Vers des systèmes plus autonomes : contributions autour de la tâche de diagnostic dans une architecture embarquée

Elodie Chanthery

HABILITATION A DIRIGER DES RECHERCHES DE L'UNIVERSITÉ DE TOULOUSE

Délivrée par : *l'Institut National Polytechnique de Toulouse (INP Toulouse)*

Présentée et soutenue le *11/07/2018* **par :**

ELODIE CHANTHERY

Vers des systèmes plus autonomes : contributions autour de la tâche de diagnostic dans une architecture embarquée

---

JURY

VINCENT COCQUEMPOT	Professeur des Universités	Rapporteur
PHILIPPE DAGUE	Professeur des Universités	Rapporteur
DIMITRI LEFEBVRE	Professeur des Universités	Rapporteur
AUDINE SUBIAS	Maître de Conférences HDR	Membre du Jury
LOUISE TRAVÉ-MASSUYÈS	Directrice de Recherches	Référente
JANAN ZAYTOON	Professeur des Universités	Président du Jury

École doctorale et spécialité :
EDSYS : Automatique, Signal, Productique, Robotique 4200046
Unité de Recherche :
LAAS-CNRS (Equipe DISCO)
Référente :
Louise TRAVE-MASSUYES
Rapporteurs :
Vincent COCQUEMPOT, *Philippe DAGUE* et *Dimitri LEFEBVRE*

Table des matières

Remerciements

Je remercie chaleureusement Vincent Cocquempot, Philippe Dague et Dimitri Lefebvre d'avoir accepté de rapporter ce travail et de me faire l'honneur de participer à ce jury. Je suis également très honorée et ravie de la participation de Janan Zaytoon. Je tiens à remercier tout particulièrement Audine Subias qui me fait la joie de représenter l'INSA de Toulouse dans ce jury. Enfin, un immense merci à Louise Travé-Massuyès, qui a accepté d'être ma garante pour cette habilitation à diriger des recherches, et sans qui rien de tout cela n'aurait été possible.

J'ai pleinement conscience de la chance que j'ai eue d'être accueillie dans deux établissements de grande qualité.

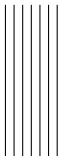
L'INSA tout d'abord, où je peux profiter d'un environnement dynamique et de collègues de confiance avec lesquels il est si agréable de travailler, monter des cours, des TP, des projets, etc. Merci à tous ceux avec qui j'ai pu travailler un jour ou un autre, avec un remerciement tout particulier pour l'équipe d'automatique, qui m'a accordé sa confiance.

Le LAAS-CNRS ensuite : j'ai trouvé dans l'équipe DISCO et au laboratoire en général un lieu où j'ai pu m'épanouir sereinement en recherche. Un grand merci aux membres de l'équipe DISCO. Je ne sais que trop bien que je dois mes travaux à des collaborations fructueuses avec de nombreuses personnes : Louise, Yannick, Audine, Euriell, Carine, Soheib, merci pour toutes ces heures de travail passées (et surtout futures !). Évidemment ce travail n'existerait pas non plus sans les étudiants avec lesquels j'ai eu le plaisir de travailler, douter, râler : mes doctorants Quentin, Gustavo, Saurahb, mais également les post-docs Matthieu, Emmanuel et tous mes stagiaires.

Merci à toutes les personnes, chercheurs, enseignants, enseignants-chercheurs, ingénieurs, techniciens, secrétaires, ... à l'INSA, au LAAS, à l'ENSEEIHT, à l'ONERA en France et ailleurs avec lesquelles j'ai pu travailler et qui se reconnaîtront : je ne veux oublier personne.

Un merci spécial à Pauline, Gwendoline, Pierre-Emmanuel, Guillaume. Évidemment merci pour tous nos moments de travail mais aussi pour tout le reste.

Enfin merci à ma famille, Fabien, Lise, Julie, mes amis : vous êtes mon équilibre. Merci de croire en moi souvent plus que je ne le fais.



À mes grands-parents

Première partie

Présentation générale

1 Curriculum vitæ

Elodie Chanthery

Situation familiale	Mariée, 2 enfants
Née le	6 septembre 1979 à Cenon (33, Gironde)
Téléphone	05 61 55 98 06/ 05 61 33 64 45
Email	elodie.chanthery@laas.fr

Formations et Diplômes

Septembre 2005	**Doctorat**, spécialité "systèmes" de l'Ecole Nationale Supérieure de l'Aéronautique de de l'Espace, SupAéro, Toulouse *Planification de Mission pour un Véhicule Aérien Autonome*
Juin 2002	**Diplôme d'Etudes Approfondies (DEA**) Systèmes Automatiques - Ecole Doctorale SYStèmes, Toulouse, mention très bien *Planification de Mission pour un Véhicule Aérien Autonome*
Juin 2002	**Diplôme d'Ingénieur ENSEEIHT** Filière Génie Electrique et Automatique Spécialisation Automatique et Informatique Industrielle Ecole Nationale Supérieure d'Electrotechnique, d'Electronique, d'Informatique, d'Hydraulique et des Télécommunications, Toulouse
Juin 1997	**Baccalauréat série S - Scientifique** (Scientifique) Mention bien

Cursus professionnel dans l'enseignement supérieur et la recherche

depuis 2006	**Maître de conférences (section 61)**, à l'Institut National des Sciences Appliquées (INSA) de Toulouse Recherches au Laboratoire d'Analyses et d'Architecture des Systèmes (LAAS/CNRS) dans l'équipe Diagnostic Supervision et COnduite (DISCO) Enseignements au Département Génie Electrique et Informatique (DGEI) de l'INSA de Toulouse
2005-2006	**Enseignant-Chercheur** à l'Ecole d'Electricité, de Production et de Méthodes Industrielles (EPMI), Cergy Pontoise, France Recherches rattachées à l'ENSEA, équipe Electronique et Commande des Systèmes (ECS), sur le diagnostic des systèmes hybrides Enseignements à l'EPMI
2002-2005	**Doctorante**, bourse ministérielle Recherches à l'ONERA, The French Aerospace Lab, Département Commande des Systèmes et Dynamique du vol (DSCD) Vacataire à l'ENSEEIHT (96h eq TD d'enseignement par an)

2 Animation scientifique et Encadrements

2.1 Formation à la recherche

La table 2.1 résume mes activités d'encadrement.

Type	Quantité	Encadrement
Post-doctorats	2	100% ; 50 %
Thèses soutenues	2	50% ; 40%
Thèses non soutenues	1	50 %
Master M2	11	50% (7), 33,3% (3), 25% (1)
Master M1	6	50% (4), 40% (1), 33,3% (1)

TABLE 2.1 – Résumé des activités d'encadrement

2.1.1 Encadrements de post-doctorats

M. Godichaud (2009-2010)
- Encadrement : E. Chanthery 100%
- Thème : Placement de capteurs dans un cadre multi-agents
- Publications : 2 articles de conférences (ROADEF 2011, IFAC World Congress 2011)

E. Bénazéra (2008-2011)
- Encadrement : L. Travé-Massuyès 50%, E. Chanthery 50%
- Thème : Lien entre diagnostic et planification sur des systèmes autonomes, utilisation des POMDPs
- Publications : 1 article de conférence (DX2008)

2.1.2 Directions de thèses

Thèses soutenues

C. G. Perez Zuniga (Avril 2014 - Août 2017)
- Encadrement : L. Travé-Massuyès 40%, E. Chanthery 40%, J. Sotomayor Moriano 20%
- Thèse en cotutelle avec l'Université Catholique du Pérou (PUCP)
- Sujet : Analyse Structurelle pour le Diagnostic des Systèmes Distribués
- Publications : 3 articles de conférences (IFAC World Congress 2017, DX 2016, 12th European Workshop on Advanced Control and Diagnosis, 2015), 1 article de journal (Journal of Physics 2015)
- Date de soutenance : 21 août 2017

Q. Gaudel (Octobre 2013 - Octobre 2016)
- Encadrement : E. Chanthery 50%, P. Ribot 50%
- Sujet : Approche intégrée de diagnostic et de pronostic pour la gestion de santé des systèmes hybrides sous incertitude

- Publications : 4 articles de conférences (DX 2014, PHM 2014, International Conference on Application and Theory of Petri Nets and Concurrency 2016, MSR 2015) 1 article de journal (Int Journal of Prognotics and Health Management)

Thèse non soutenue

S. Indra (Novembre 2009 - ...)
- Encadrement : L. Travé-Massuyès 50%, E. Chanthery 50%
- Abandon officiel de la thèse en 2017
- Sujet : Diagnostic réparti pour un satellite autonome
- Publications : 2 articles de conférences (SafeProcess 2012, EUCASS 2011) 1 article de journal (IEEE Transactions on Systems, Man, and Cybernetics : Systems)

2.1.3 Directions de stagiaires

Stages de master 2

M. Girouard (Avril - Septembre 2018)
- Encadrement : E. Chanthery 50%, G. Auriol 50%
- Sujet : Complémentarité tests logiciels/ tests matériels dans une architecture de nano-satellite

O. Bassène (Avril - Septembre 2018)
- Encadrement : E. Chanthery 33,3%, L. Travé-Massuyès 33,3% et C. Artigues 33,3%
- Sujet : Optimisation du placement de capteurs pour le diagnostic : passage à large échelle et robustesse

B.A Bouzidi (Avril - Septembre 2017)
- Encadrement : E. Chanthery 33,3%, P. Ribot 33,3% et F. Teichteil-Koenigsbuch (Airbus Group Innovation) 33,3%
- Sujet : Gestion de santé assistée par des outils à base de modèles et de données

A. Slimani (Avril - Septembre 2017)
- Encadrement : E. Chanthery 33,3%, P. Ribot 33,3% et N. Nedjemi (Altran Sud-Ouest) 33,3%
- Sujet : Fusion de méthodes pour la santé des systèmes

A. Gasmi (Mai - Octobre 2017)
- Encadrement : E. Chanthery 25%, L. Travé-Massuyès 25%, N. Jozevowiez 25% et C. Artigues 25%
- Sujet : Approche structurelle pour la sélection optimale de tests

F. Chatrie (Avril - Septembre 2015)
- Encadrement : E. Chanthery 50%, P. Ribot 50%
- Sujet : Diagnostic d'un système à base de modèles adaptatifs

N. Garin, (Avril - Septembre 2014)
- Encadrement : E. Chanthery 50%, L. Travé-Massuyès 50%
- Sujet : Diagnostic actif par "On Board Control Procedures"

S. Zabi (Avril - Septembre 2013)
- Encadrement : E. Chanthery 50%, P. Ribot 50%
- Sujet : Intégration Diagnostic/Pronostic dans un système hybride.
- Publication : 1 article (PHM 2014)

M. Maïga (Avril - Septembre 2011)
- Encadrement : E. Chanthery 50%, L. Travé-Massuyès 50%
- Sujet : Diagnostic de systèmes hybrides : utilisation de HYDIAG sur un benchmark de la compétition internationale de diagnostic (DXC1001)

- Publication : 1 article (SafeProcess2012)

P-J Meyer (Février- Septembre 2011)
- Encadrement : E. Chanthery 50%, Y. Pencolé 50%
- Sujet : Diagnostic à la demande sur un système à événements discrets

N. Bussac (Février- Septembre 2009)
- Encadrement : E. Chanthery 50%, Y. Pencolé 50%
- Sujet : Diagnostic actif dans une architecture embarquée pour un engin autonome
- Publication : 2 articles (ISAIRAS2010, MSR 2009) et une revue (JESA)

Stages de Master 1

F. Rodrigues Soares (Juin-Septembre 2017)
- Encadrement : E. Chanthery 50%, G. Le Corre 50%
- Sujet : Améliorations de manipulations en automatique : Pendule inversé

A. De Melo (Juin-Septembre 2017)
- Encadrement : E. Chanthery 50%, G. Le Corre 50%
- Sujet : Améliorations de manipulations en automatique : LégoRover

R. Roy (Juin-Septembre 2014)
- Encadrement : P. Cox 33,3%, P-E. Hladik 33,3%, E. Chanthery 33,3%
- Sujet : Simulink Parrot AR Drone Support improvements

M. Mongin (Juin-Septembre 2014)
- Encadrement : E. Chanthery 50%, P. Ribot 50%
- Sujet : Modélisation de l'automate hybride d'une PRV

N. Combes (Juin-Septembre 2014)
- Encadrement : E. Chanthery 40%, P. Ribot 40%, Q. Gaudel 20%
- Sujet : Développement d'une application web liée à un outil de diagnostic

J. Salvy (Juin-Septembre 2010)
- Encadrement : E. Chanthery 50%, Y. Pencolé 50%
- Sujet : Gestion de conflits entre processus par réseaux de Petri

Autres stages

K. Law (Avril-Juin 2016 - Stage de fin de Prépa INP)
- Encadrement : E. Chanthery 50%, L. Travé-Massuyès 50%
- Sujet : Développement d'un démonstrateur 3D de satellite

T. Fouqueray (Avril-Juin 2015 - Stage de fin de Prépa INP)
- Encadrement : E. Chanthery 50%, L. Travé-Massuyès 50%
- Sujet : Génération de RRAs pour un satellite

S. Sahin (Juin-Juillet 2012 - Stage de deuxième année)
- Encadrement : E. Chanthery 50%, G. Le Corre 50%
- Sujet : Améliorations de manipulations de Travaux Pratiques en automatique

S. Diankha (Juin-Juillet 2011 - Stage IUT GEII)
- Encadrement : E. Chanthery 50%, G. Le Corre 50%
- Sujet : Génération automatique de VHDL à partir de machines à états

X. Molles (Avril-Juin 2008 - Stage de fin de Prépa INP)
- Encadrement : E. Chanthery 50%, L. Travé-Massuyès 50%
- Sujet : Interface graphique pour un hotspotter

2.2 Diffusion et partenariats scientifiques

2.2.1 Participation à l'organisation de manifestations scientifiques

Présidente de session lors de conférences : SafeProcess 2008

IFAC 2017 World Congress
- Membre de l'organisation (gestion des bénévoles)

Participation à des Comités Internationaux de Programme : DX2016, DX2018

2.2.2 Séminaires invités

- Université Technologique de Troyes : Diagnostic et pronostic des systèmes hybrides
- DAS G2MCO : Model learning, prognosis, diagnosis and planning for heterogeneous systems

2.2.3 Collaborations internationales

- Juan Javier Sotomayor Moriano, Departement d'ingénierie, PUCP Pontificia Universidad Catolica del Peru, **Perou** (3 publications communes)
- Thierry Peynot, Science and Engineering Faculty, Electrical Engineering, Computer Science, Robotics and Autonomous Systems, QUT Queensland University of Technoloy, Brisbane, **Australie** (1 publication commune)
- Matthew Daigle, Diagnostics & Prognostics Group Research Computer Scientist, NASA Ames research Center, National Aeronautics and Space Administration, **USA** (a changé d'emploi en 2017) (1 publication commune, 1 soumission en cours)

Publications associées :

▶ C. G. Pérez, L. Travé-Massuyès, E. Chanthery et J. Sotomayor. *Fault-Driven Structural Diagnosis Approach in a Distributed Context.* In IFAC World Congress, page 1, 2017

▶ C. G. Pérez, E. Chanthery, L. Travé-Massuyès et J. Sotomayor. *Fault-Driven Minimal Structurally Overdetermined Set in a Distributed Context.* In the 27th International Workshop on Principles of Diagnosis: DX-2016, 2016

▶ C. G. Pérez, L. Travé-Massuyès, E. Chanthery et J. Sotomayor. *Decentralized diagnosis in a spacecraft attitude determination and control system.* In Journal of Physics: Conference Series, volume 659-1. IOP Publishing, 2015

▶ Y. Pencolé, E. Chanthery et T. Peynot. *Definition of Model-based diagnosis problems with Altarica.* In 27th International Workshop on Principles of Diagnosis (DX-2016), page 8p., Denver, CO, United States, Octobre 2016

▶ Q. Gaudel, P. Ribot, E. Chanthery et M. J. Daigle. Health Monitoring of a Planetary Rover Using Hybrid Particle Petri Nets, volume 9698 of *Lecture Notes in Computer Science*, chapitre Application and Theory of Petri Nets and Concurrency. PETRI NETS 2016. Lecture Notes in Computer Science, pages 196–215. Springer, Cham; Kordon F., Moldt D. (eds), 2016

▶ Q. Gaudel, P. Ribot, E. Chanthery et M. J. Daigle. *Prognosis of a Planetary Rover Using Hybrid Particle Petri Nets.* Journal of Process Control, soumis en 2018

2.3 Animation scientifique et administration de la recherche

2.3.1 Participation à des instances scientifiques ou administratives

Membre du conseil scientifique du thème Décision et Optimisation (2012-2016)

Membre du comité directeur de l'école doctorale EDSYS (depuis 2017)

Membre de 2 comités de sélection MCF 61ème section (2014 et 2015)

2.3.2 Intervention pour des rapports de lecture de revues et de conférences internationales

- Relecture pour des revues internationales : IEEE Control Systems Letters, Automatica, Journal of Process Control, Part O : Journal of Risk and Reliability
- Relecture pour des conférences internationales : IFAC World congress 2017, the International Conference on Control and Fault-Tolerant (SysTol) Systems, the Mediterranean Conference on Control and Automation, IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes (SafeProcess), the annual conference of the prognostics and health management society (PHM), International Workshop on Principles of Diagnosis, the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IFAC Symposium on System Identification (SYSID), the international Conference on Automated Planning and Scheduling (ICAPS)

3 Responsabilités pédagogiques, Animation et Enseignements

3.1 Investissements en qualité de Maître de Conférences à l'INSA de Toulouse

3.1.1 Animation et responsabilités pédagogiques

2015-2016	Co-animatrice de la réflexion sur l'évolution de la 5ème année Systèmes Embarqués Critiques avec F. Vernadat
2015-2016	Responsable de la 5ème année Systèmes Embarqués Critiques
2013 (semestre 2)	Responsable de la 4ème année Automatique et Electronique
2009-2015	Elue au conseil de département du DGEI, membre de la commission de recrutement des ATER
2008-2011	Responsable de la 4ème année Automatique et Electronique
depuis 2006	Participation aux entretiens de recrutement en 1A et 3A, participation aux jury de recrutement de 3ème année
depuis 2006	Participation aux Journées Portes Ouvertes de l'INSA

3.1.2 Pédagogie active et plurielle

Je me suis particulièrement investie dans le développement de méthodes de pédagogie actives et de dispositifs pédagogiques visant à impliquer plus les étudiants dans leurs propres apprentissages. A ce titre les activités que j'ai développées sont les suivantes :

- Participation au projet **English Medium Instruction at INSA** (EMINSA) pour l'UF "Analyse des systèmes complexes" en 2016-2017. Passage à un enseignement complètement en anglais en septembre 2017 avec toute l'équipe pédagogique de l'UF.
- Suivi de l'atelier de pédagogie DEFI Diversités "Comment mettre en œuvre une évaluation par les pairs et l'utilisation de grilles critériées ?" en 2016. Mise en place de l'**évaluation par les pairs** en TP d'automatique appliquée en 2016-2017 avec G. Le Corre.
- Suivi de l'atelier de pédagogie DEFI Diversités "Capsules vidéos - SPOOC, MOOC" en 2015.
- Mise en place de **quizz** en amphi (années 2, 3, 4) depuis 2014 avec G. Le Corre, utilisation de **iquiz**[1], mis en place par l'Université de Toulouse, depuis septembre 2017.
- Mise en place de quizz avec utilisation de **Plickers**[2] (quizz interactif) depuis 2014 en 1ère année NORGINSA.
- **Evaluation par acquis de l'apprentissage** depuis 2013 avec G. Le Corre.
- Utilisation de **QCM formatifs et certificatifs** sous Moodle depuis 2012 avec G. Le Corre et V. Mahout.

1. https ://iquiz.univ-toulouse.fr/
2. https ://plickers.com/

- Enseignements à distance via la plateforme Moodle en IFCI-CP pour la formation continue entre 2006 et 2016.
- **Pédagogie inversée** en 3ème année en commande numérique (2006-2008) avec V. Mahout.

3.1.3 Enseignements dispensés

Ma thématique d'enseignement à l'INSA de Toulouse porte sur la **modélisation et la commande de systèmes continus et séquentiels**.

La figure 3.1 illustre l'évolution de mon service ces 10 dernières années. La ligne pointillée indique le service dû, la ligne pleine les heures équivalent TD. Entre 2011 et 2014, j'ai eu mon second congé maternité puis une période à temps partiel. La surcharge sur l'année 2015-2016 s'explique par une responsabilité d'année ponctuelle, celle de l'année 2016-2017 par un accroissement du nombre de groupes de TD sur certaines filières que nous avons dû absorber dans nos services.

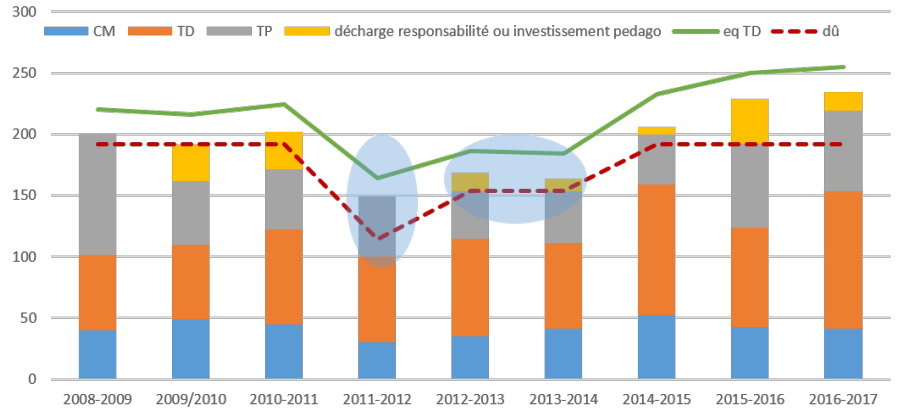


FIGURE 3.1 – Evolution de mes heures d'enseignements au cours des 10 dernières années.

Les sections suivantes présentent le détail de mes enseignements. Les heures indiquées représentent les heures élèves, mon service variant d'une année sur l'autre en fonction du nombre de groupes d'étudiants, des congés maternité, des absences ponctuelles de collègues etc, comme le montre la figure 3.1. Les heures soulignées sont les heures dans lesquelles j'interviens pour tout ou partie. Les Unités de Formation dont j'ai été ou suis encore responsable sont indiquées avec une astérisque.

Principales Unités de Formation dispensées

Systèmes* (depuis 2006)
- Thématique : Introduction à l'automatique et à l'approche systèmes
- Responsable de l'Unité de Formation entre 2015 et 2016
- Niveau : 1A (2006-2015), seulement aux NORGINSA depuis 2015
- Heures élèves : 11,25h CM, 6,25h TD

Systèmes automatiques* (depuis 2006)
- Thématique : Modélisation, analyse et commande de systèmes continus et séquentiels
- Responsable de l'Unité de Formation entre 2009 et 2015, montage des TP

- Niveau : 2A
- Heures élèves : automatique continue 11,25h CM, 7,5h TD ; logique séquentielle 5h CM, 5h TD, 8,25h TP

Informatique matérielle et logique* (depuis 2010)
- Thématique : Structure des ordinateurs, logique combinatoire et séquentielle
- Responsable de l'Unité de Formation entre 2010 et 2015
- Niveau : 2A
- Heures élèves : logique combinatoire : 5h CM, 2,5h TD ; logique séquentielle : 5h CM, 2,5h TD, SFO : 8,25h TP, 15h TD

Automatique : modélisation et commande (depuis 2015)
- Thématique : Modélisation, analyse et commande de systèmes linéaires continus et de systèmes à événements discrets
- Niveau : 3A
- Heures élèves : continu : 18,75h CM, 11,25h TD, 22h TP ; discret : 6,25h CM, 5h TD, 8,25h TP

Analyse des systèmes complexes (depuis 2006)
- Thématique : Systèmes multivariables et systèmes non linéaires
- Niveau : 4A
- Heures élèves : systèmes multivariables : 12,5h CM, 7,5h TD ; systèmes non linéaires : 17,5h CM, 12,5 hTD, 5,5 hTP

Automatique appliquée* (depuis 2006)
- Thématique : Travaux pratiques et mini-projet sur l'automatique de 4A
- Responsable de l'Unité de Formation entre 2008 et 2015
- Niveau : 4A
- Heures élèves : travaux pratiques : 2,5h CM, 16,5h TP ; mini-projet : 16,5h TP

Optimisation des systèmes discrets et continus (depuis 2015)
- Niveau : 4A
- Heures élèves : graphes et PL : 13,75h CM, 12,5h TD ; processus stochastiques, réseau de Petri : 15h CM, 17,5 hTD ; commande optimale : 11,25h CM, 8,75h TD.

Projet d'intégration systèmes embarqués critiques I et II* [3] (depuis 2013)
- Responsable d'une Unité de Formation depuis 2015, montage de la nouvelle plateforme pédagogique (voiture autonome)
- Niveau : 5A Systèmes Embarqués Critiques
- Heures élèves : management de projet : 15h CM, projet 66,25h TP ; anglais : 35h TD

Autres enseignements
- Processus stochastiques et filtrage (2006-2008) - 5A (CM, TD)
- Automatique dans l'espace d'état - Enseignement en présentiel et à distance - IFCI CP (CM, TD)
- Commande numérique (2006-2008) 3A (CM, TD), Montage de la pédagogie inversée

Projets [4]
- Année 2015-2016 : 4 équipes participant à la compétition "Mission on Mars Robot Challenge 2016" organisée par MathWorks France, 2 équipes en finale, 1 équipe gagnante
- Encadrements de projets tutorés :
  - ⋆ projet Roberto [5] (I, II, III) entre 2010 et 2013 co-encadrés avec P-E. Hladik

3. https://sites.google.com/site/projetsecinsa/
4. https://www.youtube.com/channel/UCazRcgfZU4Z0K3Q5vv9ERBA/playlists
5. https://sites.google.com/site/projetsroberto/

- ⋆ Projets de planification, relocalisation, exploration sur des robots Lego Mindstorms NXT (2014-2015) co-encadrés avec P-E. Hladik
- ⋆ Projets de localisation avec un Lidar ou par triangularisation (2014-2015) co-encadrés avec P-E. Hladik
- ⋆ Guide autonome pour les JPO (2015-2016) co-encadré avec G. Auriol
- ⋆ projets de développement d'une maquette pédagogique d'hélicoptère (2015-2017) co-encadrés avec G. Le Corre et S. Di Mercurio
- ⋆ Lien entre jeu réel et jeu virtuel (2016-2017) co-encadré avec P-E. Hladik
- ⋆ Développement de robots Ergo Jr, utilisation de ROS (2016-2017) co-encadrés avec P-E. Hladik

3.1.4 Développement de nouveaux travaux pratiques

Mon investissement en enseignement passe également dans le renouvellement des manipulations de travaux pratiques et la veille technologique sur les nouvelles maquettes proposées sur le marché. Nous avons développé plusieurs maquettes en interne grâce à une équipe variée d'enseignants, ingénieurs et techniciens. Les principales maquettes que nous avons développées sont les suivantes :

- maquette d'une montre digitale pour illustrer l'utilisation des statecharts, actuellement utilisée pour des TP en 2ème et 3ème année sur deux pré-orientations,
- développement d'un TP sur les graphes avec un Lego Mindstorms, actuellement traité par la 4ème année Automatique et Electronique,
- TP sur la commande optimale et le retour d'état avec un Lego Mindstorms en pendule inversé, actuellement traité par la 4ème année Automatique et Electronique,
- maquette d'hélicoptère en vue du développement d'une manipulation sur le thème de la commande de systèmes multi-variables,
- maquette de voiture autonome à partir d'une voiture jouet d'enfant pour les projets multidisciplinaires en Systèmes Embarqués Critiques.

Ces développements ont donné lieu à 3 publications pédagogiques, la première publication étant issue du développement d'un TP lors de ma dernière année d'étude à l'ENSEEIHT sur les graphes.

3.2 Autres enseignements

ENSEEIHT - Département Génie Electrique et Automatique

- 2009-2010 : projet long "Comparaison des diagnostiqueurs de la NASA et du LAAS" co-encadré avec L. Travé-Massuyès
- 2010-2011 projet long "Développement d'un simulateur de satellite" co-encadré avec S. Indra
- 2015-2016 : projet long "Amélioration des performances du sportif à l'aide d'un outil de reconnaissance de mouvements " co-encadré avec E. Le Corronc
- 2016-2017 : projet long "Amélioration des performances du sportif par des méthodes d'apprentissage" co-encadré avec P. Ribot
- 2017-2018 : projet long "Diagnostic et pronostic de fautes" co-encadré avec P. Ribot
- depuis 2016 : TD en systèmes multidimensionnels (10h TD/BE par an)
- 2002-2005 : TD d'optimisation statique, estimation filtrage, cours et TD de commande optimale, TP d'automatique (96h eqTD par an en tant que vacataire)

EPMI - Cergy Pontoise

- 2005-2006 : Cours et TD sur la commande des systèmes linéaires, productique, mathématiques, TP sur microprocesseur

3.3 Publications pédagogiques

▶ B. Sareni, G. Fontan, E. Chanthery et S. Caux. *OrdoNet, un outil de modélisation et d'analyse des graphes potentiel-tâche sous Matlab.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 8, no. hors série 1, page 1003, 2009

▶ A. Subias, E. Chanthery, G. Le Corre, J. Martin et V. Mahout. *A l'Heure des Statecharts et de XPC target pour la Commande d'une Montre Digitale.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 11, 2012

▶ E. Chanthery, G. Le Corre et P.-E. Hladik. *De l'illustration du guidage à l'optimisation d'un plan par un robot Lego Mindstorms NXT.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 15, Novembre 2016

Deuxième partie

Recherche scientifique

4 Synthèse des activités de recherche jusqu'en 2006

Résumé

Ce chapitre synthétise les activités de recherche que j'ai menées de la fin de mes études d'ingénieur jusqu'à l'obtention de mon poste de maître de conférences. Les thématiques traitées lors de mon DEA et mon doctorat sont rappelées, ainsi que mes premières activités de recherche au sein de l'Equipe Commande des Systèmes de l'Ecole Nationale Supérieure de l'Electronique et de ses Applications à Cergy-Pontoise.

4.1 Résumé des travaux de DEA et de doctorat

4.1.1 DEA Systèmes - 2002

Réalisé au sein de l'Office National d'Études et de Recherches Aérospatiales, dans le département Commande des Systèmes et Dynamique du vol (DCSD), sous la direction de Magali Barbier, Ingénieur de Recherche, ONERA/DCSD.
Sujet : Planification de Mission pour un Véhicule Aérien Autonome.
Mots clés : véhicule autonome, drone, UAV, planification de mission, algorithme multi-critère, guidage, ProCoSA

L'objectif était de développer un logiciel de gestion de mission pour un véhicule aérien totalement autonome basé sur la structure suivante. Le contrôle de l'exécution de la mission est réalisé avec l'outil ProCoSA (PROgrammation et Contrôle des Systèmes à forte Autonomie) dans lequel le comportement du véhicule est décrit à l'aide de réseaux de Petri interprétés. Le système global est composé de sous-systèmes codés dans des serveurs, dans lesquels sont implémentées des méthodes ou fonctions algorithmiques, dont un algorithme de planification.

Le travail comptait 2 parties :

- Mise en place d'une étude au niveau de la planification d'itinéraires. Le problème consistait à optimiser l'itinéraire du drone en tenant compte de plusieurs critères et contraintes, fonctions par exemple de l'énergie embarquée disponible et du respect de créneaux horaires. Le problème a été étudié sous deux optiques :
  — Une reformulation d'un problème multicritère existant adapté à un véhicule aérien et implémenté dans un nouveau sous-système de planification.
  — Une reformulation du problème de planification de mission d'un drone, en vue d'un travail plus vaste débouchant sur une thèse sur le même sujet, associée à des recherches d'algorithmes pouvant répondre au problème.
- Développement de briques pour la réalisation d'une mission pour un véhicule aérien autonome :
  — Etude du problème de la gestion de mission d'un véhicule aérien autonome : définition d'une mission type, spécification du comportement du drone et mod-

élisation à l'aide de réseaux de Petri. On décide d'identifier les réseaux par tâches indépendantes et hiérarchisées de manière la plus générique possible. Le réseau le plus haut dans la hiérarchie correspond au déroulement d'une mission type, et plus on descend dans la hiérarchie, plus le niveau de détails est élevé.
- Définition d'une structure pour les communications entre les sous-systèmes basée sur la centralisation des données dans un seul sous-système.
- Définition d'une mission test après un travail de recherches sur les missions possibles. La mission élaborée est une mission d'observation qui permet de tester le sous-système de planification ainsi que toutes les opérations envisagées. En collaboration avec François Legras, doctorant au DCSD, j'ai élaboré un modèle de drone de type MALE, dont l'altitude de vol et l'endurance permettent des missions d'observation intéressantes. Le sous-système de commande est adapté à ce type de véhicule.

Le travail de recherches a abouti à des implémentations d'algorithmes visant à répondre au problème. Il restait à choisir l'algorithme le plus adapté et à l'intégrer dans le logiciel. Le travail qui a suivi en thèse devait approfondir les recherches sur les algorithmes, ainsi que la manière de considérer le problème.

4.1.2 Thèse de doctorat (sept 2002-oct. 2005)

Réalisée au sein de l'Office National d'Études et de Recherches Aérospatiales (ONERA), dans le département Commande des Systèmes et Dynamique du vol (DCSD).
Sujet : Planification de Mission pour un Véhicule Aérien Autonome
Direction : Magali Barbier, Ingénieur de Recherche, ONERA/DCSD, Jean-Loup Farges, Ingénieur de Recherche, ONERA/DCSD, Raja Chatila, Directeur de Recherche, LAAS/-CNRS
Financement : Allocation du Ministère de l'Education Nationale et de la Recherche

⇨ Problématique et contexte de la thèse

Le développement de véhicules autonomes pour exécuter des missions dans des environnements dangereux et inconnus est devenu un défi important en termes d'autonomie et de planification. Les missions avec des communications limitées entre le véhicule et les opérateurs impliquent une autonomie décisionnelle à bord du véhicule. En effet, le véhicule doit non seulement suivre le plan courant, mais aussi réagir de façon autonome à des événements survenant en cours de mission et invalidant le plan.

L'objectif de ce travail est de développer une fonction de planification embarquée au sein d'une architecture dans un véhicule aérien autonome. Le contexte applicatif est une mission militaire d'observation pour un tel véhicule dans un environnement 3D, dynamique, incertain et dangereux. L'environnement inclut une zone ennemie où le véhicule effectue des opérations d'observation qui sont les objectifs de la mission. Les contraintes de la mission sont induites par les objectifs, l'environnement et par l'engin. La fonction de planification doit choisir et ordonner le meilleur sous-ensemble d'objectifs et déterminer la date d'arrivée sur chaque objectif, en maximisant les profits dus aux observations et en minimisant des critères sur le danger, la consommation de carburant et les durées, tout en respectant les contraintes de la mission.

Les systèmes de planification existants ne peuvent pas résoudre le problème de planification de mission pour des véhicules autonomes. En effet, pour ces systèmes, l'ensemble des objectifs réalisés est fixé *a priori*, les profits liés aux objectifs sont uniformes, les coûts entre les objectifs sont linéaires et les ensembles de contraintes sont définis par des programmes linéaires en nombres entiers.

⇨ Un formalisme pour le problème de planification de mission

Après une définition du terme "planification de mission" et l'étude des approches pour décrire le domaine de planification et pour trouver un plan répondant aux spécificités du problème de planification de mission, ce travail propose un formalisme pour décrire le problème de planification de mission, il utilise la notion d'abstraction et prend en compte différentes façons de traiter un objectif, dans l'espace et dans le temps. Il introduit la notion d'utilisation de ressources, décomposables ou pas le long du trajet. Ce formalisme s'applique à tous les problèmes de planification de mission. Il n'est pas spécifique au problème de planification de mission pour un véhicule aérien autonome. Le haut niveau de description correspond à la réalisation des objectifs. La prise en compte du temps et l'utilisation des ressources nécessitent une description bas niveau. Ce niveau est spécifique au domaine d'application. Pour une mission pour un véhicule aérien autonome, il décrit l'évolution de la position (x, y, z) et de la vitesse du véhicule.

⇨ Les algorithmes de planification

Un algorithme basé sur une recherche de type A^* est développé. La particularité de cet algorithme est que pour chaque nœud développé, l'évaluation précise du critère requiert une optimisation des dates de passage sur l'itinéraire défini jusqu'à ce nœud. La sortie est un chemin défini par une liste ordonnée de nœuds et par un vecteur des durées optimisées entre chaque paire de nœuds. L'algorithme de planification est adapté pour la replanification en ligne et peut donc commencer en un point quelconque en tenant compte de la nouvelle situation. Pour chaque nœud développé, un sous-problème d'optimisation est résolu. Ce problème correspond à l'optimisation d'un critère non linéaire sous des contraintes mixtes. Il est résolu par un algorithme de Frank-Wolfe [Frank & Wolfe 1956]. Les contraintes linéaires sont prises en compte dans les contraintes d'un simplexe. Les contraintes non linéaires sont ajoutées au critère sous la forme de termes de pénalisation.

Le calcul rapide d'un premier itinéraire admissible augmente la réactivité du système et l'efficacité de l'élagage en donnant une première borne utilisée ensuite dans l'algorithme de recherche. Quatre méthodes pour évaluer le coût d'un plan incomplet sont développées. Deux méthodes d'élagage sont utilisées. Elles tiennent compte du fait que l'évolution du critère n'est pas monotone en descendant une branche de l'arbre d'exploration et que de fait les méthodes d'élagage classiques ne s'appliquent pas. Les méthodes développées doivent donc tenir compte de l'évolution future du critère. Quatre méthodes de rangement des nœuds pendants sont proposées. Les combinaisons de ces méthodes permettent de tester différents algorithmes de planification.

⇨ Intégration de la planification dans une architecture de contrôle embarquée

Le module de planification ainsi développé s'inscrit dans une architecture de contrôle embarquée. On propose d'employer une structure basée sur la combinaison du module de planification avec le contrôleur d'exécution ProCoSA [Barbier *et al.* 2006]. L'originalité de ProCoSA est qu'il permet d'ordonner des tâches et gérer des événements grâce à la description de réseaux de Petri. Une architecture hybride hiérarchisée en quatre niveaux est développée de manière à gérer des niveaux d'autonomie allant de la gestion de la mission jusqu'au guidage. On montre ainsi que la planification peut être intégrée dans une architecture réactive complexe.

⇨ Les tests

Quatre missions sont proposées pour les tests. Une planification initiale est appliquée à chaque mission. Elle permet de déterminer les fenêtres temporelles appropriées pour les points d'entrée et de sortie de la zone ennemie. Pour chaque mission, on détermine trois dates de replanification : une en début de mission, une en milieu de mission et la

troisième en fin de mission. Trois événements déclencheurs sont étudiés pour ces trois moments : un manque de carburant (le véhicule n'a plus assez de carburant pour achever le plan courant) ; un changement dans la carte des zones de danger ; un changement dans la carte des zones objectif (disparition ou apparition de zones objectif, changement de gain, contraintes temporelles modifiées, ...).

La combinaison des trois moments de replanification et des trois événements déclencheurs permet de développer neuf scénarios de replanification par mission. 36 scénarios sont simulés sur 16 combinaisons de méthodes, en fonction de l'instant de replanification et du type d'événement déclencheur. L'analyse des résultats permet de dégager les méthodes présentant les meilleurs compromis qualité/temps de calcul, notamment un algorithme basé sur une heuristique évaluant les récompenses non encore obtenues et sur un rangement au meilleur d'abord.

⇨ **Conclusion**

Ce travail présente un formalisme pour une classe de problèmes relatifs à la planification de mission. Le formalisme utilise la notion d'abstraction pour décrire le problème avec une hiérarchie à deux niveaux. Le plus haut niveau décrit la réalisation des objectifs. Il est complété par un niveau moins abstrait qui dépend de l'application et décrit la dynamique du véhicule en temps et en utilisation de ressources.

Ce travail propose une solution à un problème de planification non classique, où le nombre d'objectifs n'est pas fixé *a priori*. De plus, le critère n'a pas une évolution monotone quand on explore l'arbre des chemins en profondeur et la planification doit gérer des contraintes de temps et de ressources. Par conséquent, les méthodes d'élagage classiques ne sont pas applicables.

Les algorithmes proposés utilisent une variation de l'algorithme A^*. Ces algorithmes peuvent être utilisés en planification globale, mais l'objectif principal de ce travail est de les utiliser en ligne en replanification, pour réagir à des événements survenant en cours de mission.

Le formalisme et les algorithmes sont testés pour des missions militaires d'observation pour un système aérien autonome, dans un environnement 3D, dangereux, incertain et dynamique. Enfin, on montre que l'intégration de la planification de mission associée dans une architecture embarquée est envisageable.

Mots clés : engin autonome, planification, recherche heuristique, architecture embarquée

Publications liées aux travaux de DEA et de doctorat

Revue internationale

▶ M. Barbier et E. Chanthery. *Autonomous Mission Management for Unmanned Aerial Vehicles.* Aerospace Science and Technology, vol. 8, pages 359–368, 2004

Congrès internationaux avec comité de lecture

▶ E. Chanthery, M. Barbier et J-L. Farges. *Planning algorithms for autonomous aerial vehicle.* In 16th IFAC World Congress, volume 16, 2005

▶ E. Chanthery, M. Barbier et J.-L. Farges. *Mission Planning for autonomous Aerial Vehicles.* In IAV2004 - 5th IFAC Symposium on Intelligent Autonomous Vehicles, 2004

▶ E. Chanthery et M. Barbier. *Functional Modules for Intermixed Planning and Execution of an Observation Mission.* In 18th Bristol UAV Systems Conference, April 2003

4.2 Activités de recherche pour l'année 2005/2006

Entre septembre 2005 et septembre 2006, j'ai occupé un poste d'enseignant-chercheur à temps plein à l'Ecole d'Electricité de Production et des Méthodes Industrielles (EPMI) à Cergy-Pontoise. L'EPMI est un établissement privé d'enseignement technique dont le diplôme est reconnu par la Commission des Titres d'Ingénieurs. Mes recherches étaient associées avec l'Equipe Commande des Systèmes (ECS) de l'Ecole Nationale Supérieure de l'Electronique et de ses Applications à Cergy-Pontoise.

Mes activités de recherche à l'EPMI ont concerné la partie "systèmes à événements discrets" pour le diagnostic des systèmes hybrides. On le verra plus précisément plus tard dans ce manuscrit, mais en résumé, les systèmes hybrides sont des systèmes dans lesquels il existe des interactions non triviales entre des composantes discrètes et des composantes continues.

Durant cette année de transition, j'ai entamé une collaboration avec Christophe Combastel de l'équipe ECS sur la partie DIAgnostic et Méthodes Ensemblistes (DIAME). En effet, ses recherches portaient essentiellement sur la partie continue et il aurait été intéressant de compléter ce travail par des recherches sur la partie discrète du système.

Par ailleurs, j'étais responsable de la salle de productique de l'EPMI . J'avais pour objectif d'appliquer mes recherches sur la ligne de production disponible dans cette salle. Il s'agissait d'une ligne de transfert automatisée dotée de 4 ateliers flexibles pilotés par automates industriels, le tout muni d'un système de supervision et de gestion par ordinateur. La ligne réalisée par le Groupe Schneider et sa filiale M2A, simule une usine complète de fabrication de produits pharmaceutiques. L'objectif de mes recherches était d'étudier des solutions en vue d'un diagnostic de pannes pour une ligne de production. En effet, la ligne détecte un certain nombre de défauts basiques, mais certains aspects (opérateur, défauts non prévus, défauts mécaniques) ne sont pas pris en compte et ne sont pas détectés. J'ai eu pour objectif d'étudier différentes méthodes de diagnostic appliquées ou non sur des lignes de production, notamment sur les modélisations discrètes (réseaux de Petri, machines à état, algèbre maxplus), mais aussi sur des modélisations hybrides. En effet, la ligne est un système complexe hybride autonome, puisque la ligne est entièrement automatisée, mise à part un poste d'opérateur manuel, n'ayant pas autorité sur le fonctionnement de la ligne. L'objectif était de développer un module de diagnostic basé sur l'exemple de la ligne, en comparant les différentes modélisations. A terme, en fonction des résultats obtenus, ces recherches avaient pour but d'intégrer un module de diagnostic hybride dans une architecture de supervision.

Ce travail n'a pas donné lieu à des publications, mais un stage de master de recherche en co-encadrement EPMI-ENSEA a été proposé et un encadrement de stage de fin d'études a été effectué.

5 Positionnement des travaux récents

Résumé

Ce chapitre a pour objectif de donner les motivations qui sous-tendent les travaux que j'ai effectués ces 10 dernières années. Un avant-propos rappelle rapidement les définitions des notions majeures utilisées dans ce manuscrit. La notion d'architecture de supervision d'un système est expliquée. Une architecture de gestion de santé est proposée. Les verrous scientifiques que nous avons soulevés et résolus sont exposés. Enfin, la dernière section présente les axes de travail que j'ai explorés et les contributions.

5.1 Avant-propos

La fonction de diagnostic sert à déterminer l'état de santé courant du système. Dans les travaux de [Zwingelstein 1995], le diagnostic est défini de la manière suivante.

Définition 1 (Diagnostic) *Le diagnostic est l'identification de la cause première de la (ou des) défaillance(s) à l'aide d'un raisonnement logique fondé sur un ensemble d'informations provenant d'une inspection, d'un contrôle ou d'un test.*

Cette définition repose principalement sur le terme "défaillance". Nous choisissons la définition suivante afin de le caractériser.

Définition 2 (Défaillance) *Une défaillance correspond à une cessation de l'aptitude d'une entité à accomplir une ou plusieurs fonctions requises.*

On peut distinguer deux formes de défaillances : les défaillances partielles et les défaillances complètes. Une défaillance partielle correspond à une dégradation de l'aptitude d'un système à accomplir un certain nombre de fonctions requises. Si le système n'est plus capable de réaliser les fonctions pour lesquelles il a été conçu, on parle de défaillance complète et on dit que le système est défaillant ou en panne. Dans [Villemeur 1988], une panne est définie comme l'inaptitude d'une entité à accomplir une fonction requise.

Une défaillance est générée par une faute [Isermann 1997].

Définition 3 (Faute) *Une faute représente une déviation non acceptable d'au moins une propriété caractéristique ou d'un paramètre du système.*

Les défaillances sont généralement surveillées à l'aide de grandeurs quantitatives appelées des indicateurs. Lorsque ces indicateurs révèlent un comportement anormal, ils deviennent des symptômes. Les symptômes traduisent les effets observables des défaillances.

Définition 4 (Symptôme) *Un symptôme est l'effet ou la conséquence visible d'une défaillance.*

La définition 1 met ainsi en évidence les deux tâches du diagnostic : l'observation des symptômes de la (ou des) défaillance(s) et l'identification de la (ou des) faute(s) qui en sont à l'origine à l'aide d'un raisonnement fondé sur des observations sur le système.

Les comportements anormaux ou les fautes peuvent être anticipés par un raisonnement de pronostic sur le système. Dans la littérature, on retrouve principalement deux définitions du pronostic [Brotherton *et al.* 2000, Goh *et al.* 2006].

Définition 5 *Le pronostic consiste à calculer une prédiction de l'état d'un composant ou d'un système.*

Définition 6 *Le pronostic est la capacité de prédire la durée de vie résiduelle (RUL pour* Remaining Useful Life*) de composants ou systèmes en service.*

La définition 5 est très générale. Elle assimile le pronostic à une fonction de prédiction des états futurs possibles du système. La définition 6 souligne les objectifs applicatifs de la fonction de pronostic.

La durée de vie résiduelle (RUL) d'un système correspond au temps restant avant que le système ne puisse plus réaliser avec succès ses fonctions requises et doive être remplacé [Engel *et al.* 2000]. La définition 6 met en évidence la notion temporelle liée à la prédiction. Cependant, le RUL peut également s'exprimer en terme de distance (kilomètres parcourus par un véhicule) ou en nombre d'utilisations (nombre de missions pour les navettes spatiales), par exemple. Les besoins sous-jacents sont de savoir si le système peut terminer sa mission sans rencontrer de défaillance, ou encore dans un objectif de maintenance, de déterminer la meilleure date pour réparer le système. De la définition de RUL découle la notion de date de fin de vie (EOL pour *End Of Life*), qui correspond à la date à laquelle le système n'est plus opérationnel. Le RUL propose donc une date relative à la date courante alors que la date de fin de vie est une date absolue.

Deux propriétés découlent des méthodes de diagnostic et de pronostic de systèmes : la diagnosticabilité et la pronosticabilité. La diagnosticabilité est une mesure de la capacité d'une méthode de diagnostic à diagnostiquer les fautes qui apparaissent dans un système à partir des observations disponibles sur le système. Cette propriété détermine particulièrement les fautes que l'on peut discriminer avec les observations disponibles sur le système [Sampath *et al.* 1995].

Définition 7 (Diagnosticabilité) *La diagnosticabilité est la capacité d'un système et de ses fonctions de surveillance et de diagnostic à exhiber dans un délai fini des observations différentes pour chaque situation de faute anticipée.*

La propriété de pronosticabilité détermine les fautes et les défaillances qu'il est possible de prédire avec la connaissance disponible sur le système [Ribot 2009].

Définition 8 (Pronosticabilité) *La pronosticabilité est la capacité d'un système et de ses fonctions de surveillance et de pronostic à prédire dans un délai fini la date d'une défaillance anticipée.*

Dans le cadre de la gestion de la santé, on s'intéresse au suivi des changements de dynamiques du système lorsqu'une ou plusieurs fautes surviennent.

Définition 9 (Mode, Mode nominal/dégradé/de défaillance) *Un mode décrit la ou les dynamique(s) du système dans des conditions de santé spécifiques. Tant qu'aucune faute n'est survenue, le système est dans un mode nominal (nominal mode). Dans ce cadre,*

les fautes sont supposées permanentes, c'est-à-dire que lorsqu'une faute survient, le système entre dans un mode dégradé (degraded mode) et ne reviendra plus dans un mode nominal sans réparation. Ce mode dégradé correspond à une défaillance partielle. Le système peut ainsi finir dans un mode de défaillance (failure mode), dans lequel il n'est plus opérationnel. Ce mode correspond à une défaillance complète.

Cette évolution est illustrée sur la figure 5.1.

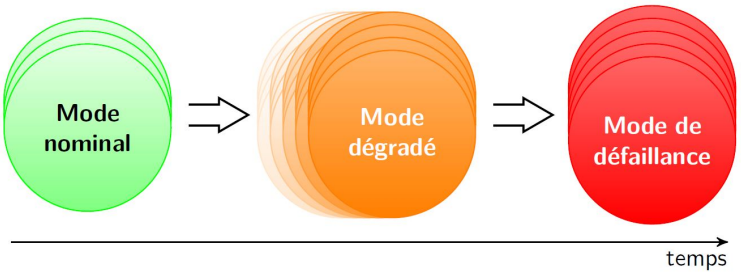


FIGURE 5.1 – Évolution unidirectionnelle d'un système sans maintenance ou action de réparation.

À travers le suivi du mode du système, la méthode de diagnostic détermine le mode courant du système et le chemin emprunté (trajectoire passée), dont la séquence de modes et l'ensemble des fautes ayant eu lieu, pour l'atteindre. La méthode de pronostic prédit quant à elle les trajectoires futures possibles et particulièrement la date d'entrée dans un mode de défaillance (EOL).

5.2 Motivation : l'autonomie des systèmes

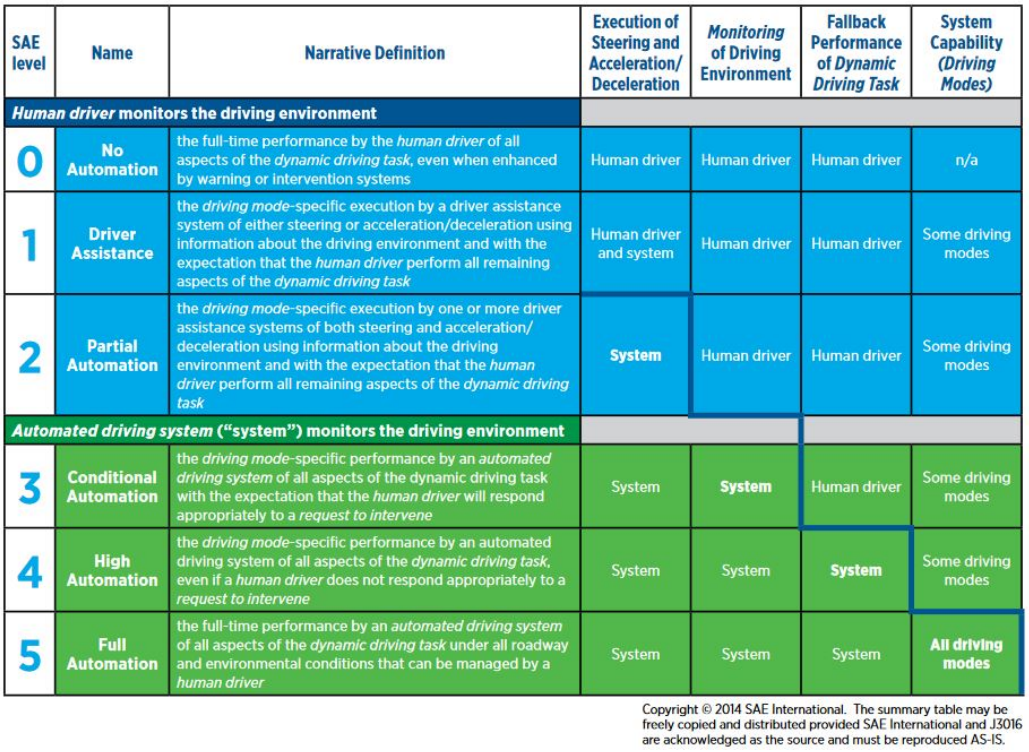
La motivation majeure de mes travaux est l'augmentation de l'autonomie des systèmes. Ainsi, les fonctions de diagnostic et de pronostic participent à l'autonomie en donnant au systèmes des capacités d'évaluation de son état. La notion d'autonomie d'un système reste centrale, il est donc important de s'y arrêter.

La notion d'autonomie est définie de plusieurs manières dans la littérature [Antsaklis & Passino 1989, Alami *et al.* 1998, Durst & Gray 2014, Grabowski 2015] et de nouvelles définitions apparaissent régulièrement pour pouvoir qualifier précisément l'autonomie des systèmes récents en fonction des besoins. C'est le cas par exemple dans le contexte actuel des voitures autonomes, des aéronefs autonomes, ou des engins spatiaux.

De manière générale, deux concepts cohabitent derrière le terme "autonomie" : la réalisation d'une mission sans intervention humaine et la faculté d'adaptation face aux perturbations de son environnement.

On retiendra ici la définition de la notion d'autonomie donnée par le National Institute of Standards and Technology (NIST), selon [Huang *et al.* 2004] et reprise de la thèse de A. Lampe [Lampe 2006].

DÉFINITION 10 (AUTONOMIE) *(1) Condition ou qualité à être auto-gouverné (2) Capacité propre d'un système, à capter, percevoir, analyser, communiquer, planifier, prendre des décisions et agir afin d'atteindre les buts qui lui ont été assignés par un opérateur humain à l'aide d'une interface homme/ machine dédiée. L'autonomie est échelonnée sur*



SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/ Deceleration	*Monitoring* of Driving Environment	Fallback Performance of *Dynamic Driving Task*	System Capability *(Driving Modes)*
***Human driver* monitors the driving environment**						
0	No Automation	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	Driver Assistance	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes
2	Partial Automation	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/ deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes
***Automated driving system* ("system") monitors the driving environment**						
3	Conditional Automation	the *driving mode*-specific performance by an *automated driving system* of all aspects of the dynamic driving task with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes
4	High Automation	the *driving mode*-specific performance by an automated driving system of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes
5	Full Automation	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**

FIGURE 5.2 – Les niveaux d'autonomie définis par la SAE International.

plusieurs niveaux, qui sont caractérisés par des facteurs incluant la complexité de la mission, les difficultés environnementales et le niveau d'interaction homme/robot nécessaire à l'accomplissement de la mission.

De même, des niveaux d'autonomie coexistent dans la littérature. On cite ici les principaux à titre d'exemples.

Le système de classification ALFUS (Autonomy Level For Unmanned Systems), issu d'un groupe de travail créé en juillet 2003, propose un système de classification générique pour l'autonomie [Huang *et al.* 2005]. Ce groupe rejoindra la SAE international en 2008. La classification ALFUS considère 3 aspects : l'indépendance par rapport à l'humain, la complexité de la mission et la complexité de l'environnement.

Une standardisation commence à voir le jour en 2014, proposée par la SAE international, anciennement Society of Automative Engineers (SAE), qui est une organisation internationale comptant plus de 128 000 membres ingénieurs et experts techniques qui échangent des informations et des idées sur l'ingénierie des véhicules. Les normes SAE dans le domaine de l'industrie aérospatiale et pour les véhicules terrestres sont reconnues et utilisées dans le monde entier. Ainsi, dans le cadre de l'autonomie d'un véhicule, l'agence fédérale américaine NHTSA (National Highway Traffic Safety Administration) ainsi que l'OICA (Organisation internationale des constructeurs automobiles) en Europe, ont adopté les définitions internationales de la SAE (J3016) pour les niveaux d'autonomie. La norme SAE J3016 [of Automotive Engineers 2014] compte 6 niveaux d'autonomie illustrés sur la figure 5.2.

Au niveau 0, le conducteur humain opère toutes les tâches de conduite. Au niveau 1, un système automatisé sur le véhicule peut parfois aider l'humain, soit pour tourner, soit pour les tâches d'accélération ou freinage. Au niveau 2, un système automatisé sur le véhicule

peut effectivement conduire durant certaines phases de conduite, tandis que l'humain continue de surveiller l'environnement de conduite et effectue le reste de la tâche de conduite. A partir du niveau 3, on considère que les systèmes de conduite sont automatisés. Au niveau 3, un système automatisé peut mener certaines tâches de conduite et surveille l'environnement de conduite dans certains cas. L'humain doit néanmoins être prêt à reprendre le contrôle sur une requête du système automatisé. Au niveau 4, un système automatisé peut mener certaines tâches de conduite et surveille l'environnement de conduite dans certains cas. Cela doit se faire même si l'humain ne répond pas de manière appropriée aux requêtes du système automatisé. Au niveau 5, le système automatisé peut effectuer toutes les tâches de conduite, dans toutes les conditions où un conducteur humain pourrait les exécuter.

Ces niveaux sont basés sur 3 concepts principaux : la responsabilité du conducteur, la capacité du système à détecter ses limites et la capacité du système à gérer les situations imprévues. C'est sur les deux dernières capacités qu'est centré mon travail : la capacité du système à détecter ses limites consiste à mener à bien des tâches de diagnostic, voire de pronostic. La capacité du système à gérer les situations imprévues est liée à la décision et à l'évaluation d'une situation.

Dans le domaine spatial, l'ESA [Vassev & Hinchey 2013] considère 4 niveaux d'autonomie pour l'exécution des opérations de missions :

- exécution sous le contrôle en temps réel du segment sol ;
- exécution d'opérations de mission pré-planifiées à bord ;
- exécution d'opérations de mission adaptatives à bord ;
- exécution d'opérations de mission orientées par des objectifs à bord.

Ces niveaux d'autonomie sont illustrés sur la table 5.1.

TABLE 5.1 – Les niveaux d'autonomie définis par l'ESA.

Niveaux d'autonomie	Description	Fonctions
E1	(1) Exécution de la mission sous contrôle du sol (2) Capacité à bord limitée pour les problèmes de sûreté de fonctionnement	(1) Contrôle en temps-réel du sol pour les opérations nominales (2) Exécution de commandes planifiées pour les problèmes de sûreté de fonctionnement
E2	Exécution d'opération de mission pré-planifiées, définies par le sol et exécution à bord	Capacité de stocker des commandes dans le planificateur de bord
E3	Exécution d'opérations de mission adaptatives à bord	Opérations autonomes à temps basée sur des événements. Exécution de procédures de commandes à bord
E4	Exécution d'opérations de mission orientées par des objectifs à bord	Replanification de mission orientée objectifs

Pour augmenter l'autonomie des systèmes, il est nécessaire de s'inscrire dans une architecture de supervision. Ces architectures font l'objet de la section suivante.

5.3 Architectures de supervision

Une architecture représente la décomposition structurelle d'une tâche à effectuer. Dans la plupart des travaux passés, cette décomposition est propre à l'application (robotique, automobile, avionique, spatiale, etc.). Ces architectures correspondent à la conception

d'un système dont l'objectif est la réalisation d'une mission avec un niveau d'autonomie le plus élevé possible. On appellera ces architectures des **architectures orientées-mission**. Une mission, généralement commanditée par un opérateur, est effectuée en réalisant des actions [Chanthery 2005] : des actions de mouvements, des actions sur l'environnement, etc. Le plan de la mission (ou plan d'actions) est l'ensemble ordonné des actions à réaliser. La planification est l'opération par laquelle le plan est déterminé.

L'intérêt de l'intégration d'un module de surveillance de santé (principalement de diagnostic) apparaît peu à peu après les premières utilisations des systèmes avec des problématiques de maintenance et de sécurité.

Ce n'est qu'il y a quelques années que des travaux proposent des architectures de gestion de santé (*Prognostic and Health Management* ou PHM) que nous appellerons **architectures orientées-santé**. De part la complexité croissante des systèmes, ces architectures intègrent dès la phase de conception les fonctions de diagnostic et de pronostic. Les objectifs vont de l'augmentation de l'autonomie d'un système (automatique, intelligence artificielle) à l'aide à la décision pour l'opérateur de maintenance, en passant par la réduction des coûts de maintenance, etc.

La table 5.2 résume une étude de la thèse de Quentin Gaudel [Gaudel 2016] sur les positions des principales architectures étudiées concernant l'intégration des fonctions de diagnostic, de pronostic et de gestion de mission. Dans ce tableau, les flèches représentent les liens entre les différentes fonctions. Par exemple, la flèche $\longleftrightarrow$ dans la colonne Mission-Diagnostic indique que le modèle du diagnostic utilise les informations relatives à la mission et que le module de décision utilise les informations issues du diagnostic.

On constate que dans tous les travaux, il existe une séparation claire entre le processus de diagnostic et le processus de décision. Le modèle du diagnostic utilise les informations, sous la forme de données capteurs ou données traitées par exemple, relatives à la mission ou au contrôle du système. En revanche, l'utilisation du diagnostic pour la décision ou le contrôle n'est pas souvent mis en place dans les architectures. Dans [Ghallab *et al.* 2001], [Nesnas *et al.* 2003], une fois qu'une faute est détectée, le processus d'isolation de faute est lancé puis une action corrective est mise en place.

Concernant le couplage Mission-Pronostic, la majorité des travaux utilisent des données relatives à la mission pour effectuer le pronostic. En revanche, seul [Narasimhan *et al.* 2012] effectue un couplage du pronostic vers la mission. On peut également citer des travaux dans le domaine des systèmes continus, qui étudient l'impact de la commande (assimilable à un objectif mission) sur la dégradation, ainsi que des politiques de maintenance utilisant les informations de pronostic [Langeron *et al.* 2017]. De même, [Salazar *et al.* 2017] propose une méthode prenant en compte l'utilisation des actionneurs pour réduire les dégradations sur le système tout en optimisant les performances du systèmes. Enfin, concernant le couplage Diagnostic-Pronostic, la majorité des travaux utilisent des informations du diagnostic pour effectuer un pronostic.

Nous distinguerons par la suite les architectures de gestion de mission (ou orientées-mission), élaborées dans l'objectif de réaliser la mission du système, des architectures de gestion de santé (ou orientées-santé), élaborées plus récemment pour répondre à des problématiques de type PHM.

5.3.1 Architectures orientées-mission

Historiquement, les premières architectures avaient pour objectif la réalisation d'une mission en totale autonomie, ou bien de donner au système un niveau d'autonomie plus élevé. Ces architectures s'appellent également des architectures de contrôle.

TABLE 5.2 – Position des principales architectures concernant l'intégration de diagnostic ou de pronostic à la supervision de la mission.

Travaux	Mission - Diag.	Mission - Pron.	Diag. - Pron.
[Antsaklis & Passino 1989]	⟷		
[Sampath *et al.* 1996]	⟷		
[Alami *et al.* 1998]	⟶		
[Muscettola *et al.* 1998]	⟷		
[Ghallab *et al.* 2001]	⟵		
[Nesnas *et al.* 2003]	⟵		
[Berenjii & Wang 2006]	⟶	⟶	⟶
[Hamilton *et al.* 2007]	⟷		
[Camci *et al.* 2007]	⟶	⟶	
[Benedettini *et al.* 2009]	⟶		⟶
[Zhang *et al.* 2009]	⟶	⟶	⟶
[Ribot 2009]	⟶	⟶	⟶
[Olive *et al.* 2011]	⟷		
[Roychoudhury & Daigle 2011]	⟶		⟶
[Narasimhan *et al.* 2012]	⟷	⟷	⟶
[Ribot *et al.* 2013]	⟶	⟶	⟶
[Daigle *et al.* 2015b]		⟶	

L'état de l'art a abouti aujourd'hui a un certain consensus sur les composantes nécessaires dans une architecture de contrôle. On admet aujourd'hui que ces architectures doivent inclure une composante délibérative pour les prises de décision et l'anticipation des actions à effectuer et une composante réactive pour leur mise en œuvre et la réaction aux événements en temps réel. Ces deux composantes constituent ce que nous appelons "module de décision".

DÉFINITION 11 (MODULE DE DÉCISION) *Dans une architecture, on appellera module de décision le module mettant en œuvre la composante délibérative pour les prises de décision et l'anticipation des actions à effectuer et la composante réactive pour leur mise en œuvre et la réaction aux événements en temps réel.*

La plupart des architectures sont décomposées en différents niveaux hiérarchiques. La figure 5.3 propose une architecture hiérarchique générique sur trois niveaux. Elle sert ici de référence pour mettre en avant les différences entre les architectures étudiées. Le niveau le plus élevé est le niveau décisionnel, dans lequel se trouve généralement le planificateur, qui génère le plan des actions à réaliser. Ce niveau nécessite pour cela une connaissance globale de l'objectif et du contexte de la mission. Le niveau exécutif est l'interface entre le niveau décisionnel et le niveau fonctionnel. Son rôle est d'exécuter le plan de mission en utilisant les différentes fonctions disponibles sur le système (génération de trajectoires, contournement d'obstacles, etc). Enfin, le niveau fonctionnel est l'interface avec le système.

Dans le domaine de la robotique, l'architecture 3T, introduite dans [Bonasso *et al.* 1995], est une architecture hiérarchique à trois niveaux, semblable à l'architecture générique présentée sur la figure 5.3. Elle se compose d'un système délibératif, d'un système séquenceur, et d'un ensemble de compétences. Le système délibératif gère la planification de mission du

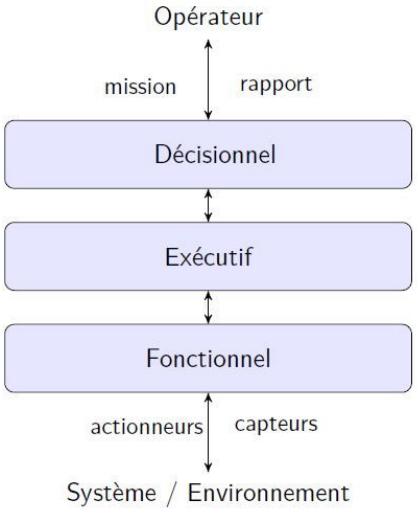


FIGURE 5.3 – Une architecture hiérarchique générique à trois niveaux : fonctionnel, exécutif et décisionnel.

robot et génère le plan des actions à réaliser. Le séquenceur instancie les actions à exécuter pour réaliser le plan envoyé par le système délibératif. Les actions sont exécutées par les compétences qui sont interfacées directement avec les capteurs et les actionneurs du robot. Aucune fonction de diagnostic (encore moins de pronostic) n'est présente. Aucune notion de panne ou de défaillance n'apparaît.

Dans le même domaine, l'architecture LAAS proposé dans [Alami *et al.* 1998] est très similaire à l'architecture 3T. Elle est également définie sur trois niveaux : un niveau décisionnel, une niveau exécutif, et un niveau fonctionnel. Le niveau décisionnel gère les fonctionnalités de haut niveau. Le niveau exécutif a pour objectif d'exécuter le plan d'actions à travers l'exécution de différentes fonctions réalisées par chaque module du niveau fonctionnel. Les modules, organisés en réseau dans le niveau fonctionnel, exécutent les fonctionnalités de bas niveaux. Outre l'utilisation de termes différents pour désigner les niveaux, l'architecture introduit la notion de défaillance. Une défaillance correspond à l'échec de la réalisation de la fonction d'un module. Si un module est défaillant, il est simplement arrêté en attendant une intervention extérieure. La prise de décision vis-à-vis de ces défaillances n'est pas abordée et la recherche d'un diagnostic plus précis n'est pas incluse.

Dans le domaine spatial, les auteurs de [Muscettola *et al.* 1998] proposent l'architecture Remote Agent. En plus des fonctionnalités déjà introduites dans les architectures existantes, elle intègre un module dédié au diagnostic du système. L'approche considérée introduit des modes de fonctionnement (mode nominal, mode de faute, etc.). Le module de diagnostic consiste donc à déterminer le mode de fonctionnement courant du système et à effectuer sa reconfiguration si nécessaire. La reconfiguration est une réorganisation logicielle ou matérielle du système. En fonction du mode du système, différents modules de bas niveau vont être utilisés pour réaliser une tâche particulière, ou bien la tâche sera réalisée par le même module, mais de manière différente. Dans tous les cas, ce résultat de diagnostic n'est pas considéré par le planificateur. Il n'y a pas de replanification des objectifs de la mission.

Les auteurs de [Olive *et al.* 2011] proposent une architecture très similaire à l'architec-

TABLE 5.3 – Description des niveaux de criticité de la stratégie FDIR.

Niveau	Description	Gestion autonome
0	Défaillance sans effet sur les performances	Oui, avec des mécanismes logiciels
1	Défaillance d'une unité de calcul ou dégradation des performances	Oui, avec la reconfiguration
2 & 3	Défaillance d'un équipement utilisé par FDIR ou défaillance non couverte par les niveaux précédents	Oui
4	Alarme matérielle ou défaillances multiples de niveau 2 ou 3	Non, passage en mode de survie (safe mode) et attente d'intervention

ture LAAS qui intègre des modules FDIR (*Fault Detection, Isolation and Recovery*) dans chacun des modules fonctionnels et également au niveau opérationnel.

L'approche FDIR est très utilisée dans le domaine spatial. Elle repose sur cinq niveaux hiérarchiques représentatifs des niveaux de criticité progressifs récapitulés dans la table 5.3. Les quatre premiers niveaux de criticité (niveaux 0 à 3) sont gérés par le système de manière autonome. Lorsque le système entre dans le dernier niveau (niveau 4), un opérateur doit prendre la main.

Bien que la stratégie FDIR soit rapide et efficace en terme d'isolation de faute, la volonté des auteurs de [Olive *et al.* 2011] est de repousser la limite du dernier niveau pour rendre le système encore plus autonome en proposant une nouvelle architecture générique. Les auteurs proposent une solution basée sur deux aspects : la mise en place d'une architecture de décision et une organisation des connaissances. L'architecture de décision est une architecture hiérarchique à trois niveaux comme présentée dans la figure 5.3. La nouvelle architecture vise la généricité pour permettre la distribution de méthodes FDIR sur les composants pour la rendre propice aux systèmes hétérogènes et répondre aux contraintes d'embarquabilité. Cependant, aucune méthode de pronostic ou autre méthode de prédiction n'est envisagée dans ce type d'architecture.

5.3.2 Architectures orientées-santé

Certains travaux prennent plus de recul par rapport aux moyens mis en place pour assurer la réalisation de la mission du système. La présence de modules de diagnostic et/ou de pronostic utilisant des connaissances globales sur le système et sur le contexte devient indispensable. La figure 5.4 présente une forme générique de ce type d'architecture, dans laquelle les différents modules sont tous liés deux à deux afin d'augmenter l'autonomie du système. Un opérateur est inclus dans l'architecture dans un cadre d'aide à la décision pour la maintenance ou pour la mise à jour des objectifs de la mission, par exemple. Contrairement aux approches hiérarchiques, les modules de diagnostic et de pronostic partagent des informations avec les modules de décision et de contrôle. Les différents travaux étudiés par la suite réalisent généralement plusieurs de ces partages mais aucun d'entre eux ne les réalise tous.

Dans [Zhang *et al.* 2009], la surveillance de santé d'un système exposé à des fautes multiples est considérée. L'approche est intéressante car les auteurs considèrent la propagation de fautes, c'est-à-dire que l'occurrence d'une faute peut avoir une conséquence sur les prochaines occurrences des fautes. L'algorithme de pronostic, dont l'objectif est de déterminer la durée de vie résiduelle avant la défaillance du système (RUL), n'est déclenché que lorsque le système est dans un état dégradé après l'occurrence d'une faute.

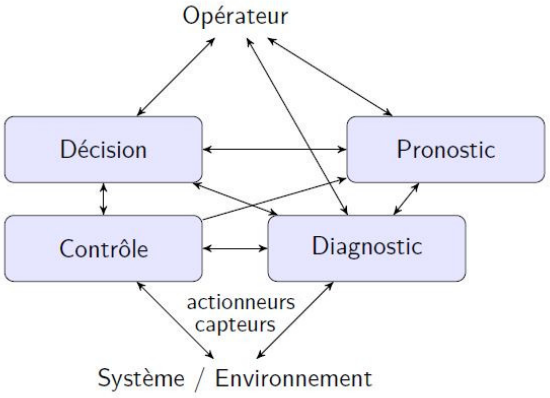


FIGURE 5.4 – Une architecture orientée-santé générique.

Dans [Roychoudhury & Daigle 2011], la surveillance de santé consiste à estimer la progression des fautes par rapport à un modèle nominal. Le pronostic se base sur le diagnostic pour produire une distribution de RUL mais les travaux ne considèrent aucun retour du pronostic vers le diagnostic.

Une architecture pour la prise de décision d'un rover autonome, utilisant les résultats de diagnostic et de pronostic pour influencer la planification de la mission est présentée dans [Narasimhan *et al.* 2012]. Un module de décision est intégré à l'architecture pour maximiser les objectifs de la mission. L'architecture est décomposée en quatre modules : contrôle de bas niveau, diagnostic, pronostic et prise de décision. L'utilisation des résultats de pronostic pour la détermination du diagnostic du système n'est pas envisagée mais le pronostic utilise la connaissance sur le plan de mission du système.

Par ailleurs, l'implémentation industrielle des architectures de maintenance a largement été étudiée. On peut citer [Rasovska *et al.* 2007] qui liste et classifie différents solutions pour la maintenance industrielle, en fonction de l'évolution de l'information utilisée et de la relations entre les systèmes. Depuis 2001, l'architecture OSA-CBM (Open System Architecture for Condition Based Maintenance) [Thurston & Lebold 2001, Lebold *et al.* 2002, Löhr *et al.* 2012], proposée par le groupe de recherche MIMOSA, s'est ainsi imposée comme un standard pour le déploiement industriel des outils de maintenance conditionnelle (CBM) et prédictive. L'architecture est structurée en 7 couches fonctionnelles nécessaires au déploiement industriel de la maintenance conditionnelle et prédictive depuis l'acquisition des données jusqu'à la présentation. La figure 5.5 tirée de [Said 2016] illustre cette organisation. Le pronostic utilise les résultats du diagnostic. La décision utilise les résultats du pronostic. L'architecture est unidirectionnelle.

5.3.3 Architectures de diagnostic

Dans le cadre des architectures orientées-santé, la fonction de diagnostic elle-même peut être organisée de différentes manières. L'architecture de diagnostic décrit quelles informations sont échangées entre les composants d'un système, le contrôleur et les modules implémentant la fonction de diagnostic.

⇨ **Architectures de diagnostic centralisées**

La plupart des approches de diagnostic se concentrent sur une vision globale du système [Patton *et al.* 2000], [Isermann 2011], [Korbicz *et al.* 2012] où toute la fonction de

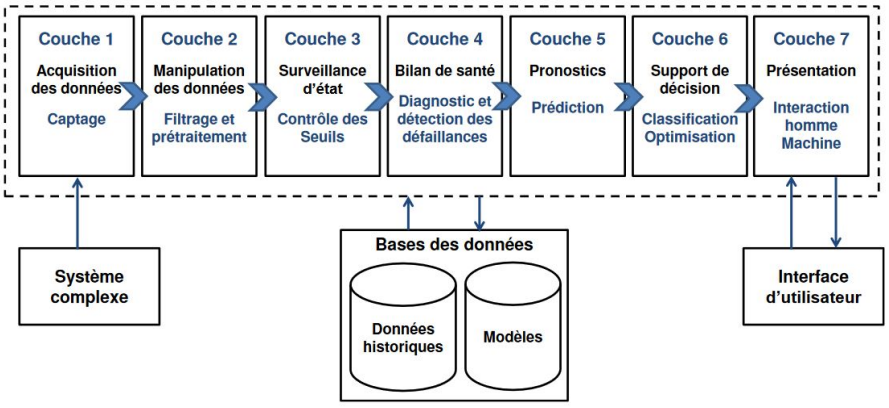


FIGURE 5.5 – L'architecture standard OSA-CBM.

diagnostic est chargée sur un seul ordinateur directement connecté au système surveillé. Toutes les informations de mesure sont disponibles directement, et par conséquent, tous les algorithmes ont toutes les informations disponibles.

Une architecture de diagnostic centralisée rassemble les données dans un système centralisé de diagnostic de fautes qui calcule ce qu'on appelle le diagnostic global. La figure 5.6 montre un exemple de ce type d'architecture.

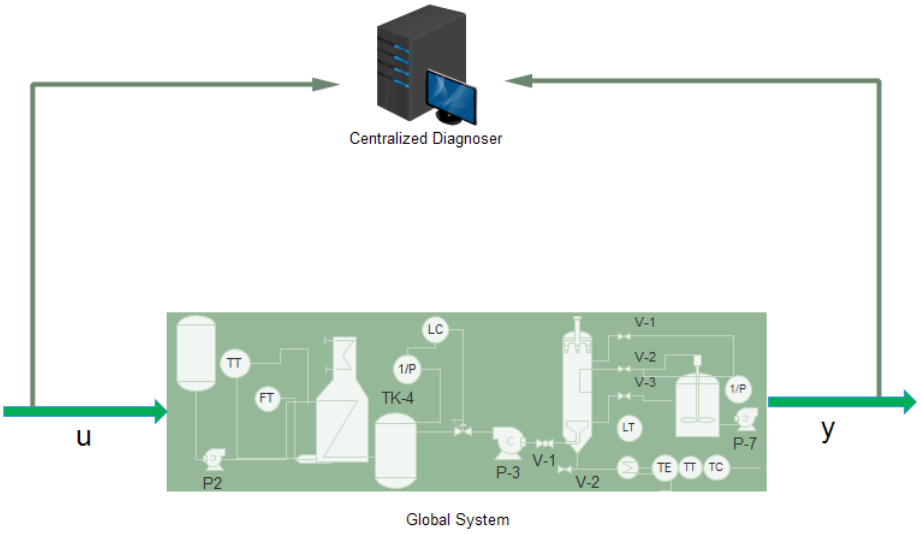


FIGURE 5.6 – Illustration d'une architecture de diagnostic centralisée.

Selon la complexité du système, cette solution peut être simple et facile à mettre en œuvre. Il n'y a pas de subdivision du système de diagnostic et donc, théoriquement, il n'y a pas de problèmes de communication. Cependant, cette solution nécessite de construire explicitement un modèle global du système, ce qui n'est généralement pas possible pour de grands systèmes pour de nombreuses raisons. Par exemple, lorsque le système couvre une grande zone géographique et que les mesures sont distribuées, elles ne peuvent pas être directement accessibles via l'ordinateur qui effectue le calcul de diagnostic. De plus, il existe des circonstances où une architecture de diagnostic centralisée, même réalisable, serait non souhaitable, en raison de plusieurs facteurs :

- la sécurité et la robustesse : en cas de défaillance du système de diagnostic centralisé,

le système doit fonctionner sans système de diagnostic ;
- la taille : les solutions centralisées ne s'adaptent pas bien à la mise à l'échelle. Plus la taille du système augmente, plus la tâche de diagnostic centralisé devient difficile. On peut citer par exemple des systèmes tels que les avions ou autres systèmes de transport, les grandes usines énergétiques ou industrielles, les réseaux de chaîne d'approvisionnement et de distribution, la production d'électricité ou d'autres processus et systèmes similaires.

Des approches de diagnostic décentralisées ou distribuées doivent donc être appliquées, et les algorithmes de diagnostic de fautes ainsi que les informations de mesure sont répartis entre différents composants.

⇨ **Architectures de diagnostic décentralisées**

Une architecture de diagnostic décentralisée suppose une décomposition du processus en sous-systèmes, chacun ayant son propre diagnostiqueur local. La tâche de diagnostic est coordonnée par un superviseur pour assurer la cohérence entre les diagnostics locaux. Il peut y avoir plusieurs niveaux de supervision, en fonction de la manière dont les sous-systèmes sont regroupés récursivement, formant ainsi une hiérarchie de supervision. Une illustration de cette hiérarchie est donnée sur la figure 5.7. Les diagnostiqueurs locaux traitent les mesures locales indépendamment les unes des autres.

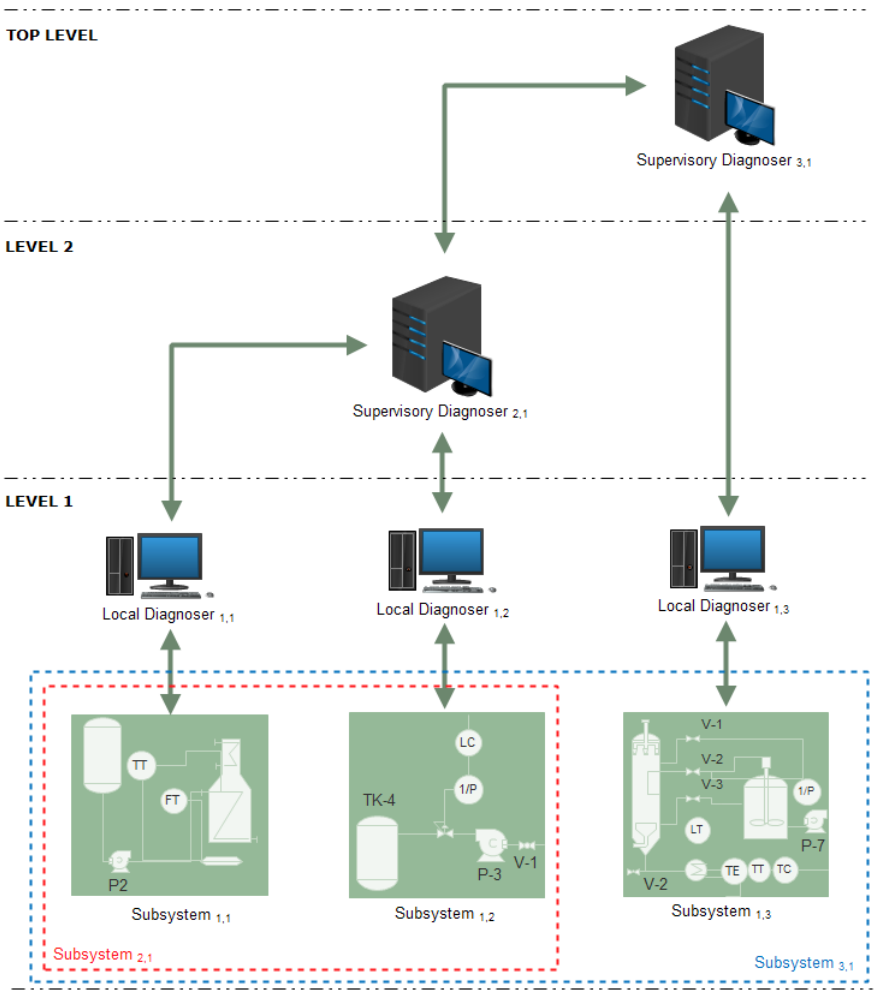


FIGURE 5.7 – Illustration d'une architecture de diagnostic décentralisée.

⇨ **Architectures de diagnostic distribuées**

Une architecture de diagnostic distribuée suppose une décomposition du processus en

sous-systèmes, chacun ayant son propre diagnostiqueur local, avec des fonctions similaires et une communication possible entre les diagnostiqueurs. Cette communication doit être correctement conçue pour que les diagnostics locaux soient globalement cohérents. On distingue deux approches de conception possibles :

1. Le système de diagnostic est conçu comme une entité unique et l'algorithme résultant est réparti sur différents composants pour faire face à l'effort de calcul nécessaire.
2. En considérant les contraintes du système, les diagnostiqueurs locaux sont conçus indépendamment, en considérant des communications entre eux, comme montré dans la figure 5.8, jusqu'à obtenir le même diagnostic qu'avec une conception de diagnostic centralisée.

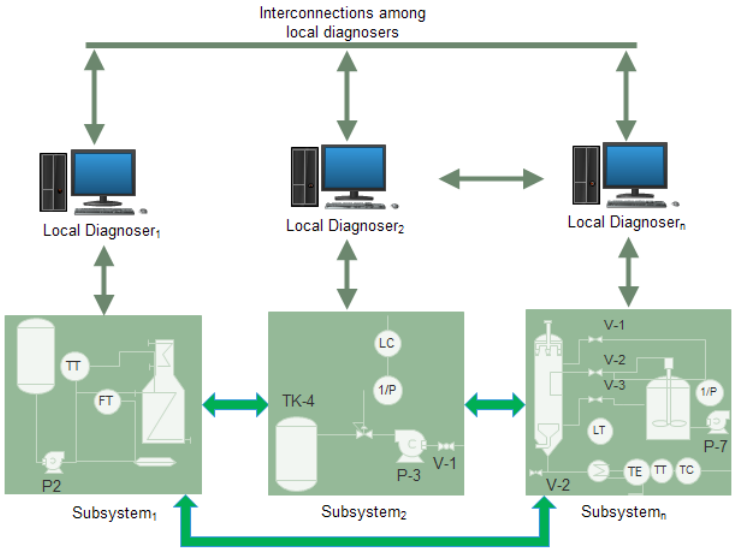


FIGURE 5.8 – Illustration d'une architecture de diagnostic distribuée.

On pourra se référer à nos articles [Chanthery *et al.* 2016b], [Pérez *et al.* 2015] pour un état de l'art sur les méthodes de diagnostic décentralisées et distribuées.

☞ Par abus de langage, on parlera parfois de diagnostic distribué pour toutes les approches non centralisées, qui regroupent à la fois le cas décentralisé et le cas distribué.

5.4 Verrous scientifiques

À notre connaissance, il n'existait donc pas d'architecture de gestion de santé intégrant un module de planification et de surveillance de santé, avec un retour du pronostic vers le diagnostic. Par ailleurs, la complexification des systèmes mène naturellement vers des études sur les architectures de diagnostic.

L'objectif de mes travaux est de proposer des solutions pour augmenter le niveau d'autonomie des systèmes en travaillant sur plusieurs aspects. Un premier aspect concerne l'intégration des différents modules : un travail a été effectué pour intégrer le diagnostic et le pronostic d'une part, puis d'autres travaux ont poussé les liens entre diagnostic et décision. Le but visé est toujours de tirer le maximum d'informations utiles de chacun des modules pour que les autres modules puissent donner des résultats les plus pertinents possibles. Cet enrichissement mutuel est au cœur de mes recherches. Plus loin que la décision, c'est le domaine entier de l'optimisation qui peut être utile pour améliorer les

fonctions de diagnostic. Enfin, comme on l'a dit plus tôt, une dernière voie d'amélioration porte sur l'adaptation ou le développement d'algorithmes de diagnostic pour les différentes architectures de diagnostic (décentralisée ou centralisée).

Les verrous scientifiques auxquels je me suis attelée durant ces dernières années peuvent ainsi être organisés selon 3 grands axes scientifiques.

5.4.1 Diagnostic et Pronostic

Les verrous identifiés pour l'axe "Diagnostic et Pronostic" sont les suivants :

- **Proposition d'une architecture de gestion de santé permettant de lier les modules de diagnostic et de pronostic**. Avant ces travaux, comme il a été souligné plus tôt dans le manuscrit, les modules de diagnostic, pronostic et de décision étaient presque indépendants les uns des autres. Les communautés de recherche étant différentes, les modules étaient très efficaces mais ne collaboraient pas. Un des challenges est donc de proposer une architecture ayant comme objectif de lier les modules de diagnostic et de pronostic pour permettre le maximum de communication et de collaboration, de manière à améliorer les résultats de chacun des modules.
- **Uniformisation des formalismes de diagnostic et de pronostic**. Les tâches de diagnostic et de pronostic possèdent de nombreux points communs : si le diagnostic consiste à estimer l'état courant d'un système en fonction des observations disponibles sur le système, le pronostic, quant à lui, consiste à prédire les états futurs du système en fonction des entrées futures sur le système. La proposition d'une uniformisation des formalismes de diagnostic et de pronostic est un verrou intéressant à lever pour lier de manière harmonieuse les deux processus.
- **Développement d'algorithmes de diagnostic et de pronostic intrinsèquement liés**. A terme, le verrou le plus important à lever est de réussir à lier les algorithmes de diagnostic et de pronostic de manière à ce qu'ils s'enrichissent l'un l'autre et que leurs résultats respectifs soient justes et les plus précis possibles. L'implémentation de ces algorithmes et leur test sur un système réel permettrait de prouver l'efficacité d'une telle proposition.

5.4.2 Diagnostic et Optimisation

Les verrous associés à l'axe "Diagnostic et Optimisation" sont les suivants :

- **Développement de nouvelles méthodes de diagnostic tirant parti d'un plan d'actions dont l'objectif terminal est de raffiner un diagnostic ambigu**. La majeure partie des systèmes réels sont non diagnosticables. Il est par ailleurs impossible d'envisager de réduire leur spectre d'actions dans le but de réduire l'espace des états accessibles à des états diagnosticables, l'objectif premier étant de réaliser une mission. Un verrou majeur est donc de trouver des solutions algorithmiques pour guider le système vers un état diagnosticable en partant d'un état où une faute est détectée, mais où elle n'est pas isolable.
- **Développement de solutions en vue d'embarquer un processus de diagnostic**. En effet, l'approche directe mène à un diagnostiqueur dont la taille n'est pas acceptable pour des systèmes réels comme des satellites. Embarquer des algorithmes efficaces mais de taille réduite reste un challenge pour les chercheurs du domaine. Certains travaux ont déjà proposé quelques débuts de solutions [Biswas *et al.* 2007, Basile *et al.* 2009, Dotoli *et al.* 2009, Grastien *et al.* 2009, Su *et al.* 2014, Vento *et al.* 2015]

.

- **Recherche de méthodes efficaces concernant la sélection de tests**. Les travaux actuels sur la sélection de tests correspondent souvent à des problèmes de priorétisation de tests, ce qui correspond à choisir le prochain meilleur test ou mesure pour désambiguiser une situation. Ceci fait donc partie intégrante du problème de troubleshooting. Ce domaine a été largement traité [Pattipati & Dontamsetty 1992, Dick & Faivre 1993, Casillas *et al.* 2013], mais les cas distribué et décentralisé ont été très peu abordés. Une voie envisagée est l'approche structurelle. La sélection de tests par cette approche a été proposée en utilisant de la programmation linéaire en nombres entiers [Bagajewicz *et al.* 2004, Sarrate *et al.* 2007a, Sarrate *et al.* 2012, Rosich *et al.* 2009]. Néanmoins les approches sont limitées sur deux points : elles n'ont pas été envisagées dans les cas distribués ni décentralisés et elles ne prennent pas en compte les éventuelles pannes capteurs.

5.4.3 Diagnostic distribué

Le troisième axe de travail s'inscrit dans une volonté d'améliorer le diagnostic dans une architecture autonome pour un système complexe. En effet, les récents développements des systèmes technologiques ont mené à une complexification des comportements. Une solution pour gérer cette complexité croissante des systèmes consiste à les considérer comme un ensemble de sous-systèmes hétérogènes et à développer des techniques distribuées pour les contrôler et les surveiller. Cette solution soulève plusieurs problèmes. Tout d'abord, au fur et à mesure que la taille et le nombre de composants augmentent, le nombre d'occurrences de pannes qui peuvent conduire le système dans un état de défaillance critique augmente d'autant. De fait, parmi les fonctions opérationnelles, les tâches de détection et d'isolation des fautes (fault detection and isolation ou FDI), de maintenance et de réparation sont devenues prédominantes et elles influent considérablement sur le coût total des produits finaux. Plusieurs verrous peuvent donc être identifiés.

- La **génération efficace de tests dans le cadre de systèmes complexes** doit se faire en tenant compte du fait que le nombre de tests possibles est important. La génération doit être automatique, efficace, et doit pouvoir s'effectuer sur des systèmes dont les modèles sont non-linéaires, ce qui reflète la nature de la majeure partie des systèmes réels.
- Le **développement de méthodes de choix de tests pertinents** constitue un deuxième verrou. Le nombre de tests disponibles étant très important dans un système complexe réel, il faut trouver des moyens pour sélectionner les meilleurs tests possibles. Cela passe par la définition de critères de choix, puis d'algorithmes d'optimisation pour la sélection de tests.

5.5 Contributions et vue d'ensemble

5.5.1 Contributions sur les liens Diagnostic et Pronostic

Ce premier axe concerne les liens entre les modules de diagnostic et de pronostic. Dans ce cadre, l'enrichissement mutuel des deux modules a été étudié. Un travail préliminaire a été effectué dans le cadre d'un stage de M2R avec Saïd Zabi, en co-encadrement avec Pauline Ribot. Ce stage a permis de proposer une première version de notre **architecture de gestion de santé** et répond partiellement au premier verrou identifié. Nous nous sommes ensuite attelées à proposer un **cadre commun de modélisation pour**

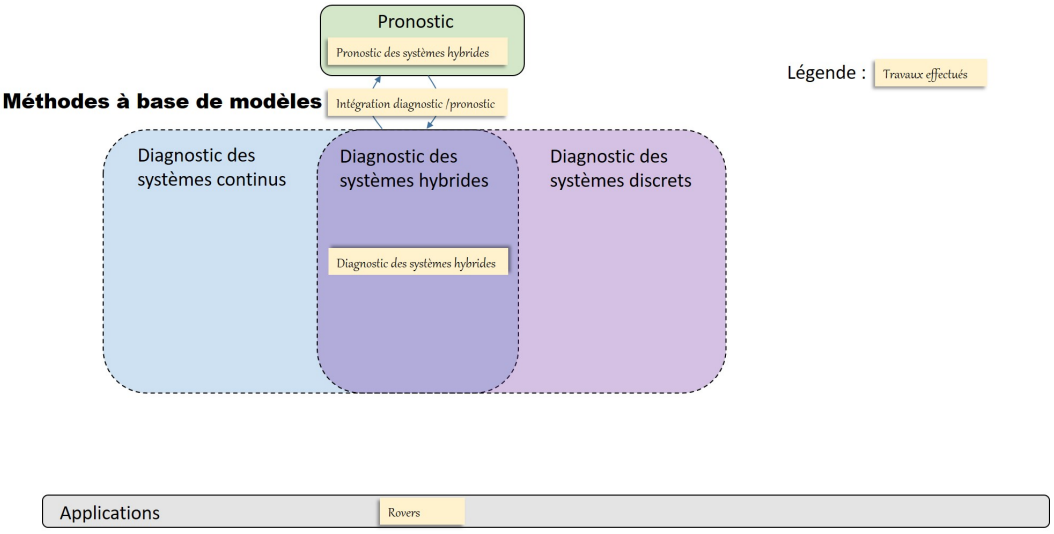


FIGURE 5.9 – Travaux effectués dans l'axe Diagnostic et Pronostic.

les problèmes de diagnostic et de pronostic dans le cadre de la thèse de Quentin Gaudel, co-encadré avec Pauline Ribot. Ce travail a levé le verrou concernant l'étude des formalismes de diagnostic et de pronostic. Nous avons ensuite proposé un algorithme de diagnostic dont les résultats sont interprétables par le module de pronostic. Enfin, nous avons proposé des pistes pour que le pronostic puisse enrichir les entrées du module de diagnostic. Ce travail algorithmique, suivi d'une application dans un cas réel de rover, a permis de lever le troisième verrou identifié.

Le schéma 5.9 résume les travaux effectués dans cet axe.

5.5.2 Contributions sur les liens Diagnostic et Optimisation

Le deuxième axe scientifique concerne les liens entre les modules de diagnostic et l'optimisation. Pour répondre aux verrous associés à cet axe, j'ai eu pour objectif de mixer les compétences que j'avais acquises lors de ma thèse en planification/optimisation et mes nouvelles connaissances en diagnostic. Le diagnostic actif a ainsi pour objectif d'élaborer un plan d'actions dans le but non pas de réaliser une mission seule, mais aussi de récupérer des observations supplémentaires pour raffiner des diagnostics ambigus. Le **développement d'algorithmes efficaces de diagnostic actif, basés sur une formalisation mathématique du problème de diagnostic et de planification associé** répond au premier verrou sur lequel j'ai travaillé. Ce travail, démarré dès 2006 se poursuit encore maintenant. Le degré de maturité des recherches a beaucoup évolué, trouvant son application industrielle en 2016 lors d'un projet R&T du CNES.

Concernant le verrou de l'**embarquabilité des algorithmes de diagnostic**, nous avons travaillé avec Yannick Pencolé sur l'élaboration d'algorithmes de diagnostic anytime. Ces algorithmes ont la propriété de pouvoir être interrompus et/ou de donner un résultat qui s'améliore au cours du temps. J'ai également collaboré avec Emmanuel Bénazéra pour étudier le bien-fondé de l'utilisation des Processus Décisionnels de Markov Partiellement Observables (PDMPO), très connus dans le monde de la planification, en vue de l'intégration directe des tâches de diagnostic, d'observation, de planification et de réparation. Enfin, nous avons travaillé avec Pauline Ribot lors de l'encadrement d'un stage de M2R avec Frédéric Chatrie sur le diagnostic d'un système à base de modèles adaptatifs, qui rentre également dans cette thématique.

Enfin, un dernier travail lié à mes compétences en planification et optimisation a été de contribuer à **optimiser la sélection de tests pour le diagnostic**. Les travaux dans le cadre de la thèse de Saurabh Indra, co-encadrée avec Louise Travé-Massuyès, puis la thèse de Gustavo Pérèz, en co-encadrement avec Louise Travé-Massuyès et Javier Sotomajor, ont permis de travailler sur la décentralisation de la sélection de tests. Des travaux récents avec l'équipe ROC, notamment avec Christian Artigues et Nicolas Jozevowiez, dans le cadre du co-encadrement de stage de master d'Asma Gasmi, permettent de lier les disciplines de l'optimisation et du diagnostic.

☞ Ce travail est également lié au troisième axe de recherche sur le diagnostic distribué. Il sera détaillé dans cette troisième partie, même si toutes les pistes investiguées ne sont pas inscrites dans un cadre distribué.

Le schéma 5.10 résume les travaux effectués dans cet axe.

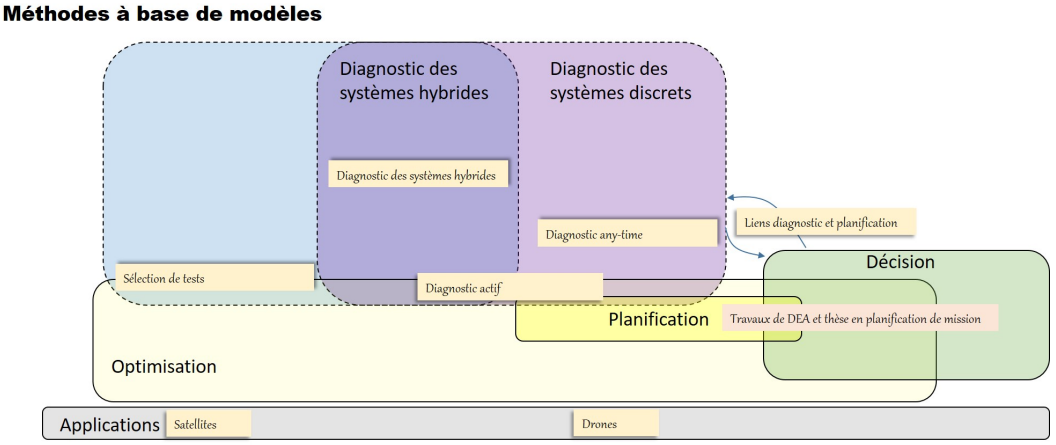


FIGURE 5.10 – Travaux effectués dans l'axe Diagnostic et Optimisation.

5.5.3 Contributions sur le Diagnostic distribué

Des travaux de thèse (Saurahb Indra et Gustavo Pérèz) ont permis de proposer des solutions pour lever les verrous associés à cet axe. La thèse de Saurahb Indra, co-encadrée avec Louise Travé-Massuyès dans le cadre d'une thèse co-financée par le CNES et Thalès Alénia Space, a proposé l'**analyse structurelle comme solution pour la génération de tests dans le cadre de systèmes complexes**. L'analyse structurelle est basée sur une abstraction du modèle qui ne conserve que les liens entre variables et équations. Malgré son apparente simplicité, l'analyse structurelle fournit un ensemble d'outils puissants, s'appuyant sur la théorie des graphes, pour analyser et inférer des informations sur le système. Par ailleurs, elle a l'avantage de s'appliquer indifféremment sur les systèmes linéaires ou non linéaires. L'isolation à la demande a été proposée et constitue une contribution majeure de la thèse pour les systèmes décentralisés.

La thèse de Gustavo Pérèz, co-encadrée avec Louise Travé-Massuyès et Javier Sotomajor de l'Université pontificale catholique du Pérou (PUCP) dans le cadre d'une thèse en co-tutelle, est allée jusqu'à formuler et résoudre le problème d'optimisation lié au choix d'un sous-ensemble de tests de diagnostic au niveau des sous-systèmes permettant une diagnosticabilité maximale pour le système global dans le cas décentralisé et dans le cas distribué, répondant ainsi au verrou de **développement de méthodes de choix de tests pertinents**.

Le schéma 5.11 résume les travaux effectués dans cet axe.

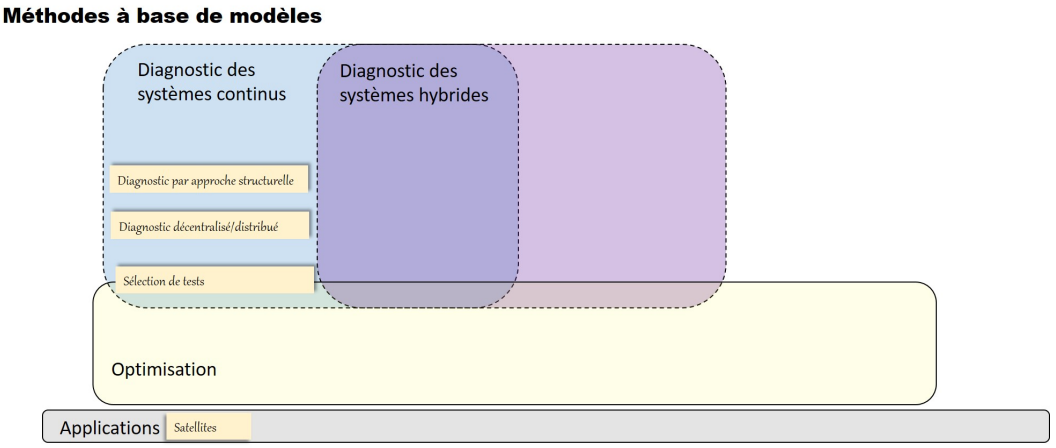


FIGURE 5.11 – Travaux effectués dans l'axe Diagnostic distribué.

5.5.4 Vue d'ensemble de mes travaux

Le schéma 5.12 donne une vue d'ensemble de mes travaux.

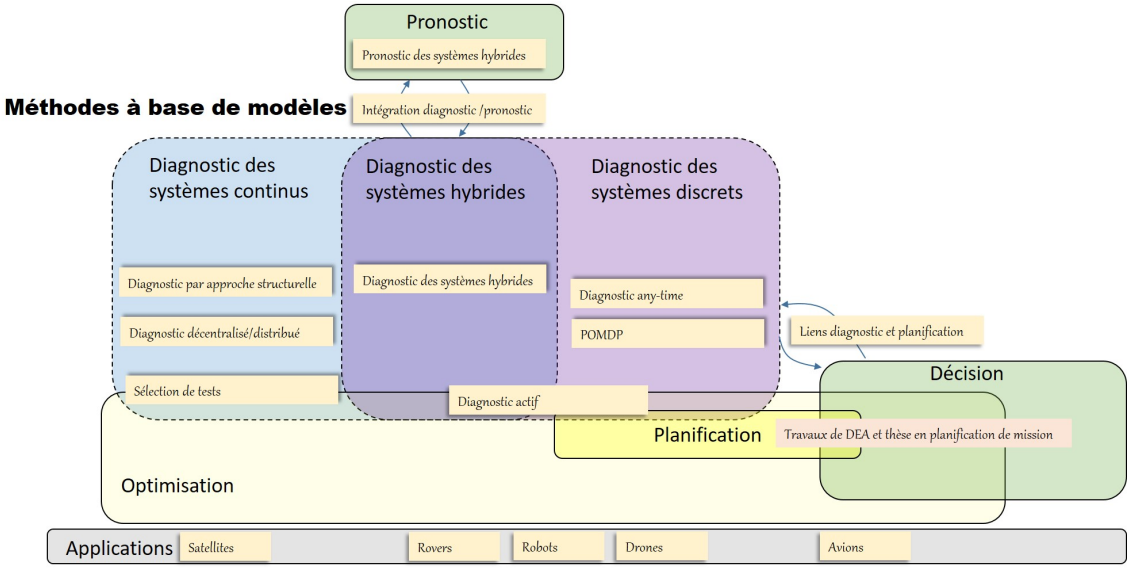


FIGURE 5.12 – Vue d'ensemble des travaux.

⇨ Implémentations

D'un point de vue implémentation, il est intéressant de noter qu'un certain nombre de travaux ont donné lieu à l'élaboration d'un logiciel sous Matlab, avec Simulink, appelé HyDiag [1]. Il a été initialement conçu lors de la thèse de M. Bayoudh. Il a subi de nombreuses modifications durant ces dix dernières années et est actuellement un logiciel conçu pour simuler et diagnostiquer l'état de santé des systèmes hybrides. Une extension au pronostic a été développée dans le cadre du stage de Saïd Zabi co-encadré avec Pauline Ribot. Une extension au diagnostic actif, appelée ActHyDiag, a permis de mettre en œuvre les algorithmes de diagnostic actif développés au cours de nos travaux, notamment en collaboration avec Thalès Alénia Space et le CNES (travail avec Louise Travé-Massuyès, Yannick Pencolé et Nicolas Garin).

1. http://projects.laas.fr/hydiag/

Ces travaux ont donné lieu aux publications suivantes.

▶ M. Maiga, E. Chanthery et L. Travé-Massuyès. *Hybrid system diagnosis: Test of the diagnoser HYDIAG on a benchmark of the international diagnostic competition DXC'2011.* IFAC Proceedings Volumes, vol. 45, no. 20, pages 271 – 276, 2012. 8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes

▶ E. Chanthery, Y. Pencolé, P. Ribot et L. Travé-Massuyès. *HYDIAG: extended diagnosis and prognosis for hybrid systems.* In the 26th International Workshop on Principles of Diagnosis (DX-2015), 2015

⇨ **Applications**

Un point central de mes travaux de recherche est qu'ils sont pour la plupart guidés par les applications. Mon travail de thèse avait pour objectif de planifier des missions pour des drones et j'ai toujours à cœur de travailler dans un cadre applicatif. Assez larges, les domaines d'application de mes travaux vont des satellites aux avions, en passant par les rovers, les robots et les drones. Le point commun entre toutes ces applications reste la volonté d'augmenter l'autonomie de systèmes complexes. Mes collaborations avec la NASA (Matthew Daigle) aux USA ou avec l'Université Technologique du Queensland (QUT) (Thierry Peynot) en Australie sont basées sur des intérêts applicatifs (rovers, robots), comme on peut le voir sur la figure 5.12.

⇨ **Transversalité des types de systèmes**

Par ailleurs, une particularité de mes travaux se situe sur la transversalité des types de systèmes étudiés. De formation généraliste en automatique, je me suis attelée à l'étude des systèmes de nature continue, mais également des systèmes à événements discrets, ce qui m'a conduit tout naturellement à travailler sur les systèmes complexes de nature hybride, dont les dynamiques peuvent être de type continu et discret. Ceci est également illustré sur la figure 5.12.

5.6 Valorisations et Partenariats

La liste ci-après trace les partenariats scientifiques et les projets majeurs auxquels j'ai participé.

Diagnostic actif par OBCP (2014-2015)
Partenaires : Thales Alenia Space (Cannes), LAAS-CNRS (Toulouse)
Financement : Action 2013 de Recherche et Technologie des Systèmes Spatiaux du CNES (axe Bord / Sol : ingénierie système, logiciel de vol et simulation)
Thème : Etude de l'intérêt du diagnostic actif dans l'analyse des causes de défaillances d'un système spatial, et de la possibilité d'embarquer des fonctions de diagnostic actif à l'aide du mécanisme d'OBCP (On-Board Control Procedures)
Contribution : développement de stratégie de diagnostic actif pour un satellite à base d'OBCP
Rôle : responsable scientifique pour le LAAS

DOPEC : Dispositif pour l'OPtimisation de l'Emploi de Capteur (2009-2010)
Partenaires : EADS DS (Elancourt), INRIA (Rennes), LAAS-CNRS (Toulouse)
Financement : PEA de la Direction Générale de l'Armement (DGA)
Contribution : optimisation de l'emploi de capteurs
Rôle : responsable scientifique pour le LAAS

ROSACE : RObots et Systèmes Auto-adaptatifs Communicants Embarqués (2008-2012)
Partenaires : ONERA (Toulouse), IRIT (Toulouse), LAAS-CNRS (Toulouse)
Financement : RTRA STAE
Contribution : diagnostic d'une architecture multi-agents

SIRASAS Stratégies Innovantes et Robustes pour l'Autonomie des Systèmes Aéronautiques et Spatiaux (2007-2010)
Partenaires : IMS (Bordeaux), ONERA (Toulouse), Thales Alenia Space (Cannes), LAAS-CNRS(Toulouse), SATIE (ENS Cachan), CRAN (Nancy), CNES (Toulouse), Airbus France (Toulouse), LRI, Univ. Paris-Sud (Orsay)
Financement : Fondation de Recherche pour l'Aéronautique et l'Espace (FRAE)
Contribution : augmenter les capacités de diagnostic embarqué à bord d'un satellite, lien entre diagnostic et planification

AGATA Architecture Générique pour l'Autonomie, Tests et Applications (2006-2009)
Partenaires : ONERA (Toulouse), CNES (Toulouse)
Financement : Programme commun (CNES-ONERA) auquel a participé le LAAS-CNRS
Contribution : diagnostic actif appliqué à un satellite, lien entre diagnostic et planification

6 Diagnostic et Pronostic

Résumé

Ce chapitre présente mes contributions sur les liens entre le diagnostic et le pronostic dans le cadre des systèmes hybrides. Après une présentation de la vision globale de l'architecture de gestion de santé proposée (section 6.3), une présentation des réseaux de Petri hybrides particulaires est faite (section 6.4). Les méthodes de diagnostic et de pronostic sont ensuite présentées (section 6.5). Leur implémentation et des tests sur le cas réel d'un rover de la NASA ont mené à des conclusions qui sont exposées dans la dernière partie du chapitre (section 6.6). Ces travaux trouvent principalement leurs sources dans l'encadrement du stage de Saïd Zabi en 2013 et dans la direction de la thèse de Quentin Gaudel entre 2013 et 2016, tous deux co-encadrés avec Pauline Ribot.

Sommaire

6.1 Introduction

Ce premier axe, dont les verrous ont été exposés page 38, concerne les liens entre les modules de diagnostic et de pronostic.

L'activité a débuté en 2013 dans le cadre d'un stage de M2R avec Saïd Zabi, en co-encadrement avec Pauline Ribot. Ce stage a permis de proposer une première version de notre **architecture de gestion de santé**. Cette architecture a été retravaillée par la suite. Une vision globale est présentée dans la section 6.3.

Le deuxième verrou vise l'uniformisation des formalismes de diagnostic et de pronostic. Nous avons ainsi proposé les réseaux de Petri hybrides particulaires (ou HPPN pour Hybrid Particle Petri Nets) comme cadre commun de modélisation pour les problèmes de diagnostic et de pronostic. Cette proposition a été effectuée dans le cadre de la thèse de Quentin Gaudel, co-encadrée avec Pauline Ribot. La présentation des HPPN est faite en section 6.4.

Le troisième verrou est le développement d'algorithmes de diagnostic et de pronostic intrinsèquement liés. C'est ce que nous avons proposé dans la suite de notre travail en

section 6.5 via la génération d'un diagnostiqueur HPPN et d'un pronostiqueur HPPN. Cette génération est effectuée dans le souci de prendre en compte les incertitudes inhérentes aux systèmes réels, mais aussi de prendre en compte les contraintes des systèmes embarqués (taille mémoire des objets utilisés, temps de calculs limités, etc). La section 6.6 a pour but de tirer des conclusions sur ces travaux au vu des résultats obtenus et d'en envisager les perspectives.

Personnes impliquées dans cette thématique et affiliation lors de la collaboration : Pauline Ribot (MCF, DISCO), Quentin Gaudel (Doctorant, DISCO), Saïd Zabi (Stagiaire, DISCO), Matthew Daigle (NASA)

6.2 Concepts généraux

Ce travail s'inscrit dans le cadre des systèmes hybrides, qui ont été définis par [Henzinger 1996].

Définition 12 (Système hybride) *Un système hybride est un système dynamique faisant intervenir explicitement et simultanément des dynamiques de type continu et discret.*

Les systèmes hybrides sont généralement décrits comme des systèmes multimodes composés d'un système à événements discrets (SED) sous-jacent représentant les changements de modes et diverses dynamiques continues sous-jacentes associées à chaque mode [Bayoudh *et al.* 2008].

Définition 13 (État discret, état continu) *L'état discret du système est défini comme l'état discret actuel du SED sous-jacent. L'évolution de l'état continu du système dépend de la dynamique continue associée au mode actuel du système.*

Les données des capteurs et les commandes sont considérées comme des observations continues ou discrètes sur le système.

Dans la plupart des systèmes industriels, si la dégradation n'est pas observable, elle est estimée comme une probabilité d'occurrence de fautes. La dégradation peut dépendre du niveau de stress du mode de santé actuel du système et, dans certains cas, de l'état continu courant et même de l'analyse des événements survenus sur le système. En raison de ces dépendances et de son importance pour le PHM, nous avons choisi d'évaluer la dégradation séparément de l'état discret et de l'état continu du système [Gaudel *et al.* 2015b].

Définition 14 (état de dégradation) *L'état de dégradation du système est la valeur courante de la dégradation dont l'évolution est représentée par la dynamique de dégradation.*

Nous avons étendu le système multi-mode en associant une dynamique de dégradation sous-jacente à chaque mode.

Définition 15 (Mode, Événement, État) *Un mode est défini comme une combinaison d'un état discret du SED avec une dynamique continue et une dynamique de dégradation. Les changements de modes sont associés à des occurrences d'événements discrets. L'état du système hybride est défini comme la combinaison de ses états discrets, continus et de dégradation.*

Plus formellement, l'ensemble $E = E_o \cup E_{uo}$ des labels d'événements du système est l'union de l'ensemble E_o des labels des événements observables et l'ensemble E_{uo} des labels des événements non observables sur le système. Un événement est défini comme un couple $e = (v, k)$ où $v \in E$ est un label (ou type) d'événement et $k \in \mathbb{R}$ est la date d'occurrence de e. Un événement (v, k) est dit non observable si au temps k, $v \in E_{uo}$.

Dans le cadre du pronostic, on s'attache également à deux notions : la fin de vie (ou EOL pour End Of Life) et la durée de vie résiduelle (ou RUL pour Remaining Useful Life).

Définition 16 (Fin de vie (EOL)) *La fin de vie (End Of Life ou EOL) est la date à laquelle le système n'est plus opérationnel.*

Définition 17 (Durée de vie résiduelle (RUL)) *La durée de vie résiduelle (Remaining Useful Life ou RUL) est la durée restante jusqu'à la fin de vie d'un système.*

6.3 Une vision globale de l'architecture de gestion de santé

Comme il a été présenté dans la section 5.3, page 29, il n'existait à notre connaissance aucune architecture de gestion de santé où le couplage diagnostic/pronostic était bilatéral.

Dans [Chanthery & Ribot 2013] et [Zabi *et al.* 2013], nous avons proposé avec Pauline Ribot et Saïd Zabi une architecture dans laquelle les méthodes de diagnostic et de pronostic sont intégrées dans un processus unique appelé *InterDP*. Les résultats du pronostic sont utilisés pour lever les ambiguïtés des résultats du diagnostic. L'utilisation des résultats de diagnostic et de pronostic dans un module de planification n'est cependant pas abordée.

Une vision globale de l'architecture de santé que nous avons finalement proposée durant la thèse de Quentin Gaudel [Gaudel 2016], co-encadrée avec Pauline Ribot, est illustrée sur la figure 6.1.

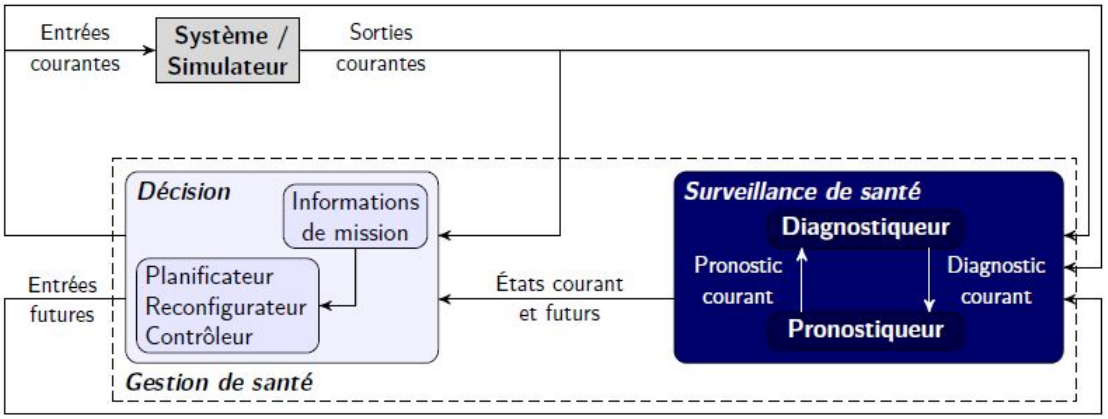


Figure 6.1 – Architecture de gestion de santé de système intégrant diagnostic et pronostic.

Un module de décision permet la replanification de la mission du système, sa reconfiguration et/ou encore son contrôle de bas niveau. Un module de surveillance de santé inclut les fonctions de diagnostic et de pronostic. En plus des données de sortie du système, chacun des deux modules de décision et de surveillance de santé utilise les données calculées par l'autre. Le diagnostiqueur (*diagnoser*) est le module qui calcule le diagnostic du système. Le pronostiqueur (*prognoser*) est le module qui calcule le pronostic du système.

Dans un contexte de surveillance de santé, l'objectif du pronostic est de prédire les états futurs du système et son RUL/EOL, à partir du diagnostic courant et des entrées futures sur le système.

On s'intéresse particulièrement à déterminer si et quand le système va entrer dans un mode de défaillance durant un horizon de prédiction τ_p, choisi par l'opérateur. Le diagnostiqueur utilise le pronostic et le pronostiqueur utilise le diagnostic, illustrant l'interaction des deux modules. L'ensemble des entrées et sorties du système, discrètes et continues, est un ensemble d'observations du point de vue du module de surveillance de santé et sera désigné comme tel dans la suite du document.

6.4 Les réseaux de Petri hybrides particulaires : un cadre de modélisation commun pour le diagnostic et le pronostic

Le formalisme des réseaux de Petri hybrides particulaires (ou HPPN pour Hybrid Particle Petri Nets), proposé initialement dans [Gaudel *et al.* 2014a], enrichit les réseaux de Petri avec des dynamiques continues et des dynamiques de dégradation. Ils ont l'avantage de pouvoir exprimer à la fois des contraintes pour des systèmes à événements discrets et pour des systèmes continus mais aussi de spécifier des contraintes sur les dégradations que subit le système. Par ailleurs, les HPPN gèrent les incertitudes sur la connaissance du système et sur les observations. La gestion des incertitudes a particulièrement été traitée dans [Gaudel *et al.* 2015a].

6.4.1 Structure des HPPN et règles de tirages

La définition 18 définit la structure complète d'un réseau de Petri hybride particulaire puis la suite de la section décrit chaque élément qui la compose. Ce travail a été initié dans [Gaudel *et al.* 2014b]. Une description détaillée et des exemples peuvent être trouvés dans la thèse [Gaudel 2016].

Définition 18 (HPPN) *Un HPPN est un 11-uplet* $\langle P, T, A, \mathcal{A}, E, X, D, \mathcal{C}, \mathcal{D}, \Omega, \mathbb{M}_0 \rangle$ *réunissant les informations pour décrire des évolutions discrètes (avec les places symboliques), continues (avec les places numériques) et les évolutions de type dégradation (avec les places de dégradation), et les relations des unes avec les autres :*

- P *est l'ensemble des places, réunissant les places symboliques* P^S*, les places numériques* P^N *et les places de dégradation* P^D*,* $P = P^S \cup P^N \cup P^D$*,*
- T *est l'ensemble des transitions,*
- $A \subset (P \times T \cup T \times P)$ *est l'ensemble des arcs,*
- $\mathcal{A}$ *est l'ensemble des annotations des arcs,*
- E *est l'ensemble des labels d'événements,*
- $X \subset \mathbb{R}^{n_N}$ *est l'espace d'état du vecteur d'état continu, avec* $n_N \in \mathbb{N}_+$ *le nombre fini de variables d'état continues,*
- $D \subseteq \mathbb{R}^{n_D}$ *est l'espace d'état du vecteur d'état de dégradation, avec* $n_D \in \mathbb{N}_+$ *le nombre fini de variables d'état de dégradation,*
- $\mathcal{C}$ *est l'ensemble des dynamiques continues associées aux places numériques,*
- $\mathcal{D}$ *est l'ensemble des dynamiques de dégradation associées aux places de dégradation,*
- Ω *est l'ensemble des conditions associées aux transitions,*
- $\mathbb{M}_0$ *est le marquage initial du réseau.*

Une représentation graphique d'un HPPN simple est présentée en figure 6.2. On y voit 3 places de type différent p_1^S, p_1^N et p_1^D en amont d'une transition t à laquelle est associée

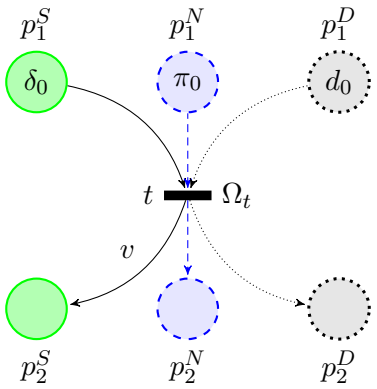


FIGURE 6.2 – Exemple illustratif d'un HPPN au temps $k = 0$.

une condition Ω_t, et 3 places en aval p_2^S, p_2^N et p_2^D. Sur la flèche partant de t et allant vers la place p_2^S, v est le label d'un événement.

Les places d'un HPPN sont marquées avec des jetons. La particularité des jetons d'un HPPN est que chacun d'eux porte une valeur.

Les places symboliques P^S représentent les états discrets du système et sont marquées avec des jetons symboliques, appelés configurations. L'ensemble des configurations au temps k est noté M_k^S. Chaque configuration du HPPN porte la trace de l'ensemble des événements qui sont survenus sur le système jusqu'au temps k et dont l'occurrence explique l'existence de la configuration. De manière plus précise, une configuration $\delta_k \in M_k^S$ est un jeton au temps k dont la valeur est l'ensemble b_k des évènements qui sont survenus jusqu'au temps k : $b_k = \{(v, \kappa) | \kappa \leq k\}$.

Les places numériques P^N représentent les dynamiques continues du système et les incertitudes associées. À chaque place numérique $p^N \in P^N$ est associé un ensemble d'équations $C_{p^N} \in \mathcal{C}$ modélisant une dynamique continue du système et les bruits associés (incertitudes sur l'évolution et sur les mesures) :

$$C_{p^N} = \left\{ \begin{array}{rcl} x_{k+1} & = & \mathbf{f}(x_k, u_k) + \mathbf{v}(x_k, u_k) \\ y_k & = & \mathbf{h}(x_k, u_k) + \mathbf{w}(x_k, u_k) \end{array} \right. , \tag{6.1}$$

où $x_k \in X$ est le vecteur d'état continu, $u_k \in \mathbb{R}^{n_u}$ est le vecteur des n_u variables d'entrée continues, $\mathbf{f}$ est la fonction d'évolution continue non bruitée, $\mathbf{v}$ est la fonction de bruit de l'évolution continue, $y_k \in \mathbb{R}^{n_y}$ est le vecteur des n_y variables de sortie continues, $\mathbf{h}$ est la fonction de sortie continue non bruitée, et $\mathbf{w}$ est la fonction de bruit de la sortie continue. Les fonctions $\mathbf{f}$, $\mathbf{v}$, $\mathbf{h}$ et $\mathbf{w}$ sont dépendantes de la place p^N considérée. Les places numériques sont marquées avec des jetons numériques, appelés particules. L'ensemble des particules au temps k est noté M_k^N. Une particule $\pi_k \in M_k^N$, est un jeton dont la valeur est un état continu possible $x_k \in X$ du système au temps k.

Les places de dégradation P^D représentent les dynamiques de dégradation du système et les incertitudes associées. À chaque place de dégradation $p^D \in P^D$ est associé un ensemble d'équations $D_{p^D} \in \mathcal{D}$ modélisant une dynamique de dégradation du système :

$$\mathtt{D}_{p^D} = \left\{ \mathtt{d}_{k+1} = \mathbf{g}(\mathtt{d}_k, b_k, x_k, u_k) + \mathbf{z}(\mathtt{d}_k, b_k, x_k, u_k) \right. , \tag{6.2}$$

où $\mathtt{d}_k \in D$ est le vecteur d'état de dégradation, $\mathbf{g}$ est la fonction d'évolution de dégradation non bruitée, et $\mathbf{z}$ est la fonction de bruit de l'évolution de dégradation. Les fonctions $\mathbf{g}$ et $\mathbf{z}$ sont dépendantes de la place p^D considérée.

Important : Rappelons ici que la différence entre les places continues et les places de dégradation tient dans le fait que les états de dégradation du système sont fonction de l'état continu du système mais également de l'ensemble des événements b_k qui sont survenus au temps k.

Les places de dégradation sont marquées avec des jetons de dégradation. L'ensemble des jetons de dégradation au temps k est noté M_k^D. Un jeton de dégradation $d_k \in M_k^D$ lie une configuration δ_k et une particule π_k, et sa valeur est un état de dégradation possible $d_k \in D$ au temps k.

L'ensemble des places du HPPN est donc :

$$P = P^S \cup P^N \cup P^D = \{p_1^S, ..., p_{\mathrm{s}}^S\} \cup \{p_1^N, ..., p_{\mathrm{n}}^N\} \cup \{p_1^D, ..., p_{\mathrm{d}}^D\}, \tag{6.3}$$

où s, n et d sont respectivement les nombres des places symboliques, numériques et de dégradation. Par exemple, dans la figure 6.2, on a $P = \{p_1^S, p_2^S, p_1^N, p_2^N, p_1^D, p_2^D\}$.

Soit M_k l'ensemble des jetons du HPPN au temps k :

$$M_k \quad = \quad M_k^S \cup M_k^N \cup M_k^D, \tag{6.4}$$

où M_k^S, M_k^N et M_k^D sont respectivement les ensembles des configurations, particules, et jetons de dégradation au temps k. Dans la figure 6.2, on a $M_0 = \{\delta_0, \pi_0, d_0\}$.

Le marquage $\mathbb{M}_k$ d'un HPPN au temps k est la distribution de ses jetons dans ses places :

$$\mathbb{M}_k \quad = \quad \mathbb{M}_k^S \cup \mathbb{M}_k^N \cup \mathbb{M}_k^D, \tag{6.5}$$

où $\mathbb{M}_k^S \in (2^{M_k^S})^{\mathrm{s}}$, $\mathbb{M}_k^N \in (2^{M_k^N})^{\mathrm{n}}$ et $\mathbb{M}_k^D \in (2^{M_k^D})^{\mathrm{d}}$, sont respectivement les marquages symbolique, numérique et de dégradation au temps k. Dans la figure 6.2, on a :

$$\begin{array}{rcl} \mathbb{M}_0^S & = & [[\delta_0] \quad \emptyset], \\ \mathbb{M}_0^N & = & [[\pi_0] \quad \emptyset], \\ \mathbb{M}_0^D & = & [[d_0] \quad \emptyset]. \end{array}$$

Le marquage initial $\mathbb{M}_0$ d'un HPPN représente donc les conditions initiales du système (état initial, ensemble des événements ayant eu lieu jusqu'au temps 0).

Les notions d'hypothèse et de cluster de particules ont été définies pour les besoins du diagnostic et du pronostic (suivi d'état par un HPPN).

Définition 19 (Hypothèse) *On définit une hypothèse à l'instant k comme un ensemble composé d'une configuration δ_k, de tous les jetons de dégradation $\{d_k^i | i \in \{1, ..., n_k\}\}$ liés à δ_k, et de chaque particule π_k^i liée à d_k^i à l'instant k. Un tel ensemble de jetons $\{\delta_k, \pi_k^1, ..., \pi_k^{n_k}, d_k^1, ..., d_k^{n_k}\}$ contient la connaissance sur l'état à l'instant k et les événements survenus sur le système.*

Par exemple, si l'ensemble b_0 des événements ayant eu lieu jusqu'au temps 0 et les états continu x_0 et de dégradation d_0 sont initialement connus précisément, l'ensemble initial de jetons $M_0 = \{\delta_0, \pi_0, d_0\}$, où d_0 lie δ_0 et π_0, est l'unique hypothèse initiale.

Si les conditions initiales sont ambiguës, l'ensemble initial de jetons peut être l'union de plusieurs hypothèses, e.g. $M_0 = \{\delta_0^1, \pi_0^1, d_0^1\} \cup \{\delta_0^2, \pi_0^2, d_0^2\}$, où d_0^1 lie δ_0^1 et π_0^1, et d_0^2 lie δ_0^2 et π_0^2.

À un instant k, une hypothèse peut aussi contenir plusieurs particules et jetons de dégradation qui représentent une connaissance imprécise sur l'état continu et sur l'état

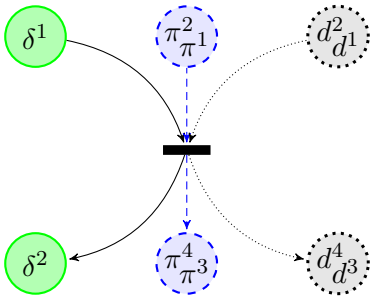


FIGURE 6.3 – Illustration des clusters de particules.

de dégradation, e.g. $\{\delta_k^1, \pi_k^1, ..., \pi_k^{n_k}, d_k^1, ..., d_k^{n_k}\}$, où $n_k \in \mathbb{N}_+$ est le nombre de particules utilisées pour représenter l'état continu, et où les n_k jetons de dégradation lient les n_k particules à la configuration δ_k^1. Le nombre n_k de particules et de jetons de dégradation est donc représentatif de la précision de l'hypothèse au temps k.

DÉFINITION 20 (CLUSTER DE PARTICULES) *L'ensemble des n_k particules liées à une même configuration à travers n_k jetons de dégradation est appelé un cluster de particules.*

Dans la figure 6.3, d^1 et d^2 lient respectivement π^1 et π^2 à δ^1, et, d^3 et d^4 lient respectivement π^3 et π^4 à δ^2. Il y a donc deux hypothèses, $\{\delta^1, \pi^1, \pi^2, d^1, d^2\}$ et $\{\delta^2, \pi^3, \pi^4, d^3, d^4\}$, et deux clusters de particules, $\{\pi^1, \pi^2\}$ et $\{\pi^3, \pi^4\}$.

Les arcs connectent de manière directionnelle des places et des transitions. Les entrées d'une transition $t \in T$ sont les places desquelles part un arc pointant vers t, et les sorties de t sont les places pointées par un arc ayant pour origine t. L'ensemble des places en entrée (resp. en sortie) de t est désigné par ${}^\circ t$ (resp. t°). Les places en entrée (et en sortie) d'une transition peuvent être de types différents.

C'est le cas pour la transition t de la figure 6.2, pour laquelle ${}^\circ t = \{p_1^S, p_1^N, p_1^D\}$ et $t^\circ = \{p_2^S, p_2^N, p_2^D\}$.

Le tirage d'une transition $t \in T$ est conditionné par un jeu de conditions $\Omega_t \in \Omega$ associé à t. Le jeu Ω_t est composé d'autant de conditions qu'il y a de places en entrée de t :

$$\forall t \in T, \quad |\Omega_t| = |{}^\circ t|. \tag{6.6}$$

Par exemple, si t a une place de chaque type en entrée, son ensemble de conditions est $\Omega_t = \langle \omega_t^S, \omega_t^N, \omega_t^D \rangle$.

Une condition $\omega : M_k \to \mathbb{B}$, avec $\mathbb{B} = \{\top, \bot\}$ (l'ensemble des valeurs logiques VRAI et FAUX), peut être soit un test sur la valeur d'un jeton, soit toujours satisfait ($\top$), ou jamais satisfait ($\bot$). Une condition symbolique ω_t^S peut donc valoir $\top$ ou $\bot$, ou elle peut tester l'occurrence d'un événement labellisé $v \in E$ (faute, événement de mission, interaction avec l'environnement, etc.). Dans ce dernier cas, elle prend la forme $\omega_t^S(\delta_k) = occ(b_k, v)$, qui teste si l'ensemble d'événements b_k de δ_k contient l'événement (v, k). Avec le même raisonnement, une condition numérique ω_t^N (resp. une condition de dégradation ω_t^D) peut valoir $\top$ ou $\bot$ ou être une contrainte sur l'état continu (resp. l'état de dégradation) du système. Dans le dernier cas, $\omega_t^N(\pi_k) = c(x_k)$ est un test sur le vecteur d'état continu x_k de π_k.

Les travaux de thèse de Quentin Gaudel ont amené à définir des règles de sensibilisation et de franchissement pour les transitions.

Définition 21 (Jeton accepté) *On dit qu'un jeton est accepté par les conditions Ω_t d'une transition t au temps k s'il respecte la condition de son type. L'ensemble des jetons acceptés par les conditions Ω_t au temps k est noté $\mathcal{S}_k^t$.*

Cette notion est formellement définie dans la thèse.

Définition 22 (Transition tirable) *Une transition $t \in T$ est tirable (ou franchissable) au temps k s'il existe au moins un jeton de chacune de ses places en entrée accepté par les conditions Ω_t :*

$$\forall p \in {}^\circ t, \quad |\mathcal{S}_k^t(p)| > 0. \tag{6.7}$$

Définition 23 (Tirage d'une transition) *Pour chaque type de place en entrée et en sortie de t, le tirage de $t \in T$ à l'instant k est défini par :*
$\forall P^o \in \{P^S, P^N, P^H\}, p \in P^o \cap {}^\circ t, p' \in P^o \cap t^\circ$,

$$\begin{array}{rcl} M_{k+1}(p) & = & M_k(p) \backslash \mathcal{S}_k^t(p), \\ M_{k+1}(p') & = & M_k(p') \cup \mathcal{S}_k^t(p), \end{array} \tag{6.8}$$

où $\mathcal{S}_k^t(p)$ est l'ensemble des jetons de $\mathcal{S}_k^t$ qui sont dans la place p.

Les travaux ont également amené à définir la propriété suivante.

Propriété 1 (Conservation des valeurs des jetons et liens) *Lors du tirage d'une transition, les jetons acceptés sont déplacés, leurs liens sont conservés, et leurs valeurs sont conservées ou mises à jour.*

Important : La propriété 1 constitue la différence majeure des HPPN avec les réseaux de Petri ordinaires, dans lesquels des jetons sont consommés (détruits) dans les places en entrée de la transition avant que de nouveaux jetons soient créés dans les places en sortie de la transition. La conservation des valeurs des jetons existe dans certaines extensions des réseaux de Petri, comme les réseaux de Petri colorés [David & Alla 2005] par exemple. En revanche, la présence de liens entre les jetons et leur déplacement lors du tirage d'une transition est spécifique aux HPPN. Le concept d'annotation, défini dans la thèse, permet de faire évoluer les valeurs des configurations lors du tirage des transitions.

6.4.2 Modélisation d'un système hybride dans le cadre HPPN

Les HPPN sont naturellement utilisables pour modéliser un système hybride.

La méthodologie de modélisation se base sur une description de l'évolution de la santé avec un système multimode, pour lequel à chaque mode sont associées une dynamique continue représentant le comportement du système et une dynamique de dégradation. Chaque mode est représenté par trois places du HPPN : une place symbolique, une place numérique, et une place de dégradation. L'état du système (i.e. les états discret, continu et de dégradation) ainsi que les événements qui sont survenus sont représentés par le marquage du HPPN. Les conditions associées aux transitions permettent de contextualiser les changements de modes du système.

Les places symboliques représentent les différents états discrets de santé du système. Chaque dynamique continue (resp. de dégradation) distincte est associée à une place

numérique (resp. de dégradation). Ainsi, une même place peut participer à la représentation de plusieurs modes. Cependant, deux modes différents ne peuvent pas être représentés avec les deux mêmes places symbolique et numérique.

Les transitions modélisent les changements de modes et les conditions décrivent les circonstances de ces changements. Cela implique qu'une transition $t \in T$ doit avoir une place de chaque type en entrée et une place de chaque type en sortie. Cela signifie également qu'un ensemble de trois conditions $\Omega_t = \langle \omega_t^S, \omega_t^N, \omega_t^D \rangle$ est associé à chaque transition t.

La condition symbolique ω_t^S spécifie le label $v \in E$ de l'événement (observable ou non observable ; faute, interaction avec l'environnement ou commande discrète) qui survient au changement de mode. Si un tel label d'événement existe, l'arc a qui connecte la transition t à sa place symbolique p^S en sortie, est annoté avec v. Si aucun label d'événement n'est associé au changement de mode, la condition symbolique est mise à $\top$.

La condition numérique ω_t^N spécifie la contrainte sur l'état continu qui est satisfaite au changement de mode. Si aucune condition sur l'état continu n'est à satisfaire, la condition numérique est mise à $\top$.

La condition de dégradation ω_t^D spécifie la contrainte sur l'état de dégradation qui est satisfaite au changement de mode. Si aucune condition sur l'état de dégradation n'est à satisfaire, la condition de dégradation est mise à $\bot$.

Un point important pour les HPPN dans le suivi de mode d'un système hybride est que si ω_t^S et ω_t^N sont des conditions qui sont supposées être satisfaites en même temps, ω_t^D, quant à elle, représente une condition alternative au changement de mode.

Important : Il est important de noter qu'une interprétation généralisée est que le système change de mode si la condition logique $(\omega_t^S \wedge \omega_t^N) \vee \omega_t^D$ est satisfaite.

Dans un modèle de santé, une faute anticipée est représentée par un label d'événement $f \in E_{uo}$ non observable. Les conditions de dégradation sont utilisées pour modéliser les changements de modes dus à la dégradation du système. Par exemple, si la dégradation est modélisée par la probabilité d'occurrence d'une faute, une condition sur l'état de dégradation peut être une fonction booléenne qui est satisfaite si cette probabilité est supérieure à 0.9.

Cette modélisation a été appliquée à un exemple de robot mobile et sur un exemple académique de réservoir d'eau, tous deux disponibles dans le manuscrit de thèse de Quentin Gaudel [Gaudel 2016].

6.5 Travail algorithmique pour l'intégration diagnostic/pronostic

6.5.1 Vue d'ensemble

Une vue d'ensemble de la méthode de surveillance de la santé est illustrée sur la figure 6.4. Trois objets différents sont définis dans le cadre HPPN : un modèle HPPN du système hybride noté $HPPN_\Phi$, un diagnostiqueur HPPN noté $HPPN_\Delta$ et un pronostiqueur HPPN noté $HPPN_\Pi$.

L'objet modèle a été décrit dans la section précédente. Les deux objets diagnostiqueur et pronostiqueur seront détaillés dans les prochaines sections.

La première étape hors ligne (à l'intérieur du cadre en pointillés) est la modélisation du système hybride en utilisant le cadre HPPN, comme décrit dans la section 6.4.2. Le

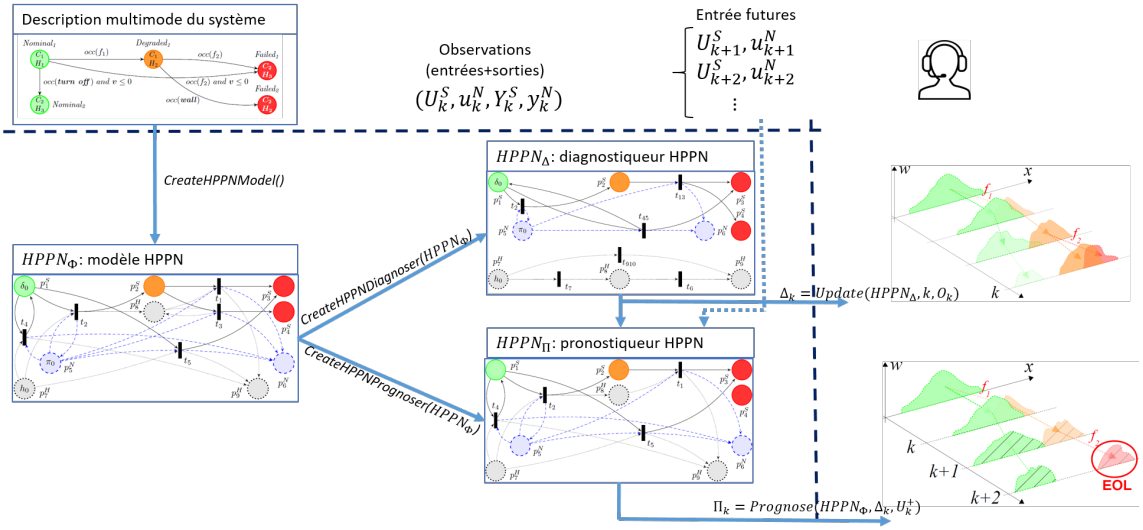


FIGURE 6.4 – Vue d'ensemble de la méthode de surveillance de santé

modèle du système $HPPN_\Phi$ peut être construit à partir d'une description multimode du système ou directement à partir de connaissances d'experts.

La seconde étape hors ligne est la génération automatique d'un diagnostiqueur $HPPP_\Delta$ et d'un pronostiqueur $HPPN_\Pi$ à partir du modèle de système $HPPN_\Phi$.

En ligne (à l'extérieur des pointillés), le diagnostiqueur $HPPN_\Delta$ utilise les différentes observations (commandes, événements discrets, mesures continues) sur le système $O_k = (U_k^S, u_k^N, Y_k^S, y_k^N)$ pour mettre à jour le résultat du diagnostiqueur et calculer le diagnostic Δ_k. Le pronostiqueur $HPPN_\Pi$ prend en entrée le résultat de diagnostic Δ_k ainsi que les entrées futures (symboliques et numériques) et calcule un pronostic Π_k.

6.5.2 Diagnostic basé sur les HPPN

Cette section présente la méthodologie de diagnostic de systèmes hybrides basée sur les HPPN.

Gestion des incertitudes

La méthode de diagnostic a pour avantage de proposer une gestion des performances calculatoires vis-à-vis de la précision des résultats souhaitée et de différentes sources d'incertitudes. Une incertitude sur la connaissance du système doit être considérée car un modèle ne reflète pas parfaitement la réalité, tant pour la partie discrète que continue. Une incertitude sur les observations doit également être prise en compte à cause de l'imprécision inhérente aux capteurs.

On distingue par la suite l'incertitude symbolique, relative au modèle discret et aux observations discrètes, de l'incertitude numérique, qui concerne l'imprécision sur le modèle continu et les valeurs numériques.

⇨ Incertitude symbolique

Au niveau de la connaissance du modèle discret, l'incertitude symbolique se traduit par la présence de séquences d'événements impossibles ou partiellement incorrectes. Concernant les observations discrètes, il est possible qu'un événement soit survenu mais qu'il

n'ait pas été observé, on parle alors d'observation manquante. De même, il est possible d'observer un événement mais que celui-ci n'ait pas physiquement eu lieu ; on parle alors de fausse observation.

Le mécanisme de pseudo-tirage [Lesire & Tessier 2005], [Zouaghi *et al.* 2011], introduit dans [Cardoso *et al.* 1999], est utilisé dans les HPPN pour gérer les incertitudes.

Le pseudo-tirage d'une transition correspond à la duplication des jetons : les jetons dans les places en entrée de la transition ne sont pas déplacés mais sont copiés et leurs copies sont déplacées dans les places en sortie de la transition.

RÈGLE 1 (PSEUDO-TIRAGE D'UNE TRANSITION) *Soit $t \in T$ une transition tirable. Pour chaque type de place en entrée et en sortie de t, le pseudo-tirage de $t \in T$ à l'instant $k-1$ est défini par :*
$\forall P^o \in \{P^S, P^N, P^D\}, p \in P^o \cap {}^\circ t, p' \in P^o \cap t^\circ$,

$$\begin{array}{rcl} M_k(p) & = & M_{k-1}(p), \\ M_k(p') & = & M_{k-1}(p') \cup \mathcal{S}^t_{k-1}(p), \end{array} \tag{6.9}$$

où $\mathcal{S}^t_{k-1}(p)$ est l'ensemble des jetons de $\mathcal{S}^t_{k-1}$ qui sont dans la place p.

Dans le diagnostiqueur, l'incertitude symbolique est gérée à deux niveaux :

- Toutes les transitions symboliques sont mises à VRAI durant la génération du diagnostiqueur. Cela signifie que le pseudo-tirage est utilisé pour ces transitions pour lesquelles les conditions symboliques ont été modifiées ;
- Durant l'étape de prédiction du processus de diagnostic, le diagnostiqueur utilise le pseudo-tirage des transitions pour envisager les occurrences de chaque événement pouvant avoir lieu en accord avec la dynamique discrète. Le pseudo-tirage crée ainsi de nouvelles hypothèses.

⇨ Incertitude numérique

Outre l'écart intrinsèque entre la réalité et le modèle continu du système, l'incertitude numérique concerne l'imprécision des valeurs numériques. C'est une problématique inévitable dans les études de cas réels. L'incertitude numérique est souvent gérée avec un estimateur (ou observateur) [Ding 2014], dont l'objectif est d'estimer l'état continu du système tout en tenant compte des bruits associés au modèle et aux mesures. Dans ces travaux, nous avons choisi d'utiliser des filtres particulaires [van der Merwe *et al.* 2000]. Dans notre contexte, le but d'un filtre particulaire est d'estimer l'état continu du système en se basant seulement sur les données observées. L'utilisation des filtres particulaires est pertinente pour l'estimation à la fois des états discret, continu et de dégradation car la représentation de l'estimation de l'état continu est déjà discrétisée en particules.

Dans ces travaux, un filtre particulaire est appliqué indépendamment à chaque cluster de particules (définition 20) grâce aux liens entre les configurations et les particules, assurés par les jetons de dégradation. Durant l'étape de prédiction du processus de diagnostic, les valeurs des particules évoluent en fonction des dynamiques continues associées aux places numériques auxquelles les particules appartiennent. Ensuite, durant l'étape de correction, chaque cluster de particules est rééchantillonné indépendamment. L'enchaînement de ces deux étapes est illustré sur la figure 6.6.

Génération du diagnostiqueur

La méthode repose sur la génération d'un diagnostiqueur $HPPN_\Delta$ à partir d'un modèle $HPPN_\Phi$. L'objectif est d'optimiser la représentation des connaissances sur l'état du système.

Définition 24 (Diagnostiqueur HPPN) *Soit un modèle HPPN défini par le 11-uplet* $HPPN_\Phi = \langle P_\Phi, T_\Phi, A_\Phi, \mathcal{A}_\Phi, E_\Phi, X_\Phi, D_\Phi, \mathcal{C}_\Phi, \mathcal{H}_\Phi, \Omega_\Phi, \mathbb{M}_{0\Phi}\rangle$ *comme présenté en section 6.4.1. Le diagnostiqueur* $HPPN_\Delta$ *de* $HPPN_\Phi$ *est un 11-uplet généré à partir de* $HPPN_\Phi$ *:*

$$HPPN_\Delta \quad = \quad \langle P_\Delta, T_\Delta, A_\Delta, \mathcal{A}_\Delta, E_\Delta, X_\Delta, D_\Delta, \mathcal{C}_\Delta, \mathcal{H}_\Delta, \Omega_\Delta, \mathbb{M}_{0\Delta}\rangle. \tag{6.10}$$

Les éléments de $HPPN_\Delta$ sont déterminés en 6 étapes.

① Copie du modèle

Le diagnostiqueur doit estimer les états discrets, continus et de dégradation du système. L'étape ① consiste à copier le modèle de système HPPN. En effet, les espaces d'états discrets, continus et de dégradation, ainsi que les dynamiques continues et de dégradation sont les mêmes que ceux du modèle. En conséquence, toutes les places, étiquettes d'événements, espaces d'états et dynamiques restent les mêmes que ceux du modèle $HPPN_\Phi$:

$$P_\Delta = P_\Phi, E_\Delta = E_\Phi, X_\Delta = X_\Phi, D_\Delta = D_\Phi, \mathcal{C}_\Delta = \mathcal{C}_\Phi, \mathcal{D}_\Delta = \mathcal{D}_\Phi. \tag{6.11}$$

Le marquage initial $\mathbb{M}_{0\Delta}$ du diagnostiqueur correspond au marquage initial $\mathbb{M}_{0\Phi}$ du modèle, qui contient la connaissance sur le mode, l'état et les événements survenus sur le système au temps 0. On a donc $\mathbb{M}_{0\Delta} = \mathbb{M}_{0\Phi}$.

② Séparation en niveau de comportement et niveau de dégradation

La seconde étape consiste à séparer le diagnostiqueur en deux niveaux : le niveau de comportement contient les places symboliques et numériques, tandis que le niveau de dégradation contient les places de dégradation. Cette séparation permet une meilleure lecture de la dégradation du système.

Chaque transition $t \in T_\Delta$ est transformée en un couple de nouvelles transitions (t', t''). La transition t' hérite des arcs liant t aux places symboliques et numériques, ainsi que des conditions symbolique et numérique. La transition t'' hérite des arcs liant t aux places de dégradation, ainsi que de la condition de dégradation. On crée ainsi t' et t'' telles que :

$$\begin{array}{rcl} {}^\circ t' & = & {}^\circ t \cap (P^S \cup P^N), \\ t'^\circ & = & t^\circ \cap (P^S \cup P^N), \end{array} \tag{6.12}$$

et :

$$\begin{array}{rcl} {}^\circ t'' & = & {}^\circ t \cap P^D, \\ t''^\circ & = & t^\circ \cap P^D, \end{array} \tag{6.13}$$

avec les conditions $\Omega_{t'}$ et $\Omega_{t''}$ suivantes :

$$\begin{array}{rcl} \Omega_{t'} & = & \langle \omega_t^S, \omega_t^N \rangle, \\ \Omega_{t''} & = & \langle \omega_t^D \rangle. \end{array} \tag{6.14}$$

Toutes les nouvelles transitions ainsi créées forment T_Δ, désigné simplement T dans la suite de la section.

③ Modification des conditions symboliques

Comme on l'a dit dans la gestion des incertitudes, la troisième étape est la mise à VRAI de toutes les conditions symboliques :

$$\forall t \in T_\Delta, \quad \omega_t^S \in \Omega_t \quad \Rightarrow \quad \omega_t^S \leftarrow \top. \tag{6.15}$$

Ainsi, toutes les configurations du système du diagnostiqueur satisfont les conditions symboliques. Les annotations des arcs restent inchangées :

$$\mathcal{A}_\Delta \quad = \quad \mathcal{A}_\Phi. \tag{6.16}$$

④ Suppression des conditions de dégradation

L'étape ④ consiste à supprimer les conditions de dégradation afin de déconnecter l'évolution du marquage du niveau de dégradation de l'état de dégradation :

$$\forall t \in T_\Delta, \quad \omega_t^D \in \Omega_t \quad \Rightarrow \quad \Omega_t \leftarrow \Omega_t \backslash \{\omega_t^D\}. \tag{6.17}$$

Cette étape permet de gérer les performances de calcul et de rester concentré sur les observations pendant le processus de diagnostic.

⑤ Fusion des transitions

L'étape ⑤ vise également à améliorer les performances de calcul en fusionnant les transitions ayant le même ensemble de places en entrée et la même place numérique en sortie. Cela réduit la taille de l'espace des états possibles. Dans le niveau de comportement, cela permet de créer, pendant l'étape de prédiction, des hypothèses partageant le même cluster de particules. En d'autres termes, il devient possible de surveiller plusieurs hypothèses suivant la même dynamique continue avec un unique cluster de particules au lieu d'avoir autant de clusters que d'hypothèses.

Dans le niveau de dégradation, cette étape élimine les transitions concurrentes qui ont la même place de dégradation en entrée et la même place de dégradation en sortie.

Deux transitions sont fusionnables si elles représentent le même changement de dynamique continue (mêmes places numériques en entrée et en sortie, et même condition numérique) et qu'elles ont la même place symbolique en entrée.

Définition 25 (Transitions fusionnables) *Deux transitions $(t', t'') \in T^2$ sont fusionnables si et seulement si :*

$$({}^\circ t' = {}^\circ t'') \quad \wedge \quad (t'^\circ \cap P^N = t''^\circ \cap P^N) \quad \wedge \quad (t'^\circ \cap P^D = t''^\circ \cap P^D) \quad \wedge \quad (\Omega_{t'} = \Omega_{t''}), \tag{6.18}$$

où ${}^\circ t$ (resp. t°) désigne l'ensemble des places en entrée (resp. en sortie) d'une transition $t \in T$.

L'étape de fusion consiste à fusionner toutes les transitions fusionnables tant qu'il y a au moins deux transitions fusionnables en utilisant la définition suivante.

Définition 26 (Fusion de deux transitions) *La fusion de deux transitions fusionnables $(t', t'') \in (T)^2$ est définie par deux étapes :*

(1) *Création d'une nouvelle transition t pour laquelle :*

$$\begin{array}{rcl} {}^\circ t & \leftarrow & {}^\circ t', \\ t^\circ & \leftarrow & t'^\circ \cup t''^\circ, \\ \Omega_t & \leftarrow & \Omega_{t'}. \end{array} \tag{6.19}$$

(2) *Introduction de t dans T et suppression de t' et t'' :*

$$T \quad \leftarrow \quad (T \backslash \{t', t''\}) \quad \cup \quad \{t\}. \tag{6.20}$$

La figure 6.5 illustre l'étape de fusion sur un HPPN simple ayant deux transitions. Les transitions t' et t'' sont fusionnables car elles ont le même ensemble de places en entrée $\{p_1^S, p_1^N\}$, la même place numérique en sortie p_2^N, et les mêmes conditions numérique et symbolique.

⑥ Suppression des transitions bouclant sur une même place de dégradation

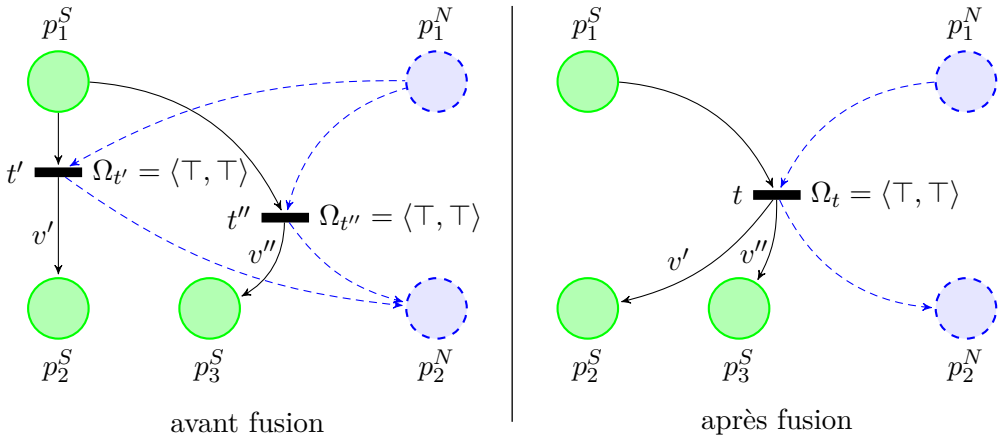


FIGURE 6.5 – Etape de fusion des transitions dans le niveau de comportement du diagnostiqueur sur un HPPN simple.

L'étape ⑥ consiste à supprimer les transitions créant une boucle élémentaire dans le niveau de dégradation (réseau de Petri pur).

$$T_\Delta \leftarrow T_\Delta \setminus \{t | {}^\circ t \cap P^D = t^\circ \cap P^D\}. \tag{6.21}$$

Le but est d'améliorer les performances de calcul en évitant le déplacement des jetons de dégradation à travers une transition qui boucle sur la même place de dégradation. Cette étape n'a aucun impact sur la qualité du suivi de la dégradation.

Processus de diagnostic

⇨ Initialisation du diagnostiqueur, réglage des performances

Le marquage initial $\mathbb{M}_0 = \{\mathbb{M}_0^S, \mathbb{M}_0^N, \mathbb{M}_0^D\}$ du diagnostiqueur HPPN représente le mode initial du système. Il est composé d'une configuration de valeur b_0, n_0^N particules de valeur x_0 et de n_0^N jetons de dégradation de valeur d_0, où n_0^N est le nombre initial de particules.

⇨ Evolution du marquage du diagnostiqueur

A partir du marquage initial et des commandes initiales, le marquage du diagnostiqueur $\hat{\mathbb{M}}_k$ évolue au temps k avec les observations $O_k = O_k^S \cup O_k^N$, où O^S et O^N représentent respectivement les observations correspondant à la partie symbolique et à la partie numérique.

L'évolution du marquage du diagnostiqueur repose sur deux étapes, la prédiction et la correction, qui combinent le pseudo-tirage de transition, des filtres particulaires et un algorithme appelé l'algorithme de mise à l'échelle stochastique (Stochastic Scaling Algorithm ou SSA). Ces étapes sont illustrées sur la figure 6.6.

Dans le filtrage particulaire, le nombre de particules définit la précision du filtre. Le but du SSA, proposé dans la thèse de Quentin Gaudel, est d'éviter l'explosion combinatoire et de limiter le nombre de jetons à chaque étape de l'algorithme. Il adapte dynamiquement la précision des hypothèses. Cet algorithme n'est pas décrit dans ce chapitre, mais le lecteur peut se référer à [Douc & Cappé 2005], [Li *et al.* 2015], [Doucet & Johansen 2009] ou bien au manuscrit de thèse pour obtenir plus d'informations sur les méthodes de rééchantillonnage pour le filtrage de particules.

L'étape de prédiction du processus de diagnostic en ligne vise à déterminer tous les états suivants possibles du diagnostiqueur $\hat{\mathbb{M}}_{k+1|k}$. Il est basé sur le déclenchement des

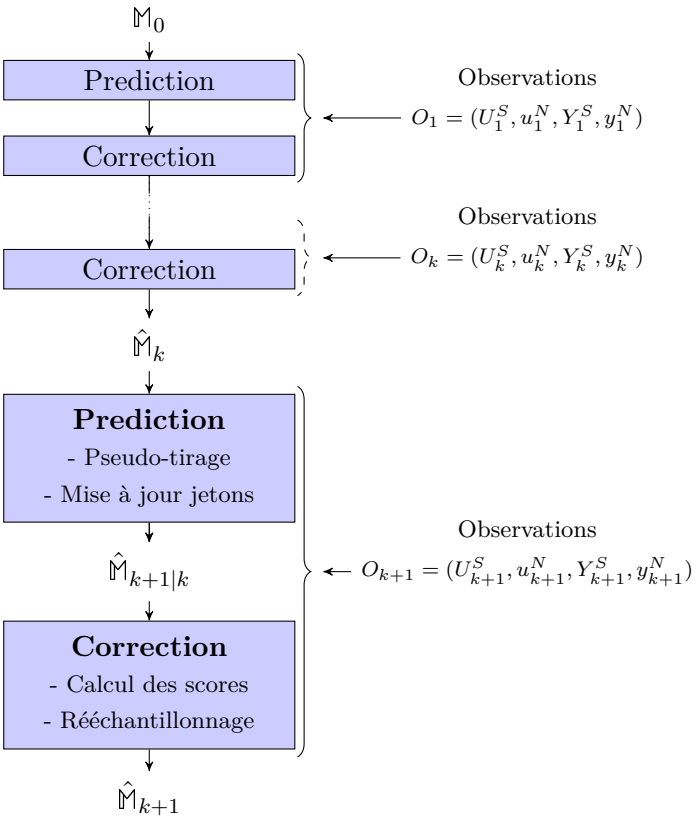


FIGURE 6.6 – Processus du diagnostiqueur HPPN.

transitions franchissables et sur la mise à jour des valeurs de jetons.

Toutes les transitions franchissables sont déclenchées selon les règles décrites dans la section 6.4.1, page 48, aux différences près que l'ensemble $\mathcal{S}_{k-1}^t$ des jetons acceptés par une transition $t \in T$ est déterminé de manière à respecter la condition logique $\omega_t^S \wedge \omega_t^N$ et que les jetons sont dupliqués et non déplacés.

On part du principe qu'un seul événement peut survenir au temps k.

L'ensemble des événements b_k d'une configuration δ_k déplacée à travers un arc $a \in A$ pendant le tirage des transitions est mis à jour : l'annotation de l'arc $\mathcal{A}(a)$ lui est ajoutée. La valeur x d'une particule π est mise à jour en fonction de la dynamique continue associée à la place numérique $p^N \in P^N$ à laquelle π appartient après le tirage de la transition. Le bruit est ajouté pendant la mise à jour de la valeur des particules pour prendre en compte l'incertitude sur la dynamique continue du modèle. La valeur `d` d'un jeton de dégradation d est mise à jour en fonction de la dynamique de dégradation associée à la place de dégradation $p^D \in P^D$ à laquelle d appartient après le tirage de transition.

L'étape de correction du processus de diagnostic en ligne met à jour le marquage prédit $\hat{\mathbb{M}}_{k+1|k}$ grâce aux nouvelles observations O_{k+1}. On obtient alors $\hat{\mathbb{M}}_{k+1|k+1}$. Cette étape est basée sur le calcul d'un score pour toutes les hypothèses contenues dans le marquage et sur le rééchantillonnage des jetons en fonction des scores des hypothèses qu'ils représentent. Les scores des hypothèses sont calculés avec Pr^S et Pr^N, les distributions de probabilité sur les états symbolique et continu, respectivement.

Pr^S donne le poids d'une configuration, calculé comme l'inverse de l'exponentielle de la distance entre l'ensemble des événements de la configuration et $O_{k+1}^- = \{O_\kappa | \kappa \leq k+1\}$, l'ensemble des observations symboliques jusqu'à $k+1$. Autrement dit, plus l'ensemble

d'événements de la configuration est semblable aux événements survenus sur le système, plus le poids de la configuration est proche de 1.

Pr^N donne les poids normalisés des particules, calculés en fonction de la distance entre les valeurs des particules et les observations numériques O^N_{k+1}.

Le score d'une hypothèse est calculé comme une fonction pondérée de la somme des poids de ses particules et du poids de sa configuration :

$$Score(\delta^i_k, \{\pi^j_k\}, \{d^l_k\}) = \alpha \times Pr^S(\delta^i_k) + (1-\alpha) \times \sum_{j=1}^{n^N_k} Pr^N(\pi^j_k), \tag{6.22}$$

où le coefficient $\alpha \in [0,1]$ représente la confiance globale que l'on a sur la partie symbolique par rapport à la partie numérique et $n^N_k = |\{\pi^j_k\}|$ est le nombre de particules considérées pour l'hypothèse.

Le score d'une hypothèse est toujours compris entre 0 et 1.

Comme on l'a déjà dit, l'algorithme stochastique de changement d'échelle (SSA) adapte dynamiquement la précision des hypothèses de diagnostic, c'est-à-dire le nombre de particules n^N_{k+1} à associer à chaque hypothèse. Il utilise pour cela trois paramètres d'échelle, $\rho^{min}_\Delta, \rho^{max}_\Delta$ et ρ^{tot}_Δ, et les scores des hypothèses.

Chaque ensemble de particules est donc rééchantillonné en un nouvel ensemble de n^N_{k+1} particules, comme dans un filtrage particulaire classique.

Les paramètres ρ^{min}_Δ et ρ^{max}_Δ sont respectivement le nombre minimum et le nombre maximum de particules pour surveiller une hypothèse. La même contrainte existe pour les jetons de dégradation. Cela signifie que tout n^N_{k+1} est choisi pour satisfaire le prédicat $\rho^{min}_\Delta \leq n^N_{k+1} \leq \rho^{max}_\Delta$. Le paramètre ρ^{tot}_Δ est le nombre total de particules (ou de jetons de dégradation) disponibles pour surveiller toutes les hypothèses. Cela signifie que le nombre total de particules après le rééchantillonnage est toujours inférieur ou égal à ρ^{tot}_Δ. Lors du rééchantillonnage, les jetons de dégradation liés aux particules dupliquées sont dupliqués tandis que ceux liés aux particules supprimées sont supprimés. Enfin, les configurations qui ne sont plus liées à des jetons de dégradation sont supprimées. Le mécanisme de correction met en évidence que les jetons de dégradation, en plus d'estimer l'état de dégradation, empêchent la distribution de particules d'une hypothèse d'être perturbée par les distributions de particules des autres hypothèses. Dans le filtrage particulaire, le nombre de particules définit la précision du filtre mais est également un facteur de performance de calcul. Les paramètres d'échelle du SSA font ainsi le compromis entre le nombre d'hypothèses à surveiller et la précision accordée à chacune d'elles, par rapport à la puissance de calcul mise en jeu (ρ^{tot}_Δ peut être paramétré pour satisfaire les contraintes de performance).

Diagnostic

Le diagnostic Δ_k est déduit du marquage au temps k du diagnostiqueur HPPN :

$$\Delta_k = \hat{\mathbb{M}}_k = \{\hat{\mathbb{M}}^S_k, \hat{\mathbb{M}}^N_k, \hat{\mathbb{M}}^D_k\}. \tag{6.23}$$

Il représente toutes les hypothèses de diagnostic comme une distribution de croyance sur les modes de santé et donne une information sur la manière dont chaque mode a été atteint. En d'autres termes, le marquage $\hat{\mathbb{M}}_k$ indique la croyance sur l'état continu, les occurrences de fautes et l'état de dégradation. Les résultats du diagnostiqueur incluent les résultats d'un diagnostiqueur classique en termes d'occurrences de fautes. Dans un diagnostiqueur classique, cependant, chaque hypothèse de diagnostic a le même degré de croyance. Un diagnostiqueur HPPN gère plus d'incertitudes et évalue l'ambiguïté en fonction des places et des valeurs des jetons.

Cette approche a été testée sur le cas d'étude d'un système de réservoirs d'eau ainsi que sur un rover K11 de la NASA. Les résultats ont été publiés dans [Gaudel *et al.* 2015a], [Gaudel *et al.* 2015b], [Gaudel *et al.* 2016].

6.5.3 Pronostic basé sur les HPPN

Cette section présente la méthodologie de pronostic de système basée sur les HPPN.

Gestion des incertitudes

Un point commun de toutes les méthodes de pronostic est que le processus, par nature, doit faire face à un certain nombre d'incertitudes. On peut les classer suivant trois sources principales : la connaissance de l'état courant du système, les bruits liés à l'évolution de l'état du système et la connaissance des futures entrées sur le système. Considérer toutes ces sources d'incertitude permet de prédire un maximum de trajectoires possibles pour le système. Les paragraphes suivants exposent nos stratégies par rapport à ces 3 sources d'incertitudes.

⇨ Incertitudes liées à la connaissance de l'état courant

Beaucoup d'études considèrent l'estimation de l'état continu courant comme le point de départ d'un processus de pronostic [Daigle *et al.* 2014a, Jouin *et al.* 2016] et éventuellement du mode courant dans le cadre des systèmes hybrides [Daigle *et al.* 2015b]. Dans notre méthodologie, le processus de pronostic se base sur le diagnostic HPPN, qui contient une distribution de croyances sur les états discrets, continus et de dégradation, mais aussi sur l'ensemble des événements survenus sur le système, nécessaire pour simuler l'évolution de l'état de dégradation : le résultat de diagnostic HPPN est l'état initial du processus de pronostic.

⇨ Incertitudes liées aux bruits sur l'état du système

Intuitivement, ignorer le bruit sur l'évolution de l'état du système empêche l'exploration de toutes les trajectoires possibles. Cependant, la question est discutable, tant pour la dynamique discrète du modèle, que pour les dynamiques continue et de dégradation.

Concernant la partie discrète du modèle, considérer à chaque instant que n'importe quel événement peut survenir provoque une explosion combinatoire. Lors du processus de pronostic, par manque d'observations, cette explosion combinatoire ne peut être corrigée, contrairement à ce qui est fait durant le processus de diagnostic. Pour éviter cette explosion combinatoire, l'occurrence d'un événement à n'importe quel instant n'est pas envisagée durant le processus du pronostiqueur. En contraste avec le processus du diagnostiqueur, les transitions ne sont donc pas pseudo-tirées et seuls les événements d'entrées (commandes) sont conservés.

Concernant la partie continue du modèle, la gestion des bruits liés à l'évolution de l'état du système est inspirée des approches de diagnostic de systèmes continus avec des estimateurs (filtres de Kalman ou filtres particulaires, etc.). Cependant, ce type d'approche peut retourner des résultats de pronostic non concluants si les bruits liés aux dynamiques des paramètres inconnus sont trop importants. La méthode propose donc une étape optionnelle à la discrétion de l'opérateur durant la génération du pronostiqueur qui permet de supprimer les bruits des dynamiques continue et de dégradation.

Les entrées discrètes sont simulées et les conditions sur l'état de dégradation sont utilisées pour simuler les futurs changements de mode.

⇨ Incertitudes liées à la connaissance des futures entrées

On définit l'ensemble, planifié au temps k, des entrées sur le système prévues durant l'horizon de prédiction τ_p, $U_k^+ = \{U_\kappa | \kappa \in \{k+1, ..., k+\tau_p\}\}$, avec $U_\kappa = (U_\kappa^S, u_\kappa^N)$ la paire de l'ensemble des entrées discrètes U_κ^S et du vecteur d'entrée continu $u_\kappa^N \in \mathbb{R}^{n_u}$, à l'instant futur κ.

Dans un cas réel, déterminer l'ensemble des futures entrées U_k^+ est une grande source d'incertitudes. Les commandes de bas niveau sont difficiles à prédire précisément, même si le plan de mission de haut niveau est connu. Considérer du bruit dans le plan de mission et dans les futures commandes est une solution à ce problème [Daigle *et al.* 2015c].

Il est également possible que les futures entrées soient entièrement inconnues. Ceci est un problème uniquement si le temps n'influence ni l'évolution de l'état continu, ni celle de l'état de dégradation du système. Une solution consiste à propager les entrées courantes en première approximation. Une autre solution consiste à générer un ensemble d'entrées à partir des entrées précédentes. Ces deux solutions ont été investiguées dans les travaux de thèse de Quentin Gaudel.

Génération du pronostiqueur

La méthode de pronostic se base sur la génération d'un pronostiqueur HPPN à partir d'un modèle HPPN et la définition d'un processus de pronostic. Les transitions et les conditions du modèle HPPN sont transformées pour pouvoir simuler l'évolution de l'état du système dans le futur et ainsi déterminer le RUL/EOL du système.

Définition 27 (Pronostiqueur HPPN) *Soit un modèle HPPN défini par le 11-uplet* $HPPN_\Phi = \langle P_\Phi, T_\Phi, A_\Phi, \mathcal{A}_\Phi, E_\Phi, X_\Phi, D_\Phi, \mathcal{C}_\Phi, \mathcal{H}_\Phi, \Omega_\Phi, \mathbb{M}_{0\Phi} \rangle$ *comme présenté en section 6.4.1. Le pronostiqueur* $HPPN_\Pi$ *de* $HPPN_\Phi$ *est un 11-uplet généré à partir de* $HPPN_\Phi$ *:*

$$HPPN_\Pi \quad = \quad \langle P_\Pi, T_\Pi, A_\Pi, \mathcal{A}_\Pi, E_\Pi, X_\Pi, D_\Pi, \mathcal{C}_\Pi, \mathcal{H}_\Pi, \Omega_\Pi, \mathbb{M}_{0\Pi} \rangle. \tag{6.24}$$

Les éléments de $HPPN_\Pi$ sont déterminés en 4 étapes.

① Copie du modèle

Le rôle du pronostiqueur est de simuler l'évolution de l'état du système dans le futur pour déterminer le EOL ou le RUL du système. L'étape ① consiste à copier les places, les labels d'événements et les espaces d'états du modèle $HPPN_\Phi$:

$$P_\Pi = P_\Phi, E_\Pi = E_\Phi, X_\Pi = X_\Phi, D_\Pi = D_\Phi. \tag{6.25}$$

Le marquage initial du pronostiqueur $\mathbb{M}_{0\Pi}$ est déterminé à partir du processus de diagnostic (section 7, page 63).

② Modification des conditions

Comme on l'a mentionné dans la gestion des incertitudes, cette étape consiste à ne conserver pour les conditions symboliques que les événements d'entrée (commandes discrètes). Les autres événements sont ignorés car on ne peut pas les prédire ; les conditions associées sont donc mises à VRAI :

$$\forall t \in T_\Pi, \quad (\omega_t^S = occ(b, v) \wedge v \notin U^S) \quad \Rightarrow \quad \omega_t^S \leftarrow \top, \tag{6.26}$$

où U^S est l'ensemble des labels de toutes les entrées discrètes.

Dans le cas où les deux conditions ω_t^S et ω_t^N sont à VRAI, on souhaite que le tirage de la transition t dépende quand même de la condition de dégradation ω_t^D. On effectue donc la modification suivante :

$$\forall t \in T_\Pi, \quad (\omega_t^S = \top \wedge \omega_t^N = \top) \quad \Rightarrow \quad (\omega_t^S \leftarrow \bot, \omega_t^N \leftarrow \bot). \tag{6.27}$$

À ce stade de la génération, les annotations des arcs restent inchangées pour enregistrer l'ensemble des événements simulés dans les ensembles d'événements des configurations :

$$\mathcal{A}_\Pi \quad = \quad \mathcal{A}_\Phi. \tag{6.28}$$

③ Suppression des transitions non tirables

L'étape ③ consiste à supprimer toutes les transitions inutiles pour la simulation dans le futur, c'est-à-dire toutes les transitions t pour lesquelles toutes les conditions de Ω_t sont $\bot$:

$$T_\Pi \quad \leftarrow \quad (T \quad \setminus \quad \{t \in T_\Pi | \Omega_t = \langle \bot, \bot, \bot \rangle\}). \tag{6.29}$$

Les ensembles des arcs A_Π, des conditions Ω_Π, et des annotations $\mathcal{A}_\Pi$ sont réduits en conséquence.

④ Suppression des bruits de l'évolution de l'état (optionnelle)

Comme on l'a précisé auparavant, cette étape est à la discrétion de l'opérateur, qui choisit ou pas de conserver les bruits liés aux dynamiques continues et de dégradation.

Processus de pronostic

Le processus de pronostic peut-être exécuté à n'importe quel instant k et retourne un pronostic Π_k. Ses entrées sont le pronostiqueur $HPPN_\Pi$, le diagnostic courant Δ_k et l'ensemble U_k^+ des entrées futures du système au temps k.

Les étapes du processus de pronostic sont données dans l'Algorithme 1.

Algorithme 1 : Pronostiquer

Entrées : $HPPN_\Pi, \Delta_k, U_k^+$
Sorties : $\hat{\mathbb{M}}_{k_{\text{EOP}}|k}$

```
Initialiser(HPPN_Π, Δ_k);
pour tous les U_κ ∈ U_k^+ faire
    si TousDéfaillants(HPPN_Π) = ⊥ alors
        M̂_{κ|k} ← MettreÀJour(HPPN_Π, κ, U_κ);
    sinon
        sortie de la boucle pour
k_EOP ← κ;
```

⇨ Initialisation du pronostiqueur, réglage des performances

Pour garder l'incertitude liée au diagnostic, le marquage initial du pronostiqueur $\mathbb{M}_{0\Pi} = \mathbb{M}_{k\Pi}$ est déterminé à partir du diagnostic Δ_k (donné par l'équation 6.23) : la distribution des configurations, particules et jetons de dégradation représentant toutes les hypothèses sur le système au temps k.

Les jetons de Δ_k sont dupliqués de telle sorte que chaque hypothèse de précision n est représentée dans le pronostiqueur par m hypothèses de précision 1, où $m \in \mathbb{N}_+$ est déterminé avec l'algorithme stochastique de changement d'échelle (SSA), dont le principe a déjà été évoqué dans la section 6.5.2, page 58.

Lorsqu'une hypothèse est complètement reproduite, on a $m = n$: chaque particule π_k ou jeton de dégradation d_k de Δ_k est dupliqué dans le pronostiqueur. Cependant, le plus souvent, et dans le but de s'adapter aux exigences sur les temps de calcul, les hypothèses dans Δ_k ne sont que partiellement reproduites : $m \leq n$. L'algorithme SSA détermine les précisions m à associer à chaque hypothèse, en se basant sur leur degré de croyance, et

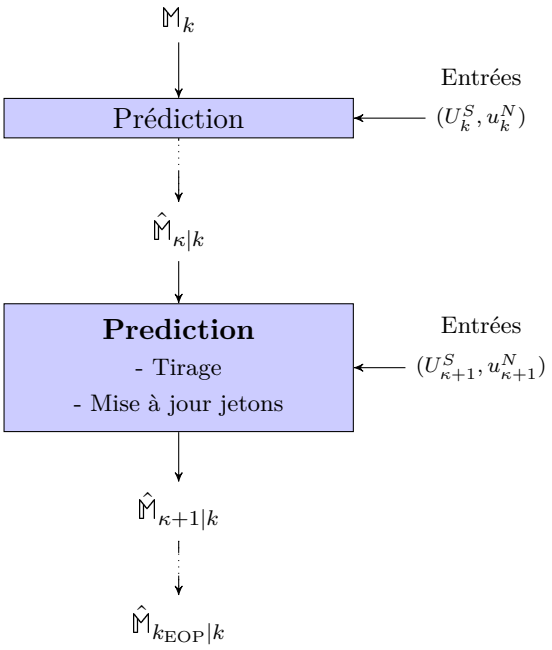


FIGURE 6.7 – Processus du pronostiqueur HPPN.

trois paramètres d'échelle ρ_{Π}^{min}, ρ_{Π}^{max} et ρ_{Π}^{tot}. Ensuite, pour chaque hypothèse de Δ_k, les m particules et m jetons de dégradation à dupliquer dans le pronostiqueur sont sélectionnés aléatoirement.

Les paramètres ρ_{Π}^{min} et ρ_{Π}^{max} représentent respectivement le nombre minimum et le nombre maximum de jetons de chaque type pour représenter une hypothèse dans le pronostiqueur. Le paramètre ρ_{Π}^{tot} est le nombre total de jetons (de chaque type) disponibles pour représenter toutes les hypothèses.

Les paramètres ρ_{Π}^{min} et ρ_{Π}^{max} sont respectivement les précisions minimale et suffisante accordées au pronostiqueur pour simuler l'évolution d'une hypothèse dans le futur. Les paramètres d'échelle offrent donc un compromis entre le nombre d'hypothèses à projeter dans le futur et la précision accordée à chacune, par rapport au nombre maximum de jetons à utiliser pour la simulation. Comme pour le processus de diagnostic, ils peuvent être choisis pour répondre à des contraintes de performance.

⇨ Evolution du marquage du pronostiqueur

Une fois le pronostiqueur $HPPN_{\Pi}$ initialisé, son marquage évolue en accord avec les entrées futures $U_k^+ = \{U_\kappa | \kappa \in \{k+1, ..., k+\tau_p\}\}$, où τ_p est l'horizon de prédiction.

L'évolution du marquage du pronostiqueur repose des étapes de prédictions successives, illustrées sur la figure 6.7.

Nous avons défini formellement une transition tirable et le tirage d'une transition dans ce cadre. Toutes les transitions tirables sont tirées simultanément selon des règles similaires à celles présentées dans la section 6.4.1, à la différence près que l'ensemble $\mathcal{S}_{\kappa+1}^t$ des jetons acceptés par une transition $t \in T$ au temps $\kappa + 1$ est déterminé de manière à respecter la condition logique $(\omega_t^S \wedge \omega_t^N) \vee \omega_t^D$.

Pendant le tirage des transitions, les valeurs des jetons sont mises à jour.

L'ensemble des événements b_κ d'une configuration déplacée à travers un arc a pendant le tirage d'une transition $t \in T$ au temps $\kappa + 1$, est mis à jour avec le label d'événement

$v \in E$ annotant a ; l'événement (v, κ) lui est ajouté. Les valeurs des jetons de dégradation puis celles des particules sont mises à jour en fonction des places auxquelles appartiennent les jetons et des entrées continues à l'instant $\kappa + 1$. Le vecteur d'état de dégradation d_κ d'un jeton de dégradation est mis à jour avec l'équation d'évolution associée à la place de dégradation à laquelle le jeton de dégradation appartient, et les valeurs de la configuration et de la particule qu'il lie. Le vecteur d'état numérique x_κ d'une particule est mis à jour avec l'équation d'évolution associée à la place numérique à laquelle la particule appartient.

Le caractère borné du pronostiqueur assuré par l'algorithme SSA a été démontré dans le manuscrit de thèse.

Pronostic

Le pronostic Π_k est le marquage du pronostiqueur HPPN à la fin du processus de pronostic :

$$\Pi_k \triangleq \hat{\mathbb{M}}_{k_{\text{EOP}}|k}. \tag{6.30}$$

Propriété 2 (Fin de prédiction) *Quel que soit l'horizon de prédiction τ_p choisi, on a toujours :*

$$k_{\text{EOP}} \leq k + \tau_p. \tag{6.31}$$

Le pronostic Π_k est une distribution de croyances sur les futurs modes du système jusqu'à l'instant k_{EOP}.

A partir de la connaissance de Π_k, on peut notamment prédire les événements menant aux futurs modes de défaillance, en particulier les fautes et leurs dates d'occurrence.

Le pronostic Π_k contient toutes les informations nécessaires pour l'obtention d'une distribution de croyances sur le EOL et le RUL, puisqu'il contient les dates d'entrée de chaque hypothèse dans un mode de défaillance.

Soit une hypothèse $\{\delta, \pi, h\}$ contenue dans le pronostic Π_k et soient p^S la place symbolique de la configuration δ, p^N la place numérique de la particule π et p^D la place de dégradation du jeton de dégradation d. Si les places p^S, p^N et p^D représentent un mode de défaillance, alors le EOL de l'hypothèse est la date d'occurrence du dernier événement survenu pour cette hypothèse, i.e. la date la plus grande de tous les événements contenus dans l'ensemble d'événement b de la configuration δ :

$$EOL(\{\delta, \pi, h\}) \triangleq max\{\kappa | (v, \kappa) \in b\}. \tag{6.32}$$

Avec le EOL d'une hypothèse, on peut retrouver son RUL en fonction du temps k à partir duquel a été lancé le calcul du pronostic :

$$RUL(\{\delta, \pi, h\}) \triangleq EOL(\{\delta, \pi, h\}) - k. \tag{6.33}$$

Ainsi, les RUL/EOL des hypothèses couplés aux scores de chacune d'entre elles calculés avec l'équation 6.22 forment la distribution de croyances sur le RUL/EOL du système.

Tous les concepts abordés dans la méthodologie de pronostic ont été illustrés sur l'exemple du robot mobile puis appliqués à un système de réservoirs d'eau. Le pronostic a également été mis en œuvre sur le rover K11 de la NASA.

6.6 Implémentations et résultats sur un cas réel

Comme on l'a dit précédemment, la méthodologie de surveillance de santé basée sur les HPPN a été appliquée au K11, un prototype de rover planétaire. L'objectif de cette section

n'est pas de présenter les résultats de la méthode de diagnostic/pronostic basée HPPN, publiés dans la thèse de Quentin Gaudel et dans [Gaudel *et al.* 2016], mais de tirer les enseignements de ces travaux pour en envisager des pistes de recherche pertinentes pour le futur.

Le K11 est un rover alimenté par une batterie possédant de nombreux capteurs ainsi que quatre roues motorisées et contrôlables. Il a été conçu pour tester les architectures économes en énergie, notamment en Antarctique [Lachat *et al.* 2006]. Sa conception a ensuite été repensée par le NASA Ames Research Center [Balaban *et al.* 2013] à des fins de diagnostic et de pronostic [Daigle *et al.* 2014b], et aussi de replanification à partir du pronostic [Sweet *et al.* 2014]. Il s'agit maintenant d'un banc d'essai sur lequel il est possible de provoquer (ou simuler) des fautes et des défaillances.

Dans les travaux que nous avons menés, le K11 est étudié comme un système en opération réalisant des missions d'exploration et exposé à des défaillances techniques, comme des pannes de capteurs, l'apparition de charges parasites sur la batterie ou des surchauffes des moteurs.

6.6.1 Modélisation, diagnostiqueur, pronostiqueur

En considérant les quatre moteurs et des fautes multiples, 192 modes et 240 changements de mode ont été identifiés. Le modèle HPPN du rover que nous avons développé possède 241 places (192 places symboliques, 48 places numériques, 1 place de dégradation) et 240 transitions.

Le diagnostiqueur HPPN possède le même nombre de places et de transitions que le modèle. L'étape de fusion de la génération du diagnostiqueur ne réduit pas le nombre de transitions car toutes les dynamiques continues sont différentes. C'est un cas très particulier et cela n'est pas à généraliser pour tous les autres systèmes. Puisqu'il n'y a qu'une seule place de dégradation, il n'y a pas de transition dans le niveau de dégradation.

Le pronostiqueur HPPN a le même nombre de places que le modèle, mais seulement 160 transitions (gain de représentation de 33 %). En effet, lors de l'étape de transformation des conditions de la génération du pronostiqueur, seules les transitions étant associées à une contrainte sur l'état continu sont gardées car tous les évènements sont des fautes (et non des commandes discrètes) et qu'aucune contrainte sur l'état de dégradation n'est associée aux transitions.

Les DESs sous-jacent de la description du K11, son modèle HPPN, son diagnostiqueur HPPN et son pronostiqueur HPPN, sont disponibles au lien suivant : `https://homepages.laas.fr/echanthe/K11`.

6.6.2 Implémentation et résultats

La méthodologie de surveillance basée sur les HPPN (modélisation, diagnostiqueur, pronostiqueur, etc.) est implémentée en Python 3.4. Les tests ont été effectués sur un processeur Intel® 4 Core™ i5-4590 à 3.30 GHz avec 16 GB de RAM, tournant sous GNU/Linux (Linux 3.13, x86_64). Dans le but de réduire le temps de calcul, les mises à jours des valeurs des jetons sont réparties sur les 4 cœurs physiques. Le reste de l'implémentation utilise un seul cœur.

Trois scénarios déjà étudiés dans la littérature [Sweet *et al.* 2014] ont été considérés.

Dans un premier temps, on s'intéresse aux résultats de la méthodologie avec des jeux de paramètres d'échelle fixes pour les algorithmes SSA du diagnostiqueur et du pronostiqueur. Dans un second temps, la performance de la méthodologie est étudiée en comparant les résultats, les temps de calcul et l'utilisation de la mémoire pour différents jeux de paramètres

d'échelle du SSA.

Les conclusions générales sont les suivantes :

⇨ **Détection des fautes**

Les fautes sont toujours détectées en une période d'échantillonnage car le diagnostiqueur HPPN considère toutes les possibilités durant l'étape de prédiction du process de diagnostic grâce au principe du pseudo-tirage et garde le marquage cohérent durant l'étape de prédiction.

⇨ **Pronostic**

Concernant les résultats de pronostic, les estimations du RUL ayant les degrés de croyance les plus élevés sont toutes supérieures à la durée réelle avant la fin de la mission. Un point important est que des hypothèses de moindre importance apparaissent, montrant des RUL plus faibles : ceci peut être intéressant dans le cadre de missions particulièrement critiques où une décision doit être prise dans le pire cas. Les pronostics sont également mis à jour grâce au diagnostic, ce qui les rend plus précis. Les résultats sont cohérents, mais souffrent parfois d'une mauvaise modélisation : c'est évidemment le point faible de toute approche à base de modèles. Une perspective intéressante est de travailler sur l'**apprentissage des modèles**. Ce point sera traité dans la troisième partie de ce manuscrit.

⇨ **Temps de calcul d'un diagnostic et d'un pronostic**

Les temps de calcul d'un diagnostic et d'un pronostic, ainsi que la mémoire RAM maximum utilisée pour différents jeux de paramètres d'échelle ont été étudiés. Ces métriques soulignent le fait que les temps de calcul avec le jeu de paramètres d'échelle initial restent acceptables mais ne respectent pas les contraintes de temps réel. Ces performances s'expliquent principalement par le fait que les méthodes de diagnostic et de pronostic se basent sur la simulation pas à pas de l'évolution des nombreux jetons des HPPN. Bien que les générations et les processus du diagnostiqueur et du pronostiqueur HPPN aient été pensés pour optimiser ces simulations, ce type de méthodes dites séquentielles restent par principe coûteuses en terme de calcul. Une comparaison avec les performances calculatoires de la méthode de surveillance sur le système des réservoirs d'eau avec les mêmes paramètres d'échelle permet de déduire que les performances dépendent également du modèle HPPN du système. La complexité du modèle HPPN est par ailleurs difficile à évaluer, car les nombres de places et de transitions, la structure du réseau, les complexités calculatoires des dynamiques continues et de dégradation, le nombres de variables d'états, sont des paramètres qui influent sur les performances générales de la méthode. Une perspective prometteuse du travail est donc de trouver un moyen de **développer des algorithmes de diagnostic et de pronostic à temps contraint pour répondre aux exigences temps réel de certaines applications.** L'idée peut être également d'intégrer ces algorithmes via une machine d'exécution assurant une interaction correcte entre les composants. Ces points seront abordés dans la troisième partie de ce manuscrit.

⇨ **Contrôle des performances**

S'il est difficile de déterminer la complexité d'un modèle HPPN et d'évaluer son impact sur les performances calculatoires de la méthode de surveillance, on peut contrôler ces performances à l'aide des paramètres d'échelle. Le compromis, instauré par l'algorithme SSA, entre la précision des résultats et les performances calculatoires de la méthode est proposé comme une solution au problème de temps de calcul. On montre par les expériences que réduire les paramètres d'échelle diminue les temps de calcul et la mémoire RAM maximum utilisée. Concernant le diagnostic, la qualité de la surveillance est évidemment impactée car le diagnostiqueur n'a à sa disposition qu'entre 5 et 10 hypothèses. Pour le pronostic, les résultats incluent beaucoup moins d'incertitudes mais la seule hypothèse

du pronostiqueur permet d'obtenir un RUL du système exploitable par des techniques de gestion de santé comme la replanification de la mission ou la planification d'actions de maintenance.

6.6.3 Intégration diagnostic/pronostic

Cette section présente une solution envisagée pour mettre en œuvre un enrichissement mutuel des deux modules de diagnostic et de pronostic.

En ce qui concerne l'influence du diagnostic courant du système sur le processus de pronostic, il peut être pertinent, par exemple, de ne lancer un calcul de pronostic que sur les hypothèses incluant un mode dégradé ou bien qui considèrent que le système a récemment changé de mode. Le pronostiqueur projetterait alors moins d'hypothèses dans le futur mais avec des précisions plus importantes. Ceci permettrait de mieux gérer le compromis temps de calcul/précision.

Concernant l'influence du pronostic courant du système sur le processus de diagnostic, il peut être intéressant que le diagnostiqueur accorde plus de précision aux hypothèses ayant les RUL les plus courts ou encore des RUL qui remettent en cause la réalisation de la mission. Le diagnostiqueur aurait ainsi une meilleure estimation de ces hypothèses et pourrait potentiellement les incriminer ou les discriminer plus rapidement.

Dans le cadre de la gestion de santé, un tel rapprochement des deux méthodes est propice à l'obtention de meilleurs résultats de surveillance de santé, et donc à de meilleures prises de décision concernant le système. Les fonctions d'influence dépendent cependant davantage des attentes de la méthode de surveillance que de la qualité des techniques employées pour réaliser cette surveillance. La définition de ces fonctions d'influence relève donc des objectifs relatifs à la gestion de santé du système étudié.

L'idée ici n'est pas de définir des fonctions d'influence mais de montrer qu'elles peuvent être intégrées de manière cohérente dans le cadre de la surveillance de santé basée sur les HPPN. Nous proposons de rajouter une étape dans chacun des processus durant laquelle la fonction d'influence transforme la distribution des scores juste avant l'utilisation de l'algorithme SSA. L'étape ajoutée au processus de diagnostic influence le rééchantillonnage des jetons dans le diagnostiqueur et donc la précision de chacune des hypothèses. Celle ajoutée au processus de pronostic influence la reproduction des hypothèses dans le pronostiqueur et donc leur précision lors de leur projection dans le futur.

On distingue par la suite la fonction d'influence du diagnostic courant du système sur le processus de pronostic ($\Delta \rightarrow \Pi$) de celle de l'influence du pronostic courant du système sur le processus de diagnostic ($\Pi \rightarrow \Delta$). Ces deux fonctions d'influence sont mises en évidence dans la figure 6.8, dans laquelle est présentée l'intégration avancée des méthodes de diagnostic et de pronostic basées sur les HPPN.

On peut voir dans la figure que le processus de diagnostic utilise les observations consécutives sur le système pour réaliser les étapes de prédiction et de correction et déterminer le diagnostic courant. En fonction des scores associés à chaque hypothèse et de la fonction d'influence $\Delta \rightarrow \Pi$, le pronostiqueur peut ainsi être initialisé de manière pertinente vis-à-vis des objectifs fixés dans le cadre de la gestion de santé du système. Le pronostiqueur utilise ensuite les entrées futures sur le système pour réaliser les étapes de prédiction et déterminer le pronostic courant. En fonction des scores des hypothèses calculés avec les nouvelles observations et de la fonction d'influence $\Pi \rightarrow \Delta$, le diagnostiqueur peut alors déterminer les nouvelles précisions des hypothèses de manière pertinente vis-à-vis des objectifs relatifs à la gestion de santé du système. Le nouveau diagnostic calculé va pouvoir influencer le calcul du prochain pronostic, et ainsi de suite.

Dans le cadre de la surveillance de santé basée sur les HPPN, cette intégration est

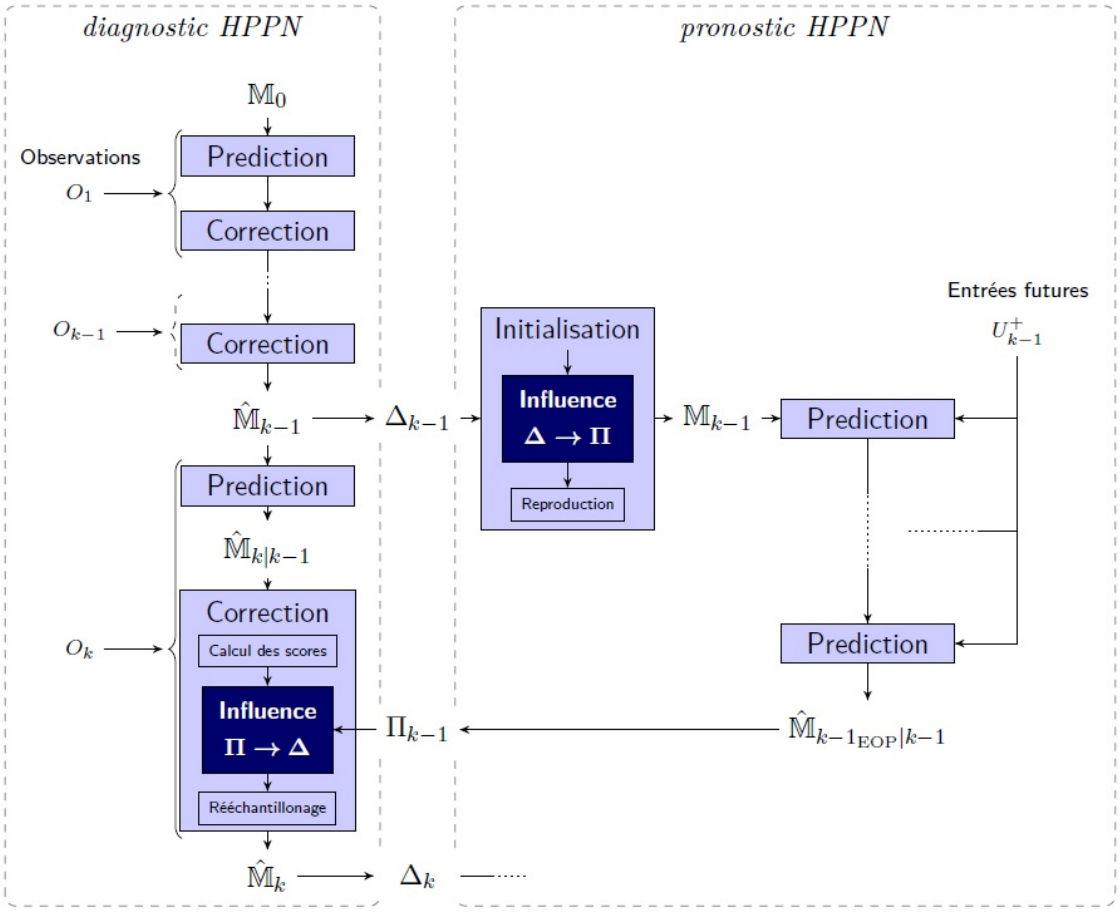


FIGURE 6.8 – Intégration avancée des méthodes de diagnostic et de pronostic basées sur les HPPN.

avantageuse pour plusieurs raisons. Premièrement, les résultats de l'un des deux modules ne sont pas utilisés pour interpréter directement les résultats de l'autre mais pour influencer la précision avec laquelle ils vont être déterminés. Deuxièmement, les fonctions d'influence $\Delta \rightarrow \Pi$ et $\Pi \rightarrow \Delta$ sont intégrées de manière cohérente vis-à-vis du déroulement des processus de diagnostic et pronostic et de leurs gestions des incertitudes. Enfin, la mise en places des fonctions d'influence $\Delta \rightarrow \Pi$ et $\Pi \rightarrow \Delta$ est générique vis-à-vis du type d'objectif à réaliser, ce qui pourrait mener à une réflexion quant à l'exploitation des résultats de surveillance basée sur les HPPN dans le cadre de la gestion de santé des systèmes.

6.7 Publications liées à cette partie

▶ E. Chanthery et P. Ribot. *An Integrated Framework for Diagnosis and Prognosis of Hybrid Systems.* In 3^{rd} Workshop on Hybrid Autonomous System, Italy, 2013

▶ S. Zabi, P. Ribot et E. Chanthery. *Health Monitoring and Prognosis of Hybrid Systems.* In Annual Conf. of the PHM Society, 2013

▶ Q. Gaudel, E. Chanthery, P. Ribot et E. Le Corronc. *Hybrid systems Diagnosis using modified particle Petri nets.* In 25^{th} Int. Workshop on Principles of Diagnosis, Austria, 2014

▶ Q. Gaudel, E. Chanthery et P. Ribot. *Health Monitoring of Hybrid Systems Using Hybrid Particle Petri Nets.* In Annual Conf. of the PHM Society, USA, 2014

▶ Q. Gaudel, E. Chanthery et P. Ribot. *Hybrid Particle Petri Nets for Systems Health Monitoring under Uncertainty.* Int. Journal of Prognostics and Health Management, vol. 6, no. 022, 2015

▶ Q. Gaudel, P. Ribot et E. Chanthery. *Vers une architecture de surveillance de santé d'un système hybride sous incertitudes.* In Modélisation des Systèmes Réactifs, France, 2015

▶ Q. Gaudel, P. Ribot, E. Chanthery et M. J. Daigle. Health Monitoring of a Planetary Rover Using Hybrid Particle Petri Nets, volume 9698 of *Lecture Notes in Computer Science*, chapitre Application and Theory of Petri Nets and Concurrency. PETRI NETS 2016. Lecture Notes in Computer Science, pages 196–215. Springer, Cham; Kordon F., Moldt D. (eds), 2016

▶ P. Ribot, E. Chanthery et Q. Gaudel. *HPPN-based Prognosis for Hybrid Systems.* In Annual Conference of the Prognostics and Health Management Society 2017, Proceedings of the Annual Conference of the Prognostics and Health Management Society 2017, St. Petersburg, United States, Octobre 2017

▶ Q. Gaudel, E. Chanthery, P. Ribot et M. J. Daigle. Fault diagnosis of hybrid dynamic and complex systems, chapitre Diagnosis of Hybrid Systems Using Hybrid Particle Petri Nets: Theory and Application on a Planetary Rover, pages 209–241. Springer, 2018

7 Diagnostic et Optimisation

Résumé

Ce chapitre présente mes travaux sur les liens entre le processus de diagnostic et l'optimisation. La section 7.2 présente les travaux relatifs au diagnostic actif, qui consiste à raffiner le diagnostic courant grâce aux capacités d'actions du système. La section 7.3 présente un résumé des travaux que j'ai effectués pour améliorer des algorithmes de diagnostic. Enfin, la section 7.4 se rapporte à des travaux dans le cadre de l'optimisation de la sélection d'actions ou de tests pour le diagnostic.

7.1 Introduction

L'autonomie décisionnelle d'un système est sa capacité à prendre des décisions et à agir de manière autonome dans un environnement dynamique. Pour cela, le système a besoin d'estimer en temps-réel son état interne. L'autonomie décisionnelle requiert par conséquent des fonctions de diagnostic en ligne (détection et isolation de fautes) et de planification/replanification en ligne [Chanthery *et al.* 2005b]. Ces fonctions doivent être intégrées dans une architecture embarquée.

Le diagnostic en ligne consiste à recevoir un flot d'observations à partir des capteurs et à en déduire une estimation de l'état du système. Le but est de suivre l'évolution temporelle de l'état du système. Cependant, cette tâche est souvent limitée par le nombre réduit d'observations fournies par les capteurs ou par des capacités limitées en termes de puissance de calcul et de mémoire.

Pour améliorer le diagnostic en ligne, une idée consiste à effectuer des actions en plus sur le système de manière à créer artificiellement des observations à des fins de diagnostic. C'est ce que l'on appelle le diagnostic actif. La section 7.2 présente ainsi les travaux relatifs au diagnostic actif, qui consiste à raffiner le diagnostic courant grâce aux capacités d'actions du système. La section 7.3 présente ensuite des travaux visant à améliorer l'embarquabilité des algorithmes de diagnostic par des techniques d'optimisation. Enfin, la section 7.4 se rapporte aux travaux sur l'optimisation de la sélection d'actions ou de tests pour le diagnostic.

7.2 Diagnostic actif

Le diagnostic hors ligne se concentre sur la localisation de la faute. Le but est de déterminer l'information à ajouter pour raffiner le diagnostic à moindre coût. Ce problème est à rapprocher du séquencement de tests. Décrit dans [Pattipati & Dontamsetty 1992] pour des tests binaires, puis étendu dans le cadre de la méthode AGENDA [Olive *et al.* 2003] pour des tests multi-valués, le problème de séquencement de tests est une procédure hors ligne dont le but est de trouver la meilleure séquence de tests permettant d'isoler la faute,

tout en minimisant le coût des tests.

L'objectif des travaux que nous avons entrepris avec Yannick Pencolé était de présenter une méthode de diagnostic en ligne pour un système autonome en prenant en compte son autonomie opérationnelle, c'est-à-dire la capacité de ce système à agir par lui-même. Ces travaux ont été effectués dans le cadre des Systèmes à Evénements Discrets. Une manière d'améliorer la performance du processus de diagnostic en ligne est d'utiliser l'autonomie opérationnelle du système pour effectuer des actions permettant de raffiner le diagnostic en cas d'ambiguïté : c'est ce que l'on appelle le *diagnostic actif*.

Le travail a été mené avec des collaborations avec Louise Travé-Massuyès, Medhi Bayoudh et Fabien Perrot de l'équipe DISCO dans le cadre du programme AGATA (Autonomy Generic Architecture : Test and Applications), commun au CNES et à l'ONERA, auquel le LAAS-CNRS a participé. Il a ensuite été poursuivi lors de stages de master en 2009 et 2010, en co-encadrement avec Yannick Pencolé. Un projet R&T du CNES avec Thalès a permis de continuer ces travaux en 2014.

Personnes impliquées dans cette thématique et affiliation lors de la collaboration : Yannick Pencolé (CR, DISCO), Louise Travé-massuyès (DR, DISCO), Medhi Bayoudh (Doctorant, DISCO), Fabien Perrot (Doctorant, DISCO), Quentin Gaudel (Doctorant, DISCO), Nicolas Bussac (Stagiaire M2, DISCO), Julien Salvy (Stagiaire M1, DISCO), Nicolas Garin (Stagiaire M2, DISCO), Régis De Ferluc (TAS), Brice Dellandre (TAS), Raymond Soumagne (CNES)

7.2.1 Contexte

Comme on l'a soulevé lors de l'introduction, le choix de l'ensemble des actions à effectuer en ligne peut être comparé au choix effectué lors d'un problème de séquencement de tests. Cependant, il reste des différences notables. Pour les systèmes dynamiques autonomes ayant pour objectif de réaliser une mission, un des défis de la fonction de diagnostic actif est de proposer une séquence d'actions (ou *plan*) permettant de raffiner le diagnostic sans changer radicalement le plan de la mission. En d'autres termes, le diagnostic actif doit être intégré dans une architecture embarquée dont le but premier est l'achèvement d'une mission. L'architecture embarquée inclut un planificateur de mission optimisant les actions pour réaliser la mission. L'ensemble des actions proposées par le diagnostic actif peut donc parfois être en conflit avec la réalisation de la mission. Les conflits entre le plan pour le diagnostic et le plan de la mission doivent être gérés. Par ailleurs, les méthodes de séquencement de tests utilisent des tests dont le résultat est binaire (oui/non) ou multi-valué. Dans le cas du diagnostic actif, le résultat d'une action est un nouvel état du diagnostiqueur.

La contribution majeure sur le diagnostic actif des systèmes à événements discrets est le travail de [Sampath *et al.* 1998]. Le diagnostic actif est formulé comme un problème de supervision [Ramadge & Wonham 1989] où le langage légal est un sous-langage régulier "approprié" du langage régulier du système. Une procédure itérative permet d'obtenir le sous-langage diagnosticable, observable, contrôlable minimal et d'obtenir le superviseur qui synthétise ce langage. La solution proposée est de construire un contrôleur de manière à ce qu'il satisfasse à la fois les objectifs de la commande et que le système contrôlé résultant soit diagnosticable. En d'autres termes, le domaine des actions est restreint de façon à ce que le système résultant soit diagnosticable. Cette approche semble restrictive pour les systèmes autonomes, qui ont besoin de toute leur capacité d'action pour réaliser leur mission. On préférera perdre de manière ponctuelle la propriété de diagnosticabilité plutôt que de ne pas pouvoir achever la mission. Dans nos travaux, l'idée est donc de combiner la supervision, les capacités de diagnostic et le contrôleur d'exécution, sans pour autant réduire les capacités d'actions du système.

Une approche pour réaliser un diagnostic actif dans le cadre des systèmes hybrides est proposée par [Bayoudh *et al.* 2008]. Une méthode d'analyse de diagnosticabilité est utilisée pour déterminer, à partir d'une région non diagnosticable, la séquence d'actions contrôlables à appliquer sur le système pour l'amener vers une région diagnosticable. Étant donné un état incertain du diagnostiqueur actif, l'objectif du diagnostic actif est de trouver un chemin contrôlable menant à un état certain. Les auteurs proposent de résoudre ce problème comme un problème de planification conditionnelle en utilisant un graphe ET-OU. Cependant, l'algorithme n'est pas explicité, notamment le type d'exploration et le critère d'exploration de l'arbre ne sont pas donnés. Nous avons proposé de donner une définition formelle du diagnostiqueur actif et de discuter le problème de planification plus en détail.

Dans [Kuhn *et al.* 2008], l'objectif du diagnostic est de déterminer quelles ont été les actions qui ont échoué (fautes) lors de la production, sans chercher à expliquer les raisons (physiques ou logicielles) de ces fautes. Une fois le diagnostic confirmé, une intervention de réparation extérieure rapide est nécessaire afin de maintenir le système de production opérationnel. L'expérimentation porte sur des outils d'impression. La méthode s'appuie sur un diagnostiqueur qui met à jour l'état courant du système et stimule le planificateur afin qu'il génère des plans d'actions plus informatifs pour déterminer quelles actions ont échoué au cours de la production tout en maintenant autant que possible l'objectif de production. L'avantage de l'imbrication directe du diagnostic dans la planification réside dans le fait que le diagnostic est confirmé ou infirmé rapidement. Contrairement à [Kuhn *et al.* 2008], notre étude se focalise sur des systèmes dont la finalité est l'autonomie (robots, satellites) face à des pannes physiques et/ou logicielles alors qu'aucune intervention extérieure n'est possible. En conséquence les besoins de diagnostic explicatif sont plus importants car il est crucial de déterminer l'état de santé du système et le mode opérationnel qui en découle afin de savoir si la mission peut effectivement être réalisée en présence de la panne ou si une planification en mode dégradé est nécessaire.

7.2.2 Formalisation du problème de diagnostic actif

La formalisation du problème de diagnostic actif dans les Systèmes à Evénements discrets (SED) a été publiée dans [Chanthery & Pencolé 2009a, Chanthery *et al.* 2005b]. Le cadre classique du diagnostic de fautes dans les SED a été étendu au diagnostic actif. Par la suite, une adaptation aux systèmes hybrides dans lesquels la partie continue est abstraite a été proposé, dans la suite du travail de Medhi Bayoudh [Bayoudh *et al.* 2008] et dans le cadre d'un projet R&T du CNES, en collaboration avec Thalès Alénia Space.

On rappelle ici le fondement théorique de la construction d'un diagnostiqueur actif pour une faute F.

Du point de vue du modèle du système, une *action* exécutée par le contrôleur est représentée comme un événement contrôlable [Ramadge & Wonham 1989].

Définition 28 (Action) *Une action est un événement du système qui survient si et seulement si le contrôleur du système exécute l'action.*

Dans la suite, on suppose que le contrôleur notifie le diagnostiqueur de l'action exécutée. Cela signifie que n'importe quelle action sera observée par le diagnostiqueur. Soit Σ l'ensemble des événements pouvant survenir sur le système et $\Sigma_o \subseteq \Sigma$ l'ensemble des événements observables. On peut écrire $\Sigma_a \subseteq \Sigma_o$, avec Σ_a l'ensemble des actions. A la suite d'une action peut se produite une suite d'événements réactifs, observables ou pas.

Finalement, pour s'assurer que le contrôleur est toujours capable d'exécuter une action sur le système, l'étude du problème de diagnostic actif est limitée à la sous-classe de

systèmes à événements discrets pour lesquels l'hypothèse suivante tient.

HYPOTHÈSE 1 *De n'importe quel état $x \in X$, il est toujours possible d'exécuter une action $a \in \Sigma_a$ après l'occurrence d'une séquence finie d'événements réactifs $e \in \Sigma \setminus \Sigma_a$.*

Si cette hypothèse ne tient pas, le système peut atteindre un état où le contrôleur ne peut plus exécuter d'action. Dans ce cas, le problème de diagnostic actif n'a évidemment aucune solution.

L'ensemble des séquences observables possibles du système quand une faute F est survenue sur le système peut être représenté par une machine à états finis déterministe et minimale $M(F) = (S, \Sigma_o, \delta, s_0, etiq)$ où S est un ensemble fini d'états ; Σ_o est l'alphabet de la machine ; $\delta : S \times \Sigma_o \to S$ est la fonction de transition ; s_0 est l'état initial ; $etiq : S \to \{F - possible, F - impossible\}$ est la fonction d'étiquetage. Une étiquette $F - possible$ sur un état s indique qu'il est possible que F ait eu lieu entre s_0 et s. Une étiquette $F - impossible$ indique qu'il est impossible que F ait eu lieu entre s_0 et s.

Soit $M(F) = (S_1, \Sigma_o, \delta_1, s_{01}, etiq_1)$ la machine à états représentant $Traces(F)$ et $M(\neg F) = (S_2, \Sigma_o, \delta_2, s_{02}, etiq_2)$ celle de $Traces(\neg F)$. $Traces(F)$ peut être vue comme l'ensemble des séquences observables possibles du système quand la faute F est survenue sur le système, tandis que $Traces(\neg F)$ peut être vue comme l'ensemble des séquences observables possibles du système quand la faute F n'a pas lieu. La définition du diagnostiqueur actif $\Delta(F)$ tient sur les faits suivants :

1. s'il existe un plan d'actions pour diagnostiquer la faute F avec certitude alors ce plan va produire une trace observable qui appartient nécessairement à $Traces(F) \setminus Traces(\neg F)$;
2. s'il existe un plan d'actions pour diagnostiquer l'absence de la faute F avec certitude alors ce plan va produire une trace observable qui appartient nécessairement à $Traces(\neg F) \setminus Traces(F)$.

Le diagnostiqueur actif peut alors être défini comme le résultat de la synchronisation des machines $M(F)$ et $M(\neg F)$ sur les événements observables.

Le diagnostiqueur actif d'une faute F est défini comme la machine à état déterministe $\Delta(F) = (S, \Sigma_o, \delta, s_0, etiq)$ où :

- $S = S_1 \times S_2$ est l'ensemble des états ;
- Σ_o est l'ensemble des événements observables ;
- $\delta : S \times \Sigma_o \to S$ est la fonction de transition : $\forall s_1 \in S_1, s_2 \in S_2, o \in \Sigma_o, \delta((s_1, s_2), o) = (\delta_1(s_1, o), \delta_2(s_2, o))$;
- $s_0 = (s_{01}, s_{02})$ est l'état initial ;
- la fonction $etiq : S \to \{F{-}sain, F{-}s\hat{u}re, F{-}discriminable, F{-}nonDiscriminable, nonAdmissible\}$ est définie séquentiellement par les étapes suivantes. $\forall s = (s_1, s_2) \in S$:
  1. si $etiq_1(s_1) = F - possible$ et $etiq_2(s_2) = \neg F - impossible$ alors $etiq(s) = F - s\hat{u}re$;
  2. si $etiq_1(s_1) = F - impossible$ et $etiq_2(s_2) = \neg F - impossible$ alors $etiq(s) = nonAdmissible$;
  3. si $etiq_1(s_1) = F - impossible$ et $etiq_2(s_2) = \neg F - possible$ alors $etiq(s) = F - sain$;
  4. si $etiq_1(s_1) = F - possible$ et $etiq_2(s_2) = \neg F - possible$, alors on étudie deux cas : s'il existe une séquence de transitions de s vers s' dans $\Delta(F)$ telle que $etiq(s') = F - s\hat{u}re$ ou $etiq(s') = F - sain$, alors $etiq(s) = F - discriminable$, sinon $etiq(s) = F - nonDiscriminable$.

En résumé, le diagnostiqueur actif de la faute F rajoute donc une étiquette pour chaque état. Le tableau 7.1 résume les étiquettes possibles sur chaque état et leur pertinence par rapport au diagnostic actif.

TABLE 7.1 – Résumé des étiquettes possibles sur chaque état d'un diagnostiqueur actif.

Etiquette	Interprétation	Intérêt pour le diagnostic actif
$F - nonDiscriminable$	On ne pourra jamais dire si la faute F a eu lieu ou non	NON : ces états ne sont pas intéressants
$F - discriminable$	Il existe au moins un moyen de savoir si la faute F a eu lieu ou non	OUI : ces états sont intéressants pour le diagnostic actif
$F - sain$	La faute F n'a pas eu lieu	NON : le diagnostic n'est pas ambigu
$F - sûre$	La faute F a eu lieu	NON : le diagnostic n'est pas ambigu

Si on considère un système G sur lequel les fautes $F_1, \ldots, F_n$ sont les fautes anticipées, on note $\Delta(F_i) = (S_i, \Sigma_o, \delta_i, s_{0i}, etiq_i)$ le diagnostiqueur actif de la faute F_i. Le diagnostiqueur actif Δ de G est alors défini comme l'union des diagnostiqueurs spécialisés définie comme suit. Pour toute séquence d'observations σ, le diagnostiqueur spécialisé $\Delta(F_i)$ atteint l'état s_i avec une étiquette $etiq(s_i)$. Le résultat du diagnostiqueur actif Δ après l'observation de σ est donc : $s = s_1, \ldots, s_n$ et $etiq(s) = \{etiq(s_1), \ldots, etiq(s_n)\}$.

Le diagnostiqueur actif fournit donc les informations suivantes :

- le diagnostic courant : pour n'importe quelle faute F, il fournit le statut courant de la présence de F. Par exemple, si $\forall i \in \{1, \ldots, n\}$, le statut de F_i est sain, cela signifie qu'aucune faute n'est survenue ;
- le statut de la session de diagnostic actif : le diagnostiqueur indique l'utilité du déclenchement d'une session de diagnostic actif via l'étiquette de chaque faute.

La figure 7.1 illustre deux diagnostiqueurs actifs pour deux fautes F_1 et F_2 dans un exemple de système à deux réservoirs.

Le diagnostiqueur spécialisé de la faute F_1 est donné à gauche de la figure et celui de la faute F_2 à droite. On constate que seuls deux états sont intéressants pour débuter une session de diagnostic actif : il s'agit des deux états bleus, qui correspondent à des états étiquetés $F - discriminable$. L'objectif du diagnostic actif va être de partir d'un état bleu et de guider le système vers un état vert (étiqueté $F - sain$) ou rouge (étiqueté $F - sûre$). Les états oranges (étiquetés $F - nonDiscriminable$) sont à éviter.

7.2.3 Algorithme de diagnostic actif

Un algorithme de diagnostic vise à résoudre le problème suivant : étant donné les diagnostiqueurs actifs de chacune des fautes F d'un système, étant donné l'état courant dont au moins une faute est étiquetée $F - discriminable$, on cherche à trouver une séquence d'actions qui permette de mener le système dans un état où l'ambiguïté est réduite [1].

Le travail sur la partie algorithmique du diagnostic actif a été publié dans [Chanthery *et al.* 2010c, Chanthery & Pencolé 2009].

L'algorithme proposé repose sur un AO* [Bonet & Geffner 2000] et calcule un plan conditionnel visant à raffiner le diagnostic. Un algorithme AO* est un algorithme qui permet

1. Au mieux, l'ambiguïté est complètement levée pour toutes les fautes. Il peut y avoir néanmoins des fautes pour lesquelles l'ambiguïté reste.

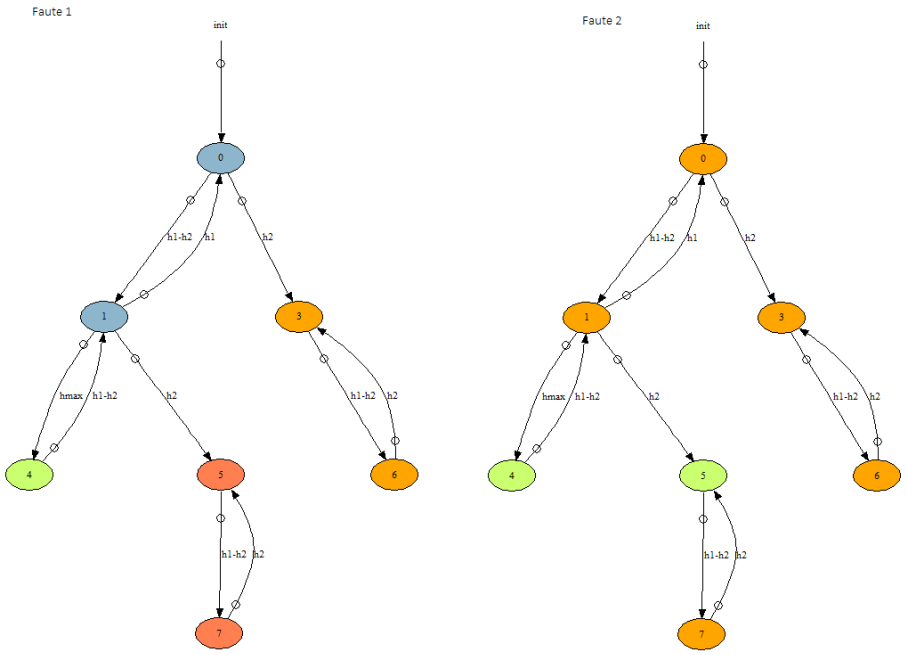


FIGURE 7.1 – Diagnostiqueurs actifs pour un système G.

de résoudre un problème de génération de plans basé sur un arbre ET/OU. Contrairement à l'algorithme A*, qui permet de résoudre un problème basé sur un arbre ET, la notion de dépendance vis-à-vis des nœuds parents doit être prise en compte.

Dans un algorithme de diagnostic actif, les nœuds ET correspondent à une ou plusieurs actions effectuées par le système, et les nœuds OU correspondent à une séquence d'observations possible résultant de cette action.

Un premier point à soulever est la difficulté de passer du diagnostiqueur actif à l'arbre ET/OU sur lequel se base l'algorithme. En effet, toutes les combinaisons d'actions doivent pouvoir être envisagées, ainsi que toutes les séquences d'actions, le diagnostiqueur n'étant pas uniquement composé d'une succession action/observation. Des hypothèses simplificatrices ont été adoptées dans certains travaux, notamment dans [Chanthery *et al.* 2010c]. Les algorithmes ont ensuite été améliorés pour répondre à cette contrainte.

L'algorithme 2 est l'algorithme de base pour le diagnostic actif. Il prend en entrée un diagnostiqueur actif et construit à la volée une structure d'arbre ET/OU sur laquelle est effectuée une recherche de plus court chemin vers un nœud où toutes les fautes sont discriminées.

Plusieurs points sont à étudier.

A l'initialisation, ligne 1, le nœud de départ doit être choisi dans l'ensemble des nœuds du diagnostiqueur actif ayant un intérêt. Autrement dit, il faut qu'il existe au moins une faute F ayant une étiquette $F - discriminable$ pour que le lancement d'une recherche de plan de diagnostic actif ait un intérêt. Lignes 4 et 17, il faut sélectionner un nœud OU à développer dans la suite de l'algorithme. Cette sélection se fait en utilisant un critère de coût, pouvant prendre en compte des notions de coûts de réparation de certaines fautes et des évaluations (heuristiques) sur les explorations futures. Ligne 7, les nœuds ET successeurs du nœud OU considéré doivent être créés. Dans certains cas, tous les successeurs ne doivent pas être considérés (circuit, nœuds physiquement impossibles, etc). Ligne 10, une heuristique est calculée pour chaque nœud ET de l'ensemble des suc-

Algorithme 2 : Algorithme de Diagnostic Actif

```
1  Initialisation ;
2  Build RootNode ;
3  ORnodeToBeDeveloped ← RootNode ;
4  CurrentNode = NodeSelection(ORnodeToBeDeveloped) ;
5  tant que ORnodeToBeDeveloped n'est pas vide faire
6      tant que (CurrentNode.status = unsolved) et (CurrentNode.status = not
        unsolvable) faire
7          NextAndNodes ← CreateSuccessorNodes(CurrentNode) ;
8          pour tous les ANDnode dans NextAndNodes faire
9              ANDnode.tag ← GetTag(ANDnode.Successor) ;
10             ANDnode.Heuristic = ComputesHeuristic(ANDnode);
11         ANDnodeToBeDeveloped = BestSuccessor(NextAndNodes);
12         NextOrNodes = CreateSuccesorsNodes(ANDnodeToBeDeveloped);
13         Trier les nœuds de NextOrNodes suivant un critère;
14         Elimination des cycles;
15         Placer NextOrNodes au début de ORnodeToBeDeveloped;
16         CurrentNode = NodeSelection(ORnodeToBeDeveloped);
17     CurrentNode = NodeSelection(ORnodeToBeDeveloped) ;
18 Edition du plan conditionnel ;
```

cesseurs. Cette heuristique doit être minorante pour être admissible. La recherche d'heuristiques admissibles efficaces est encore en cours d'investigation. Des propositions ont été faites dans [Chanthery *et al.* 2010c]. La ligne 11 consiste à sélectionner un nœud ET à développer. Ceci peut avoir lieu dans un ordre quelconque étant donné que tous les nœuds ET sont à considérer. La ligne 12 recherche les nœuds OU successeurs d'un nœud ET. La ligne 14 consiste à éliminer les circuits de la recherche. Cette étape est particulière dans la recherche d'un plan de diagnostic actif. En effet, en planification "traditionelle", l'étape d'élimination des cycles est classique : elle permet d'éviter de remettre le système dans un état qu'il a déjà visité. Dans le cas du diagnostic actif, il est possible qu'un plan doive effectuer un cycle pour raffiner le diagnostic d'une faute, puis revienne dans un état déjà exploré pour raffiner le diagnostic d'une autre faute. L'élimination de cycle doit donc être effectuée de manière très minutieuse.

Cet algorithme a été implémenté en C++ lors du stage de Nicolas Bussac (2010), en collaboration avec Yannick Pencolé, puis sous Matlab lors du projet R&T du CNES, par Nicolas Garin, Quentin Gaudel et moi-même, en collaboration avec Louise Travé-Massuyès et Yannick Pencolé.

7.2.4 Le diagnostic actif dans une architecture embarquée

Plusieurs pistes ont été envisagées pour inclure le diagnostic actif dans une architecture embarquée.

Trois modules peuvent être identifiés et doivent coopérer au niveau décisionnel dans l'architecture embarquée. Le diagnostiqueur actif contrôle le système et détecte des situations où il est utile de déclencher une session de diagnostic actif ; le planificateur pour le diagnostic prend comme entrée un problème de planification P donné par le diagnostiqueur actif et calcule, via un algorithme de diagnostic actif, un arbre solution pour raffiner le

diagnostic. Enfin, le planificateur de mission prend comme entrée l'état courant du système et calcule un plan de mission qui réalise un sous-ensemble faisable d'objectifs en satisfaisant les contraintes temporelles et les limites des ressources, tout en maximisant les récompenses attendues. Ces trois tâches informatiques doivent être intégrées dans une architecture embarquée. Pour gérer les conflits entre les différents modules fonctionnels et maintenir une description logique de leurs états, une couche de contrôle d'exécution doit être introduite.

Dans un premier temps, une architecture a été proposée en utilisant l'outil ProCoSA pour effectuer le contrôle d'exécution [Chanthery & Pencolé 2009]. La solution proposée pour gérer les conflits entre la planification pour le diagnostic et la planification de mission est d'arrêter l'exécution du plan de mission si le diagnostiqueur actif détecte un état ambigu qui doit être désambiguïsé. Suite à cette détection, le contrôleur d'exécution doit ouvrir une session de diagnostic actif, appliquer le plan pour le diagnostic, la réparation si nécessaire, fermer la session de diagnostic actif et lancer une replanification de mission. La replanification est nécessaire car le système et son environnement sont dynamiques : appliquer une action implique une évolution de l'état du système et de son environnement. La solution est illustrée sur la figure 7.2. À droite, on trouve les trois modules fonctionnels impliqués : le diagnostiqueur actif, le planificateur pour le diagnostic et le planificateur de mission. L'outil ProCoSA est utilisé pour spécifier les interactions entre ces modules grâce au réseau de Petri à gauche de la figure. Au début, le système exécute sa mission. Quand le diagnostiqueur actif détecte une situation où une session de diagnostic actif est nécessaire, il envoie un message "état ambigu". Le contrôleur d'exécution arrête l'exécution de la mission et envoie une requête de planification pour le diagnostic, associé au problème P fourni par le diagnostiqueur actif. Le planificateur pour le diagnostic cherche l'arbre solution et l'envoie au contrôleur. La place "ACTION pour le DIAGNOSTIC" représente l'application de l'arbre solution. Trois cas sont alors possibles : (Cas 1) L'arbre solution "réussit", c'est-à-dire que le diagnostiqueur actif est dans un état où toutes les fautes sont étiquetées $F-sûre$ ou $F-sain$. Le contrôleur envoie alors une requête au planificateur de mission pour replanifier avec le nouveau contexte de mission. Le nouveau plan peut inclure des actions de réparation. (Cas 2) Après une action, le diagnostiqueur actif se retrouve dans un état où une faute est étiquetée $F-nonDiscriminable$. L'arbre solution échoue car il n'y a aucun moyen pour affiner le diagnostic : la session de diagnostic actif doit être fermée. Le contrôleur d'exécution envoie une requête au planificateur de mission pour replanifier avec un contexte de mission dégradé et incertain. (Cas 3) L'arbre solution est réduit à un nombre limité d'actions et après la dernière action, le diagnostiqueur actif est dans un état où certaines fautes sont encore étiquetées $F-discriminable$. L'arbre solution échoue mais le diagnostiqueur actif est dans un état où certaines fautes sont toujours étiquetées F-discriminable, le contrôleur d'exécution envoie alors une nouvelle requête de planification pour le diagnostic, associé à un nouveau problème P fourni par le diagnostiqueur actif. Quand une requête arrive sur le planificateur de mission, il replanifie la mission et envoie le meilleur plan de mission au contrôleur qui suit son exécution.

Le stage de Julien Salvy a permis de mettre en œuvre ce contrôle d'exécution. L'ensemble des réseaux de Petri ont été codés, ainsi que les différents processus envisagés.

Par la suite, le projet R&T du CNES en 2014, intitulé "diagnostic actif par OBCP (On-Board Control Procedure) a donné lieu a une autre implémentation de l'architecture, en collaboration avec Thalès Alénia Space. A cette occasion le logiciel ActHyDiag sous Matlab, a été développé. La publication [Chanthery *et al.* 2015] en reprend l'idée principale. Une OBCP est une procédure à exécuter à bord d'un satellite pouvant être chargée, exécutée et remplacée sans modifier le logiciel de vol. Le cas d'étude retenu pour l'expéri-

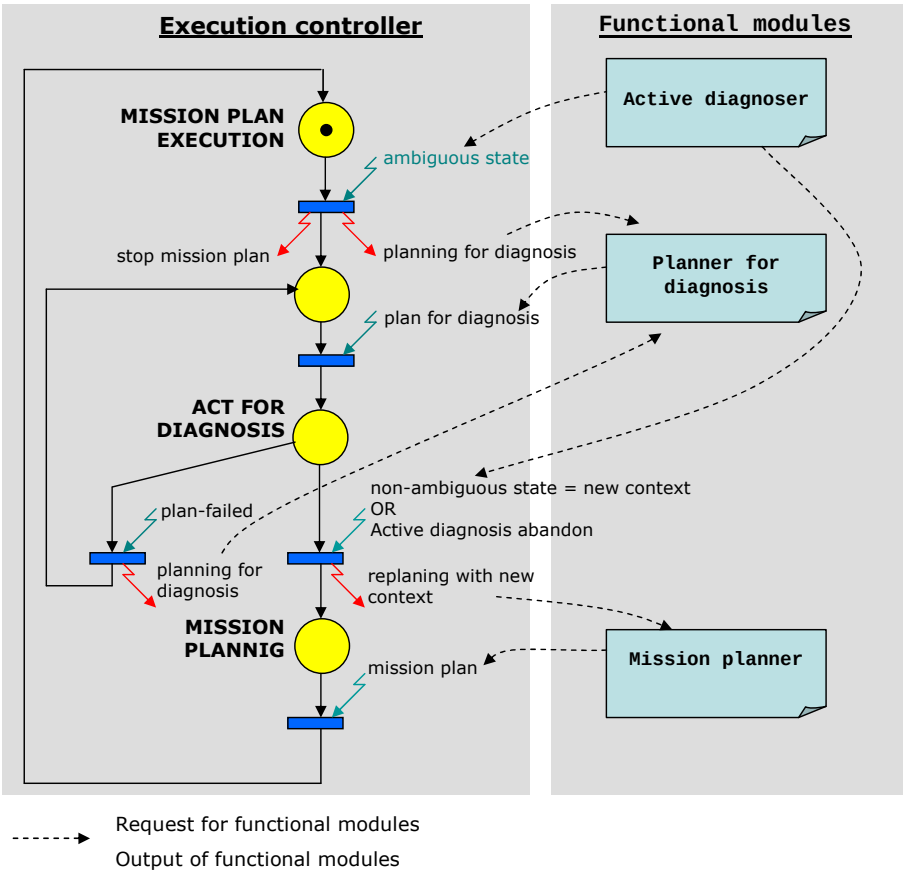


FIGURE 7.2 – Diagnostic actif et planification de mission dans une architecture embarquée.

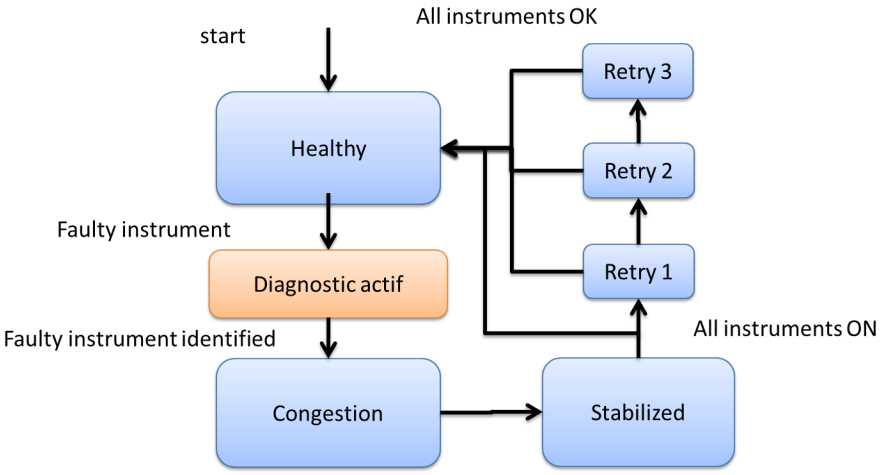


FIGURE 7.3 – Machine à état de l'OBCP de surveillance du réseau FDIR incluant le diagnostic actif.

mentation est celui de la surveillance d'un réseau Spacewire. Une modélisation générique d'un composant a été proposée, et nous avons généré automatiquement un modèle à 6 composants. La génération peut être étendue à n composants si nécessaire pour l'étude du passage à l'échelle. Les diagnostiqueurs actifs ont également été générés. Un plan de diagnostic actif a été produit et validé en réel sur une maquette fournie par Thalès Alénia Space. La gestion des différents processus a été menée par Thalès Alénia Space. La figure 7.3 présente la machine à état de l'OBCP de surveillance du réseau SpaceWire incluant le diagnostic actif proposé par Thalès Alénia Space. L'expérimentation du diagnostic actif par OBCP à l'aide de la maquette a permis de soulever des questions intéressantes (passage à l'échelle, complexité de la modélisation des aspects dynamiques,...) et de mettre en lumière l'intérêt de la méthode outillée définie dans cette étude. Cette approche est validée par l'expérimentation sur un cas simple, et permet de se faire une idée des différentes étapes nécessaires à l'implémentation de diagnostic actif à l'aide d'OBCP.

Les rapports de projet [Chanthery *et al.* 2014], [Delandrea *et al.* 2014] et [Ferluc *et al.* 2014] présentent nos travaux lors du R&T du CNES avec Thalès Alénia Space sur le diagnostic actif par OBCP.

7.3 Embarquabilité des algorithmes de diagnostic

Personnes impliquées dans cette thématique et affiliation lors de la collaboration : Yannick Pencolé (CR, DISCO), Louise Travé-massuyès (DR, DISCO), Emmanuel Bénazéra (Post-doc, DISCO), Pierre-Jean Meyer (Stagiaire, DISCO)

7.3.1 Diagnostic anytime

Un des premiers travaux concernant l'embarquabilité des algorithmes de diagnostic actif concerne le diagnostic anytime ou "à la demande". Nous avons abordé cette problématique avec Yannick Pencolé, en proposant un stage de M2 sur le thème du diagnostic à la demande sur un système à événements discrets, effectué par P-J Meyer [2] (février/septembre

2. Les publications de P-J. Meyer et notamment son mémoire de M2 sont disponibles sur internet à l'adresse suivante https://people.kth.se/~pjmeyer/publi.html

2011).

Nous sommes partis du constat que le diagnostiqueur de Sampath [Sampath *et al.* 1996], extrêmement efficace et rapide, nécessite néanmoins un espace mémoire trop important pour être embarquable pour des systèmes réels complexes. En effet, le nombre d'états du diagnostiqueur est au pire exponentiel en le nombre d'états du modèle global, alors que le nombre d'états du modèle global est au pire exponentiel en le nombre de composants (synchronisation des automates des modèles des composants). La taille du diagnostiqueur est donc doublement exponentielle en le nombre de composants du système. Or, comme cela a été dit précédemment, le diagnostic est principalement utilisé pour améliorer l'autonomie des systèmes, notamment dans des systèmes embarqués qui ont généralement de très fortes contraintes en terme d'espace mémoire. Il existe tout un spectre d'algorithmes de diagnostic allant de l'algorithme sans pré-compilation à base de composants au diagnostiqueur de Sampath [Schumann *et al.* 2007]. Afin de pouvoir embarquer un algorithme de diagnostic, on peut travailler à partir d'un modèle moins développé que celui du diagnostiqueur de Sampath. Cette solution permet d'abaisser les besoins en espace mémoire à une valeur plus convenable pour une application embarquée. Toutefois, cette réduction de la pré-compilation va augmenter le temps de calcul, qui pourra dépasser le temps entre deux observations consécutives, et donc de ne pas respecter les contraintes temps réel de l'application. Il faudrait donc trouver une solution pour pouvoir traiter une observation, sans dépasser le temps de calcul disponible et sans avoir recours directement à la machine à états finis du diagnostiqueur complet.

Une autre difficulté de la mise en place d'un diagnostiqueur vient de la nécessité de donner l'état de santé du système dès que le planificateur en fait la demande. Il faut donc trouver une solution pour permettre au diagnostiqueur de renvoyer un résultat au planificateur même lorsqu'il n'a pas eu le temps de terminer les calculs menant au résultat final. Pour ces deux raisons, nous avons décidé de nous orienter vers l'utilisation d'un algorithme de type "anytime". Les algorithmes anytime donnent en effet la possibilité de faire des concessions sur la qualité du résultat dans le but de diminuer le temps de calcul [Zilberstein 1996]. Ces algorithmes sont aussi construits de manière à améliorer la qualité du résultat avec le temps de calcul disponible. Deux types d'algorithmes coexistent dans la littérature : les algorithmes sont dits « à contrat » lorsque le temps alloué à l'algorithme est planifié à l'avance ; les algorithmes « interruptibles » peuvent quant à eux être interrompus sans savoir à l'avance quand en viendra la demande. C'est un algorithme de ce dernier type que nous avons développé dans le stage de P-J. Meyer.

Dans ses travaux, Zilberstein introduit aussi la notion étendue de « qualité » évoquée précédemment. Le choix du type de mesure de qualité liée à un algorithme anytime dépend à la fois de la manière dont le résultat exact est approché, mais aussi de l'utilisation qui en est faite ensuite. Parmi les exemples les plus utilisés, on peut mesurer la qualité du résultat selon :

- la *certitude*, en calculant par exemple la probabilité que le résultat soit correct ;
- la *précision* du résultat par rapport à la solution optimale, qui est généralement utilisée dans les algorithmes de planification [Likhachev *et al.* 2005] ;
- la *spécificité*, qui mesure le niveau de détail du résultat.

Dans le cadre d'un algorithme de diagnostic, c'est ce dernier type de mesure de qualité que nous allons choisir. En effet, le plus important lorsque l'on retourne un résultat au planificateur, c'est que ce résultat soit correct quelle que soit la situation : on préfère obtenir un résultat ambigu que de générer un diagnostic erroné. Ainsi, dans les algorithmes, le résultat doit toujours contenir la solution finale. La qualité du résultat s'améliore lorsque les possibilités présentes dans le résultat courant sont confirmées ou infirmées.

Ces notions de qualité du diagnostic dépendent également des liens entre le diagnostic et la décision [van Harmelen & ten Teije 1995]. En effet, il n'est par exemple pas toujours nécessaire d'obtenir un diagnostic totalement certain pour pouvoir prendre la bonne décision : un diagnostic ambigu peut suffire.

Il existe beaucoup d'autres types d'approximations possibles pour faciliter l'obtention et l'utilisation des résultats d'un algorithme de diagnostic, qu'il soit classique (comme dans le dernier exemple) ou anytime. Si par exemple on souhaite réaliser un diagnostic sur un système distribué ou décentralisé, composé d'un ensemble de sous-systèmes, il est possible de partir d'un diagnostic ambigu concernant une faute sur le système, puis d'améliorer la qualité du diagnostic en étudiant ses sous-systèmes [Cobb *et al.* 1999]. Il est aussi possible de travailler dans l'autre sens en cherchant tous les sous-systèmes fautifs, pour connaître ensuite l'ensemble de sous-systèmes le plus petit à remplacer. Une autre possibilité est de chercher le diagnostic impliquant le moins de sous-systèmes possibles [Verberne *et al.* 2000]. Ainsi, sur un même type de modélisation, on a déjà trouvé trois types d'approximations pour extraire un diagnostic, ce qui montre bien que la manière de réaliser les recherches est grandement influencée par l'objectif du diagnostic : isolation, réparation rapide, réparation économique, ...

Un dernier exemple d'approximation pour le diagnostic peut être utilisé lorsque le modèle du système est sous la forme d'une liste de règles de comportement [Mouaddib & Zilberstein 1995]. Un diagnostic peut être obtenu grâce à une abstraction consistant à omettre les règles les moins importantes. S'il reste du temps de calcul disponible après cela, on pourra alors relancer l'algorithme avec davantage de règles pour raffiner le diagnostic.

Un algorithme de diagnostic « interruptible » a été mis en place.

L'architecture de cet algorithme est composée de plusieurs processus fonctionnant en parallèle.

Une fonction principale va lancer et interrompre la fonction de diagnostic en fonction des besoins sur le système (requête du module de décision, observation particulière...).

La fonction de diagnostic prend en entrée le modèle global du système, les observations courantes et le dernier diagnostic obtenu, appelé *diagPrécédent*. Le diagnostic initial, appelé *diagCourant* est obtenu à partir des observations courantes : il contient tous les états cibles d'une transition associée à l'observation courante. L'objectif de la fonction de diagnostic est donc de "raccrocher" les états de *diagPrécédent* et ceux de *diagCourant* via la séquence d'observations connue. Deux procédés peuvent améliorer la qualité du diagnostic. La confirmation d'une solution pour un état intervient quand un chemin est trouvé entre un état de *diagPrécédent* et un état de *diagCourant*. Le raffinement d'un état intervient lorsque l'on sait qu'aucune autre solution ne peut être confirmée pour cet état. Le raffinement peut avoir lieu lorsque toutes les combinaisons de fautes sont présentes dans les solutions confirmées liées à cet état, ou bien lorsque tous les chemins menant à cet état ont été explorés. Lorsqu'un état est raffiné, alors il est retiré des états connus pour être non raffinés.

Deux stratégies de recherche sont combinées pour parcourir l'automate du modèle afin de trouver des solutions à confirmer : une recherche en avant, en largeur d'abord, démarre des états de *diagPrécédent* et essaye de trouver ceux de *diagCourant* ; une recherche en arrière, en profondeur d'abord, démarre à partir des états de *diagCourant* pour retrouver ceux de *diagPrécédent*. Ces deux recherches allient leurs avantages en une recherche bidirectionnelle. La recherche en arrière raffine les états, mais est lente. La recherche en avant est rapide pour l'exploration et la confirmation de solutions, mais ne raffine pas les états.

Enfin, une fonction permet de raffiner les abstractions obtenues dans les anciennes

étapes interrompues.

L'utilisation de processus permet de n'utiliser qu'une partie de ces fonctions, en créant ainsi 6 versions différentes de l'algorithme. Afin de pouvoir mesurer les performances de l'algorithme et de ses variantes, un ensemble de critères de qualité a été mis en place, ainsi que deux programmes permettant de comparer les performances de l'algorithme à celles d'un algorithme classique (pas "anytime"). Enfin, une série de tests a été effectuée pour démontrer l'utilité de notre algorithme, trouver ses limitations, et déterminer quelles versions sont les plus intéressantes.

On ne détaille pas ici les résultats de ces tests qui sont disponibles dans le rapport de stage de Pierre-Jean Meyer [Meyer 2011]. Néanmoins on peut tirer de ces travaux les conclusions suivantes :

⇨ **Variabilité du modèle**

Les résultats obtenus varient suivant le type de modèle à traiter. Le nombre d'états, le nombre de transitions, d'événements, mais surtout la répartition des événements (proportion de fautes, d'observations dans l'ensemble des événements) va beaucoup jouer sur les performances des algorithmes.

⇨ **Transitions sortantes**

En règle générale, l'algorithme anytime est moins performant que l'algorithme de base, non interruptible. Si on considère un système avec 80 états et 40 événements, utiliser un algorithme anytime ne commence à être intéressant que pour des modèles avec un grand nombre de transitions sortantes, au delà de 9. L'algorithme parvient à calculer des solutions d'une qualité égale à 80% de la qualité d'un algorithme non interruptible.

⇨ **Observations**

Lorsque le modèle du système contient beaucoup d'observables par rapport au nombre d'événements, l'algorithme de base est très rapide face à l'algorithme anytime. Au contraire, lorsque le système est très peu observable, le temps de calcul de l'algorithme anytime peut être jusqu'à 4 fois plus faible que pour l'algorithme de base.

⇨ **Fautes**

Si on fait varier le nombre de fautes dans le système, on observe que, comme pour un algorithme classique, le temps de calcul est exponentiel en le nombre de fautes. Le temps de calcul de l'algorithme anytime varie de la même manière que celui de l'algorithme de base.

⇨ **Conclusion**

Quelle que soit la version utilisée pour l'algorithme anytime, s'il n'est pas interrompu, il obtiendra les mêmes résultats qu'un algorithme non interruptible. Les performances des algorithmes anytime ne sont évidemment pas meilleures que celles d'un algorithme non interruptible, mais restent néanmoins acceptables en terme de qualité, sauf pour des systèmes trop "simples" (peu d'états, d'événements, de transitions, peu de fautes, grande observabilité). De même, pour les modèles avec un grand nombre de fautes, l'algorithme obtient de mauvais résultats. Certaines pistes pourraient être explorées pour améliorer ces travaux. Tout d'abord, on pourrait envisager de prendre en compte des requêtes du module de décision pour orienter les recherches vers des états d'intérêt. Enfin, pour pallier la difficulté des problèmes trop grands, on peut envisager de décentraliser ou de distribuer l'approche anytime en considérant des sous-systèmes traitables efficacement.

7.3.2 Une intégration directe du diagnostic dans les plans : la solution des POMDP

Une seconde voix d'exploration pour l'embarquabilité des algorithmes de diagnostic a été menée lors du projet SIRASAS (Stratégies Innovantes et Robustes pour l'Autonomie

des Systèmes Aéronautiques et Spatiaux) avec Emmanuel Bénazéra entre novembre 2007 et avril 2009. Notre objectif était d'établir des liens directs entre les tâches de diagnostic/pronostic et de planification/reconfiguration. Ayant tous les deux des compétences dans le domaine de la planification, nous nous sommes penchés sur le formalisme des Processus Décisionnels de Markov Partiellement Observables (PDMPO) comme une solution holistique pour intégrer dès la conception du plan des tâches de reconfiguration et de réparation, en considérant que l'observation faite sur le système pouvait donner une information de diagnostic/pronostic pertinente. Ces travaux ont donné lieu à un article [Benazera & Chanthery 2008] et à des rapports de projets [Staroswiecki *et al.* 2009], [Zolghadri *et al.* 2009].

Ce travail part du constat que de nombreuses difficultés apparaissent lorsque les tâches de diagnostic, de monitoring et de reconfiguration/décision sont mises en jeu. Comme on l'a souligné dans la section 5.3, page 29, la plupart des architectures considèrent séparément les processus de diagnostic, décision et pronostic. Par ailleurs, il est rare que l'état d'un système, et donc son diagnostic, soit connu avec certitude. Enfin, dans la plupart des cas, les actions de réparation de plan à court terme sont souvent préférées à des replanifications globales. Néanmoins, les conséquences d'une panne dans l'accomplissement d'une mission peuvent être catastrophiques. On souhaite donc anticiper les pannes au plus tôt en faisant ce que l'on appelle de la maintenance préventive. L'idée de ces travaux est d'intégrer les tâches de diagnostic et de pronostic aux tâches de planification et d'observation.

Les PDMPO permettent d'appréhender des problèmes de commande pour lesquels les actions ont des effets stochastiques et les capteurs fournissent des informations imparfaites et incomplètes. Ce modèle a été largement adopté par la communauté de l'Intelligence Artificielle dans les domaines de la planification et de la commande sous incertitude. Le modèle permet de modéliser l'incertitude des capteurs et des actions, l'incertitude sur la connaissance de l'état et des objectifs multiples. La solution d'un PDMPO est la chaîne d'actions optimale pour toutes les croyances possibles sur le monde. Cette chaîne d'actions optimale est appelée politique optimale. Ainsi, s'il y a un gain à faire en évitant ou en atténuant certains effets (notamment les pannes), la solution d'un PDMPO le capturerait dans sa politique optimale.

Bien que le cadre des PDMPO soit intéressant, de nombreuses difficultés restent à appréhender. La première difficulté réside dans le calcul de solutions pour des systèmes réels. Différentes solutions existent pour obtenir des solutions approximées de très bonne qualité : les fonctions de valeurs linéaires par morceaux, les grilles à pas fixes ou variables ou encore la compression de l'espace d'état de croyance. On peut les ramener à trois techniques : l'analyse d'atteignabilité, les recherches heuristiques et les représentations factorisées. L'article [Benazera & Chanthery 2008] dresse l'analyse de ces solutions, en particulier pour des modèles avec des fautes. On retiendra les difficultés suivantes pour les problèmes de diagnostic de fautes :

- les modèles qui incluent des fautes ont généralement un nombre d'états élevé (supérieur à 100) et les méthodes qui utilisent une représentation convexe linéaire par morceaux de la fonction de valeur sont souvent limitées à une centaine d'états.
- les techniques qui utilisent des analyses d'atteignabilité souffrent du fait que les fautes peuvent survenir à tout instant : l'espace atteignable est donc très large très rapidement. De plus, les états défaillants sont souvent les moins probables, ils sont donc rejetés dans la plupart des méthodes par compression ou abstraction.

L'idée générale de ces travaux, qui visent à inclure directement des actions de diagnostic et de maintenance dans le plan de mission d'un système, a été reprise lors du stage de Ben Alia Bouzidi co-encadré avec Pauline Ribot et Florent Teichteil-Koenigsbuch du groupe

Airbus en 2017, intitulé "Gestion de santé assistée par des outils à base de modèles et de données". Le cadre d'un stage n'a pas été suffisant pour une avancée significative des travaux, mais a ouvert des perspectives intéressantes à étudier dans de futurs travaux.

7.4 Optimisation de la sélection de tests pour le diagnostic

Personnes impliquées dans cette thématique et affiliation lors de la collaboration : Louise Travé-massuyès (DR, DISCO), Christian Artigues (DR, ROC), Nicolas Jozefowiez (MdC INSA, ROC), Carine Jauberthie (MdC UPS, DISCO), Matthieu Godichaud (Post-Doc, DISCO), Olivier Buffet (INRIA), Marc Contat (Cassidian), Asma Gasmi (Stagiaire, ROC-DISCO).

On peut définir un test comme une mesure ou un ensemble de mesures sur le système dont l'objectif est de déduire un diagnostic ou un pronostic.

Le *problème de sélection de tests optimaux* vise à minimiser le coût des tests tout en satisfaisant certaines contraintes d'isolation sur les fautes. Ce problème est également lié au *problème de placement de capteurs*, qui recherche l'ensemble le moins coûteux de capteurs qui satisfait certaines propriétés (détection et isolation possible de toutes les fautes par exemple). Ce problème a été largement abordé dans la littérature.

☞ Rappelons qu'une large partie de mes travaux sur cette thématique se positionnent dans un cadre non centralisé, en utilisant l'approche structurelle. Parce qu'ils nécessitent la compréhension de concepts de base en analyse structurelle, tous ces travaux sont exposés dans le chapitre suivant.

J'ai néanmoins pris part à d'autres travaux en lien avec l'optimisation de la sélection de tests, qui n'abordent pas l'analyse structurelle.

Entre 2009 et 2011, j'ai été responsable d'un projet pour l'équipe DISCO sur le thème de l'optimisation sous contraintes avec M. Gaudichaud. Ce projet faisait partie d'un programme d'études amont conduit par la DGA. Nous avons collaboré avec EADS Defense & security et le LORIA. L'objectif était d'optimiser l'emploi de systèmes capteurs. Au cours de ce projet, nous avons rédigé deux rapports techniques [Chanthery *et al.* 2010a], [Chanthery *et al.* 2010b] et deux articles de conférences [Godichaud *et al.* 2011b], [Godichaud *et al.* 2011a].

J'ai également collaboré avec Carine Jauberthie sur ses travaux sur la recherche d'entrées optimales pour l'estimation de paramètres incertains. L'approche développée a consisté à combiner une méthode de programmation dynamique et l'analyse par intervalles pour la synthèse de commandes optimales. Ces travaux ont fait l'objet d'une publication [Jauberthie & Chanthery 2013].

7.5 Publications liées à cette partie

▶ E. Chanthery, M. Barbier et J-L. Farges. Planning, scheduling and constraint satisfaction: from theory to practice, chapitre Integration of Mission Planning and Flight Scheduling for Unmanned Aerial Vehicles. IOS Press, 2005

▶ E. Chanthery et Y. Pencolé. *Monitoring and active diagnosis for discrete-event systems*. IFAC Proceedings Volumes, vol. 42, no. 8, pages 1545–1550, 2009

▶ E. Chanthery, Y. Pencolé et N. Bussac. *An AO*-like algorithm implementation for active diagnosis*. In 10th International Symposium on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS, 2010

▶ E. Chanthery et Y. Pencolé. *Modélisation et intégration du diagnostic actif dans une architecture embarquée*. Journal européen des systèmes automatisés, vol. 43, no. 7-9, pages 789–803, 2009

▶ E. Chanthery, B. Delandrea, R. De Ferluc, N. Garin et L. Travé-Massuyès. *Diagnostic actif par OBCP. WP1000: analyse de la problématique.* Rapport de contrat : Thales alenia space. diagnostic embarqué par OBCP, no. 14368, Thalès Alenia Space; LAAS-CNRS, DISCO, 2014

▶ B. Delandrea, C. Le Peuvedic, R. De Ferluc, E. Chanthery, L. Travé-Massuyès et N. Garin. *R&T CNES diagnostique actif par OBCP. Rapport de Lot 2 : analyse et solutions de diagnostic actif à base d'OBCP.* Rapport de contrat : Thales alenia space. diagnostic embarqué par OBCP, no. 14691, Thalès Alenia Space; LAAS-CNRS, DISCO, 2014

▶ R. De Ferluc, E. Chanthery et L. Travé-Massuyès. *Diagnostic actif par OBCP. WP4000: prototypage de diagnostic actif à base d'OBCP.* Rapport de contrat : Thales alenia space. diagnostic embarqué par obcp, no. 5461, Thalès Alenia Space; LAAS-CNRS, DISCO, 2014

▶ E. Benazera et E. Chanthery. *The Challenge of Solving POMDPs for Control, Monitoring and Repair of Complex Systems.* In Proceedings of the 19th International Workshop on Principles of Diagnosis (DX'08), 2008

▶ M. Staroswiecki, J. Ragot, D. Henry, A. Zolghadri, J. Cieslak, D. Maquin, B. Marx, T. Raissi, R. Pons, C. Jauberthie, L. Travé-Massuyès, E. Benazera, E. Chanthery, D. Berdjag, C. Join, D. Theilliol, S. Canitrot, T. Hamel et F. Hamelin. *SIRASAS. Deliverable no.1. WP2.0.* Rapport de contrat : Projet SIRASAS - contrat FRAE, mars 2009, 289p. , no. 09149, DISCO; IMS Bordeaux; CRAN-ENSEM; SATIE, 2009

▶ A. Zolghadri, M. Staroswiecki, L. Travé-Massuyès, J. Ragot, P. Dague, E. Bensana, M-C. Charmeau, P. Goupil, X. Olive, E. Benazera, E. Chanthery, C. Jauberthie et R. Pons. *Appel à projet no 3. Programme de recherche "Autonomie des systèmes aéronautiques et spatiaux".* Rapport de contrat : Projet SIRASAS - contrat FRAE, mars 2009, 20p. , no 09150, Thalès Alenia Space; AIRBUS France; CNES; ONERA; LRI; LIPN; CRAN-ENSEM; DISCO; SATIE; IMS Bordeaux, 2009

▶ E. Chanthery, O. Buffet, M. Gaudichaud, M. Contat et I. Jauer. *Rapport sur la modélisation et la formalisation du problème d'optimisation de l'emploi des capteurs.* Rapport de contrat : EADS DOPEC-0050 édition 1.0, 26 mars 2010, 76p. , no 10412, EADS; LORIA; DISCO, 2010

▶ E. Chanthery, M. Gaudichaud, O. Buffet, I. Jauer et T. Oms. *Rapport d'étude théoriques appliquées au problème d'OPEC.* Rapport de contrat : EADS DOPEC-0050 édition 1.0, 26 mars 2010, 76p. , no 10412, EADS; LORIA; DISCO, 2010

▶ M. Godichaud, E. Chanthery, O. Buffet et M. Contat. *Formalisation et résolution de problèmes d'acquisition d'informations par des systèmes autonomes.* In ROADEF (12e congrès annuel de la Société française de Recherche Opérationnelle et d'Aide à la Décision), Saint-Etienne, France, 2011

▶ M. Godichaud, E. Chanthery, O. Buffet et M. Contat. *Formalizing and Solving Information Collection Problems with Autonomous Sensor Systems.* IFAC Proceedings Volumes, vol. 44, no. 1, pages 2208–2213, 2011

▶ C. Jauberthie et E. Chanthery. *Optimal input design for a nonlinear dynamical uncertain aerospace system.* IFAC Proceedings Volumes, vol. 46, no. 23, pages 469–474, 2013

8 Diagnostic distribué

Résumé

Ce chapitre présente mes travaux sur le diagnostic distribué, qui considèrent le système comme un ensemble de sous-systèmes connectés. L'analyse structurelle est une approche pertinente pour la génération de tests dans ce cadre. Elle a l'avantage de la simplicité et de l'efficacité, même sur des systèmes à dynamiques continues non linéaires. Dans ce cadre, nous avons développé des approches de diagnostic décentralisé, puis distribué. Enfin, et pour faire le rapprochement avec mes travaux sur la décision, une partie optimisation des choix de tests, basée entre autres sur une approche originale utilisant des algorithmes de recherche dans des graphes, est proposée.

8.1 Introduction

Ce troisième axe de travail s'inscrit dans une volonté d'optimiser le diagnostic embarqué dans une architecture autonome pour un système complexe. Par "optimiser", on entend par exemple choisir un nombre minimal de tests à effectuer en ligne, ou bien respecter des contraintes embarquées de confidentialité par exemple.

Ce travail a été initié dans le cadre des systèmes continus, dans l'optique d'une extension vers les systèmes hybrides. Cependant, pour le moment, tous les développements ont été effectués pour des systèmes continus, dans la communauté scientifique FDI (Fault Detection and Isolation).

Les récents développements des systèmes technologiques ont mené à une complexification des comportements. Une solution pour gérer cette complexité croissante des systèmes consiste à les considérer comme un ensemble de sous-systèmes hétérogènes et à développer des techniques distribuées pour les contrôler et les gérer. Cette solution soulève plusieurs problèmes. Tout d'abord, au fur et à mesure que la taille et le nombre de composants augmentent, le nombre d'occurrences de pannes qui peuvent conduire le système dans un état de défaillance critique augmente d'autant. De fait, parmi les fonctions opérationnelles, les tâches de détection et d'isolation des fautes (fault detection and isolation ou FDI), de maintenance et de réparation sont devenues prédominantes et elles influent considérablement sur le coût total des produits finaux. Plusieurs verrous peuvent donc être identifiés.

- La **génération de tests dans le cadre de systèmes complexes** doit se faire en tenant compte du fait que le nombre de tests possibles est important. La génération doit être automatique, efficace, et doit pouvoir s'effectuer sur des systèmes dont les modèles sont non-linéaires, ce qui reflète la nature de la majeure partie des systèmes réels.
- Le **développement de méthodes de choix de tests pertinents** constitue un deuxième verrou. Le nombre de tests disponibles étant très important dans un système complexe réel, il faut trouver des moyens pour sélectionner les meilleurs tests possibles. Cela passe par la définition de critères de choix, puis d'algorithmes

d'optimisation pour la sélection de tests.

Deux travaux de thèse (Saurahb Indra et Gustavo Pérèz) ont permis de proposer des solutions pour lever les verrous associés à cet axe. La thèse de Saurahb Indra, entre 2009 et 2012, co-encadrée avec Louise Travé-Massuyès et co-financée par le CNES et Thalès Alénia Space, a proposé l'**analyse structurelle comme solution pour la génération de tests dans le cadre de systèmes complexes**. L'analyse structurelle est basée sur une abstraction du modèle qui ne conserve que les liens entre variables et équations. Malgré son apparente simplicité, l'analyse structurelle fournit un ensemble d'outils puissants, s'appuyant sur la théorie des graphes, pour analyser et inférer des informations sur le système. Par ailleurs, elle a l'avantage de s'appliquer indifféremment sur les systèmes linéaires ou non linéaires. L'isolation à la demande a été proposée et constitue une contribution majeure de la thèse pour les systèmes décentralisés.

La thèse en cotutelle de Gustavo Pérèz, entre 2014 et 2017, co-encadrée avec Louise Travé-Massuyès et Javier Sotomajor de l'Université pontificale catholique du Pérou (PUCP) est allée jusqu'à formuler et résoudre le problème d'optimisation lié au choix d'un sous-ensemble de tests de diagnostic au niveau des sous-systèmes permettant une diagnosticabilité maximale pour le système global dans le cas décentralisé et dans le cas distribué, répondant ainsi au verrou de **développement de méthodes de choix de tests pertinents**.

Personnes impliquées dans cette thématique et affiliation lors de la collaboration : Louise Travé-Massuyès (DR, DISCO), Saurabh Indra (Doctorant, DISCO), Raymond Soumagne (CNES), Xavier Olive (Thalès Alénia Space), Gustavo Pérez (Doctorant, DISCO-PUCP), Javier Sotomajor (PUCP, Pérou).

Enfin, l'encadrement du stage de master d'Asma Gasmi dans le cadre d'une collaboration avec l'équipe ROC du LAAS-CNRS en 2016-2017, a permis de formuler un problème intéressant d'optimisation en nombres entiers pour envisager le cas de choix de tests dans l'éventualité de pannes de capteurs.

Personnes impliquées dans cette thématique et affiliation lors de la collaboration : Louise Travé-Massuyès (DR, DISCO), Christian Artigues (DR, ROC), Nicolas Jozefowiez (MCF, ROC), Asma Gasmi (Stagiaire, DISCO-ROC), Gustavo Pérez (Doctorant, DISCO-PUCP).

8.2 L'analyse structurelle pour la génération de tests dans le cadre de systèmes complexes

8.2.1 Introduction

Le modèle structurel d'un système est une abstraction de son modèle de comportement dans le sens où seule la structure, c'est-à-dire l'existence de liens entre variables et contraintes, est considérée et non les contraintes elles-mêmes.

Un système est composé d'un ensemble de n_e équations impliquant un ensemble de variables partitionnées en un ensemble Z de n_Z variables connues (ou mesurées) et un ensemble X de n_X variables inconnues (ou non mesurées). On note z le vecteur des variables connues et x le vecteur des variables inconnues. Le système peut être affecté par n_F fautes qui apparaissent comme des paramètres dans les équations. L'ensemble des fautes est noté F. Le vecteur des fautes est noté $\boldsymbol{f}$.

Définition 29 (Système) *Un système, noté $\Sigma(z, x, \boldsymbol{f})$ ou plus simplement Σ, est un ensemble d'équations reliant z, x et $\boldsymbol{f}$. Les équations $e_i(z, x) \in \Sigma(z, x, \boldsymbol{f})$, $i = 1, \ldots, n_e$, sont supposées différentielles ou algébriques en z et x.*

Pour illustrer ces concepts, considérons un système (Table 8.1) pour lequel le modèle $\Sigma(z, x, f)$ est composé de six équations notées de e_1 à e_6 reliant les variables connues $Z = \{z_1, z_2\}$, les variables inconnues $X = \{x_1, x_2, x_3, x_4, x_5\}$ et l'ensemble des fautes $F = \{f_1, f_2, f_3\}$. a, b et c sont des paramètres constants.

Relation	Expression
e_1	$\dot{x}_3 = e^{x_3} - c$
e_2	$x_3^2 = b\dot{x}_4 + f_1$
e_3	$z_1 = x_4$
e_4	$z_2 = x_1 + a^2 + x_4$
e_5	$\dot{x}_1 = e^{x_2} + x_5$
e_6	$\dot{x}_3 = x_4 + b + f_2$

TABLE 8.1 – Exemple d'un système algébro-différentiel.

L'approche structurelle va par exemple exprimer le lien entre e_1 et x_3 tout en faisant abstraction de l'équation e_1 elle-même.

L'hypothèse principale est que chaque composant du système est décrit par une ou plusieurs contraintes. La violation d'au moins une contrainte indique que le composant est défectueux. Les propriétés structurelles conduisent à générer des résidus basés sur le principe de redondance analytique [Blanke *et al.* 2006].

L'analyse structurelle est un outil très efficace dans le cadre du diagnostic à base de modèles car les tests obtenus par analyse structurelle peuvent être transformés en relations de redondance analytiques (RRA) et sont conçus hors ligne. Le système de détection de défaut vérifie ensuite en ligne la cohérence des observations par rapport à chacun de ces tests.

Par ailleurs, l'analyse structurelle est une approche intéressante lorsque le système devient complexe. Les travaux de [Kallesoe *et al.* 2006] proposent par exemple de diviser un système complexe non linéaire en plusieurs sous-systèmes. L'objectif est d'identifier les sous-ensembles d'équations du modèle qui incluent une redondance. L'avantage de l'analyse structurelle est que cette approche est applicable à une grande classe de modèles, quelles que soient les valeurs des paramètres, que le système soit linéaire ou non-linéaire. L'approche permet également de réduire la complexité des calculs au moyen d'outils efficaces basés sur la théorie des graphes [Cassar & Staroswiecki 1997]. Développée à l'origine pour la résolution hiérarchique de grands systèmes d'équations, l'analyse structurelle a été adoptée avec succès depuis quelques décennies par la communauté FDI (Fault Detection and Isolation) [Cassar & Staroswiecki 1997], [Patton *et al.* 2000], [Travé-Massuyès *et al.* 2006], [Blanke *et al.* 2006], [Krysander *et al.* 2010]. Étant donné que seules des informations structurelles sont utilisées, cette approche s'applique aux systèmes à grande échelle décrits par un grand nombre de variables, même lorsque leurs modèles analytiques ne sont pas connus avec précision [Düstegör *et al.* 2006, Cassar & Staroswiecki 1997].

Les propriétés structurelles peuvent en outre permettre de déterminer les capacités de surveillance sur le système.

8.2.2 Représentations structurelles

Le modèle structurel peut être représenté sous forme d'un graphe biparti dans lequel les valeurs numériques et les expressions analytiques sont ignorées. Plus précisément, un graphe biparti peut être utilisé pour étudier quelles variables non observées sont impliquées dans les équations modélisant le système. De ce graphe, on peut déduire un chemin possible pour la substitution des variables inconnues. Trouver les redondances dans un modèle

revient alors à résoudre un problème en théorie des graphes utilisant le concept clé de couplage parfait (ou perfect matching). Des méthodes efficaces ont été développées pour les graphes bipartis [Dulmage & Mendelsohn 1958]. Ces méthodes ne présentent pas de problèmes numériques et ont en général une plus faible complexité de calcul que les méthodes d'élimination algébriques [Khorasgani *et al.* 2015].

Le modèle structurel d'un système $\Sigma(z, x, f)$, ou Σ dans la suite du document, peut être obtenu en définissant le graphe biparti $G(\Sigma \cup X \cup Z, A)$, ou de manière équivalente $G(\Sigma \cup X, \mathcal{A})$ avec $\mathcal{A} \subseteq A$ et $\mathcal{A}$ l'ensemble d'arcs tels que $a(x_j, e_i) \in \mathcal{A}$ si et seulement si la variable x_j est impliquée dans l'équation e_i. Les sommets du graphe biparti sont partitionnés en deux ensembles $\Sigma = \{e_1, e_2, ..., e_m\}$ et $X = \{x_1, x_2, ..., x_n\}$.

Le graphe biparti associé à l'exemple est représenté sur la figure 8.1.

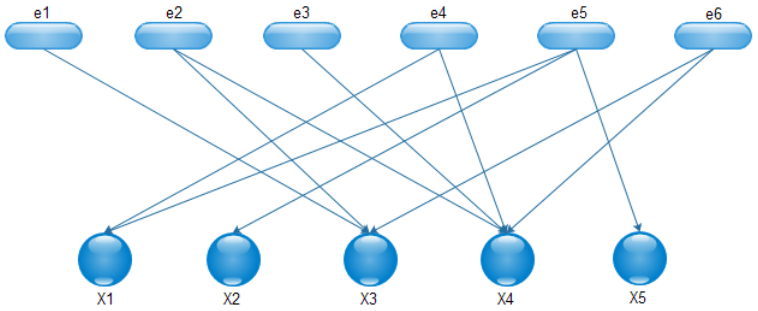


FIGURE 8.1 – Graphe biparti pour l'exemple.

Le graphe peut aussi être représenté par sa matrice d'adjacence. La matrice d'adjacence $\mathcal{A}^d$ pour un graphe biparti G est une matrice $n_e \times n_X$ [1] définie par :

$$\mathcal{A}^d_{i,j} = \begin{cases} 1 & \text{si } e_i \text{ et } x_j \text{ sont adjacents} \\ 0 & \text{sinon.} \end{cases} \tag{8.1}$$

Il y a donc un 1 dans la matrice d'adjacence si x_j ou l'une de ses dérivées temporelles apparaît dans l'équation e_i. Plus généralement, un 1 indique qu'une variable apparaît dans l'équation e_i.

La matrice d'adjacence de l'exemple est la suivante.

équations	variables inconnues					variables connues		fautes	
	x_1	x_2	x_3	x_4	x_5	z_1	z_2	f_1	f_2
e_1			1						
e_2			1	1				1	
e_3				1		1			
e_4	1			1			1		
e_5	1	1			1				
e_6			1	1					1

TABLE 8.2 – Représentation structurelle de l'exemple.

DÉFINITION 30 (COUPLAGE) *Un couplage entre X et Σ est un sous-ensemble de $\mathcal{A}$ tel qu'aucun sommet dans $X \cup \Sigma$ ne soit incident avec plus d'un arc du couplage.*

1. ou bien $n_e \times (n_X + n_Z + n_F)$ si on inclut dans les variables d'intérêt les variables connues et les fautes. Dans le cas le plus simple, on ne considère que les variables inconnues.

Définition 31 (Couplage parfait (ou couplage complet)) *Un couplage parfait $\mathcal{M}$ sur les variables de X, aussi appelé couplage complet sur X, est un couplage tel que toutes les variables de X sont couplées. Le couplage est dit parfait sur les contraintes lorsque toutes les contraintes sont couplées.*

Dans le cadre du diagnostic, on s'intéresse bien évidemment à des couplages parfaits sur les variables de X. Un tel couplage sera noté $\mathcal{M}(X, \Sigma)$ ou $\mathcal{M}$ lorsqu'il n'y a pas d'ambiguïté. $\mathcal{M}(X, \Sigma)$ permet d'identifier les chemins pour calculer les variables inconnues à partir des variables mesurées ou partagées.

8.2.3 Relations de redondance analytiques et analyse structurelle

RRA et MSO

Le concept principal de la génération de résidus dans le cas de systèmes continus est la redondance analytique. Les relations de redondance analytiques (RRA) sont des équations qui sont déduites du modèle analytique d'un système et qui ne font intervenir que des variables mesurées. Ce sont des contraintes statiques ou dynamiques qui capturent le comportement temporel des variables connues lorsque le système fonctionne en conditions nominales. La procédure de détection des fautes va vérifier si la relation est satisfaite ou pas. Dans le cas où la relation n'est pas satisfaite, la procédure d'isolation de fautes identifie les composants du système qui doivent être suspectés. Ces relations constituent des tests pour le diagnostic du système.

Définition 32 (RRA pour $\Sigma(z, x, f)$) *Soit $\Sigma(z, x, f)$ un système. Une relation $rra(z, \dot{z}, \ddot{z}, ...) = 0$ est une relation de redondance analytique pour $\Sigma(z, x, f)$ si pour tout z cohérent avec $\Sigma(z, x, f)$ la relation est satisfaite.*

Définition 33 (Générateur de résidu pour $\Sigma(z, x, f)$) *Un système ayant pour entrées un sous-ensemble de variables z et ayant pour sortie un signal scalaire arr est un générateur de résidu pour le modèle $\Sigma(z, x, f)$ si, pour tout z cohérent avec $\Sigma(z, x, f)$, la relation $\lim_{t\to\infty} arr(t) = 0$ est vérifiée.*

Les RRA peuvent donc être utilisées pour vérifier si les variables mesurées z sont cohérentes avec le modèle du système et comme base des générateurs de résidus pour des fins de diagnostic.

Il a été montré [Blanke *et al.* 2006] que les RRA peuvent être dérivées de couplages parfaits entre X et Σ sur le graphe biparti $G(\Sigma \cup X, \mathcal{A})$.

Par ailleurs, de nombreuses méthodes pour calculer des ensembles avec une redondance structurelle à partir de $G(\Sigma \cup X, \mathcal{A})$ sont basées sur la décomposition canonique de Dulmage-Mendelsohn (DM) [Murota 2000], [Dulmage & Mendelsohn 1958]. La décomposition de Dulmage-Mendelsohn est une partition des sommets d'un graphe biparti en sous-ensembles. Deux sommets adjacents appartiennent au même sous-ensemble si et seulement si ils sont liés par un couplage parfait.

La figure 8.2 montre la décomposition canonique de Dulmage-Mendelsohn d'un graphe biparti $G(\Sigma \cup X, \mathcal{A})$. On distingue 3 parties.

- La partie **structurellement surdéterminée** (structurally overdetermined ou SO) Σ^+ qui a plus d'équations que de variables inconnues. Le surplus d'équations est redondant ;
- La partie **structurellement juste déterminée** Σ^0 qui a un nombre fini de solutions. Elle peut être résolue indépendamment des variables de Σ^-, une fois que Σ^+ est résolue.

- La partie **structurellement sous-déterminée** Σ^- qui a plus de variables inconnues que d'équations. Ce système ne peut être résolu, même si les variables de Σ^+ et Σ^0 ont été calculées.

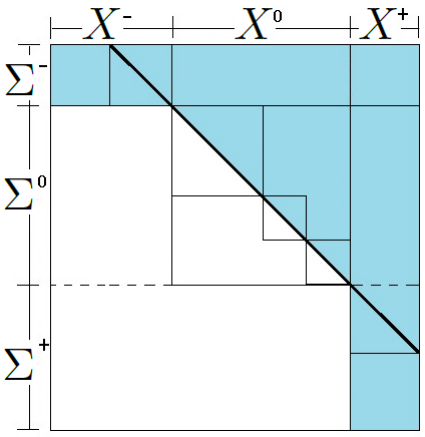


FIGURE 8.2 – Décomposition de Dulmage-Mendelsohn de Σ.

On définit plus formellement les ensembles SO (Structurally Overdetermined sets) et par suite la notion de redondance structurelle.

DÉFINITION 34 (ENSEMBLE SO (STRUCTURALLY OVERDETERMINED SET)) *Un ensemble d'équations $\Sigma' \subseteq \Sigma$ est un ensemble SO si Σ' a plus d'équations que de variables.*

DÉFINITION 35 (REDONDANCE STRUCTURELLE) *La redondance structurelle $\rho_{\Sigma'}$ d'un ensemble d'équations $\Sigma' \subseteq \Sigma$ est la différence entre le nombre d'équations dans Σ' et le nombre de variables inconnues impliquées dans les équations :*

$$\rho_{\Sigma'} = \mid \Sigma' \mid - \mid X_{\Sigma'} \mid \tag{8.2}$$

où $X_{\Sigma'}$ est l'ensemble des variables inconnues impliquées dans Σ' et $\mid \Sigma' \mid$ est le cardinal de Σ'.

On peut ainsi définir les ensembles MSO (Minimal Structurally Overdetermined set ou MSO set).

DÉFINITION 36 (ENSEMBLE MSO (MINIMAL STRUCTURALLY OVERDETERMINED SET)) *Un ensemble MSO est un ensemble SO avec une redondance structurelle de 1.*

☞ On parlera par la suite de "MSO" à la place d'"ensemble MSO" pour plus de fluidité.

À l'aide de l'analyse structurelle, il est possible de déterminer la partie du système sur laquelle des RRA peuvent être générées [Cocquempot *et al.* 1998]. En effet, il a été montré que les RRA sont l'interprétation causale des MSO [Krysander *et al.* 2010]. On notera que les résultats obtenus dans un cadre structurel sont les cas les plus optimistes : les considérations de causalité, les boucles algébriques et différentielles, ... définissent au final quelles redondances structurelles peuvent être utilisées pour la conception de "vrais" générateurs de résidus [Armengol *et al.* 2009].

On pourra se référer au manuscrit de thèse de Gustavo Pérèz [Pérez Zuniga 2017] pour une discussion sur les algorithmes de génération de résidus à partir des MSO. On retiendra que la recherche des MSO est de complexité exponentielle en temps de calcul. Pour de grands systèmes, il est donc crucial de trouver un moyen de ne pas calculer tous les MSO et de réduire la recherche à un nombre plus restreint de MSO, notamment ceux qui sont pertinents pour le diagnostic de fautes.

Recherche des MSO guidée par les fautes

⇨ **Support de test et support minimal d'équations de test**

Comme on l'a indiqué précédemment, une idée est de concentrer la recherche autour des MSO pertinents.

[Krysander *et al.* 2010] a proposé de rechercher un plus petit ensemble de modèles testables appelés supports d'équations de test (Test Equation Supports ou TES) qui est un ensemble d'équations exprimant la redondance spécifique à un ensemble de fautes considérées. Cet ensemble de fautes est appelé support de test (TS), ou support de fautes. On définit également un support de test minimal (MTS) et un support minimal d'équations de tests (MTES) lorsqu'aucun de leurs sous-ensembles n'est un TS et un TES, respectivement.

⇨ **Ensemble MSO guidé par les fautes**

Alors qu'un MSO est juste surdéterminé et présente donc une redondance structurelle de 1, un MTES peut avoir une redondance structurelle plus élevée. Cela peut être un avantage pour développer les tests les plus puissants possible. Cependant, si l'on envisage la décentralisation ou la distribution du processus de diagnostic, le but peut être de minimiser les informations partagées par les sous-systèmes. On a donc introduit dans la thèse de Gustavo Pérèz [Pérez *et al.* 2015] le concept d'ensemble MSO guidé par les fautes (Fault-Driven Minimal Structurally Overdetermined set ou FMSO set). On parlera par la suite d'ensemble FMSO ou plus simplement de FMSO.

Soit $Z_\varphi \subseteq Z$, $X_\varphi \subseteq X$, les ensembles des variables connues et inconnues dans un ensemble FMSO φ, et $F_\varphi \subseteq F$ l'ensemble des fautes à diagnostiquer.

Définition 37 (Ensemble FMSO) *Un sous-ensemble d'équations $\varphi \subseteq \Sigma(z, x, \boldsymbol{f})$ est un ensemble FMSO de $\Sigma(z, x, \boldsymbol{f})$ si*

1. *$F_\varphi \neq \emptyset$ et $\rho_\varphi = 1$ ce qui signifie $|\varphi| = |X_\varphi| + 1$,*
2. *aucun sous-ensemble de φ n'est surdéterminé.*

On définit ensuite les concepts de faute détectable et de faute isolable, en utilisant le concept de FMSO.

Définition 38 (Faute détectable) *Une faute $f_i \in F$ est détectable dans le système $\Sigma(z, x, \boldsymbol{f})$ s'il existe un ensemble FMSO $\varphi \in \Phi$ tel que $f_i \in F_\varphi$.*

Le concept d'isolation est basé sur la détermination de l'ensemble des fautes qui peuvent être isolées d'une faute donnée.

Définition 39 (Faute isolable) *Soit deux fautes détectables f_j et f_k de F, $j \neq k$, f_j est isolable de f_k s'il existe un FMSO $\varphi \in \Phi$ tel que $f_j \in F_\varphi$ et $f_k \notin F_\varphi$.*

On définit également le concept d'ensemble CMSO (Clear Minimal Structurally Overdetermined set) comme un MSO de $\Sigma(z, x, \boldsymbol{f})$ dont le support de fautes est vide. On simplifiera également l'appellation en CMSO pour plus de fluidité s'il n'y a pas d'ambiguïté.

Définition 40 (Ensemble CMSO) *Un sous-ensemble d'équations $\psi \subseteq \Sigma(z, x, \boldsymbol{f})$ est un ensemble CMSO de $\Sigma(z, x, \boldsymbol{f})$ si*

1. *$F_\psi = \emptyset$ et $\rho_\psi = 1$ ce qui signifie $|\psi| = |X_\psi| + 1$,*
2. *aucun sous-ensemble de ψ n'est surdéterminé.*

8.3 Concepts et propriétés dans le cadre du diagnostic distribué/décentralisé

8.3.1 Concepts généraux

Comme on l'a souligné dans la section 5.3.3, page 34, il existe principalement 2 manières de gérer la complexité d'un système. On distingue les architectures de diagnostic décentralisées et distribuées. Cette section vise à décrire les notions communes à ces deux approches et à rappeler les propriétés qui nous ont permis de développer nos approches de diagnostic décentralisées et distribuées.

Dans la suite, le "niveau global" fait référence à l'approche de diagnostic centralisée. Sans perte de généralité, on considère deux niveaux hiérarchiques pour le diagnostic, le niveau dit "local" et le niveau "hiérarchique".

Définition 41 (Ensemble FMSO global) *Un ensemble FMSO global est un ensemble FMSO de $\Sigma(z, x, f)$. L'ensemble des ensembles FMSO globaux est noté Φ.*

On parlera par la suite de FMSO global pour plus de fluidité.

Dans le cas décentralisé, la décomposition du modèle du système Σ en plusieurs sous-systèmes Σ_i se définit comme une organisation hiérarchique de ses équations sur plusieurs niveaux comme le montre la figure 8.3.

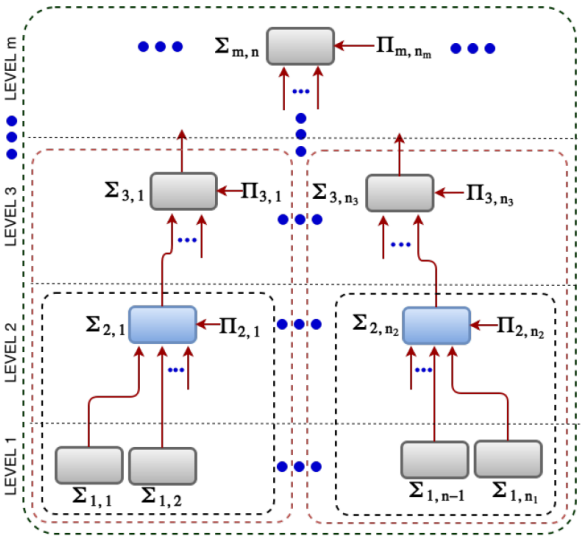


Figure 8.3 – Exemple d'une architecture décentralisée.

Les équations contenues dans l'ensemble $\Pi_{i,j}$ sont des équations qui ne sont disponibles qu'à partir du niveau i, à cause de contraintes spécifiques, par exemple des contraintes de confidentialité, de distance ou d'accès difficile, et donc pas disponible au niveau $i-1$ [Pérez *et al.* 2015]. Chaque rectangle de niveau 1 correspond à un sous-système. Chaque rectangle en pointillé correspond ensuite à un sous-système de niveau supérieur. On remarque que les sous-systèmes de niveau supérieur incluent des sous-systèmes de niveau inférieur.

Dans le cas distribué, une décomposition du système Σ en plusieurs sous-systèmes Σ_i se définit comme une partition de ses équations. Soit $\Sigma = \{\Sigma_1, \Sigma_2, ..., \Sigma_n\}$ avec $\Sigma_i \subseteq \Sigma$, $\bigcup_{i=1}^{n} \Sigma_i = \Sigma$, $\Sigma_i \neq \varnothing$ et $\Sigma_i \cap \Sigma_j = \varnothing$ si $i \neq j$.

Sans perte de généralité, ces deux décompositions vont mener à n sous-systèmes que l'on note $\Sigma_i(z_i, x_i, \boldsymbol{f}_i)$, avec $i = 1, ..., n$, où z_i est le vecteur des variables connues pour Σ_i, x_i est le vecteur des variables inconnues pour Σ_i et $\boldsymbol{f}_i$ est le vecteurs des fautes pour Σ_i. Les ensembles des variables inconnues, connues et des fautes pour le sous-système Σ_i, sont respectivement notés X_i, Z_i, et F_i. Ces ensembles sont les sous-ensembles de X, Z, et F respectivement, qui sont impliqués dans Σ_i.

Définition 42 (Variables locales) *L'ensemble des variables locales du $i^{ème}$ sous-système, noté X_i^l, est défini comme le sous-ensemble des variables de X_i qui sont impliquées uniquement dans le sous-système Σ_i :*

$$X_i^l = X_i \backslash (\bigcup_{j=1, j\neq i}^{n} (X_i \cap X_j)). \tag{8.3}$$

Définition 43 (Variables partagées) *L'ensemble des variables partagées du $i^{ème}$ sous-système, noté X_i^s, est défini par :*

$$X_i^s = \bigcup_{j=1, j\neq i}^{n} (X_i \cap X_j) = X_i \backslash X_i^l \tag{8.4}$$

L'ensemble des variables partagées du système entier Σ est noté X^s.

Sans perte de généralité, on considère que toutes les variables connues de Z_i sont locales au sous-système Σ_i, pour $i = 1, \ldots, n$. Si la même entrée était appliquée à plusieurs sous-systèmes, elle pourrait être reproduite artificiellement.

8.3.2 Ensembles FMSO dans le cadre distribué/décentralisé

⇨ Concepts communs
Cette section étend la définition de FMSO à un contexte distribué/décentralisé.

Définition 44 (Ensemble FMSO local) *φ est un ensemble FMSO local $\Sigma_i(z_i, x_i, \boldsymbol{f}_i)$ si φ est un ensemble FMSO de $\Sigma(z, x, \boldsymbol{f})$ et si $\varphi \subseteq \Sigma_i$, $X_\varphi \subseteq X_i$ et $Z_\varphi \subseteq Z_i^l$. On simplifiera l'appellation en FMSO local pour plus de fluidité. L'ensemble des FMSO locaux de Σ_i est noté Φ_i^l. L'ensemble des FMSO locaux du système est noté $\Phi^l = \bigcup_{i=1}^{n} \Phi_i^l$.*

On définit également les ensembles FMSO partagés (ou plus simplement FMSO partagés) pour un sous-système Σ_i en considérant les variables partagées comme des variables connues et en calculant les FMSO. Les FMSO comprenant des équations avec des variables partagées sont appelés des ensemble FMSO partagés.

Définition 45 (Ensemble FMSO partagé) *φ est un ensemble FMSO partagé du sous-système $\Sigma_i(z_i, x_i, \boldsymbol{f}_i)$ si φ est un ensemble FMSO de $\tilde{\Sigma}_i(\tilde{z}_i, \tilde{x}_i, \tilde{\boldsymbol{f}}_i)$, où $\tilde{z}_i$ est le vecteur des variables dans $\tilde{Z}_i = Z_i \cup X_i^s$, $\tilde{x}_i$ est le vecteur des variables dans $\tilde{X}_i = X_i^l$, et $\tilde{\boldsymbol{f}}_i = \boldsymbol{f}_i$. L'ensemble des FMSO partagés pour Σ_i est noté Φ_i^s. L'ensemble de tous les FMSO partagés du système est noté $\Phi^s = \bigcup_{i=1}^{n} \Phi_i^s$.*

Les définitions 44 et 45 peuvent également être appliquées aux ensembles CMSO pour définir des ensembles CMSO locaux Ψ_i^l et des ensembles CMSO partagés Ψ_i^s (ou plus simplement des CMSO locaux et des CMSO partagés). L'ensemble de tous les ensembles CMSO partagés du système est noté Ψ^s.

Les notions de fautes isolables et détectables sont étendues dans le cas distribué/décentralisé.

Définition 46 (Faute localement détectable) *La faute $f \in F_i$ est localement détectable dans le sous-système $\Sigma_i(z_i, x_i, \boldsymbol{f}_i)$ s'il existe un FMSO $\varphi \in \Phi_i^l$ tel que $f \in F_\varphi$.*

Définition 47 (Faute localement isolable) *Pour deux fautes localement détectables f_j et f_k de F_i, $j \neq k$, f_j est localement isolable de f_k s'il existe un FMSO local $\varphi \in \Phi_i^l$ tel que $f_j \in F_\varphi$ et $f_k \notin F_\varphi$.*

⇨ Ensembles FMSO dans le cadre distribué

Dans un contexte distribué, on rajoute le concept d'ensemble FMSO complété (ou plus simplement FMSO complété).

Définition 48 (Ensemble FMSO complété) *Un FMSO global φ incluant au moins un FMSO partagé $\varphi' \in \Phi_i^s$ est appelé un ensemble FMSO complété. L'ensemble des FMSO complétés de Σ_i est noté Φ_i^c. L'ensemble de tous les FMSO complétés est noté $\Phi^c = \bigcup_{i=1}^{n} \Phi_i^c$.*

Définition 49 (Ensemble FMSO racine, générateur) *Tout FMSO partagé $\varphi' \in \Phi^s$ inclus dans un FMSO complété $\varphi \in \Phi^c$ est appelé un ensemble FMSO racine de φ (ou plus simplement FMSO racine). On dit de l'ensemble des FMSO racine de φ qu'il est un générateur de φ.*

8.3.3 Equivalences pour la recherche de RRA en centralisé et en décentralisé/distribué

Nos travaux nous ont menés à prouver un certain nombre de propriétés d'équivalence pour le diagnostic centralisé et le diagnostic décentralisé/distribué. L'idée est de les rappeler ici, les détails étant disponibles dans nos publications.

⇨ Equivalence entre la recherche des RRA au niveau local et au niveau global

Nous avons tout d'abord montré, dans la thèse de Saurabh Indra que les ensembles des RRA dérivés d'une architecture globale ou d'une architecture décentralisée étaient identiques. La preuve a été publiée dans [Chanthery *et al.* 2016b]. Dans ces travaux, on appelle relations hiérarchiques des relations dont les variables ne peuvent pas être substituées localement et qui nécessitent des substitutions via des relations d'autres sous-systèmes.

La proposition s'énonce ainsi.

Proposition 1 *Supposons qu'un système Σ est décomposé en sous-systèmes $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$ suivant une architecture décentralisée. L'ensemble des RRA qui peuvent être dérivées pour Σ avec une architecture centralisée est identique à l'ensemble des RRA qui peuvent être dérivées avec une architecture décentralisée en cherchant les RRA pour chaque sous-système Σ_i, $i = 1, \ldots, n$ et en considérant les relations hiérarchiques.*

⇨ Equivalence pour les MTES

A la suite de ces travaux, nous avons focalisé la recherche des tests sur des ensembles de fautes pertinents, en utilisant le concept de support minimal d'équations de test (MTES).

La proposition s'énonce comme suit.

PROPOSITION 2 *L'ensemble des MTES calculés de manière centralisée et l'ensemble des MTES calculés de manière décentralisée mènent au même ensemble de RRA.*

Cette proposition a été démontrée dans [Chanthery *et al.* 2016b].

⇨ **Equivalence FMSO locaux, FMSO globaux**

Le travail de Gustavo Pérèz a pris la suite du travail de Saurabh Indra. Dans ce cadre [Pérez *et al.* 2017], nous avons notamment montré les propriétés suivantes :

- un FMSO complété contient des équations d'au moins deux sous-systèmes ;
- un FMSO local d'un sous-système Σ_i est aussi un FMSO de Σ, c'est donc un FMSO global ;
- un FMSO global $\varphi \in \Phi$ pour lequel $\exists! i \in 1, \ldots, n$ tel que $X_\varphi \subseteq X_i^l$ est aussi un FMSO local de Σ_i.

Nous avons ainsi démontré la propriété d'équivalence entre l'approche centralisée et l'approche décentralisée/distribuée pour les FMSO. La proposition s'énonce ainsi.

PROPOSITION 3 *L'ensemble des FMSO globaux peut être obtenu à partir des FMSO calculés localement, en formant des FMSO complétés avec des FMSO partagés et des CMSO partagés.*

Cette proposition a été démontrée dans [Pérez *et al.* 2017].

L'ensemble de ces propositions nous a permis de proposer des algorithmes pour mettre en œuvre les différentes architectures décentralisées/distribuées pour le diagnostic.

On a également montré la propriété suivante.

PROPRIÉTÉ 3 *Soit* $\{\Phi_j\}$, $j = 1, \ldots, n_{\Phi^s}$ *un sous-ensemble des FMSO partagés.* $\{\Phi_j\}$ *est un générateur d'un FMSO complété si et seulement si :*

- $card(\Phi_j) = ns + 1$ *où* ns *est le nombre de variables partagées apparaissant dans les FMSO de* $\{\Phi_j\}$ *;*
- *il existe un couplage parfait sur les variables partagées impliquées dans les FMSO de* $\{\Phi_j\}$*. Toutes les variables partagées seront couplées avec les FMSO inclus dans* $\{\Phi_j\}$*.*

8.4 Approche décentralisée pour le diagnostic

Les architectures de diagnostic décentralisées sont composées de diagnostiqueurs locaux qui travaillent avec des modèles locaux de leurs sous-systèmes. Une ambiguïté de diagnostic au niveau local sera résolue par un diagnostiqueur de niveau supérieur.

8.4.1 Utilisation des MTES

Une première proposition d'une architecture décentralisée pour le diagnostic a été proposée dans les travaux de Saurabh Indra, illustrée sur la figure 8.4.

Les diagnostiqueurs locaux sont conçus hors ligne. La conception est effectuée sur chaque sous-système $\Sigma_{i,j}$ $j = 1, \ldots, n_i$ à chaque niveau $i = 1, \ldots, m$ grâce à une boucle imbriquée. i est le niveau dans la hiérarchie, et j l'énumération des sous-systèmes à ce niveau. On appelle MTES hiérarchiques des MTES qui sont calculés grâce à des combinaisons de relations provenant de plusieurs sous-systèmes dans la hiérarchie. Ces MTES vont générer ce qu'on appelle des résidus hiérarchiques. La procédure, illustrée sur la figure 8.4, est la suivante.

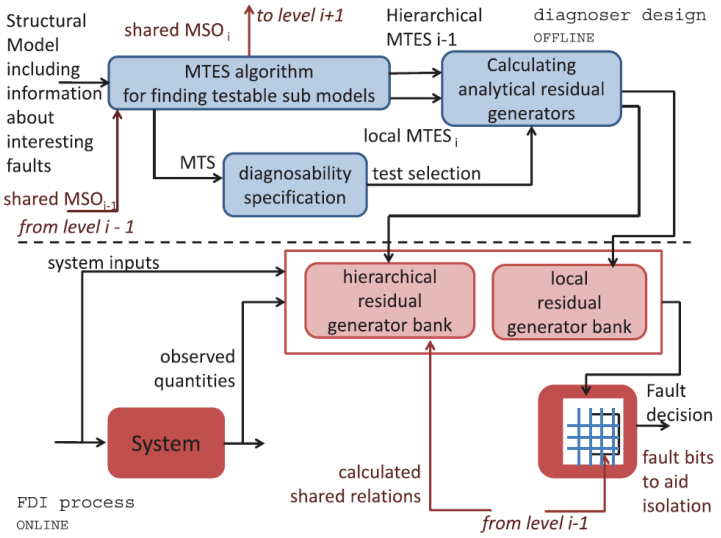


FIGURE 8.4 – Schéma de calculs pour une architecture de diagnostic décentralisée à un niveau i.

1. Utilisation de l'algorithme de génération des MTES avec le modèle structurel du sous-système $\Sigma_{i,j}$ comme entrée au niveau i, en considérant les variables partagées connues (boîte "*MTES algorithm for finding testable submodels*")
   Sortie : MTES locaux pour le sous-système $\Sigma_{i,j}$ au niveau i ; MTS pour le sous-système $\Sigma_{i,j}$ au niveau i ; MSO partagés pour le sous-système $\Sigma_{i,j}$ au niveau i à envoyer au niveau $i+1$ (flèche "*to level i+1*").
2. Utilisation de l'algorithme de génération des MTES avec les MSO partagés venant des diagnostiqueurs locaux reliés du niveau $i-1$
   Sortie : MTES hiérarchiques des sous-systèmes de niveau $i-1$.
3. Utilisation des MTS et des spécifications de diagnosticabilité pour choisir les générateurs de résidus à implémenter (boîte "*diagnosability specification*")
4. Calcul des générateurs de résidus à partir des MTES locaux
   Sortie : générateurs de résidus pour le sous-système $\Sigma_{i,j}$ stockés dans la banque locale de générateurs de résidus.
5. Calcul des générateurs de résidus à partir des MTES hiérarchiques
   Sorties : générateurs de résidus hiérarchiques pour les diagnostiqueurs locaux reliés du niveau $i-1$, stockés dans la banque de générateurs de résidus hiérarchiques.

8.4.2 Utilisation des FMSO

⇨ Procédure finale

La procédure précédente a été améliorée dans les travaux de Gustavo Pérèz. Au lieu d'utiliser les MTES, nous avons établi une procédure utilisant les FMSO, ce qui permet d'implémenter un nombre moins important de tests.

La conception du diagnostiqueur est effectuée hors-ligne. Les étapes de la procédure sont effectuées pour chaque sous-système $\Sigma_{i,j}$ $j = 1, \ldots, n_i$ à chaque niveau $i = 1, \ldots, m$, grâce à une boucle imbriquée. i est le niveau de la hiérarchie, et j l'énumération des sous-systèmes à chaque niveau.

Si les fautes ne sont pas détectables ou isolables à un certain niveau, un diagnostiqueur de plus haut niveau hiérarchique est développé jusqu'à ce que l'objectif de diagnosticabilité

Algorithme 3 : Conception des diagnostiqueurs décentralisés

$n_0 = 1$, $E_{0,1} = \emptyset$;
`/* Pour chaque niveau de 1 à m */`
pour tous les *i de 1 à m* **faire**
  $\Omega_{i,j} \leftarrow$ Rassembler tous les sous-systèmes de niveau $i-1$ qui sont liés au niveau i par le sous-système j;
  `/* Pour chaque sous-système de 1 à` n_i `*/`
  pour tous les *j de 1 à* n_i **faire**
    $\Pi_{i,j} \leftarrow$ Charger les équations supplémentaires du système $\Sigma_{i,j}$;
    $\Sigma_{i,j} = \Pi_{i,j} \cup (\bigcup_{j \in \Upsilon_{i,j}} E_{i-1,j})$;
    $\Phi^l_{i,j} \leftarrow$ Calculer les FMSO locaux de $\Sigma_{i,j}$;
    `/* En cas d'optimisation */`
    Exécuter une sélection optimale des FMSO locaux via la résolution d'un problème linéaire en nombres entiers;
    `/* Dans tous les cas */`
    Calculer les RRA de $\Sigma_{i,j}$ à partir des FMSO locaux sélectionnés;
    $RRA_{i,j} \leftarrow$ Calculer les générateurs de résidus analytiques de $\Phi^h_{i,j}$;
    si *il y a une faute* $f \in F_{i,j}$ *qui n'est pas localement détectable ou isolable avec l'ensemble des FMSO locaux* $\Phi^l_{i,j}$ **alors**
      $\Phi^s_{i,j} \leftarrow$ Calculer les FMSO partagés de $\Sigma_{i,j}$;
      $\Psi^s_{i,j} \leftarrow$ Calculer les ensembles CMSO partagés de $\Sigma_{i,j}$;
      $E_{i,j} = \{e \in \Sigma / \exists S_{i,j} \in \Phi^s_{i,j} \cup \Psi^s_{i,j} \wedge e \in S_{i,j}\}$;

soit atteint. La hiérarchie est contrainte par la communication inter-niveau, que l'on définit formellement comme un ensemble de graphes bipartis.

Définition 50 (Communication inter-niveau) *La communication inter-niveau est représentée par un ensemble de* $m-1$ *graphes bipartis* $S^i_{i-1}(\mathbb{N}^i_{i-1}, \mathbb{L}^i_{i-1})$, $i = 2, \ldots, m$. $S^i_{i-1}(\mathbb{N}^i_{i-1}, \mathbb{L}^i_{i-1})$ *est un graphe biparti tel que* $\mathbb{N}^i_{i-1} = \mathbb{N}_{i-1} \cup \mathbb{N}_i$, *où :*

- $\mathbb{N}_{i-1} = \{n_{i-1,j}, j = 1, \ldots, n_{i-1}\}$ *est un ensemble de nœuds correspondant aux sous-systèmes* $\Sigma_{i-1,j}, j = 1, \ldots, n_{i-1}$, *du niveau* $i-1$,
- $\mathbb{N}_i = \{n_{i,j}, j = 1, \ldots, n_i\}$ *est un ensemble de nœuds correspondant aux sous-systèmes* $\Sigma_{i,j}, j = 1, \ldots, n_i$,*du niveau* i,
- $\mathbb{L}^i_{i-1} = \{l_{\nu,\xi}, \nu = 1, \ldots, n_{i-1}, \xi = 1, \ldots, n_i\}$ *est ensemble d'arcs tels que l'arc* $l_{\nu,\xi}$ *entre le nœud* $n_\nu \in \mathbb{N}_{i-1}$ *et le nœud* $n_\xi \in \mathbb{N}_i$ *existe si la communication est possible entre le sous-système* $\Sigma_{i-1,\nu}$ *du niveau* $i-1$ *et le sous-système* $\Sigma_{i,\xi}$ *du niveau* i.

Dans l'algorithme 3, la communication inter-niveau est prise en compte par $\Upsilon_{i,j}$, où $\Upsilon_{i,j} = \{\Sigma_{i-1,\nu} / l_{\nu,j}$ existe dans $S^i_{i-1}(\mathbb{N}^i_{i-1}, \mathbb{L}^i_{i-1})\}$, pour $i = 2, \ldots, m$, et $\Upsilon_{1,j} = \emptyset$. Autrement dit, pour les niveaux de $i = 2, \ldots, m$, $\Upsilon_{i,j}$ contient tous les sous-systèmes de niveau $i-1$ qui ont des connexions avec le sous-système $\Sigma_{i,j}$.

L'algorithme calcule donc récursivement, en développant les niveaux nécessaires, tous les générateurs de résidus analytiques qui garantissent l'isolation de toutes les fautes isolables.

⇨ **Optimisation locale du choix des FMSO locaux**

Dans l'ensemble des FMSO locaux calculés pour chaque sous-système, pour sélectionner efficacement les FMSO locaux, nous avons utilisé une approche par optimisation d'un

problème linéaire en nombres entiers, dont le critère vise à garantir un maximum de diagnosticabilité pour le système, tout en minimisant le nombre de tests utilisés pour le sous-système $\Sigma_{i,j}$.

Soit $\mathbb{z}_{\varphi_i}$ une variable booléenne qui vaut 1 si le FMSO φ_i est sélectionné et 0 sinon. On note $\mathbb{f}_{\varphi_i kl}$ une variable booléenne qui indique si la faute f_k est isolable de la faute f_l en utilisant le FMSO φ_i et e_{kl} une variable réelle.

L'objectif est de maximiser l'isolabilité entre toutes les paires de fautes de l'ensemble $F_{i,j}$ du sous-système $\Sigma_{i,j}$, tout en minimisant le cardinal de l'ensemble des FMSO locaux sélectionnés $\Phi^l_{i,j}$. On note m une pondération entre la maximisation sur l'isolabilité et la minimisation du nombre de FMSO locaux. Le problème peut se traduire ainsi :

$$max(\, m\sum_{(f_k,f_l)\in F_{i,j}} e_{kl} - (m-1)\sum_{\varphi_i\in\Phi^l_{i,j}} \mathbb{z}_{\varphi_i}) \tag{8.5}$$

sous les contraintes :

$$\sum_{\varphi_i\in\Phi^l_{i,j}} \mathbb{f}_{\varphi_i kl}\mathbb{z}_{\varphi_i} \geq e_{kl} \tag{8.6}$$

$$\mathbb{z}_{\varphi_i} \in \{0,1\} \text{ pour } \varphi_i \in \Phi^l_{i,j}, \quad (f_k,f_l)\in F_{i,j}, \text{ et} \quad m\in[0,1]. \tag{8.7}$$

L'équation 8.5 est la fonction objectif. La contrainte 8.6 est utilisée pour exprimer l'isolabilité entre les fautes f_k et f_l.

Après la conception du diagnostiqueur décentralisé hors ligne à l'aide de l'algorithme 3, l'implémentation en ligne du diagnostiqueur est basée sur la notion de matrice locale de signatures de fautes (ou FSM) des sous-systèmes.

Définition 51 (Matrice locale des signatures de fautes) *Soit un ensemble $\mathcal{R}_{i,j}$ composé de $n^r_{i,j}$ RRA et $F_{i,j}$ l'ensemble des $n^f_{i,j}$ fautes considérées pour le sous-système $\Sigma_{i,j}$. La signature d'une faute $f\in F_{i,j}$ est un vecteur binaire $FS_{i,j}(f) = [\tau_1,\tau_2,...\tau_{n^r_{i,j}}]^T$ où τ_k, $k=1...n^r_{i,j}$, est calculé à partir de $\mathcal{R}_{i,j}\times F_{i,j} \longrightarrow \{0,1\}$ tel que $\tau_k = 1$, si l'équation $rra_k \in \mathcal{R}_{i,j}$ est affectée par la faute f, $\tau_k = 0$ sinon. Les signatures de l'ensemble des fautes de $F_{i,j}$ constituent la matrice locale des signatures de fautes, notée $\mathcal{S}^l_{i,j}$ pour le sous-système $\Sigma_{i,j}$, i.e. $\mathcal{S}^l_{i,j} = [FS_{i,j}(f_1),\ldots,FS_{i,j}(f_{n^f_{i,j}})]$.*

Cette définition a évidemment son pendant pour le système global. On rappelle ici sa définition.

Définition 52 (Matrice globale des signatures de fautes) *Soit un ensemble $\mathcal{R}$ composé de n^r RRA et F l'ensemble des n^f fautes considérées pour le système Σ. La signature d'une faute $f\in F$ est un vecteur binaire $FS(f) = [\tau_1,\tau_2,...\tau_{n^r}]^T$ où τ_k, $k=1...n^r$, est calculé à partir de $\mathcal{R}\times F \longrightarrow \{0,1\}$ tel que $\tau_k = 1$, si l'équation $rra_k\in\mathcal{R}$ est affectée par la faute f, $\tau_k = 0$ sinon. Les signatures de l'ensemble des fautes de F constituent la matrice globale des signatures de fautes, notée $\mathcal{S}$, i.e. $\mathcal{S} = [FS(f_1),\ldots,FS(f_{n^f})]$.*

⇨ Isolation à la demande

Le diagnostiqueur est implémenté en ligne comme une banque de générateurs de résidus hiérarchiques basée sur les FMSO locaux sélectionnés pour chaque sous-système à chaque niveau. Avec les entrées et les sorties du système, tous les générateurs de résidus hiérarchiques calculés sont utilisés en ligne pour détecter et isoler les fautes à un certain niveau de la hiérarchie. Le processus d'isolation des fautes n'est utilisé en pratique que via la

matrice de signature. Les calculs ne sont déportés au niveau supérieur que si la faute n'est pas localement isolable. C'est l'idée de l'isolation à la demande, présentée initialement dans [Chanthery *et al.* 2016b].

8.4.3 Application et contributions

L'approche décentralisée a été testée sur un système de commande d'attitude et d'orbite (SCAO) d'un satellite en orbite terrestre basse (LEO). Concernant l'utilisation des MTES, les ensembles MTES et TES ont été calculés d'une manière globale. Ces ensembles sont comparés aux ensembles calculés de manière décentralisée. Les deux méthodes sont ensuite comparées sur des scénarios simulés en terme de temps de réponse et d'efforts de mise en œuvre dans [Chanthery *et al.* 2016b]. L'approche a été améliorée dans [Pérez *et al.* 2015] en utilisant les FMSO.

Les travaux exposés ici sont des travaux d'ingénierie, au sol, qui permettent la conception et le développement de systèmes de diagnostic basés sur des modèles dans le cadre d'une méthodologie classique d'ingénierie des systèmes. De cette façon, le développement de la fonction de diagnostic peut être rapproché de la méthodologie de développement d'une fonctionnalité nominale. On énonce ici les avantages et les inconvénients du développement d'une architecture de diagnostic distribuée/décentralisée suivant deux aspects : les efforts d'ingénierie et les coûts de calculs.

⇨ **Efforts d'ingénierie**

Au lieu d'exiger une visibilité complète sur le système, une telle approche d'ingénierie système décentralisée revêt plusieurs intérêts, mais nécessite quelques points précis.

1. L'effort de conception et d'implémentation des fonctions de diagnostic peut être partagé entre les partenaires du projet. Dans cette optique, une interface entre les diagnostiqueurs locaux doit être mise en place entre les différents partenaires responsables des différentes unités fonctionnelles ;
2. Une simulation du système peut être partagée, mais seuls les signaux d'interface ont besoin d'être visibles par tous les partenaires ;
3. Les partenaires peuvent fournir des diagnostiqueurs implémentés dans un premier temps dans un langage de modélisation tel que MATLAB/Simulink, puis, au fur et à mesure que le projet progresse, glisser vers un langage cible et valider l'ensemble sur un simulateur d'intégration.
4. L'architecture hiérarchique permet une mise à l'échelle de la méthode beaucoup plus aisée. Cet avantage est d'autant plus important que le développement de ces systèmes de grande dimension nécessite une plus grande rigueur de développement et des fonctions de diagnostic pour des fonctions souvent critiques.

Cette architecture correspond naturellement au processus d'ingénierie des systèmes qui procède à une décomposition fonctionnelle d'un système en sous-systèmes. L'architecture est hiérarchiquement évolutive et implémente un diagnostic basé sur les RRA.

Ces considérations de développement et de vérifications fonctionnelles pour des systèmes de FDIR réels peuvent être trouvées dans [Pecover 2010]. Des travaux antérieurs ont également comparé les efforts de conception et de développement par une approche décentralisée, et par une approche classique [Indra & Travé-Massuyès 2013].

⇨ **Coûts de calcul**

Nous avons également étudié le nombre de seuils, de dérivateurs et leurs filtres associés, d'intégrateurs et leurs filtres associés nécessaires à la mise en place des approches par résidus envisagées. Cette étude fournit un moyen simple d'évaluer les coûts du changement vers une architecture décentralisée. Alors que la mise au point de seuils va ajouter un coût

de mise au point et de validation, les dérivateurs et les intégrateurs introduisent quant à eux des coûts de calculs et une complexité numériques qui impactent la mise au point, la validation et les efforts en temps-réel. Dans les applications réelles actuelles, ce sont ces dernières considérations qui font pencher la balance en faveur de l'approche décentralisée. En effet, bien que nos études n'aient pris en compte que deux niveaux de conception, l'impact du passage à l'échelle pour un système réel de grande taille serait considérable.

8.5 Approche distribuée pour le diagnostic

8.5.1 Conception distribuée d'un diagnostiqueur

Une architecture distribuée suppose que chaque unité de diagnostic est identique en termes de rôle, avec une communication possible entre deux nœuds de diagnostic quelconques. Contrairement à l'architecture de diagnostic décentralisée, il n'y a pas de hiérarchie dans la décomposition du système. Au lieu de cela, le système Σ est décomposé en n sous-systèmes $\Sigma_i, i = 1, \ldots, n$ dont les modèles associés définissent une partition des équations Σ.

⇨ Génération distribuée de tous les FMSO globaux

Cette approche a été envisagée en premier dans les travaux de [Khorasgani *et al.* 2015]. On suppose que le modèle du système global n'est pas disponible, mais garantit néanmoins une diagnosticabilité maximale, c'est-à-dire la même diagnosticabilité qu'une approche centralisée. En plus de ce que propose [Khorasgani *et al.* 2015], nous avons prouvé, grâce aux propriétés sur les FMSO, qu'il est possible d'obtenir l'ensemble des FMSO globaux sans recalculer les FMSO pour les modèles locaux étendus par les modèles de sous-systèmes voisins. Au lieu de cela, notre approche utilise un algorithme de recherche qui identifie les ensembles de FMSO/CMSO partagés calculés localement qui peuvent devenir des FMSO globaux. L'algorithme 4 détaillé dans [Pérez *et al.* 2016] implémente cette procédure distribuée de calcul de l'ensemble des FMSO globaux.

Algorithme 4 : Génération de l'ensemble des FMSO globaux.

```
Φ = ∅;
pour i=1...n faire
    Φ_i^l ← Calculer les FMSO locaux de Σ_i;
    Φ_i^s ← Calculer les FMSO partagés de Σ_i;
    Ψ_i^s ← Calculer les CMSO partagés de Σ_i;
    pour chaque FMSO partagé φ ∈ Φ_i^s faire
        Etiqueter φ comme un FMSO racine : φ_r ← φ;
        Soit X^s_{φ_r} l'ensemble des variables partagées de φ_r;
        tant que il est possible de trouver un ensemble φ^c ⊇ φ_r en complétant φ_r
          grâce à la procédure P_1 et tel que φ^c n'est pas inclus dans Φ faire
            Compléter l'ensemble des FMSO globaux φ^c : Φ ← Φ ∪ φ^c;
    Φ ← Φ ∪ Φ_i^l;
retourner Φ
```

La procédure pour calculer un FMSO global φ^c à partir d'un FMSO partagé φ_r, notée $\mathcal{P}_1$, utilise la propriété 3. On commence par chercher une correspondance qui couvre chaque variable partagée du FMSO racine φ_r. Cette procédure est répétée pour les nouveaux ensembles de variables partagées qui apparaissent avec les FMSO partagés introduits à

chaque itération. Les itérations s'arrêtent lorsqu'aucune nouvelle variable partagée n'est introduite.

☞ Cette procédure explique comment obtenir un FMSO global à partir d'un FMSO partagé. Elle n'explique pas comment obtenir tous les FMSO globaux générables à partir d'un FMSO partagé.

La complexité de ce calcul augmente évidemment avec le nombre de variables partagées. Cependant, dans la pratique, les sous-systèmes sont généralement conçus de manière à ce que leurs liens soient assez faibles, de sorte qu'ils partagent peu de variables. Cela rend l'approche proposée applicable à des systèmes dynamiques complexes composés de plusieurs sous-systèmes.

L'architecture distribuée nécessite que chacun des diagnostics locaux effectue ses calculs indépendamment des autres, il n'est donc pas nécessaire que chaque diagnostic local partage son modèle local puisque seules les mesures sont partagées, ce qui présente un avantage de confidentialité. L'algorithme 4 montre comment il est possible d'obtenir les mêmes FMSO globaux et donc les mêmes relations de redondance analytiques que l'approche centralisée tout en maintenant la confidentialité de chaque modèle local.

⇨ Génération distribuée d'un sous-ensemble de FMSO globaux

Si les résidus correspondant à tous les FMSO globaux étaient générés et utilisés en ligne pour surveiller le système, on atteindrait bien évidemment une détectabilité et une isolabilité maximales. Cependant, tous ne sont pas nécessaires et il est plus efficace de minimiser leur nombre tout en conservant la même propriété.

Le but est d'obtenir un ensemble de diagnostiqueurs locaux distribués qui, ensemble, rendent le système complètement diagnosticable grâce à des FMSO locaux et des FMSO complétés. Ces diagnostiqueurs locaux sont conçus pour atteindre une diagnosticabilité maximale avec une communication restreinte entre les sous-systèmes. Tout d'abord, les ensembles FMSO locaux sont déterminés pour chaque sous-système Σ_i. Si ceux-ci ne sont pas suffisants pour détecter et isoler toutes les fautes dans F_i, alors un ensemble de FMSO complétés est déterminé pour obtenir une diagnosticabilité complète pour toutes les fautes de F_i, en considérant les contraintes de distance et de communication entre sous-systèmes [Pérez *et al.* 2016].

La conception des diagnostiqueurs est faite hors ligne et consiste à suivre les étapes données dans l'algorithme 5, effectué pour chaque sous-système Σ_i, $i = 1...n$. La procédure de calcul des "bons" FMSO complétés est d'autant plus pertinente qu'elle choisit des FMSO complétés dont les capacités d'isolation sont efficaces pour atteindre l'isolabilité maximale. Elle utilise une heuristique basée sur le nombre de variables partagées qui sera détaillée plus tard. Dans l'algorithme 5 [Pérez *et al.* 2016], le terme "meilleur" est utilisé dans le sens de cette heuristique.

8.5.2 Application et contributions

Après la conception hors ligne des diagnostiqueurs locaux à l'aide de l'algorithme 5, l'implémentation en ligne du diagnostiqueur distribué repose sur une banque de générateurs de résidus constituée de l'ensemble des RRA sélectionnées pour chaque diagnostiqueur local. Ces RRA sont instanciées en ligne à partir des signaux mesurés localement, pour chaque sous-système. La figure 8.5 illustre un diagnostiqueur distribué. Les RRA sont calculées et sélectionnées hors ligne. Chaque sous-système Σ_i possède un diagnostiqueur local noté $LD_i, i = 1, \ldots, n$. L'isolation des fautes est effectuée après la détection en utilisant des matrices locales de signatures de fautes similaires à celles définies dans la définition 51, page 100. Contrairement à l'architecture décentralisée, il n'y a pas de niveau supérieur.

Algorithme 5 : Génération des diagnostiqueurs locaux.

pour $i = 1...n$ **faire**
  $\Phi_i = \emptyset$;
  $\Phi_i^l \leftarrow$ Calculer les FMSO locaux de Σ_i;
  si *il existe une faute* $f \in F_i$ *qui est non détectable localement, ou non isolable localement avec les FMSO locaux* Φ_i^l **alors**
    $\Phi_i^s \leftarrow$ Calculer les FMSO partagés de Σ_i;
    $\Psi_i^s \leftarrow$ Calculer les CMSO partagés de Σ_i;
  tant que *il existe une faute* $f \in F_i$ *qui n'est pas détectable ou isolable* **faire**
    Soit $\varphi^* \in \Phi_i^s$ tel que $f \in F_{\varphi^*}$ le 'meilleur' FMSO partagé (non encore sélectionné) de Φ_i^s ;
    Étiqueter φ^* comme un FMSO racine : $\varphi_r \leftarrow \varphi^*$;
    Soit $X_{\varphi_r}^s$ l'ensemble des variables partagées de φ_r;
    $\Phi_i^{c*} \leftarrow$ Trouver un 'bon' FMSO complété incluant φ^* en sélectionnant les 'meilleurs' FMSO partagés pour couvrir toutes les nouvelles variables partagées introduites de manière récursive;
    $\Phi_i \leftarrow \Phi_i \cup \Phi_i^{c*}$;
    $\Phi_i^{l*} \leftarrow$ Sélectionner le plus petit ensemble de FMSO locaux couvrant la même diagnosticabilité que l'ensemble des FMSO locaux;
    $\Phi_i \leftarrow \Phi_i \cup \Phi_i^{l*}$;
  $RRA_i \leftarrow$ Générer les relations de redondance analytiques de LD_i à partir des FMSO de Φ_i;

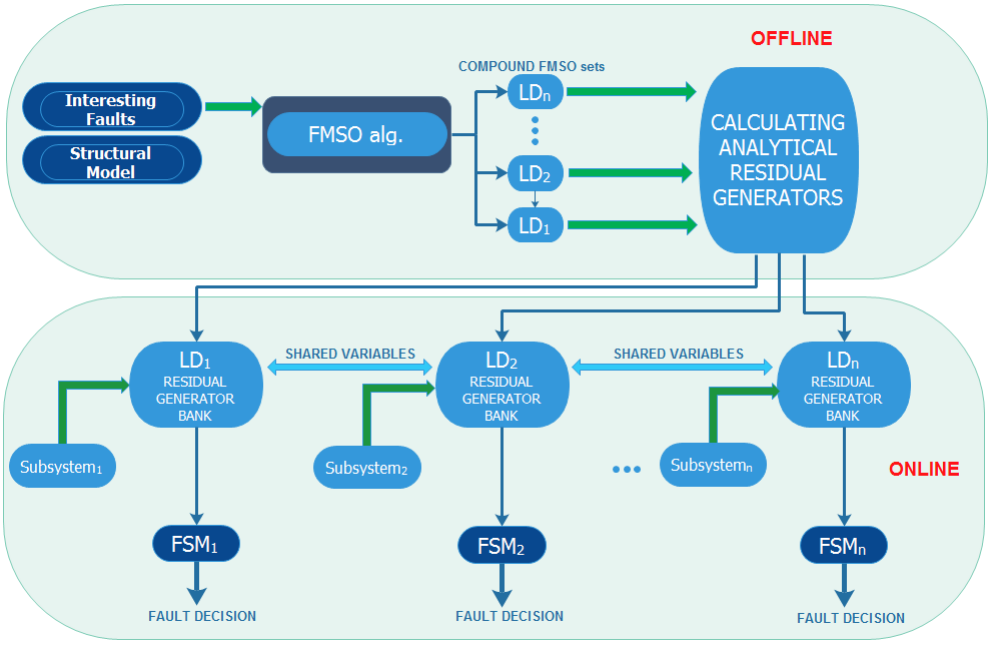


FIGURE 8.5 – Illustration du diagnostiqueur distribué.

L'architecture distribuée a été appliquée à un système de 4 réservoirs mis en série. Ce système a été utilisé dans [Khorasgani *et al.* 2015]. L'algorithme 4 a été implémenté sous Matlab et testé pour générer de manière distribuée tous les FMSO globaux. On retrouve les 165 FMSO globaux de manière distribuée.

L'architecture distribuée a également été appliquée sur un système industriel de dessalement par osmose inverse, proposé par l'Université Catholique du Pérou. Le modèle mathématique complet a été donné dans le manuscrit de thèse de Gustavo Pérèz [Pérez Zuniga 2017]. 7 fautes sont considérées sur le système, lui-même décomposé en 3 sous-systèmes, avec un total de 25 équations. L'algorithme 4 a été appliqué et les 5173 FMSO globaux ont été trouvés de manière distribuée.

Important : On rappelle que l'avantage de cet algorithme est de ne pas recalculer les FMSO pour tous les sous-systèmes voisins, tout en gardant la garantie de diagnosticabilité maximale. On obtient donc la même qualité de diagnostic en maintenant la confidentialité de chaque modèle local, ce qui est un apport industriel indéniable.

L'algorithme 5 a permis de calculer des RRA pour chacun des 4 sous-systèmes définis dans l'architecture distribuée pour les réservoirs. De même, pour le système de dessalement par osmose inverse, on constate que le nombre de FMSO globaux est très élevé (5173). Parmi ces FMSO, et grâce à l'algorithme 5, on sélectionne les FMSO optimaux qui minimisent le partage d'informations entre les sous-systèmes. Suivant ce critère, 7 FMSO partagés sont sélectionnés et donnent lieu à 7 FMSO complétés permettant de garantir la même diagnosticabilité.

Important : L'avantage de cet algorithme est de ne maintenir qu'un nombre très réduit de tests sur un système complexe, tout en gardant la confidentialité de chaque modèle local lors de l'élaboration des diagnostiqueurs. Le nombre de FMSO choisis pour les tests est au maximum égal au nombre de fautes sur le système.

8.6 Optimisation du choix des tests

L'objectif est ici de proposer un ensemble de méthodes d'optimisation pour sélectionner les tests de diagnostic générés à partir des FMSO. Ces méthodes optimisent le processus de diagnostic puisque comme on l'a déjà dit, tous les tests n'ont pas besoin d'être construits : seuls les tests nécessaires pour réaliser la détectabilité et l'isolabilité des fautes considérées sont utilisés.

Après une section sur les travaux existants sur la sélection de tests optimaux (section 8.6.1), deux types de résolution sont investigués. D'un côté, nous avons effectué des travaux dans le contexte non centralisé dans le cadre de la thèse de Gustavo Pérèz. Ces travaux, exposés dans la section 8.6.2 portent sur la résolution de problèmes de plus court chemin. Une première approche a été développée à partir des FMSO globaux. Une seconde approche utilise des algorithmes dont le point de départ est l'ensemble des FMSO partagés dans une architecture distribuée.

D'un autre côté, nous avons envisagé le problème comme un problème d'optimisation en nombres entiers. Cette approche est proposée dans la section 8.6.3.

8.6.1 Travaux existants sur la sélection de tests optimaux

Le problème de sélection de tests vise à minimiser le coût des tests tout en satisfaisant des contraintes d'isolabilité. Ce problème est également lié au problème de placement de capteurs, bien connu dans la littérature, qui consiste à rechercher la configuration de capteurs de coût minimum qui satisfait un ensemble donné de spécifications pour le diagnostic de panne (détectabilité, isolabilité).

Dans la littérature, la sélection de tests est aussi souvent associée au problème de la priorisation des tests qui correspond au choix du meilleur test, ou mesure, à effectuer pour isoler une faute. En pratique, cela fait partie intégrante de la tâche de dépannage. Ce domaine a reçu beaucoup d'attention [Pattipati & Alexandridis 1988, Pattipati & Dontamsetty 1992, Dick & Faivre 1993] mais pour autant que l'on sache, le problème n'a jamais été traité dans un cadre décentralisé/distribué. C'est l'une des principales contributions de la thèse de Gustavo Pérèz. Le manuscrit propose une étude de la littérature dans le cadre centralisé que l'on résume brièvement ici.

On citera notamment les travaux de [Rosich *et al.* 2007] qui ajoutent des capteurs de manière itérative pendant que les MSO sont générés de manière incrémentale. La principale amélioration de ce travail est l'inutilité de générer tous les MSO. [Krysander & Frisk 2008] présente un algorithme de placement de capteurs pour la détectabilité et l'isolabilité qui prend en compte une spécification d'isolabilité et les emplacements possibles des capteurs. Dans [Sarrate *et al.* 2012], le problème de positionnement des capteurs appliqué aux réseaux de distribution est également traité dans le cadre de l'analyse structurelle. Enfin, [Leal *et al.* 2015] présente une approche pour l'analyse de diagnosticabilité et le placement de capteurs basés sur des algorithmes génétiques. L'approche sélectionne le nombre minimum de MTES à sélectionner pour générer des tests de diagnostic. Les algorithmes génétiques apparaissent comme un outil efficace pour résoudre le problème combinatoire de la sélection de MTES à partir d'un modèle structurel. L'objectif de ce travail est assez proche du nôtre mais il est formulé dans un cadre centralisé. Dans notre travail, nous préférons utiliser le concept de FMSO qui est bien plus adapté pour un cadre non centralisé puisque contrairement aux MTES, chaque FMSO pointe vers un seul test.

8.6.2 Approches par résolution d'un problème de plus court chemin

Avec Gustavo Pérèz et Louise Travé-Massuyès, nous avons considéré le problème de la manière suivante. On considère que chaque diagnostiqueur local dispose d'un ensemble de capteurs candidats S (qui peuvent être communs à d'autres diagnostiqueurs locaux via des variables partagées). L'ensemble des capteurs permet de générer un certain nombre de tests. L'ensemble des tests est noté T. Le but est de sélectionner les tests de façon optimale pour satisfaire des contraintes de diagnostic.

Nous avons ainsi proposé trois algorithmes pour résoudre le problème de sélection des tests. La caractéristique commune de ces trois algorithmes est qu'ils s'apparentent tous les trois à des recherches heuristiques.

Dans ces travaux, nous avons avancé qu'il est possible et intéressant d'analyser un problème de sélection de test optimal en tant que planification et plus précisément un problème de recherche de chemin en partant d'un nœud initial dont l'état est totalement ambigu jusqu'à un diagnostic non (ou moins) ambigu. Ce point de vue est une des contributions de la thèse de Gustavo Pérèz.

La planification consiste à organiser de façon optimale un ensemble d'actions limité afin d'atteindre un objectif. Les actions consomment et produisent généralement des ressources, qui ont un coût et l'objectif est exprimé en tant que valeur souhaitée pour certaines de ces ressources. Du point de vue du diagnostic, la planification peut également être considérée comme pilotant un processus modélisé par un automate vers un état objectif, de manière optimale, lorsque toutes les transitions sont contrôlables. Chaque état représente alors un tuple de valeurs, une par ressource, et les transitions dérivent des actions possibles. Le problème consiste à trouver le plus court chemin dans un graphe orienté pondéré, d'un nœud initial à un ensemble de nœuds finaux possibles.

Dans cette partie, nous proposons d'utiliser des algorithmes de type A* pour résoudre

différentes variantes du problème de sélection des tests et sélectionner de façon optimale les meilleurs FMSO afin d'obtenir la meilleure détectabilité et isolation possible des fautes.

Le problème est connu pour être NP-difficile, mais des algorithmes peuvent être proposés, notamment des variantes du A*. En pratique, à condition que les heuristiques soient intelligemment conçues, ces approches fonctionnent beaucoup mieux que le pire cas, qui nécessite d'explorer le graphe entier.

☞ On rappelle ici qu'il n'y a pas d'échange d'informations de diagnostic entre les diagnostiqueurs locaux, seulement un échange de mesures ; ensuite, les FMSO sont nécessaires pour élaborer des tests de diagnostic qui prennent la forme de RRA. Sélectionner un test revient donc à sélectionner un FMSO. C'est pour cette raison que dans ce qui suit, un FMSO sera parfois appelé test, ou RRA.

Pour résoudre le problème de recherche de plus court chemin, les algorithmes de recherche de type "force brute" ne nécessitent aucune connaissance spécifique au domaine. A titre d'exemple, on peut citer la recherche en largeur d'abord, la recherche à coût uniforme, la recherche en profondeur d'abord [Korf 2010]. L'idée a été de rapprocher mes travaux en décision sur les algorithmes de recherche de plus court chemin et mes travaux en sélection de tests. L'algorithme le plus adapté pour un tel rapprochement est l'algorithme A*.

⇨ Notions élémentaires

On rappelle ici rapidement quelques notions élémentaires nécessaires à la compréhension des travaux, mais plus de détails peuvent être trouvés dans [Bondy & Murty 1976].

Un graphe G est un triplet ordonné $(V(G), E(G), \Gamma)$ où $V(G)$ est un ensemble non vide de sommets (ou nœuds), $E(G)$ est l'ensemble, disjoint de $V(G)$, des arêtes du graphe et Γ est une fonction d'incidence qui associe à chaque arête de G une paire non ordonnée de sommets.

Un problème de recherche est défini par un ensemble d'états (ou nœuds), un état de départ (ou nœud racine) et un ensemble d'états buts (ou nœuds buts). Une fonction successeur fournit un mapping d'un état à un ensemble d'états successeurs. Un chemin dans G est une suite finie non nulle $P = n_0 e_1 n_1 e_2 n_2 \dots e_k n_k$, dont les termes sont alternativement des sommets et des arêtes, tels que, pour $1 \leq i \leq k$, les extrémités de e_i sont n_{i-1} et n_i et tous les sommets sont distincts. Le problème de plus court chemin consiste à associer un nombre réel $w(e)$ à chaque arête e de G, appelé poids (également appelé coût ou score). Le graphe G est alors un graphe pondéré. Le but est de trouver un chemin de poids minimum reliant deux sommets spécifiés n_0 (le nœud racine) et n_b (n'importe quel nœud de l'ensemble d'états buts).

Dans les algorithmes de type "force brute" au meilleur d'abord, à chaque étape, le nœud suivant n à développer est souvent celui dont le coût $g(n)$ est le plus faible, où $g(n)$ représente la somme des coûts du nœud racine n_0 au nœud n. Les recherches heuristiques utilisent une fonction d'évaluation heuristique. Dans un problème de recherche de chemin à un seul agent, une fonction d'évaluation heuristique estime le coût du chemin optimal entre deux états. Pour un état d'objectif fixe, une évaluation heuristique $h(n)$ est une évalution du coût entre le nœud courant n et un état but. Par exemple, la distance euclidienne ou aérienne est une estimation de la distance à parcourir sur les routes entre deux lieux [Korf 2010].

L'algorithme A*, initialement présenté dans [Hart *et al.* 1968], combine une fonction de recherche au meilleur d'abord et une recherche heuristique pour calculer efficacement des solutions optimales.

Le score (ou coût) d'un nœud est $\mathcal{E}(n) = g(n) + h(n)$, où :

- $g(n)$ est le score du chemin de l'état initial n_0 au nœud actuel n,

- $h(n)$ est l'estimation heuristique du score du chemin depuis le nœud n vers un nœud but. L'heuristique doit être minorante pour être admissible. Plus l'heuristique est précise, plus l'état but est atteint rapidement et plus la précision est élevée.
- $\mathcal{E}(n) = g(n) + h(n)$ est l'approximation du chemin le plus court jusqu'au but. $\mathcal{E}$ est appelé la *fonction d'évaluation*. $\mathcal{E}(n)$ est calculée pour tout nœud n pendant [2] afin de déterminer quel nœud doit être développé ensuite.

A chaque étape, le nœud ayant la plus faible valeur de fonction d'évaluation est choisi pour l'exploration. A condition que l'heuristique soit minorante, l'algorithme se termine lorsqu'un nœud but est choisi pour l'exploration. Le principal inconvénient de l'algorithme A*, et en fait de toute recherche au meilleur d'abord, est son exigence en taille mémoire [Korf 2010].

⇨ **Enoncé du problème de plus court chemin pour la sélection de tests**

Dans nos travaux, nous avons développé une variation de l'algorithme A* qui nous permet de sélectionner correctement les FMSO (et CMSO) [3] étant donné un ensemble de fautes à isoler.

Variables du problème Le problème est connu comme un problème d'optimisation combinatoire. Au lieu de choisir des actions comme dans un problème de planification, on choisira quels FMSO (CMSO) doivent être inclus pour générer les tests. Un nœud du graphe correspondra donc à un FMSO (ou CMSO) φ associé à son support de fautes.

☞ Notons que le graphe n'est pas explicite.

Critère Le but est d'inclure dans chaque test le plus petit nombre de connexions avec les autres sous-systèmes. Il est possible de pondérer l'ordre de chaque sous-système impliqué [Khorasgani *et al.* 2015]. Nous définirons pour chaque algorithme les différentes fonctions de score et l'heuristique.

Etats buts Les états but sont définis comme les états pour lesquels toutes les fautes sont isolables (et détectables). Les FMSO (CMSO) doivent être choisis de façon à ce que l'union de leurs supports de test inclue toutes les fautes et que celles-ci soient isolables les unes des autres. On définit pour cela la notion d'ensemble d'ambiguïté et de degré d'isolabilité.

Définition 53 (Ensembles d'ambiguïté) *Un ensemble d'ambiguïté est un ensemble de fautes qui ne sont pas isolables deux à deux. L'ensemble des ensembles d'ambiguïté à l'étape i est noté $\mathcal{A}_i$.*

Définition 54 (degré d'isolabilité) *Le degré d'isolabilité à l'étape i est le cardinal de l'ensemble $\mathcal{A}_i$.*

On définit une matrice de signature de fautes notée $\mathcal{S}$. Si on reprend la définition de matrice de signatures de fautes, chaque ligne de la matrice $\mathcal{S}$ correspond à un test, c'est-à-dire à un FMSO global. Chaque colonne correspond à une faute. On définit également une colonne pour le cas nominal. L'isolabilité de toutes les fautes est atteinte lorsque les colonnes de $\mathcal{S}$ sont toutes différentes. C'est la propriété que l'algorithme de sélection doit atteindre.

Dans le cas distribué, l'ensemble des fautes couvertes par un FMSO complété est l'union des ensembles de fautes des FMSO partagés qui le composent. Par conséquent, la propriété d'isolation peut être vérifiée de manière équivalente avec une matrice similaire à $\mathcal{S}$ mais

2. On rappelle qu'un nœud pendant est un nœud appartenant à l'ensemble des voisins des nœuds déjà explorés, mais qui n'a pas encore été exploré.

3. On rappelle que la sélection s'effectue sur les FMSO globaux dans le cas décentralisé et parmi les FMSO et CMSO partagés dans le cas distribué.

dans laquelle les lignes correspondent aux FMSO/CMSO partagés, étant donné que ce sont les éléments sélectionnés par l'algorithme dans ce cas. Par un léger abus de langage, on appellera également cette matrice, la matrice de signatures de fautes et on la notera aussi $\mathcal{S}$.

La matrice $\mathcal{S}$ est de dimension $n^r \times n^f$ où n^r est le nombre total de FMSO (CMSO) et $n^f - 1$ est le nombre de fautes. Elle évolue à chaque étape de l'algorithme en fonction du FMSO/CMSO choisi. La matrice de signatures de fautes à l'étape i est notée $\mathcal{S}_i$. Au début de l'algorithme, la matrice $\mathcal{S}_0$ est remplie de 0 : il n'y a qu'un seul ensemble d'ambiguïté composé de toutes les fautes.

À chaque étape i, on modifie une ligne de la matrice S_i. Le FMSO choisi couvre un ensemble de fautes et doit améliorer au mieux l'isolabilité, de sorte que la propriété suivante est vérifiée : $Card(\mathcal{A}_i) > Card(\mathcal{A}_{i-1})$. Le but est de choisir le test qui maximise $Card(\mathcal{A}_i) - Card(\mathcal{A}_{i-1})$. C'est le cas lorsque chaque ensemble d'ambiguïté de $\mathcal{A}_{i-1}$ est coupé en deux ensembles isolables. Nous appelons cette opération la *coupe dichotomique*.

Le problème consiste à construire la matrice de signatures pour que toutes les fautes (ou un nombre maximum de fautes) deviennent isolables de manière optimale.

L'état à l'étape i est donc défini par l'ensemble des ensembles d'ambiguïté $\mathcal{A}_i$. Au début de l'algorithme, l'état $\mathcal{A}_0 = \{\{f_0, f_1, \ldots, f_n\}\}$ inclut un ensemble d'ambiguïté unique qui inclut toutes les fautes.

Les FMSO qui n'augmentent pas le degré d'isolabilité ne sont pas utiles. Cela signifie qu'ils ne coupent aucun ensemble d'ambiguïté.

Les états but sont définis comme des états où aucun FMSO supplémentaire n'augmente le degré d'isolabilité. Si toutes les fautes sont isolables, ces états ont un degré d'isolabilité égale à $n + 1$:

$$\mathcal{A}_{n_b} = \{\{f_0\}, \{f_1\}, \ldots, \{f_n\}\}. \tag{8.8}$$

⇨ Une solution à partir de FMSO globaux

Le premier algorithme de sélection prend comme entrée un ensemble de FMSO globaux. Ceux-ci peuvent avoir été générés soit par l'algorithme 4 (Génération distribuée de tous les FMSO globaux), à travers la hiérarchie décentralisée pour assurer la détectabilité et l'isolabilité de toutes les fautes du système, ou bien de manière centralisée.

Il propose une solution qui peut être utilisée pour une architecture centralisée classique ou pour une architecture décentralisée. Dans le cas décentralisé, une fois la sélection effectuée, chaque FMSO sélectionné est affecté à son sous-système d'origine dans la hiérarchie décentralisée.

Les principes de l'algorithme A* sont les suivants (on suppose ici que toutes les fautes sont isolables. L'algorithme peut être adapté facilement si ce n'est pas le cas).

- Le nœud de départ n_0 a pour état $\mathcal{A}_0 = \mathcal{F} = \{\{f_0, f_1, \ldots, f_n\}\}$ (ensemble des fautes).
- Un nœud n_i du graphe est identifié par l'ensemble d'ambiguïté $\mathcal{A}_i$ résultant des FMSO qui ont été utilisés sur le chemin du nœud de départ n_0 au nœud n_i.
- Les voisins d'un nœud sont tous les nœuds qui peuvent être atteints en sélectionnant un FMSO qui augmente le cardinal de l'ensemble d'ambiguïté $\mathcal{A}_i$.
- Un nœud but a pour état $\mathcal{A}_f = \{\{f_0\}, \{f_1\}, \ldots, \{f_n\}\}$.

Le score g de tout FMSO supplémentaire est de 1 car on minimise le nombre de FMSO. Soit n_i le nœud courant, alors $g(n_i)$ est le nombre de FMSO globaux inclus dans la solution à l'étape i. Il s'agit du nombre de tests renseignés dans la matrice $\mathcal{S}_i$ à l'étape courante.

Étant donné un nœud n_i du graphe de recherche, la valeur heuristique de n_i à un nœud but est calculée par la formule suivante :

$$h(n_i) = Max_j \left\lceil \frac{ln(|\mathcal{A}_i^j|)}{ln(2)} \right\rceil, \tag{8.9}$$

où $\mathcal{A}_i^j$ sont les différents ensembles d'ambiguïté de l'ensemble $\mathcal{A}_i$ à l'étape i et $|.|$ est le cardinal de l'ensemble.

Cette heuristique calcule le nombre minimum de FMSO qui sont nécessaires pour désambiguïser tous les ensembles A_i^j de l'ensemble d'ambiguïté $\mathcal{A}_i$. Pour l'un de ces ensembles A_i^j, le nombre minimal de FMSO est $\left\lceil \frac{ln(|A_i^j|)}{ln(2)} \right\rceil$. Donc pour tous les ensembles de $\mathcal{A}_i$, le maximum de ces nombres est requis. Cette heuristique provient des propriétés de la "coupe dichotomique" qui sont énoncées juste après.

Cet algorithme A* pour la sélection de FMSO globaux est appelé *algorithme A* global*. On peut en trouver une description détaillée dans le manuscrit [Pérez Zuniga 2017].

On démontre les deux propriétés suivantes.

Propriété 4 *Le degré d'isolabilité est une fonction monotone strictement croissante.*

Propriété 5 *La coupe dichotomique est la manière la plus efficace d'augmenter le degré d'isolabilité.*

La coupe dichotomique permet de partitionner chaque ensemble d'ambiguïté de $\mathcal{A}_i$ en deux, de sorte que le degré d'isolabilité est doublé. Or il a été prouvé que le degré d'isolabilité peut au mieux doubler lors de l'ajout d'un nouvel FMSO, par conséquent la coupe dichotomique est la manière la plus efficace d'augmenter le degré d'isolabilité.

⇨ **Algorithmes dans le cas distribué**

Les deuxième et troisième algorithmes sont conçus pour une architecture distribuée. Ils partent des FMSO partagés obtenus pour chaque sous-système et génèrent uniquement les FMSO globaux requis pour atteindre la détectabilité et l'isolabilité pour chaque faute de chaque sous-système. Ils diffèrent dans la façon dont chaque sous-système est traité : en parallèle pour le second et itérativement pour le troisième.

L'algorithme *FirstLocalThenComplete* utilise une stratégie A* en parallèle pour chaque sous-système, puis considère les FMSO partagés sélectionnés comme des FMSO racine pour former des FMSO composés qui sont finalement des FMSO globaux du système. On appelle cela "compléter" un FMSO partagé.

L'algorithme *IterativeFindAndComplete* applique une stratégie A* à un sous-système et complète l'ensemble des FMSO partagés sélectionnés avec des FMSO / CMSO partagés des autres sous-systèmes. Ensuite, l'algorithme traite un autre sous-système (s'il n'est pas déjà diagnosticable à partir des opérations d'achèvement précédentes), et ainsi de suite jusqu'à ce que tous les sous-systèmes aient été traités. Contrairement au premier algorithme d'optimisation, les deux algorithmes du cas distribué sont eux-mêmes distribués.

Ces algorithmes aboutissent à une sélection de tests qui est globalement non optimale.

8.6.3 Approche par optimisation d'un problème en nombres entiers

La sélection de tests optimaux par optimisation d'un problème en nombres entiers a été entreprise lors d'une collaboration avec l'équipe ROC (Recherche Opérationnelle, Optimisation Combinatoire et Contraintes), initiée en 2013.

⇨ **Notations et notions élémentaires**

En entrée des problèmes en nombres entiers que l'on considère, on dispose de la matrice de signatures de fautes du système $\mathcal{S}$. On rappelle ici que la matrice des signatures de fautes

contient l'ensemble des résidus (ou tests) du système au niveau des lignes, et les n^f fautes considérées pour le système sur les colonnes. On suppose dans cette partie que tous les FMSO globaux du système ont été calculés au préalable. L'ensemble Φ des FMSO globaux génère l'ensemble T des tests, de cardinal n^t. L'ensemble des fautes est noté F, de cardinal n^f.

Le but est de sélectionner les tests de façon optimale pour satisfaire des contraintes de diagnostic. On choisit ici de parler de tests pour indiquer la sélection d'un FMSO global.

Trois problèmes ont été modélisés, puis implémentés et testés. Les différences entre les problèmes reposent sur les notions suivantes : tests instrumentés, capteurs fautifs ou non fautifs, objectifs. Ces notions vont être définies dans le paragraphe suivant.

La matrice de signatures de fautes du système $\mathcal{S}$ est augmentée d'un certain nombre de colonnes qui vont correspondre aux capteurs dont dispose le système et qui vont permettre de mesurer les variables incluses dans les tests. Soit S l'ensemble des capteurs, de cardinal n^s. L'ensemble des capteurs permet de mettre en œuvre l'ensemble des tests.

La signature d'un capteur $s \in S$ est un vecteur binaire $SS(s) = [\sigma_{1s}, \sigma_{2s}, ...\sigma_{n^t s}]^T$ où σ_{ks}, $k = 1...n^t$ est calculé à partir de $T \times S \longrightarrow \{0, 1\}$ tel que $\sigma_{ks} = 1$, si le test $t_k \in T$ utilise le capteur s et $\sigma_{ks} = 0$ sinon. Il existe des cas où les capteurs peuvent être considérés comme fautifs. La faute d'un capteur est noté f_s, pour $s \in S$. L'ensemble des fautes capteurs est noté F^S.

La matrice de signatures de fautes augmentée est appelée matrice de signatures et est notée $\mathbf{S}$. Elle possède $n^f + n^s$ colonnes et n^t lignes.

On note $\mathcal{S}^*$ la sous-matrice sélectionnée pour effectuer les tests : il s'agit d'un choix de lignes et de colonnes dans $\mathbf{S}$.

Définition 55 (test instrumenté) *Un test est instrumenté pour une sous-matrice $\mathcal{S}^*$ si tous les capteurs nécessaires pour le réaliser sont disponibles dans $\mathcal{S}^*$.*

⇨ Problème (P1) : tests instrumentés, capteurs non fautifs

On suppose ici que tous les tests sont instrumentés. On suppose également que les capteurs utilisés ne tombent jamais en panne. Le problème de sélection de tests dans $\mathbf{S}$ revient à faire une sélection directement dans la matrice de signatures de fautes $\mathcal{S}$. Une dernière hypothèse suppose que toutes les fautes de F sont isolables deux à deux.

L'objectif est de minimiser le nombre de tests.

Soit $\mathbb{z}_t$ une variable booléenne qui indique si un test t est sélectionné ou non ($\mathbb{z}_t = 1$ si oui et $\mathbb{z}_t = 0$ sinon).

Soit $\mathbb{f}_{tjk}$ une variable booléenne qui indique si la faute f_j est isolable de la faute f_k grâce au test t ou non ($\mathbb{f}_{tjk} = 1$ si oui et $\mathbb{f}_{tjk} = 0$ sinon). Les $\mathbb{f}_{tjk}$ sont calculées a priori.

Le problème d'optimisation se résume ainsi :

$$min \sum_{t \in T} \mathbb{z}_t, \tag{8.10}$$

sous :

$$\sum_{t \in T} \mathbb{f}_{tjk} \mathbb{z}_t \geq 1 \quad \text{avec} \quad (f_j, f_k) \in F^2 \quad et \quad f_j \neq f_k \tag{8.11}$$

$$\mathbb{z}_t \in \{0, 1\} \quad \text{pour} \quad t \in T. \tag{8.12}$$

L'équation 8.11 sert à vérifier l'isolabilité dans la matrice $\mathcal{S}^*$.

Le problème peut facilement être étendu à la minimisation des coûts des capteurs. Soit y_s une variable réelle, p_s le coût du capteur s et σ_{ts} une variable booléenne qui indique si un test t utilise un capteur s ou non ($\sigma_{ts} = 1$ si oui et $\sigma_{ts} = 0$ sinon). Cette information provient de la signature du capteur s.

Le problème d'optimisation se résume ainsi :

$$min \sum_{s \in S} p_s y_s, \tag{8.13}$$

sous :

$$y_s \geq \mathbb{z}_t \quad t \in T \quad et \quad s \in S \quad \text{tel que} \quad \sigma_{ts} = 1 \tag{8.14}$$

$$\sum_{t \in T} \mathbb{f}_{tjk} \mathbb{z}_t \geq 1 \quad \text{avec} \quad (f_j, f_k) \in F^2 \quad et \quad f_j \neq f_k \tag{8.15}$$

$$y_s \geq 0 \quad \text{avec} \quad s \in S; \quad \mathbb{z}_t \in \{0,1\} \quad \text{avec} \quad t \in T \tag{8.16}$$

⇨ **Problème (P2) : tests instrumentés, capteurs pouvant être fautifs**

Dans cette partie, les tests sont encore considérés instrumentés. Par contre les capteurs peuvent être fautifs, c'est à dire qu'ils peuvent donner une mauvaise information.

L'objectif est de minimiser le coût des capteurs sans réduire la capacité de détection et en garantissant l'isolabilité de chacune des fautes. On définit e_{jk} une variable réelle positive. Les autres variables ont été définies dans le problème (P1). Le problème d'optimisation se résume ainsi :

$$min \sum_{s \in S} p_s y_s, \tag{8.17}$$

sous :

$$y_s \geq \mathbb{z}_t \quad t \in T \quad et \quad s \in S \quad \text{tel que} \quad \sigma_{ts} = 1 \tag{8.18}$$

$$\sum_{t \in T} \mathbb{f}_{tjk} \mathbb{z}_t \geq e_{jk} \quad \text{avec} \quad t \in T, \quad f_j \in F \text{ et } f_k \in (F \cup F^S), \quad f_j \neq f_k \quad et \quad 0 \leq e_{jk} \tag{8.19}$$

$$y_s \geq 0 \quad \text{avec} \quad s \in S \tag{8.20}$$

$$0 \leq e_{jk} \leq 1 \quad \text{avec } f_j \in F \text{ et } f_k \in (F \cup F^S), \quad f_j \neq f_k \tag{8.21}$$

$$\mathbb{z}_t \in \{0,1\} \quad \text{avec} \quad t \in T \tag{8.22}$$

L'équation 8.18 sert à vérifier l'instrumentation des tests. L'équation 8.19 sert à vérifier l'isolabilité dans la matrice $\mathcal{S}^*$ en faisant l'hypothèse que tous les capteurs sont isolables entre eux.

⇨ **Problème (P3) : tests instrumentés, capteurs pouvant être fautifs**

La dernière modélisation vise à maximiser l'isolabilité tout en minimisant le coût des capteurs. Le modèle doit aussi être robuste aux pannes des capteurs. On suppose dans un premier temps que seul un capteur tombe en panne à la fois. On note m une pondération entre la maximisation sur l'isolabilité et la minimisation du nombre de tests. Soit T'_s l'ensemble de tests qui sont instrumentés par le capteur s'. Les autres variables ont été définies dans les problèmes (P1) et (P2).

Le problème d'optimisation se résume ainsi :

$$max(\, m \sum_{(k,l)} e_{kl} - (m-1) \sum_{s \in S} p_s y_s) \quad \text{avec} \quad f_k \in F \quad et \quad f_l \in (F \cup F^S), \tag{8.23}$$

sous :

$$y_s \geq \mathbb{z}_t \quad t \in T \quad et \quad s \in S \quad \text{tel que} \quad \sigma_{ts} = 1 \tag{8.24}$$

$$\sum_{t \in T} \mathbb{f}_{tjk} \mathbb{z}_t \geq e_{jk} \quad \text{avec} \quad t \in T, \quad f_j \in F, \quad f_k \in (F \cup F^S), \quad f_j \neq f_k \quad et \quad 0 \leq e_{jk} \tag{8.25}$$

$$\sum_{t \in T \setminus T'_s} \mathbb{f}_{tjk} \mathbb{z}_t \geq e_{jk} \quad \text{avec} \quad s' \in S, \quad f_j \in F, \quad f_k \in (F \cup F^S), \quad f_j \neq f_k \quad et \quad 0 \leq e_{jk} \tag{8.26}$$

$$y_s \geq 0 \quad \text{avec} \quad s \in S \tag{8.27}$$

$$0 \leq e_{jk} \leq 1 \quad \text{avec} \quad f_j \in F, f_k \in (F \cup F^S) \tag{8.28}$$

$$\mathbb{z}_t \in \{0, 1\} \quad \text{avec} \quad t \in T. \tag{8.29}$$

L'équation 8.24 sert à vérifier l'instrumentation des tests. L'équation 8.25 sert à vérifier l'isolabilité dans la matrice $\mathcal{S}^*$ en faisant l'hypothèse que tous les capteurs sont isolables entre eux. L'équation 8.26 sert à vérifier l'isolabilité de la matrice $\mathcal{S}^*$ dans le cas où un capteur tombe en panne et quel que soit ce capteur.

8.6.4 Applications et contributions

⇨ Choix de tests par recherche de plus courts chemins

L'*algorithme A* global* a été implémenté sous Matlab et testé sur le procédé de dessalement à osmose inverse. Les deux algorithmes *FirstLocalThenComplete* et *IterativeFindAndComplete* ont été implémentés sous Matlab et testés. A partir de ces recherches heuristiques, il est possible de déterminer le nombre de FMSO minimal pour isoler toutes les fautes. L'*algorithme A* global* trouve que pour détecter et isoler les 7 fautes du système, seuls 3 FMSO globaux sont nécessaires.

En conclusion, on peut souligner que ces algorithmes peuvent ne pas être globalement optimaux, mais seulement localement optimaux. Ils sont néanmoins très efficaces dans la sélection de tests dans des architectures non centralisées (décentralisées ou distribuées).

Le problème de la sélection optimale des FMSO et CMSO partagés peut être formulé comme un problème d'optimisation combinatoire. Une telle formulation a déjà été proposée en utilisant la programmation en nombres entiers [Bagajewicz *et al.* 2004, Sarrate *et al.* 2007b, Rosich *et al.* 2009] mais pas dans un contexte distribué, ce travail est une perspective de nos travaux.

⇨ Choix de tests par optimisation d'un problème en nombres entiers

Concernant les solutions par optimisation d'un problème en nombres entiers, les modèles ont été implémentés sous Gurobi en python. Les systèmes déjà utilisés précédemment (satellite LEO et procédé de dessalement à osmose inverse) ont été utilisés. Un ensemble de tests optimaux a été trouvé pour chaque cas, dans un cadre centralisé. Ces résultats montrent que l'approche est extrêmement efficace en centralisé. Nous souhaitons par la suite éprouver la robustesse de l'approche à plusieurs pannes capteurs et sur des problèmes à grande échelle. Ceci fait l'objet d'un sujet de stage proposé conjointement par l'équipe DISCO et l'équipe ROC pour le printemps 2018.

8.7 Publications liées à cette partie

▶ S. Indra, L. Travé-Massuyès et E. Chanthery. *A decentralized FDI scheme for spacecraft: Bridging the gap between model based FDI research and practice.* Progress in Flight Dynamics, Guidance, Navigation, Control, Fault Detection, and Avionics, vol. 6, 2011. also in "4th European Conference for Aerospace Sciences, St Petersburg, 2011, Russia"

▶ S. Indra, L. Travé-Massuyès et E. Chanthery. *Decentralized diagnosis with isolation on request for spacecraft.* IFAC Proceedings Volumes, vol. 45, no. 20, pages 283–288, 2012

▶ E. Chanthery, L. Travé-Massuyès et S. Indra. *Fault isolation on request based on decentralized residual generation.* IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 46, no. 5, pages 598–610, 2016

▶ C. G. Pérez, L. Travé-Massuyès, E. Chanthery et J. Sotomayor. *Decentralized diagnosis in a spacecraft attitude determination and control system.* In Journal of Physics: Conference Series, volume 659-1. IOP Publishing, 2015

▶ C. G. Pérez, E. Chanthery, L. Travé-Massuyès et J. Sotomayor. *Fault-Driven Minimal Structurally Overdetermined Set in a Distributed Context.* In the 27th International Workshop on Principles of Diagnosis: DX-2016, 2016

▶ C. G. Pérez, L. Travé-Massuyès, E. Chanthery et J. Sotomayor. *Fault-Driven Structural Diagnosis Approach in a Distributed Context.* In IFAC World Congress, page 1, 2017

Troisième partie

Prospectives

9 Projet scientifique

Mon projet de recherche s'articule suivant quatre grands axes, illustrés sur la figure 9.1. Trois d'entre eux sont dans la continuité de mes recherches actuelles. Comme le montre la figure, les intersections des axes de recherche ne sont évidemment pas vides et plusieurs thématiques de mon projet sont à cheval entre deux, voire trois de ces axes. Les rectangles en noir représentent les thématiques que je souhaite aborder dans le futur.

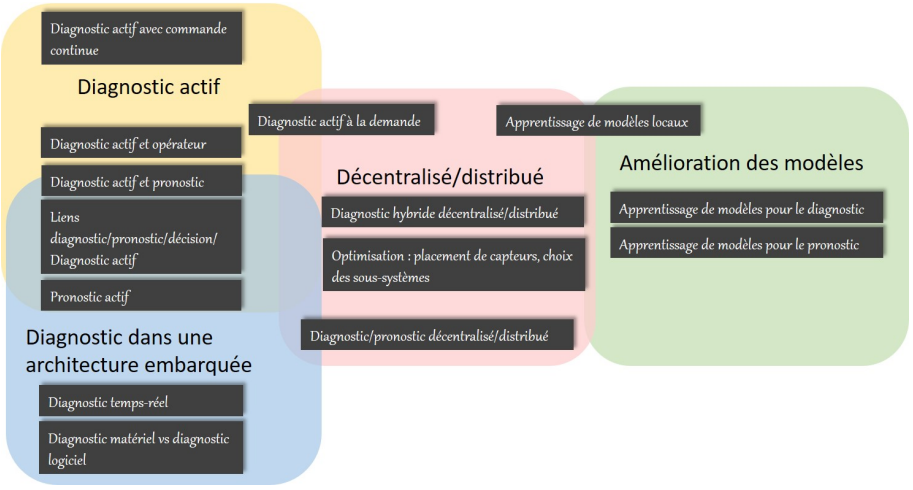


FIGURE 9.1 – Projet de recherche : vue globale.

9.1 Amélioration des modèles

Le premier axe a pour objectif de traiter l'amélioration des modèles pour le diagnostic et pour le pronostic. En effet, tous les travaux que j'ai exposés dans ce manuscrit se situent dans une approche à base de modèles. Une conclusion à ces travaux est que cette approche trouve bien souvent sa limitation dans l'élaboration même du modèle. La première partie de mon projet de recherche vise donc à utiliser de nouvelles méthodes, notamment des méthodes à base de données, pour améliorer les résultats des méthodes déjà développées.

Collaborations envisageables pour cette thématique (liste non exhaustive) : Pauline Ribot, Louise Travé-Massuyès, Audine Subias, Euriell Le Corronc, Marie-Véronique Le Lann.

9.1.1 Identification de verrous

L'amélioration des modèles passent par trois grands verrous.

- Le premier verrou est la recherche de solutions (modèles, algorithmes) pour prendre en compte la nature intrinsèquement incertaine d'un système réel. Cette **gestion des incertitudes dans les modèles et les algorithmes** vient de la volonté de refléter au mieux les exigences des applications industrielles. Les incertitudes à

prendre en compte proviennent de sources variées : tout d'abord la modélisation du système, mais aussi les incertitudes relatives aux observations sur le système.
- Le deuxième verrou est la **prise en compte de la dynamique d'un modèle au cours du temps**. Penser qu'un modèle développé lors de la conception d'un système restera valide tout au long de sa vie est malheureusement utopique. Un verrou majeur est donc de faire évoluer les modèles de manière à ce qu'ils soient adaptés au vieillissement du système, à ses modifications tout au long de sa vie.
- Le dernier verrou vise la **simplification du processus de modélisation**. En effet, le travail de modélisation est une tâche souvent ardue, à la frontière entre les spécialistes du domaine applicatif et du domaine d'utilisation. Ainsi, pour développer un modèle de diagnostic pour un satellite, il faudra mobiliser les connaissances de la partie métier mais également les compétences de chercheurs en diagnostic. La simplification de la tâche de modélisation peut faire gagner énormément de temps et d'argent aux deux parties et constitue un enjeu important.

9.1.2 Premières contributions sur l'amélioration des modèles

Concernant la gestion des incertitudes dans les modèles et les algorithmes, plusieurs pistes ont déjà été investiguées, on les rappelle ici brièvement. Les travaux effectués avec Emmanuel Bénazéra pour utiliser les Processus Décisionnels de Markov Partiellement Observables (PDMPO) en vue de l'intégration directe des tâches de diagnostic, d'observation, de planification et de réparation est liée à la volonté d'utiliser des modèles pour le diagnostic plus fidèles aux réalités des systèmes, et notamment entâchés d'incertitude. Par ailleurs, les travaux de thèse de Quentin Gaudel, co-encadrés avec Pauline Ribot, ont permis de proposer un nouveau formalisme appelé les Réseaux de Petri Hybrides Particulaires (Hybrid Particle Petri Nets ou HPPN), dont l'objectif est de prendre en compte un certain nombre d'incertitudes, notamment les incertitudes sur les observations ou les bruits de mesure, mais également dans le modèle.

Nous avons également travaillé avec Pauline Ribot sur la prise en compte de modifications de la dynamique d'un système au cours du temps. Lors de l'encadrement du stage de M2R de Frédéric Chatrie sur le diagnostic d'un système à base de modèles adaptatifs, nous avons proposé des pistes pour développer un diagnostiqueur dynamique, prenant en compte des modifications du modèle dans le temps. De même, lors de l'encadrement du stage de M2R de Ben Alia Bouzidi, en collaboration avec Florent Teichteil (Airbus Group), nous avons eu pour objectif d'observer le système au cours de sa vie pour prendre en compte, grâce à des techniques d'apprentissage, des modifications de modèles au cours du temps. Ces travaux ont permis de lancer des pistes intéressantes qui doivent être investiguées plus en avant sur le thème de l'apprentissage de modèles pour le diagnostic et l'apprentissage de modèles pour le pronostic (apprentissage de paramètres de dégradation notamment).

Ceci mène au troisième verrou de cet axe, lié à la simplification du processus de modélisation. Pour lever ce verrou, nous avons envisagé avec Pauline Ribot de mêler les travaux de diagnostic à base de modèles aux travaux de diagnostic à base de données. L'idée est d'apprendre directement des modèles à partir des observations récupérées sur un système en fonctionnement, ou bien d'enrichir un modèle pré-existant. Ceci permet d'une part de simplifier la tâche de modélisation, mais également d'adapter le modèle si le système se dégrade au cours du temps. C'est ce point qui est détaillé dans les sections suivantes.

9.1.3 Apprentissage de modèles continus, discrets, hybrides, de dégradation

Dans ce travail, on souhaite examiner une approche à mi-chemin entre les méthodes à base de modèles et les méthodes à base données pour améliorer l'efficacité des algorithmes utilisés pour la gestion de santé de systèmes complexes.

On part du constat que les algorithmes à base de modèles obtiennent d'excellents résultats à partir du moment où les modèles en entrée sont corrects. L'idée est donc de conserver ces algorithmes quasiment à l'identique, et de travailler sur l'amélioration des entrées des algorithmes.

L'idée est de partir d'un premier modèle écrit par exemple à la conception, par un expert. Ce modèle sera ensuite enrichi en continu et modifié par des observations au moyen de techniques d'apprentissage agissant tout autant sur les paramètres que sur la structure [Kwong & Yonge-Mallo 2011], [Barbosa Roa 2016]. L'utilisation de méthodes d'apprentissage pour identifier des structures précises de modèles formels diffère radicalement de l'approche standard consistant à construire des modèles de type "boîtes noires", dans le sens où le modèle structuré appris pourra être exploité par des algorithmes efficaces conçus pour raisonner sur une structure spécifique.

On envisage l'apprentissage de modèles pour le diagnostic et le pronostic sous divers angles.

- Apprentissage de transitions discrètes entre états discrets dans une machine à états ou un réseau de Petri : les états du système sont supposés tous connus. Suite aux observations du fonctionnement du système, on apprend l'existence de transitions entre les états. On peut éventuellement étiqueter la transition d'une ou plusieurs conditions (logiques ou pas). Une information sur la fréquence d'occurrence de franchissement de la transition peut également être apprise. On envisage d'effectuer cette étape en utilisant notamment des méthodes de clustering.
- Apprentissage de nouveaux états : les états du système sont inconnus, ou partiellement connus. L'idée est ici d'utiliser des méthodes de classification non supervisée de manière à détecter de nouveaux états sur le système. Ces états apparaissent en cours de fonctionnement, soit parce qu'ils n'ont pas été anticipés lors de la conception, soit parce qu'ils sont issus de la dérive d'un état suite au vieillissement du système, soit encore parce que suite à l'évolution non anticipée du système au cours du temps, un nouvel état non anticipé apparaît. On crée alors un nouvel état dans le modèle à événements discrets et les transitions associées.

L'apprentisage de machines à états finis ou de Réseaux de Petri a déjà été traité dans la littérature [Zheng *et al.* 1993], [Leclercq *et al.* 2008]. En 2004, la thèse de Tatiana Kempowski [Kempowsky 2004] a proposé la construction d'un modèle discret sous la forme d'un automate à états finis, soumise à un expert, via des méthodes de classification. Néanmoins cette proposition est uniquement pour les Ssystèmes à événements discrets, et chaque nouvelle solution est traitée hors-ligne. Plus récemment, les travaux de Nathalie Barbosa [Barbosa Roa 2016], [Barbosa *et al.* 2017], proposent une méthodologie pour générer un modèle à événements discrets à partir de la méthode de partitionnement Dyclee. Elle est basée sur l'utilisation d'automates temporisés. Chaque fois qu'une nouvelle classe est détectée, elle est ajoutée au modèle. Néanmoins cette proposition n'a pas été formalisée. En outre, elle n'envisage pas d'apprendre la dynamique continue du système, seulement de s'y adapter. De même, elle ne prévoit pas l'apprentissage de la dégradation. On envisage donc d'utiliser en plus des méthodes d'apprentissage de paramètres, déjà bien connues dans la littérature, pour combiner les deux types de méthodes à des fins de diagnostic/pronostic pour des systèmes hétérogènes.

Cela passe par deux points supplémentaires :

- Apprentissage de paramètres pour des équations d'état : dans le cas de systèmes continus ou hybrides, on envisage d'apprendre ou de suivre les modifications de paramètres pour les dynamiques continues. Des techniques d'estimation et d'identification de paramètres seront étudiées [Lei *et al.* 2017], [Gertler 2015]. On s'attachera en particulier à des techniques qui pourront gérer des paramètres dynamiques et des observations sous incertitudes.
- Apprentissage de paramètres de lois de dégradation : de la même façon que pour les paramètres des équations de la dynamique continue du système, on envisage d'apprendre les paramètres des équations de dégradation d'un système. [An *et al.* 2015] propose une étude comparative d'un certain nombre d'algorithmes bien adaptés pour le pronostic. Il pointe un certain nombre de difficultés à surmonter pour les méthodes à base de données et les méthodes à base de modèles, dont l'une des principales est l'incertitude sur les données, que nous avons déjà identifiée. Une difficulté majeure pour cette partie reste également l'horizon de temps d'observation nécessaire pour l'apprentissage ainsi qu'un nombre de données d'entraînement limité [Singleton *et al.* 2015]. A priori, cette solution ne pourra être envisagée que sur des systèmes utilisés en masse et dont le vieillissement est assez rapide. On pourra penser par exemple aux systèmes intelligents (smart systems) qui impliquent un nombre très important de composants (capteurs notamment).

Certains travaux ont déjà envisagé le couplage diagnostic/pronostic et apprentissage. Les travaux de [Le *et al.* 2014] proposent l'utilisation de modèles de Markov cachés pour le diagnostic et le pronostic, dans le cas de systèmes multi-modes où la dégradation varie suivant les modes. Cependant, des probabilités a priori sont utilisées pour chaque mode et un modèle de dégradation simple est utilisé. Les travaux de [Yu 2017] proposent quant à eux une méthode d'apprentissage d'un modèle évolutif utilisant des modèles de Markov cachés adaptatifs. La méthode reconnaît en ligne de nouveaux états de santé et les dégradations sont quantifiées. Ces travaux peuvent être un point de départ intéressant pour nos études, néanmoins l'interdépendance entre les états de santé doit être étudiée. Par ailleurs, pour prendre en compte ces modèles évolutifs enrichis par des techniques d'apprentissage, les algorithmes de diagnostic/pronostic devront être adaptés.

9.1.4 Apprentissage de modèles locaux

A cause de la complexification des systèmes, il est difficile de travailler à partir d'un modèle global pour un système complexe. Pour lutter contre la complexité inhérente à ces systèmes, comme il a déjà été dit dans ce manuscrit, une méthode consiste à considérer le système comme un ensemble de composants et d'adopter une démarche décentralisée/distribuée. Les techniques d'apprentissage de modèles locaux seront semblables aux techniques pour des modèles en centralisé. Dans le domaine de la commande, en robotique, [Meier *et al.* 2014] propose un algorithme permettant de mixer des modèles analytiques et la régression bayésienne via l'apprentissage de modèles locaux. A notre connaissance, il n'existe pas de travaux dans ce sens dans le domaine du diagnostic et du pronostic de fautes.

Un verrou important sera la gestion des interactions entre les modèles des sous-systèmes ainsi que la pertinence d'un apprentissage sur un sous-système précis. Une piste serait de vérifier des propriétés de diagnosticabilité et de pronosticabilité localement pour savoir si le modèle local actuel d'un composant est suffisant ou s'il doit être enrichi par des informations supplémentaires provenant de sous-systèmes adjacents. De nombreux travaux ont étudié les propriétés de diagnos-

ticabilité dans un cadre distribué [Pencolé 2004], [Ye & Dague 2017] ou décentralisé [Moreira *et al.* 2011], [Cabasino *et al.* 2011]. Concernant les systèmes hybrides, on peut notamment citer [Zaatiti *et al.* 2018], qui propose des abstractions intéressantes pour étudier la diagnosticabilité de manière hiérarchique. Concernant la pronosticabilité, on peut également faire référence aux travaux de [Genc & Lafortune 2009], [Kumar & Takai 2010], [Ribot *et al.* 2013], [Chen & Kumar 2014]. L'ensemble de ces travaux et les critères associés serviront de base de décision pour lancer un apprentissage local à la demande.

Évidemment, les algorithmes de diagnostic existants vont devoir être adaptés pour utiliser des modèles locaux au niveau des composants, qui peuvent de surcroit être évolutifs.

9.2 Approches non centralisées

Le deuxième axe de mon projet de recherche prend la suite des travaux sur le diagnostic décentralisé/distribué. Outre les problématiques d'apprentissage de modèles locaux évoquées dans la partie précédente, trois nouveaux thèmes pourraient être abordés.

Collaborations envisageables pour cette thématique (liste non exhaustive) : Pauline Ribot, Louise Travé-Massuyès, Yannick Pencolé.

9.2.1 Optimisation du placement de capteurs en décentralisé

L'optimisation du placement de capteurs a déjà été investiguée à la fois dans la thèse de Gustavo Pérèz et dans des travaux connexes concernant une approche par optimisation combinatoire. Une des perspectives à long terme de ce travail est de s'appuyer sur des techniques d'optimisation combinatoire. Cet axe de travail serait à mener en collaboration avec l'équipe ROC du LAAS-CNRS. On pourra envisager des études de type optimisation multi-objectif [Deb 2014] pour le problème de sélection de tests, ou bien des architectures distribuées, parallèles ou multi-agents [Ma *et al.* 2015].

Au niveau des thématiques portées par l'équipe DISCO, on pourra s'intéresser à d'autres problèmes que celui de la sélection de capteurs. En décentralisé ou en distribué, l'optimisation du choix de la "découpe" du système en sous-systèmes pertinents peut être intéressante à étudier : en effet, la majorité des travaux partent d'un ensemble de sous-systèmes fixés a priori par des contraintes fonctionnelles ou par l'utilisateur [Daigle *et al.* 2015a]. Dans le cadre des systèmes non linéaires, les travaux de [Boem *et al.* 2015] ont proposé une méthode de conception permettant d'optimiser la topologie d'une architecture de diagnostic distribuée, notamment le nombre minimal de diagnostiqueurs locaux pour garantir la diagnosticabilité. Ces résultats peuvent être étendus dans le cas des pannes de capteurs, mais également pour la problématique du pronostic.

9.2.2 Diagnostic hybride décentralisé/distribué

Suite aux travaux de thèse de Saurabh Indra et Gustavo Pérèz, le diagnostic de systèmes hybrides dans un cadre décentralisé et/ou distribué reste à formaliser.

Pour cela, on peut s'intéresser à des travaux passés qui pourront servir de socle de travail. Le travail que nous avons mené jusque là lors des thèses de Saurabh Indra et Gustavo Pérez a permis d'étudier la décentralisation du diagnostic pour les systèmes continus.

En ce qui concerne les systèmes à événements discrets, le diagnostic décentralisé ainsi que le diagnostic distribué ont déjà été largement traités, notamment dans les travaux de [Debouk *et al.* 2000, Pencolé & Cordier 2005, Cordier & Grastien 2007].

Enfin, les travaux de Medhi Bayoudh sur le diagnostic des systèmes hybrides [Bayoudh & Travé-Massuyès 2014], mais aussi les travaux de Quentin

Gaudel [Gaudel 2016], et les travaux de [Khorasgani & Biswas 2017] apportent la composante "diagnostic de systèmes hybrides".

Ces trois parties forment la pierre angulaire qui pourrait permettre d'élaborer une solution dans le cadre du diagnostic de systèmes hybrides dans le cadre décentralisé et/ou distribué.

Dans la littérature, peu de travaux se sont intéressés au diagnostic de systèmes hybrides en décentralisé/distribué. Les travaux de [Louajri & Sayed-Mouchaweh 2014] traitent à la fois les fautes paramétriques et les fautes discrètes dans des systèmes hybrides, via une approche modulaire et une approche décentralisée. Cependant, les travaux ne traitent que les fautes simples, et ne prennent pas en compte les interactions possibles entre les fautes, ni la dégradation. [Feng *et al.* 2016] propose une approche de diagnostic distribuée basée sur les conflits possibles pour les systèmes hybrides. [Bregon & Daigle 2016] présente quant à lui un cadre pour l'isolation de fautes qualitatives pour des systèmes hybrides, basé sur la décomposition structurelle du modèle.

9.2.3 Pronostic et diagnostic décentralisés

On a constaté, dans le chapitre 6, qu'une limitation des approches à base de modèles est également la difficulté d'aborder les modèles globaux et de gérer la complexité des systèmes. Par ailleurs, les liens entre les processus de diagnostic et de pronostic ont été également étudiés. Lors du passage aux cas non centralisés, les algorithmes de diagnostic et de pronostic vont devoir être révisés de manière à prendre en compte des diagnostics et pronostics locaux. La propagation de l'information au niveau des autres composants, et les interactions locales et non locales des résultats de diagnostic et de pronostic doivent être étudiées.

La thématique du pronostic décentralisé a déjà été abordée dans la littérature [Yin & Li 2016], [Kumar & Takai 2010], mais pour des systèmes à événements discrets et se rapporte plus à un problème d'atteignabilité qu'à un pronostic au sens d'une prédiction de RUL comme on l'entend dans nos travaux. Les travaux de [Daigle *et al.* 2012] proposent un pronostic distribué sous la forme d'une estimation/prédiction distribuée. Cette approche se rapporte plus à nos travaux, mais doit être adaptée dans le cas de systèmes hybrides où la dégradation évolue par mode. Elle suppose par ailleurs que le problème de prédiction est décomposable en sous-problèmes indépendants.

Dans [Ferdowsi & Jagannathan 2017], les auteurs proposent la conception d'un diagnostic et d'un pronostic décentralisés pour les systèmes à temps discret non linéaires. Chaque diagnostiqueur local est un observateur basé sur un réseau de neurones. Les fautes sont isolées grâce à superviseur qui reçoit les informations de tous les diagnostiqueurs locaux. Cette unité permet également de donner l'information du RUL en utilisant des informations locales. Cette approche est la seule à lier le diagnostic et le pronostic en décentralisé. Elle nécessite néanmoins les mesures de tous les états. On préférera dans notre cas utiliser des interactions entre les résultats de diagnostic et de pronostic pour choisir d'affiner des informations pertinentes pour le système, de manière à réduire la quantité d'information nécessaire, en fonction des propriétés de diagnosticabilité et/ou de pronosticabilité du système, et des demandes de l'opérateur (fautes les plus intéressantes par exemple).

9.2.4 Diagnostic actif décentralisé ou diagnostic actif à la demande

Enfin un dernier point sur les aspects non centralisés concerne le diagnostic actif. Dans le cas où le diagnostic du système est ambigu, l'idée est de mettre au point une commande

locale pour chacun des sous-systèmes dont l'état de santé reste ambigu. Le diagnostic actif pourrait se dérouler à plusieurs niveaux : au niveau local, la mise au point d'une commande (continue et/ou discrète) pourrait permettre d'isoler localement une faute dans un sous-système. Si le diagnostic actif local échoue, on pourrait élaborer une commande impliquant plusieurs sous-systèmes et permettant de désambiguiser la situation. Cette idée peut s'appliquer aux systèmes continus, en prenant la suite des travaux de thèse de Gustavo Pérèz, ou aux systèmes à événements discrets, en envisageant la décentralisation des travaux sur le diagnostic actif de Medhi Bayoudh. Enfin, elle peut être envisagée sur les systèmes hybrides, une fois que les travaux sur le diagnostic des systèmes hybrides en non centralisés seront achevés ou en prenant la suite des travaux de thèse de Quentin Gaudel.

Dans le cadre des systèmes continus, les travaux de [Franceschelli *et al.* 2009] proposent une détection passive locale, puis une identification active de la faute par les sous-systèmes voisins du sous-système fautif. [Raimondo *et al.* 2016] présentent une méthodologie de commande tolérante aux fautes décentralisée basée sur une approche de diagnostic actif. La détection est faite de manière passive, et l'isolation s'effectue avec des techniques d'isolation actives. En informatique, le diagnostic actif distribué a été envisagé comme un test actif par un programme [Brodie *et al.* 2003]. La conclusion partagée par ces travaux est une solution dans laquelle la détection des fautes est passive et locale, et déclenche un processus actif utilisant des actions de sous-systèmes au voisinage de la faute détectée. Le diagnostic actif décentralisé peut ainsi être appelé "diagnostic actif à la demande", et être mis en regard de l'isolation à la demande, proposée dans la thèse de Saurabh Indra.

9.3 Un diagnostic actif avancé

Le troisième axe de mon projet de recherche prend la suite des travaux sur le diagnostic actif. Outre les problématiques de diagnostic actif décentralisé évoquées dans la partie précédente, quatre nouveaux thèmes pourraient être abordés.

Collaborations envisageables pour cette thématique (liste non exhaustive) : Louise Travé-Massuyès, Audine Subias, Soheib Ferghani, Carine Jauberthie, Pauline Ribot, Euriell Le Corronc.

9.3.1 Diagnostic actif avec des actions continues

Les travaux sur le diagnostic actif auxquels j'ai contribué s'appliquent aux systèmes à événements discrets, d'une part, et d'autre part aux systèmes hybrides dont la dynamique continue a été complètement abstraite. Les actions envisagées sur le système sont des actions discrètes (de type on/off) uniquement. Des travaux de l'équipe DISCO visent maintenant à étendre la problématique du diagnostic actif à des commandes continues. Les travaux de thèse de Florian De Mortain [de Mortain *et al.* 2015] proposent d'associer un algorithme de diagnostic à base de modèle multi-modes et une planification conditionnelle optimale basée sur les MDP (Processus Décisionnels de Markov). Par ailleurs, des travaux basés sur une approche à incertitudes bornées vont être abordés dans une future thèse encadrée par Carine Jauberthie et Soheib Ferghani. On envisage par la suite de combiner les différentes approches et d'étudier l'intégration des objectifs de mission lors de la phase de diagnostic actif. Par ailleurs, on voudrait s'inspirer de travaux issus du monde de la commande [Habets & van Schuppen 2005], [Tabatabaeipour *et al.* 2009a], [Tabatabaeipour *et al.* 2009b], [Campbell *et al.* 2002] pour intégrer également le diagnostic actif par commande continue pour les systèmes hybrides. L'idée est de synthétiser une commande maximisant la différence entre la sortie normale et la sortie fautive pour pou-

voir isoler la faute, tout en restant dans les critères de performance souhaités pour la sortie du système.

9.3.2 Diagnostic actif influencé par le pronostic

La perspective à long terme de mes travaux est de lier le diagnostic, le pronostic, la réparation et le diagnostic actif. En ce qui concernent les liens entre la fonction de diagnostic actif et la fonction de pronostic, l'idée est la suivante : dans le cas où le diagnostic est ambigu, plusieurs mécanismes peuvent se mettre en place pour aider à la décision. D'une part un pronostic peut être lancé, pour donner une indication sur les fautes à venir sur le système à plus ou moins longue échéance. D'autre part, un diagnostic actif peut être lancé. Néanmoins, dans beaucoup de cas, l'algorithme de diagnostic actif a pour difficulté majeure de choisir vers où lancer ses recherches pour obtenir un plan efficace : en effet il y a souvent trop d'actions possibles (trop de branchements possibles à chaque étape) ou une heuristique peu efficace. Si on rajoute en entrée du diagnostic actif des informations de pronostic, on peut guider la recherche pour désambiguiser les fautes les plus probables dans le futur : on gagne ainsi en efficacité.

9.3.3 Pronostic actif

Une autre idée est de mettre en place des mises en garde concernant les séquences d'actions effectuées par le système. Le diagnostic actif consiste pour nous à effectuer des actions pour raffiner un diagnostic ambigu. Néanmoins, dans [Sampath *et al.* 1998], le diagnostic actif consiste à réduire l'espace des actions du système pour garder le système dans des régions diagnosticables et faire ainsi de la synthèse de contrôleur.

Pour le pronostic, on se rapproche de ce dernier cas. On appellera pronostic actif un processus qui indiquera au système des actions ou séquences d'actions déconseillées. On cherchera à découvrir quelles séquences d'actions pourraient mener vers des défaillances précoces et on les indiquera en sortie de l'algorithme de pronostic actif. Cette sortie pourra être mise en entrée du processus de décision : lorsque plusieurs plans de coût équivalent seront possibles pour atteindre un objectif, la donnée du pronostic actif pourra être déterminante pour optimiser les actions à effectuer. Ceci a déjà été largement étudié pour des systèmes continus : les travaux de [Langeron *et al.* 2017], [Salazar *et al.* 2017] ou moins récemment [Khelassi *et al.* 2011] étudient non seulement l'impact de la commande sur la dégradation des composants, mais proposent également des politiques de reconfiguration de la commande prenant en compte le RUL calculé pour le système. L'article [Chemweno *et al.* 2018] donne un état de l'art récent sur les méthodes de prise de décision pour la maintenance sous incertitudes, en tenant compte d'informations de pronostic notamment [Hu *et al.* 2012].

Une étude pour les systèmes hybrides pourrait être intéressante. De même, la sortie du pronostic actif pourra être mise en entrée du diagnostic actif.

9.3.4 Diagnostic actif et opérateur

Bien que le mouvement actuel aille vers l'automatisation totale des procédés, des domaines critiques, comme l'aéronautique ou le spatial, gardent l'humain dans la boucle de décision et de maintenance.

Ce point vise à intégrer ou ré-intégrer un opérateur humain dans la boucle de gestion de santé pendant le processus de diagnostic actif. On peut envisager cette intégration suivant deux aspects.

Le premier aspect consiste à utiliser un algorithme de diagnostic actif qui peut intégrer des actions de l'opérateur, associées à des actions effectuées par le système de manière autonome. Concernant les observations, les résultats des actions peuvent être observés de manière semi-automatique : les observations peuvent remonter automatiquement sur le système ou bien provenir d'une observation effectuée par un humain. Cet aspect ressemble évidemment aux problématiques de troubleshooting habituelles, assistées par le diagnostic actif.

Le deuxième aspect consiste à faire interagir l'algorithme lui-même avec l'opérateur. En plus des indications heuristiques que peut obtenir l'algorithme de diagnostic actif, l'avis d'un opérateur expert sur le système peut pondérer les choix dans l'exploration du graphe. Cette solution peut être mieux acceptée par l'opérateur et donc être plus applicable dans des domaines critiques cités précédemment.

Ces idées ne sont évidemment pas nouvelles. Déjà énoncées dans [Isermann 1997] ou [Rouse 1978], elles sont de nouveau d'actualité face aux questions d'éthique soulevées par le "tout autonome" et la volonté des industriels de contrôler et comprendre les algorithmes, notamment d'apprentissage.

9.4 Intégration avancée du diagnostic dans une architecture embarquée

Le dernier axe vise à continuer les travaux sur l'intégration du diagnostic dans une architecture embarquée. Bien que les derniers points de la partie sur le diagnostic actif participent à cet axe, comme on peut le voir sur la figure 9.1, une vue globale de l'architecture et des points particuliers relatifs au domaine de l'embarqué peuvent être mis en lumière.

9.4.1 Vers une vue globale de l'architecture de gestion de santé

Beaucoup de travaux de mon projet de recherche pourraient rentrer dans cet axe de recherche. L'objectif de mes recherches est en effet globalement de faire de lien entre le processus de diagnostic et les autres processus intervenant dans une architecture embarquée, voire l'opérateur s'il s'agit d'un système semi-autonome.

On a déjà posé dans les sections précédentes quelques briques d'interactions envisagées : on citera notamment les liens diagnostic actif et pronostic, pronostic et décision, diagnostic actif et opérateur.

Une boucle opérationnelle a déjà été envisagée dans un stage de M2R mais mérite un approfondissement plus important, notamment au niveau apprentissage et influence des processus les uns sur les autres.

L'idée générale est la suivante : le système est observé, ces observations peuvent nourrir un processus qui va apprendre ou mettre à jour le modèle du système. Ce modèle évolutif va être utilisé par les fonctions de diagnostic et de pronostic pour suivre l'état de santé du système. Si des zones de l'espace d'état restent inconnues ou trop incertaines et qu'elles intéressent l'opérateur, alors on engagera ce qu'on appelle un apprentissage actif : il s'agira alors de trouver un plan d'actions pour que les observations récupérées ou stockées sur le système soient pertinentes pour raffiner les modèles et les rendre de meilleure qualité, soit pour un diagnostic plus précis, soit pour un pronostic plus précis.

Par ailleurs, les actions de réparation sont également à prendre en compte lors des recherches. Pour le moment, c'est une dimension que nous avons toujours négligée ; néanmoins, concernant le pronostic de fautes, la réparation ne remet absolument pas "les comp-

teurs à zéro". Il s'agit donc de modéliser l'impact d'une réparation sur les diagnostics et les pronostics affectés par l'intervention d'un opérateur humain. Cette révision de modèles peut être prise en compte comme une évolution discontinue du modèle, plus complexe qu'un simple remplacement de composant.

Un dernier point vise à collaborer avec des chercheurs experts en planification. L'objectif serait ici de prendre en compte le diagnostic et surtout le pronostic lors de l'élaboration du plan de manière à assurer la mission, minimiser l'usure des composants et/ou maximiser la disponibilité du système.

9.4.2 Diagnostic/Pronostic temps-réel

Cette problématique de travail répond aux travaux déjà effectués en diagnostic à la demande, ou diagnostic à temps contraint. Dans le cadre d'une application sur des systèmes critiques, il est nécessaire de vérifier des propriétés sur le temps de réponse de l'algorithme de diagnostic et/ou de pronostic. Une collaboration est envisagée avec Pierre-Emmanuel Hladik de l'équipe VERTICS du LAAS sur une vérification formelle de certaines propriétés de l'algorithme et des modèles, ainsi que sur l'orchestration des tâches dans une architecture.

Collaboration déjà identifiée pour cette thématique : Pierre-Emmanuel Hladik (VERTICS, LAAS-CNRS).

9.4.3 Diagnostic matériel et diagnostic logiciel

Dans cet axe, on souhaite étudier la complémentarité des approches de diagnostic matériel et des approches plus logicielles du domaine de la sûreté de fonctionnement. En effet, les logiciels des systèmes embarqués critiques nécessitent une part de sécurité importante que l'on peut garantir à l'aide de tests logiciels menés avant et après le déploiement sur le système cible. D'autre part les composants matériels embarqués sont souvent soumis à un environnement hostile qui peut entraîner des fautes sur les composants que les algorithmes de diagnostic peuvent détecter, voire isoler. Le couplage des tests logiciels avec le diagnostic permettrait la détection des anomalies sur le fonctionnement de composants matériels et logiciels et pourrait donc améliorer la sûreté des systèmes autonomes.

Un premier travail devrait être effectué dans le cadre d'un stage de M2R entre avril et septembre 2018 dans la thématique de l'axe espace, en collaboration avec Guillaume Auriol de l'équipe TSF du LAAS-CNRS.

Collaboration déjà identifiée pour cette thématique : Guillaume Auriol (TSF, LAAS-CNRS).

A Liste des publications

Aussi disponibles sur https://homepages.laas.fr/echanthe/ avec les pdf associés.

Revues et chapitres de livres

▶ Q. Gaudel, E. Chanthery, P. Ribot et M. J. Daigle. Fault diagnosis of hybrid dynamic and complex systems, chapitre Diagnosis of Hybrid Systems Using Hybrid Particle Petri Nets: Theory and Application on a Planetary Rover, pages 209–241. Springer, 2018

▶ Q. Gaudel, P. Ribot, E. Chanthery et M. J. Daigle. Health Monitoring of a Planetary Rover Using Hybrid Particle Petri Nets, volume 9698 of *Lecture Notes in Computer Science*, chapitre Application and Theory of Petri Nets and Concurrency. PETRI NETS 2016. Lecture Notes in Computer Science, pages 196–215. Springer, Cham; Kordon F., Moldt D. (eds), 2016

▶ E. Chanthery, L. Travé-Massuyès et S. Indra. *Fault isolation on request based on decentralized residual generation.* IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 46, no. 5, pages 598–610, 2016

▶ C. G. Pérez, L. Travé-Massuyès, E. Chanthery et J. Sotomayor. *Decentralized diagnosis in a spacecraft attitude determination and control system.* In Journal of Physics: Conference Series, volume 659-1. IOP Publishing, 2015

▶ Q. Gaudel, E. Chanthery et P. Ribot. *Hybrid Particle Petri Nets for Systems Health Monitoring under Uncertainty.* Int. Journal of Prognostics and Health Management, vol. 6, no. 022, 2015

▶ S. Indra, L. Travé-Massuyès et E. Chanthery. *A decentralized FDI scheme for spacecraft: Bridging the gap between model based FDI research and practice.* Progress in Flight Dynamics, Guidance, Navigation, Control, Fault Detection, and Avionics, vol. 6, 2011. also in "4th European Conference for Aerospace Sciences, St Petersburg, 2011, Russia"

▶ E. Chanthery et Y. Pencolé. *Modélisation et intégration du diagnostic actif dans une architecture embarquée.* Journal européen des systèmes automatisés, vol. 43, no. 7-9, pages 789–803, 2009

▶ M. Barbier et E. Chanthery. *Autonomous Mission Management for Unmanned Aerial Vehicles.* Aerospace Science and Technology, vol. 8, pages 359–368, 2004

Conférences internationales avec comité de lecture

▶ P. Ribot, E. Chanthery et Q. Gaudel. *HPPN-based Prognosis for Hybrid Systems.* In Annual Conference of the Prognostics and Health Management Society 2017, Proceedings of the Annual Conference of the Prognostics and Health Management Society 2017, St. Petersburg, United States, Octobre 2017

▶ C. G. Pérez, L. Travé-Massuyès, E. Chanthery et J. Sotomayor. *Fault-Driven Structural Diagnosis Approach in a Distributed Context.* In IFAC World Congress, page 1, 2017

▶ C. G. Pérez, E. Chanthery, L. Travé-Massuyès et J. Sotomayor. *Fault-Driven Minimal Structurally Overdetermined Set in a Distributed Context.* In the 27th International Workshop on Principles of Diagnosis: DX-2016, 2016

▶ Y. Pencolé, E. Chanthery et T. Peynot. *Definition of Model-based diagnosis problems with Altarica.* In 27th International Workshop on Principles of Diagnosis (DX-2016), page 8p., Denver, CO, United States, Octobre 2016

▶ Q. Gaudel, P. Ribot, E. Chanthery et M. J. Daigle. Health Monitoring of a Planetary Rover Using Hybrid Particle Petri Nets, volume 9698 of *Lecture Notes in Computer Science*, chapitre Application and Theory of Petri Nets and Concurrency. PETRI NETS 2016. Lecture Notes in Computer Science, pages 196–215. Springer, Cham; Kordon F., Moldt D. (eds), 2016

▶ Q. Gaudel, P. Ribot et E. Chanthery. *Vers une architecture de surveillance de santé d'un système hybride sous incertitudes.* In Modélisation des Systèmes Réactifs, France, 2015

▶ E. Chanthery, Y. Pencolé, P. Ribot et L. Travé-Massuyès. *HYDIAG: extended diagnosis and prognosis for hybrid systems.* In the 26th International Workshop on Principles of Diagnosis (DX-2015), 2015

▶ Q. Gaudel, E. Chanthery et P. Ribot. *Health Monitoring of Hybrid Systems Using Hybrid Particle Petri Nets.* In Annual Conf. of the PHM Society, USA, 2014

▶ Q. Gaudel, E. Chanthery, P. Ribot et E. Le Corronc. *Hybrid systems Diagnosis using modified particle Petri nets.* In 25^{th} Int. Workshop on Principles of Diagnosis, Austria, 2014

▶ S. Zabi, P. Ribot et E. Chanthery. *Health Monitoring and Prognosis of Hybrid Systems.* In Annual Conf. of the PHM Society, 2013

▶ C. Jauberthie et E. Chanthery. *Optimal input design for a nonlinear dynamical uncertain aerospace system.* IFAC Proceedings Volumes, vol. 46, no. 23, pages 469–474, 2013

▶ E. Chanthery et P. Ribot. *An Integrated Framework for Diagnosis and Prognosis of Hybrid Systems.* In 3^{rd} Workshop on Hybrid Autonomous System, Italy, 2013

▶ S. Indra, L. Travé-Massuyès et E. Chanthery. *Decentralized diagnosis with isolation on request for spacecraft.* IFAC Proceedings Volumes, vol. 45, no. 20, pages 283–288, 2012

▶ M. Maiga, E. Chanthery et L. Travé-Massuyès. *Hybrid system diagnosis: Test of the diagnoser HYDIAG on a benchmark of the international diagnostic competition DXC'2011.* IFAC Proceedings Volumes, vol. 45, no. 20, pages 271 – 276, 2012. 8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes

▶ M. Godichaud, E. Chanthery, O. Buffet et M. Contat. *Formalizing and Solving Information Collection Problems with Autonomous Sensor Systems.* IFAC Proceedings Volumes, vol. 44, no. 1, pages 2208–2213, 2011

▶ E. Chanthery, Y. Pencolé et N. Bussac. *An AO*-like algorithm implementation for active diagnosis.* In 10th International Symposium on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS, 2010

▶ E. Chanthery et Y. Pencolé. *Principles of self-maintenance in an on-board architecture including active diagnosis.* Self-* and Autonomous Systems: reasoning and integration challenges (SAS-09), page 43, 2009

▶ E. Chanthery et Y. Pencolé. *Monitoring and active diagnosis for discrete-event systems.* IFAC Proceedings Volumes, vol. 42, no. 8, pages 1545–1550, 2009

▶ E. Benazera et E. Chanthery. *The Challenge of Solving POMDPs for Control, Monitoring and Repair of Complex Systems.* In Proceedings of the 19th International Workshop on Principles of Diagnosis (DX'08), 2008

▶ E. Chanthery, M. Barbier et J-L. Farges. *Planning algorithms for autonomous aerial vehicle.* In 16th IFAC World Congress, volume 16, 2005

▶ E. Chanthery, M. Barbier et J.-L. Farges. *Integration of Mission Planning and Flight Scheduling for Unmanned Aerial Vehicles.* In ECAI'04 - Workshop on "Planning and Scheduling: Bridging Theory to Practice", Valencia, Spain, August 22-23 2004

▶ E. Chanthery, M. Barbier et J.-L. Farges. *Mission Planning for autonomous Aerial Vehicles.* In IAV2004 - 5th IFAC Symposium on Intelligent Autonomous Vehicles, 2004

▶ E. Chanthery et M. Barbier. *Functional Modules for Intermixed Planning and Execution of an Observation Mission.* In 18th Bristol UAV Systems Conference, April 2003

Conférences nationales avec comité de lecture

▶ M. Godichaud, E. Chanthery, O. Buffet et M. Contat. *Formalisation et résolution de problèmes d'acquisition d'informations par des systèmes autonomes.* In ROADEF (12e congrès annuel de la Société française de Recherche Opérationnelle et d'Aide à la Décision), Saint-Etienne, France, 2011

Publications pédagogiques

▶ B. Sareni, G. Fontan, E. Chanthery et S. Caux. *OrdoNet, un outil de modélisation et d'analyse des graphes potentiel-tâche sous Matlab.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 8, no. hors série 1, page 1003, 2009

▶ A. Subias, E. Chanthery, G. Le Corre, J. Martin et V. Mahout. *A l'Heure des Statecharts et de XPC target pour la Commande d'une Montre Digitale.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 11, 2012

▶ E. Chanthery, G. Le Corre et P.-E. Hladik. *De l'illustration du guidage à l'optimisation d'un plan par un robot Lego Mindstorms NXT.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 15, Novembre 2016

Bibliographie

[Alami *et al.* 1998] R. Alami, R. Chatila, S. Fleury, M. Ghallab et F. Ingrand. *An Architecture for Autonomy.* International Journal of Robotics Research, vol. 17, no. 4, pages 315–337, 1998. (Cité en pages 27, 31 et 32.)

[An *et al.* 2015] D. An, N. H. Kim et J.-H. Choi. *Practical options for selecting data-driven or physics-based prognostics algorithms with reviews.* Reliability Engineering & System Safety, vol. 133, pages 223–236, 2015. (Cité en page 120.)

[Antsaklis & Passino 1989] P.J. Antsaklis et K.M. Passino. *Towards Intelligent Autonomous Control Systems : Architecture and Fundamental Issues.* Journal of Intelligent and Robotic Systems, pages 315–342, 1989. (Cité en pages 27 et 31.)

[Armengol *et al.* 2009] J. Armengol, A. Bregón, T. Escobet, E. Gelso, M. Krysander, M. Nyberg, X. Olive, B. Pulido et L. Travé-Massuyès. *Minimal Structurally Overdetermined sets for residual generation : A comparison of alternative approaches.* In Proceedings of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, SAFEPROCESS09, vol. 42(8) pp. 1480-1485, 2009. (Cité en page 92.)

[Bagajewicz *et al.* 2004] M. Bagajewicz, A. Fuxman et A. Uribe. *Instrumentation network design and upgrade for process monitoring and fault detection.* AIChE Journal, vol. 50(8), pages 1870–1880, 2004. (Cité en pages 39 et 113.)

[Balaban *et al.* 2013] E. Balaban, S. Narasimhan, M. J. Daigle, I. Roychoudhury, A. Sweet, C. Bond, J. R. Celaya et G. Gorospe. *Development of a Mobile Robot Test Platform and Methods for Validation of Prognostics-Enabled Decision Making Algorithms.* Int. Journal of Prognostics and Health Management, vol. 4, no. 006, 2013. (Cité en page 66.)

[Barbier & Chanthery 2004] M. Barbier et E. Chanthery. *Autonomous Mission Management for Unmanned Aerial Vehicles.* Aerospace Science and Technology, vol. 8, pages 359–368, 2004. (Non cité.)

[Barbier *et al.* 2006] M. Barbier, J-F Gabard, D. Vizcaino et O. Bonnet-Torrès. *ProCoSA : a software package for autonomous system supervision.* In First National Workshop on Control Architectures of Robots, 2006. (Cité en page 21.)

[Barbosa Roa 2016] N. A. Barbosa Roa. *A data-based approach for dynamic classification of functional scenarios oriented to industrial process plants.* Theses, Université Paul Sabatier - Toulouse III, Décembre 2016. (Cité en page 119.)

[Barbosa *et al.* 2017] N. A. Barbosa, L. Travé-Massuyès et V. H. Grisales. *Diagnosability improvement of dynamic clustering through automatic learning of discrete event models.* IFAC-PapersOnLine, vol. 50, no. 1, pages 1037–1042, 2017. (Cité en page 119.)

[Basile *et al.* 2009] F. Basile, P. Chiacchio et G. De Tommasi. *An efficient approach for online diagnosis of discrete event systems.* IEEE Transactions on Automatic Control, vol. 54, no. 4, pages 748–759, 2009. (Cité en page 38.)

[Bayoudh & Travé-Massuyès 2014] M. Bayoudh et L. Travé-Massuyès. *Diagnosability analysis of hybrid systems cast in a discrete-event framework*. Discrete Event Dynamic Systems, vol. 24, no. 3, pages 309–338, 2014. (Cité en page 121.)

[Bayoudh *et al.* 2008] M. Bayoudh, L. Travé-Massuyès et X. Olive. *Towards Active Diagnosis of Hybrid Systems*. In DX'08, 2008. (Cité en pages 46 et 73.)

[Benazera & Chanthery 2008] E. Benazera et E. Chanthery. *The Challenge of Solving POMDPs for Control, Monitoring and Repair of Complex Systems*. In Proceedings of the 19th International Workshop on Principles of Diagnosis (DX'08), 2008. (Cité en page 84.)

[Benedettini *et al.* 2009] O. Benedettini, T.S. Baines, H.W. Lightfoot et R.M. Greenough. *State-of-the-art in integrated vehicle health management*. Proceedings of the Institution of Mechanical Engineers, Part G : Journal of Aerospace Engineering, vol. 223, no. 2, pages 157–170, 2009. (Cité en page 31.)

[Berenjii & Wang 2006] H.R. Berenjii et Y. Wang. *Case-Based Reasoning for Fault Diagnosis and Prognosis*. In IEEE International Conference on Fuzzy Systems, Canada, 2006. (Cité en page 31.)

[Biswas *et al.* 2007] S. Biswas, D. Sarkar, P. Bhowal et S. Mukhopadhyay. *Diagnosis of delay–deadline failures in real time discrete event models*. ISA transactions, vol. 46, no. 4, pages 569–582, 2007. (Cité en page 38.)

[Blanke *et al.* 2006] M. Blanke, M. Kinnaert, J. Lunze et M. Staroswiecki. Diagnosis and fault-tolerant control. Springer-Verlag Berlin Heidelberg, 2006. (Cité en pages 89 et 91.)

[Boem *et al.* 2015] F. Boem, R. MG Ferrari, T. Parisini et M. M Polycarpou. *Optimal Topology for Distributed Fault Detection of Large-scale Systems*. IFAC-PapersOnLine, vol. 48, no. 21, pages 60–65, 2015. (Cité en page 121.)

[Bonasso *et al.* 1995] R.P. Bonasso, J. Firby, E. Gat, D. Kortenkamp, D.P. Miller et Slack M.G. *Experiences with an Architecture for Intelligent, Reactive Agents*. Journal of Experimental & Theoretical Artificial Intelligence, vol. 9, no. 2/3, 1995. (Cité en page 31.)

[Bondy & Murty 1976] J. A. Bondy et U. S. R. Murty. Graph theory with applications, volume 290. Macmillan London, 1976. (Cité en page 107.)

[Bonet & Geffner 2000] B. Bonet et H. Geffner. *Planning with incomplete information as heuristic search in belief space*. In Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems, pages 52–61. AAAI Press, 2000. (Cité en page 75.)

[Bregon & Daigle 2016] A. Bregon et I. Daigle M.and Roychoudhury. *Qualitative fault isolation of hybrid systems : A structural model decomposition-based approach*. In Third european conference of the PHM society, 2016. (Cité en page 122.)

[Brodie *et al.* 2003] M. Brodie, I. Rish, S. Ma, N. Odintsova et A. Beygelzimer. *Active probing strategies for problem diagnosis in distributed systems*. In IJCAI, volume 3, pages 1337–1338, 2003. (Cité en page 123.)

[Brotherton *et al.* 2000] T. Brotherton, G. Jahns, J. Jacobs et D. Wroblewski. *Prognosis of Faults in Gas Turbine Engines*. In IEEE Aerospace Conference Proceedings, volume 6, pages 163–171, USA, 2000. (Cité en page 26.)

[Cabasino *et al.* 2011] M. P. Cabasino, A. Giua, A. Paoli et C. Seatzu. *Decentralized diagnosability analysis of discrete event systems using Petri nets*. IFAC Proceedings Volumes, vol. 44, no. 1, pages 6060–6066, 2011. (Cité en page 121.)

[Camci *et al.* 2007] F. Camci, G. Valentine et K. Navarra. *Methodologies for Integration of PHM Systems with Maintenance Data.* In IEEE Aerospace Conference Proceedings, 2007. (Cité en page 31.)

[Campbell *et al.* 2002] S. L Campbell, K. G Horton et R. Nikoukhah. *Auxiliary signal design for rapid multi-model identification using optimization.* Automatica, vol. 38, no. 8, pages 1313–1325, 2002. (Cité en page 123.)

[Cardoso *et al.* 1999] J. Cardoso, R. Valette et D. Dubois. *Possibilistic Petri nets.* IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 29, no. 5, pages 573–582, 1999. (Cité en page 55.)

[Casillas *et al.* 2013] M. V. Casillas, V. Puig, L. E. Garza-Castanón et A. Rosich. *Optimal sensor placement for leak location in water distribution networks using genetic algorithms.* Sensors, vol. 13, no. 11, pages 14984–15005, 2013. (Cité en page 39.)

[Cassar & Staroswiecki 1997] J. Cassar et M. Staroswiecki. *A structural approach for the design of failure detection and identification systems.* In IFAC Conference on Control of Industrial Systems, 1997. (Cité en page 89.)

[Chanthery & Barbier 2003] E. Chanthery et M. Barbier. *Functional Modules for Intermixed Planning and Execution of an Observation Mission.* In 18th Bristol UAV Systems Conference, April 2003. (Non cité.)

[Chanthery & Pencolé 2009] E. Chanthery et Y. Pencolé. *Modélisation et intégration du diagnostic actif dans une architecture embarquée.* Journal européen des systèmes automatisés, vol. 43, no. 7-9, pages 789–803, 2009. (Cité en pages 75 et 78.)

[Chanthery & Pencolé 2009a] E. Chanthery et Y. Pencolé. *Monitoring and active diagnosis for discrete-event systems.* IFAC Proceedings Volumes, vol. 42, no. 8, pages 1545–1550, 2009. (Cité en page 73.)

[Chanthery & Pencolé 2009b] E. Chanthery et Y. Pencolé. *Principles of self-maintenance in an on-board architecture including active diagnosis.* Self-* and Autonomous Systems : reasoning and integration challenges (SAS-09), page 43, 2009. (Non cité.)

[Chanthery & Ribot 2013] E. Chanthery et P. Ribot. *An Integrated Framework for Diagnosis and Prognosis of Hybrid Systems.* In 3^{rd} Workshop on Hybrid Autonomous System, Italy, 2013. (Cité en page 47.)

[Chanthery *et al.* 2004a] E. Chanthery, M. Barbier et J.-L. Farges. *Integration of Mission Planning and Flight Scheduling for Unmanned Aerial Vehicles.* In ECAI'04 - Workshop on "Planning and Scheduling : Bridging Theory to Practice", Valencia, Spain, August 22-23 2004. (Non cité.)

[Chanthery *et al.* 2004b] E. Chanthery, M. Barbier et J.-L. Farges. *Mission Planning for autonomous Aerial Vehicles.* In IAV2004 - 5th IFAC Symposium on Intelligent Autonomous Vehicles, 2004. (Non cité.)

[Chanthery *et al.* 2005a] E. Chanthery, M. Barbier et J-L. Farges. *Planning algorithms for autonomous aerial vehicle.* In 16th IFAC World Congress, volume 16, 2005. (Non cité.)

[Chanthery *et al.* 2005b] E. Chanthery, M. Barbier et J-L. Farges. Planning, scheduling and constraint satisfaction : from theory to practice, chapitre Integration of Mission Planning and Flight Scheduling for Unmanned Aerial Vehicles. IOS Press, 2005. (Cité en pages 71 et 73.)

[Chanthery *et al.* 2010a] E. Chanthery, O. Buffet, M. Gaudichaud, M. Contat et I. Jauer. *Rapport sur la modélisation et la formalisation du problème d'optimisation de l'emploi des capteurs.* Rapport de contrat : EADS DOPEC-0050 édition 1.0, 26 mars 2010, 76p. , no 10412, EADS ; LORIA ; DISCO, 2010. (Cité en page 85.)

[Chanthery *et al.* 2010b] E. Chanthery, M. Gaudichaud, O. Buffet, I. Jauer et T. Oms. *Rapport d'étude théoriques appliquées au problème d'OPEC.* Rapport de contrat : EADS DOPEC-0050 édition 1.0, 26 mars 2010, 76p. , no 10412, EADS ; LORIA ; DISCO, 2010. (Cité en page 85.)

[Chanthery *et al.* 2010c] E. Chanthery, Y. Pencolé et N. Bussac. *An AO*-like algorithm implementation for active diagnosis.* In 10th International Symposium on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS, 2010. (Cité en pages 75, 76 et 77.)

[Chanthery *et al.* 2014] E. Chanthery, B. Delandrea, R. De Ferluc, N. Garin et L. Travé-Massuyès. *Diagnostic actif par OBCP. WP1000 : analyse de la problématique.* Rapport de contrat : Thales alenia space. diagnostic embarqué par OBCP, no. 14368, Thalès Alenia Space ; LAAS-CNRS, DISCO, 2014. (Cité en page 80.)

[Chanthery *et al.* 2015] E. Chanthery, Y. Pencolé, P. Ribot et L. Travé-Massuyès. *HYDIAG : extended diagnosis and prognosis for hybrid systems.* In the 26th International Workshop on Principles of Diagnosis (DX-2015), 2015. (Cité en page 78.)

[Chanthery *et al.* 2016a] E. Chanthery, G. Le Corre et P.-E. Hladik. *De l'illustration du guidage à l'optimisation d'un plan par un robot Lego Mindstorms NXT.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 15, Novembre 2016. (Non cité.)

[Chanthery *et al.* 2016b] E. Chanthery, L. Travé-Massuyès et S. Indra. *Fault isolation on request based on decentralized residual generation.* IEEE Transactions on Systems, Man, and Cybernetics : Systems, vol. 46, no. 5, pages 598–610, 2016. (Cité en pages 37, 96, 97 et 101.)

[Chanthery 2005] E. Chanthery. *Planification de mission pour un véhicule aérien autonome.* PhD thesis, École nationale superieure de l'aeronautique et de l'espace (ISAE), Toulouse, France, 2005. (Cité en page 30.)

[Chemweno *et al.* 2018] P. Chemweno, L. Pintelon, P. N. Muchiri et A. Van Horenbeek. *Risk assessment methodologies in maintenance decision making : A review of dependability modelling approaches.* Reliability Engineering & System Safety, vol. 173, pages 64–77, 2018. (Cité en page 124.)

[Chen & Kumar 2014] J. Chen et R. Kumar. *Failure prognosability of stochastic discrete event systems.* In American Control Conference (ACC), 2014, pages 2041–2046. IEEE, 2014. (Cité en page 121.)

[Cobb *et al.* 1999] P. Cobb, E. S Yager et C. Jacobus. *Anytime diagnosis using model-based methods for satellite diagnostics.* Ann Arbor, vol. 1001, page 48108, 1999. (Cité en page 82.)

[Cocquempot *et al.* 1998] V. Cocquempot, R. Izadi-Zamanabadi, M. Staroswiecki et M. Blanke. *Residual generation for the ship benchmark using structural approach.* In UKACC International Conference on Control (CONTROL 98), vol. 2, pp. 1480 - 1485, 1998. (Cité en page 92.)

[Cordier & Grastien 2007] M.-O. Cordier et A. Grastien. *Exploiting independence in a decentralised and incremental approach of diagnosis.* In 20th International Joint Conference on Artificial Intelligence, pp. 292-297, 2007. (Cité en page 121.)

[Daigle *et al.* 2012] M. Daigle, A. Bregon et I. Roychoudhury. *A distributed approach to system-level prognostics.* In Annual Conference of the Prognostics and Health Management Society, 2012. (Cité en page 122.)

[Daigle *et al.* 2014a] M. Daigle, C. S Kulkarni et G. Gorospe. *Application of model-based prognostics to a pneumatic valves testbed.* In IEEE Aerospace Conference, pages 1–8, 2014. (Cité en page 61.)

[Daigle *et al.* 2014b] M. Daigle, I. Roychoudhury et A. Bregon. *Integrated Diagnostics and Prognostics for the Electrical Power System of a Planetary Rover.* In Annual Conf. of the PHM Society, USA, 2014. (Cité en page 66.)

[Daigle *et al.* 2015a] M. Daigle, A. Bregon et I. Roychoudhury. *A Structural Model Decomposition Framework for Hybrid Systems Diagnosis.* In 26^{th} Int. Workshop on Principles of Diagnosis, 2015. (Cité en page 121.)

[Daigle *et al.* 2015b] M. Daigle, I. Roychoudhury et A. Bregon. *Model-based Prognostics of Hybrid Systems.* In Annual Conf. of the PHM Society, USA, 2015. (Cité en pages 31 et 61.)

[Daigle *et al.* 2015c] M. Daigle, S. Sankararaman et C. S. Kulkarni. *Stochastic Prediction of Remaining Driving Time and Distance for a Planetary Rover.* In IEEE Aerospace Conf., 2015. (Cité en page 62.)

[David & Alla 2005] R. David et H. Alla. Discrete, Continuous, and Hybrid Petri Nets. Springer, 2005. (Cité en page 52.)

[de Mortain *et al.* 2015] F. de Mortain, A. Subias, L. Travé-Massuyès et V. de Flaugergues. *Towards Active Diagnosis of Hybrid Systems leveraging Multimodel Identification and a Markov Decision Process.* IFAC-PapersOnLine, vol. 48, no. 21, pages 171–176, 2015. (Cité en page 123.)

[Deb 2014] K. Deb. *Multi-objective optimization.* In Search methodologies, pages 403–449. Springer, 2014. (Cité en page 121.)

[Debouk *et al.* 2000] R. Debouk, S. Lafortune et D. Teneketzis. *Coordinated Decentralized Protocols for Failure Diagnosis of Discrete Event Systems.* Discrete Event Dynamic Systems, vol. 10(1), pp. 33-86, 2000. (Cité en page 121.)

[Delandrea *et al.* 2014] B. Delandrea, C. Le Peuvedic, R. De Ferluc, E. Chanthery, L. Travé-Massuyès et N. Garin. *R&T CNES diagnostique actif par OBCP. Rapport de Lot 2 : analyse et solutions de diagnostic actif à base d'OBCP.* Rapport de contrat : Thales alenia space. diagnostic embarqué par OBCP, no. 14691, Thalès Alenia Space ; LAAS-CNRS, DISCO, 2014. (Cité en page 80.)

[Dick & Faivre 1993] J. Dick et A. Faivre. *Automating the generation and sequencing of test cases from model-based specifications.* In FME'93 : Industrial-Strength Formal Methods, pages 268–284. Springer, 1993. (Cité en pages 39 et 106.)

[Ding 2014] S. X. Ding. Data-driven Design of Fault Diagnosis and Fault-tolerant Control Systems. Springer, 2014. (Cité en page 55.)

[Dotoli *et al.* 2009] M. Dotoli, M. P. Fanti, A. M. Mangini et W. Ukovich. *On-line fault detection in discrete event systems by Petri nets and integer linear programming.* Automatica, vol. 45, no. 11, pages 2665–2672, 2009. (Cité en page 38.)

[Douc & Cappé 2005] R. Douc et O. Cappé. *Comparison of resampling schemes for particle filtering.* In Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on, pages 64–69. IEEE, 2005. (Cité en page 58.)

[Doucet & Johansen 2009] A. Doucet et A. M Johansen. *A tutorial on particle filtering and smoothing : Fifteen years later.* Handbook of nonlinear filtering, vol. 12, no. 656-704, page 3, 2009. (Cité en page 58.)

[Dulmage & Mendelsohn 1958] A. L. Dulmage et N. S. Mendelsohn. *Coverings of bipartite graphs.* Canadian Journal of Mathematics, vol. 10, pp. 517-534, 1958. (Cité en pages 90 et 91.)

[Durst & Gray 2014] P. J. Durst et W. Gray. *Levels of Autonomy and Autonomous System Performance Assessment for Intelligent Unmanned Systems.* Rapport technique, Engineer Research and Development Center Vicksburg Ms Geotechnical and Structures Lab, 2014. (Cité en page 27.)

[Düstegör *et al.* 2006] D. Düstegör, E. Frisk, V. Cocquempot, M. Krysander et M. Staroswiecki. *Structural analysis of fault isolability in the DAMADICS benchmark.* Control Engineering Practice, 2006. (Cité en page 89.)

[Engel *et al.* 2000] S.J. Engel, B.J. Gilmartin, K. Bongort et A. Hess. *Prognostics, the real issues involved with predicting life remaining.* In Aerospace Conf., IEEE, volume 6, pages 457–469, 2000. (Cité en page 26.)

[Feng *et al.* 2016] W. Feng, R. Qin, W. Zhang et Q. Zhao. *A possible conflicts based distributed diagnosis method for hybrid system.* In Prognostics and System Health Management Conference (PHM-Chengdu), 2016, pages 1–6. IEEE, 2016. (Cité en page 122.)

[Ferdowsi & Jagannathan 2017] H. Ferdowsi et S. Jagannathan. *Decentralized Fault Diagnosis and Prognosis Scheme for Interconnected Nonlinear Discrete-Time Systems.* International Journal of Prognostics and Health Management, vol. 8, no. 009, 2017. (Cité en page 122.)

[Ferluc *et al.* 2014] R. De Ferluc, E. Chanthery et L. Travé-Massuyès. *Diagnostic actif par OBCP. WP4000 : prototypage de diagnostic actif à base d'OBCP.* Rapport de contrat : Thales alenia space. diagnostic embarqué par obcp, no. 5461, Thalès Alenia Space ; LAAS-CNRS, DISCO, 2014. (Cité en page 80.)

[Franceschelli *et al.* 2009] M. Franceschelli, A. Giua et C. Seatzu. *Decentralized fault diagnosis for sensor networks.* In Automation Science and Engineering, 2009. CASE 2009. IEEE International Conference on, pages 334–339. IEEE, 2009. (Cité en page 123.)

[Frank & Wolfe 1956] M. Frank et P. Wolfe. An algorithm for quadratic programming, volume 3. Naval Research Logistic Quaterly, 1956. (Cité en page 21.)

[Gaudel *et al.* 2014a] Q. Gaudel, E. Chanthery et P. Ribot. *Health Monitoring of Hybrid Systems Using Hybrid Particle Petri Nets.* In Annual Conf. of the PHM Society, USA, 2014. (Cité en page 48.)

[Gaudel *et al.* 2014b] Q. Gaudel, E. Chanthery, P. Ribot et E. Le Corronc. *Hybrid systems Diagnosis using modified particle Petri nets.* In 25^{th} Int. Workshop on Principles of Diagnosis, Austria, 2014. (Cité en page 48.)

[Gaudel *et al.* 2015a] Q. Gaudel, E. Chanthery et P. Ribot. *Hybrid Particle Petri Nets for Systems Health Monitoring under Uncertainty.* Int. Journal of Prognostics and Health Management, vol. 6, no. 022, 2015. (Cité en pages 48 et 61.)

[Gaudel *et al.* 2015b] Q. Gaudel, P. Ribot et E. Chanthery. *Vers une architecture de surveillance de santé d'un système hybride sous incertitudes.* In Modélisation des Systèmes Réactifs, France, 2015. (Cité en pages 46 et 61.)

[Gaudel *et al.* 2016] Q. Gaudel, P. Ribot, E. Chanthery et M. J. Daigle. Health Monitoring of a Planetary Rover Using Hybrid Particle Petri Nets, volume 9698 of *Lecture Notes in Computer Science*, chapitre Application and Theory of Petri Nets and Concurrency. PETRI NETS 2016. Lecture Notes in Computer Science, pages 196–215. Springer, Cham ; Kordon F., Moldt D. (eds), 2016. (Cité en pages 61 et 66.)

[Gaudel *et al.* 2018a] Q. Gaudel, E. Chanthery, P. Ribot et M. J. Daigle. Fault diagnosis of hybrid dynamic and complex systems, chapitre Diagnosis of Hybrid Systems Using Hybrid Particle Petri Nets : Theory and Application on a Planetary Rover, pages 209–241. Springer, 2018. (Non cité.)

[Gaudel *et al.* 2018b] Q. Gaudel, P. Ribot, E. Chanthery et M. J. Daigle. *Prognosis of a Planetary Rover Using Hybrid Particle Petri Nets.* Journal of Process Control, soumis en 2018. (Non cité.)

[Gaudel 2016] Q. Gaudel. *Integrated approach of diagnosis and prognosis for hybrid system health management under uncertainty.* Theses, INSA de Toulouse, Septembre 2016. (Cité en pages 30, 47, 48, 53 et 122.)

[Genc & Lafortune 2009] S. Genc et S. Lafortune. *Predictability of event occurrences in partially-observed discrete-event systems.* Automatica, vol. 45, no. 2, pages 301–311, 2009. (Cité en page 121.)

[Gertler 2015] J. Gertler. Fault detection and diagnosis. Springer, 2015. (Cité en page 120.)

[Ghallab *et al.* 2001] M. Ghallab, F. Ingrand, S. Lemai et F. Py. *Architecture and tools for autonomy in space.* ISAIRAS, Montreal, 2001. (Cité en pages 30 et 31.)

[Godichaud *et al.* 2011a] M. Godichaud, E. Chanthery, O. Buffet et M. Contat. *Formalisation et résolution de problèmes d'acquisition d'informations par des systèmes autonomes.* In ROADEF (12e congrès annuel de la Société française de Recherche Opérationnelle et d'Aide à la Décision), Saint-Etienne, France, 2011. (Cité en page 85.)

[Godichaud *et al.* 2011b] M. Godichaud, E. Chanthery, O. Buffet et M. Contat. *Formalizing and Solving Information Collection Problems with Autonomous Sensor Systems.* IFAC Proceedings Volumes, vol. 44, no. 1, pages 2208–2213, 2011. (Cité en page 85.)

[Goh *et al.* 2006] K. Goh, B. Tjahjono, T. Baines et S. Subramaniam. *A Review of Research in Manufacturing Prognostics.* In Proceedings of the IEEE International Conference on Industrial Informatics, pages 417–422, USA, 2006. (Cité en page 26.)

[Grabowski 2015] R. Grabowski. *Big Picture for Autonomy Research in DoD.* In Soft and Secure Systems and Software Symposium, 2015. (Cité en page 27.)

[Grastien *et al.* 2009] A. Grastien, A. Anbulagan*et al. Incremental diagnosis of DES with a non-exhaustive diagnosis engine.* Proceedings of the 20th International Workshop on Principles of Diagnosis (DX 2009), 2009. (Cité en page 38.)

[Habets & van Schuppen 2005] LCGJM Habets et J. H van Schuppen. *Control to facet problems for affine systems on simplices and polytopes–with applications to control of hybrid systems.* In Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on, pages 4175–4180. IEEE, 2005. (Cité en page 123.)

[Hamilton *et al.* 2007] K. Hamilton, D.M. Lane, K.E. Brown, J. Evans et N.K. Taylor. *An Integrated Diagnostic Architecture for Autonomous Underwater Vehicles.* Journal of Field Robotics, vol. 6, no. 24, pages 497–526, 2007. (Cité en page 31.)

[Hart *et al.* 1968] P.E. Hart, N.J. Nilsson et B. Raphael. *A formal basis for the heuristic determination of minimum cost paths.* IEEE Transactions on Systems Science and Cybernetics, vol. 4, pages 100–107, 1968. (Cité en page 107.)

[Henzinger 1996] T. Henzinger. *The theory of hybrid automata.* In 11^{th} Annual IEEE Symposium on Logic in Computer Science, pages 278–292, 1996. (Cité en page 46.)

[Hu *et al.* 2012] J. Hu, L. Zhang et W. Liang. *Opportunistic predictive maintenance for complex multi-component systems based on DBN-HAZOP model.* Process Safety and Environmental Protection, vol. 90, no. 5, pages 376–388, 2012. (Cité en page 124.)

[Huang *et al.* 2004] H.-M. Huang, E. Messina, R. Wade, R. English, B. Novak et J. Albus. *Autonomy measures for robots.* In Proceedings of the 2004 ASME International Mechanical Engineering Congress & Exposition, Anaheim, California, pages 1–7, 2004. (Cité en page 27.)

[Huang *et al.* 2005] H.-M. Huang, K. Pavek, J. Albus et E. Messina. *Autonomy levels for unmanned systems (alfus) framework : An update.* In Proceedings of the 2005 SPIE Defense and Security Symposium, volume 27, pages 439–448, 2005. (Cité en page 28.)

[Indra & Travé-Massuyès 2013] S. Indra et L. Travé-Massuyès. *Spacecraft fault detection and isolation system design using decentralized analytical redundancy.* In Advances in Aerospace Guidance, Navigation and Control, pages 247–263. Springer, 2013. (Cité en page 101.)

[Indra *et al.* 2011] S. Indra, L. Travé-Massuyès et E. Chanthery. *A decentralized FDI scheme for spacecraft : Bridging the gap between model based FDI research and practice.* Progress in Flight Dynamics, Guidance, Navigation, Control, Fault Detection, and Avionics, vol. 6, 2011. also in "4th European Conference for Aerospace Sciences, St Petersburg, 2011, Russia". (Non cité.)

[Indra *et al.* 2012] S. Indra, L. Travé-Massuyès et E. Chanthery. *Decentralized diagnosis with isolation on request for spacecraft.* IFAC Proceedings Volumes, vol. 45, no. 20, pages 283–288, 2012. (Non cité.)

[Isermann 1997] R. Isermann. *Supervision, fault-detection and fault-diagnosis methods. An introduction.* Control Engineering Practice, vol. 5, pages 639–652, 1997. (Cité en pages 25 et 125.)

[Isermann 2011] R. Isermann. Fault-diagnosis applications. Springer Heidelberg Dordrecht London New York, 2011. (Cité en page 34.)

[Jauberthie & Chanthery 2013] C. Jauberthie et E. Chanthery. *Optimal input design for a nonlinear dynamical uncertain aerospace system.* IFAC Proceedings Volumes, vol. 46, no. 23, pages 469–474, 2013. (Cité en page 85.)

[Jouin *et al.* 2016] M. Jouin, R. Gouriveau, D. Hissel, M.C. Péra et N. Zerhouni. *Joint Particle Filters Prognostics for Proton Exchange Membrane Fuel Cell Power Prediction at Constant Current Solicitation.* IEEE Transactions On Reliability, vol. 65, no. 1, pages 336–349, 2016. (Cité en page 61.)

[Kallesoe *et al.* 2006] C. S. Kallesoe, V. Cocquempot et R. Izadi-Zamanabadi. *Model based fault detection in a centrifugal pump application.* IEEE transactions on control systems technology, vol. 14, no. 2, pages 204–215, 2006. (Cité en page 89.)

[Kempowsky 2004] Tatiana Kempowsky. *Surveillance de procédés à base de méthodes de classification : conception d'un outil d'aide pour la détection et le diagnostic des défaillances.* Theses, INSA de Toulouse, Décembre 2004. (Cité en page 119.)

[Khelassi *et al.* 2011] A. Khelassi, D. Theilliol, P. Weber et J.-C. Ponsart. *Fault-tolerant control design with respect to actuator health degradation : An LMI approach.* In Control Applications (CCA), 2011 IEEE International Conference on, pages 983–988. IEEE, 2011. (Cité en page 124.)

[Khorasgani & Biswas 2017] H. Khorasgani et G. Biswas. *Structural Fault Detection and Isolation in Hybrid Systems.* IEEE Transactions on Automation Science and Engineering, 2017. (Cité en page 122.)

[Khorasgani *et al.* 2015] H. Khorasgani, D. Jung et G. Biswas. *Structural Approach for Distributed Fault Detection and Isolation.* In 9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS 2015, vol. 48(21), pp. 72-77, 2015. (Cité en pages 90, 102, 104 et 108.)

[Korbicz *et al.* 2012] J. Korbicz, J. M. Kościelny, Z. Kowalczuk et W. Cholewa, editeurs. Fault diagnosis : Models, artificial intelligence, applications. Springer, 2012. (Cité en page 34.)

[Korf 2010] R. E. Korf. Algorithms and theory of computation handbook. Chapman & Hall/CRC, 2010. (Cité en pages 107 et 108.)

[Krysander & Frisk 2008] M. Krysander et E. Frisk. *Sensor Placement for Fault Diagnosis.* IEEE Trans. Syst. Man Cy. A., Vol 38(6)., 2008. (Cité en page 106.)

[Krysander *et al.* 2010] M. Krysander, J. Aslund et E. Frisk. *A Structural Algorithm for Finding Testable Sub-models and Multiple Fault Isolability Analysis.* In 21st International Workshop on the Principles of Diagnosis, 2010. (Cité en pages 89, 92 et 93.)

[Kuhn *et al.* 2008] L. Kuhn, B. Price, J. de Kleer, M. B. Do et R. Zhou. *Pervasive Diagnosis : The Integration of Diagnostic Goals into Production Plans.* In AAAI 2008, pages 1306–1312, 2008. (Cité en page 73.)

[Kumar & Takai 2010] R. Kumar et S. Takai. *Decentralized prognosis of failures in discrete event systems.* IEEE Transactions on Automatic Control, vol. 55, no. 1, pages 48–59, 2010. (Cité en pages 121 et 122.)

[Kwong & Yonge-Mallo 2011] R. H Kwong et D. L Yonge-Mallo. *Fault diagnosis in discrete-event systems : Incomplete models and learning.* IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 41, no. 1, pages 118–130, 2011. (Cité en page 119.)

[Lachat *et al.* 2006] D. Lachat, A. Krebs, T. Thueer et R. Siegwart. *Antarctica Rover Design And Optimization For Limited Power Consumption.* In 4^{th} IFAC Symposium on Mechatronic Systems, 2006. (Cité en page 66.)

[Lampe 2006] A. Lampe. *Méthodologie d'évaluation du degré d'autonomie d'un robot mobile terrestre.* PhD thesis, Institut National Polytechnique de Toulouse-INPT, 2006. (Cité en page 27.)

[Langeron *et al.* 2017] Y. Langeron, A. Grall et A. Barros. *Joint maintenance and controller reconfiguration policy for a gradually deteriorating control system.* Proceedings of the Institution of Mechanical Engineers, Part O : Journal of Risk and Reliability, vol. 231, no. 4, pages 339–349, 2017. (Cité en pages 30 et 124.)

[Le *et al.* 2014] T. T. Le, F. Chatelain et C. Bérenguer. *Hidden Markov Models for diagnostics and prognostics of systems under multiple deterioration modes.* In European Safety and Reliability Conference (ESREL 2014), pages 1197–1204, Wroclaw, Poland, Septembre 2014. Taylor & Francis - CRC Press/Balkema. (Cité en page 120.)

[Leal *et al.* 2015] R. Leal, J. Aguilar, L. Travé-Massuyès, E. Camargo et A. Ríos-Bolivar. *An Approach for Diagnosability Analysis and Sensor Placement for Continuous Processes Based on Evolutionary Algorithms and Analytical Redundancy.* Applied Mathematical Sciences, Vol. 9, 2015, no. 43, 2125 - 2146, 2015. (Cité en page 106.)

[Lebold *et al.* 2002] M. Lebold, K. Reichard, C. S Byington et R. Orsagh. *OSA-CBM architecture development with emphasis on XML implementations.* In Maintenance and Reliability Conference (MARCON), pages 6–8, 2002. (Cité en page 34.)

[Leclercq *et al.* 2008] E. Leclercq, S. O. el Medhi et D. Lefebvre. *Petri nets design based on neural networks.* In ESANN, pages 529–534, 2008. (Cité en page 119.)

[Lei *et al.* 2017] B. Lei, G. Xu, M. Feng, F. van der Heijden, Y. Zou, D. de Ridder et D. MJ Tax. Classification, parameter estimation and state estimation : an engineering approach using matlab. John Wiley & Sons, 2017. (Cité en page 120.)

[Lesire & Tessier 2005] C. Lesire et C. Tessier. *Particle Petri nets for aircraft procedure monitoring under uncertainty.* In Applications and Theory of Petri Nets, pages 329–348. Springer, 2005. (Cité en page 55.)

[Li *et al.* 2015] T. Li, M. Bolic et P. M. Djuric. *Resampling methods for particle filtering : classification, implementation, and strategies.* IEEE Signal processing magazine, vol. 32, no. 3, pages 70–86, 2015. (Cité en page 58.)

[Likhachev *et al.* 2005] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz et S. Thrun. *Anytime Dynamic A* : An Anytime, Replanning Algorithm.* In International Conference on Automated Planning and Scheduling (ICAPS), 2005. (Cité en page 81.)

[Löhr *et al.* 2012] A. Löhr, C. Haines et M. Buderath. *Data Management Backbone for Embedded and PC-based Systems Using OSA-CBM and OSA-EAI.* In AAIA Infotech@ Aerospace Conference, 2012. (Cité en page 34.)

[Louajri & Sayed-Mouchaweh 2014] H Louajri et M Sayed-Mouchaweh. *Decentralized diagnosis and diagnosability of a class of hybrid dynamic systems.* In Informatics in Control, Automation and Robotics (ICINCO), 2014 11th International Conference on, volume 2, pages 708–715. IEEE, 2014. (Cité en page 122.)

[Ma *et al.* 2015] L. Ma, H. Li et X. Lv. *Optimal Test Selection of Complex Electronic Systems Based on Improved Discrete Particle Swarm Optimization Algorithm.* In Proceedings of the Second International Conference on Mechatronics and Automatic Control, pages 549–557. Springer, 2015. (Cité en page 121.)

[Maiga *et al.* 2012] M. Maiga, E. Chanthery et L. Travé-Massuyès. *Hybrid system diagnosis : Test of the diagnoser HYDIAG on a benchmark of the international diagnostic competition DXC'2011.* IFAC Proceedings Volumes, vol. 45, no. 20, pages 271 – 276, 2012. 8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes. (Non cité.)

[Meier *et al.* 2014] F. Meier, P. Hennig et S. Schaal. *Efficient Bayesian Local Model Learning for Control.* In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, pages 2244 – 2249, 2014. clmc. (Cité en page 120.)

[Meyer 2011] P.-J. Meyer. Anytime diagnosis of discrete event systems. Master's thesis, ENSEEIHT, LAAS-CNRS, 2011. (Cité en page 83.)

[Moreira *et al.* 2011] M. V Moreira, T. C Jesus et J. C. Basilio. *Polynomial time verification of decentralized diagnosability of discrete event systems.* IEEE Transactions on Automatic Control, vol. 56, no. 7, pages 1679–1684, 2011. (Cité en page 121.)

[Mouaddib & Zilberstein 1995] A.-I. Mouaddib et S. Zilberstein. *Knowledge-based anytime computation.* In IJCAI, volume 95, pages 775–781, 1995. (Cité en page 82.)

[Murota 2000] K. Murota. Matrices and matroids for system analysis. Springer, 2000. (Cité en page 91.)

[Muscettola *et al.* 1998] N. Muscettola, P. Nayak, B. Pell et B. Williams. *Remote Agent : To Boldly Go Where No AI System Has Gone Before.* Artificial Intelligence, vol. 103, no. 1-2, pages 5–48, 1998. (Cité en pages 31 et 32.)

[Narasimhan *et al.* 2012] S. Narasimhan, E. Balaban, M. Daigle, I. Roychoudhury, A. Sweet, J. Celaya et K. Goebel. *Autonomous Decision Making for Planetary Rovers Using Diagnostic and Prognostic Information.* In 8^{th} IFAC Symposium on Fault Dectection, Supervision and Safety of Technical Processes, pages 289–294, Mexico, 2012. (Cité en pages 30, 31 et 34.)

[Nesnas *et al.* 2003] I.A. Nesnas, A. Wright, M. Bajracharya, R. Simmons, T. Estlin et Won Soo Kim. *CLARAty : An Architecture for Reusable Robotic Software.* In SPIE Aerosense Conference, 2003. (Cité en pages 30 et 31.)

[of Automotive Engineers 2014] Society of Automotive Engineers. *Taxonomy and Definitions for Terms Related to On-road Motor Vehicle Automated Driving Systems.* Rapport technique, Society of Automotive Engineers, 2014. (Cité en page 28.)

[Olive *et al.* 2003] X. Olive, L. Trave-Massuyes et H. Poulard. *AO* Variant Methods for Automatic Generation of Near-optimal Diagnosis Trees.* In DX'03, 2003. (Cité en page 71.)

[Olive *et al.* 2011] X. Olive, S. Clerc et D. Losa. *Smart architecture for highly available, robust and autonomous satellite.* In 18^{th} IFAC World Congress, 2011. (Cité en pages 31, 32 et 33.)

[Pattipati & Alexandridis 1988] K. R. Pattipati et M. G. Alexandridis. *Application of heuristic search and information theory to sequential fault diagnosis.* In Proceedings IEEE International Symposium on Intelligent Control, vol. 4, pp. 291-296, 1988. (Cité en page 106.)

[Pattipati & Dontamsetty 1992] K. R. Pattipati et M. Dontamsetty. *On a generalized test sequencing problem.* IEEE Trans. Syst. Man Cy. A. vol. 22(2), pp. 392-396, 1992. (Cité en pages 39, 71 et 106.)

[Patton *et al.* 2000] R. J. Patton, P. M. Frank et R. N. Clark, editeurs. Issues of fault diagnosis for dynamic systems. Springer Verlag London, 2000. (Cité en pages 34 et 89.)

[Pecover 2010] D. Pecover. *Functional verification of the ADM-AEOLUS autonomy requirements.* In Proc. SpaceOps Conf., 2010. (Cité en page 101.)

[Pencolé & Cordier 2005] Y. Pencolé et M.-O. Cordier. *A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks.* Artificial Intelligence, vol. 164(1-2) pp. 121-170, 2005. (Cité en page 121.)

[Pencolé *et al.* 2016] Y. Pencolé, E. Chanthery et T. Peynot. *Definition of Model-based diagnosis problems with Altarica.* In 27th International Workshop on Principles of Diagnosis (DX-2016), page 8p., Denver, CO, United States, Octobre 2016. (Non cité.)

[Pencolé 2004] Y. Pencolé. *Diagnosability analysis of distributed discrete event systems.* In Proceedings of the 16th European Conference on Artificial Intelligence, pages 38–42. IOS Press, 2004. (Cité en page 121.)

[Pérez *et al.* 2015] C. G. Pérez, L. Travé-Massuyès, E. Chanthery et J. Sotomayor. *Decentralized diagnosis in a spacecraft attitude determination and control system.* In Journal of Physics : Conference Series, volume 659-1. IOP Publishing, 2015. (Cité en pages 37, 93, 94 et 101.)

[Pérez *et al.* 2016] C. G. Pérez, E. Chanthery, L. Travé-Massuyès et J. Sotomayor. *Fault-Driven Minimal Structurally Overdetermined Set in a Distributed Context.* In the 27th International Workshop on Principles of Diagnosis : DX-2016, 2016. (Cité en pages 102 et 103.)

[Pérez *et al.* 2017] C. G. Pérez, L. Travé-Massuyès, E. Chanthery et J. Sotomayor. *Fault-Driven Structural Diagnosis Approach in a Distributed Context.* In IFAC World Congress, page 1, 2017. (Cité en page 97.)

[Pérez Zuniga 2017] C. G. Pérez Zuniga. *Structural analysis for the diagnosis of distributed systems.* Theses, Institut National des Sciences Appliquées de Toulouse, Août 2017. (Cité en pages 92, 105 et 110.)

[Raimondo *et al.* 2016] D. M Raimondo, F. Boem, A. Gallo et T. Parisini. *A decentralized fault-tolerant control scheme based on Active Fault Diagnosis.* In Decision and Control (CDC), 2016 IEEE 55th Conference on, pages 2164–2169. IEEE, 2016. (Cité en page 123.)

[Ramadge & Wonham 1989] P.J.G. Ramadge et W.M. Wonham. *The control of discrete event processes.* In IEEE Proc. : Special issue on Discrete Event Systems, 1989. (Cité en pages 72 et 73.)

[Rasovska *et al.* 2007] I. Rasovska, B. Chebel-Morello et N. Zerhouni. *Classification des différentes architectures en maintenance.* In 7ème Congrès International de Génie Industriel, GI'2007, Trois Rivières., pages sur–CD. UQTR, 2007. (Cité en page 34.)

[Ribot *et al.* 2013] P. Ribot, Y. Pencolé et M. Combacau. *Generic characterization of diagnosis and prognosis for complex heterogeneous systems.* Int. Journal of Prognostics and Health Management, vol. 4, no. 023, 2013. (Cité en pages 31 et 121.)

[Ribot *et al.* 2017] P. Ribot, E. Chanthery et Q. Gaudel. *HPPN-based Prognosis for Hybrid Systems.* In Annual Conference of the Prognostics and Health Management Society 2017, Proceedings of the Annual Conference of the Prognostics and Health Management Society 2017, St. Petersburg, United States, Octobre 2017. (Non cité.)

[Ribot 2009] P. Ribot. *Vers l'intégration diagnostic/pronostic pour la maintenance des systèmes complexes.* PhD thesis, Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier), Toulouse, France, 2009. (Cité en pages 26 et 31.)

[Rosich *et al.* 2007] A. Rosich, R. Sarrate, V. Puig et T. Escobet. *Efficient optimal sensor placement for model-based FDI using an incremental algorithm.* Proc. 46th IEEE Conference on Decision and Control, pages 2590-2595, 2007. (Cité en page 106.)

[Rosich *et al.* 2009] A. Rosich, R. Sarrate et F. Nejjari. *Optimal sensor placement for fdi using binary integer linear programming.* In 20th International Workshop on Principles of Diagnosis, pages 235–242, 2009. (Cité en pages 39 et 113.)

[Rouse 1978] W. B. Rouse. *Human problem solving performance in a fault diagnosis task.* IEEE Transactions on Systems, Man, and Cybernetics, vol. 8, no. 4, pages 258–271, 1978. (Cité en page 125.)

[Roychoudhury & Daigle 2011] I. Roychoudhury et M. Daigle. *An Integrated Model-Based Diagnostic and Prognostic Framework.* In 22^{nd} Int. Workshop on Principle of Diagnosis, Germany, 2011. (Cité en pages 31 et 34.)

[Said 2016] A. B. Said. *Gestion dynamique des connaissances de maintenance pour des environnements de production de haute technologie à fort mix produit.* PhD thesis, Université Grenoble Alpes, 2016. (Cité en page 34.)

[Salazar *et al.* 2017] J. C Salazar, P. Weber, F. Nejjari, R. Sarrate et D. Theilliol. *System reliability aware model predictive control framework.* Reliability Engineering & System Safety, vol. 167, pages 663–672, 2017. (Cité en pages 30 et 124.)

[Sampath *et al.* 1995] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen et D. Teneketzis. *Diagnosability of discrete-event systems.* IEEE Transactions on Automatic Control, vol. 40, no. 9, pages 1555–1575, 1995. (Cité en page 26.)

[Sampath *et al.* 1996] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen et D. Teneketzis. *Failure Diagnosis Using Discrete-Event Models.* IEEE Transactions on Control Systems Technology, vol. 4, no. 2, pages 105–124, 1996. (Cité en pages 31 et 81.)

[Sampath *et al.* 1998] M. Sampath, S. Lafortune et D. Teneketzis. *Active diagnosis of discrete-event systems.* IEEE Trans. on Automatic Control, vol. 43, 1998. (Cité en pages 72 et 124.)

[Sareni *et al.* 2009] B. Sareni, G. Fontan, E. Chanthery et S. Caux. *OrdoNet, un outil de modélisation et d'analyse des graphes potentiel-tâche sous Matlab.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 8, no. hors série 1, page 1003, 2009. (Non cité.)

[Sarrate *et al.* 2007a] R. Sarrate, V. Puig, T. Escobet et A. Rosich. *Optimal sensor placement for model-based fault detection and isolation.* In 46th IEEE Conference on Decision and Control, pages 2584–2589. IEEE, 2007. (Cité en page 39.)

[Sarrate *et al.* 2007b] R. Sarrate, V. Puig, T. Escobet et A. Rosich. *Optimal sensor placement for model-based fault detection and isolation.* Proc. 46th IEEE Conference on Decision and Control, pp. 2584-2589, 2007. (Cité en page 113.)

[Sarrate *et al.* 2012] R. Sarrate, F. Nejjari et A. Rosich. *Sensor placement for fault diagnosis performance maximization in distribution networks.* In 20th Mediterranean Conference on Control & Automation (MED), pages 110–115. IEEE, 2012. (Cité en pages 39 et 106.)

[Schumann *et al.* 2007] A. Schumann, Y. Pencolé, S. Thiébaux*et al. A spectrum of symbolic on-line diagnosis approaches.* In National Conference of Artificial Intelligence, volume 22-1, page 335. Menlo Park, CA ; Cambridge, MA ; London ; AAAI Press ; MIT Press ; 1999, 2007. (Cité en page 81.)

[Singleton *et al.* 2015] R. K. Singleton, E. G. Strangas et S. Aviyente. *Extended Kalman filtering for remaining-useful-life estimation of bearings.* IEEE Transactions on Industrial Electronics, vol. 62, no. 3, pages 1781–1790, 2015. (Cité en page 120.)

[Staroswiecki *et al.* 2009] M. Staroswiecki, J. Ragot, D. Henry, A. Zolghadri, J. Cieslak, D. Maquin, B. Marx, T. Raissi, R. Pons, C. Jauberthie, L. Travé-Massuyès, E. Benazera, E. Chanthery, D. Berdjag, C. Join, D. Theilliol, S. Canitrot, T. Hamel et F. Hamelin. *SIRASAS. Deliverable no.1. WP2.0.* Rapport de contrat : Projet SIRASAS - contrat FRAE, mars 2009, 289p. , no. 09149, DISCO ; IMS Bordeaux ; CRAN-ENSEM ; SATIE, 2009. (Cité en page 84.)

[Su *et al.* 2014] X. Su, A. Grastien et Y. Pencolé. *Window-based diagnostic algorithms for discrete event systems : what information to remember.* In 25th International Workshop on Principles of Diagnosis (DX-14), 2014. (Cité en page 38.)

[Subias *et al.* 2012] A. Subias, E. Chanthery, G. Le Corre, J. Martin et V. Mahout. *A l'Heure des Statecharts et de XPC target pour la Commande d'une Montre Digitale.* Journal sur l'enseignement des sciences et technologies de l'information et des systèmes, vol. 11, 2012. (Non cité.)

[Sweet *et al.* 2014] A. Sweet, G. Gorospe, M. Daigle, J. R. Celaya, E. Balaban, I. Roychoudhury et S. Narasimhan. *Demonstration of Prognostics-Enabled Decision Making Algorithms on a Hardware Mobile Robot Test Platform.* In Annual Conf. of the PHM Society, USA, 2014. (Cité en page 66.)

[Tabatabaeipour *et al.* 2009a] S. Tabatabaeipour, A. P Ravn, R. Izadi-Zamabadi et T. Bak. *Active diagnosis of hybrid systems - A model predictive approach.* In Control and Automation, 2009. ICCA 2009. IEEE International Conference on, pages 465–470. IEEE, 2009. (Cité en page 123.)

[Tabatabaeipour *et al.* 2009b] S. Tabatabaeipour, A. P Ravn, R. Izadi-Zamanabadi et T. Bak. *Active fault diagnosis of linear hybrid systems.* IFAC Proceedings Volumes, vol. 42, no. 8, pages 211–216, 2009. (Cité en page 123.)

[Thurston & Lebold 2001] M. Thurston et M. Lebold. *Standards Development For Condition-Based Maintenance Systems.* In New Frontiers in Integrated Diagnostics and Prognostics, 55 th Meeting of the Society for Machinery Failure Prevention Technology. Citeseer, 2001. (Cité en page 34.)

[Travé-Massuyès *et al.* 2006] L. Travé-Massuyès, T. Escobet et X. Olive. *Diagnosability Analysis Based on Component-Supported Analytical Redundancy Relations.* IEEE Trans. Syst., Man, Cybern. Part A : Sys and Humans, VOL. 36, NO. 6, PP. 1146-1160, Nov. 2006, 2006. (Cité en page 89.)

[van der Merwe *et al.* 2000] R. van der Merwe, A. Doucet, N. De Freitas et E. Wan. *The unscented particle filter.* In NIPS, volume 2000, pages 584–590, 2000. (Cité en page 55.)

[van Harmelen & ten Teije 1995] F. van Harmelen et A. ten Teije. *Approximations in diagnosis : motivations and techniques.* Proceedings of the Symposium on Abstraction, Reformulation and Approximation, 1995. (Cité en page 82.)

[Vassev & Hinchey 2013] E. Vassev et M. Hinchey. *On the autonomy requirements for space missions.* In IEEE 16th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC), pages 1–10. IEEE, 2013. (Cité en page 29.)

[Vento *et al.* 2015] J. Vento, L. Travé-Massuyès, V. Puig et R. Sarrate. *An incremental hybrid system diagnoser automaton enhanced by discernibility properties.* IEEE Transactions on Systems, Man, and Cybernetics : Systems, vol. 45, no. 5, pages 788–804, 2015. (Cité en page 38.)

[Verberne *et al.* 2000] A. Verberne, F. van Harmelen et A. ten Teije. *Anytime Diagnostic Reasoning using Approximate Boolean Constraint Propagation.* In Seventh International Conference on Principles of Knowledge Representation and Reasoning, 2000. (Cité en page 82.)

[Villemeur 1988] A. Villemeur. Sûreté de fonctionnement des systèmes industriels : fiabilité, facteurs humains, informatisation. Eyrolles, 1988. (Cité en page 25.)

[Ye & Dague 2017] L. Ye et P. Dague. *An Optimized Algorithm of General Distributed Diagnosability Analysis for Modular Structures.* IEEE Transactions on Automatic Control, vol. 62, no. 4, pages 1768–1780, 2017. (Cité en page 121.)

[Yin & Li 2016] X. Yin et Z. Li. *Decentralized fault prognosis of discrete event systems with guaranteed performance bound.* Automatica, vol. 69, pages 375–379, 2016. (Cité en page 122.)

[Yu 2017] J. Yu. *Adaptive hidden Markov model-based online learning framework for bearing faulty detection and performance degradation monitoring.* Mechanical Systems and Signal Processing, vol. 83, pages 149–162, 2017. (Cité en page 120.)

[Zaatiti *et al.* 2018] H. Zaatiti, L. Ye, P. Dague, J.-P. Gallois et L. Travé-Massuyès. *Abstractions Refinement for Hybrid Systems Diagnosability Analysis.* In Diagnosability, Security and Safety of Hybrid Dynamic and Cyber-Physical Systems, pages 279–318. Springer, 2018. (Cité en page 121.)

[Zabi *et al.* 2013] S. Zabi, P. Ribot et E. Chanthery. *Health Monitoring and Prognosis of Hybrid Systems.* In Annual Conf. of the PHM Society, 2013. (Cité en page 47.)

[Zhang *et al.* 2009] B. Zhang, C. Sconyers, R. Patrick et G. Vachtsevanos. *A Multi-Fault Modeling Approach for Fault Diagnosis and Failure Prognosis of Engineering Systems.* In Annual Conf. of the PHM Society, 2009. (Cité en pages 31 et 33.)

[Zheng *et al.* 1993] Z. Zheng, R.M. Goodman et P. Smyth. *Learning Finite State Machines With Self-Clustering Recurrent Networks.* Neural Computation, vol. 5, no. 6, pages 976–990, 1993. (Cité en page 119.)

[Zilberstein 1996] S. Zilberstein. *Using anytime algorithms in intelligent systems.* AI magazine, vol. 17, no. 3, page 73, 1996. (Cité en page 81.)

[Zolghadri *et al.* 2009] A. Zolghadri, M. Staroswiecki, L. Travé-Massuyès, J. Ragot, P. Dague, E. Bensana, M-C. Charmeau, P. Goupil, X. Olive, E. Benazera, E. Chanthery, C. Jauberthie et R. Pons. *Appel à projet no 3. Programme de recherche "Autonomie des systèmes aéronautiques et spatiaux".* Rapport de contrat : Projet SIRASAS - contrat FRAE, mars 2009, 20p. , no 09150, Thalès Alenia Space ; AIRBUS France ; CNES ; ONERA ; LRI ; LIPN ; CRAN-ENSEM ; DISCO ; SATIE ; IMS Bordeaux, 2009. (Cité en page 84.)

[Zouaghi *et al.* 2011] L. Zouaghi, A. Alexopoulos, A. Wagner et E. Badreddin. *Modified particle petri nets for hybrid dynamical systems monitoring under environmental uncertainties.* In IEEE/SICE Int. Symposium on System Integration, pages 497–502, 2011. (Cité en page 55.)

[Zwingelstein 1995] G. Zwingelstein. Diagnostic des défaillances : Théorie et pratique pour les systèmes industriels. Hermes, 1995. (Cité en page 25.)