'humain dans leur mise en œuvre, il faut donc proposer de nouveaux outils et méthodes pour l'étude de la sûreté de fonctionnement et du pronostic et management de la santé des usines intelligentes qui prennent en compte la collaboration entre l'humain et ces systèmes « intelligents ». Il faut imaginer les solutions pour intégrer l'humain dans cette « disruption digitale » pour garantir les critères de performance, de production, mais également sécuritaires homme-machine et impliquer l'humain au cœur des dynamiques de l'Industrie 4.0.

Ces projets et ambitions de recherche permettront d'énoncer plusieurs sujets de thèses pour former de nouveaux jeunes chercheurs à la sûreté de fonctionnement et au pronostic et management de la santé dans la révolution industrielle 4.0.

1 Projets de recherche

1.1 Diagnostic de systèmes dynamiques complexes à base de modèles incomplets

Ce projet de recherche est issu des résultats des travaux de recherche que j'ai développé dans l'axe A) des *méthodes de diagnostic basées sur des modèles* et dans l'axe B) des *outils pour le diagnostic et la maintenance*.

Un verrou scientifique majeur pour le diagnostic à base de modèles d'un système industriel complexe est le manque de connaissance sur l'évolution de son comportement en présence de fautes multiples.

Avec la connaissance d'experts, sa dynamique dans le mode fonctionnement nominal (sans faute) peut être décrite dans plusieurs formalismes (équations différentielles, bond graphs, automates hybrides), cependant la modélisation du comportement du système industriel avec la prise en compte de défauts multiples est difficile, voire impossible, par manque de connaissance sur les effets conjoints des défaillances.

En effet, l'usage des diagrammes FAST (Function Analysis System Technique) et de la méthodologie AMDEC (Analyse des Modes de Défaillance, de leurs Effets et de leurs Criticités) ne permet pas toujours de connaitre les conséquences des interactions possibles entre plusieurs composants défaillants.

La contribution de ce projet de recherche est de développer une méthodologie de diagnostic en présence de fautes multiples lorsqu'on ne possède qu'une modélisation du comportement nominal du système complexe.

Nos intentions pour mener à bien ce projet sont d'une part de développer une méthode d'analyse de la fiabilité de systèmes dynamiques complexes afin d'identifier quelles sont les combinaisons de fautes qui mènent à des défaillances redoutées et sont le plus susceptibles de se produire et, d'autre part, de développer une méthode de détection et de localisation de fautes multiples par une approche mixte : on utilisera tout d'abord des automates temporisés pour détecter les défaillances puis des arbres de défaillances dynamiques pour localiser les fautes.

Les objectifs principaux de ce projet sont les suivants :

1. Développer une méthodologie de modélisation hiérarchique et multi-vues de systèmes dynamiques complexes exprimant les dépendances entre les composants et permettant de construire / déduire sous la forme d'arbres de défaillances dynamiques les enchaînements de défaillances menant aux modes de fonctionnement défaillant du système. Cette représentation permettra de pallier l'absence de modèles avec fautes du système.
2. Développer une méthodologie d'analyse qualitative et quantitative d'arbres de défaillances dynamiques afin d'identifier les séquences de coupes et estimer leurs probabilités. Ces résultats permettront d'identifier quelles sont les combinaisons de fautes qui mènent à des défaillances redoutées et sont le plus susceptibles de se produire.
3. Développer une approche hybride basée sur les automates temporisés et les arbres

de défaillances dynamiques pour le diagnostic de systèmes dynamiques avec une connaissance incomplète des modes de défaillance. On utilisera tout d'abord des automates temporisés pour détecter les défaillances puis des arbres de défaillances dynamiques pour localiser les composants défaillants. La modélisation sans défaut du système dynamique permettra de définir des automates temporisés de surveillance pour la détection de défauts. Les arbres de défaillances dynamiques, déduits de la modélisation hiérarchique et multi-vues de systèmes dynamiques (objectif 1 ci-dessus) ou construits par des experts et exprimant les liens de cause à effet des différents défauts de composants, permettront d'identifier si le système se trouve dans un mode de fonctionnement défaillant, de localiser les composants défaillants puis d'identifier les défauts.

4. Si les principaux objectifs de ce projet de recherche sont de nature méthodologique, il sera important d'évaluer leur efficacité sur des cas d'usage réels de systèmes de production. On pourra s'appuyer sur les moyens disponibles de la plateforme CIM (Computer Integrated Manufacturing) que propose le pôle S.mart Grenoble Alpes, c'est une plateforme technologique dédiée à l'industrie 4.0 ou développer une coopération industrielle avec un fleuron de la recherche industrielle grenobloise tel que STMicroelectronics ou Schneider Electric.

1.2 Optimisation de la maintenance distribuée par une approche guidée par les données

Ce projet de recherche est la poursuite des travaux initiés dans [326]-[328] et vise à proposer des outils pour l'optimisation de la maintenance d'une entreprise possédant plusieurs sites de production dans un contexte distribué. La maintenance distribuée consiste à ce que les activités de maintenance soient réalisées par deux structures : une première structure réalise le processus de réparation, c'est l'atelier de maintenance central, et une seconde structure effectue des inspections et des remplacements d'équipements sur différents sites de production, c'est l'atelier de maintenance mobile.

Le cadre général du projet est l'analyse et l'exploitation des données (management du Big Data) générées par des capteurs sur les équipements de production ou produites par les opérateurs de maintenance et l'optimisation de la maintenance des équipements de production. Ce projet de recherche étudie ce verrou scientifique dans le contexte de la maintenance distribuée.

Pour accompagner les entreprises de production dans la transition vers l'économie verte, il est nécessaire d'améliorer l'efficacité environnementale et énergétique de ces entreprises. Un levier de transition écologique est la maintenance prédictive. La maintenance prédictive en détectant les premiers signes de dégradation d'un équipement, permet aux opérateurs de maintenance de le remettre en état à l'aide de réparations souvent minimes et ainsi d'éviter son vieillissement prématuré et a des coûts moindres que la maintenance corrective.

Les verrous scientifiques que nous chercherons à lever dans ce projet de recherche sont d'une part la bonne estimation de la dégradation des équipements de production et de

leur évolution afin de pouvoir estimer les dates efficientes de leur maintenance prédictive et, d'autre part, de prendre en compte les demandes aléatoires de maintenance corrective dans un contexte de maintenance distribué (atelier de maintenance central et atelier de maintenance mobile).

La contribution de ce projet de recherche est de proposer une méthodologie de planification de la maintenance prédictive pour une entreprise ayant plusieurs sites de production et dont les activités de maintenance sont distribuées en combinant des approches basées sur les données (apprentissage automatique, intelligence artificielle) avec des approches d'optimisation.

Nos intentions pour réaliser ce projet de recherche sont les suivants :

1. Développement de capteurs virtuels pour la maintenance prédictive : l'utilisation des capteurs virtuels est en plein essor, car ils fournissent des alternatives réalisables et économiques à des capteurs physiques. Un capteur virtuel utilise les données générées par d'autres instruments de mesure pour fournir des propriétés sur un composant à l'aide d'algorithmes d'intelligence artificielle et de modèles mathématiques. Le capteur virtuel permet également de compléter ou de remplacer, en cas de défaillance, les informations mesurées par des capteurs physiques.

2. Combiner des méthodes d'apprentissage automatique qui proposent une planification de la maintenance prédictive des équipements avec des méthodes d'optimisation pour planifier le meilleur routage de l'atelier de maintenance mobile tout en minimisant le coût de maintenance. La problématique est de trouver les bons seuils pour décider quand la maintenance prédictive sur un équipement est à faire en minimisant les coûts induits par la mise en œuvre de cette maintenance dans le cas où celle-ci se fait de manière distribuée. Nous souhaitons explorer sur ce thème l'usage de la simulation à base d'agents (*agent-based simulation*) combinée avec l'apprentissage par renforcement (*reinforcement learning*).

3. Développer des algorithmes de routage dynamique ou adaptatif de l'atelier de maintenance mobile pour prendre en compte des pannes d'équipement dans des sites de production. L'objectif est de prendre en compte des demandes fluctuantes dans le temps de maintenance corrective pendant la tournée de l'atelier de maintenance mobile dont la finalité principale est la mise en œuvre de la maintenance préventive des équipements de production. Il faut réorganiser *à la volée* le planning de visite de l'atelier de maintenance mobile pour d'une part réaliser la maintenance corrective d'un site de production tout en respectant les contraintes de plages temporelles de maintenance préventive sur les autres sites de production.

4. Evaluer nos méthodologies et algorithmes pour la maintenance préventive avec un partenaire industriel ayant plusieurs sites de production et un atelier de maintenance centralisé.

1.3 Aide à la décision pour les systèmes cyber-physiques de production

Avec la révolution industrielle 4.0, l'utilisation des systèmes cyber-physiques dans la production industrielle a introduit le concept de *système cyber-physique de production* (CPPS). Olivier Cardin [420] complète la définition des CPPS proposée par Monostori [152] ainsi : « Un CPPS consiste en un système de systèmes d'éléments coopérants et autonomes, connectés au travers d'une relation contextualisée, au sein et au travers de tous les niveaux de la production, du process aux réseaux logistiques, améliorant les processus de prise de décision en temps réel, la réponse à des conditions imprévues et leur évolution au cours du temps. »

Les CPPS possèdent une capacité d'autoadaptabilité et apportent de l'agilité aux systèmes de production : ses équipements cyber-physiques opèrent sur un réseau *Plug & Play* et offrent la fonctionnalité *Plug & Produce*.

Un verrou scientifique majeur dans l'accompagnement de la transformation digitale l'industrie est le manque d'outils d'aide à la décision pour la surveillance des systèmes cyber-physiques de production pour l'optimisation du diagnostic, du pronostic et de la maintenance.

Nos intentions dans ce projet de recherche sont d'une part de proposer des outils d'analyse de la testabilité et de la diagnosticabilité des CPPS dans le but de fournir des informations pertinentes d'aide à la décision sur le placement de capteurs physiques ou virtuels pour optimiser la surveillance des équipements cyber-physiques et, d'autre part, de proposer des méthodologies de diagnostic et de pronostic des équipements cyber-physiques de production basées sur des outils d'apprentissage automatique de l'intelligence artificielle.

Les objectifs principaux de ce projet de recherche sont les suivants :

1. Etude de la testabilité des systèmes cyber-physiques de production pour l'optimisation de leur maintenance :
   Les exigences de sûreté de fonctionnement et la complexité croissante des systèmes de production nécessitent des techniques de test et de diagnostic de plus en plus élaborées. Ce besoin devient nécessaire dans le cas de systèmes cyber-physiques de production comportant un grand nombre de composants hétérogènes issus de différents domaines tels que l'électronique, la mécanique, le réseau (Internet of Thing par exemple) et le logiciel. Chaque domaine a ses propres méthodes de test que l'on trouve largement dans la littérature. Cet objectif de recherche concerne la mise en œuvre d'une approche efficiente de test qui combine les différentes méthodes de façon à se situer à l'échelle du système cyber-physique. Ces développements permettront de définir de façon formelle les exigences de testabilité afin de minimiser les coûts de maintenance. La finalité de ce travail est l'optimisation de l'activité de maintenance qui repose sur la prise en compte de l'aptitude à diagnostiquer un système et suivre les éventuelles dérives en exploitation. La feuille de route de cet objectif de recherche est tout d'abord la définition de métriques de testabilité pour chacun des domaines. Ensuite, dans un second temps, il s'agira de modéliser les interactions entre les différents domaines ainsi que les effets de propagation pour proposer une approche de

testabilité au niveau système. Finalement, dans une dernière étape, le développement d'une méthode d'ordonnancement des tests sera développé. Ce travail de recherche sera ensuite poursuivi par l'étude de la diagnosticabilité de systèmes cyber-physiques en utilisant les résultats de l'analyse de la testabilité pour identifier les points d'observations supplémentaires qui sont nécessaires pour garantir la propriété de diagnosticabilité locale au niveau des équipements de production et également globale du système cyber-physique de production.

2. Intégration d'outils d'apprentissage automatique de l'intelligence artificielle dans les méthodologies de diagnostic et de pronostic de systèmes cyber-physiques de production :
   Les équipements cyber-physiques de production sont des systèmes dynamiques complexes dont la formalisation du comportement dynamique en présence de défauts ou de processus de dégradation est très difficile à obtenir. Notre objectif de recherche est de proposer des méthodologies d'apprentissage automatique du comportement d'un CPS en présence de défauts ou de processus de dégradation afin de fournir des modèles « boite noire » de ces systèmes cyber-physiques. Nous utiliserons des approches basées sur les données pour les construire en exploitant les données issues des capteurs physiques et virtuels disséminés sur les équipements du système de production. Nous proposerons ensuite aux opérateurs de maintenance une méthodologie de diagnostic de ces systèmes complexes qui utilisera nos modèles « boite noire » comme observateurs. Nous utiliserons des méthodes de diagnostic basées sur l'usage des automates temporisés, des matrices de signature des défaillances et des arbres de défaillances dynamiques. Notre ambition dans cet objectif de recherche est d'étendre nos approches de simulation fonctionnelle d'un système à la simulation par jumeau numérique (*Digital Twin*).
3. Mise en œuvre de nos travaux sur un système cyber-physique de production par une coopération avec un acteur industriel ou en s'appuyant sur la plateforme CIM (Computer Integrated Manufacturing) du pôle S.mart Grenoble Alpes qui propose un atelier de production flexible et de la plateforme Operations management commune au laboratoire G-SCOP et à l'école de Génie Industriel de Grenoble INP qui propose d'émuler une chaîne logistique réelle. Ces plateformes disposent de cobots, de robots autonomes de logistiques, de postes de travail connectés, d'ateliers flexibles, de convoyage, de traitement et de distribution de palettes et de réseaux industriels.

2 Ambitions de recherche

Avec l'essor de l'industrie du futur, le rôle de l'opérateur humain dans l'industrie de production (chaînes logistiques, industrie manufacturière) évolue très rapidement. Des tâches répétitives et dangereuses réalisées auparavant par des humains sont maintenant automatisées et confiées à des robots. Il convient alors de concevoir de nouveaux métiers dans les usines intelligentes pour utiliser les aptitudes humaines au mieux dans des tâches requérant

leurs intelligences pour traiter des informations stratégiques et demandant une prise de décision et de prendre en considération que les opérateurs humains sont amenés à interagir avec des équipements issus de la robotique ou de la cobotique (robots autonomes, cobots, exosquelettes).

Un verrou scientifique majeur dans cette révolution 4.0 est de faire évoluer les méthodes et outils existants de la sûreté de fonctionnement et du pronostic et management de la santé (PHM) pour prendre en compte cette nouvelle place de l'humain dans l'usine intelligente.

Plusieurs questions clés se posent sur la problématique de la prise en compte de l'humain dans la sûreté de fonctionnement et le PHM de l'Industrie du futur, en particulier : Comment répartir la prise de décision entre la machine et l'humain au regard de la problématique de l'interprétation d'un très grand volume de données (Big Data) ? Dans le cadre d'un travail collaboratif humain-robot, lors d'une dérive de performance ou d'une défaillance, comment intégrer l'opérateur humain dans le processus de diagnostic ?

Notre ambition est d'étudier ces questions clés au travers de deux projets de recherche.

2.1 Utilisation de la réalité virtuelle /réalité augmentée pour le diagnostic et l'aide à la maintenance d'équipement de production

L'usage des objets connectés dans les usines intelligentes et la dissémination de capteurs physiques et virtuels sur toute la chaîne de production permet d'accéder pour un équipement donné à une grande quantité d'informations de différentes natures. Pour répondre aux enjeux de surveillance (détection), de diagnostic, de pronostic et de maintenance, il est nécessaire d'analyser ces flux d'informations, de réaliser leur synthèse pour offrir aux opérateurs humains de l'équipe de maintenance une aide à la décision pour faire un diagnostic fiable et robuste et décider d'une stratégie de maintenance et ainsi éviter un arrêt de production. Les équipements de production pouvant être non démontable sur site ou disséminé géographiquement (ou même inaccessible pour raisons de sécurité), l'équipe de maintenance a besoin d'outils d'analyse des informations issues des capteurs en temps réel et à distance par des techniques de Réalité Augmentée / Réalité Virtuelle.

Les objectifs principaux de ce projet sont :

1. La création d'un *jumeau numérique* du système de production afin de réaliser la collecte des données et de permettre la commande à distance du système de production.
2. Le développement d'indicateurs d'aide à la décision de maintenance par la construction d'un tableau de bord interactif pour suivre l'évolution des paramètres de productivité, de santé de chaque équipement et du système de production dans sa globalité.
3. L'insertion de ces indicateurs de productivité et de santé dans des tablettes pour la réalité augmentée afin de permettre aux opérateurs humains lorsqu'ils réalisent une maintenance sur site d'avoir des informations spécialisées sur chaque équipement visité.
4. La « navigation » par réalité virtuelle dans le jumeau numérique avec l'insertion des indicateurs de productivité et de santé dans les casques de réalité virtuelle.

5. Mise en œuvre de nos travaux : on utilisera les moyens disponibles de la plateforme CIM (Computer Integrated Manufacturing) que propose le pôle S.mart Grenoble Alpes et de la plateforme VISIONAIR (VISION Advanced Infrastructure for Research) coordonnée par le laboratoire G-SCOP qui offre les outils, les compétences et une expertise de la réalité virtuelle et de la réalité augmentée.

Notre ambition est de créer de nouveaux outils technologiques pour le diagnostic et la maintenance et de proposer leur usage pour la formation à distance des opérateurs en diagnostic et en maintenance de systèmes de production.

2.2 Prise en compte de l'humain dans la surveillance de production avec un robot collaboratif

Un robot collaboratif est un robot autonome conçu pour interagir avec les opérateurs. Il est intégré dans un espace de travail collaboratif qui peut être complété par des équipements de prévention / protection (espace de travail contrôlé) selon le type de robot et l'usage envisagé. L'interaction entre l'humain et le robot peut se décliner de différentes façons : l'opérateur et le robot participent à des tâches distinctes dans un même environnement, l'opérateur et le robot travaillent simultanément à la réalisation d'une tâche commune (collaboration directe) ou l'opérateur et le robot travaillent à tour de rôle à la réalisation d'une même tâche (collaboration indirecte).

Dans certaines situations (dysfonctionnement d'un composant physique du robot, perte de signal de contrôle à distance du robot, etc.), l'apparition d'une défaillance peut faire échouer le robot dans la réalisation de sa tâche et bloquer le processus de production. Pour éviter ce blocage, le diagnostic en temps réel intégrant l'opérateur humain dans la prise de décision semble être une solution prometteuse.

Ce projet de recherche vise à proposer un système de surveillance et d'aide au diagnostic de robot collaboratif par son opérateur humain. Le verrou scientifique que nous voulons lever est de développer un système d'aide au diagnostic co-construit à partir de connaissance experte sur le robot (à partir de modèles, de données) et de connaissances exprimées par les opérateurs humains. Le module d'aide au diagnostic que nous souhaitons construire mettrait en œuvre d'une part l'application d'un module d'autodiagnostic par le robot (le diagnostiqueur intégré dans le robot serait défini à l'aide de nos méthodes de diagnostic de systèmes dynamiques hybride et de systèmes cyber-physiques) et, d'autre part, l'intégration de l'opérateur humain dans la boucle de diagnostic en proposant des solutions et en validant / amendant les propositions du module d'autodiagnostic du robot.

Pour réaliser nos études sur les robots collaboratifs, nous pourrons nous appuyer sur les moyens disponibles de la plateforme CIM et de la plateforme Operations Management accessible aux chercheurs du laboratoire G-SCOP. Ces plateformes proposent des cobots, des robots autonomes de logistique interne et des bras robots.

Ce projet de recherche permettrait ainsi de capitaliser l'expérience et l'intelligence humaine et placerait l'opérateur humain au cœur des dynamiques 4.0.

Conclusions générales
Bibliographie

Conclusions générales

Ce manuscrit d'habilitation à diriger des recherches retrace les activités de recherche que nous menons depuis une décennie sur la thématique de la sûreté de fonctionnement (SdF) et du pronostic et management de la santé (PHM) des systèmes industriels complexes. Nos principaux travaux sont présentés dans ce mémoire au travers de neuf chapitres répartis sur trois axes de recherche.

Le premier axe de mes activités de recherches a porté sur le développement de méthodes de diagnostic avec des approches basées sur les modèles. Nous nous sommes intéressés à définir des méthodologies de modélisation et de diagnostic de systèmes industriels en utilisant différents formalismes tels que les automates temporisés communicants (chapitre I), les automates hybrides et les bond graphs (chapitre IV), les graphes d'événements temporisés (chapitre VI) ou en utilisant une modélisation hiérarchique et multi-vues (chapitre V). La problématique de l'existence d'un diagnostiqueur pour un système industriel complexe nous a conduits à étudier en amont la diagnosticabilité des systèmes à événements discrets (chapitre II). Afin de faciliter l'obtention du modèle global du système comprenant les modes nominaux et défaillants, une démarche de génération automatique du comportement dysfonctionnel d'un système de production dans le formalisme des automates temporisés communicants a été définie (chapitre III).

Pour compléter nos travaux sur le développement de méthodes de diagnostic pour des systèmes dynamiques complexes, dans un deuxième axe de recherche, nous avons développé de nouveaux outils d'aide à la décision pour le diagnostic et la maintenance. Nous avons étudié l'usage des arbres de défaillances dynamiques pour la sûreté de fonctionnement (chapitre VII) en proposant, d'une part, des méthodologies d'analyse qualitative des défaillances par la recherche de leurs causes racines et le filtrage de fausses alarmes et, d'autre part, une méthode d'analyse quantitative des arbres de défaillances dynamiques. Nous avons également proposé une méthodologie de maintenance corrective pour un système de production multi-sites (chapitre VIII). Celle-ci utilise les réseaux de files d'attente pour l'étude du dimensionnement optimal d'un atelier de maintenance centralisé ainsi que les métaheuristiques pour la recherche de la tournée optimale de l'atelier de maintenance mobile.

Finalement, dans un troisième axe de recherche, nos travaux de recherche se sont portés sur le développement de méthodes de pronostic guidées par les données. Nous avons proposé de nouvelles méthodes de pronostic de l'état de santé et de la durée de vie résiduelle de batteries lithium-ion dans un contexte d'économie circulaire (chapitre IX).

Ma démarche de recherche s'est construite conjointement sur ces trois axes pour leur complémentarité dans les domaines de la SdF et du PHM. Cette connaissance transversale me permet d'avoir une vision globale des différentes problématiques de recherche sur le diagnostic, le pronostic et la maintenance pour d'identifier puis travailler sur les verrous scientifiques de leur mise en œuvre dans le cadre de la transformation digitale de l'industrie, l'industrie du futur (Industrie 4.0).

Mes activités de recherche portent sur le triptyque « diagnostic, pronostic et aide à la décision pour la maintenance ». Mes intérêts de recherche sont ainsi multiples et complémentaires :

— Des travaux « théoriques » pour le développement de méthodes et d'outils pour modéliser, analyser et simuler des systèmes dynamiques complexes. Mon questionnement scientifique est de trouver l'outil de modélisation formelle adéquat qui me permet de représenter un système industriel et d'extraire un ensemble de connaissances à partir desquelles je peux élaborer des méthodes de diagnostic, de pronostic ou des indicateurs pour l'aide à la décision pour la maintenance. Je développe / propose alors des approches théoriques et génériques pour résoudre ces problématiques. Mes contributions sont dans ce cadre d'ordre méthodologique.

— Des travaux « techniques » qui appliquent mes approches théoriques et génériques à des cas d'études industriels. Ces études de cas (robot de téléprésence, batterie lithium-ion, ligne de remanufacturing de toners d'imprimantes, etc.) me permettent de valider mes méthodologies et de compléter et d'étendre mes approches. Mes contributions sont dans ce cadre d'ordre applicatif.

— Des travaux « pluridisciplinaires » pour adapter des solutions éprouvées dans un domaine scientifique à une problématique de recherche dans un autre domaine : c'est ainsi que j'apporte mes compétences en informatique et en modélisation et spécification formelle au domaine de l'automatique et de la productique.

Nous proposons comme projets et ambitions de recherche (chapitre X) de poursuivre nos travaux actuels en les ciblant sur les technologies de l'Industrie 4.0 (robot autonome, cobot, jumeau numérique, usine intelligente, maintenance distribué, etc.) et d'accompagner l'opérateur humain dans la transformation digitale de l'industrie en lui proposant de nouveaux outils et méthodes pour la sûreté de fonctionnement des systèmes cyber-physiques de production.

Pour conclure ... Ce manuscrit d'HDR présente la démarche scientifique, la stratégie de recherche et les travaux d'encadrement de jeunes chercheurs que j'ai réalisé au sein du laboratoire G-SCOP ces dix dernières années. J'ai souhaité retranscrire au lecteur la démarche d'apprentissage, de réflexion, de construction de nouveaux savoirs et de restitution que j'ai menés autour des thèmes du diagnostic, du pronostic et de l'aide à la décision dans la maintenance. A travers mon activité de recherche et d'enseignement, je souhaite en effet faire des ponts entre différents domaines de recherche, transmettre des savoir-faire et être un passeur de connaissances.

Références bibliographiques

[1] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Diagnosability of discrete-event systems", *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1555–1575, 1995.

[2] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Failure diagnosis using discrete-event models", *IEEE Transactions on Control Systems Technology*, vol. 4, no. 2, pp. 105–124, 1996.

[3] G. Vachtsevanos, F. Lewis, R. M., A. Hess, and B. Wu, *Intelligent fault diagnosis and prognosis for engineering systems*. Wiley Online Library, 2006, vol. 456.

[4] M. G. Singh, K. S. Hindi, G Schmidt, and S. G. Tzafestas, *Fault Detection & Reliability: Knowledge Based & Other Approaches*. Pergamon Press, 1987.

[5] N. J. Bahr, *System safety engineering and risk assessment: a practical approach*. CRC press, 2014.

[6] A. K. Verma, S. Ajit, and D. R. Karanki, *Reliability and Safety Engineering, 2nd Edition*. Springer Publishers, London, 2016.

[7] G. Zwingelstein, "Méthodes de diagnostic et de pronostic de défaillances basées sur les données – etat de l'art", *Techniques de l'ingénieur*, 2020.

[8] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning* (Springer Series in Statistics). Springer New York Inc., 2001.

[9] A. C. Faul, *A concise introduction to machine learning*. CRC Press, 2019.

[10] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[11] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis: part i: quantitative model-based methods", *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 293–311, 2003.

[12] S. Simani, C. Fantuzzi, and R. J. Patton, *Model-based fault diagnosis in dynamic systems using identification techniques* (Advances in industrial control). Springer-Verlag London, 2003.

[13] V. Venkatasubramanian, R. Rengaswamy, and S. N. Kavuri, "A review of process fault detection and diagnosis: part ii: qualitative models and search strategies", *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 313–326, 2003.

[14] M. Sayed-Mouchaweh, "Diagnostic des systèmes à événements discrets (sed) - état de l'art", *Techniques de l'ingénieur*, 2011.

[15] E. Gascard and Z. Simeu-Abazi, "Modular Modelling for the Diagnostic of Complex Discrete-Event Systems", *IEEE Transactions on Automation Science and Engineering*, vol. 10, no. 4, pp. 1101–1123, 2013.

[16] G. Behrmann, A. David, and K. Larsen, "A tutorial on UPPAAL", in *Formal Methods for the Design of Real-Time Systems*, ser. LNCS, vol. 3185, 2004, pp. 200–236.

[17] E. Gascard and Z. Simeu-Abazi, "Automatic Construction of Diagnoser for Complex Discrete Event Systems", in *Proceedings of the 3rd International Workshop on Dependable Control of Discrete Systems (DCDS 2011)*, IEEE Press, 2011, pp. 90–95.

[18] R. Reiter, "A theory of diagnosis from first principles", *Artificial intelligence*, vol. 32, no. 1, pp. 57–95, 1987.

[19] J. De Kleer and B. C. Williams, "Diagnosing multiple faults", *Artificial intelligence*, vol. 32, no. 1, pp. 97–130, 1987.

[20] J. De Kleer, A. K. Mackworth, and R. Reiter, "Characterizing diagnoses and systems", *Artificial intelligence*, vol. 56, no. 2-3, pp. 197–222, 1992.

[21] M.-O. Cordier, P. Dague, F. Levy, J. Montmain, M. Staroswiecki, and L. Trave-Massuyes, "Conflicts versus analytical redundancy relations: a comparative analysis of the model based diagnosis approach from the artificial intelligence and automatic control perspectives", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 5, pp. 2163–2177, 2004.

[22] F. Lin, "Diagnosability of discrete event systems and its applications", *Discrete Event Dynamic Systems*, vol. 4, no. 2, pp. 197–212, 1994.

[23] T. Ushio, I. Onishi, and K. Okuda, "Fault detection based on petri net models with faulty behaviors", in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC'98)*, IEEE, vol. 1, 1998, pp. 113–118.

[24] S. H. Zad, R. H. Kwong, and W. M. Wonham, "Fault diagnosis in discrete-event systems: framework and model reduction", *IEEE Transactions on Automatic Control*, vol. 48, no. 7, pp. 1199–1212, 2003.

[25] S. Jiang, Z. Huang, V. Chandra, and R. Kumar, "A polynomial algorithm for testing diagnosability of discrete-event systems", *IEEE Transactions on Automatic Control*, vol. 46, no. 8, pp. 1318–1321, 2001.

[26] T.-S. Yoo and S. Lafortune, "Polynomial-time verification of diagnosability of partially observed discrete-event systems", *IEEE Transactions on automatic control*, vol. 47, no. 9, pp. 1491–1495, 2002.

[27] F. Cassez and S. Tripakis, "Fault diagnosis with dynamic observers", in *Proceedings of the 9th International Workshop on Discrete Event Systems*, IEEE, 2008, pp. 212–217.

[28] M. N. Hosseini, B. Lennartson, M. P. Cabasino, and C. Seatzu, "A survey on efficient diagnosability tests for automata and bounded petri nets", in *Proceedings of the 18th IEEE Conference on Emerging Technologies & Factory Automation (ETFA)*, IEEE, 2013, pp. 1–6.

[29] A. Boussif, M. Ghazel, and K. Klai, "Combining enumerative and symbolic techniques for diagnosis of discrete-event systems", in *Proceedings of the 9th Workshop on Verification and Evaluation of Computer and Communication Systems (VECOS 2015)*, 2015, 11p.

[30] A. Cimatti, C. Pecheur, and R. Cavada, "Formal verification of diagnosability via symbolic model checking", in *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, Morgan Kaufmann Publishers Inc., 2003, 363—369.

[31] A. Boussif and M. Ghazel, "Une approche par décomposition de modèles pour l'analyse de la diagnosticabilité des seds par model-checking", in *Proceedings des 10ième Colloque sur la Modélisation des Systèmes Réactifs (MSR 2015)*, 2015.

[32] T. M. Tuxi, L. K. Carvalho, E. V. Nunes, and A. E. Da Cunha, "Is ltl model-checking effective for diagnosability verification?", *IFAC-PapersOnLine*, vol. 53, no. 4, pp. 256–262, 2020.

[33] D. Lefebvre and C. Delherm, "Diagnosis of des with petri net models", *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 114–118, 2007.

[34] D. Lefebvre and E. Leclercq, "Diagnosability of petri nets with observation graphs", *Discrete Event Dynamic Systems*, vol. 26, no. 3, pp. 539–559, 2016.

[35] G. Jiroveanu and R. K. Boel, "The diagnosability of petri net models using minimal explanations", *IEEE Transactions on Automatic Control*, vol. 55, no. 7, pp. 1663–1668, 2010.

[36] M. Dotoli, M. P. Fanti, A. M. Mangini, and W. Ukovich, "On-line fault detection in discrete event systems by petri nets and integer linear programming", *Automatica*, vol. 45, no. 11, pp. 2665–2672, 2009.

[37] A. Madalinski, F. Nouioua, and P. Dague, "Diagnosability verification with petri net unfoldings", *International Journal of Knowledge-Based and Intelligent Engineering Systems*, vol. 14, no. 2, pp. 49–55, 2010.

[38] B. Liu, M. Ghazel, and A. Toguyéni, "Toward an efficient approach for diagnosability analysis of des modeled by labeled petri nets", in *Proceedings of the IEEE European Control Conference (ECC)*, IEEE, 2014, pp. 1293–1298.

[39] M. P. Cabasino, A. Giua, S. Lafortune, and C. Seatzu, "A new approach for diagnosability analysis of petri nets using verifier nets", *IEEE Transactions on Automatic Control*, vol. 57, no. 12, pp. 3104–3117, 2012.

[40] M. Sayed-Mouchaweh, "Diagnostic décentralisé des systèmes à evénements discrets", *Techniques de l'Ingénieur*, 2014.

[41] R. Debouk, S. Lafortune, and D. Teneketzis, "Coordinated decentralized protocols for failure diagnosis of discrete event systems", *Discrete Event Dynamic Systems*, vol. 10, no. 1, pp. 33–86, 2000.

[42] O. Contant, S. Lafortune, and D. Teneketzis, "Diagnosability of discrete event systems with modular structure", *Discrete Event Dynamic Systems*, vol. 16, no. 1, pp. 9–37, 2006.

[43] Y. Wang, T.-S. Yoo, and S. Lafortune, "Diagnosis of discrete event systems using decentralized architectures", *Discrete Event Dynamic Systems*, vol. 17, no. 2, pp. 233–263, 2007.

[44] W. Qiu and R. Kumar, "Decentralized failure diagnosis of discrete event systems", *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 36, no. 2, pp. 384–395, 2006.

[45] M. V. Moreira, T. C. Jesus, and J. C. Basilio, "Polynomial time verification of decentralized diagnosability of discrete event systems", *IEEE Transactions on Automatic Control*, vol. 56, no. 7, pp. 1679–1684, 2011.

[46] M. P. Cabasino, A. Giua, A. Paoli, and C. Seatzu, "Decentralized diagnosability analysis of discrete event systems using petri nets", *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 6060–6066, 2011.

[47] N. Ran, H. Su, A. Giua, and C. Seatzu, "Codiagnosability analysis of bounded petri nets", *IEEE Transactions on Automatic Control*, vol. 63, no. 4, pp. 1192–1199, 2018.

[48] Y. Pencolé, "Diagnosability analysis of distributed discrete event systems", in *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI)*, IOS Press, vol. 16, 2004, pp. 38–43.

[49] A. Schumann and Y. Pencolé, "Scalable diagnosability checking of event-driven systems.", in *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, AAAI Press, 2007, pp. 575–580.

[50] L. Ye and P. Dague, "An optimized algorithm for diagnosability of component-based systems", *IFAC Proceedings Volumes*, vol. 43, no. 12, pp. 143–148, 2010.

[51] P. Marangé, A. Philippot, J.-F. Pétin, and F. Gellot, "Diagnosability evaluation by model-checking", *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 308–313, 2015.

[52] S. Tripakis, "Fault diagnosis for timed automata", in *Proceedings of the 7th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT)*, Springer, 2002, pp. 205–221.

[53] P. Bouyer, F. Chevalier, and D. D'Souza, "Fault diagnosis using timed automata", in *Proceedings of the 8th International Conference on Foundations of Software Science and Computation Structures (FOSSACS)*, Springer, 2005, pp. 219–233.

[54] J Pan and S. Hashtrudi-Zad, "Diagnosability test for timed discrete-event systems", in *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)*, IEEE, 2006, pp. 63–72.

[55] F. Cassez and S. Tripakis, "Chapter 6. diagnostic des systèmes temporisés", in *Systèmes embarqués – Approches formelles*. Hermes Science, 2008.

[56] B. Liu, M. Ghazel, and A. Toguyéni, "Evaluation àa la volée de la diagnosticabilité des systèmes à événements discrets temporisés", *Journal Européen des Systemes Automatisés, Edition spéciale MSR*, vol. 13, pp. 227–242, 2013.

[57] X. Wang, C. Mahulea, and M. Silva, "Diagnosis of time petri nets using fault diagnosis graph", *IEEE Transactions on Automatic Control*, vol. 60, no. 9, pp. 2321–2335, 2015.

[58] P. Bonhomme, "Towards a diagnosability technique of p-time petri nets systems", in *Proceedings of the 3rd International Conference on Control, Decision and Information Technologies (CoDIT)*, IEEE, 2016, pp. 18–23.

[59] F. Basile, M. P. Cabasino, and C. Seatzu, "Diagnosability analysis of labeled time petri net systems", *IEEE Transactions on Automatic Control*, vol. 62, no. 3, pp. 1384–1396, 2016.

[60] F. Cassez, "The complexity of codiagnosability for discrete event and timed systems", *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1752–1764, 2012.

[61] P. Bonhomme, "Decentralized diagnosis of p-time petri nets systems", in *Proceedings of the 4th International Conference on Control, Decision and Information Technologies (CoDIT)*, IEEE, 2017, pp. 483–0488.

[62] G. Viana, M. V. S. Alves, and J. C. Basilio, "Codiagnosability of networked discrete event systems with timing structure", *IEEE Transactions on Automatic Control*, 2021.

[63] T. A. Henzinger, "The theory of hybrid automata", in *Verification of Digital and Hybrid Systems*, M. K. Inan and R. P. Kurshan, Eds. Springer Berlin Heidelberg, 2000, pp. 265–292.

[64] J. Lunze and F. Lamnabhi-Lagarrigue, *Handbook of hybrid systems control: theory, tools, applications*. Cambridge University Press, 2009.

[65] G. Fourlas, K. J. Kyriakopoulos, and N. Krikelis, "Diagnosability of hybrid systems", in *Proceedings of the 10th IEEE Mediterranean Conference on Control and Automation (MED2002)*, 2002.

[66] M. Bayoudh, L. Travé-Massuyes, and X. Olive, "Hybrid systems diagnosability by abstracting faulty continuous dynamics", in *Proceedings of the 17th International Workshop on Principles of Diagnosis (DX)*, 2006, pp. 9–15.

[67] S Biswas, D Sarkar, S Mukhopadhyay, and A Patra, "Diagnosability analysis of real time hybrid systems", in *Proceedings of the IEEE International Conference on Industrial Technology (ICIT)*, IEEE, 2006, pp. 104–109.

[68] M. J. Daigle, X. D. Koutsoukos, and G. Biswas, "An event-based approach to integrated parametric and discrete fault diagnosis in hybrid systems", *Transactions of the Institute of Measurement and Control*, vol. 32, no. 5, pp. 487–510, 2010.

[69] M. Bayoudh and L. Travé-Massuyès, "Diagnosability analysis of hybrid systems cast in a discrete-event framework", *Discrete Event Dynamic Systems*, vol. 24, no. 3, pp. 309–338, 2014.

[70] A. Grastien, L. Travé-Massuyès, and V. Puig, "Solving diagnosability of hybrid systems via abstraction and discrete event techniques", *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 5023–5028, 2017.

[71] H. Zaatiti, L. Ye, P. Dague, and J.-P. Gallois, "Counterexample-guided abstraction-refinement for hybrid systems diagnosability analysis.", in *Proceedings of the 28th International Workshop on Principles of Diagnosis (DX)*, 2017, pp. 124–143.

[72] O. Diene, M. V. Moreira, E. A. Silva, V. R. Alvarez, and C. F. Nascimento, "Diagnosability of hybrid systems", *IEEE Transactions on Control Systems Technology*, vol. 27, no. 1, pp. 386–393, 2017.

[73] D. Thorsley and D. Teneketzis, "Diagnosability of stochastic discrete-event systems", *IEEE Transactions on Automatic Control*, vol. 50, no. 4, pp. 476–492, 2005.

[74] F. Liu, D. Qiu, H. Xing, and Z. Fan, "Decentralized diagnosis of stochastic discrete event systems", *IEEE Transactions on Automatic Control*, vol. 53, no. 2, pp. 535–546, 2008.

[75] J. Chen, C. Keroglou, C. N. Hadjicostis, and R. Kumar, "Revised test for stochastic diagnosability of discrete-event systems", *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 1, pp. 404–408, 2016.

[76] H. Bazille, E. Fabre, and B. Genest, "Complexity reduction techniques for quantified diagnosability of stochastic systems", *IFAC-PapersOnLine*, vol. 51, no. 7, pp. 82–87, 2018.

[77] H. Ibrahim, "Sat-based diagnosability and predictability analysis in centralized and distributed discrete event systems", Ph.D. dissertation, Université Paris Saclay, 2016.

[78] B. Li, "Diagnosis and diagnosability of complex discrete event systems modeled by labeled petri nets", Ph.D. dissertation, Ecole Centrale de Lille, 2017.

[79] R. Ammour, "Contribution au diagnostic et pronostic des systèmes à évènements discrets temporisés par réseaux de petri stochastiques", Ph.D. dissertation, Normandie Université, 2017.

[80] H. Zaatiti, "Modeling and qualitative simulation of hybrid systems", Ph.D. dissertation, Université Paris-Saclay, 2018.

[81] S. Rachidi, "Diagnostic des défauts dans les systèmes à événements discrets soumis à des contraintes temporelles", Ph.D. dissertation, Normandie Université, 2019.

[82] H. Bazille, "Detection and quantification of events in stochastic systems", Ph.D. dissertation, Université de Rennes1, 2019.

[83] E. Gascard, Z. Simeu-Abazi, and B. Suiphon, "A Polynomial-Time Algorithm for Diagnosability Verification of Discrete Event Systems", in *Proceedings of the 2nd World Conference on Complex Systems (WCCS 2014)*, IEEE Press, 2014, pp. 286–291.

[84] E. Gascard, Z. Simeu-Abazi, and B. Suiphon, "A Polynomial Algorithm for Diagnosability Analysis of Discrete Event Systems", *International Journal of Adaptive, Resilient and Autonomic Systems (IJARAS)*, vol. 6, no. 2, pp. 1–22, 2015.

[85] J. Zaytoon and S. Lafortune, "Overview of fault diagnosis methods for discrete event systems", *Annual Reviews in Control*, vol. 37, no. 2, pp. 308–320, 2013.

[86] A. K. A. Toguyeni, E. Craye, and J.-C. Gentina, "A method of temporal analysis to perform online diagnosis in the context of flexible manufacturing system", in *Proceedingsof the 16th Annual Conference of IEEE Industrial Electronics Society*, IEEE, 1990, pp. 445–450.

[87] D. N. Pandalai and L. E. Holloway, "Template languages for fault monitoring of timed discrete event processes", *IEEE transactions on automatic control*, vol. 45, no. 5, pp. 868–882, 2000.

[88] A. M'Halla, S. Collart Dutilleul, E. Craye, and M. Benrejeb, "Monitoring of a milk manufacturing workshop using chronicle and fault tree approaches", *Studies in Informatics and Control*, vol. 19, no. 4, pp. 377–390, 2010.

[89] A. M'Halla, "Online diagnosis based on chronicle recognition of a coil winding machine", *International Journal of Engineering and Applied Physics*, vol. 1, no. 2, pp. 76–83, 2021.

[90] M. Roth, J.-J. Lesage, and L. Litz, "The concept of residuals for fault localization in discrete event systems", *Control Engineering Practice*, vol. 19, no. 9, pp. 978–988, 2011.

[91] M. V. Moreira and J.-J. Lesage, "Fault diagnosis based on identified discrete-event models", *Control Engineering Practice*, vol. 91, p. 104 101, 2019.

[92] S. Schneider, L. Litz, and M. Danancher, "Timed residuals for fault detection and isolation in discrete event systems", in *Proceedings of the 3rd International Workshop on Dependable Control of Discrete Systems*, IEEE, 2011, pp. 35–40.

[93] M. Sayed-Mouchaweh, "Decentralized fault free model approach for fault detection and isolation of discrete event systems", *European Journal of Control*, vol. 18, no. 1, pp. 82–93, 2012.

[94] Y. Pencolé, G. Steinbauer, C. Mühlbacher, and L. Travé-Massuyès, "Diagnosing discrete event systems using nominal models only", in *Proccedings of the 28th International Workshop on Principles of Diagnosis (DX'17)*, 2017, pp. 169–183.

[95] N. Bauer, S. Engell, R. Huuck, *et al.*, "Verification of plc programs given as sequential function charts", in *Integration of software specification techniques for applications in Engineering*. Springer, 2004, pp. 517–540.

[96] E. Gascard, Z. Simeu-Abazi, and G. Mayol, "Elaboration du comportement dynamique d'un système pour le diagnostic des défaillances", in *10ème Congrès International de Génie Industriel (CIGI 2013)*, 2013.

[97] J. Zaytoon, *Systèmes dynamiques hybrides*. Hermès Science publications, 2001.

[98] M. Bertrand, C. Iung, and J. Zaytoon, "Systèmes dynamiques hybrides - modélisation et simulation", *Techniques de l'ingénieur*, 2004.

[99] P. J. Antsaklis, X. D. Koutsoukos, and N Dame, "Hybrid systems control", *Encyclopedia of Physical Science and Technology*, vol. 7, pp. 445–458, 2002.

[100] D. E. C. Belkhiat, "Diagnostic d'une classe de systèmes linéaires à commutations: approche à base d'observateurs robustes", Ph.D. dissertation, Université de Reims Champagne Ardenne, 2011.

[101] R. Alur and D. L. Dill, "A theory of timed automata", *Theoretical computer science*, vol. 126, no. 2, pp. 183–235, 1994.

[102] R. David and H. Alla, *Discrete, continuous, and hybrid Petri nets*. Springer, 2010, vol. 1.

[103] J. Buisson, "Analysis of switching devices with bond graphs", *journal of the Franklin Institute*, vol. 330, no. 6, pp. 1165–1175, 1993.

[104] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho, "Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems", in *Hybrid systems*, Springer, 1992, pp. 209–229.

[105] R. Alur, C. Courcoubetis, N. Halbwachs, *et al.*, "The algorithmic analysis of hybrid systems", *Theoretical computer science*, vol. 138, no. 1, pp. 3–34, 1995.

[106] Y. Kesten and A. Pnueli, "Timed and hybrid statecharts and their textual representation", in *International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*, Springer, 1992, pp. 591–620.

[107] H. Alla and R. David, "Continuous and hybrid petri nets", *Journal of Circuits, Systems, and Computers*, vol. 8, no. 01, pp. 159–188, 1998.

[108] P. J. Mosterman and G. Biswas, "Behavior generation using model switching: a hybrid bond graph modeling technique", *Trans. Soc. Simul.*, vol. 27, no. 1, pp. 177–182, 1995.

[109] M. Sayed-Mouchaweh, "Diagnostic des systèmes dynamiques hybrides (sdh)", *Techniques de l'ingénieur*, 2015.

[110] M. I. Rahal, "Génération d'algorithmes de diagnostic robustes à base de modèles bond graph hybrides", Ph.D. dissertation, Université des sciences et technologies de Lille 1, 2016.

[111] J. Gertler, "Analytical redundancy methods in fault detection and isolation-survey and synthesis", *IFAC Proceedings Volumes*, vol. 24, no. 6, pp. 9–21, 1991.

[112] J. J. Gertler and R. Monajemy, "Generating directional residuals with dynamic parity relations", *Automatica*, vol. 31, no. 4, pp. 627–635, 1995.

[113] X. Koutsoukos, F. Zhao, H. Haussecker, J. Reich, and P. Cheung, "Fault modeling for monitoring and diagnosis of sensor-rich hybrid systems", in *Proceedings of the 40th IEEE Conference on Decision and Control*, IEEE, vol. 1, 2001, pp. 793–801.

[114] V. Cocquempot, T. El Mezyani, and M. Staroswiecki, "Fault detection and isolation for hybrid systems using structured parity residuals", in *Proceedings of the 5th Asian Control Conference*, IEEE, vol. 2, 2004, pp. 1204–1212.

[115] V Coquempot, T. El Mezyani, and M. Staroswiecki, "Hybrid dynamical systems monitoring using structured analytical redundancy relations", in *Proceedings of 17ème IMACS World Congress*, 2005.

[116] S. Narasimhan and G. Biswas, "Model-based diagnosis of hybrid systems", *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and humans*, vol. 37, no. 3, pp. 348–361, 2007.

[117] T. Zouari, "Diagnostic des systèmes dynamiques hybrides à modes non linéaires", Ph.D. dissertation, Université des sciences et technologies de Lille 1, 2013.

[118] H. Khorasgani and G. Biswas, "Structural fault detection and isolation in hybrid systems", *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 4, pp. 1585–1599, 2018.

[119] O. Prakash, A. K. Samantaray, and R. Bhattacharyya, "Model-based diagnosis of multiple faults in hybrid dynamical systems with dynamically updated parameters", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 6, pp. 1053–1072, 2019.

[120] E. R. Loures and J.-C. Pascal, "Detection and diagnosis of hybrid dynamic systems based on time fuzzy petri nets", in *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, IEEE, vol. 2, 2004, pp. 1825–1831.

[121] E. R. Loures and J. Pascal, "A diagnosis framework of hybrid dynamic systems based on time fuzzy petri nets", *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 55–60, 2005.

[122] P. Bhowal, D. Sarkar, S. Mukhopadhyay, and A. Basu, "Fault diagnosis in discrete time hybrid systems–a case study", *Information Sciences*, vol. 177, no. 5, pp. 1290–1308, 2007.

[123] J. Lunze, "Diagnosis of quantised systems", *IFAC Proceedings Volumes*, vol. 33, no. 11, pp. 29–40, 2000.

[124] J. Lunze, "Diagnosis of discretely controlled continuous systems (diagnose ereignisgesteuerter systeme)", 2006.

[125] F. Zhao, X. Koutsoukos, H. Haussecker, J. Reich, and P. Cheung, "Monitoring and fault diagnosis of hybrid systems", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 6, pp. 1225–1240, 2005.

[126] G. Karsai, S. Abdelwahed, and G. Biswas, "Integrated diagnosis and control for hybrid dynamic systems", in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003, p. 5673.

[127] R. Mohammadi, S. Hashtrudi-Zad, and K. Khorasani, "Diagnosis of hybrid systems: part 2-residual generator selection and diagnosis in the presence of unreliable residual generators", in *2009 IEEE International Conference on Systems, Man and Cybernetics*, IEEE, 2009, pp. 3340–3345.

[128] H. Derbel, H. Alla, N. B. Hadj-Alouane, and M. Yeddes, "Online diagnosis of systems with rectangular hybrid automata models", *IFAC Proceedings Volumes*, vol. 42, no. 4, pp. 954–959, 2009.

[129] M. Bayoudh, L. Travé-Massuyes, and X. Olive, "Towards active diagnosis of hybrid systems", in *International Workshop on Principles of Diagnosis*, Citeseer, 2008.

[130] M. Bayoudh, L. Travé-Massuyes, and X. Olive, "Hybrid systems diagnosis by coupling continuous and discrete event techniques", *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 7265–7270, 2008.

[131] N. Olivier-Maget, G. Hétreux, J. M. Le Lann, and M. V. Le Lann, "Integration of a failure monitoring within a hybrid dynamic simulation environment", *Chemical Engineering and Processing: process intensification*, vol. 47, no. 11, pp. 1942–1952, 2008.

[132] N. Olivier-Maget, G. Hetreux, J. M. Le Lann, and M. V. Le Lann, "Model-based fault diagnosis for hybrid systems: application on chemical processes", *Computers & Chemical Engineering*, vol. 33, no. 10, pp. 1617–1630, 2009.

[133] J. Perret, G. Hétreux, and J.-M. Le Lann, "Object hybrid formalism for modeling and simulation of chemical processes", *IFAC Proceedings Volumes*, vol. 36, no. 6, pp. 277–282, 2003.

[134] D. Belkhiat, N Messai, and N Manamanni, "Design of a robust fault detection based observer for linear switched systems with external disturbances", *Nonlinear Analysis: Hybrid Systems*, vol. 5, no. 2, pp. 206–219, 2011.

[135] H. Louajri, M Sayed-Mouchaweh, and C Labarre, "Diagnoser with hybrid structure for fault diagnosis of a class of hybrid dynamic systems", *Chemical engineering transactions*, vol. 33, pp. 85–90, 2013.

[136] T. Asma, L. Islem, N. Zanzouri, and M. Ksouri, "Robust diagnosis for hybrid dynamical systems", in *2015 IEEE 12th International Multi-Conference on Systems, Signals & Devices (SSD15)*, IEEE, 2015, pp. 1–6.

[137] M. I. Rahal, B. Ould Bouamama, and A. Meghebbar, "Hybrid bond graph model based for robust fault detection and isolation", *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 230, no. 2, pp. 145–163, 2016.

[138] B. Maaref, "Contribution au diagnostic de défauts des systèmes hybrides complexes", Ph.D. dissertation, Communauté Université Grenoble Alpes & Université de Monastir (Tunisie), 2019.

[139] L. Belkacem, "Mise en œuvre d'une architecture pour le diagnostic et le pronostic des systèmes dynamiques hybrides", Ph.D. dissertation, Université de Monastir, 2019.

[140] G. Dauphin-Tanguy, *Les bond graphs*. Hermès science, 2000.

[141] A. K. Samantaray and B. O. Bouamama, *Model-based process supervision: a bond graph approach*. Springer, 2008.

[142] B. Maaref, Z. Simeu-Abazi, H. Dhouibi, H. Messaoud, and E. Gascard, "Mixed approach for fault diagnosis and fault location of hybrid systems", in *Proceedings of the 8th IFAC Conference on Manufacturing Modelling, Management, and Control (MIM 2016)*, ser. IFAC-PaperOnLine, ELSEVIER, vol. 49, 2016, pp. 1002–1007.

[143] L. Belkacem, Z. Simeu-Abazi, H. Dhouibi, E. Gascard, and H. Messaoud, "Diagnostic and prognostic of hybrid dynamic systems: Modeling and RUL evaluation

for two maintenance policies", *Reliability Engineering & System Safety*, vol. 164, pp. 98–109, 2017.

[144] L. Belkacem, Z. Simeu-Abazi, L. Mhamdi, H. Messaoud, and E. Gascard, "Diagnosis of Hybrid Dynamical Systems through Hybrid Automata", in *Proceedings of the 8th IFAC Conference on Manufacturing Modelling, Management, and Control (MIM 2016)*, ser. IFAC-PaperOnLine, ELSEVIER, vol. 49, 2016, pp. 990–995.

[145] B. Suiphon, Z. Simeu-Abazi, and E. Gascard, "Implementation of a fault diagnosis method for timed discrete-event systems", in *Proceedings of the 5th International Conference on Industrial Engineering and Systems Management (IESM 2013)*, IEEE Press, 2013, pp. 870–877.

[146] Z. Simeu-Abazi and E. Gascard, "Fault diagnosis method for timed discrete-event systems: Application to autonomous electric vehicle", in *Proceedings of the 3rd International Conference on Control, Automation and Diagnosis (ICCAD'19)*, IEEE Press, 2019, pp. 374–379.

[147] I. Hnayen, H. Dhouibi, C. Ben Njima, E. Gascard, and Z. Simeu-Abazi, "Fault diagnosis and modeling of hybrid systems through hybrid automata", in *Proceedings of the 6th International Conference on Control, Automation and Diagnosis (ICCAD'22)*, IEEE Press, 2022.

[148] M. A. Haj Kacem, "Contribution au développement d'une méthodologie de diagnostic des systèmes cyber-physique", Ph.D. dissertation, Communauté Université Grenoble Alpes, 2018.

[149] E. A. Lee, "Cyber-physical systems-are computing foundations adequate", in *Proceedings on the NSF Workshop on Cyber-Physical Systems: Research Motivation, Techniques and Roadmap*, 2006.

[150] E. A. Lee, "Cyber physical systems: design challenges", in *Proceedings of the 11th IEEE international symposium on object and component-oriented real-time distributed computing (ISORC)*, IEEE, 2008, pp. 363–369.

[151] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: the next computing revolution", in *Proceedings of the 47th Design Automation Conference*, Association for Computing Machinery, 2010, 731–736.

[152] L. Monostori, "Cyber-physical production systems: roots, expectations and r&d challenges", *Procedia Cirp*, vol. 17, pp. 9–13, 2014.

[153] V. Gunes, S. Peter, T. Givargis, and F. Vahid, "A survey on concepts, applications, and challenges in cyber-physical systems", *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 8, no. 12, pp. 4242–4268, 2014.

[154] J. Klimeš, "Using formal concept analysis for control in cyber-physical systems", *Procedia Engineering*, vol. 69, pp. 1518–1522, 2014.

[155] O. Cardin, "Contribution à la conception, l'évaluation et l'implémentation de systèmes de production cyber-physiques", Habilitation à diriger des recherches, Université de Nantes, 2016.

[156] H. Chen, "Applications of cyber-physical system: a literature review", *Journal of Industrial Integration and Management*, vol. 2, no. 03, p. 1 750 012, 2017.

[157] E. A. Lee and S. A. Seshia, *Introduction to embedded systems: A cyber-physical systems approach*. Mit Press, 2016.

[158] G. Zwingelstein, "Sûreté de fonctionnement - principaux concepts", *Techniques de l'ingénieur*, 2019.

[159] M. Dievart, "Architectures de diagnostic et de pronostic distribuées de systèmes techniques complexes de grande dimension", Ph.D. dissertation, Institut National Polytechnique de Toulouse, 2010.

[160] M. Diévart, P. Charbonnaud, and X. Desforges, "Applicative architecture for embedded distributed technical diagnosis", in *7th Worshop on Advanced Control and Diagnosis (ACD'2009)*, 2009, pp–1.

[161] B. Dowdeswell, R. Sinha, and S. G. MacDonell, "Finding faults: a scoping study of fault diagnostics for industrial cyber–physical systems", *Journal of systems and software*, vol. 168, p. 110 638, 2020.

[162] A. Zolghadri, J. Cieslak, D. Efimov, *et al.*, "Signal and model-based fault detection for aircraft systems", *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 1096–1101, 2015.

[163] S. Windmann and O. Niggemann, "Efficient fault detection for industrial automation processes with observable process variables", in *13th international conference on industrial informatics (indin)*, IEEE, 2015, pp. 121–126.

[164] J. P. N. Gonzalez, L. E. G. Castanon, A. Rabhi, A. El Hajjaji, and R. Morales-Menendez, "Vehicle fault detection and diagnosis combining aann and anfi", *IFAC Proceedings Volumes*, vol. 42, no. 8, pp. 1079–1084, 2009.

[165] H. Fang, H. Shi, Y. Dong, H. Fan, and S. Ren, "Spacecraft power system fault diagnosis based on dnn", in *2017 prognostics and system health management conference (phm-harbin)*, IEEE, 2017, pp. 1–5.

[166] C. Sankavaram, A. Kodali, K. Pattipati, S. Singh, Y. Zhang, and M. Salman, "An inference-based prognostic framework for health management of automotive systems", *International Journal of Prognostics and Health Management*, vol. 7, no. 2, 2016.

[167] Y.-g. Chen, "Applications of bayesian network in fault diagnosis of braking system", in *Third International Conference on Intelligent Human-Machine Systems and Cybernetics*, IEEE, vol. 1, 2011, pp. 234–237.

[168] Y. Zhou and Y. Zhang, "Applications of bayesian network in fault diagnosis of braking deviation system", in *Fourth International Symposium on Computational Intelligence and Design*, IEEE, vol. 1, 2011, pp. 170–173.

[169] M. P. Cabasino, C. Seatzu, C. Mahulea, and M. Silva, "Fault diagnosis of manufacturing systems using continuous petri nets", in *IEEE international conference on systems, man and cybernetics*, IEEE, 2010, pp. 534–539.

[170] D. Kim, S. C. Han, Y. Lin, B. H. Kang, and S. Lee, "Rdr-based knowledge based system to the failure detection in industrial cyber physical systems", *Knowledge-Based Systems*, vol. 150, pp. 1–13, 2018.

[171] N. Ali and J.-E. Hong, "Failure detection and prevention for cyber-physical systems using ontology-based knowledge base", *Computers*, vol. 7, no. 4, p. 68, 2018.

[172] M. A. Saez, F. P. Maturana, K. Barton, and D. M. Tilbury, "Context-sensitive modeling and analysis of cyber-physical manufacturing systems for anomaly detection and diagnosis", *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 1, pp. 29–40, 2020.

[173] A. Bunte, B. Stein, and O. Niggemann, "Model-based diagnosis for cyber-physical production systems based on machine learning and residual-based diagnosis models", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 2727–2735.

[174] D. Ait-Kadi, J. Gao, and M.-C. Portmann, "Minimisation des coûts de détection de pannes à partir des coupes minimales du diagramme de fiabilité", in *3ème Conférence Francophone de MOdélisation et SIMulation "Conception, Analyse et Gestion des Systèmes Industriels" MOSIM'01*, vol. 2, 2001, pp. 781–787.

[175] F. Kechaou, "Implémentation d'une méthode de diagnostic à un robot de téléprésence robair", M.S. thesis, GRENOBLE INP – Génie industriel, 2017.

[176] M. A. Haj Kacem, Z. Simeu-Abazi, and E. Gascard, "Diagnostic des défaillances des systèmes cyber-physiques par la modélisation des connaissances", in *11ième Congrès International de Génie Industriel (CIGI 2015)*, 2015.

[177] M. A. Haj Kacem, Z. Simeu-Abazi, and E. Gascard, "Failure identification and propagation analysis of cyber-physical systems", in *3rd Workshop on Advanced Maintenance Engineering, Service and Technology (AMEST 2016)*, 2016.

[178] M. A. Haj Kacem, Z. Simeu-Abazi, E. Gascard, G. Lemasson, and J. Maisonnasse, "Application of a modeling approach on a cyber-physical system RobAIR", in *Proceedings of the 20th World Congress of the International Federation of Automatic Control (IFAC WC 2017)*, ser. IFAC-PaperOnLine, ELSEVIER, vol. 50, 2017, pp. 14 230–14 235.

[179] Z. Simeu-Abazi, E. Gascard, and M. A. Haj Kacem, "Implementation of start-up tests for system health assessment: Application to a telepresence robot RobAIR", in *Proceedings of the 3rd International Conference on Control, Automation and Diagnosis (ICCAD'19)*, IEEE Press, 2019, pp. 33–38.

[180] M. Sayed-Mouchaweh, *Diagnosability, Security and Safety of Hybrid Dynamic and Cyber-Physical Systems*. Springer, 2018.

[181] Y. Luo, Y. Xiao, L. Cheng, G. Peng, and D. Yao, "Deep learning-based anomaly detection in cyber-physical systems: progress and opportunities", *ACM Computing Surveys*, vol. 54, no. 5, pp. 1–36, 2021.

[182] Q. Xu, S. Ali, and T. Yue, "Digital twin-based anomaly detection in cyber-physical systems", in *Proceedings of the 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, IEEE, 2021, pp. 205–216.

[183] J Carlier, P. Chretienne, and C. Girault, "Modelling scheduling problems with timed Petri nets", in *Advances in Petri Nets*, Springer, 1985, pp. 62–82.

[184] A. Di Febbraro and R. Minciardi, "Representation of Manufacturing Systems by Timed Event Graphs", in *International Conference on Systems, Man, and Cybernetics*, IEEE, 1992.

[185] G. Schullerus and V. Krebs, "Diagnosis of batch processes based on parameter estimation of discrete event models", in *European Control Conference*, IEEE, 2001.

[186] G. M. Provan, "An algebraic approach for diagnosing discrete-time hybrid systems.", in *International Workshop on Principles of Diagnosis*, 2017, pp. 37–51.

[187] C. Paya, E. Le Corronc, Y. Pencolé, and P. Vialletelle, "Observer-based detection of time shift failures in (max,+)-linear systems", in *International Workshop on Principles of Diagnosis*, 2020.

[188] E. Le Corronc, Y. Pencolé, A. Sahuguède, and C. Paya, "Failure detection and localization for timed event graphs in (max,+)-algebra", *Discrete Event Dynamic Systems*, vol. 31, no. 4, pp. 513–552, 2021.

[189] C. Ramchandani, "Analysis of asynchronous concurrent systems by timed petri nets", Ph.D. dissertation, Massachusetts Institute of technology, 1973.

[190] E. Gascard, Z. Simeu-Abazi, and N. Moussa, "Detection and Localization of Time Shift Failures in Timed Event Graphs: Application to a Remanufacturing Line", in *Proceedings of the 31st European Safety and Reliability Conference (ESREL 2021)*, Research Publishing, Singapore, 2021, pp. 1–8.

[191] E. Le Corronc, A. Sahuguède, Y. Pencolé, and C. Paya, "Localization of time shift failures in (max,+)-linear systems", *IFAC-PapersOnLine*, vol. 51, no. 7, pp. 186–191, 2018.

[192] J.-C. Laprie, "Sûreté de fonctionnement des systèmes informatiques et tolérance aux fautes", *Techniques de l'ingénieur*, 1989.

[193] Y. Mortureux, "La sûreté de fonctionnement : méthodes pour maîtriser les risques", *Techniques de l'ingénieur*, 2005.

[194] W. Vesely, F. Goldberg, N. Roberts, and D. Haasl, *Fault Tree Handbook*. U.S. Nuclear Regulatory Commission, 1981.

[195] A. Villemeur, *Sureté de fonctionnement des systèmes industriels: fiabilité-facteurs humains, informatisation*. Eyrolles, 1988.

[196] Y. Mortureux, "Arbres de défaillance, des causes et d'événement", *Techniques de l'ingénieur*, 2002.

[197] J.-P. Signoret, "Arbre de défaillance - aspects temporels", *Techniques de l'ingénieur*, 2017.

[198] J. Dugan, S. Bavuso, and M. Boyd, "Fault trees and sequence dependencies", in *Proceedings of the Annual Reliability and Maintainability Symposium, RAMS'90*, 1990.

[199] J. B. Dugan, S. J. Bavuso, and M. A. Boyd, "Dynamic fault tree models for fault tolerant computer systems", *IEEE Trans. Reliability*, vol. 41, no. 3, pp. 363–377, 1992.

[200] M. Stamatelatos, W. E. Vesely, J. Dugan, J. Fragola, J. Minarick, and J. Railsback, *Fault Tree Handbook with Aerospace Applications*. NASA Office of Safety and Mission Assurance, 2002.

[201] M. Čepin, "Chapter 5. fault tree analysis", in *Assessment of Power System Reliability: Methods and Applications*. Springer London, 2011, pp. 61–87.

[202] J. Fussell, E. Aber, and R. Rahl, "On the Quantitative Analysis of Priority-AND Failure Logic", *IEEE Transactions on Reliability*, vol. 25, no. 5, pp. 324–326, 1976.

[203] D. Coppit, K. Sullivan, and J. Dugan, "Formal Semantics of Models for Computational Engineering: A Case Study on Dynamic Fault Trees", in *Proceedings of the 11th International Symposium on Software Reliability Engineering*, 2000, pp. 270–282.

[204] M. Čepin and B. Mavko, "A dynamic fault tree", *Reliability Engineering & System Safety*, vol. 75, no. 1, pp. 83–91, 2002.

[205] G. K. Palshikar, "Temporal fault trees", *Information and Software Technology*, vol. 44, no. 3, pp. 137–150, 2002.

[206] M. Bouissou and J.-L. Bon, "A new formalism that combines advantages of fault-trees and markov models: boolean logic driven markov processes", *Reliability Engineering & System Safety*, vol. 82, no. 2, pp. 149–163, 2003.

[207] A. Lefebvre, Z. Simeu-Abazi, J.-P. Derain, and M. Glade, "Diagnostic of the avionic equipment based on dynamic fault tree", in *International Conference on Cost Effective Automation in Networked Product Development and Manufacturing*, 2007, p. 12.

[208] M. Walker and Y. Papadopoulos, "Pandora: the time of priority-and gates", *IFAC Proceedings Volumes*, vol. 39, no. 3, pp. 237–242, 2006.

[209] M. Walker and Y. Papadopoulos, "Pandora 2: the time of priority-or gates", *IFAC Proceedings Volumes*, vol. 40, no. 6, pp. 25–30, 2007.

[210] J. Xiang, F. Machida, K. Tadano, K. Yanoo, W. Sun, and Y. Maeno, "Combinatorial analysis of dynamic fault trees with priority-and gates", in *Proceedings of the 23rd International Symposium on Software Reliability Engineering Workshops*, IEEE, 2012, pp. 3–4.

[211] J. Xiang, F. Machida, K. Tadano, K. Yanoo, W. Sun, and Y. Maeno, "A static analysis of dynamic fault trees with priority-and gates", in *Proceedings of the Sixth Latin-American Symposium on Dependable Computing*, IEEE, 2013, pp. 58–67.

[212] Z. Tang and J. B. Dugan, "Minimal cut set/sequence generation for dynamic fault trees", in *Proceedings of the 2004 Annual Symposium Reliability and Maintainability*, IEEE, 2004, pp. 207–213.

[213] S.-I. Minato, "Zero-suppressed BDDs and their applications", *International Journal on Software Tools for Technology Transfer*, vol. 3, no. 2, pp. 156–170, 2001.

[214] D. Liu, W. Xing, C. Zhang, R. Li, and H. Li, "Cut Sequence Set Generation for Fault Tree Analysis", *Embedded Software and Systems*, pp. 592–603, 2007.

[215] H.-L. Zhang, C.-Y. Zhang, D. Liu, and R. Li, "A Method of Quantitative Analysis for Dynamic Fault Tree", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 2011, pp. 1–6.

[216] G. Merle, "Algebraic modelling of dynamic fault trees, contribution to qualitative and quantitative analysis", Ph.D. dissertation, Ecole normale supérieure de Cachan, 2010.

[217] A. B. Rauzy, "Sequence algebra, sequence decision diagrams and dynamic fault trees", *Reliability Engineering & System Safety*, vol. 96, no. 7, pp. 785–792, 2011.

[218] M. Walker and Y. Papadopoulos, "Qualitative temporal analysis: towards a full implementation of the fault tree handbook", *Control Engineering Practice*, vol. 17, no. 10, pp. 1115–1125, 2009.

[219] P.-Y. Piriou, J.-M. Faure, and J.-J. Lesage, "Finding the minimal cut sequences of dynamic, repairable, and reconfigurable systems from generalized boolean logic driven markov process models", *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 236, no. 1, pp. 209–220, 2022.

[220] D. Liu, W. Xiong, L. Zhi, P. Wang, and H. Zhang, "The Simplificafion of Cut Sequence Set Analysis for Dynamic Systems", in *Proceedings of the 2nd International Conference on Computer and Automation Engineering, ICCAE 2010*, vol. 3, 2010, pp. 140–144.

[221] L. Yi, W. Bo, L. Dong, Y. Haitao, and Y. Fande, "Complete Temporal Rules for Cut Sequence Generation in Dynamic Fault Tree Analysis", in *Proceedings of the World Congress of Engineering, WCE 2013*, 2013, pp. 122–128.

[222] D. Ge, R. Zhang, Q. Chou, and Y. Yang, "Probabilistic model–based multi-integration formulas for quantifying a generalized minimal cut sequence", *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 229, no. 1, pp. 73–82, 2015.

[223] G. Merle, J.-M. Roussel, and J.-J. Lesage, "Algebraic determination of the structure function of Dynamic Fault Trees", *Reliability Engineering & System Safety*, vol. 96, no. 2, pp. 267–277, 2011.

[224] M. Shrestha, L. Xing, and H. Xu, "Complete Sequence Generation Algorithm for Reliability Analysis of Dynamic Systems with Sequence-Dependent Failures", in *Proceeding of the 16th International Conference on Reliability and Quality in Design*, ISSAT, 2010, pp. 382–386.

[225] L. Xing, A. Shrestha, and Y. Dai, "Exact combinatorial reliability analysis of dynamic systems with sequence-dependent failures", *Reliability Engineering & System Safety*, vol. 96, no. 10, pp. 1375–1385, 2011.

[226] L. Xing, O. Tannous, and J. B. Dugan, "Reliability Analysis of Nonrepairable Cold-Standby Systems Using Sequential Binary Decision Diagrams", *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 42, no. 3, pp. 715–726, 2012.

[227] D. Ge and Y. Yang, "Reliability analysis of non-repairable systems modeled by dynamic fault trees with priority AND gates", *Applied Stochastic Models in Business and Industry*, vol. 31, no. 6, pp. 809–822, 2015.

[228] W. Long, Y. Sato, and M. Horigome, "Quantification of sequential failure logic for fault tree analysis", *Reliability Engineering & System Safety*, vol. 67, no. 3, pp. 269–274, 2000.

[229] D. Liu, C. Zhang, W. Xing, R. Li, and H. Li, "Quantification of cut sequence set for fault tree analysis", in *International Conference on High Performance Computing and Communications*, Springer, 2007, pp. 755–765.

[230] T. Yuge and S. Yanagi, "Quantitative analysis of a fault tree with priority AND gates", *Reliability Engineering & System Safety*, vol. 93, no. 11, pp. 1577–1583, 2008.

[231] G. Merle, J.-M. Roussel, and J.-J. Lesage, "Quantitative Analysis of Dynamic Fault Trees Based on the Structure Function", *Quality and Reliability Engineering International*, vol. 30, no. 1, pp. 143–156, 2014.

[232] H. Boudali, P. Crouzen, and M. Stoelinga, "Dynamic fault tree analysis using input/output interactive Markov chains", in *Proceedings of the 37th International Conference on Dependable Systems and Networks, DSN 2007*, IEEE, 2007, pp. 708–717.

[233] H. Boudali, P. Crouzen, and M. Stoelinga, "A compositional semantics for Dynamic Fault Trees in terms of Interactive Markov Chains", in *International Symposium on Automated Technology for Verification and Analysis*, Springer, 2007, pp. 441–456.

[234] F. Arnold, A. Belinfante, F. Van der Berg, D. Guck, and M. Stoelinga, "DFTCalc: a tool for efficient fault tree analysis", in *32nd International Conference on Computer Safety, Reliability and Security (SAFECOMP'13)*, ser. LNCS, vol. 8153, Springer, 2013, pp. 293–301.

[235] O. Yevkin, "An efficient approximate markov chain method in dynamic fault tree analysis", *Quality and Reliability Engineering International*, vol. 32, no. 4, pp. 1509–1520, 2016.

[236] R. Manian, D. W. Coppit, K. J. Sullivan, and J. B. Dugan, "Bridging the gap between systems and dynamic fault tree models", in *Proceedings of the Annual Reliability and Maintainability*, IEEE, 1999, pp. 105–111.

[237] L. L. Pullum and J. B. Dugan, "Fault tree models for the analysis of complex computer-based systems", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 1996, pp. 200–207.

[238] R. Gulati and J. B. Dugan, "A Modular Approach for Analyzing Static and Dynamic Fault Trees", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 1997, pp. 57–63.

[239] S. Amari, G. Dill, and E. Howald, "A new approach to solve dynamic fault trees", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 2003, pp. 374–379.

[240] C.-Y. Huang and Y.-R. Chang, "An improved decomposition scheme for assessing the reliability of embedded systems by using dynamic fault trees", *Reliability Engineering & System Safety*, vol. 92, no. 10, pp. 1403–1412, 2007.

[241] O. Yevkin, "An improved modular approach for dynamic fault tree analysis", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 2011, pp. 1–5.

[242] W. Han, W. Guo, and Z. Hou, "Research on the method of dynamic fault tree analysis", in *The Proceedings of 2011 9th International Conference on Reliability, Maintainability and Safety*, IEEE, 2011, pp. 950–953.

[243] T. Liu and S. Chiou, "The application of petri nets to failure analysis", *Reliability Engineering & System Safety*, vol. 57, no. 2, pp. 129–142, 1997.

[244] A. Bobbio, E. Ciancamerla, G. Franceschinis, R. Gaeta, M. Minichino, and L. Portinale, "Sequential application of heterogeneous models for the safetyanalysis of a control system: a case study", *Reliability Engineering & System Safety*, vol. 81, no. 3, pp. 269–280, 2003.

[245] A. Bobbio and D. Codetta-Raiteri, "Parametric fault trees with dynamic gates and repair boxes", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 2004, pp. 459–465.

[246] D. Codetta-Raiteri, "The conversion of dynamic fault trees to stochastic petri nets, as a case of graph transformation", *Electronic Notes in Theoretical Computer Science*, vol. 127, no. 2, pp. 45–60, 2005.

[247] X. Zhang, Q. Miao, X. Fan, and D. Wang, "Dynamic fault tree analysis based on petri nets", in *Proceedings of the 8th International Conference on Reliability, Maintainability and Safety*, IEEE, 2009, pp. 138–142.

[248] L. Herscheid and P. Tröger, "Specification of dynamic fault tree concepts with stochastic petri nets", in *Proceedings of the 8th International Conference on Software Security and Reliability (SERE)*, IEEE, 2014, pp. 177–186.

[249] S. Junges, J.-P. Katoen, M. Stoelinga, and M. Volk, "One net fits all", in *Proceedings of the International Conference on Applications and Theory of Petri Nets and Concurrency*, Springer, 2018, pp. 272–293.

[250] S. Kabir, M. Walker, and Y. Papadopoulos, "Quantitative evaluation of Pandora Temporal Fault Trees via Petri Nets", *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 458–463, 2015.

[251] H. Boudali and J. B. Dugan, "A discrete-time Bayesian network reliability modeling and analysis framework", *Reliability Engineering & System Safety*, vol. 87, no. 3, pp. 337–349, 2005.

[252] S. Montani, L. Portinale, A. Bobbio, and D. Codetta-Raiteri, "Radyban: A tool for reliability analysis of dynamic fault trees through conversion into dynamic Bayesian networks", *Reliability Engineering & System Safety*, vol. 93, no. 7, pp. 922–932, 2008.

[253] H. Boudali and J. B. Dugan, "A continuous-time Bayesian network reliability modeling, and analysis framework", *IEEE Transactions on Reliability*, vol. 55, no. 1, pp. 86–97, 2006.

[254] H. Boudali and J. B. Dugan, "Corrections on "A Continuous-Time Bayesian Network Reliability Modeling and Analysis Framework"", *IEEE Transactions on Reliability*, vol. 57, no. 3, pp. 532–533, 2008.

[255] D. Codetta-Raiteri, "Applying generalized continuous time bayesian networks to a reliability case study", *IFAC-PapersOnLine*, vol. 48, no. 21, pp. 676–681, 2015.

[256] D. Codetta-Raiteri and L. Portinale, "Generalized continuous time bayesian networks as a modelling and analysis formalism for dependable systems", *Reliability Engineering & System Safety*, vol. 167, pp. 639–651, 2017.

[257] S. Kabir, M. Walker, and Y. Papadopoulos, "Reliability analysis of dynamic systems by translating temporal fault trees into bayesian networks", in *Proceedings of the International Symposium on Model-Based Safety and Assessmemt*, Springer, 2014, pp. 96–109.

[258] J. Faulin, A. A. Juan, S. Martorell, and J.-E. Ramirez-Marquez, *Simulation methods for reliability and availability of complex systems*. Springer Science & Business Media, 2010.

[259] E. Zio, *The Monte Carlo simulation method for system reliability and risk analysis*. Springer, 2013.

[260] M. A. Boyd and S. J. Bavuso, "Simulation modeling for long duration spacecraft control systems", in *Reliability and Maintainability Symposium, 1993. Proceedings., Annual*, IEEE, 1993, pp. 106–113.

[261] R. Manian, J. B. Dugan, D. Coppit, and K. J. Sullivan, "Combining various solution techniques for dynamic fault tree analysis of computer systems", in *High-Assurance Systems Engineering Symposium, 1998. Proceedings. Third IEEE International*, IEEE, 1998, pp. 21–28.

[262] H. Boudali, A. P. Nijmeijer, and M. I. A. Stoelinga, "DFTSim: A simulation tool for extended dynamic fault trees", in *Proceedings of the 2009 Spring Simulation Multiconference*, Society for Computer Simulation International, 2009, p. 31.

[263] K. Durga Rao, V. Gopika, V. Sanyasi Rao, H. Kushwaha, A. Verma, and A. Srividya, "Dynamic fault tree analysis using Monte Carlo simulation in probabilistic safety assessment", *Reliability Engineering & System Safety*, vol. 94, no. 4, pp. 872–883, 2009.

[264] G. Manno, F. Chiacchio, L. Compagno, D. D'Urso, and N. Trapani, "MatCarloRe: An integrated FT and Monte Carlo Simulink tool for the reliability assessment of dynamic fault tree", *Expert Systems with Applications*, vol. 39, no. 12, pp. 10 334–10 342, 2012.

[265] G. Manno, F. Chiacchio, L. Compagno, D. D'Urso, and N. Trapani, "Conception of Repairable Dynamic Fault Trees and resolution by the use of RAATSS, a Matlab® toolbox based on the ATS formalism", *Reliability Engineering & System Safety*, vol. 121, pp. 250–262, 2014.

[266] O. Yevkin, "An improved Monte Carlo method in fault tree analysis", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 2010, pp. 1–5.

[267] G. Merle, J.-M. Roussel, J.-J. Lesage, V. Perchet, and N. Vayatis, "Quantitative Analysis of Dynamic Fault Trees Based on the Coupling of Structure Functions and Monte Carlo Simulation", *Quality and Reliability Engineering International*, vol. 32, no. 1, pp. 7–18, 2014.

[268] D. Ge, D. Li, M. Lin, and Y.-H. Yang, "SFRs-based numerical simulation for the reliability of highly-coupled DFTS", *Eksploatacja i Niezawodnosc – Maintenance and Reliability*, vol. 17, no. 2, pp. 199–206, 2015.

[269] L. J. Aslett, T. Nagapetyan, and S. J. Vollmer, "Multilevel Monte Carlo for Reliability Theory", *Reliability Engineering & System Safety*, vol. 165, no. Supplement C, pp. 188 –196, 2017.

[270] S. Kabir, K. Aslansefat, I. Sorokos, Y. Papadopoulos, and S. Konur, "A hybrid modular approach for dynamic fault tree analysis", *IEEE Access*, vol. 8, pp. 97 175–97 188, 2020.

[271] A. Ejlali and S. G. Miremadi, "FPGA-based Monte Carlo simulation for fault tree analysis", *Microelectronics Reliability*, vol. 44, no. 6, pp. 1017–1028, 2004.

[272] H. Aghassi and F. Aghassi, "Fault tree analysis speed-up with gpu parallel computing", *Computer Information Systems and Industrial Management Applications, Tehran*, vol. 5, pp. 106–114, 2012.

[273] H. Aliee and H. R. Zarandi, "A fast and accurate fault tree analysis based on stochastic logic implemented on field-programmable gate arrays", *IEEE Transactions on Reliability*, vol. 62, no. 1, pp. 13–22, 2013.

[274] G. Manno, "Reliability modelling of complex systems: an adaptive transition system approach to match accuracy and efficiency", Ph.D. dissertation, Università di Catania, 2012.

[275] A. Ejlali and S.-G. Miremadi, "Time-to-failure tree", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 2003, pp. 148–152.

[276] E. Ruijters and M. Stoelinga, "Fault tree analysis: a survey of the state-of-the-art in modeling, analysis and tools", *Computer science review*, vol. 15, pp. 29–62, 2015.

[277] S. Kabir, "An overview of fault tree analysis and its application in model based dependability analysis", *Expert Systems with Applications*, vol. 77, pp. 114–135, 2017.

[278] K. Aslansefat, S. Kabir, Y. Gheraibia, and Y. Papadopoulos, "Chapter 4. dynamic fault tree analysis: state-of-the-art in modeling, analysis, and tools", in *Reliability Management and Engineering: Challenges and Future Trends*. CRC Press, 2020, pp. 73–112.

[279] E. Gascard and Z. Simeu-Abazi, "Failure Root Causes Analysis of Complex Systems – Dynamic Fault Tree Approach", in *Proceedings of the 25th European Safety and Reliability Conference (ESREL 2015)*, CRC Press, 2015, pp. 695–703.

[280] E. Gascard, Z. Simeu-Abazi, and Y. Sidqui, "Quantitative analysis of Dynamic Fault Tree by probabilistic approach", in *Proceedings of the 25th European Safety and Reliability Conference (ESREL 2015)*, CRC Press, 2015, pp. 735–743.

[281] E. Gascard and Z. Simeu-Abazi, "Quantitative Analysis of Dynamic Fault Trees by means of Monte Carlo Simulations: Event-Driven Simulation Approach", *Reliability Engineering & System Safety*, vol. 180, pp. 487–504, 2018.

[282] E. Gascard, Z. Simeu-Abazi, and J. Younes, "Exploitation of Built in test for diagnosis by using Dynamic Fault Trees: Implementation in Matlab Simulink", in *Proceedings of the 20th European Safety and Reliability Conference, ESREL 2011*, CRC Press, 2011, pp. 436–444.

[283] M. Paradies and D. Busch, "Root cause analysis at savannah river plant (nuclear power station)", in *Proceedings of 4th IEEE Conference on Human Factors and Power Plants*, 1988, pp. 479–483.

[284] A. D. Livingston, G. Jackson, and K. Priestley, *Root cause analysis: improving performance for bottom-line results*. HSE Books, 2011.

[285] G. Zwingelstein, "Méthodes d'analyse des causes racines des défaillances", *Techniques de l'ingénieur*, 2018.

[286] S. Junges, D. Guck, J.-P. Katoen, A. Rensink, and M. Stoelinga, "Fault trees on a diet: automated reduction by graph rewriting", *Formal aspects of computing*, vol. 29, no. 4, pp. 651–703, 2017.

[287] K. J. Sullivan, J. B. Dugan, and D. Coppit, "The galileo fault tree analysis tool", in *Proceedings of the 29th Annual International Symposium on Fault-Tolerant Computing*, IEEE, 1999, pp. 232–235.

[288] F. Chiacchio, L. Compagno, D. D'Urso, G. Manno, and N. Trapani, "An open-source application to model and solve dynamic fault tree of real industrial systems", in *Proceedings of the 5th International Conference on Software, Knowledge Information, Industrial Management and Applications (SKIMA)*, IEEE, 2011, pp. 1–8.

[289] J. B. Dugan, B. Venkataraman, and R. Gulati, "Diftree: a software package for the analysis of dynamic fault tree models", in *Proceedings of the Annual Reliability and Maintainability Symposium*, IEEE, 1997, pp. 64–70.

[290] M. D. Walker, "Pandora: a logic for the qualitative analysis of temporal fault trees", Ph.D. dissertation, The University of Hull, 2009.

[291] T. Prosvirnova, M. Batteux, P.-A. Brameret, *et al.*, "The altarica 3.0 project for model-based safety assessment", *IFAC Proceedings Volumes*, vol. 46, no. 22, pp. 127–132, 2013.

[292] A. Rauzy and C. Blériot-Fabre, "Towards a sound semantics for dynamic fault trees", *Reliability Engineering & System Safety*, vol. 142, pp. 184–191, 2015.

[293] A. Lefebvre, "Contribution à l'amélioration de la testabilité et du diagnostic de systèmes complexes : application aux systèmes avioniques", Ph.D. dissertation, University Joseph Fourier - Grenoble (FRANCE), 2009.

[294] G. Rubino and B. Tuffin, *Rare event simulation using Monte Carlo methods*. John Wiley & Sons, 2009.

[295] M. Bevilacqua and M. Braglia, "The analytic hierarchy process applied to maintenance strategy selection", *Reliability Engineering & System Safety*, vol. 70, no. 1, pp. 71–83, 2000.

[296] D. S. Thomas, *The costs and benefits of advanced maintenance in manufacturing*. US Department of Commerce, National Institute of Standards and Technology, 2018.

[297] H. Wang, "A survey of maintenance policies of deteriorating systems", *European journal of operational research*, vol. 139, no. 3, pp. 469–489, 2002.

[298] Z. Simeu-Abazi and A. Alali Alhouaij, "Optimisation of distributed maintenance: modelling and application to the multi-factory production", *Reliability Engineering & System Safety*, vol. 96, no. 11, pp. 1564–1575, 2011.

[299] G. Zwingelstein, *La maintenance basée sur la fiabilité: guide pratique d'application de la RCM*. Hermès, 1996.

[300] F. Monchy and J.-P. Vernier, *Maintenance: Méthodes et organisations*. Dunod Paris, 2000.

[301] D. Bouami, *Le grand livre de la maintenance: concepts, démarches, méthodes, outils et techniques*. Afnor éditions, 2019.

[302] A. Banerjee and B. B. Flynn, "A simulation study of some maintenance policies in a group technology shop", *International Journal of Production Research*, vol. 25, no. 11, pp. 1595–1609, 1987.

[303] H. Löfsten, "Management of industrial maintenance–economic evaluation of maintenance policies", *International Journal of Operations & Production Management*, 1999.

[304] E. Zio and M. Compare, "Evaluating maintenance policies by quantitative modeling and analysis", *Reliability Engineering & System Safety*, vol. 109, pp. 53–65, 2013.

[305] S.-H. Ding and S. Kamaruddin, "Maintenance policy optimization—literature review and directions", *The International Journal of Advanced Manufacturing Technology*, vol. 76, no. 5, pp. 1263–1283, 2015.

[306] B. de Jonge and P. A. Scarf, "A review on maintenance optimization", *European journal of operational research*, vol. 285, no. 3, pp. 805–824, 2020.

[307] A. Sleptchenko, M. C. van der Heijden, and A. van Harten, "Trade-off between inventory and repair capacity in spare part networks", *Journal of the Operational Research Society*, vol. 54, no. 3, pp. 263–272, 2003.

[308] A. Sleptchenko, H. H. Turan, S. Pokharel, and T. Y. ElMekkawy, "Cross-training policies for repair shops with spare part inventories", *International Journal of Production Economics*, vol. 209, pp. 334–345, 2019.

[309] M. Driessen, G.-J. van Houtum, W. H. M. Zijm, and W. Rustenburg, "Capacity assignment in repair shops with high material uncertainty", *International journal of production economics*, vol. 221, p. 107 484, 2020.

[310] A. Al-Refaie, H. Al-Shalaldeh, and N. Lepkova, "Proposed procedure for optimal maintenance scheduling under emergent failures", *Journal of Civil Engineering and Management*, vol. 26, no. 4, pp. 396–409, 2020.

[311] A. Z. Kabir and A. S. Al-Olayan, "A stocking policy for spare part provisioning under age based preventive replacement", *European journal of operational research*, vol. 90, no. 1, pp. 171–181, 1996.

[312] M. A. Ilgin and S. Tunali, "Joint optimization of spare parts inventory and maintenance policies using genetic algorithms", *The International Journal of Advanced Manufacturing Technology*, vol. 34, no. 5, pp. 594–604, 2007.

[313] A. Van Horenbeek, J. Buré, D. Cattrysse, L. Pintelon, and P. Vansteenwegen, "Joint maintenance and inventory optimization systems: a review", *International Journal of Production Economics*, vol. 143, no. 2, pp. 499–508, 2013.

[314] J. Poppe, R. J. Basten, R. N. Boute, and M. R. Lambrecht, "Numerical study of inventory management under various maintenance policies", *Reliability Engineering & System Safety*, vol. 168, pp. 262–273, 2017.

[315] J. C. García-Benito and M.-L. Martín-Peña, "A redistribution model with minimum backorders of spare parts: a proposal for the defence sector", *European Journal of Operational Research*, vol. 291, no. 1, pp. 178–193, 2021.

[316] C. M. F. Lapa, C. M. N. Pereira, and M. P. de Barros, "A model for preventive maintenance planning by genetic algorithms based in cost and reliability", *Reliability Engineering & System Safety*, vol. 91, no. 2, pp. 233–240, 2006.

[317] I. Ayadi, L. Bouillaut, P. Aknin, and P. Siarry, "Optimisation par algorithme génétique de la maintenance préventive dans un contexte de modélisation par modèles graphiques probabilistes", in *Proceedings 17ème Congrès de Maîtrise des Risques et de Sûreté de Fonctionnement*, 2010.

[318] R. Bris, E. Châtelet, and F. Yalaoui, "New method to minimize the preventive maintenance cost of series–parallel systems", *Reliability engineering & system safety*, vol. 82, no. 3, pp. 247–255, 2003.

[319] M. Samrout, F. Yalaoui, E. Châtelet, and N. Chebbo, "New methods to minimize the preventive maintenance cost of series–parallel systems using ant colony optimization", *Reliability Engineering & System Safety*, vol. 89, no. 3, pp. 346–354, 2005.

[320] M. Doostparast, F. Kolahan, and M. Doostparast, "Optimisation of pm scheduling for multi-component systems–a simulated annealing approach", *International Journal of Systems Science*, vol. 46, no. 7, pp. 1199–1207, 2015.

[321] L. H. Lee, K. C. Tan, K. Ou, and Y. H. Chew, "Vehicle capacity planning system: a case study on vehicle routing problem with time windows", *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 33, no. 2, pp. 169–178, 2003.

[322] R Bolanos, J Escobar, and M Echeverri, "A metaheuristic algorithm for the multi-depot vehicle routing problem with heterogeneous fleet", *International Journal of Industrial Engineering Computations*, vol. 9, no. 4, pp. 461–478, 2018.

[323] G. Nagy and S. Salhi, "Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries", *European journal of operational research*, vol. 162, no. 1, pp. 126–141, 2005.

[324] R. Abbou, "Contribution à la mise en oeuvre d'une maintenance centralisée: conception et optimisation d'un atelier de maintenance", Ph.D. dissertation, Université Joseph-Fourier-Grenoble I, 2003.

[325] A. Alhouaij Alali, "Contribution à l'optimisation de la maintenance dans un contexte distribué", Ph.D. dissertation, Institut National Polytechnique de Grenoble-INPG, 2010.

[326] Z. Simeu-Abazi, M. Di Mascolo, and E. Gascard, "Performance Evaluation of Centralized Maintenance Workshop by using Queuing Networks", in *Proceedings of the 2nd Workshop on Advanced Maintenance Engineering, Service and Technology (AMEST 2012)*, 2012.

[327] Z. Simeu-Abazi, M. Di Mascolo, and E. Gascard, "Queuing network-based methodology for designing and assessing performance of centralized maintenance workshops", *Journal of Manufacturing Technology Management*, vol. 25, no. 4, pp. 510–527, 2014.

[328] Z. Simeu-Abazi and E. Gascard, "Implementation of a cost optimization algorithm in a context of distributed maintenance", in *Proceedings of the 4th International Conference on Control, Automation and Diagnosis (ICCAD'20)*, IEEE, 2020, pp. 1–6.

[329] A. K. Erlang, "The theory of probabilities and telephone conversations", *Nyt. Tidsskr. Mat. Ser. B*, vol. 20, pp. 33–39, 1909.

[330] A. K. Erlang, "Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges", *Post Office Electrical Engineer's Journal*, vol. 10, pp. 189–197, 1917.

[331] J. R. Jackson, "Jobshop-like queueing systems", *Management science*, vol. 10, no. 1, pp. 131–142, 1963.

[332] M. D. Mascolo, Y. Frein, and Y. Dallery, "An analytical method for performance evaluation of kanban controlled production systems", *Operations Research*, vol. 44, no. 1, pp. 50–64, 1996.

[333] S. Balsamo, "Queueing networks with blocking: analysis, solution algorithms and properties", in *Network performance engineering*, Springer, 2011, pp. 233–257.

[334] A. Bušić, S. Durand, B. Gaujal, and F. Perronnin, "Perfect sampling of jackson queueing networks", *Queueing Systems*, vol. 80, no. 3, pp. 223–260, 2015.

[335] C. Rovetta, "Simulation parfaite de réseaux fermés de files d'attente et génération aléatoire de structures combinatoires", Ph.D. dissertation, Paris Sciences et Lettres (ComUE), 2017.

[336] P. Siarry, *Métaheuristiques*. Editions Eyrolles, 2014.

[337] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[338] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing", *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[339] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines", *The journal of chemical physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[340] I. 13381-1, *Condition Monitoring and Diagnostics of Machines – Prognostics – Part 1: General guidelines*. International Organization for Standardization, 2015.

[341] K. Ranasinghe, R. Sabatini, A. Gardi, *et al.*, "Advances in integrated system health management for mission-essential and safety-critical aerospace applications", *Progress in Aerospace Sciences*, vol. 128, 2022.

[342] N.-H. Kim, D. An, and J.-H. Choi, *Prognostics and health management of engineering systems*. Springer, 2017.

[343] R. Gouriveau, K. Medjaher, and N. Zerhouni, *Du concept de PHM à la maintenance prédictive 1: Surveillance et pronostic*. ISTE Group, 2017, vol. 3.

[344] G. Zwingelstein, "Méthodes hybrides de diagnostic et de pronostic", *Techniques de l'ingénieur*, 2021.

[345] C. S. Byington, M. J. Roemer, and T. Galie, "Prognostic enhancements to diagnostic systems for improved condition-based maintenance", in *Proceedings, IEEE aerospace conference*, IEEE, vol. 6, 2002, pp. 6–6.

[346] A. K. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance", *Mechanical systems and signal processing*, vol. 20, no. 7, pp. 1483–1510, 2006.

[347] A. Heng, S. Zhang, A. C. Tan, and J. Mathew, "Rotating machinery prognostics: state of the art, challenges and opportunities", *Mechanical systems and signal processing*, vol. 23, no. 3, pp. 724–739, 2009.

[348] O. E. Dragomir, R. Gouriveau, F. Dragomir, E. Minca, and N. Zerhouni, "Review of prognostic problem in condition-based maintenance", in *2009 European Control Conference (ECC)*, IEEE, 2009, pp. 1587–1592.

[349] Y. Peng, M. Dong, and M. J. Zuo, "Current status of machine prognostics in condition-based maintenance: a review", *The International Journal of Advanced Manufacturing Technology*, vol. 50, no. 1, pp. 297–313, 2010.

[350] J. Z. Sikorska, M. Hodkiewicz, and L. Ma, "Prognostic modelling options for remaining useful life estimation by industry", *Mechanical systems and signal processing*, vol. 25, no. 5, pp. 1803–1836, 2011.

[351] E. Zio, "Prognostics and health management of industrial equipment", *Diagnostics and prognostics of engineering systems: methods and techniques*, pp. 333–356, 2013.

[352] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel, "Prognostics and health management design for rotary machinery systems—reviews, methodology and applications", *Mechanical systems and signal processing*, vol. 42, no. 1-2, pp. 314–334, 2014.

[353] H. M. Elattar, H. K. Elminir, and A. Riad, "Prognostics: a literature review", *Complex & Intelligent Systems*, vol. 2, no. 2, pp. 125–154, 2016.

[354] M. Kordestani, M. Saif, M. E. Orchard, R. Razavi-Far, and K. Khorasani, "Failure prognosis and applications—a survey of recent literature", *IEEE transactions on reliability*, vol. 70, no. 2, pp. 728–748, 2019.

[355] K. Shahin, "Modèle graphique probabiliste appliqué au diagnostic de l'état de santé des systèmes, au pronostic et à l'estimation de la durée de vie résiduelle", Ph.D. dissertation, Université de Lorraine, 2020.

[356] O. Blancke, "Développement d'une approche de pronostic pour les équipements complexes permettant l'application de la maintenance prévisionnelle", Ph.D. dissertation, Université du Québec, 2020.

[357] E. Zio, "Prognostics and health management (phm): where are we and where do we (need to) go in theory and practice", *Reliability Engineering & System Safety*, vol. 218, p. 108 119, 2022.

[358] G. Kacprzynski, A Sarlashkar, M. Roemer, A. Hess, and B Hardman, "Predicting remaining life by fusing the physics of failure modeling with diagnostics", *JOM*, vol. 56, no. 3, pp. 29–35, 2004.

[359] N. Bolander, H. Qiu, N. Eklund, E. Hindle, and T. Rosenfeld, "Physics-based remaining useful life prediction for aircraft engine bearing prognosis", in *Annual Conference of the PHM Society*, vol. 1, 2009.

[360] M. Pecht and J. Gu, "Physics-of-failure-based prognostics for electronic products", *Transactions of the Institute of Measurement and Control*, vol. 31, no. 3-4, pp. 309–322, 2009.

[361] M. J. Daigle and K. Goebel, "A model-based prognostics approach applied to pneumatic valves", *International journal of prognostics and health management*, vol. 2, no. 2, pp. 84–99, 2011.

[362] C. S. Kulkarni, G. Biswas, J. R. Celaya, and K. Goebel, "Physics based degradation models for electrolytic capacitor prognostics under thermal overstress conditions", *International Journal of Prognostics and Health Management*, vol. 4, no. 1, 2013.

[363] D. Zhang, P. Baraldi, C. Cadet, N. Yousfi-Steiner, C. Bérenguer, and E. Zio, "An ensemble of models for integrating dependent sources of information for the prognosis of the remaining useful life of proton exchange membrane fuel cells", *Mechanical Systems and Signal Processing*, vol. 124, pp. 479–501, 2019.

[364] K. Medjaher, "Contribution au pronostic de défaillances guidé par des données", Habilitation à diriger des recherches, Université de Franche-Comté, 2014.

[365] F. L. Greitzer and R. A. Pawlowski, "Embedded prognostics health monitoring", in *International instrumentation symposium on embedded health monitoring workshop*, 2002.

[366] W. Wang, "Toward dynamic model-based prognostics for transmission gears", in *Component and Systems Diagnostics, Prognostics, and Health Management II*, SPIE, vol. 4733, 2002, pp. 157–167.

[367] W. Q. Wang, M. F. Golnaraghi, and F. Ismail, "Prognosis of machine health condition using neuro-fuzzy systems", *Mechanical Systems and Signal Processing*, vol. 18, no. 4, pp. 813–831, 2004.

[368] W. Wu, J. Hu, and J. Zhang, "Prognostics of machine health condition using an improved arima-based prediction method", in *Proceedings of the 2nd IEEE Conference on Industrial Electronics and Applications*, IEEE, 2007, pp. 1062–1067.

[369] D. A. Tobon-Mejia, K. Medjaher, N. Zerhouni, and G. Tripot, "A data-driven failure prognostics method based on mixture of gaussians hidden markov models", *IEEE Transactions on reliability*, vol. 61, no. 2, pp. 491–503, 2012.

[370] A. Soualhi, K. Medjaher, and N. Zerhouni, "Bearing health monitoring based on hilbert–huang transform, support vector machine, and regression", *IEEE Transactions on instrumentation and measurement*, vol. 64, no. 1, pp. 52–62, 2014.

[371] K. Javed, R. Gouriveau, and N. Zerhouni, "A new multivariate approach for prognostics based on extreme learning machine and fuzzy clustering", *IEEE transactions on cybernetics*, vol. 45, no. 12, pp. 2626–2639, 2015.

[372] J. Zhang, P. Wang, R. Yan, and R. X. Gao, "Long short-term memory for machine remaining life prediction", *Journal of manufacturing systems*, vol. 48, pp. 78–86, 2018.

[373] Y. Wang, Y. Zhao, and S. Addepalli, "Remaining useful life prediction using deep learning approaches: a review", *Procedia manufacturing*, vol. 49, pp. 81–88, 2020.

[374] K. Medjaher and N. Zerhouni, "Hybrid prognostic method applied to mechatronic systems", *The International Journal of Advanced Manufacturing Technology*, vol. 69, no. 1, pp. 823–834, 2013.

[375] L. Liao and F. Köttig, "Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction", *IEEE Transactions on Reliability*, vol. 63, no. 1, pp. 191–207, 2014.

[376] N. Daroogheh, A. Baniamerian, N. Meskin, and K. Khorasani, "A hybrid prognosis and health monitoring strategy by integrating particle filters and neural networks for gas turbine engines", in *2015 IEEE Conference on Prognostics and Health Management (PHM)*, IEEE, 2015, pp. 1–8.

[377] B. Wang, Y. Lei, N. Li, and N. Li, "A hybrid prognostics approach for estimating remaining useful life of rolling element bearings", *IEEE Transactions on Reliability*, vol. 69, no. 1, pp. 401–412, 2018.

[378] M. Baptista, E. M. Henriques, I. P. de Medeiros, J. P. Malere, C. L. Nascimento Jr, and H. Prendinger, "Remaining useful life estimation in aeronautics: combining data-driven and kalman filtering", *Reliability Engineering & System Safety*, vol. 184, pp. 228–239, 2019.

[379] S. Li, H. Fang, and B. Shi, "Remaining useful life estimation of lithium-ion battery based on interacting multiple model particle filter and support vector regression", *Reliability Engineering & System Safety*, vol. 210, p. 107 542, 2021.

[380] Y. Zang, W. Shangguan, B. Cai, H. Wang, and M. G. Pecht, "Hybrid remaining useful life prediction method. a case study on railway d-cables", *Reliability Engineering & System Safety*, vol. 213, p. 107 746, 2021.

[381] Z. Chen, T. Xia, Y. Li, and E. Pan, "A hybrid prognostic method based on gated recurrent unit network and an adaptive wiener process model considering measurement errors", *Mechanical Systems and Signal Processing*, vol. 158, p. 107 785, 2021.

[382] A. Basia, "Evaluation of the state of the health of li-ion batteries in the context of circular economy", Ph.D. dissertation, Université Grenoble Alpes, 2021.

[383] A. Basia, E. Gascard, Z. Simeu-Abazi, and P. Zwolinski, "Overcoming the Barriers in Diagnostics and Prognostics of the Circular Industrial System by Hidden Markov Model", in *Proceedings of the 3rd International Conference on Control, Automation and Diagnosis (ICCAD'19)*, IEEE Press, 2019, pp. 314–319.

[384] A. Basia, Z. Simeu-Abazi, E. Gascard, and P. Zwolinski, "First step towards the development of a Prognosis Health Management (PHM) System for Li-ion batteries: An FMMEA based approach", in *Proceedings of the 29th European Safety and Reliability Conference (ESREL 2019)*, Research Publishing, Singapore, 2019, pp. 1194–1200.

[385] A. Basia, Z. Simeu-Abazi, E. Gascard, and P. Zwolinski, "State of Health Estimation for Lithium-ion Battery by Incremental Capacity Based ARIMA - SVR Model", in *Proceedings of the 31st European Safety and Reliability Conference (ESREL 2021)*, Research Publishing, Singapore, 2021, pp. 1–6.

[386] A. Basia, Z. Simeu-Abazi, E. Gascard, and P. Zwolinski, "Comparison of data driven algorithms for SoH estimation of Lithium-ion batteries", in *Proceedings of the 5th International Conference on Control, Automation and Diagnosis (ICCAD'21)*, IEEE, 2021, pp. 1–6.

[387] A. Basia, Z. Simeu-Abazi, E. Gascard, and P. Zwolinski, "Review on State of Health estimation methodologies for lithium-ion batteries in the context of circular economy ", *CIRP Journal of Manufacturing Science and Technology*, vol. 32, pp. 517–528, 2021.

[388] Y. Xing, E. W. Ma, K. L. Tsui, and M. Pecht, "Battery management systems in electric and hybrid vehicles", *Energies*, vol. 4, no. 11, pp. 1840–1857, 2011.

[389] D.-I. Stroe and E. Schaltz, "Lithium-ion battery state-of-health estimation using the incremental capacity analysis technique", *IEEE Transactions on Industry Applications*, vol. 56, no. 1, pp. 678–685, 2020.

[390] E. Schaltz, D.-I. Stroe, K. Nørregaard, L. S. Ingvardsen, and A. Christensen, "Incremental capacity analysis applied on electric vehicles for battery state-of-health estimation", *IEEE Transactions on Industry Applications*, vol. 57, no. 2, pp. 1810–1817, 2021.

[391] L. Wang, C. Pan, L. Liu, Y. Cheng, and X. Zhao, "On-board state of health estimation of lifepo4 battery pack through differential voltage analysis", *Applied energy*, vol. 168, pp. 465–472, 2016.

[392] X. Tan, Y. Tan, D. Zhan, *et al.*, "Real-time state-of-health estimation of lithium-ion batteries based on the equivalent internal resistance", *Ieee Access*, vol. 8, pp. 56 811–56 822, 2020.

[393] N. Noura, L. Boulon, and S. Jemeï, "A review of battery state of health estimation methods: hybrid electric vehicle challenges", *World Electric Vehicle Journal*, vol. 11, no. 4, p. 66, 2020.

[394] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition", *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[395] R Chang and J. Hancock, "On receiver structures for channels having memory", *IEEE Transactions on Information Theory*, vol. 12, no. 4, pp. 463–468, 1966.

[396] L. Baum, "An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process", *Inequalities*, vol. 3, pp. 1–8, 1972.

[397] C. M. Bishop, "Pattern recognition and machine learning", in springer, 2006, ch. 13, pp. 605–652.

[398] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm", *IEEE transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[399] G. D. Forney, "The Viterbi algorithm", *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[400] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains", *The annals of mathematical statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[401] L. R. Welch, "Hidden Markov models and the Baum-Welch algorithm", *IEEE Information Theory Society Newsletter*, vol. 53, no. 4, pp. 10–13, 2003.

[402] B.-H. Juang and L. R. Rabiner, "The segmental k-means algorithm for estimating parameters of hidden markov models", *IEEE Transactions on acoustics, speech, and signal Processing*, vol. 38, no. 9, pp. 1639–1641, 1990.

[403] B. Mor, S. Garhwal, and A. Kumar, "A systematic review of hidden markov models and their applications", *Archives of computational methods in engineering*, vol. 28, no. 3, pp. 1429–1448, 2021.

[404] D. A. Tobon-Mejia, K. Medjaher, N. Zerhouni, and G. Tripot, "Hidden markov models for failure diagnostic and prognostic", in *2011 Prognostics and System Health Managment Confernece*, IEEE, 2011, pp. 1–8.

[405] B. Roblès, M. Avila, F. Duculty, P Vrignat, S Bégot, and F Kratz, "Modélisation du niveau de dégradation d'un système industriel à l'aide de modèles de markov cachés", *Congrés Lambda Mu 19 de Maîtrise des Risques et Sûreté de Fonctionnement*, 2014.

[406] P. Vrignat, M. Avila, F. Duculty, and F. Kratz, "Failure event prediction using hidden markov model approaches", *IEEE Transactions on Reliability*, vol. 64, no. 3, pp. 1038–1048, 2015.

[407] T. Aggab, "Pronostic des systèmes complexes par l'utilisation conjointe de modèle de markov caché et d'observateur", Ph.D. dissertation, Université d'Orléans, 2016.

[408] A. Delmas, M. Sallak, W. Schön, and L. Zhao, "Méthodes de prédiction de durée de vie en vue de modèles de maintenance prévisionnelle: calcul d'intervalles et stratégies en présence de données incertaines", in *Congrès Lambda Mu 21 de Maîtrise des risques et et Sûreté de Fonctionnement*, IMdR, Institut pour la Maîtrise des risques, 2018.

[409] Z. Chen, Y. Li, T. Xia, and E. Pan, "Hidden markov model with auto-correlated observations for remaining useful life prediction and optimal maintenance policy", *Reliability Engineering & System Safety*, vol. 184, pp. 123–136, 2019.

[410] S. Yang, C. Zhang, J. Jiang, W. Zhang, L. Zhang, and Y. Wang, "Review on state-of-health of lithium-ion batteries: characterizations, estimations and applications", *Journal of Cleaner Production*, vol. 314, p. 128 015, 2021.

[411] K. Medjaher, D. A. Tobon-Mejia, and N. Zerhouni, "Remaining useful life estimation of critical components with application to bearings", *IEEE Transactions on Reliability*, vol. 61, no. 2, pp. 292–302, 2012.

[412] A. Delmas, "Contribution à l'estimation de la durée de vie résiduelle des systèmes en présence d'incertitudes", Ph.D. dissertation, Université de Technologie de Compiègne, 2019.

[413] X. Tang, C. Zou, K. Yao, *et al.*, "A fast estimation algorithm for lithium-ion battery state of health", *Journal of Power Sources*, vol. 396, pp. 453–458, 2018.

[414] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression", *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[415] M. Awad and R. Khanna, "Support vector regression", in *Efficient learning machines*, Springer, 2015, pp. 67–80.

[416] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[417] S. Makridakis and M. Hibon, "Arma models and the box–jenkins methodology", *Journal of forecasting*, vol. 16, no. 3, pp. 147–163, 1997.

[418] R. Jamil, "Hydroelectricity consumption forecast for pakistan using arima modeling and supply-demand analysis for the year 2030", *Renewable Energy*, vol. 154, pp. 1–10, 2020.

[419] B. Saha and K. Goebel, "Battery data set", *NASA AMES prognostics data repository*, 2007.

[420] O. Cardin, "Classification of cyber-physical production systems applications: proposition of an analysis framework", *Computers in Industry*, vol. 104, pp. 11–21, 2019.

Méthodologies et algorithmes pour l'analyse de la sûreté de fonctionnement des systèmes industriels complexes

Résumé

Inscrits dans les défis des transitions écologiques, numériques et sociétales des systèmes de production industrielle étudiés au laboratoire G-SCOP, les travaux présentés dans cette HDR relèvent de la thématique de la sûreté de fonctionnement et du pronostic et management de la santé des systèmes industriels complexes.

Mes activités de recherche présentées dans cette HDR se structurent en trois axes. Le premier axe porte sur le développement de méthodes de diagnostic avec des approches basées sur les modèles. Nous nous sommes intéressés à définir des méthodologies de modélisation mathématique/informatique qui représentent le fonctionnement de systèmes de production industrielle et à élaborer des méthodes et algorithmes qui permettent de réaliser leur diagnostic. Nous proposons ainsi l'usage de différents formalismes tels que les automates temporisés communicants, les automates hybrides, les bonds graphs et les graphes d'événements temporisés ou en utilisant une modélisation hiérarchique et multi-vues. Le second axe propose de nouveaux outils d'aide à la décision pour le diagnostic et la maintenance en utilisant les arbres de défaillances dynamiques, les réseaux de files d'attente et les métaheuristiques. Le dernier axe se focalise sur les méthodes de pronostic basées sur les données. Nous proposons des méthodologies de diagnostic et de pronostic de santé des systèmes industriels à base de méthodes d'apprentissage issues de l'intelligence artificielle.

Mes projets de recherche proposés dans cette HDR s'inscrivent dans la continuité de ces trois axes de recherche en développant de nouvelles méthodologies de diagnostic et de pronostic de systèmes de production cyber-physiques afin de participer à une transition réussie vers l'industrie du futur (Industrie 4.0). Dans ce sens, mes projets de recherche sont structurés sur trois directions. La première direction est de poursuivre nos travaux sur les méthodes de diagnostic en considérant des systèmes dynamiques complexes dont on possède uniquement une modélisation incomplète. La seconde direction est de compléter nos méthodes d'optimisation de maintenance distribuée par une approche guidée par les données. La dernière direction est d'étendre nos outils d'aide à la décision dans le diagnostic et la maintenance aux systèmes cyber-physiques de production.

Mes ambitions de recherche sont de répondre à l'un des défis majeurs de la révolution industrielle 4.0, la prise en compte de l'humain dans le développement des nouvelles méthodes et technologies pour la fiabilité, le diagnostic et la maintenance des usines intelligentes (smart factory). Il est en effet nécessaire de proposer des solutions pour intégrer l'humain dans cette disruption digitale et l'impliquer au cœur des dynamiques de l'Industrie 4.0. Je propose d'étudier ces questions clés au travers de deux projets de recherche. Dans le premier projet de recherche, nous rechercherons à utiliser la réalité virtuelle / réalité augmentée pour le diagnostic et l'aide à la maintenance d'équipement de production. Dans le second projet, nous développerons un système de surveillance et d'aide au diagnostic de robot collaboratif par son opérateur humain afin de capitaliser l'expérience et l'intelligence humaine et de placer l'opérateur humain au cœur des dynamiques de l'Industrie 4.0.

Mots-clés : Diagnostic,maintenance, pronostic, industrie du futur