



Pronostic des événements de défaillance basé sur les réseaux de Petri Temporels labellisés

Redouane Kanazy

► To cite this version:

Redouane Kanazy. Pronostic des événements de défaillance basé sur les réseaux de Petri Temporels labellisés. Automatique / Robotique. Université de Lyon, 2020. Français. NNT : 2020LYSEI132 . tel-03510389

HAL Id: tel-03510389

<https://theses.hal.science/tel-03510389>

Submitted on 4 Jan 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



N°d'ordre NNT : 2020LYSEI132

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
INSA de Lyon

Ecole Doctorale N° 160
Ecole Doctorale Electronique,
Electrotechnique, Automatique

Spécialité/ discipline de doctorat :
Automatique

Soutenue publiquement/à huis clos le 15/12/2020, par :
Redouane KANAZY

**Pronostic des événements de
défaillance basé sur les réseaux de
Petri temporels labellisés**

Devant le jury composé de :

KAMACH, Oulaid	Professeur ENSA de Tanger	Président
Armand TOGUYENI	Professeur, Centrale de Lille Institut	Rapporteur
Zineb SIMEU-ABAZI	Professeur, Université Joseph Fourier	Rapporteur
Oulaid KAMACH	Professeur, ENSA de Tanger	Examineur
Éric NIEL	Professeur, INSA de Lyon	Directeur de thèse
Samir CHAFIK	Professeur, EMSI de Casablanca	Co-directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	CHIMIE DE LYON http://www.edchimie-lyon.fr Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr INSA : R. GOURDON	M. Stéphane DANIELE Institut de recherches sur la catalyse et l'environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 Avenue Albert EINSTEIN 69 626 Villeurbanne CEDEX directeur@edchimie-lyon.fr
E.E.A.	ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE http://edeea.ec-lyon.fr Sec. : M.C. HAVGOUDOUKIAN ecole-doctorale.eea@ec-lyon.fr	M. Gérard SCORLETTI École Centrale de Lyon 36 Avenue Guy DE COLLONGUE 69 134 Écully Tél : 04.72.18.60.97 Fax 04.78.43.37.17 gerard.scorletti@ec-lyon.fr
E2M2	ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION http://e2m2.universite-lyon.fr Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : H. CHARLES secretariat.e2m2@univ-lyon1.fr	M. Philippe NORMAND UMR 5557 Lab. d'Ecologie Microbienne Université Claude Bernard Lyon 1 Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX philippe.normand@univ-lyon1.fr
EDISS	INTERDISCIPLINAIRE SCIENCES-SANTÉ http://www.ediss-lyon.fr Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : M. LAGARDE secretariat.ediss@univ-lyon1.fr	Mme Sylvie RICARD-BLUM Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Curien - 3ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tel : +33(0)4 72 44 82 32 sylvie.ricard-blum@univ-lyon1.fr
INFOMATHS	INFORMATIQUE ET MATHÉMATIQUES http://edinfomaths.universite-lyon.fr Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	M. Hamamache KHEDDOUCI Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tel : 04.72.44.83.69 hamamache.kheddouci@univ-lyon1.fr
Matériaux	MATÉRIAUX DE LYON http://ed34.universite-lyon.fr Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction ed.materiaux@insa-lyon.fr	M. Jean-Yves BUFFIÈRE INSA de Lyon MATEIS - Bât. Saint-Exupéry 7 Avenue Jean CAPELLE 69 621 Villeurbanne CEDEX Tél : 04.72.43.71.70 Fax : 04.72.43.85.28 jean-yves.buffiere@insa-lyon.fr
MEGA	MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE http://edmega.universite-lyon.fr Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction mega@insa-lyon.fr	M. Jocelyn BONJOUR INSA de Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69 621 Villeurbanne CEDEX jocelyn.bonjour@insa-lyon.fr
ScSo	ScSo* http://ed483.univ-lyon2.fr Sec. : Véronique GUICHARD INSA : J.Y. TOUSSAINT Tél : 04.78.69.72.76 veronique.cervantes@univ-lyon2.fr	M. Christian MONTES Université Lyon 2 86 Rue Pasteur 69 365 Lyon CEDEX 07 christian.montes@univ-lyon2.fr

Cette thèse est accessible à l'adresse : <http://theses.insa-lyon.fr/publication/2020LYSEI132/these.pdf>

© (R) Kinazyt, 2020, INSA Lyon, tous droits réservés. Géologie, Science politique, Sociologie, Anthropologie

REMERCIEMENTS

La thèse de doctorat représente un travail s'inscrivant dans la durée, et pour cette raison, constitue le fil conducteur d'une tranche de vie de son auteur, parfois au crépuscule de la candeur étudiante, et souvent à l'aube de la maturité scientifique. De nombreuses personnes se retrouvent ainsi de manière fortuite ou non, pour le pire ou le meilleur, entre le doctorant et son doctorat. Ce sont ces personnes que j'aimerais mettre en avant dans ces remerciements.

Je remercie chaleureusement et grandement mon directeur de thèse le professeur Éric NIEL, pour la confiance qu'il m'a accordée et le temps qu'il a consacré à diriger cette recherche. J'aimerais lui dire à quel point j'ai apprécié ses qualités humaines et scientifiques, surtout pendant la période de la rédaction de mon mémoire de thèse. Ses conseils et son soutien, je ne les oublierai jamais. Ce travail n'aurait pu être mené à bien sans son soutien, sa grande disponibilité. Je joins également à ces remerciements mon co-directeur de thèse le professeur Samir CHAFIK, un homme de valeurs, un homme de rigueur et de qualités humaines et professionnelles. Celui qui a le mérite - après Dieu Tout-Puissant.

Je remercie l'ensemble des membres du laboratoire Ampère de l'INSA de Lyon, qu'ils soient doctorants, enseignants chercheurs ou directeurs, sans oublier le personnel de l'école doctorale EEA et FEDORA,

Ce travail n'aurait pas été possible sans le soutien de l'EMSI, qui m'a permis, de me consacrer sereinement à l'élaboration de ma thèse. Merci aux directeurs, aux collègues enseignants et tout le personnel sans exception.

Au terme de ce parcours, je remercie enfin celles et ceux qui me sont chers et que j'ai quelque peu délaissés ces derniers mois pour achever cette thèse. Leurs attentions et encouragements m'ont accompagné tout au long de ces années. Je suis redevable à mes parents, à ma femme Atiyate et mes filles Sirine et Yousra, mon frère Younes et mes sœurs Myriem, Assia et Soundous, à mon grand-père que Dieu le garde en bonne santé, à toute la famille KANAZY, MAKAR et DIAI pour leur soutien et leur confiance indéfectible dans mes choix. Enfin, j'ai une pensée toute particulière pour mes deux grand-mères et mon grand-père maternel qui nous ont quittés, dont la mémoire partagée n'est pas étrangère à mon goût pour l'histoire.

C'est par la grâce de mon Seigneur que ce travail est accompli.

Résumé

Le pronostic de défaillance est une exigence à laquelle les communautés scientifiques répondent constamment en développant de nouvelles techniques réduisant l'impact des arrêts accidentels du système. Dans un cadre de réactivité aux événements de défaillance, deux pistes ont été exploitées dans le domaine des systèmes à événements discret (SED) : une aide à la décision passive, appelée "diagnostic", qui identifie la causalité des pannes; réduisant ainsi à l'opérateur la difficulté de déterminer la source d'un dysfonctionnement et une aide à la décision active, appelée "pronostic", qui agit avant l'état de panne en signalant à l'opérateur à l'avance l'occurrence d'un événement de défaillance susceptible d'altérer le bon fonctionnement du système. Les approches développées pour analyser les deux pistes ont été réparties en deux catégories : à base de modèle et sans modèle. Les approches sans modèle extrapolent la séquentialité des événements pour déterminer le comportement du système. Leurs performances dépendent de la qualité et de la disponibilité de données représentatives des situations de fautes ou du nombre de défaillances dissociables du système. Les approches à base de modèle utilisent une abstraction du comportement du système, par la construction d'un modèle logique qui utilise la connaissance physique ou fonctionnelle du système. Le but dans les deux approches est d'avoir un modèle du fonctionnement du système qui permet d'effectuer une analyse logique (l'ordre d'occurrence des événements), probabiliste (la probabilité d'occurrence des événements) ou temporelle (la date d'occurrence des événements).

Nos travaux s'insèrent dans le cadre d'un pilotage d'un système soumis à des événements de défaillance. Nous avons développé une approche de pronostic à base de modèle, où tout comportement est supposé être connu et explicitement inclus dans le modèle. Notre analyse comportementale du système est basée sur la séquentialité et la date d'occurrence des événements, ce qui justifie le choix de modéliser le système étudié par des réseaux de Petri temporels labellisés (RdPTL). Nous avons proposé un modèle du système réduit à 3 modes de fonctionnement : nominal, dégradé et critique, appelé MMF (modèle de modes de fonctionnement). Nous avons exploité les états et les transitions d'états du système, de façon à circonscrire l'espace d'états du système et à aboutir à un modèle à la fois compact et expressif (RdP temporels sont généralement avec un espace d'états explosif). À partir du graphe d'accessibilité du MMF, nous avons générer un modèle pronostiqueur avec une paramétrisation des états i.e. l'introduction d'une horloge et un système d'inéquation des horloges pour chaque état du système. Les états obtenus par la discrétisation du temps sont alors regroupés dans un seul état et le système d'inéquations déterminera les valeurs des horloges. Après vérification des événements de défaillances pronosticable et sur la base des résultats du système d'inéquations nous déterminons à l'avance la date au plus tôt de l'occurrence d'un événement de défaillance. À partir de cette information temporelle, l'opérateur planifie ses interventions de réparation, évitant ainsi, le risque d'un arrêt imprévisible du système. Nous avons choisi

comme benchmark à cette proposition la cellule de batterie d'une voiture électrique.

Abstract

Failure prognosis is a requirement that the scientific communities constantly respond to through the development of new techniques that reduce the potential impact of accidental system shutdowns. In order to provide reactivity to failure events, two techniques have been exploited in the field of discrete event systems (DES): a passive decision aid, called "diagnosis", which identifies the causality of failures, thus reducing the difficulty for the operator to determine the source of a dysfunction. Active decision aid, called "prognosis", which proactively anticipates the failure state by informing the operator in advance of the occurrence of a failure event that could affect the efficient system operation. The approaches developed to analyse the two techniques have been divided into two categories: model-based and non-model-based. The non-model approaches extrapolate the sequentiality of events to determine system behaviour. Their performance depends on the quality and availability of data representative of fault situations or the number of dissociable system failures. Model-based approaches use an abstraction of the system's behaviour by constructing a logical model that uses physical or functional information about the system. The goal in both approaches is to have a model of system operation that allows for logical (the order of occurrence of events), probabilistic (the probability of occurrence of events) or temporal (the date of occurrence of events) analysis.

Our work is part of a monitoring of a system affected by failure events. We have developed a model-based prognostic approach, where all behaviour is assumed to be known and explicitly included in the model. Our behavioural analysis of the system is based on the sequentiality and the date of occurrence of the events, which justifies the choice to model the studied system by labelled temporal Petri nets (TLPN). We proposed a reduced system model to 3 operating modes: nominal, degraded and critical, called MMF (mode of operation model). We used the states and state transitions of the system, to circumscribe the state space of the system and lead to a model that is compact and expressive (temporal PN are generally with an explosive state space.). From the accessibility tree of the MMF, we generated a prognosor model with states parameterization i.e. the introduction of a clock and a clock inequality system for each state of the system. The states obtained from the discretization of time are then combined into a single state and the inequation system will determine the values of the clocks. After checking the prognosticable failure events and based on the results of the inequation system we determine in advance the earliest date of occurrence of a failure event. From this temporal information, the operator plans his repair interventions, thus avoiding the risk of an unpredictable system shutdown. We have chosen the battery cell of an electric car as a benchmark for this proposal.

Lexique

Dans le cadre de l'organisation de notre mémoire, cette première partie définira les différentes notions dont l'interprétation de l'élaboration de notre proposition. Dans le but de distinguer l'importance des événements et des pannes dans les fonctions de diagnostic et pronostic.

Système : Une partie limitée du monde réel contenant des éléments associés entre eux.

Modèle : une représentation simplifiée d'un système.

Simulation : la construction de modèles mathématiques, et l'étude de leurs propriétés, avec, comme référence, les propriétés du système étudié.

Systèmes à événements discrets : Un système à événement discret (SED) est un système à espace d'états discrets dont les transitions entre les états sont associées à l'occurrence d'événements discrets asynchrones. [1].

Événement : C'est un changement d'état du système qui peut apparaître à tout instant.

Etat : Une situation qui représente un comportement stable au cours de la vie opérationnelle du système.

Erreur : C'est un état du système à partir duquel la poursuite de l'exécution est susceptible de conduire à une défaillance.

Faute : C'est un événement ayant entraîné une erreur.

Défaut : C'est un état de déviation du système par rapport à son comportement nominal, qui ne l'empêche pas de remplir sa fonction. Un défaut est donc une anomalie instable qui concerne une ou plusieurs propriétés du système.

Défaillance : C'est un événement qui empêche l'aptitude d'une unité fonctionnelle à accomplir la fonction souhaitée.

Panne : C'est un état instable qui est la conséquence d'une défaillance affectant le système, aboutissant à une interruption permanente de sa capacité à remplir une fonction requise et pouvant provoquer son arrêt complet.

Fiabilité : C'est la caractéristique d'un dispositif, exprimée par la fiabilité, qu'il accomplisse une fonction requise, dans des conditions données, pendant une durée donnée [2].

Modes de fonctionnement : Un système présente généralement plusieurs modes de fonctionnement. Nous retiendrons les modes de plusieurs types parmi lesquels [3] :

- Mode de fonctionnement nominal : dans ce mode, le fonctionnement du système est conforme aux exigences requises. Les performances du système coïncident avec le cahier des charges établi par l'utilisateur.
- Mode de fonctionnement dégradé : le système répond partiellement aux exigences requises. Malgré l'absence de défaillances, les performances du système sont inférieures à celles attendues par l'utilisateur. Cette dégradation progressive des performances du système est généralement due au vieillissement d'un ou plusieurs composants du système.
- Mode de fonctionnement défaillant : le système passe à ce mode à cause de l'occurrence d'une ou plusieurs défaillances. Les conséquences de ces défaillances sur le système caractérisent ce mode. Retenons, qu'un système peut avoir plusieurs modes de défaillances, cependant, il admet un seul mode de bon fonctionnement.

Localisation de défauts : Consiste à remonter les symptômes pour retrouver l'ensemble des éléments défaillants.

Identification de défauts : Elle consiste à identifier les causes qui ont mené à une situation anormale.

Diagnostic : [4] Définit le diagnostic comme une détermination du type, de la taille, de l'endroit et de l'instant d'occurrence d'un défaut ; il suit la détection de défauts et inclut la localisation et l'identification. Selon [5] le diagnostic est la fonction qui permet de localiser les éléments défaillants et de déterminer les causes qui ont entraîné la dégradation du système. [6] introduit le diagnostic comme une identification de la cause probable 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. [7] définit le diagnostic comme une fonction qui vise à détecter au plus tôt une déviation du comportement nominal du système et d'en déterminer les causes.

Effet de défaillance : Conséquence d'un mode de défaillance sur l'opération en cours, la fonction, ou le statut d'une variable.

Modèle qualitatif : C'est un modèle qui décrit le comportement du système avec les relations (état/événement) entre des variables et les paramètres en termes heuristiques tels que des causalités ou des règles. Il permet d'avoir des coupes minimales en vue de pointer l'événement le plus critique.

Modèle quantitatif : C'est un modèle qui décrit le comportement de système avec les relations entre des variables et les paramètres en termes analytiques tels que des équations différentielles.

Observabilité : La notion d'observabilité dans le cadre des SED a été introduite dans les travaux de [8]. Les événements qui décrivent l'évolution d'un SED ont été classés en événements observables et événements non observables.

Signature Temporelle Causale (STC) : C'est un sous-ensemble d'événements observables partiellement ordonnés pouvant caractériser un comportement défaillant d'un système, après la détection d'une défaillance. Ces événements sont contraints par un ensemble de contraintes temporelles portant sur leurs occurrences [9].

Chronique : C'est un graphe orienté définissant des contraintes temporelles entre les dates d'occurrence d'un ensemble d'événements généré pour un procédé.

Pronostic : [10] a défini le pronostic comme une prédiction de l'évolution future de la défaillance qui indique le temps restant avant que la défaillance ne devienne critique. Les outils du pronostic sont basés sur des techniques de traitement du signal (méthodes de lissage et d'extrapolation) ou de modélisation. Le but est de prédire les pannes induites par la défaillance. La norme ISO 13381-1 l'a défini comme une estimation de la durée de fonctionnement avant défaillance et du risque d'existence ou d'apparition ultérieure d'un ou de plusieurs modes de défaillance. D'autres l'interprètent comme une estimation probabiliste d'une défaillance future, ou une prédiction de temps résiduel avant défaillance (communément appelé RUL pour Remaining Useful Life). Pour [11] le pronostic a pour objectif de prédire ces comportements afin d'entreprendre les actions nécessaires d'arrêt, de reconfiguration ou encore de maintenance prédictive. [12] définissent le pronostic comme une fonction qui détermine les conséquences d'une défaillance sur le fonctionnement futur du système. Deux types de pronostic peuvent être distingués. Le premier, appelé

pronostic préliminaire, cherche les conséquences inévitables d'une défaillance. Le deuxième, appelé pronostic préventif, est centré sur l'erreur latente et sur la propagation de défaillance. Ce type de pronostic est utilisé afin d'éviter l'exécution d'activités menant à la détection du même symptôme. [6] introduit le pronostic comme une fonction qui consiste à calculer une prédiction de l'état d'un composant ou d'un système dans le futur. [13] le définit comme une prédiction du temps de vie restant sous contrainte temporelle. Il s'agit d'anticiper l'apparition des défaillances sur des horizons de temps de prédiction étendus.

Contrairement au diagnostic, au lieu d'attendre la détection d'une défaillance pour déterminer sa causalité, la fonction du pronostic signale l'occurrence d'un événement de défaillance avant son apparition. Cette prédiction peut être :

- Logique (ordre d'occurrence de l'événement défaillant),
- Temporelle (date d'occurrence d'un événement de défaillance),
- Stochastique (probabilité d'occurrence de l'événement de défaillance).

Table des matières

CHAPITRE 1 : Introduction	16
CHAPITRE 2 : Etat de l'art.....	25
2.1. Introduction.....	26
2.2. Systèmes à événements discrets	26
2.3. Approches sans modèle.....	27
2.4. Approches avec modèle.....	32
2.4.1. Automates à états finis	33
2.4.2. Réseaux de Petri	35
2.4.3. Les chroniques	47
2.5. Diagnostic des SED	49
2.5.1. Contexte.....	49
2.5.2. Diagnostiqueur	50
2.5.3. Diagnosticabilité	55
2.5.4. Comparatif.....	57
2.6. Pronostic des SED	58
2.6.1. Contexte.....	58
2.6.2. Pronostiqueur.....	58
2.6.3. Travaux.....	59
2.6.4. Comparatif.....	63
2.6.5. Pronosticabilité et observabilité des événements	64
2.7. Conclusion.....	68
CHAPITRE 3 : Proposition d'une approche de pronostic temporel paramétrique	70
3.1. Introduction.....	71
3.2. Approche de pronostic basée sur les RdPTL.....	72
3.2.1 Démarche générale de l'approche du pronostic	72
3.2.2 Description de l'approche du pronostic.....	73
3.2.3 Modèle des modes de fonctionnement	76
3.2.4 Etats et séquences paramétriques	78
3.2.5 Module pronostiqueur	84
3.2.6 Propriété de pronosticabilité	91
3.3. Discussions	96
3.3.1. Multi-composant	96
3.3.2. Limites sur les hypothèses	97
3.3.3. Simulation INA	98

3.4. Conclusion.....	99
CHAPITRE 4 : Benchmark.....	100
4.1. Introduction.....	101
4.2. Description du système : Les cellules de batterie	101
4.3. Méthode de pronostic.....	103
4.3.1 Modèle RdPTL du système	104
4.3.2 Modèle pronostiqueur	105
4.3.3 Propriété de pronosticabilité	115
4.4. Simulation.....	117
4.5. Conclusion.....	118
CHAPITRE 5 : Conclusion & perspectives	119

Liste des figures

Figure 1.1: Processus de propagation d'une faute [15].	17
Figure 1.2: Etats & événements dans un SED.	18
Figure 1.3: Technique du chien de garde.	19
Figure 1.4: Pronostic et diagnostic en fonction des états et événements.	21
Figure 1.5: Approche de pronostic sans modèle.	23
Figure 1.6: Approche de pronostic avec modèle.	23
Figure 2.1: Exemple d'arbre de défaillances.	28
Figure 2.2: Exemple SED modélisé par un automate à états finis.	33
Figure 2.3: Automate temporisé.	34
Figure 2.4: Réseau de Petri.	36
Figure 2.5: Réseaux de Petri Temporel.	37
Figure 2.6: Exemple de réseau de Petri Temporel.	39
Figure 2.7: RdPT et Graphe d'état.	43
Figure 2.8: Graphe d'états du RdPT selon POPOVA.	44
Figure 2.9: Réseau de Petri Labellisé.	44
Figure 2.10: Réseau de Petri temporel labellisé.	46
Figure 2.11: Exemple d'une chronique.	48
Figure 2.12: Principe général de la reconnaissance de chroniques.	48
Figure 2.13: Approche de Sampath.	50
Figure 2.14: Un SED et son diagnostiqueur selon Sampath.	51
Figure 2.15: Diagnostiqueur selon l'approche de ZAD.	51
Figure 2.16: Modèle diagnostiqueur d'un SED temporel.	52
Figure 2.17: Diagnostiqueur Selon l'approche Boussif.	52
Figure 2.18: Approche de diagnostic SADDEM.	54
Figure 2.19: Diagnostiqueur Sampath Vs Viana.	56
Figure 2.20: Modèle d'automate à état.	60
Figure 2.21: Diagnostiqueur GENC.	60
Figure 2.22: Architecture du pronostic.	61
Figure 2.23: Représentation des trajectoires [52].	63
Figure 3.1: Framework général de la démarche.	72
Figure 3.2: États et modes de fonctionnement.	73
Figure 3.3: Le pronostic et le diagnostic par rapport à l'évolution du système.	74
Figure 3.4: Approche de pronostic.	75
Figure 3.5: Structure et modes de fonctionnement d'un MMF.	76
Figure 3.6: Exemple 1 d'un modèle RdPTL.	76
Figure 3.7: Paramétrisation d'états.	78
Figure 3.8: Génération du pronostiqueur.	84
Figure 3.9: Graphe d'accessibilité du MMF de la figure 3.5.	85
Figure 3.10: Le modèle pronostiqueur avec la première approche.	89
Figure 3.11: Modèle pronostiqueur avec la deuxième approche.	91
Figure 3.12: Vérification de la pronosticabilité.	92
Figure 3.13: Pronosticabilité (modèles RdPL sans observateur).	92
Figure 3.14: Pronosticabilité (modèles RdPTL avec observateur).	94
Figure 3.15: $G(R_TL)$ Graphe d'accessibilité.	94
Figure 3.16: Démarche de pronostic.	96

Figure 3.17: Un RdPTL non sauf.	97
Figure 3.18: Etape de simulation d'un modèle RdP	98
Figure 4.1: Représentation simplifiée d'une batterie Lithium.	102
Figure 4.2: MMF d'une cellule de batterie électrique.....	104
Figure 4.3: Modèle pronostiqueur d'une cellule de Batterie électrique (Approche 1).....	106
Figure 4.4: Modèle pronostiqueur d'une cellule de Batterie électrique (Approche 2).....	114
Figure 4.5: MMF d'une cellule avec cycles d'exécution.	116
Figure 4.6: Pronostiqueur du MMF avec cycles d'exécution.	117

Liste des tableaux

Tableau 1.1: Classification des travaux du diagnostic.....	20
Tableau 1.2: Travaux orientés pronostic.....	22
Tableau 2.1: Portes logiques d'un arbre de défaillances.	27
Tableau 2.2: Approches sans modèle.....	31
Tableau 2.3: Tableau comparatif des méthodes de diagnostic.	58
Tableau 2.4: Travaux sur le pronostic.	64

CHAPITRE 1 : Introduction

1. Introduction

Dans ce premier chapitre, nous définirons de façon concise le lexique qui sera utilisé tout au long de ce mémoire, ensuite nous présenterons le contexte général dans lequel nos travaux de recherche se sont basés. Nous déterminerons la problématique et justifierons notre positionnement par rapport aux travaux existants. La dernière partie de ce chapitre sera consacrée à la description de l'organisation du mémoire.

À ce jour, les exigences en termes de performance des systèmes automatisés ne cessent de s'accroître pour garantir leur disponibilité. Un arrêt accidentel qu'il soit partiel ou total d'un système, provoque parfois des conséquences désastreuses et coûteuses pour l'entreprise. Des applications de supervision de type SCADA (Supervisory Control And Data Acquisition) sont mises à la disposition des ingénieurs de contrôle pour augmenter la productivité, l'efficacité, l'agilité et la qualité de ces systèmes, tout en minimisant les coûts relatifs au maintien de leur bon fonctionnement. Le comportement d'un système est lié à ses états de fonctionnement possibles et à sa réaction en cas d'occurrence d'un événement imprévisible. La surveillance permet de détecter les états d'erreur ou de panne pour attirer l'attention de l'opérateur de supervision sur l'apparition d'un ou de plusieurs événements (fautes/défaillants) susceptibles d'affecter le bon fonctionnement du système surveillé [14].

Pour comprendre le principe d'apparition d'une défaillance, [15] décrit par un processus récursif la causalité qui existe entre faute, erreur et défaillance. La figure suivante illustre cette propagation :



Figure 1.1: Processus de propagation d'une faute [15].

La faute et la défaillance sont des événements mais l'erreur représente un état. Une faute peut être accidentelle ou intentionnelle. Une fois activée elle produit une erreur. La défaillance survient lorsqu'une erreur affecte le service délivré par le système. Les conséquences d'une défaillance sont permanentes tant que la réparation ou la récupération complète du système n'a pas eu lieu. Dans la figure 1.1 nous avons représenté un processus récursif où il n'y a qu'une seule entrée et une seule sortie. Cependant, dans des cas plus complexes on peut envisager qu'à la sortie d'une défaillance on peut avoir plusieurs fautes différentes et que plusieurs erreurs peuvent être générées avant l'occurrence d'une défaillance.

Nos travaux s'insèrent dans un pilotage correct d'un système soumis à des événements de défaillances imprévisibles. A partir du processus de propagation de faute de la figure 1.1 on peut en déduire que l'occurrence d'une faute est la source principale d'une défaillance future. Il faut donc déterminer ce qu'on doit détecter et sur quoi on doit agir. Sachant que les commutations des états du système dépendent des occurrences imprévisibles d'événements, nous pouvons dire que l'évolution de ces états est fonction d'une dynamique discrète des événements. Ce qui est cohérent avec l'évolution des systèmes à événements discrets (SED) qui est conditionnée par l'occurrence d'événements discrets. Ses évolutions, qui ont lieu à des instants discrets, sont régies par des événements associés à des conditions logiques internes ou externes au SED. Lorsqu'un SED est spécifié sans informations temporelles sur les occurrences d'événements, on génère un modèle qui décrit le comportement logique du système. À l'inverse, le modèle temporisé associe des dates d'occurrences aux événements.

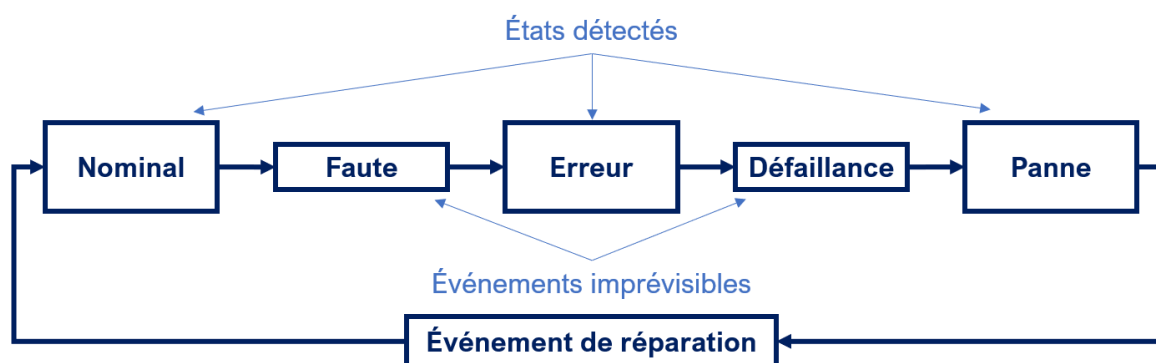


Figure 1.2: Etats & événements dans un SED.

Dans la figure 1.2 nous supposons que la faute est la cause d'un phénomène logique. Le but de cette représentation de propagation de faute sur un SED est de déterminer ce qui est détecté (états) et ce qui est subi (événements). Pour simplifier l'explication, nous avons illustré trois modes de fonctionnement du système (nominal, erreur et panne). A l'état initial le système est dans un état de fonctionnement nominal. C'est l'occurrence d'un événement de faute qui entraîne le changement d'état du système vers un état d'erreur. L'occurrence d'un événement de défaillance bascule l'état du système vers un état de panne. La détection d'état repose sur une comparaison entre le comportement prévu et le comportement observé du système, l'écart entre ces comportements est synonyme de défaillance.

De la figure 1.2 nous pouvons conclure que : soit il faut attendre l'apparition d'événement de défaillance imprévisible et subir les conséquences de son apparition, soit bien agir avant son occurrence et dans ce cas il faut définir comment. La constitution du comportement du système peut être réaliser à partir des événements et leurs états conséquents.

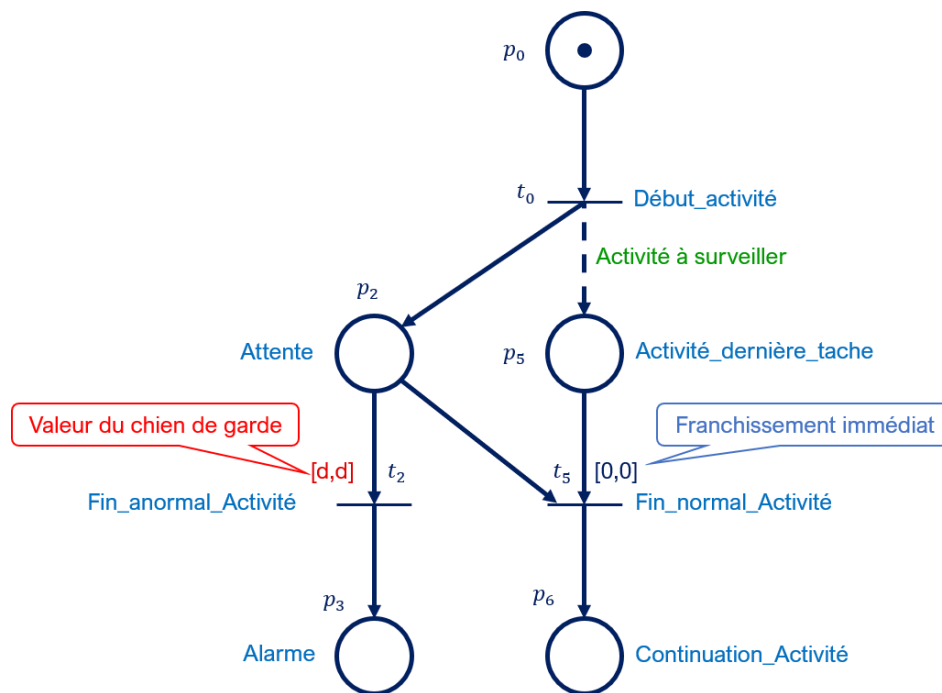
Par ailleurs, le comportement des systèmes à événements discrets peut être représenté selon différentes approches :

L'approche à base de modèle qui repose sur une représentation mathématique du comportement physique du système. Tout comportement nominal ou défaillant doit être explicitement inclus et il est supposé être connu. Cette méthode offre des résultats précis à la surveillance du système pour détecter l'évolution de ses états. Néanmoins, il est difficile de transposer le modèle construit sur un autre système. Dans le cas d'un système multi-composants par exemple, la construction de son modèle sera difficile, ce qui restreint l'avantage de son utilisation pour les systèmes génériques.

L'approche à base de données qui repose quant à elle sur l'utilisation des données extraites des techniques tels que le traitement de signal, l'intelligence artificielle, les systèmes experts, la classification des données etc. Elle n'a pas besoin de connaissance approfondie sur le système. Sa performance est liée à la qualité et la quantité des données disponibles. Les données récupérées permettent de construire le modèle du système par l'apprentissage de son comportement. Cet apprentissage est parfois difficile (temps de calcul, représentation du comportement réel). Néanmoins, elle reste intéressante dans le cas où on ne dispose pas suffisamment de connaissance sur le système.

L'approche à base d'expériences qui repose sur les connaissances des experts et sur l'historique d'expériences passées qui représentent toutes les conditions d'utilisation du système. Ces connaissances servent à exploiter des outils statistiques et des fonctions de fiabilité qui expriment l'évolution du système. Son utilisation est un peu difficile pour les nouveaux systèmes, vu que l'historique d'expérience d'utilisation n'est pas disponible et elle est moins précise que les autres approches.

Pour assurer la surveillance d'un système, [16] propose une fonction de détection de divergence entre le comportement du système modélisé et le comportement du système observé; Elle permet de signaler la présence de défauts dans un système. Pour minimiser le risque d'erreurs de détection, [5] utilise la technique du chien de garde qui consiste à temporiser chaque opération du graphe de commande. Cette temporisation nécessite la connaissance de la durée maximale de chaque opération comme le montre la figure 1.3 [17].



La technique du chien de garde est un mécanisme simple, mais elle présente cependant de nombreux inconvénients :

Les chercheurs de la communauté SED ont exploité la notion de détection d'état pour proposer une fonction d'aide à la décision après la détection de pannes.

Au milieu des années 1990, [18], [19] a proposé une fonction de diagnostic, qui permet d'établir un lien entre un symptôme observé, la défaillance qui est survenue et ses causes. Sampath a construit un diagnostiqueur qui permet de séparer les états nominaux de ceux qui sont fautifs, pour suivre séparément l'évolution des séquences avec et sans événements de défaillances. Son principal intérêt était de formaliser la propriété de diagnosticabilité qui consiste à déterminer s'il est possible d'effectuer un diagnostic (oui ou non). [5], [12], ont considéré la détection comme un élément distinct de la fonction de diagnostic qui représente plutôt une entité de la surveillance.

Le [tableau 1.1](#) présente la classification de quelques travaux orientés diagnostic selon l'approche sans ou avec modèle :

Méthodes de diagnostic		
Méthodes sans modèle		
Méthodes à base de connaissances		
AMDEC		[20]
Arbre de défaillances		[21]
		[22],
		[23]
Systèmes experts		[24]
		[25]
Méthode à base de traitement de données		
Reconnaissance de formes et classification		[26]
		[27]
Machine learning, réseaux de neurones		[28]
		[29]
Méthodes à base de modèles		
Modèles quantitatifs	Estimations paramétriques, observateurs	[30]
		[31]
	Filtre Kalman	[32]
Modèles qualitatifs	Réseaux de Petri	[33]
		[34],
		[35]
		[11]
	Réseaux Bayésien	[36]
		[37]
	Graphe causal	[38]

Tableau 1.1: Classification des travaux du diagnostic.

Les approches de diagnostic à base ou sans modèle permettent, après la détection d'un comportement défaillant, d'identifier sa causalité. La signature causale représente une séquence d'occurrence d'événements ou une séquence datée d'occurrence d'événements. L'introduction des contraintes temporelles a permis d'améliorer la précision du diagnostic.

Lorsqu'un diagnostiqueur est face à deux séquences d'événements avec un ordre d'occurrence identique, il ne peut pas décider sur la cause d'apparition de la panne (cas d'indéterminisme). Cependant, une signature datée des événements permet à l'aide des informations temporelles sur la date d'occurrence de chaque événement, de surmonter ce problème d'indéterminisme.

Le pilotage d'un système soumis à des défaillances critiques qui altère son fonctionnement nécessite une aide à la décision avant une panne. Une aide active au lieu d'une passive permet d'agir au lieu d'attendre l'occurrence d'un événement de défaillance pour chercher sa causalité. L'objectif est de prédire l'occurrence d'un événement défaillant pouvant conduire le système dans un état critique. La communauté SED a proposé la fonction de pronostic comme solution [39][47][50][51]. Une fonction qui offre aux ingénieurs de contrôle une visibilité sur l'état d'évolution du système, pour programmer à l'avance les dates d'intervention et de réparation du système afin d'éviter tout arrêt accidentel. Plusieurs définitions du pronostic ont été proposées dans la littérature pour améliorer sa précision dans la détection à l'avance d'une défaillance.

La figure 1.4 présente dans un premier niveau l'évolution des états du système en fonction de l'occurrence des événements. Dans le second niveau la figure représente les zones du pronostic et du diagnostic. Le pronostic doit être fait avant l'occurrence d'une défaillance, contrairement au diagnostic qui ne peut s'exécuter qu'après l'occurrence d'une défaillance.

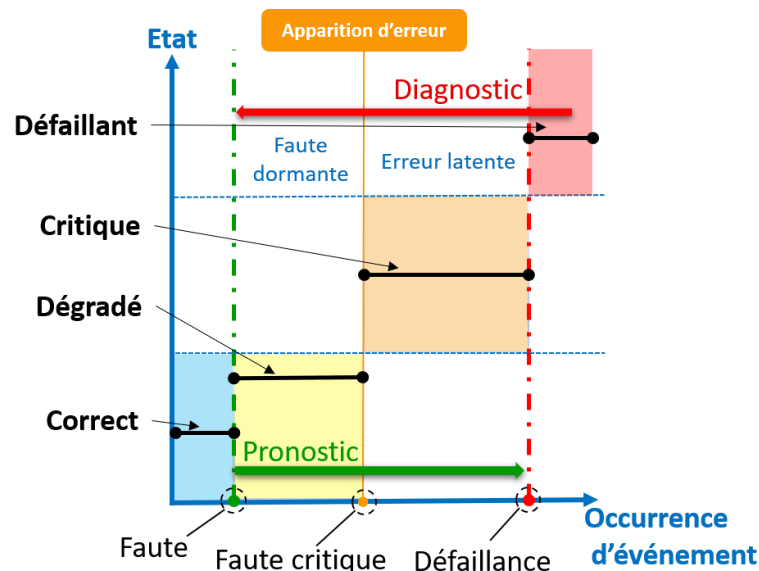


Figure 1.4: Pronostic et diagnostic en fonction des états et événements.

Au début le système opère dans un état de fonctionnement nominal, l'occurrence d'une faute bascule le système vers une exécution dégradée tolérable. L'occurrence d'une faute critique produit une erreur qui met le système dans un état de fonctionnement critique. Cette erreur est considérée latente tant que l'événement de défaillance n'a pas eu lieu. L'occurrence d'un événement de défaillance considéré catalectique provoque un état de panne.

La fonction de diagnostic attend la détection d'un état de panne pour identifier la cause de la défaillance, alors que la fonction de pronostic doit intervenir le plus tôt possible avant l'occurrence d'un événement de défaillance. Dans la figure 1.4, la fonction de pronostic intervient pour signaler la dégradation du système suite à l'occurrence d'un événement de faute. Le but est d'éviter que le système bascule dans un état de panne.

Le tableau 1.2 présente quelques travaux orientés pronostic appliqué sur les systèmes à événements discrets, selon l'approche sans modèle ou à base de modèle.

Etant donné que l'objectif du pronostic est de prédire l'occurrence d'événements de défaillance. [39], [40] ont proposé une approche de prédiction d'occurrence d'événements défaillants en se basant sur l'ordre d'occurrence d'événements.[11] a exploité l'aspect probabiliste pour calculer le taux d'occurrence d'un événement futur, vu que les événements de défaillance sont imprévisibles et donc incertains. [6] propose un pronostic basé sur l'extrapolation du diagnostic pour donner une estimation sur la durée de vie résiduelle d'un composant.

Comme mentionné dans le [tableau 1.2](#), la fonction de pronostic est basée sur 2 types d'approches (avec et sans modèle).

Méthode de pronostic	
Méthodes sans modèle	
Approche à base de connaissance	
Systèmes experts	[41]
Expérience	[42]
Approche probabiliste	
Loi de Weibull	[6]
	[43]
Approche à base de données	
Apprentissage	[44]
	[45]
Estimation d'état	[46]
Méthode à base de modèle SED	
Automates à états finis	[47]
	[48]
	[39]
Réseaux de Petri	[49]
	[50]
	[51]
	[52]

Tableau 1.2: Travaux orientés pronostic.

L'approche sans modèle consiste à l'extrapoler la signature du système. Sa performance dépend de la qualité et de la disponibilité de données représentatives des situations de fautes ou du nombre de défaillances dissociables futures du système.

L'approche avec modèle se base sur une abstraction du comportement du système, par la construction d'un modèle logique qui utilise la connaissance physique ou fonctionnelle du système. Le pronostic se rapporte alors à la détermination des séquences défaillantes à base d'une modélisation approfondie et complète du système qui offre une bonne base d'analyse pour faire du pronostic. Cependant, la difficulté de modélisation augmente avec la complexité du système.

Les [figures 1.5](#) et [1.6](#) représentent respectivement les deux types d'approches de pronostic, sans modèle et avec modèle, respectivement :

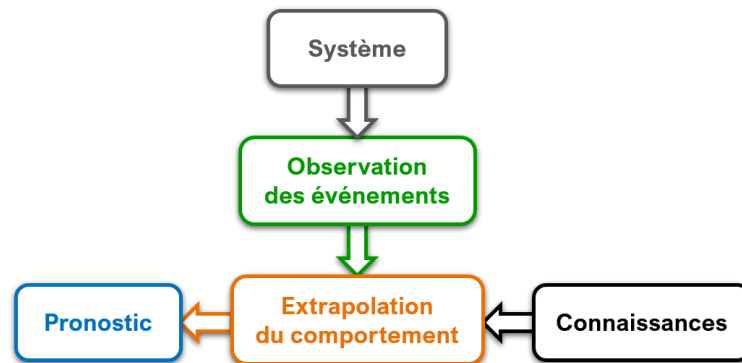


Figure 1.5: Approche de pronostic sans modèle.

La figure 1.5 représente une approche de pronostic sans modèle, qui repose sur des connaissances acquises par les experts ou par un retour d'expériences pendant un temps passé significatif de fonctionnement du système. Ces connaissances permettent la construction d'un modèle. A partir de ce modèle il est possible d'avoir des informations sur l'état actuel du système et prédire l'occurrence d'un événement défaillant grâce à l'extrapolation du comportement du système soit à des seuils de dégradations où la possibilité de défaillance est très importante, soit par rapport à une durée de fonctionnement du système.

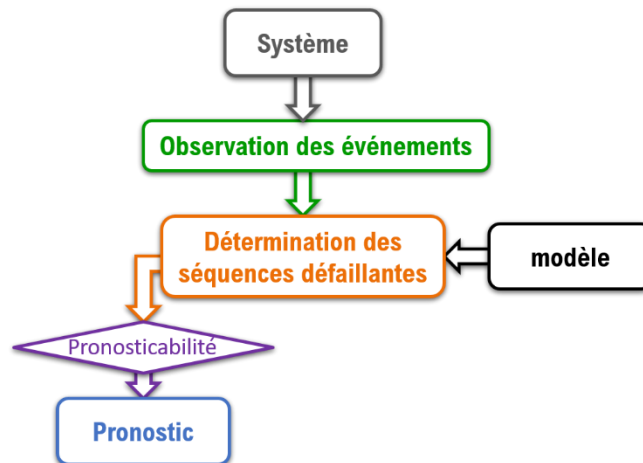


Figure 1.6: Approche de pronostic avec modèle.

La figure 1.6 présente l'approche de pronostic à base de modèle. Une approche qui repose sur un modèle physique du système ne nécessitant pas d'historique de données. Ce modèle représente les commutations des états du système en fonction des événements imprévisibles. L'extrapolation dans ce cas, peut être traduite par une projection sur des séquences d'événements conduisant à un événement de défaillance. Ces séquences d'événements peuvent être exploitées pour faire le pronostic d'une façon logique, temporelle ou combiné des événements. Le pronostic basé sur le séquençement logique, s'intéresse à l'ordre d'occurrence des événements pour prédire N-étapes à l'avance l'occurrence d'un événement défaillant. Il est aussi possible, d'utiliser les probabilités d'occurrence des événements pour déterminer le taux d'occurrence d'un événement de défaillance futur sur un horizon donné. Le pronostic à base temporelle, s'intéresse à la signature temporelle des séquences pour déterminer la date d'occurrence d'un événement de défaillance sans prendre en considération l'ordre d'occurrence des événements, ce qui augmente le risque d'ambiguïté entre les séquences et limite la possibilité de faire le pronostic. Le pronostic qui analysera à la fois l'ordre et la date d'occurrence des événements, va nous permettre de prédire la date d'occurrence d'un événement de défaillance futur.

Dans ce mémoire 2 contributions sont proposées, une dans un cadre formel et l'autre dans un cadre méthodologique.

Dans le cadre formel : nous proposons une approche de pronostic liée à des modes de fonctionnement (nominal, dégradé et critique), qui garantit un homomorphisme entre le système et son modèle comportemental (i.e. une représentation simplifiée du système). Elle nécessite une description et des explications sur le fonctionnement du système et voire même une prédiction de son comportement. Le modèle doit permettre une analyse à la fois logique et temporelle afin de vérifier l'approche du pronostic. L'introduction de l'aspect temporel dans le modèle le rend complexe et explosif. Il faut donc soulever cette problématique et voir s'il existe des méthodes qui permettent de réduire l'espace d'états d'un modèle temporel.

Dans le cadre méthodologique, nous avons proposé un pronostiqueur temporel qui permet d'identifier toutes les séquences qui se terminent avec un événement défaillant. Un pronostiqueur logique, s'intéresse uniquement à l'ordre d'occurrence des événements. En cas d'indéterminisme (i.e. deux séquences avec le même ordre d'événements), la vérification de la propriété de pronosticabilité est difficile. Le pronostiqueur temporel que nous proposons permet de contourner ce problème. La pronosticabilité évalue la capacité de prédire les défauts d'un système [11]. La propriété de pronosticabilité doit être vérifiée avant de faire le pronostic. Un événement de défaillance est dit pronosticable si et seulement si l'identification d'une séquence temporelle contenant cet événement de défaillance est sûre (pas de d'erreur d'identification due au problème d'indéterminisme par exemple) et que la date d'occurrence de cet événement au plus tôt est certaine (Pas d'erreur de prédiction). On considère que tous les événements sont observables et que toute défaillance est non intermittente. Du point de vue méthodologique, nous proposons un algorithme basé sur un système d'inéquations où les variables constituent le temps de franchissement des transitions afin de calculer le temps d'exécution d'une séquence et d'optimiser le nombre d'états possibles. L'objectif est de trouver l'ensemble des valeurs minimales solution du système d'inéquations, ces valeurs constitueront le temps minimal au bout duquel l'occurrence de l'événement de défaillance est certaine.

Dans ce chapitre nous avons présenté le contexte de notre approche de pronostic qui est basée sur les systèmes à événements discrets. Des références ont été citées, elles seront reprises plus en détail dans le chapitre suivant de façon à justifier le choix de l'utilisation d'une approche à base de modèle pour représenter le comportement du système et déterminer l'intérêt de pronostiquer la date d'occurrence d'un événement de défaillance à l'avance.

CHAPITRE 2 : Etat de l'art

2. État de l'art

2.1. Introduction

Ce deuxième chapitre est un état de l'art faisant le point sur les travaux réalisés dans le cadre des approches orientées pronostic. Comme annoncé dans le chapitre précédent, notre objectif est de proposer un cadre formel et méthodologique d'une approche de pronostic qui permet de prédire la date au plus tôt de l'occurrence d'un événement défaillant. Nous allons commencer par déterminer la cohérence entre la dynamique du système qu'on souhaite étudier et celle des systèmes à événements discrets. Nous allons présenter les différentes approches existantes pour la représentation comportementale du système, qu'elles soient avec ou sans modèle. Ainsi, pour se positionner par rapport aux travaux existants, nous avons choisi de présenter des travaux orientés diagnostic et pronostic qui mettent en évidence les notions d'événement et de date. Nous allons démontrer l'intérêt du pronostic par rapport au diagnostic et conclure sur les avantages et les limites de chacune de ces approches. Nous allons ensuite présenter les approches formelles vérifiant la propriété de pronosticabilité, qui valide la possibilité ou non de faire le pronostic. A la fin du chapitre, nous allons conclure sur les différents travaux discutés afin de mettre en évidence l'apport et de notre contribution.

2.2. Systèmes à événements discrets

Bien adaptés aux moyens de suivre des historiques d'événements, les systèmes à événements discrets (SED) ont une dynamique asynchrone dont l'espace d'états est discret. Leur évolution se fait par l'occurrence des événements qui représentent le changement d'états du système. Au lieu de s'intéresser à l'évolution continue des états (évolution graduelle dans le temps), les modèles SED s'intéressent uniquement à l'occurrence des événements (discrets) et à leur séquençement logique, dynamique ou temporel. Dans certains cas ces événements sont datés pour spécifier leurs intervalles d'occurrence. Cependant, le changement d'état d'un SED est décrit formellement par un couple (événement, instant d'occurrence).

Un état dans un SED est représenté par des grandeurs discrètes, comme le nombre de processus en activité, le nombre de messages en attente, etc. L'événement quant à lui, se produit instantanément et cause la transition de l'état d'une valeur (discrète) à une autre valeur. Il peut être identifié avec une action spécifique qui a été prise, ou comme une occurrence spontanée dictée par la nature ou bien comme la conséquence de plusieurs conditions soudainement toutes satisfaites. Ainsi, les transitions d'états s'effectuent seulement lorsque des événements se produisent en des instants discrets du temps.

Nos travaux s'insèrent dans le cadre d'un pronostic de la date d'occurrence d'un événement de défaillance dans un système. On s'intéresse donc aux transitions d'états due à l'occurrence imprévisible d'événements dans des intervalles prédéfinies. Une dynamique qui renvoie à une cohérence avec l'évolution des SED où l'espace d'état est discret et les transitions entre les états sont due aux occurrences d'événements asynchrones [1], [53]. Il est alors nécessaire de modéliser le comportement du système aux événements défaillants pour pouvoir faire le pronostic.

La construction d'un modèle comportemental est réalisée selon deux approches :

- A partir de l'expérience issue de la manipulation du système, des règles de cause à effet sont générées pour réaliser un modèle qui décrit le comportement du système. Les approches à base de ces connaissances sont appelées "des méthodes sans modèle".
- A partir d'une compréhension fondamentale de la dynamique théorique du système, une modélisation du comportement du système est proposée. Les approches à base de ces connaissances sont appelées "des méthodes à base de modèle".

2.3. Approches sans modèle

Une approche sans modèle est basée sur la reconstitution d'un modèle du système par l'apprentissage de son comportement. Des règles de causes et effets sont générées à partir des informations récupérées d'un fonctionnement antérieur du système ou des données d'experts. Parmi les méthodes exploitées dans le domaine des événements imprévisibles, on peut citer :

- **L'arbre de défaillances**, proposée par [54] dans le cadre d'un projet commandé par L'US Air Force. C'est une technique qui recherche les causes d'un événement redouté dans le système. Elle utilise des portes logiques de l'algèbre de Boole pour établir les relations entre les différents événements pour pouvoir construire un arbre complet et calculer ensuite la probabilité d'occurrence de l'événement redouté.

Le tableau 2.1 présente l'ensemble des portes logiques utilisées par l'arbre de défaillances et leurs significations :

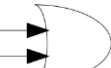
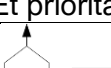
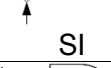
Nom & Symbole	Signification
 ET	La sortie est générée si et seulement si toutes les entrées existent.
 OU	La sortie est générée si et seulement si au moins une des entrées existe.
 OU Exclusif	La sortie est générée si une entrée et une seule existe.
 Et prioritaire	La sortie est générée si et seulement si toutes les entrées existent avec un ordre d'apparition donné.
 SI	La sortie est générée si l'entrée existe et si la condition C est vérifiée.
 K sur n combinaison	La sortie est générée si k parmi les n entrées existent.

Tableau 2.1: Portes logiques d'un arbre de défaillances.

La [figure 2.1](#) représente un exemple d'arbre de défaillances, où sa construction se fait à partir d'événement reliés entre eux par des portes logiques. Le sommet défaillance du système est décomposé de deux défaillances causales (Cause 1 et Cause 2) sachant que le sommet cause 1 est décomposé lui-même en deux sommets causales (Sous-Cause 1.1 et Sous-Cause 1.2) et ainsi de suite jusqu'à ce que l'on trouve une défaillance causale non-décomposable.

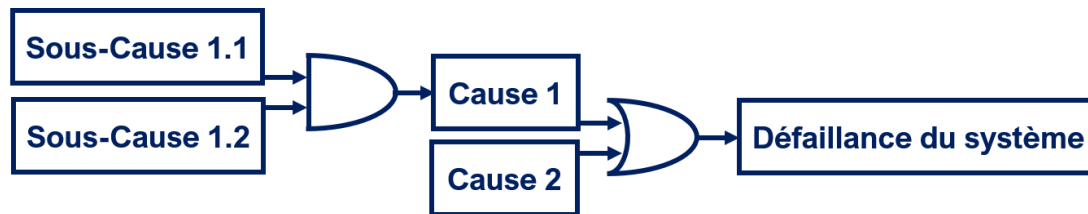


Figure 2.1: Exemple d'arbre de défaillances.

L'arbre de défaillances s'interprète par une séquentialité d'événements qui représente comment l'événement initiateur de défaillance se propage dans le système. Il est surtout utilisé pour ses interprétations qualitatives : cibler le composant critique dans le dysfonctionnement total du système. Cependant, exploiter sa version quantitative semble aussi intéressante car il permet d'avoir une équation logique exprimant la probabilité d'occurrence de l'événement redouté. Ainsi, dans le cas où il existe deux événements initiateurs de la défaillance, il serait alors possible à l'aide des probabilités de déterminer celui qui a causé la défaillance.

A l'origine, l'approche des arbres de défaillances élimine totalement les aspects temporels des scénarios; lorsqu'un arbre de défaillances intègre l'approche de fonctionnement dynamique, les propriétés classiques de l'algèbre de Boole ne peuvent être exploitées. [55] a par exemple défini deux nouveaux types de portes (ET prioritaire, OU prioritaire) qui permettent de décrire des contraintes temporelles sur l'ordre d'occurrence des événements. Toutefois, Il est impossible d'exploiter directement les modèles qui utilisent ces portes, vu qu'elles n'ont pas d'opérations équivalentes dans l'algèbre de Boole.

La séquentialité événementielle dans l'arbre de défaillances a été exploitée pour l'étude a priori du système ; à partir de l'occurrence d'un événement de défaillance, l'arbre de défaillances remonte vers sa causalité à l'aide des combinaisons d'événements qui ont contribué à son apparition. A partir des probabilités de ces événements, il est possible de calculer la probabilité de l'événement redouté. Une méthode qui pourrait être utilisée pour faire du pronostic ; au lieu d'exploiter la séquentialité des événements pour identifier la causalité d'un événement redouté, il serait intéressant de l'utiliser pour déterminer toute séquence d'événements menant à un événement redouté. Il serait alors intéressant de proposer (dans un cadre formel) une sémantique, qui permet d'exploiter en plus de la séquentialité des événements, leurs contraintes temporelles d'occurrences respectives.

L'introduction de l'aspect temporels dans les arbres de défaillances, permet de calculer la probabilité d'observer une défaillance sur une durée (un intervalle $[a,b]$) et de déterminer l'indisponibilité et la fréquence de défaillance du système. [56] propose d'ajouter d'autres portes temporelles (ET PRIORITAIRE et OU PRIORITAIRE) aux arbres de défaillances, leur donnant ainsi, une dimension temporelle. Cependant, leur utilisation peut générer des modèles dont la signification mathématique est ambiguë, puisqu'ils n'ont pas d'opérations équivalentes dans l'algèbre de Boole.

- **Le système expert (SE)**, est utilisé pour le diagnostic et la supervision des systèmes complexes. Les connaissances sont récupérées auprès des experts pour la création des règles. Leur principal inconvénient c'est la difficulté d'acquisition de connaissances et l'incohérence des règles.

Les principaux éléments constituant un système expert :

- La base de données qui inclut un ensemble d'actions effectuées dans le cas où elles correspondent avec la règle stockée dans la base de connaissance.
- Le moteur d'inférence qui effectue un raisonnement pour que le système expert trouve une solution. Il met en relation les règles dans la base de connaissance avec les actions de la base de données.
- Le module des explications permet à l'utilisateur de demander au système expert comment une conclusion particulière a été atteinte et pourquoi il y a besoin d'une action particulière.
- L'interface utilisateur sert de base à l'utilisateur pour communiquer avec le système et vice versa.

La plupart des approches par SE sont utilisés en diagnostic, notamment dans les systèmes médicaux et industriels en ferroviaire et avionique. Le module d'explication des SE établit la démarche des raisonnements. Chaque étape du raisonnement est confirmée par les conclusions tirées des règles de la base de connaissances. Le SE explique le choix d'un raisonnement par rapport à un autre et détermine les règles de la base de connaissances qui l'ont bloqué. Les explications fournies par le SE aident l'utilisateur à perfectionner la base de connaissances, en montrant les points faibles pouvant conduire à des conclusions erronées.

La capitalisation de connaissances d'un SE peut aider à la construction approfondie d'un modèle comportemental du système, permettant ainsi d'avoir un modèle plus expressif. Les démarches de raisonnement qui sont établies par le module d'explication, peuvent être exploitées dans une approche orientée pronostic ; pour déterminer les séquences conduisant à un événement de défaillance en se reposant sur les règles de connaissances du SE basé sur les contraintes temporelles d'occurrence des événements et en identifiant aussi les séquences ambiguës sur le modèle comportemental du système.

- **La classification de données**, se base sur des mesures en temps réel pour la construction du modèle. Deux stratégies sont adoptées par cette méthode :
 - La construction de classes à partir des données d'une manière supervisée (à l'aide d'un expert) ou non supervisée. Un classifieur est généré pour la classification des nouvelles mesures selon leur représentation donnant un comportement normal ou défaillant.
 - La construction d'un modèle statistique pour prédire les valeurs des nouvelles variables à partir de la redondance de l'historique, pour générer des résidus en comparant les prédictions aux valeurs mesurées.

On peut s'inspirer de cette méthode pour effectuer une classification de séquences d'événements (nominale, critique) au lieu d'une classification de défauts. Le but de cette classification de séquences, est de réduire le temps de calcul du pronostic en s'intéressant uniquement aux séquences d'événements critiques. Sur la base d'un historique de séquences critiques, il serait possible de surveiller l'évolution du système et signaler toute exécution de séquence susceptible d'altérer le fonctionnement du système.

- **Les réseaux bayésiens (RB)** initiés par J. Pearl, à la fin des années 1980, ils sont basés à la fois sur une approche probabiliste et une représentation des connaissances sous format d'un graphe acyclique orienté, où les nœuds représentent les variables aléatoires et les

arcs les relations de dépendance entre eux. A partir du fonctionnement du système et des informations des experts, des calculs probabilistes sur l'occurrence de défauts sont effectuées [36], [57]; la probabilité d'occurrence d'un événement peut être calculée à partir de l'occurrence de n'importe quel événement du réseau. Un réseau bayésien offre un modèle compact réduisant le nombre des nœuds en fusionnant ceux avec des probabilités équivalentes d'occurrence d'événements. Les RB offrent à la fois une représentation qualitative et quantitative des relations entre les nœuds. Cependant, les informations probabilistes ne sont pas toujours réalistes et les hypothèses sont trop importantes.

[58] montre dans ses travaux sur les RB que l'analyse de dépendance doit évoquer le temps d'occurrence d'événement, l'ordre des événements et la dépendance d'occurrence de l'événement avec une évolution temporelle. Il utilise l'algorithme de discrétisation introduit par [59], où il introduit une discrétisation dynamique des nœuds par l'utilisation de l'erreur d'entropie, où la précision de probabilité d'occurrence d'un événement augmente à chaque passage d'un nœud à l'autre.

Parmi les avantages des RB et principalement ceux qui nous intéressent dans ce mémoire, malgré le fait qu'on n'adopte pas une méthode probabiliste dans notre approche, car elle est plus fonctionnelle et moins quantitative :

- La représentation compacte du comportement du système.
- Le modèle graphique peut être utilisé à la fois pour le diagnostic et pour le pronostic. Dans [60] les RB statiques sont utilisés dans un premier cas pour calculer les probabilités des causes probables d'une anomalie observée (diagnostic). Dans le second cas, les RB sont utilisés pour prendre en considération la dynamique du système et prédire son comportement futur en fonction de son état actuel et d'autres variables ou contraintes exogènes.
- L'existence d'applications industrielles pratiques, exemple : Probayes [61], [62], BayesiaLab [63], et Netica [64].

Le [tableau 2.2](#) présente une vue générale sur l'ensemble des approches sans modèle introduites dans cette section. Les avantages et les inconvénients de chaque approche sont indiqués, ainsi que l'intérêt de chacune d'entre elles par rapport au pronostic de la date d'occurrence d'un événement défaillant, qui est le sujet de notre thèse.

Les approches sans modèle reposent sur le retour du système et les données fournies par des experts pour capitaliser un volume important d'informations. Ces informations sont déployées sous format de connaissances pour la construction du modèle comportemental du système. La séquentialité d'événements sur ce modèle est exploitée pour déterminer la causalité d'un événement défaillant (diagnostic). En se basant sur la dynamique du système, il est possible de prédire la probabilité d'évolution d'un événement de défaillance. Ces approches ont suscité l'intérêt de la communauté scientifique, dont différentes propositions sur le pronostic ont été développées. L'efficacité des approches sans modèle est liée à la quantité et à la qualité de connaissances disponibles, c'est-à-dire plus on a un flux important d'informations sûres, plus le modèle généré décrira mieux l'ensemble des comportements possibles du système. Il est vrai que certaines approches sans modèle comme les systèmes expert et les réseaux bayésiens, peuvent être exploités efficacement pour faire du pronostic (date d'occurrence d'un événement au plus tôt, probabilité d'occurrence d'un événement dans une date future...).

Approches	Avantages	Inconvénients	Intérêt
Arbre de défaillance	- Une base de connaissances très importante - Permet à la fois le suivre l'évolution d'une défaillance et déterminer sa causalité.	- Difficulté d'utilisation pour des systèmes avec un nombre d'événements important. - Les propriétés classiques de l'algèbre de Boole ne permettent pas d'exploiter l'aspect temporel.	- Séquentialité d'éléments. - Expression logique de séquences, voire de chroniques.
Système expert	- Efficacité au niveau temps de calcul. - Implantation simple.	- Difficulté d'acquisition de l'expertise. - Le manque de généricité. - Renouvellement d'expertise en cas d'amélioration du système. - Pas robuste face à des situations non reconnues. - Données incertaines lors de la manipulation des probabilités ou d'incertitude. - Manque de connaissances profondes - Incohérence des règles. (ajout/suppression/modification) difficulté de détection de l'impact.	- Exploiter la capitalisation des connaissances pour construire le modèle comportemental du système.
Classification de données	- Identification des événements ambigus	- N'offre pas une visibilité d'occurrence future ou antérieure d'un événement.	- Classification des séquences d'événements
Réseaux bayésiens	- Modélise des événements aléatoires. - Modélisation compacte - Combine des aspects statistiques, probabilistes - Calcul des probabilités même si les données sont incertaines.	- Les informations probabilistes ne sont pas toujours disponibles - Leur évaluation est difficile	- Modèle graphique compact. - Séquentialité évidente des événements. - Calcul des indépendances.

Tableau 2.2: Approches sans modèle.

Parmi les pistes possibles pour faire le pronostic :

- Exploiter la qualité des données offerte par les systèmes expert, pour construire un modèle comportemental plus expressif et représentatif du fonctionnement réel du système.
- Utiliser les réseaux bayésiens pour calculer la probabilité d'occurrence d'un événement de défaillance avant une date.

Le seul souci cette approche, serait dans garantir une bonne qualité d'information sur le système (événements de transition, états, date d'occurrence, séquentialité...). La section suivante présente une approche avec modèle, qui peut soulever une partie de la problématique.

2.4. Approches avec modèle

Les approches avec modèle sont souvent coûteuses en termes de calcul mais présentent des avantages, tels que la formalisation ou l'instanciation. Les modèles peuvent représenter un comportement normal ou anormal et peuvent être basés sur des équations ou sur des relations de cause à effet.

L'étude formelle des systèmes à événements discrets permet d'analyser les interactions du système avec son environnement ; L'occurrence des événements provoque le changement d'état du système. La modélisation et l'étude comportementale sont souvent réalisées en exploitant la théorie des langages, les automates à états finis, les réseaux de Petri et l'algèbre de processus. Cette modélisation peut être réalisée au niveau logique (ordre d'occurrence d'événement), temporel (date d'occurrence d'événement) ou stochastique (probabilité d'occurrence d'événement).

En vue de modéliser un SED pour analyser à la fois les contraintes d'ordre qualitatifs (l'ordre et la synchronisation entre les événements) et quantitatifs (respect des délais d'occurrence), nous avons introduit dans la section suivante différentes approches de modélisation solutions. L'analyse du modèle peut être effectuée par une méthode orientée soit états soit événements. L'analyse orientée états, représente l'ensemble des états accessibles et les événements sur un graphe d'états, dans lequel un seul changement d'état élémentaire est possible à la fois. Une activité dans le modèle n'est prise en considération que par des états et sa durée ne peut être associée ni aux états ni aux événements. L'analyse orientée événements, quant à elle, observe comment, quand et dans quel ordre les événements apparaissent. Le graphe d'états doit prendre en compte tous les comportements possibles du système. Une telle méthode met en évidence l'analyse qualitative du temps pour distinguer le début et la fin d'un processus ou d'une activité sur le modèle. Pour détecter la divergence entre l'ordre et de la date d'occurrence d'un événement entre le modèle et le comportement observé du système, le modèle doit représenter l'ensemble des relations possibles entre les états et les événements conséquents. Ces relations de cause à effet entre événements et états seront la base de notre analyse pour signaler l'évolution future du système vers un fonctionnement défaillant.

L'étude du pronostic par la communauté scientifique consiste dans la plupart des travaux à modéliser le comportement du système par des exécutions parallèles, asynchrones, distribuées, concurrentes, non déterministes ou stochastiques. Lors de la conception d'un système, il est possible de vérifier les blocages, les contraintes temporelles, les conflits d'accès aux ressources, etc. et apporte des solutions optimales. Lors de l'étude du système, la recherche des propriétés du système permet d'évaluer ses performances. Nous allons présenter dans la section suivante les deux principales techniques de modélisation utilisée dans le cadre des travaux orientés pronostic, qui sont les automates à états finis et les réseaux de Petri. On expliquera au fur et à mesure chacun technique et ses extensions pour pouvoir justifier le choix de modélisation. Notre intérêt est d'avoir un modèle qui nous permet d'effectuer une analyse à la fois qualitative et quantitative du système ; un modèle générique qui peut être étendu pour des architectures multi-composants.

2.4.1. Automates à états finis

La représentation d'un SED dépend de l'aspect que l'on souhaite favoriser : une activité, un événement ou une entité (ressource, objet).

Les automates à états finis sont très utilisés dans le domaine de la spécification formelle des systèmes et appliqués dans les domaines de la théorie combinatoire, la compilation, la modélisation et la vérification des protocoles.

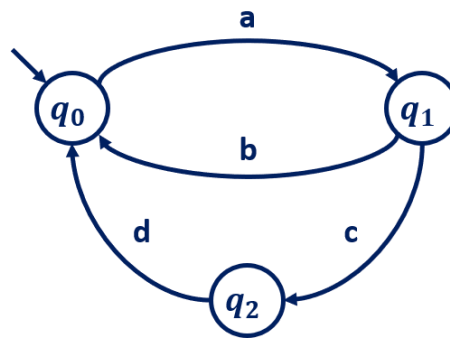


Figure 2.2: Exemple SED modélisé par un automate à états finis.

C'est un modèle orienté état dans lequel les activités sont considérées comme des états et les événements sont interprétés comme le changement de ces états. Les entités chargées d'exécuter les activités ne sont pas considérées. Les nœuds représentent les états et les arcs les événements (voir figure 2.2).

2.4.1.1. Définition formelle d'un automate à états finis

La définition formelle d'un automate à états finis, permet de décrire le langage et la méthode utilisée pour la construction du modèle d'un système. Il est défini par un 5-tuplet $G = (Q, \Sigma, \delta, q_0, Q_m)$, où :

- Q : ensemble fini d'états ;
- Σ : ensemble fini d'événements ;
- q_0 : état initial ;
- Q_m : ensemble des états finaux ou marqués ($Q_m \subseteq Q$) ;
- $\delta : Q \times \Sigma \rightarrow Q$: fonction de transition entre états.

A chaque état de G , la fonction de transition δ définit les événements dont l'occurrence est possible et l'état atteint suite à l'occurrence de chaque événement. Ainsi, à partir de l'état initial, on peut parcourir une séquence d'états qui dépend de la séquence d'événements exécutés.

L'ensemble de toutes les séquences (ou traces) d'événements exécutées par un automate à états finis est appelé langage de G , que l'on note $\mathcal{L}(G) \subseteq \Sigma^*$, Σ^* est l'ensemble des mots sur Σ , mot vide ε inclu. L'ensemble des séquences qui atteignent les états marqués (qui seraient dans notre cas des états de pannes) est appelé langage marqué et noté $\mathcal{L}_m(G)$. On a $\mathcal{L}_m(G) \subseteq \mathcal{L}(G)$.

La figure 2.2 représente un exemple d'automate à états finis qui contient trois états : état de repos (q_0), état de marche (q_1) et état de panne (q_2) avec $\Sigma = \{a, b, c, d\}$ l'alphabet de ce système. A l'état initial le système est au repos, l'occurrence de l'événement "a" met le système dans un état q_1 . Ainsi $\delta(q_0, a) = q_1$. L'événement "b" conduit le système dans un état de repos q_0 , l'occurrence de l'événement "c" qui est un événement défaillant, bascule le

système vers un état de panne q_2 et "d" représente l'événement de réparation ramenant ainsi le système vers l'état de repos q_0 . Ce modèle spécifie l'ordre des événements, à chaque état on peut conclure une information sur le passé (événement conséquent). Cette information est très utile pour déterminer l'évolution du système. Le modèle permet alors de représenter la séquentialité d'occurrence des événements conformément au comportement du système.

La définition formelle des automates à états finis va nous aider à décrire la séquentialité des événements dans le système et vérifier ses propriétés. Dans la section suivante nous introduisons un modèle temporisé des automates qui permet d'associer à chaque événement des instants d'occurrence.

2.4.1.2. Automates temporisés

[65] a proposé une extension des automates nommée "Automates temporisés" (AT) qui intègre l'aspect temporel par des horloges réinitialisables. Ces horloges indiquent l'écoulement du temps et des contraintes temporelles associées aux arcs pour symboliser les conditions de franchissement (changement d'état). L'occurrence d'un événement peut remettre à zéro ces horloges pour pouvoir mesurer le temps écoulé depuis le franchissement d'une transition. La figure 2.3 est un exemple d'automate temporisé.

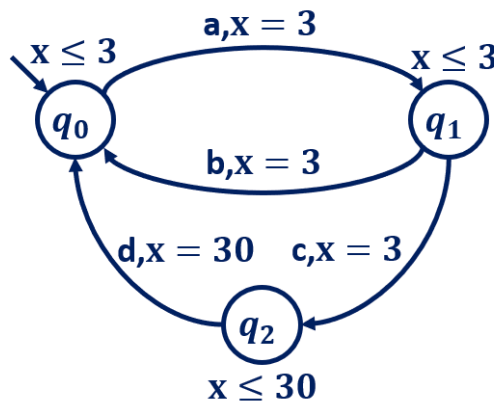


Figure 2.3: Automate temporisé.

Définition 2.1 : [65]

Un automate temporisé est un 6-uplet $\mathcal{A} = \langle Q, C, \Sigma, Inv, Edg, q_0 \rangle$ avec :

- Q : ensemble des états.
- C : vecteur de variables des horloges $[x_1, \dots, x_k]$, k représente le nombre d'horloges.
- Σ : Ensemble d'événements.
- Inv : fonction d'invariant qui associe des contraintes temporelles aux états. Ces contraintes sont définies comme une conjonction de forme $x_i \sim c$ avec $\sim \in \{<, \leq, \geq, >\}$, et c un nombre.
- Edg : ensemble des arcs définissant les conditions de transition entre les états à l'occurrence d'un événement. Un arc est un 5-uplet (l, g, e, λ, l') avec :
 - l et l' qui représentent respectivement l'état initial et l'état de destination de la transition.
 - g la garde de l'arc qui représente une contrainte numérique sur la valeur des horloges. Elle se définit comme une conjonction sous forme de $x_i \sim c$, ou $x_i - x_j \sim c$ avec x_i, x_j des horloges, c une constante, et $\sim$ symbolise un opérateur de comparaison tel que $\sim \in \{=, <, \leq, >, \geq\}$.
 - e un événement de Σ pour franchir l'arc.
 - λ : ensemble des horloges à remettre à zéro après le franchissement de l'arc.
- q_0 : l'état initial à partir duquel débutent toutes les exécutions de l'automate.

Les arcs déterminent les commutations d'états dans un AT. Ces commutations d'états sont conséquentes soit de l'occurrence d'un événement (transition événementielle) ou de l'écoulement d'une durée (transition temporelle par écoulement du temps dans un état).

L'automate temporisé de la [figure 2.3](#) décrit le comportement temporel d'un SED. L'horloge x permet de mesurer le temps écoulé dans chaque état discret du système. La contrainte $x \leq 30$ restreint le temps de séjour dans l'état q_2 à 30 unités de temps. Lorsque x atteint 30 unités de temps une transition sera franchie vers un état q_0 sur l'occurrence de l'événement "d". La garde de cette transition $x = 30$ est évidemment satisfaite par la valeur de x .

Le langage d'un AT est constitué de l'ensemble de ses séquences d'événements possibles. Chaque séquence d'événements est représentée par un vecteur d'horloges γ qui représente sa signature temporelle.

L'association d'états, événements et signature temporelle peut offrir un module d'aide à la décision après et avant défaillance. Après la défaillance, en effectuant une comparaison des signatures temporelles des séquences susceptibles d'amener le système à cet état de panne et donc faire appel à une méthode de diagnostic. Avant la panne, il serait possible à partir d'un état nominal de déterminer les exécutions des séquences qui peuvent conduire le système vers un état de panne et en déduire une durée de vie, un temps d'exécution ou une date d'occurrence d'un événement défaillant et donc exploitable pour pronostiquer la date d'occurrence d'un événement.

L'analyse des automates à états finis est limitée face au problème d'explosion combinatoire surtout lors de la modélisation des systèmes multi-composants. D'un point de vue logique, l'automate à états finis présente un aspect mono-horloge. Toutefois, l'aspect multi-horloge peut être obtenu en effectuant le produit de plusieurs automates, où chaque automate possède sa propre horloge logique, ce qui nécessite une synchronisation de ces horloges (mécanisme de rendez-vous). La modélisation par réseaux de Petri RdP peut palier à ces problèmes. En plus de leur puissance d'expression implicite, ils ont l'avantage d'intégrer l'aspect multi-horloge. Les RdP ne sont pas réduits à une expression d'états et permettent une modélisation plus compacte des systèmes. Ils ont un formalisme qui a l'avantage d'avoir une représentation graphique intuitive et facile à appréhender et donc à utiliser. Ils sont particulièrement bien adaptés pour décrire les aspects dynamiques ou comportementaux d'un système.

2.4.2. Réseaux de Petri

Les réseaux de Petri (RdP) sont des modèles graphiques abstraits et formels, introduits par Carl Adam PETRI en 1962 [\[66\]](#) et sont utilisés pour représenter des comportements concurrents, asynchrones ou non déterministes. Leur aspect graphique met en évidence et synthétise le comportement du système modélisé et leur aspect mathématique permet une analyse des équations d'état du système pour en déduire ses propriétés.

Les réseaux de Petri prennent en considération les trois aspects des SED évoqués au début de la [section 2.4.1](#) ; les places représentent les activités, les transitions représentent les événements et les jetons représentent les entités. L'ensemble des états accessibles sont décrits implicitement pour contourner le problème d'explosion d'états (sauf dans le cas temporel) due aux énumérations explicites des états comme dans le cas des automates à états finis.

2.4.2.1. Définition formelle des Réseaux de Petri

Un réseau de Petri est un graphe orienté composé de places et de transitions. Les places symbolisent les conditions d'évolution du système et elles sont représentées par des cercles. Les transitions symbolisent les événements dont l'apparition modifie l'état du système et elles sont représentées par des barres. Les barres et les cercles sont reliés par des arcs orientés. Ces liens sont des applications d'incidence avant notée "*pré*" (reliant une place à une transition) et d'incidence après notée "*post*" (reliant une transition à une place). L'incidence évoque la modification des états après franchissement des transitions.

Un RdP est un 4-uplet $R = \langle P, T, \text{pré}, \text{post} \rangle$ où :

- P : un ensemble de places ;
- T : un ensemble de transitions avec $P \cap T = \emptyset$;
- $\text{pré} : P \times T \rightarrow \mathbb{N}$, l'ensemble fini des arcs entrants ;
- $\text{post} : T \times P \rightarrow \mathbb{N}$, l'ensemble fini des arcs sortants.

A chaque arc est associé un poids $n \in \mathbb{N}$, si ce poids n'est pas mentionné sur l'arc cela signifie que le poids $n = 1$.

La présence de jetons dans une place signifie que la *pré* ou *post* condition est réalisée. Le marquage m est une application de l'ensemble des places vers $\mathbb{N}$, il est représenté par un vecteur de dimension $|P|$ (le nombre fini de places du réseau) dont chaque élément d'indice i représente le nombre de jetons contenus dans la place p_i . Ce marquage ou l'ensemble des places marquées exprime l'état du système. L'évolution du marquage traduit la dynamique du système.

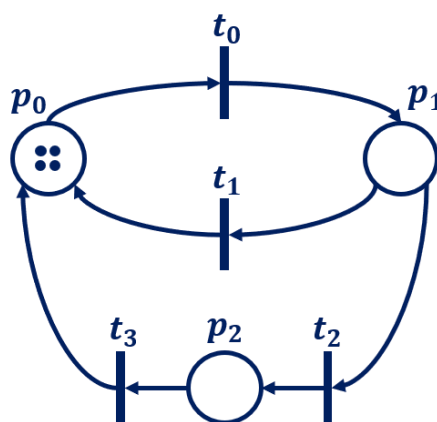


Figure 2.4: Réseau de Petri.

La figure 2.4 présente un modèle RdP avec 3 places (p_0, p_1, p_2) et 4 transitions (t_0, t_1, t_2, t_3). La place p_0 est marquée par 4 jetons. Dans ce mémoire on s'intéresse à des réseaux de Petri saufs, où au plus un jeton est possible dans chacune des places.

Les changements des états du système sont des conséquences des franchissements de transitions. Le franchissement d'une transition est instantané et ne peut avoir lieu que si la transition est validée : $m(p_i) \geq \text{pré}(p_i, t_j)$, cela consiste à retirer un (ou plusieurs selon le poids de l'arc) jeton(s) de la place p_i qui est en entrée de la transition t_j puis de rajouter un (ou plusieurs) jeton(s) à la place p_k en sortie de la transition t_j . On note $m[t_j > m'$ avec m' le nouveau marquage du réseau. Cette notation signifie encore que m valide t_j et donne après son franchissement un nouveau marquage m' .

2.4.2.2. Réseaux de Petri temporels

Comme indiqué précédemment, notre objectif est d'analyser l'aspect qualitatif (ordre ou séquencement) et quantitatif (date) de l'occurrence d'événements dans le RdP modélisant le comportement d'un système. Les contraintes temporelles quantitatives ont été introduites dans plusieurs extensions des réseaux de Petri, où la relation entre événements et contraintes temporelles ont été associées aux transitions.

Les réseaux de Petri temporels sont introduits par Merlin [67], puis étudiés par Berthomieu [68], [69], Popova [70], [71], Berthelot [72], Aalst [73] etc.

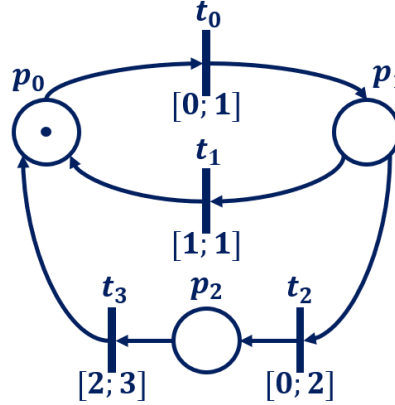


Figure 2.5: Réseaux de Petri Temporel.

Définition 2.2 :

Un réseau de Petri Temporel (RdPT) est un réseau $R_T = \langle P, T, \text{pré}, \text{post}, m_0, I \rangle$, dans lequel $\langle P, T, \text{pré}, \text{post}, m_0 \rangle$, est un réseau de Petri marqué, et $I: T \rightarrow \mathbb{Q}^+ \times (\mathbb{Q}^+ \cup \{\infty\})$ est une fonction de temps associant un intervalle statique $[\min(t), \max(t)]$ à chaque transition t , avec $\min(t) \leq \max(t)$ ($\max(t)$ pouvant être infinie) et $\mathbb{Q}^+$ est l'ensemble des nombres rationnels positifs ou nuls.

Une transition t peut être franchie au temps τ si le temps écoulé depuis l'activation de t appartient à l'intervalle $I(t)[\min(t), \max(t)]$.

Si $\min(t) = \max(t) = 0$ (i.e., $I(t) = [0, 0]$) la transition t est dite immédiate sinon elle est dite temporisée.

La figure 2.5 présente un exemple de RdPT où à chacune des transitions t_0, t_1, t_2 et t_3 est associé un intervalle temporel, indiquant les contraintes temporelles de leur franchissement.

Ainsi on peut diviser l'ensemble T en deux sous-ensembles T^t et T^{im} [74] où T^t est l'ensemble des transitions temporisées et T^{im} est l'ensemble des transitions immédiates avec : $T^t \cap T^{im} = \emptyset$ et $T^t \cup T^{im} = T$. Les transitions de l'ensemble T^{im} impliquent la prise en considération d'états instables dans le graphe d'accessibilité.

2.4.2.2.1. Etats et relations d'accessibilité

La construction d'un graphe d'état pour un RdP temporel est souvent complexe ; le franchissement d'une transition dans un RdP temporel labellisé RdPTL est possible à tout instant dans son intervalle de tir, ce qui génère une infinité d'états. Une première solution restreinte au modèle temporel borné avec une évolution du temps continu a été proposé par Berthomieu. Son but est de contracter cet espace d'états par un graphe de classes fini.

Définition 2.3 :

Selon [75], un état d'un réseau temporel est un couple $Q = (m, I)$ dans lequel m est un vecteur marquage et I est un ensemble d'intervalles associés aux transitions validées par rapport à m . L'état initial est constitué du marquage initial m_0 et de l'application I_0 qui associe à chaque transition sensibilisée son intervalle statique :

$$q_0 = (m_0, I_0), \text{ si } m_0 \geq \text{pré}(\bullet, t_k) \text{ alors } I_0(t_k) = I(t_k) \text{ sinon } I_0(t_k) = \emptyset.$$

Toute transition sensibilisée doit être tirée dans l'intervalle de temps qui lui est associé. Cet intervalle est relatif à la date de sensibilisation de la transition. Franchir t , à une date relative θ , depuis un état $Q = (m, I)$, est donc permis si et seulement si :

$$m \geq \text{pré}(\bullet, t) \wedge \theta \in I(t) \wedge \forall t_k \neq t, m \geq \text{pré}(\bullet, t_k), \Rightarrow \theta \leq \max(I(t_k))$$

L'état $Q' = (m', I')$ atteint depuis Q par le tir de t à θ est alors déterminé par :

$$1) m' = m + \Delta t \text{ avec } \Delta t = -\text{pré}(\bullet, t) + \text{post}(\bullet, t)$$

2) Pour chaque transition t_k :

- Si t_k est non sensibilisée par m , alors $I'(t_k) = \emptyset$;
- Si t_k est distincte de t , sensibilisée par m , et non en conflit avec t , alors $I'(t_k) = [\max(0, \min(I(t_k)) - \theta), \max(I(t_k)) - \theta]$;
- Sinon $I'(t_k) = I(t_k)$.

Il est important de mentionner que les événements de défaillance sont souvent associés à des transitions de l'ensemble T^{im} , d'où l'intérêt de la distinction des transitions.

[75] introduit ensuite la notion de classe d'états $C = (m, D)$ où D représente un domaine de tir (un vecteur de dates). Ce domaine de tir exprime l'ensemble des solutions du système d'inéquations linéaires avec les intervalles associés aux transitions sensibilisées. L'ensemble des états sont représentés par un vecteur de marquage m avec : $\{(m, D) | \exists \sigma, m_0 [\sigma > m, C_0(m_0, D_0) \xrightarrow{\sigma} C(m, D)]\}$ où : σ est la séquence d'événements et $C_0 \xrightarrow{\sigma} C$ ou $(m_0, D_0) \xrightarrow{\sigma} (m, D)$ dénote que m est accessible à partir de m_0 par l'exécution d'une séquence d'événements σ .

Exemple :

Soit le modèle RdPT de la figure 2.6, où $P = \{p_1, p_2, p_3, p_4, p_5\}$ et $T = \{t_1, t_2, t_3, t_4, t_5, t_6\}$.

Si on suppose une séquence de transitions $\sigma = t_1 t_3 t_4 t_2 t_3$.

$c_0 = (m_0, D_0)$; c_0 est la classe d'état initiale.

$m_0 : (p_1, 2p_2, p_3)$; est le marquage initial.

D_0 : l'ensemble solution en (t_1, t_3, t_5) du système

$$0 \leq t_1 \leq 2,$$

$$0 \leq t_3 \leq 5 ;$$

$$4 \leq t_5 \leq 5 ;$$

- Le tir de t_1 depuis c_0 à une date $\theta_1 \in [0, 2]$, mène à $c_1 = (m_1, D_1)$:

$c_1 = (m_1, D_1)$, avec :

$$m_1 = (2p_2, 2p_3, 2p_4) ;$$

D_1 : l'ensemble solution en (t_3, t_5) du système

$$\max(0, 0 - \theta_1) \leq t_3 \leq 5 - \theta_1,$$

$$\max(0, 4 - \theta_1) \leq t_5 \leq 5 - \theta_1 ;$$

- Le tir de t_3 depuis c_1 à une date $\theta_2 \in [0, 5]$, mène à $c_2 = (m_2, D_2)$:

$c_2 = (m_2, D_2)$, avec :

$m_2 = (3p_2, 2p_3, 2p_4, p_5)$;

D_2 : l'ensemble solution en (t_3, t_4, t_5, t_6) du système

$\max(0, 0 - \theta_2) \leq t_3 \leq 5 - \theta_2$,

$\max(0, 0 - \theta_2) \leq t_4 \leq 2 - \theta_2$,

$\max(0, 4 - \theta_2) \leq t_5 \leq 5 - \theta_2$,

$\max(0, 3 - \theta_2) \leq t_6 \leq 6 - \theta_2$;

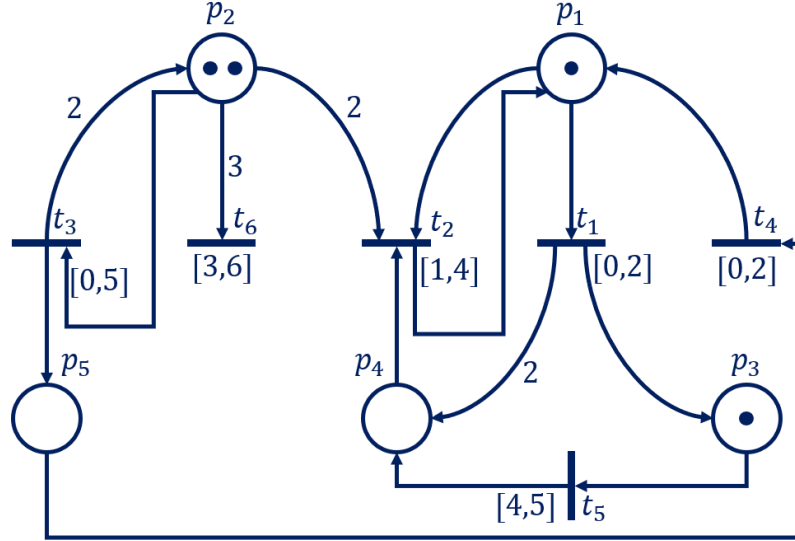


Figure 2.6: Exemple de réseau de Petri Temporel.

- Le tir de t_4 depuis c_2 à une date $\theta_3 \in [0,2]$, mène à c_3 :

$c_3 = (m_3, D_3)$, avec :

$m_3 = (p_1, 3p_2, 2p_3, 2p_4)$;

D_3 : l'ensemble solution en $(t_1, t_2, t_3, t_5, t_6)$ du système

$\max(0, 0 - \theta_3) \leq t_1 \leq 2 - \theta_3$,

$\max(0, 1 - \theta_3) \leq t_2 \leq 4 - \theta_3$,

$\max(0, 0 - \theta_3) \leq t_3 \leq 5 - \theta_3$,

$\max(0, 4 - \theta_3) \leq t_5 \leq 5 - \theta_3$,

$\max(0, 3 - \theta_3) \leq t_6 \leq 6 - \theta_3$;

- Le tir de t_2 depuis c_3 à une date $\theta_4 \in [1,4]$, mène à c_4 :

$c_4 = (m_4, D_4)$, avec :

$m_4 = (p_1, p_2, 2p_3)$;

D_4 : l'ensemble solution en (t_1, t_3, t_5) du système

$\max(0, 0 - \theta_4) \leq t_1 \leq 2 - \theta_4$,

$\max(0, 0 - \theta_4) \leq t_3 \leq 5 - \theta_4$,

$\max(0, 4 - \theta_4) \leq t_5 \leq 5 - \theta_4$;

- Le tir de t_3 depuis c_4 à une date $\theta_5 \in [0,5]$, mène à c_5 :

$c_5 = (m_5, D_5)$, avec :

$m_5 = (p_1, 2p_2, 2p_3)$;

D_5 : l'ensemble solution en (t_1, t_3, t_5) du système

$\max(0, 0 - \theta_5) \leq t_1 \leq 2 - \theta_5$,

$\max(0, 0 - \theta_5) \leq t_3 \leq 5 - \theta_5$,

$\max(0, 4 - \theta_5) \leq t_5 \leq 5 - \theta_5$;

Le délai de réalisation de la séquence $t_1 t_3 t_4 t_2 t_3$ correspond aux dates de franchissement des transitions telles que :

$$\begin{aligned}
0 &\leq t_1 \leq 2, \\
0 &\leq t_3 \leq 5 - \theta_1, \\
0 &\leq t_4 \leq 2 - \theta_2, \\
\max(0, 1 - \theta_3) &\leq t_2 \leq 4 - \theta_3, \\
0 &\leq t_3 \leq 5 - \theta_4;
\end{aligned}$$

$\theta_1, \theta_2, \theta_3, \theta_4$ peuvent prendre toute valeur réelle dans leur intervalle respectifs.

Exemple : soit la séquence $t_1 t_3 t_4 t_2 t_3$, une séquence réalisable à partir de q_0 dans 2.5 UT (unité de temps), avec t_1 est franchissable à 1UT et t_4 à 1.5 et le franchissement des autres transitions (t_2, t_3) est immédiat.

[76] interprète un état d'un RdPT comme un couple $Q = (m, h)$ dans lequel m est un marquage de places (noté p_marquage) et h est un vecteur horloge (de dimension égale au nombre de transitions du réseau) qui correspond aux marquages des transitions (noté t_marquage). Ainsi, le p_marquage décrit la situation des places et le t_marquage celle des transitions. Une telle paire (p_marquage, t_marquage) décrit un état du RdPT.

Définition 2.4 : [76]

Soit R_T un RdPT.

- Un p_marquage dans R_T est une fonction $m: P \rightarrow \mathbb{N}$.
- Un t_marquage dans R_T est une fonction $h: T \rightarrow \mathbb{R}_0^+ \cup \{\$\}$

Définition 2.5 :

Soit $R_T = \langle P, T, \text{pré}, \text{post}, m_0, I \rangle$ un RdPT, m un p_marquage et h un t_marquage. $Q = (m, h)$ est appelé un état dans R_T ssi :

- 1) m est un marquage accessible dans $\mathbb{R}$.
- 2) $\forall t ((t \in T \wedge \text{pré}(\bullet, t) \not\leq m \rightarrow h(t) = \$)$.
- 3) $\forall t ((t \in T \wedge \text{pré}(\bullet, t) \leq m \rightarrow h(t) \in \mathbb{R}_0^+ \wedge h(t) \leq \max(t))$.

La définition 2.5 montre que chaque transition t a une horloge. Si t n'est pas sensibilisée par le marquage m , l'horloge associée n'est pas enclenchée (signe \$). Si la transition t est sensibilisée par m , son horloge indique le temps écoulé depuis sa dernière activation.

L'état initial est donné par $q_0 = (m_0, h_0)$ avec $h_0(t) = \begin{cases} 0 & \text{ssi } \text{pré}(\bullet, t) \leq m_0 \\ \$ & \text{ssi } \text{pré}(\bullet, t) \not\leq m_0 \end{cases}$

Une transition t est franchissable à partir de l'état $Q = (m, h)$ (noté $Q[t >]$ ssi $\text{pré}(\bullet, t) \leq m$ et $h(t) \geq \min(t)$). Le franchissement de la transition t conduit R_T vers un nouvel état $Q' = (m', h')$ (noté $Q[t > Q']$).

La construction du graphe d'accessibilité d'un RdP Temporel est en général impossible. En effet, les transitions peuvent être tirées à tout instant dans leur intervalle de tir. Les états admettront ainsi une infinité de successeurs. De ce fait, les classes d'états sont introduites pour réduire cet espace d'états et fournir une représentation finie du graphe d'accessibilité [75], [76] propose une description paramétrique pour réduire cet espace d'états sans affecter les propriétés du réseau. Cet espace d'états réduit permet de définir le graphe d'accessibilité d'un RdPT.

2.4.2.2. Etat et séquence paramétriques

Soit R_T un RdPT arbitraire. Soit $\sigma = t_1 \dots t_n$ une séquence de franchissement dans R_T et soit $\tau = \tau_0 \dots \tau_n$ une séquence de temps avec $\tau_i \in \mathbb{R}^{*+}$. Alors il existe au moins une séquence datée $\sigma(\tau) = \tau_0 t_1 \tau_1 \dots \tau_{n-1} t_n \tau_n$ de σ dans R_T appelée séquence réalisable de σ qui conduit le réseau de l'état initial q_0 vers l'état q (noté $q_0[\sigma(\tau) > q]$) avec $q = (m, h)$.

Considérons par exemple la séquence suivante conduisant le réseau de l'état initial q_0 vers un état $q_n: q_0[2.0 > q'_0[t_1 > q_1[2.3 > q'_1[t_2 > \dots [1.5 > q'_n]$. Les valeurs 2.0, 2.3 et 1.5 expriment les dates de franchissement des transitions.

Ainsi, il est évident qu'il existe une infinité de séquences exécutables conduisant R_T de q_0 vers q , ce qui rend le graphe d'accessibilité infini. Au lieu de considérer des τ_i fixes, on utilise une variable x_i pour dénoter le temps écoulé entre le franchissement de la transition t_i et la transition t_{i+1} dans σ . Ainsi au lieu d'avoir une infinité de séquences d'exécutions entre les états q_0 et q_n , on étudiera une seule séquence que l'on appellera séquence paramétrique $\sigma(x) = x_0 t_1 x_1 t_2 \dots x_{n-1} t_n x_n$ conduisant le réseau de l'état q_0 vers l'état q_n avec :

$$q_n: q_0[x_0 > q'_0[t_1 > q_1[x_1 > q'_1[t_2 > \dots t_n > q_n[x_n > q'_n]$$

Définition 2.6 : (état paramétrique et séquence paramétrique)

Soient $R_T = \langle P, T, \text{pré}, \text{post}, m_0, I \rangle$ un RdPT, $\sigma = t_1 \dots t_n$ une séquence de franchissements dans R_T et $\sigma(x) = x_0 t_1 x_1 t_2 \dots x_{n-1} t_n x_n$. Alors, l'exécution paramétrique $(\sigma(x), B_\sigma)$ de σ et l'état paramétrique (q_σ, B_σ) dans R_T sont déterminés par :

* lorsque $\sigma = \varepsilon$, i.e., $\sigma(x) = x_0$.

Alors $q_\sigma = (m_\sigma, h_\sigma)$ et β_σ sont donnés par :

- 1) $m_\sigma := m_0$
- 2) $h_\sigma(t) = \begin{cases} x_0 & \text{si } \text{pré}(\cdot, t) \leq m_0 \\ \$ \text{ sinon} \end{cases}$
- 3) $B_\sigma := \{0 \leq h_\sigma(t) \leq \max(t) \mid t \in T \wedge \text{pré}(\cdot, t) \leq m_\sigma\}$

* on suppose que q_σ et β_σ sont déjà définis pour la séquence $\sigma = t_1 \dots t_n$.

Pour $\sigma = t_1 \dots t_n t_{n+1} = \gamma t_{n+1}$, et $\sigma(x) = x_0 t_1 x_1 t_2 \dots x_{n-1} t_n x_n t_{n+1} x_{n+1}$ on pose

- 1) $m_\sigma := m_\gamma + \Delta t_{n+1}$,
- 2) $h_\sigma(t) = \begin{cases} - \$ \text{ si } \text{pré}(\cdot, t) \not\leq m_\sigma & \text{donc } t \text{ n'est pas sensibilisée par } m_\sigma \\ - h_\gamma(t) + x_{n+1} & \text{si } \text{pré}(\cdot, t) \leq m_\sigma \wedge \text{pré}(\cdot, t) \leq m_\gamma \wedge \\ & \bullet t_{n+1} \cap \bullet t = \emptyset \wedge t \neq t_{n+1} \\ "t \text{ était sensibilisée pour } m_\gamma \text{ et reste sensibilisée pour } m_\sigma" & \\ - x_{n+1} & \text{sinon} \quad "car t \text{ est nouvellement sensibilisée} " \end{cases}$
- 3) $B_\sigma := B_\gamma \cup \{\min(t_{n+1}) \leq h_\gamma(t_{n+1})\} \cup \{0 \leq h_\sigma(t) \leq \max(t) \mid t \in T \wedge \text{pré}(\cdot, t) \leq m_\sigma\}$.

$h_\sigma(t)$ est une somme de variables ($h_\sigma(t)$ est un t -marquage paramétrique), c'est un vecteur de fonctions linéaires : $h_\sigma(t) = f(x)$ avec $x := (x_0, \dots, x_{|\sigma|})$, β_σ est un ensemble de conditions (un système d'inéquations).

Exemple :

Reprenons le même modèle RdPT de la figure 2.6 et considérons la séquence de transition suivante : $\sigma = t_1 t_3 t_4 t_2 t_3$.

- Au début $\sigma = \varepsilon$;

$$m_0 = (p_1, 2p_2, p_3);$$

$$h_\sigma := (x_0, \$, x_0, \$, x_0, \$)^T;$$

$$B_\sigma := \{0 \leq x_0 \leq 2\};$$

- Pour $\sigma = t_1$;

$$m_1 = (2p_2, 2p_3, 2p_4);$$

$$h_\sigma := (\$, \$, x_0 + x_1, \$, x_0 + x_1, \$)^T;$$

$$B_\sigma := \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 0 \leq x_0 + x_1 \leq 5 \end{array} \right\};$$

- Pour $\sigma = t_1 t_3$;

$$m_2 = (3p_2, 2p_3, 2p_4, p_5);$$

$$h_\sigma := (\$, \$, x_2, x_2, x_0 + x_1 + x_2, x_2)^T;$$

$$B_\sigma := \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 0 \leq x_2 \leq 2 \\ 0 \leq x_0 + x_1 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 \leq 5 \end{array} \right\};$$

- Pour $\sigma = t_1 t_3 t_4$;

$$m_3 = (p_1, 3p_2, 2p_3, 2p_4);$$

$$h_\sigma := (x_3, x_3, x_2 + x_3, \$, x_0 + x_1 + x_2 + x_3, x_2 + x_3)^T;$$

$$B_\sigma := \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 0 \leq x_2 \leq 2 \\ 0 \leq x_3 \leq 2 \\ 0 \leq x_0 + x_1 \leq 5 \\ 0 \leq x_2 + x_3 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 + x_3 \leq 5 \end{array} \right\};$$

- Pour $\sigma = t_1 t_3 t_4 t_2$;

$m_4 = (p_1, p_2, 2p_3, p_4)$; $'t_1 = \{p_2\}$, $'t_2 = \{p_1, p_2\}$, $'t_3 = \{p_2\}$; donc t_1 est en conflit avec t_2 et t_3 , ainsi :

$$h_\sigma := (x_4, \$, x_4, \$, x_0 + x_1 + x_2 + x_3 + x_4, \$)^T;$$

$$B_\sigma := \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 0 \leq x_2 \leq 2 \\ 1 \leq x_3 \leq 2 \\ 0 \leq x_4 \leq 2 \\ 0 \leq x_0 + x_1 \leq 5 \\ 0 \leq x_2 + x_3 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 + x_3 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 + x_3 + x_4 \leq 5 \end{array} \right\};$$

- Pour $\sigma = t_1 t_3 t_4 t_2 t_3$;

$$m_5 = (p_1, 2p_2, 2p_3, p_4, p_5);$$

$$h_\sigma := (x_4 + x_5, x_5, x_5, x_5, x_0 + x_1 + x_2 + x_3 + x_4 + x_5, \$)^T;$$

$$B_\sigma := \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 0 \leq x_1 \\ 0 \leq x_2 \leq 2 \\ 1 \leq x_3 \leq 2 \\ 0 \leq x_4 \leq 2 \\ 0 \leq x_5 \leq 5 \\ 0 \leq x_0 + x_1 \leq 5 \\ 0 \leq x_2 + x_3 \leq 5 \\ 0 \leq x_4 + x_5 \leq 2 \\ 0 \leq x_0 + x_1 + x_2 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 + x_3 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 + x_3 + x_4 \leq 5 \\ 0 \leq x_0 + x_1 + x_2 + x_3 + x_4 + x_5 \leq 5 \end{array} \right\};$$

Alors la séquence datée $\sigma = x_0 t_1 x_1 t_3 x_2 t_4 x_3 t_2 x_4 t_3 x_5$ est exécutable avec :

$$0 \leq x_0 \leq 2, 0 \leq x_1, 0 \leq x_2 \leq 2, 1 \leq x_3 \leq 2, 0 \leq x_4 \leq 2, 0 \leq x_5 \leq 5, x_0 + x_1 + x_2 + x_3 + x_4 + x_5 \leq 5;$$

Pour montrer la réduction de l'espace d'états proposée par [76], nous avons choisi le modèle RdPT de la figure.

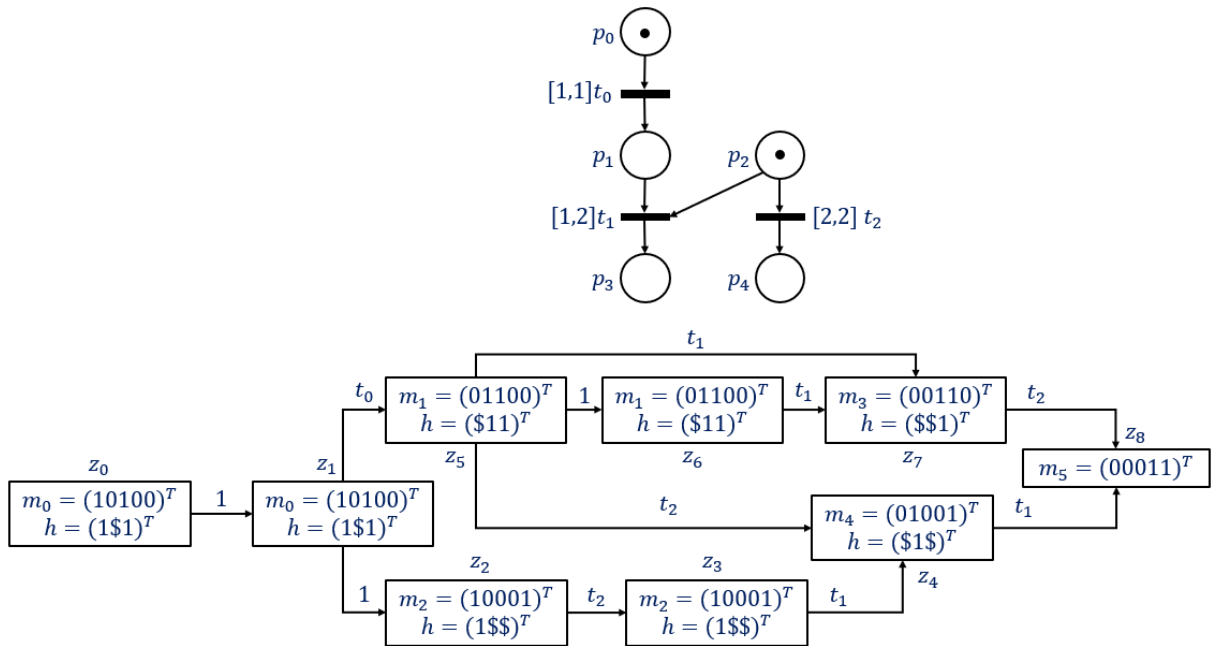


Figure 2.7: RdPT et Graphe d'états.

Les graphes d'états d'un RdPT sont généralement infinis. La figure 2.7 représente le RdPT et son graphe d'accessibilité. La figure 2.8 représente une abstraction d'états du graphe d'accessibilité pour éviter l'énumération infinie des états. Une abstraction qui conserve le comportement du modèle RdPT pour vérifier les marquages accessibles et les traces temporelles.

L'approche POPOVA permet non seulement de réduire l'espace d'états du système (en ne considérant que les états essentiels [71]), mais aussi de déterminer le temps minimal nécessaire pour arriver à chaque état. Cette solution qui consiste à utiliser les signatures temporelles des séquences d'événements, dans un système d'inéquations qui exploite l'interprétation paramétrique des états et des séquences (espace d'état réduit), pourrait pronostiquer la date d'occurrence d'événement de défaillance au plus tôt.

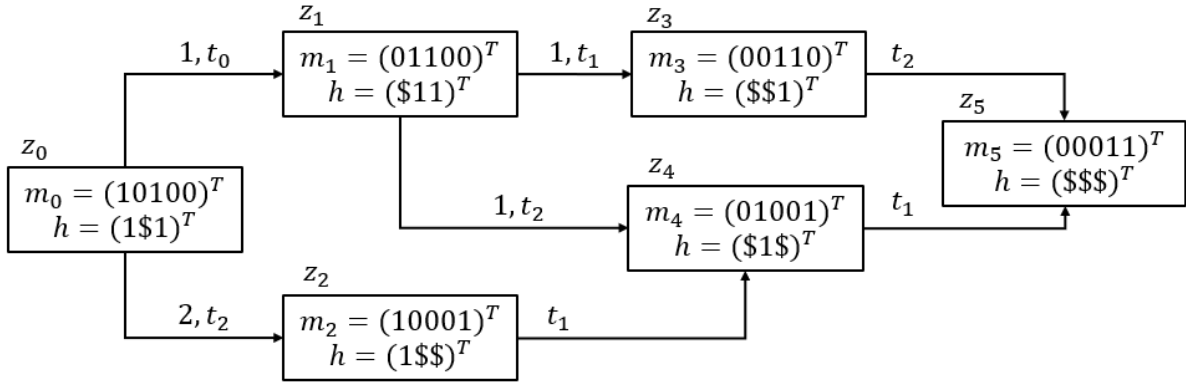


Figure 2.8: Graphe d'états du RdPT selon POPOVA.

On remarque que les transitions des états dans le modèle de la figure 2.6 sont basées sur des intervalles de franchissement. Ce qui a permis à l'aide des systèmes d'inéquations proposés par BERTHOMIEU et POPOVA et en fonction des instants de tir de chaque transition, de calculer la durée minimale d'exécution d'une séquence de transitions. Pour pouvoir analyser l'occurrence des événements et particulièrement ceux qui sont défaillants, il serait nécessaire de les représenter sur le modèle. Les RdP proposent une extension nommée réseaux de Petri labellisés qui permettent d'associer aux transitions du modèle des événements.

Dans le cas où l'un des intervalles a une borne maximale égale à l'infini, nous allons considérer lors du calcul du système d'inéquations sa borne minimale.

2.4.2.3. Réseaux de Petri labellisés

Les réseaux de Petri labellisés (RdPL) sont des réseaux de Petri étendus par une fonction de labellisation $\mathcal{L} : T \rightarrow \Sigma$, où Σ est l'ensemble d'événements, avec $\Sigma = \Sigma_o \cup \Sigma_u$. Σ_o est l'ensemble des événements observables et Σ_u est l'ensemble des événements non observables. La fonction de labellisation associe un label aux transitions. Le même label peut être affecté à des transitions différentes. Cette représentation définit la séquentialité (logique) de l'ensemble des événements sur le modèle.

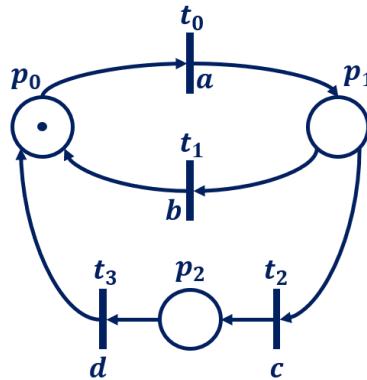


Figure 2.9: Réseau de Petri Labellisé.

Définition 2.7 : (Réseau de Petri labellisé) [74]

Un réseau de Petri Labellisé (RdPL) est un réseau $R_L = \langle P, T, \text{pré}, \text{post}, m_0, \Sigma, \mathcal{L} \rangle$ dans lequel $\langle P, T, \text{pré}, \text{post}, m_0 \rangle$, est un RdP, Σ est l'ensemble des labels associés aux transitions, $\mathcal{L} : T \rightarrow$

$\Sigma \cup \{\varepsilon\}$ est la fonction de labellisation des transitions associant un label (événement) $e \in \Sigma \cup \{\varepsilon\}$ à chaque transition $t \in T$, avec ε l'événement vide. Ainsi : $\mathcal{L}(t) = e$ signifie que le label associé à la transition t est e .

La figure 2.9 présente un exemple de RdPL où à chacune des transitions t_0, t_1, t_2 et t_3 est associé un événement, indiquant que l'occurrence d'événement est produite par le franchissement de la transition sensibilisée à laquelle il est associé.

Dans ce mémoire nous supposons que le même label $e \in \Sigma$ peut être associé à plusieurs transitions, c'est à dire, dans un RdPL, deux transitions t_i et t_j avec $t_i \neq t_j$ peuvent être labellisées avec le même événement e .

Soit Σ^* l'ensemble de toutes les chaînes d'événements de Σ contenant le label ε , la fonction de labellisation des transitions $\mathcal{L}$ peut être étendue aux séquences : $\mathcal{L}: T^* \rightarrow \Sigma^*$ tel que :

- (i) si $t_j \in T$ alors $\mathcal{L}(t_j) = e_k$ pour $e_k \in \Sigma$;
- (ii) si $\sigma \in T^* \wedge t_j \in T$ alors $\mathcal{L}(\sigma t_j) = \mathcal{L}(\sigma).\mathcal{L}(t_j)$;

De plus, si $\mathcal{L}(\lambda) = \varepsilon$ alors λ est la séquence vide.

Soit ω l'ensemble des événements de σ c'est-à-dire, $\omega = \mathcal{L}(\sigma)$. Le langage généré par le RdPL R_L est $\mathcal{L}(R_L) = \{\omega \in \Sigma^* | (\exists \sigma, m_0 [\sigma >) \mathcal{L}(\sigma) = \omega\}$. Ainsi, $m_1[\omega > m_2$ signifie que $\exists \sigma \in T^*$, $\mathcal{L}(\sigma) = \omega = e_1 e_2 \dots e_n$ et à partir de m_1 et en franchissant σ on atteint m_2 . Notons que la longueur d'une séquence σ est supérieure ou égale au mot correspondant ω (c'est à dire $|\sigma| \geq |\omega|$). En effet, si σ contient n transitions étiquetées par ε , alors $|\sigma| = n + |\omega|$.

Étant donné l'ensemble des événements ω , la fonction de labellisation inverse $\mathcal{L}^{-1}(\omega)$ est l'ensemble $\{\sigma \in T^* | \mathcal{L}(\sigma) = \omega\}$.

Exemple : [74]

Soit l'alphabet suivant $\Sigma = \{e_1, e_2, e_3\}$, l'ensemble des transitions $T = \{t_1, t_2, t_3, t_4\}$ et la fonction de labellisation $\mathcal{L}$ telle que $\mathcal{L}(t_j) = \begin{cases} e_j & \text{si } j = \{1, 2\} \\ e_3 & \text{si } j = \{3, 4\} \end{cases}$

Considérons l'ensemble $\omega = \{e_1, e_3\}$, alors $\mathcal{L}^{-1}(\omega) = \{\{t_1, t_3\}, \{t_1, t_4\}\}$.

Ainsi avec les RdPL on parvient à déterminer l'ordre logique d'occurrence des événements et par conséquent, identifier ainsi les séquences d'événements qui pourrait se terminer avec un événement défaillant. Néanmoins, la séquentialité de ces événements est souvent confrontée à une ambiguïté lors du diagnostic ou du pronostic d'un événement défaillant; avoir deux séquences avec le même ordre d'occurrence des événements crée un indéterminisme. Dans la section suivante, nous allons présenter une autre extension de RdP avec un aspect temporel et voir son apport dans le pronostic d'un événement défaillant.

2.4.2.4. Réseaux de Petri temporels labellisés

Les réseaux de Petri Temporels labellisés sont introduits pour leur capacité de représentation à la fois des événements et leurs dates d'occurrences sur le modèle. Les séquences d'événements seront datées, ce qui offre une signature temporelle pour chaque séquence d'événements.

Un RdP-TL est une extension des RdP temporels pour laquelle à chaque transition est associée à un événement qui peut être observable ou non [77]–[79].

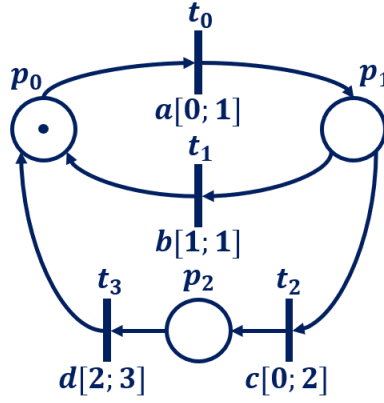


Figure 2.10: Réseau de Petri temporel labellisé.

La figure 2.10 présente un réseau de Petri labellisé temporel. A chaque transition du réseau on associe un intervalle de franchissement ($\min(t)$ et $\max(t)$). La transition t_2 est franchissable à l'occurrence de l'événement " c " dans un intervalle $([0, 2]$ pas $[2, 3])$.

Définition 2.8 : [74], [78]

Un RdPTL est un réseau $R_{TL} = \langle P, T, \text{pré}, \text{post}, m_0, \Sigma, \mathcal{L}, I \rangle$ dans lequel $\langle P, T, \text{pré}, \text{post}, m_0 \rangle$, est un réseau de Petri marqué, Σ est l'ensemble des labels associés aux transitions, $\mathcal{L}$ est la fonction de labellisation des transitions et I est la fonction associant un intervalle statique temporel à chaque transition.

La figure 2.10 présente un exemple de RdPTL où à chacune des transitions t_0, t_1 et t_2 est associé un événement et intervalle temporel, indiquant que l'occurrence de l'événement dans la contrainte temporelle associée à la transition conditionne son franchissement.

Un RdPTL peut être traité comme un RdPT dont chaque transition est labellisée avec un événement de $\Sigma \cup \{\varepsilon\}$. Un changement d'état de RdPTL peut se produire soit sur un franchissement de transition soit sur une durée de temps écoulée. Ici, la définition d'état et sa fonction de transition sont les mêmes que pour un RdPT selon l'approche POPOVA présentée dans la section précédente [76].

Définition 2.9 : (Langage généré par un RdPTL)

Soit $\sigma = t_1 \dots t_n$, une séquence de franchissement dans le réseau RdPTL et soit $\sigma(x)$ sa séquence datée réalisable, avec $\sigma(x) = x_0 t_1 x_1 \dots x_{n-1} t_n x_n$ [76].

On note par $\text{Timed}(\sigma)$ la séquence réalisable $\sigma(x) : \text{Timed}(\sigma) = \sigma(x)$. Réciproquement, on note par $\text{Logic}(\sigma(x))$ la séquence de franchissement dans le réseau : $\text{Logic}(\sigma(x)) = \sigma \in T^*$.

Il serait alors intéressant de retrouver l'usage des RdPTL dans l'élaboration des diagnostiqueurs voire pronostiqueurs. Le diagnostiqueur et le pronostiqueur seront introduits plus en détails dans les prochaines sections. Ils sont des modèles générés à partir du graphe d'accessibilité qui permettent (de prédire) d'identifier la cause de l'occurrence d'un événement de défaillance dans le cas du (pronostic) diagnostic. Avec l'approche de paramétrisation proposée par POPOVA l'espace d'état important dû à l'introduction de l'aspect temporel sera réduit et permettra ainsi la construction d'un modèle diagnostiqueur ou pronostiqueur compact.

Parmi les travaux que nous allons discuter dans la section dédiée au pronostic ([section 2.6](#)), il existe ceux qui utilisent une approche stochastique, d'où l'intérêt d'introduire les RdP stochastiques dans la section suivante.

2.4.2.5. Réseaux de Petri stochastiques

Les réseaux de Petri stochastiques sont parmi les outils formels et graphiques les plus utilisés pour l'analyse des performances des systèmes informatiques et industriels. Le processus stochastique est un modèle mathématique utile pour la description de phénomènes de nature probabiliste en fonction du temps. Ainsi, nous présentons dans cette section une méthode de modélisation dynamique avec des transitions aléatoires, pour déterminer la probabilité d'occurrence d'un événement défaillant dans un intervalle de temps donné.

Les réseaux de Petri stochastiques (RdPS) sont une extension de RdP dans lesquels on associe aux transitions des délais de franchissement aléatoires. Le franchissement de transition est atomique, c'est-à-dire que les jetons sont retirés des places d'entrée et remis dans les places de sortie avec une opération unique et indivisible. Les délais du franchissement des transitions représentent l'évolution de comportement du système sur une échelle temporelle. La spécification du délai de franchissement est de nature probabiliste et représentée par une variable aléatoire, de façon que la fonction de densité de probabilité (FDP) du délai associé à une transition doit être spécifiée [\[80\]](#).

Un réseau de Petri stochastique est un 6-tuplet $RdPS = (P, T, prè, post, m_0, \Lambda)$, où $(P, T, prè, post, m_0)$ est un réseau marqué et $\Lambda = \lambda_1, \lambda_2, \dots, \lambda_n$ est un ensemble de taux de tirs possibles associés aux transitions. La FDP associée à une transition t_i un taux de franchissement λ_i . Ce taux de franchissement peut dépendre du marquage, il faut donc l'écrire comme $\lambda_i(m_i)$. Le délai moyen de franchissement de la transition t_i pour le marquage m_i est $[\lambda_i(m_i)]^{-1}$. Si plusieurs transitions sont franchissables à la fois (conflit), le taux associé à chaque transition permet de désigner laquelle d'entre elles doit être franchie au premier. Ce qui permet de lever l'indéterminisme engendré par des transitions de l'ensemble T^{im} en conflit dans le cas des *RdPS* généralisés.

Ces probabilités peuvent être exploitées pour calculer la probabilité d'occurrence d'un événement de défaillance. Il faut identifier l'ensemble des séquences d'événements à risque potentiel de défaillance et calculer leurs probabilités causales. Cela consiste à analyser toutes les séquences d'événements possibles qui peuvent conduire le système à un événement de défaillances et les traduire par des probabilités.

Dans ce mémoire on s'intéresse à un comportement déterministe d'un modèle générique, pour déterminer la date d'occurrence d'un événement de défaillance future. L'étude probabiliste basée sur des variables aléatoire (environnement incertain) nécessite une prise en compte d'hypothèses sur des événements imprévisibles dits incompatibles ou indépendants que nous ne prendrons pas en considération à cette étape de travail. Elle devrait impliquer des développements plus conséquents que nous discuterons en perspectives.

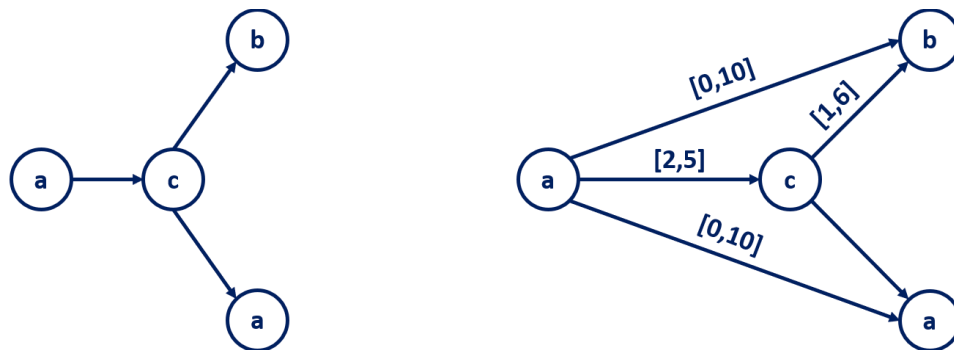
2.4.3. Les chroniques

C'est une approche qui exploite la succession des contraintes temporelles de l'occurrence des événements dans le système. Définie par [\[81\]](#), une chronique est un ensemble de motifs d'événements qui sont liés par des contraintes temporelles. Dans le cadre du diagnostic, elle est utilisée pour représenter les évolutions possibles du système pour chaque type de défauts ou pour modéliser les signatures temporelles causales. [\[82\]](#) propose un système de

reconnaissance à base de chronique capable d'identifier en temps réel tous les motifs correspondants [83], [84].

Un modèle de chronique est formé d'un ensemble d'observations et d'un ensemble de contraintes temporelles entre les instants d'occurrences de celles-ci. Cet ensemble de contraintes temporelles est représenté sous forme d'un graphe d'instant [85].

Exemple : [86]



Ordre entre les événements de la chronique La chronique représentant un comportement du système

Figure 2.11: Exemple d'une chronique.

L'exemple de la figure 2.11 représente l'évolution du système par rapport à l'occurrence ordonnée d'événements $\{a,b,c\}$. Le système de reconnaissance est basé sur la prévision des dates possibles pour l'occurrence de chaque événement qui n'a pas encore eu lieu. Les transitions sont étiquetées par $[\min(t), \max(t)]$ entre les événements pour représenter les contraintes temporelles de ces événements. L'événement c par exemple aura lieu entre 2 et 5 unités de temps après l'occurrence de l'événement a et cette occurrence de c précède l'occurrence d'un événement b avec 1 à 6 unités de temps, sachant qu'il y a une nouvelle possibilité d'occurrence de l'événement a dans moins de 10 unités de temps.

Le processus de reconnaissance gère un ensemble d'instances partielles de la chronique comme un ensemble de fenêtres de temps (une pour chaque événement à venir) qui est progressivement limité par chaque nouvel événement correspondant : ce système est prédictif dans le sens où il prédit les événements à venir qui sont pertinents pour les instances de chroniques en cours de développement ; il se concentre sur elles et maintient leurs fenêtres de temps.

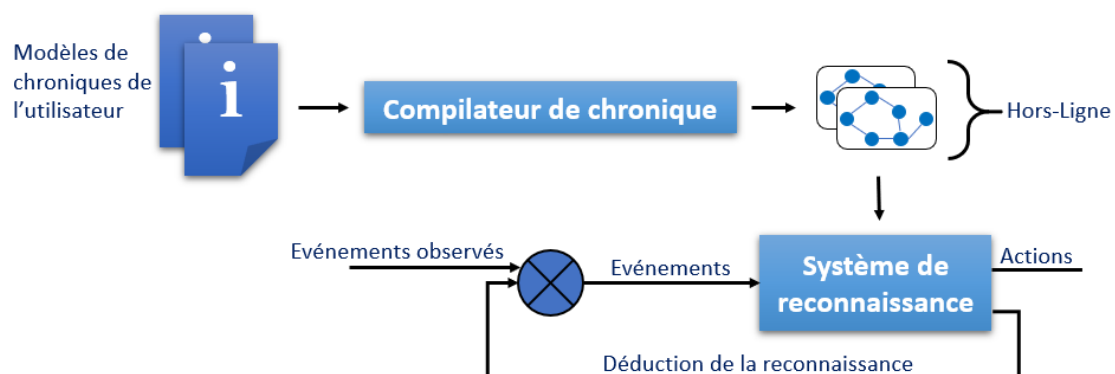


Figure 2.12: Principe général de la reconnaissance de chroniques.

La figure 2.12 présente le principe général de la reconnaissance de chronique. Les modèles de chroniques, fournis par l'utilisateur sont compilés en structures de données efficaces. Le module hors-ligne assure la vérification de la cohérence des modèles de chronique. Le système reçoit le flux d'événements observés, instanciés et datés.

La reconnaissance de chronique [86] offre un formalisme enrichi des modèles de chronique, qui permet de créer des scénarios génériques d'occurrence de plusieurs événements dans le même intervalle de temps.

La technique de chien de garde est exploitée dans une chronique pour détecter le dépassement de la date d'occurrence des événements. Ce dépassement de temps signale une divergence dans le comportement du système. Cette méthode est ainsi exploitée pour détecter les défaillances et identifie les signatures temporelle grâce aux chroniques.

Ainsi plusieurs techniques de représentation, possédant leur propre intérêt et hypothèses d'interprétation des chroniques à choisir selon le contexte et l'objectif. La caractéristique essentielle était certainement la performance des méthodes à l'imprévu ou en cas d'événements non observables.

L'observabilité dans les SED est représentée par une répartition des événements Σ du système en deux sous-ensembles $\Sigma = \Sigma_o \cup \Sigma_{uo}$: des événements observables Σ_o et des événements non observables Σ_{uo} qui représentent dans la plupart des cas des événements de défaillance. La non-observabilité des événements est due soit à un problème technologique (défaillance dans un ou des capteurs) soit financière (nombre de capteurs insuffisants). Une observation partielle dans un SED est de moins bonne qualité qu'une observation totale. En revanche, il est possible de surmonter ce manque d'observation par une reconstruction d'un modèle qui prend en considération la représentation de ce type d'événements.

Dans le cadre d'une approche orientée pronostic, il serait intéressant d'exploiter le formalisme du processus de reconnaissance de chronique pour vérifier l'évolution du système au lieu de revenir en arrière pour remonter aux événements étant à l'origine d'une défaillance. Cependant, il nécessite de récupérer l'ensemble séquences d'événements datés et déterminer parmi ces séquences celles qui se terminent par événement défaillant et qui sont certaines.

Nous avons choisi dans la section suivante, d'introduire le diagnostic et ses travaux avant de présenter les approches développées dans le pronostic. Les techniques utilisées pour l'identification de la causalité d'un événement défaillant, ont été source d'inspiration de plusieurs travaux développés dans le cadre d'une approche orienté pronostic.

2.5. Diagnostic des SED

2.5.1.Contexte

Dans les années 90 la complexité des systèmes s'accroît et les exigences en matière de performance et fiabilité augmentent. La détection et l'isolement des défaillances pour les systèmes complexes est devenue une tâche cruciale et difficile. Il était nécessaire de développer des méthodes sophistiquées et systématiques, pour un diagnostic rapide et précis des défaillances. Rappelons que la fonction du diagnostic est d'indiquer la cause d'un dysfonctionnement du système. Ce diagnostic est réalisé de deux façons, la première consiste à construire un graphe d'états orienté, appelé diagnostiqueur, afin d'extraire les ordres logiques et ou temporels d'occurrence des événements, afin de vérifier la propriété de diagnosticabilité (la possibilité de diagnostic en cas de séquence d'événements ambiguë, ...) des séquences apparues. La deuxième façon s'intéresse aux violations des spécifications et discute la propriété de la diagnosticabilité avant de construire un diagnostiqueur.

2.5.2. Diagnostiqueur

Parmi les diagnostiqueurs proposés on cite les travaux de :

[18] propose une approche de modélisation et de diagnostic basée sur une représentation logique des SED. Sur un modèle à automate à états finis, les fautes sont des événements non observables et incluent ainsi tout comportement nominal ou défaillant du système. A partir de ce modèle un diagnostiqueur est construit et déployé pendant le fonctionnement du système.

Le diagnostiqueur fournit une estimation de l'état courant du système et les fautes affectant son fonctionnement par l'observation des événements générés par le système. La figure 2.13 illustre le principe de cette approche. On représente les états du système sur un module diagnostiqueur dans la figure 2.14, par des étiquettes : F indique état de panne, N l'état nominal et A l'état ambiguë.

Ce diagnostiqueur est défini par : $D = (Q_D, \Sigma, \delta, q_0)$ où :

- q_0 est l'état initial du diagnostiqueur qui représente l'état nominal du système.
- δ est la fonction de transition à partir de l'état q_0 , définie par :
- Une fonction de propagation d'étiquettes $PE : Q \times \Delta \times \Sigma^* \rightarrow \Delta$:
 - $PE(q, l, \sigma) = \begin{cases} \{N\} & \text{si } l = \{N\} \\ \{A\} & \text{si } l = \{A\} \\ \{F\} & \text{si non} \end{cases}$
 - σ désigne une séquence d'événements.
 - Δ est l'ensemble des étiquettes $\Delta = \Delta_N \cup \Delta_A \cup \Delta_F$; Δ_N est l'ensemble des étiquettes nominales, Δ_A est l'ensemble des étiquettes ambiguës et Δ_F est l'ensemble des étiquettes de fautes.
 - l est un élément de Δ , tel que $l \in \{N, A, F\}$.
- Q_D représente les états atteignables à partir de q_0 par la fonction de transition δ . Q est l'ensemble des états du système.

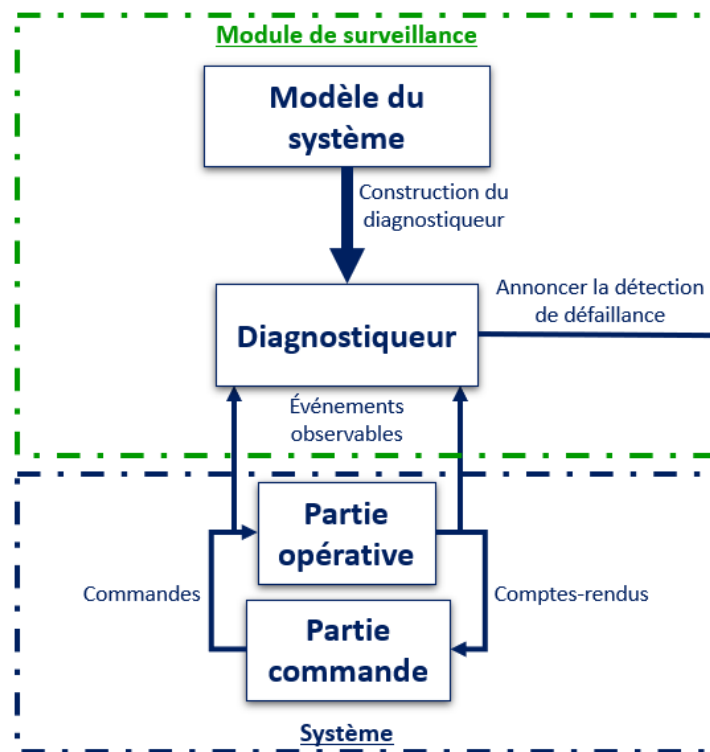


Figure 2.13: Interprétation de l'approche de Sampath.

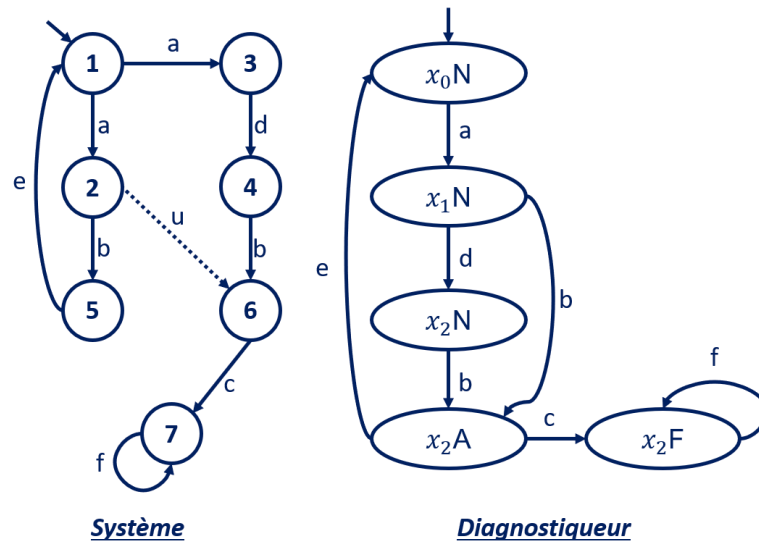


Figure 2.14: Un SED et son diagnostiqueur selon Sampath.

La difficulté de cette approche réside dans l'obligation d'initialiser le diagnostiqueur en même temps que le système pour qu'il puisse suivre son évolution. La partie innovante dans cette approche est la génération d'un module diagnostiqueur (observateur) à partir du modèle comportemental qui peut suivre l'évolution des événements produit par le système.

[87] propose d'améliorer la précision du diagnostic par l'utilisation des informations temporelles. Cela permet de réduire significativement les besoins en puissance de calcul en ligne et la taille du diagnostiqueur au détriment des calculs de conception hors ligne. Mais avant cela, un schéma de réduction du modèle avec un polynôme de complexité temporelle, est proposé dans l'intérêt de réduire la complexité du calcul lors de la conception du diagnostiqueur.

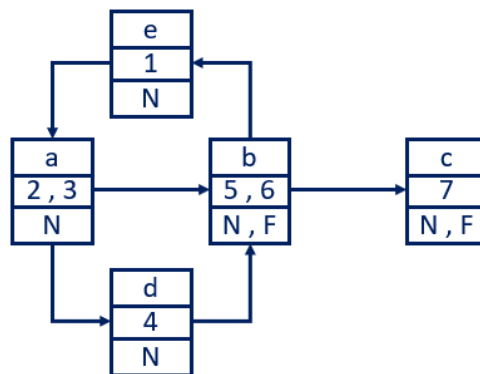


Figure 2.15: Diagnostiqueur selon l'approche de ZAD.

La figure 2.15 représente le diagnostiqueur du modèle de la figure 2.14 selon l'approche proposée par [88], chaque état du modèle diagnostiqueur est représenté par l'événement observé, les états d'évolution possibles X_i et les étiquettes (N pour nominal et F pour faute). Les arcs en pointillés dans la figure 2.14 représentent des événements de défaillance. Le Sommet $X_i\{N\}$ représente le sommet initial du diagnostiqueur. A partir du sommet $\{2,3\}\{N\}$ par exemple et après occurrence de l'événement "b" le diagnostiqueur évolue vers un sommet incertain $\{5,6\}\{N,F\}$, puisque l'évolution de l'état 1 vers l'état 2 admet deux trajets un avec un événement défaillant et l'autre non. L'occurrence d'un événement e fait évoluer le diagnostiqueur vers le sommet $\{7\}\{F\}$ certain où 7 est un état de panne. Tu es sûr que e fait évoluer le diagnostiqueur vers le sommet $\{7\}\{F\}$,

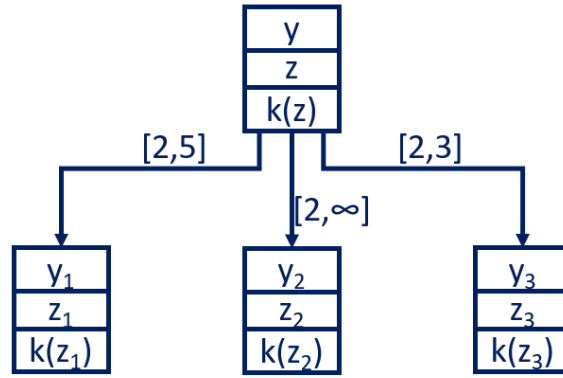


Figure 2.16: Modèle diagnostiqueur d'un SED temporel.

Ce diagnostiqueur (figure 2.16) est construit à partir d'un SED temporel et traite les tics d'horloge comme une information supplémentaire pour passer d'un sommet à un autre. Les symboles y , y_1 , y_2 , y_3 représentent les événements et z , z_1 , z_2 , z_3 les états et $k(z)$ est le comportement du système après occurrence de y . Les intervalles conditionnent le temps dans lequel le passage d'un sommet à un autre est possible ; par exemple le passage du sommet $(z, k(z))$ à un sommet $(z_1, k(z_1))$ est possible dans un intervalle de temps $[2, 5]$. Ce qui permet en plus d'avoir une visibilité sur l'ordre d'occurrence des événements, de déterminer date d'occurrence. Ce qui est intéressant dans ce modèle et ce qui peut nous être utile dans le développement de notre approche, c'est la signature temporelle des séquences d'événements qu'on peut extraire du modèle pour pouvoir déterminer le temps d'exécution d'une séquence au plus tôt et au plus tard. Sans oublier que la taille d'un tel diagnostiqueur dépend des contraintes temporelles représentées sur le modèle comportemental du système.

[89] propose une nouvelle variante de diagnostiqueurs qui sépare les états nominaux des états défaillants pour suivre leur évolution séparément et développe une procédure d'analyse de la diagnosticabilité basée sur un algorithme de vérification à la volée. Le diagnostiqueur construit à partir du modèle de la figure 2.17 illustre la représentation séparée des séquences défaillantes de celles nominales. Le cycle f -incertain est dû à l'existence d'un cycle normal en parallèle. L'algorithme à la volée est utilisé pour construire le diagnostiqueur et analyser les différentes conditions de diagnosticabilité en parallèle.

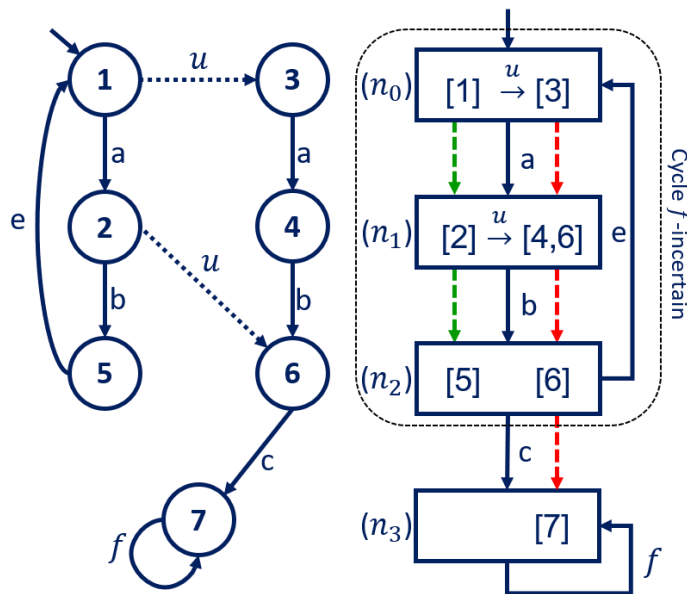


Figure 2.17: Diagnostiqueur Selon l'approche Boussif [89].

Ce qui est intéressant dans cette approche c'est la distinction des séquences défaillantes de celles nominales lors de la construction du diagnostiqueur, cela permet d'identifier rapidement les séquences ambiguës. La construction d'un diagnostiqueur temporel réduira considérablement ce problème d'ambiguïté, par une comparaison des signatures temporelles de séquences.

Pencolé [90] propose une formulation du problème du diagnostic de motifs d'un système temporel. Il a construit un diagnostiqueur qui prend en entrée une séquence temporelle d'événements observables et détermine si cette séquence est certaine, nominale ou ambiguë. A partir d'une fonction du diagnostic et dans un délai borné, il détermine si le comportement décrit par le motif s'est exécuté ou non pendant l'évolution du système. Cette fonction de diagnostic est basée sur la notion de concordance des séquences d'évaluation du système cohérentes avec un motif de comportement (cohérentes avec une séquence d'observation). Il considère le diagnostic de motif temporel comme un problème d'atteignabilité et utilise le modèle checking sur l'outil TINA (Time Petri Net Analyzer) pour le résoudre. Il commence par une modélisation des langages temporels du système et du motif de comportement par des réseaux de Petri temporels étiquetés et les combine pour appliquer la relation de concordance entre système et motif. Cette combinaison sera après synchronisée avec un réseau de Petri temporel représentant une séquence d'observations donnée.

[91] propose une approche décentralisée de diagnostic sur des automates temporisés pour éviter l'explosion combinatoire lors de la construction du modèle. Il utilise les contraintes temporelles afin de caractériser le fonctionnement normal, dégradé ou défaillant de chaque sous-système avec un certain degré de certitude représenté par un indicateur de dégradation. L'approche est réalisée en deux phases :

La première phase permet de construire un module de diagnostic hors ligne par un apprentissage des contraintes temporelles et elle est réalisée en trois étapes. La première étape consiste à décomposer le système pour réduire sa complexité. Il suppose que l'occurrence d'une défaillance dans un sous-système ne peut être propagée aux autres et que chaque défaillance est diagnosticable. La deuxième étape consiste à modéliser des observateurs des sous-systèmes à l'aide des contraintes temporelles par une transformation du modèle logique en Template et chroniques. Le Template est utilisé comme un graphe temporel correspondant à l'occurrence d'une séquence d'événements consécutive d'un événement généré par le contrôleur. La chronique correspond à une séquence d'événements qui permet d'identifier un événement de défaillance (signature d'un événement de défaillance). La troisième étape attribue des distributions de probabilité aux contraintes temporelles, pour exprimer la probabilité d'occurrence d'un événement à une date précise.

La deuxième phase met en œuvre le module de diagnostic construit pour détecter et localiser les événements de défaillances qui affecte le système. La première étape de cette phase consiste à récupérer les informations du système afin de suivre son évolution. Ensuite, procédé à la détection des défaillances à l'aide des conditions d'autorisation d'événements. La troisième étape est la localisation du défaut.

La notion de Template introduite dans les travaux de [91] détermine à partir de l'occurrence d'un événement l'ensemble des événements possibles qui le suivent associés à des intervalles de temps appropriés. Elle est utilisée principalement pour suivre l'évolution d'une défaillance. L'exploitation de cette technique afin de déterminer à partir de n'importe quel événement du réseaux les séquences possibles est intéressant. A partir de ces séquences il suffit de sélectionner celle qui sont défaillantes et à l'aide des chroniques en déduire sa signature temporelle.

[9] Exploite conjointement l'ordonnancement partiel des événements et les contraintes temporelles des défaillances. Elle vérifie la cohérence des signatures temporelles causales (STC introduites par [10]) afin de détecter les défaillances à partir de l'occurrence d'événements observables. La méthode de diagnostic basée sur les STC vérifie l'occurrence des événements et le respect de leurs contraintes temporelles afin de déterminer si l'état du système est normal ou défaillant. Elle propose aussi, un algorithme de correction de diagnostic avec STC multi-défaillances et montre qu'il est possible d'utiliser les STC pour le diagnostic au lieu de construire le diagnostiqueur. Cette approche est basée sur un ensemble de règles issues d'une expertise (arbres de défaillances) qui n'est pas toujours exhaustive ; l'oubli d'une règle par exemple peut provoquer l'absence de la détection d'une défaillance ou la génération d'une fausse alarme.

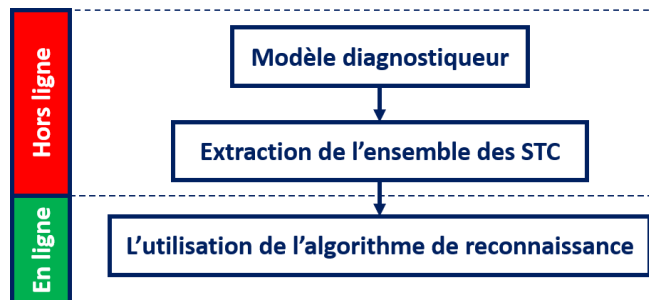


Figure 2.18: Approche de diagnostic SADDEM [92].

La figure 2.18 représente l'approche de diagnostic introduite dans [92] où est proposée une méthode pour garantir l'exhaustivité d'un ensemble des STC. La méthode est basée sur une traduction du formalisme et du modèle d'un diagnostiqueur en STC, pour palier à la complexité d'utilisation du diagnostiqueur pour des système temps réel. A partir de ces STC, un algorithme de reconnaissance basé sur le concept de " monde " est utilisé. Elle définit un " monde " comme un ensemble d'hypothèses cohérentes d'affectation de l'événement reçu par la tâche de diagnostic. La construction des STC à partir d'une abstraction comportementale du système semble intéressante dans une approche de pronostic à base temporelle. Néanmoins, il faudrait réadapter l'algorithme de reconnaissance pour qu'il puisse pronostiquer la (les) séquence(s) susceptible(s) de provoquer la défaillance au lieu de détecter la défaillance.

[93] s'intéresse aux réseaux de Petri partiellement mesurés (par rapport à la labellisation des transitions et aux marquages des places). Il construit un graphe d'accessibilité avec un nombre de nœuds inférieur à ceux du graphe des marquages, où chaque nœud du graphe est un couple (m, x) avec m le marquage atteint et x une valeur binaire qui désigne la présence ou non d'une ambiguïté (existence de deux séquences qui évoluent à partir du marquage m tel que l'une se termine par un événement de défaillance et l'autre non) sur l'occurrence d'une faute. A partir de ce graphe il construit un graphe appelé diagnostiqueur où chaque nœud contient une étiquette et un tag ; une étiquette N est associée au nœud si le marquage m est atteint sans l'occurrence d'une faute et l'étiquette F dans le cas contraire. Le tag est une variable entière qui mentionne la non-occurrence d'une faute par la valeur 1, l'ambiguïté par rapport à l'occurrence d'une faute par la valeur 2 et la certitude d'occurrence d'une faute par la valeur 3. Pour pallier le problème d'exploitation du graphe d'états dans l'analyse du modèle temporel, il utilise des classes d'états introduites dans les travaux de [69] (voir la définition 2.3 la section 2.4-2.2.1) regroupant à la fois le marquage et les contraintes temporelles. Il exploite l'ordre des événements, leurs dates et leurs probabilités d'occurrence pour calculer les trajectoires qui permettent de détecter l'occurrence des fautes. La notion de labellisation des états et la construction d'un modèle diagnostiqueur réduit la complexité d'analyse du modèle comportemental du système et donne une vue synthétique sur son évolution. La notion de

trajectoire qui fournit à la fois l'ordre, la date et la probabilité d'occurrence des événements renforce la qualité du diagnostic en termes de précision de la détection d'une faute.

2.5.3.Diagnosticabilité

La diagnosticabilité d'un système consiste à appliquer des algorithmes qui vérifient les propriétés formelles du système. Le but est de s'assurer que le modèle diagnostiqueur, contient l'ensemble des événements observables à l'avance qui permettent de détecter et de distinguer l'ensembles des défaillances possibles sur le système. L'étude de la diagnosticabilité permet de savoir si les événements observables du système sont suffisants pour distinguer, le fonctionnement normal du fonctionnement fautif. Dans un SED la diagnosticabilité est vérifiée si et seulement si chaque apparition d'une faute est suivie par une séquence finie d'événement observables qui n'apparaît pas en l'absence de la faute [94].

Basile a discuté dans [95], la notion de la K -diagnosticabilité basée sur les RdP, où K désigne le nombre de transitions observables et non observables tirées après la transition défaillante. La K -diagnosticabilité est résolue comme un problème de programmation linéaire. Dans [95], il propose une approche exhaustive pour la vérification de la capacité de la K -diagnosticabilité pour les RdP labellisés et non labellisés. Dans cette approche, le comportement du système est représenté par une série d'équations linéaires. La diagnosticabilité des modèles RdP ne peut être vérifiée que si le comportement défaillant est décrit par un nombre fini d'équations. L'approche est efficace pour vérifier la K -diagnosticabilité, mais elle ne convient pas à l'analyse classique de la diagnosticabilité, puisque pour examiner la diagnosticabilité classique, de nouveaux systèmes d'équations doivent être construits pour chaque K , et l'espace d'état existant ne peut pas être suffisamment utilisé. Dans [74] il introduit la notion r -diagnosticabilité qui signifie que tout défaut peut être détecté après au plus r unités de temps, où il utilise le Modified State Class Graph, un graphe d'estimation du marquage des RdPTL, qui fournit une description exhaustive du comportement du système.

[89] propose des techniques de vérification de la diagnosticabilité des fautes permanentes et intermittentes par model-checking. Il introduit la notion de la diagnosticabilité sur des automates à états finis avec une approche à base de diagnostiqueur. Un diagnostiqueur qui distingue à l'intérieur de chaque nœud les états nominaux des états fautifs afin suivre efficacement l'évolution des séquences. Il démontre que l'efficacité du diagnostiqueur de Sampath se dégrade drastiquement avec l'augmentation des événements non observables.

[96] propose une solution de calcul du délai minimum nécessaire pour l'analyse de la diagnosticabilité des systèmes à événements discrets temporisés. Il modélise le système sur des réseaux de Petri T-temporels labellisés (RdPTL) considérés bornés et utilise un algorithme à la volée, basé sur une structure de données qui contient des informations sur l'état du réseau sans avoir nécessairement à explorer l'intégralité de son espace d'états. Son objectif est d'exploiter l'ensemble des événements observables datés pour vérifier la possibilité de diagnostiquer une faute permanente dans un délai fini après son occurrence. Pour résoudre le problème de l'explosion combinatoire lors de la construction d'un graphe avec l'énumération des classes d'états de l'RdPTL, [96] propose la construction à la volée d'un observateur appelé graphe de classe d'états enrichi analogue au diagnostiqueur proposé dans les travaux de [18]. L'utilisation d'un observateur (diagnostiqueur) sans revenir au modèle du système permet d'exploiter une partie de l'espace d'état optimisant ainsi le temps du traitement de l'algorithme du diagnostic, mais sans pour autant réduire sa complexité.

[97], [98] propose une méthode de diagnosticabilité basée sur un automate de type diagnostiqueur qui, contrairement à celui proposé par [18], n'exige pas d'hypothèses

restrictives sur la vivacité du langage et sur l'inexistence de cycles d'états liés à des événements non-observables uniquement. La façon la plus simple de transformer un langage non-vivant en un langage vivant est d'ajouter des boucles automatiques labellisées avec des événements inobservables à toutes les séquences du langage qui n'ont pas de continuité. Selon [18], un état du diagnostiqueur est dit être F-certain (resp. normal) si tous ses éléments sont étiquetés F (resp. N). Si un état possède à la fois des éléments étiquetés N et F, il est dit incertain.

La figure 2.19 représente la différence entre les modèles diagnostiqueurs proposés par [18] et [97]. Le but est de démontrer que la prise en considération des cycles d'exécutions dues à l'occurrence d'un événement non-observable peut changer les résultats du diagnostiqueur. Soit le modèle (a) de la figure 2.19, l'ensemble des événements du modèle sont réparties tels que : l'ensemble des événements observables est $\Sigma_o = \{a, b, c\}$ et l'ensemble des événements non-observables est $\Sigma_{uo} = \{u, f\}$ avec « f » un événement de défaillance. Selon le modèle diagnostiqueur (b) l'état $\{7F\}$ est certain à partir de l'occurrence de l'événement « b », mais nous n'avons pas pris en considération la possibilité que l'événement « u » peut apparaître et empêche l'évolution du pronostiqueur (b) vers l'état $\{7F\}$.

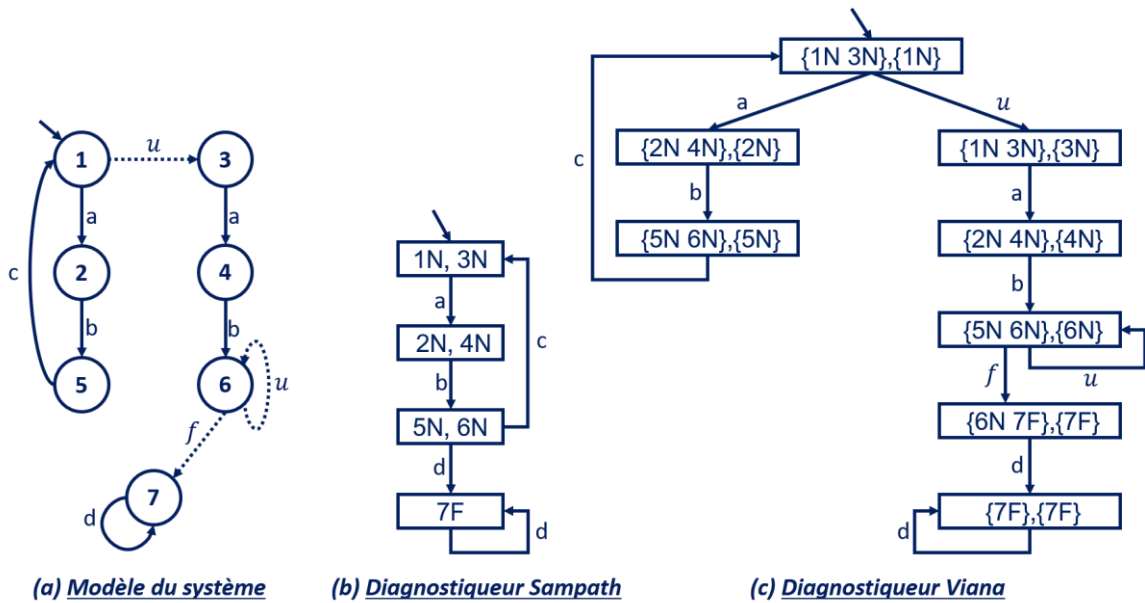


Figure 2.19: Diagnosticteur Sampath Vs Viana.

Le modèle diagnostiqueur proposé par [97] inclut la représentation des événements inobservables, permettant ainsi de corriger les résultats du diagnostiqueur [18]. Ainsi, l'état $\{7F\}, \{7F\}$ du diagnostiqueur (c) n'est pas diagnosticable à partir de l'occurrence de l'événement « b », puisqu'il existe une possibilité d'occurrence de l'événement « u » à partir de l'état $\{5N, 6N\}, \{6N\}$. On remarque qu'avec la représentation des événements non-observable sur le diagnostiqueur (c) le nombre d'états est devenu supérieur par rapport à celui du diagnostiqueur (b), mais ses résultats le rendent avantageux.

On remarque que la propriété de diagnosticabilité dépend de la qualité de la représentation des états dans le diagnostiqueur, qui dépend lui-même de la qualité d'observation. Autrement dit plus l'observation des événements du système est exhaustive (l'observation de l'ensemble des événements nominaux et défectueux), plus le diagnostiqueur est expressif en termes d'états et transitions d'états, ce qui permet de réduire considérablement le nombre de cas ambigus. En conséquence, d'avantage de fautes seront diagnosticables. Dans le cas où la propriété de diagnosticabilité à base de diagnostiqueur n'est pas vérifiée, la reconstruction du modèle diagnostiqueur est alors nécessaire, afin d'intégrer les événements qui n'ont pas permis la

diagnosticabilité. Si la diagnosticabilité est basée sur un observateur, il faut mettre à jour l'algorithme de détection.

2.5.4.Comparatif

Le [tableau 2.3](#) représente les avantages et les inconvénients des méthodes de diagnostic. Selon les approches existantes du diagnostic on peut distinguer deux types de diagnostiqueur : à base d'événements et à base d'états. Les modèles à base d'événements exigent l'initialisation du diagnostiqueur en même temps que le procédé pour qu'il puisse suivre son évolution. En revanche, les diagnostiqueurs à base d'états supposent lors de la modélisation que les sources d'information utilisées pour l'observation des événements sont robustes et ne peuvent être défaillantes ce qui n'est pas le cas lors de la modélisation à base d'événement.

Approches	Outils de représentation	Avantages	Inconvénients	Diagnostiqueur
SAMPATH [18]	Automate à états finis	- Détection des pannes intermittentes - Description précise, théorie des langages et outils de composition et de projection	- Explosion combinatoire	- Détermine l'état du système via les observations d'événements.
ZAD [87]	Automates à états finis	- Diagnostic basé sur un modèle SED à temps discret.	- Le nombre d'états du diagnostiqueur très important dû à l'explosion combinatoire causée par l'intégration de l'information temporelle - Complexité de l'algorithme de synthèse du diagnostiqueur.	- utilise les signatures temporelles pour détecter l'occurrence d'un événement défaillant.
BOUSSIF [89]	Automates à états finis	- Diminution de l'explosion combinatoire - Identifier en temps réel les éventuelles défaillances	- Prise en compte des événements concurrentiels	- Sépare les séquences défaillantes de celles nominales pour réduire la complexité du diagnostic.
MALKI [91]	Automates temporisés	- Diminution de l'explosion combinatoire	- La diagnosticabilité n'est pas développée.	- Utilise les Template pour déterminer les séquences d'événements possibles et les chroniques pour comparer les séquences d'événements et détecter celles qui sont défaillantes.
PENCOLE [90]	Automates temporels	- Diagnostic temps réel	- L'algorithme utilisé est complexe à analyser	- Détermine si une séquence est normale, incertaine ou ambiguë.
SADDEM [92]	Automate temporisés RdP T-temporels	- L'utilisation des signatures temporelles pour faire le diagnostic sans construire un diagnostiqueur	- La garantie de l'exhaustivité d'une base donnée.	- Signatures temporelles causales
AMMOUR [93]	RdP partiellement mesurés	- Analyse temporelle et probabiliste.	- Evaluer les probabilités en fonction des dates d'occurrences	- Détermine les trajectoires qui amènent vers des fautes.
LIU [78]	RdPTL	- Temps de diagnostic réduit	- Nécessite une observation exhaustive.	- Un Observateur à la volée.

VIANA [97]	Automate à états finis	- Pas de restriction sur la vivacité du langage - Prise en considération des cycles	- Nombre d'états plus important que le diagnostiqueur de [18].	- Un observateur qui représente à la fois les événements observable et non observable pour déterminer les états certain et incertain.
----------------------	------------------------	--	--	---

Tableau 2.3: Tableau comparatif des méthodes de diagnostic.

On remarque que les travaux initiaux sur le diagnostic se sont intéressés à l'ordre d'occurrence des événements. L'observation totale ou partielle de ces événements permettait de vérifier la propriété de diagnosticabilité, une propriété nécessaire et suffisante pour le diagnostic. Elle est souvent confrontée à l'ambiguïté, dues au problème d'indéterminisme ; une décision sur la possibilité de faire le diagnostic ou non, ne peut être vérifiée s'il existe deux traces avec la même projection et le même ordre d'événements, tel que l'une conduit à un événement de défaillance et l'autre non. L'utilisation des contraintes temporelles peu résoudre ce problème, mais leur exploitation est à risque (explosion de l'espace d'états). Ce qui a incité certains chercheurs à étudier et proposer des techniques d'optimisation de l'espace d'états. Notre approche étant basée sur une analyse temporelle et logique du modèle comportemental du système afin de déterminer la date d'occurrence des événements, le calcul de la durée d'exécution des séquences d'événements semble intéressant. L'exploitation des signatures temporelles ou des chroniques est passive dans le cadre du diagnostic, puisqu'il faut attendre l'occurrence d'un événement défaillant pour identifier sa causalité. Dans le contexte du pronostic, il faut être proactif pour signaler l'occurrence ou la date d'occurrence de l'événement à l'avance. Permettant ainsi, de laisser un temps d'intervention avant la panne. Certaines techniques utilisées dans le diagnostic peuvent être exploitées dans le cadre du pronostic ; par exemple un modèle pronostiqueur équivalent au diagnostiqueur, le calcul de la durée d'exécution d'une séquence future en se basant sur la notion de signature temporelle causale etc. La section suivante, présente quelques travaux orientés pronostic basés sur des automates à états et d'autres sur des RdP.

2.6. Pronostic des SED

2.6.1.Contexte

Les algorithmes de pronostic pour les systèmes critiques posent des défis majeurs aux concepteurs de logiciels et aux ingénieurs de contrôle pour agir avant la panne. La prédiction précise et fiable de la date de défaillance est absolument essentielle dans le milieu industriel. Cette section présente les différentes approches du pronostic réalisées par la communauté scientifique des SED. On rappelle qu'on ne s'intéresse dans notre étude qu'aux travaux basés sur des modèles SED comportementaux. La première partie représente les approches basées sur des automates à états finis et la seconde, sur des réseaux de Petri. Dans certains travaux la propriété de pronosticabilité sera introduite comme condition nécessaire et suffisante de validation.

2.6.2.Pronostiqueur

Le module pronostiqueur est construit à partir du modèle comportemental du système. Il consiste à représenter sur un graphe d'états orienté, l'ensemble des évolutions d'états possibles, afin de prédire l'occurrence d'un comportement indésirable ou défaillant.

[99] suppose que l'occurrence des défauts ne peut être prédite avec certitude, il s'intéresse alors à calculer le degré de pronosticabilité de l'ensemble des observations sur un système à

événements discrets stochastique. Il utilise des étiquettes d'états dans un automate stochastique et les convertit en une chaîne de Markov. Sur la base du modèle du système et de son pronostiqueur, la probabilité d'occurrence future des défauts est déduite. Selon les résultats des observations du système obtenus, la probabilité d'occurrence d'une faute est mise à jour. Le pronostiqueur proposé n'est pas modélisé graphiquement mais il est sous format d'un algorithme avec des calculs probabilistes, sont alors déterminés l'estimation d'état actuel du système (ES), le vecteur de distribution de probabilité d'état actuel (VDP) et la probabilité actuelle du pronostic (Pr). L'idée d'actualiser les calculs de probabilité au fur et à mesure avec l'évolution du système est très intéressante. Néanmoins, le manque d'observation et le problème d'ambiguïté qui n'ont pas été évoqués, c'est une perspective pour améliorer la qualité des calculs probabilistes.

[40] Introduit le pronostiqueur comme un observateur de l'occurrences des événements, qui détecte le caractère imminent de toute défaillance sur le modèle du système. Il considère la défaillance comme une violation des spécifications et non pas comme une exécution d'événement. Le pronostiqueur prédit la défaillance imminente sur la base d'une observation d'un événement nominal qui appartient à une séquence qui se termine avec un événement défaillant. La notion de m-pronostiqueur est introduite pour indiquer le nombre des états existants avant la défaillance. La complexité de l'algorithme de pronostic proposé est linéaire par rapport au nombre d'états de spécification. La distinction des séquences qui se terminent avec une défaillance de celles qui sont nominales permet de réduire le temps de traitement lors de la vérification du pronostic. Profiter de la séquentialité des événements pour déterminer les m étapes avant l'occurrence d'une panne semble intéressant et consolider ces séquences d'événements avec des signatures temporelles permet de prédire la date d'occurrence d'une défaillance au lieu d'indiquer le nombre d'étape avant panne.

Le rôle du pronostiqueur est alors de vérifier l'évolution du comportement du système dans le futur, pour signaler l'occurrence d'une défaillance imminente. Soit il est construit sous format graphique à partir du modèle comportemental du système pour un pronostic hors-ligne, soit il est sous format algorithmique pour un pronostic en ligne.

2.6.3. Travaux

2.6.3.1. Pronostic basé sur les automates à états finis

La problématique du pronostic a été abordé initialement dans la littérature avec des automates à états finis. Nous avons choisi de présenter d'une façon non exhaustive quelques travaux.

[100] Définit la propriété de prédictibilité de l'occurrence d'un événement défaillant f basée sur des événements observables. Son étude s'inspire du diagnostic des défauts introduit par Sampath (Voir [section 2.5-2](#)), l'objectif est de former une théorie qui intègre du diagnostic et du pronostic dans le cadre des langages formels. Elle démontre qu'une prédictibilité à base de vérificateur est moins complexe en termes de calcul que celle basée sur un diagnostiqueur. Le vérificateur est un automate à états finis non déterministe construit par rapport à une projection sur l'ensemble des événements observables et d'événements de défaut à prédire. Il est défini par $V_G = (Q_V, \Sigma_V, \delta_V, q_{V,0}, e_p)$, avec :

- Q_V : est l'ensemble des états finis du vérificateur ;
 - δ_V : est la relation entre état-transition du vérificateur ;
 - $q_{V,0}$: est l'état initial du vérificateur ;
- Un état du vérificateur $q_V \in Q_V$ a la forme $q_V = [(q_1, l_1)(q_2, l_2)]$, où $q_i \in Q$ et $l_i \in \{N, F\}$

- $\delta_V(q_V, e)$: est l'ensemble des états du vérificateur défini par :

Si $e \in \Sigma_{uo}$ alors

$$\delta_V([(q_1, l_1), (q_2, l_2)], e) = \{[(\delta(q_1, e), l'_1), (q_2, l_2)], [(q_1, l_1), (\delta(q_2, e), l'_2)], [(\delta(q_1, e), l'_1), (\delta(q_2, e), l'_2)]\};$$

Si $e \in \Sigma_o$ alors

$$\delta_V([(q_1, l_1), (q_2, l_2)], e) = [(\delta(q_1, e), l'_1), (\delta(q_2, e), l'_2)];$$

Soit E_V l'ensemble des états normaux du vérificateur définis comme suit :

$E_V = \{x_V \in Q_V^N : \delta_V(x_V, s_{uo}e_p) \text{ est défini pour } e \in \Sigma, s_{uo} \in \Sigma_{uo}^* \text{ et } e_p \notin s_{uo}\}$. Avec : x_V un état du diagnostiqueur. $x_{V,0}$ est l'état initial du diagnostiqueur.

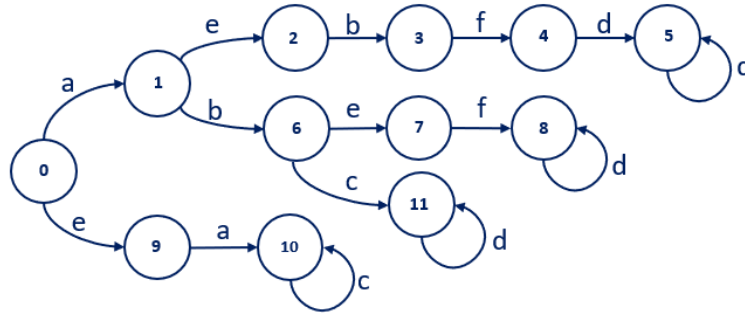


Figure 2.20: Modèle d'automate à état.

Si on considère l'automate à états finis de la [figure 2.20](#) avec son diagnostiqueur à la [figure 2.21](#) où $\Sigma_o = \{a, b, c, d\}$ et $\Sigma_{uo} = \{e, f\}$ alors :

$$E_V = \{[6N, 6N], [7N, 6N], [3N, 6N], [3N, 7N], [3N, 3N]\}.$$

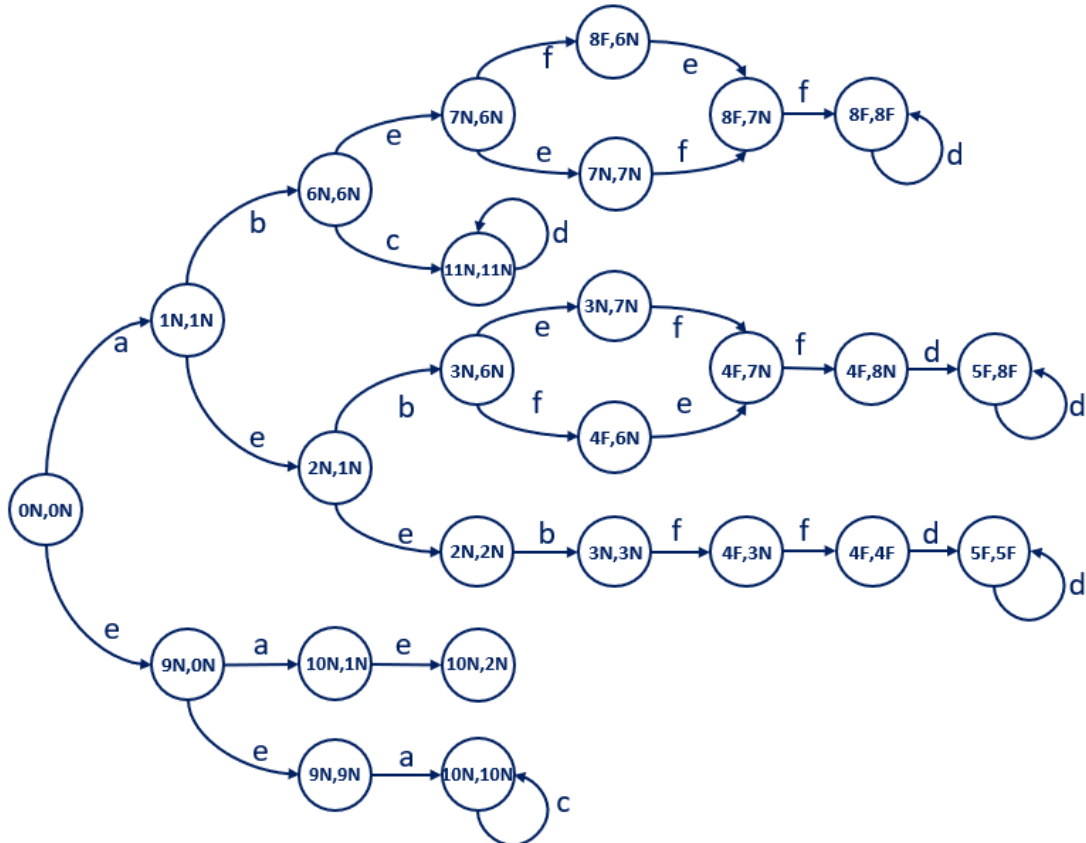


Figure 2.21: Diagnostiqueur GENC.

L'automate à états finis accessible de [6N, 6N] contient un cycle qui comporte un état de vérificateur normal. Ainsi, l'apparition de f n'est pas prédictible. On remarque que l'espace d'état du diagnostiqueur est plus important que le modèle ce qui représente l'avantage d'une bonne expressivité des états du système. Cependant, cette expressivité représente une difficulté de réalisation pour des modèles beaucoup plus complexes.

[101] Propose une méthode de pronostic basée sur la reformulation du problème de pronostic temps réel sous une forme non-temps réel, en utilisant une transformation des automates à états finis temporels en automates à états finis où les contraintes de temps sont détectées par deux types d'événements : les événements Set et Exp qui correspondent respectivement à l'activation et à l'expiration des horloges.

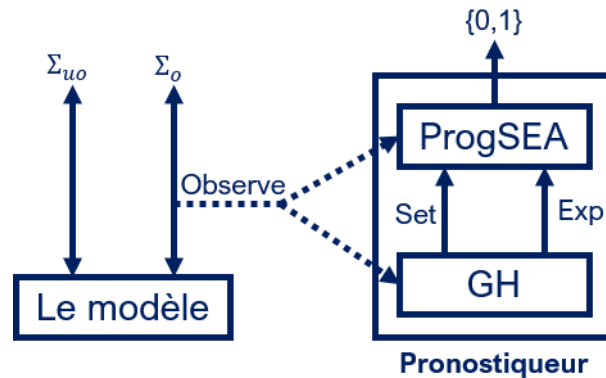


Figure 2.22: Architecture du pronostic.

La figure 2.22 représente l'architecture du pronostic. Cette architecture comprend le modèle comportemental et le pronostiqueur, ce dernier étant composé de deux modules : ProgSEA qui observe les événements (Set, Exp) et un gestionnaire d'horloge (GH). L'approche fournit une architecture de pronostic pratique et réduit considérablement le problème d'explosion d'états causé par le nombre de constantes utilisées dans les contraintes temporelles. Ainsi, au lieu vérifier les instants d'occurrence d'un événement dans un intervalle statique par exemple, il utilise les deux événements Set et Exp. Le pronostiqueur observe alors l'occurrence des événements et vérifie si leur occurrence a eu lieu avant ou après l'expiration de l'horloge et signale toute incohérence à l'opérateur du système.

Dans [102] toujours avec la même logique (le pronostic des défaillances à base d'observation du système), il considère que le système à pronostiquer est modélisé par un langage à préfixe clos L sur un alphabet Σ . L est partitionné en F et H qui modélisent des comportements avec et sans défaillance, respectivement. Il détermine comme conditions pour le pronostiqueur :

- 1) une défaillance est prédite avant qu'elle ne se produise, et
- 2) une défaillance n'est prédite que lorsqu'elle est inévitable, c'est-à-dire si son occurrence est certaine dans un futur limité.

Ce pronostiqueur est une variante du pronostic par inférence introduit par [103].

Soit ω une séquence d'alphabet sans défaillance, $\omega \in H$. Il note la décision prise par le pronostiqueur par $Prog(\omega)$. Après l'exécution ω :

$Prog(\omega) = 1$, prédit une défaillance inévitable.

$Prog(\omega) = 0$ indique qu'aucune défaillance n'est actuellement inévitable.

$Prog(\omega) = \phi$ prédit une défaillance inévitable non imminente et l'absence de défaillance imminente (ambiguïté).

Cette spécification de la nature de prédiction d'événement défaillant, permet en plus d'alerter l'occurrence future d'une défaillance, de déterminer si elle est inévitable ou imminente. Donnant ainsi, à l'opérateur du système le degré d'urgence de la défaillance future.

La plupart des travaux basés sur les automates à états finis, s'inspirent de la méthode du diagnostic proposée par [18] pour faire le pronostic. Le modèle compact du diagnostiqueur avec des états étiquetés, permet d'identifier les séquences indicatrices de défaillance. L'introduction de l'aspect temporel en plus de l'ordre d'occurrence des événements dans le pronostiqueur est très intéressante, mais conduit à une explosion d'états. L'utilisation des événements Set et Exp réduit considérablement cette explosion d'état, mais ne fournit pas des informations sur les instants d'occurrence des événements. Ce qui nous ne permettons pas prédire la date d'occurrence d'un événement défaillant. Dans la section suivant nous allons voir d'avantage travaux orientés pronostic basés sur des RdP.

2.6.3.2. Pronostic basé sur les réseaux de Petri

Nous présentons dans cette section quelques travaux sur l'approche de pronostic des événements de défaillance à base des réseaux de Petri, pour exploiter les performances des RdP en termes de représentation graphique et l'analyse des propriétés du système :

[49] Prouve formellement qu'une seule estimation du marquage est suffisante pour effectuer à la fois le diagnostic et le pronostic sous l'hypothèse que le langage est préfix-clos. Son approche permet de déterminer si une faute pourrait se produire ou non et, si oui, dans combien de temps, à l'aide d'un algorithme de détection de défaillance, inspiré du graphe de classes d'états introduit par [69]. Par conséquent, il n'est pas nécessaire de stocker tous les états éventuels qui pouvaient être atteints à partir du marquage initial.

[51] Introduit une dynamique stochastique dans les processus de défaillance avec des modèles de Markov, afin d'évaluer la probabilité d'apparition des défaillances. Il utilise comme modèle les réseaux de Petri stochastique partiellement observés (RdPSPO) dont le franchissement d'une transition est soit détecté (événement observable) soit silencieux (événement non-observable). A partir de ces modèles, une fonction de mesure élabore des séquences d'observation chronométrées qui stocke l'historique des données recueillies par les systèmes de surveillance et d'acquisition de données. Ces séquences comprennent les événements observables, la date de ces événements et la partie mesurée des trajectoires de marquage. Les trajectoires de marquage chronométrées et non chronométrées correspondant à une séquence d'observation chronométrée donnée qui est calculée selon une programmation linéaire de problèmes entiers (LPP), et la probabilité d'existence de chaque trajectoire est évaluée. Les probabilités de défaillances passées et futures sont obtenues en conséquence.

[52] Etudie l'approche du pronostic sur des systèmes à événements discrets stochastiques. Pour y parvenir, il considère que les réseaux de Petri stochastiques partiellement observés modélisent le comportement du système avec faute. Le modèle représente alors à la fois les comportements normaux et les comportements défaillants du système. Il introduit les dates dans son approche pour caractériser l'incertitude des dates d'occurrences des événements qui sont modélisés par des variables aléatoires. L'intérêt est de discerner avec précision le comportement du système pour conclure sur le pronostic.

Il estime la probabilité d'occurrence d'une faute sur un intervalle de temps futur. L'idée de son approche est de déterminer à partir d'une séquence observable dans un intervalle $[\tau_0, \tau_{fin}]$ l'occurrence d'une faute dans un intervalle $[\tau_{fin}, \tau + i]$. L'analyse du comportement du systèmes a été ainsi primordiale, pour identifier les séquences possibles à partir du marquage

à l'instant τ_{fin} qui représente la date de fin d'observation. Sachant que le pronostic est réalisé à partir de cette date. Il note l'ensemble de marquage possible à partir de τ_{fin} par $Mark(\tau_{fin})$ et l'ensemble des séquences cohérentes avec les séquences observées SO par $\Gamma^{-1}(SO)$ avec $P(m, \tau_{fin})$ la probabilité m du marquage à la date τ_{fin} .

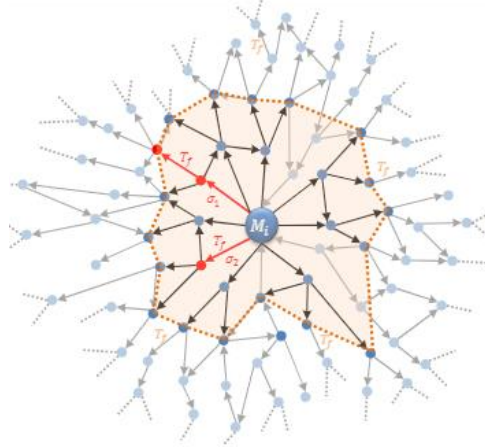


Figure 2.23: Représentation des trajectoires [52].

Il propose deux méthodes de pronostic :

La première méthode permet à partir d'une SO dans un intervalle $[\tau_0, \tau_{fin}]$ d'évaluer la probabilité d'occurrence d'une faute future. Pour y parvenir, il détermine l'ensemble des séquences d'événements d'une taille maximale l_{max} qui se termine par une faute. La taille maximale des séquences est introduite pour éviter d'avoir des séquences infinies. Ce qui limite l'exploitation de l'ensemble du graphe des marquages comme le montre la figure 2.23. La complexité de cette phase dépend du nombre de transitions dans le RdP et de l_{max} des séquences. Ensuite, il évalue la probabilité d'occurrence d'une faute à un intervalle $[\tau_{fin}, \tau + i]$ futur à partir de l'état actuel du système.

La deuxième méthode est basée sur l'horizon du pronostic δ et le seuil d'erreur d'estimation ρ . Il détermine l'ensemble des séquences possibles à partir d'un marquage m et qui se terminent par une faute. L'erreur d'estimation de la probabilité d'apparition de la faute doit être bornée par ρ dans un horizon de pronostic qui ne dépasse pas δ à partir d'un marquage m .

La complexité due à la taille des séquences dans la première méthode et au test des condition de probabilité dans la deuxième, ouvrent des perspectives pour étendre cette approche de pronostic. L'exploitation des contraintes temporelles liées aux transitions peut diminuer l'ambiguïté entre des séquences du modèle RdP et les observations du système. Néanmoins, la difficulté du calcul probabiliste des séquences infinie reste présente.

2.6.4.Comparatif

Le tableau 2.4 représente les avantages et les inconvénients des différentes méthodes de pronostic qui sont présentées ici (non exhaustives). Les approches de pronostic qui ont introduit l'aspect probabiliste, justifient le choix d'utilisation des modèles stochastiques par l'imprévisibilité des événements défaillants qu'il faut pronostiquer. Le contexte de l'approche de pronostic dans ce mémoire s'intéresse à la date d'occurrence d'un événement défaillant. L'aspect probabiliste n'est pas négligeable et peut être exploité dans des travaux ultérieurs pour vérifier le taux de certitude (la probabilité d'occurrence d'un événement défaillant à une date au plus tôt) du pronostic. D'autre travaux ont intégré l'aspect temporel en introduisant la notion d'horizon du pronostic ; une zone délimitée et à partir de laquelle le pronostic ne peut

être effectué. Cette zone regroupe un ensemble d'états. Les organiser en mode de fonctionnement serait utile dans la construction du modèle comportemental. L'approche de pronostic qu'on propose dans ce mémoire considère que les événements de défaillance ne sont pas nécessairement des événements non observables. Il pourrait s'agir d'événements indésirables, dont la date d'apparition peut être prévue à l'avance (durée de vie). Une date qui permet aux opérateurs d'être proactifs, afin de limiter toute évolution future pouvant endommager le système ou même l'arrêter.

Approches	Outils de représentation	Objectif	Intérêt	Pronostiqueur	Observation
TAKAI [40]	Automates à états finis stochastiques	Prédit l'occurrence d'une faute m-étapes à l'avance	- La séquentialité des événements. - La distinction des séquences fautives de celles nominales	Un observateur de l'occurrence des événements, qui détecte le caractère imminent de toute défaillance sur le modèle du système	Partielle
CHEN [104]	Automates à états finis stochastique	Prédit m-étapes à l'avance l'occurrence d'une faute	Un algorithme de pronostic récursif en ligne pour calculer la distribution des états	Le pronostiqueur qui fait correspondre les observations aux décisions en comparant une statistique appropriée avec un seuil,	Partielle
LIAN [99]	Automates à états finis stochastique	Calculer la probabilité d'occurrence future des défauts	Mise à jour du pronostic avec l'évolution du système.	Un algorithme avec des calculs probabilistes de l'occurrence des événements défaillants.	Partielle
KHOUMSI [102]	Automates temporisés	Prédire tout comportement qui n'est pas conforme à une spécification.	La réduction d'espace d'état	observe le système afin de synthétiser un pronostic correct.	Partielle
LEFEBRE [51]	Réseaux de Petri	Evalue la probabilité d'apparition des défaillances.	Séquence d'observation chronométrée	Calcul les distributions de croyance avec les observations du système	partielle
AMMOUR [93]	Réseaux de Petri	Estime la probabilité d'occurrence d'une faute sur un intervalle de temps futur.	- L'horizon du pronostic - L'estimation d'erreur	Un algorithme qui calcul la probabilité d'apparition d'une faute dans un horizon δ à partir d'un marquage m .	Partielle

Tableau 2.4: Travaux sur le pronostic.

2.6.5. Pronosticabilité et observabilité des événements

Un SED est dit partiellement observable si l'occurrence de certains de ses événements ne peut être observée. La notion d'observation partielle intervient naturellement dans de nombreux systèmes : un système réel est souvent plongé dans un environnement hostile, et celui-ci ne peut pas être complètement connu et maîtrisé. On suppose alors que le système ne connaît pas totalement son environnement et reçoit partiellement des informations de l'extérieur. On dit alors que le système n'observe que partiellement l'environnement qui l'entoure.

La notion de pronosticabilité a été introduite par [105], [106]. Un système est dit pronosticable si, pour toute défaillance qu'il pourrait subir, son occurrence peut être prédite, c'est-à-dire que l'on peut être certain, à partir d'une observation, que la défaillance se produira certainement après l'occurrence d'un nombre fini d'événements. Nous allons présenter quelques travaux sur la pronosticabilité basés sur différents modèles et approches.

[106] est l'un des premiers travaux qui abordent le problème de pronostic d'événements sur des SED partiellement observables. Des modèles d'automates finis modélisent le système, où l'ensemble des événements est partitionné en deux sous-ensembles : observables et non-observables. La notion de prédictibilité (pronosticabilité) est la capacité de déduire les occurrences de défaillances futures sur la base des enregistrements de mesures réelles. La propriété de la diagnosticabilité s'inspire des travaux de [18]. Un module pronostiqueur est basé sur des observations d'événements. Une méthode polynomiale, vérifie la prédictibilité directement sur un vérificateur qui est une structure de type Twin Plant. Le but est de chercher l'ensemble d'états limites entre les états nominaux et défaillant dans ce vérificateur, il est similaire à l'ensemble des états N-certains limites avant la première occurrence d'une défaillance utilisé dans le diagnostiqueur, on le notera ici ζ_f . Les transitions dans le vérificateur sont de la forme $(x, x') \xrightarrow{e} (y, y')$, où x, x', y, y' sont des états dans le système, e est un événement et les transitions $(x) \xrightarrow{e} (y), (x') \xrightarrow{e} (y')$ sont toutes les deux définies dans la relation de transition du système. L'ensemble ζ_f est calculé en deux étapes : Tout d'abord, en déterminant le passage de l'état nominal à l'état défaillant ou ambigu dans le vérificateur, on obtient l'ensemble ζ'_f . Ensuite, cet ensemble ζ'_f est complété par l'ajout de tout état potentiel correspondant à des séquences dont les suites futures ne contiendront pas l'événement f mais qui partagent cependant leur préfixe avec d'autres états qui auront au moins une occurrence de l'événement f dans une de leurs suites futures. Enfaite, l'ajout de ces états, avec $\zeta'_f \neq \zeta_f$, ainsi que la vivacité du langage $L(T)$ définie par le système, indiquent la non-pronosticabilité de l'événement f . Pour qu'un événement f soit pronosticable, il faut que chaque cycle réalisable dans le vérificateur à partir des états de cet ensemble ζ_f soit un cycle défaillant. La propriété de pronosticabilité est alors vérifiée de deux façons : une à base d'un diagnostiqueur hors-ligne et l'autre à base vérificateur en-ligne. Celle à base de vérificateur ne nécessite pas la construction d'un modèle pronostiqueur et vérifie la propriété de pronosticabilité directement sur un modèle comportemental du système.

[39] considère le pronostic de défaillance dans un cadre décentralisé pour réduire la complexité du système, ce qui impose qu'une décision de pronostic global soit calculée à partir des décisions locales. La notion de Sm-pronosticabilité ou m-étapes pronosticabilité stochastique, désigne la capacité de prédire un défaut en moins de m-étapes avant son apparition. L'approche est utilisée dans des contextes non temporels et pose comme conditions nécessaires et suffisantes pour la pronosticabilité, que le pronostiqueur ne génère pas de fausse alarme et que pour tout événement prédit son occurrence future est sûre. Un algorithme de complexité polynomiale vérifie la Sm-pronosticabilité, afin de vérifier sur une paire de traces indiscernables l'accessibilité d'une paire d'états, dont l'un est un état sans défaut m-étapes à l'avance et l'autre est un état non-indicateur de défaut (une telle paire est accessible si et seulement si la Sm-pronosticabilité n'est pas vérifiée).

[107] propose que la prédictibilité soit conditionnée par une limite uniforme à l'intérieur de laquelle le défaut peut être prédit. Il propose la notion de "copronosticabilité sans limite uniforme" qui rend moins contraignante l'exigence d'une limite uniforme. Cela va donc caractériser la performance des pronostiqueurs.

[108] aborde le problème de la pronosticabilité des défaillances dans le cadre des SED flous. Le concept de pronosticabilité floue est formalisé dans les SED flous au moyen de fonctions de pronosticabilité floue qui prennent leurs valeurs dans l'intervalle $[0, 1]$ plutôt que

dans l'ensemble binaire $\{0, 1\}$. Ces fonctions quantifient le degré de pronosticabilité d'un événement de défaillance à différents niveaux, notamment :

- Le degré de pronosticabilité d'un préfixe non défectueux d'une séquence qui se termine par un événement de défaillance.
- le degré de pronosticabilité d'une séquence qui se termine par une défaillance.
- Le degré de pronosticabilité d'un événement de défaillance dans le SED flou.

Les fonctions de pronosticabilité floue indiquent qu'un événement flou défaillant dans un système flou est pronosticable à un certain degré. En particulier, si la fonction de pronosticabilité floue d'un événement défaillant est égale à un, l'événement flou est pronosticable à 1 ou complètement pronosticable ; Lorsque la fonction de pronosticabilité floue est égale à zéro, l'événement flou est 0-pronosticable ou complètement non pronosticable ; enfin, si la fonction de pronosticabilité floue est égale à $\lambda \in]0, 1[$, alors l'événement flou est λ -pronosticable ou partiellement pronosticable avec un degré λ . L'intervalle $]0, 1[$ des valeurs de λ caractérise un spectre continu des degrés possibles de pronosticabilité partielle. En fait, le degré de pronosticabilité est une mesure utilisée pour améliorer la décision de prédiction des défaillances. Dans le cas où les conséquences d'une défaillance ne sont pas critiques, il est préférable de tolérer un système avec un degré de pronosticabilité suffisamment important que d'ajouter les captages d'information qui peuvent être très coûteux. Ainsi, les indications données par les degrés de pronosticabilité aident le concepteur à trouver un compromis raisonnable entre la pronosticabilité globale du système et le coût nécessaire pour l'assurer.

Pour vérifier la pronosticabilité des défaillances, un test de complexité polynomiale (en nombre d'états) basé sur le vérificateur est proposé par [106], afin de réduire l'espace d'états. Celui-ci est exponentiel dans le cas du diagnostiqueur par rapport au nombre d'états du système et une fonction de transition d'état modifiée inspirée de [96] est alors proposée. Ce qu'on peut conclure de ces travaux et ce qui peut être intéressant dans le cadre du pronostic temporel est la technique de réduction de la complexité polynomiale utilisée pour vérifier la pronosticabilité des défaillances. Vu que l'introduction l'aspect temporel lors de la modélisation du vérificateur ou du diagnostiqueur augmente considérablement avec la complexité du modèle du système.

[109] présente la notion de (i, j) -pronosticabilité où i (resp. j) est une limite inférieure (resp. supérieure) de l'occurrence de la faute. Une extension de la pronosticabilité qui précise qu'il existe un intervalle de temps pendant lequel l'occurrence de la défaillance est inévitable dans le système. Cette notion est très utile car elle permet d'exprimer différents types de pronosticabilité, à savoir si une défaillance peut être prédite suffisamment à l'avance et si le moment de la défaillance peut être prédit avec précision.

Le pronostiqueur est une machine P qui, étant donné une séquence o d'observations, renvoie un intervalle de temps $(x, y) = P(o)$, ce qui signifie que toute trace qui correspond à cette séquence ne sera pas défaillante avant que x autres observations soient collectées (si $x = 0$, la défaillance est peut-être déjà produite) mais sera définitivement défaillante avant que y autres observations le soient (ou renvoie $y = \infty$ si la défaillance n'est pas prédite - elle peut ne jamais se produire).

Il existe deux types d'incertitude : l'incertitude sur ce qui s'est passé jusqu'à présent (nous savons seulement que le comportement a généré la séquence o mais le comportement réel est inconnu) ; l'incertitude sur ce qui va se passer à partir de maintenant.

Discuter la propriété de pronosticabilité permet de déterminer les raisons pour lesquelles un événement ne peut être pronostiqué (manque de capteurs pour des raisons économiques par exemple) et nous pousse à réfléchir à une solution pour le rendre possible (utilisation des STC par exemple). Certaines approches utilisent un module pronostiqueur pour vérifier cette

propriété. La construction d'un pronostiqueur robuste, nécessite un modèle comportemental affiné (réalisé à partir des connaissances sur l'ensemble des événements non-observables susceptible d'altérer le fonctionnement du système). Comme c'est le cas pour la diagnosticabilité, le problème d'ambiguïté est également présent. La détection d'une ambiguïté limite les performances du pronostic en termes de précision de la détection des séquences, qui risquent de faire évoluer le système vers un état de panne.

[110] indique que la pronosticabilité robuste exige que, quel que soit le modèle réel, une séquence qui mène à un état limite peut toujours être identifiée par rapport aux séquences qui mènent à des états non indicateurs de défaillance dans le modèle; une alarme de défaillance peut donc être émise sans ambiguïté. Il définit un modèle robustement pronosticable par :

$(\forall \omega \in L(G^N) : \delta(\omega) \in \partial(G^N))$ and $(\forall s \in L(G^N) : \delta(s) \in Y(G^N))[P(s) \neq P(\omega)]$, où :

- G^N est le modèle nominal.
- $L(G^N)$ est le langage générer par G and $L(G^N)$ les séquences non défailtantes de G .
- $\delta(\omega)$ est l'état atteint par la séquence ω à partir de l'état initial.
- $\partial(G^N)$ l'ensemble des états limites dans G^N
- $Y(G^N)$ l'ensemble des états non indicateurs de défaillance.

L'incertitude des états de défaillance dans le modèle du système, représente une ambiguïté pour la propriété de pronosticabilité et par conséquence la défaillance ne peut être pronostiquée.

Palier au problème d'ambiguïté est très important dans une approche de pronostic. Cela permet de distinguer toute séquence susceptible d'altérer le fonctionnement du système des séquences nominales. Introduire l'aspect temporelle lors de la modélisation du système peut augmenter la qualité d'analyse et par conséquence la robustesse du pronostiqueur. L'identification des séquences défailtante serait alors possible par l'identification des états limites ; identifier toutes séquences qui se terminent par des états limites et vérifier si les signatures temporelles et logiques ne sont pas identiques.

Selon [111] l'analyse du pronostic proposée par [106] ne produit pas toujours un résultat correct, et cela, à cause de l'approche diagnostiqueur utilisée ; elle cache tous les cycles formés avec des événements non-observables. A l'aide de l'observateur introduit dans [97], il vérifie la propriété de pronosticabilité, où les états pronosticables, sont des états de l'observateur qui précèdent la première occurrence de la défaillance dans une séquence, nommés E_f . Par conséquent, un langage L est pronosticable si, et seulement si, tous les cycles atteints par des états $E \in E_f$ sont composés d'état F-certains seulement. Dans son algorithme de vérification de la pronosticabilité, il commence par étiqueter les états du modèle comportemental par N (état nominal) et F (état fautif). Pour s'assurer que le langage est vivant, il ajoute des boucles d'exécutions avec des événements non-observables dans tous les états qui non pas d'évolution possible. Ensuite, il construit un diagnostiqueur [97] pour identifier l'ensemble des états pronosticables et les considèrent comme les nouveaux états initiaux du diagnostiqueur. S'il existe des projections identiques de séquences à partir des états pronosticables, alors l'occurrence de la défaillance n'est pas pronosticable. L'exploitation du diagnostiqueur pour vérifier la propriété de pronosticabilité est très intéressante. L'intégration des contraintes temporelles dans le diagnostiqueur permettra la pronosticabilité de la défaillance en cas de projections identiques. Le seul cas où la défaillance n'est pas pronosticable : s'il existe des signatures temporelles de séquences identique et des projections de séquences identiques.

La vérification de la propriété de pronosticabilité est primordiale dans une approche de pronostic. L'occurrence d'un événement de défaillance ne peut être prédite sans la vérification de l'existence ou non d'une ambiguïté entre les séquences menant à cet événement de

défaillance. Certains proposent une vérification hors ligne, sur la base d'un pronostiqueur construit à partir du modèle comportemental du système et d'autres proposent un algorithme de vérification en ligne, qui suite à l'observation du système signale la présence de divergences. Soulever l'ambiguïté entre des séquences qui mènent à un événement de défaillance serait alors important. Cela exige un pronostiqueur ou un observateur qui permet une analyse qualitative et quantitative exhaustive.

2.7. Conclusion

Le chapitre 2 a présenté un état de l'art sur le pronostic des événements de défaillance basé sur des SED. Nous l'avons organisé de façon à mettre en évidence l'intérêt des notions d'ordre et de date d'occurrence d'un événement dans le cadre d'une approche orientée pronostic. Nous avons commencé par analyser la dynamique des SED et démontrer sa cohérence avec celle des systèmes que l'on souhaite étudier. Pour décrire les méthodes d'analyse comportementale du système, nous avons présenté des approches de représentation du système avec et sans modèle. Nous avons jugé opportuns avant de discuter des approches dans le cadre du pronostic, d'évoquer quelques travaux réalisés sur le diagnostic. Le but est d'identifier les techniques utilisées pour l'optimisation de l'espace d'états et pour le calcul des temps d'exécution des séquences. Déterminer la durée d'une séquence par exemple, nécessite un modèle du système qui représente à la fois l'ordre et les dates d'occurrence des événements. L'objectif était aussi d'identifier l'apport d'une approche de pronostic par rapport à celle du diagnostic. Nous avons présenté et discuté différentes approches proposées par la communauté scientifique des SED dans le cadre du pronostic, telles que des approches logiques, stochastique et temporelles. L'intérêt est d'avoir une bonne base pour positionner nos travaux par rapport aux travaux existants. Deux propriétés nécessaires et suffisantes pour vérifier le diagnostic et le pronostic ont été aussi évoquées : la diagnosticabilité et la pronosticabilité. Le calcul du pronostic (respectivement diagnostic) ne peut être effectué sans vérifier la possibilité ou non de pronostiquer (diagnostiquer) un événement de défaillance. Des travaux proposent la construction d'un modèle pronostiqueur pour conclure sur la propriété de pronosticabilité d'autre proposent d'utiliser un vérificateur.

Parmi les points importants qui ont attiré notre attention lors de l'étude de l'état de l'art, on cite :

- La construction d'un pronostiqueur exige une modélisation comportementale du système affiné.
- Le pronostic de l'occurrence d'un événement de défaillance n'est pas suffisant, une approche à base temporel ou stochastique serait plus appropriée.
- La signature temporelle des séquences qui ont causées une défaillance, nommées signature temporelle causale (STC) dans le cadre du diagnostic, peut être exploitée dans le calcul du temps avant défaillance.
- La vérification de la propriété de pronosticabilité est très importante pour identifier les séquences avec des événements de défaillance pronosticables.
- Il serait intéressant de développer un système d'inéquations pour calculer la durée minimale d'exécution d'une séquence.

Ainsi, au regard des attendus, la démarche de pronostic pour les SED proposée repose sur :

- Un modèle comportemental basé sur des modes de fonctionnement.
- Un pronostiqueur qui permet une analyse à la fois qualitative et quantitative.
- Une étude de pronosticabilité qui discerne l'ensemble des séquences pronosticables et identifie l'existence d'ambiguïtés.

- Un système d'inéquations qui permet de calculer la date d'occurrence au plus tôt d'un événement de défaillance.

Avec comme hypothèses :

- Une défaillance est unique et catalectique ; pour pouvoir identifier les séquences de terminaison défaillante et calculer la date d'occurrence au plus tôt de l'événement de défaillance.
- Le modèle comportemental du système est sauf.
- L'ensemble des événements du système sont observables.
- Une défaillance n'est prédite que lorsque son occurrence est certaine dans un futur limité.

L'originalité majeure de cette proposition serait de lier les modes de fonctionnement du système au pronostic et développer un système d'inéquations pour pouvoir calculer à partir d'une séquence pronosticable, la date d'occurrence au plus tôt d'un événement de défaillance.

L'assurance de la faisabilité du pronostic serait définie par l'expression de la propriété de pronosticabilité, fortement dépendant des capacités du système à fournir un ensemble minimum d'information sur les dates d'occurrence des événements.

CHAPITRE 3 : Proposition d'une approche de pronostic temporel paramétrique

3.1. Introduction

Dans le cadre du pilotage d'un système soumis à des événements de défaillance, nous proposons dans ce chapitre, une approche de pronostic à base de modèle, où tout comportement est supposé être connu et explicitement inclus dans le modèle. L'analyse du modèle comportemental du système est basée sur la séquentialité et la date d'occurrence des événements, c'est la raison pour laquelle nous avons choisi de modéliser le système par des réseaux de Petri temporels labellisés (RdPTL). L'intérêt est de développer une approche proactive qui permet de prédire la date au plus tôt de l'occurrence d'un événement de défaillance. A partir de cette information temporelle, l'opérateur peut planifier les interventions de réparation sur des composants susceptibles d'altérer le bon fonctionnement. Les modèles temporels sont par nature complexes et explosifs en nombre d'états (généralement non-finis) ; nous avons proposé une solution pour palier à cette difficulté. L'objectif est d'assurer une exploitation des états et transitions d'états du système, en vue de circonscrire l'espace d'états et d'aboutir à un modèle générique à la fois compact et expressif. Nous avons représenté ces dynamiques au travers d'une modélisation dans un contexte d'analyse de modes, limitée à 3 modes de fonctionnement (nominal, dégradé et critique). Nous avons supposé que le modèle RdPTL est sauf, une seule transition est tirée à la fois, son tir est immédiat (ne nécessite pas un temps de tir), il n'y a pas de cycles d'exécution* et un seul mode de fonctionnement est actif à la fois.

Afin d'assurer la surveillance du système, nous avons construit un pronostiqueur à partir de l'arbre d'accessibilité du modèle RdPTL. Ce pronostiqueur permet de repérer l'ensemble des séquences d'événements qui se terminent par un événement de défaillance. Le franchissement d'une transition temporelle dans un RdPTL est possible à tout instant dans son intervalle de tir, ce qui génère une infinité d'états et rend la recherche du graphe d'accessibilité complexe et par conséquent une construction difficile du pronostiqueur. C'est la raison pour laquelle, nous avons utilisé la notion de paramétrisation des états du système *i.e.* l'introduction d'une horloge et un système d'inéquations des horloges pour chaque état du système. Les états obtenus de la discrétisation du temps sont alors regroupés dans un seul état (appelé état paramétrique) et le système d'inéquations déterminera les valeurs des horloges. Nous avons proposé deux approches pour la génération du pronostiqueur :

La première approche introduit les variables d'horloge dans la fonction de transition du pronostiqueur et calcule pour chaque sommet du pronostiqueur un système d'inéquations correspondant.

La deuxième approche considère un pronostiqueur basé sur le graphe d'accessibilité logique, à partir duquel nous déterminons la ou les séquences d'événements qui se terminent par un événement de défaillance. A partir de ces seules séquences nous introduisons l'aspect temporel et déduisons le système d'inéquations temporelles, associé.

Le choix de l'une ou l'autre dépend de la complexité du système modélisé. La première option est plus expressive (puisque elle calcule un système d'inéquations pour l'ensemble des sommets du pronostiqueur), et permet de vérifier la propriété de pronosticabilité alors que la seconde approche est une méthode intuitive qui optimise le temps de calcul du système d'inéquations en ne s'intéressant qu'aux séquences d'événements pronosticables, ce qui favorise son utilisation pour les modèles complexes. L'algorithme intégrant le système d'inéquations est introduit avec des exemples qui illustrent son application.

Le pronostic ne peut être toujours établi, la propriété de pronosticabilité (ou de non pronosticabilité), permet de discerner les séquences qui sont pronosticables de celles qui ne le sont pas. Généralement, l'ambiguïté ou l'incertitude dans le pronostic est due au manque d'observations sur le système, pour des raisons économiques ou technologiques. Dans la dernière étape de notre approche, nous proposons un algorithme intégrant un système d'inéquations, qui exploite les séquences jugées pronosticables, pour calculer la date au plus tôt de l'occurrence d'un événement de défaillance. A la fin de ce chapitre, en guise de validation de la proposition, nous exploiterons l'outil de simulation INA, pour générer le graphe d'accessibilité, déterminer l'ensembles des séquences possibles et identifier celle avec une exécution minimale. A notre connaissance, il n'existe pas d'outil de simulation qui permette de calculer la date d'occurrence d'un événement sur une séquence donnée. Avec l'outil INA (Integrated Net Analyzer) [117] nous identifions la séquence défaillante pour calculer le système d'inéquations.

3.2. Approche de pronostic sur RdPTL

3.2.1 Démarche générale de l'approche du pronostic

Le Framework général de la démarche du pronostic est présenté sur la [figure 3.1](#).

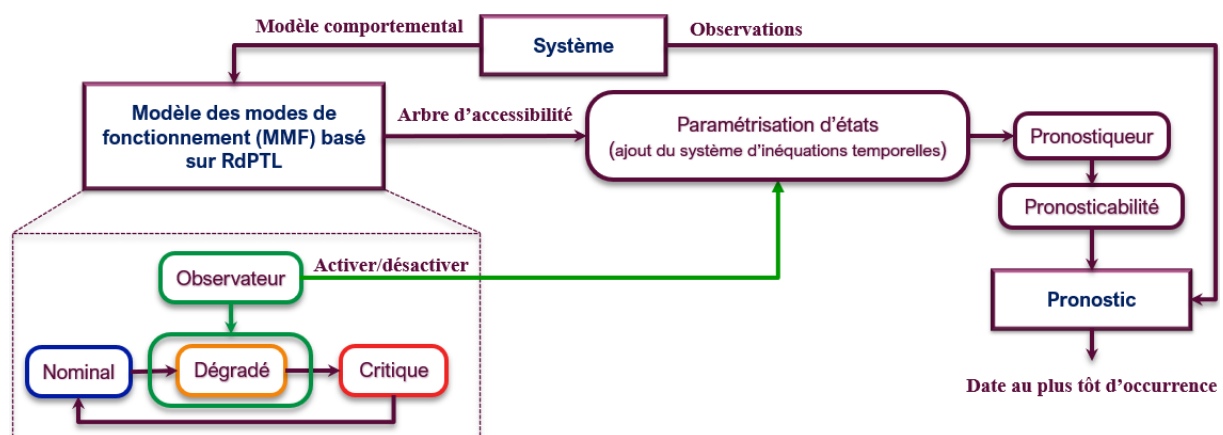


Figure 3.1: Framework général de la démarche.

L'étape la plus importante dans toute approche de pronostic est la représentation du comportement du système ; une représentation qui permet d'analyser son évolution et d'identifier les non-conformités susceptibles d'affecter son fonctionnement. La [figure 3.1](#) est le framework de notre approche, nous avons choisi une méthode avec modèle basé sur les RdPTL, où les conditions de commutation d'états sont représentées par des événements et des intervalles de franchissement permettant ainsi, une analyse à la fois logique (ordre des événements) et temporelle (date d'occurrence des événements) du comportement du système. Pour mieux suivre l'évolution des ressources (jetons), nous avons opté pour une modélisation par des modes de fonctionnement sur un RdPTL sauf, appelée modèle des modes de fonctionnement (MMF) ; une modélisation plus expressive, puisqu'elle permet de représenter le comportement du système dans son mode de fonctionnement nominal, dégradé ou critique. Pour pouvoir appliquer notre approche de pronostic, nous avons introduit dans le MMF un observateur. Il est introduit pour faciliter la détermination du système d'inéquations lors de la paramétrisation du modèle. Son intégration n'influence pas le modèle de modes et ne change pas le comportement du système.

A partir de l'arbre d'accessibilité du MMF, nous construisons un modèle pronostiqueur, afin de pronostiquer en temps réel l'occurrence future des événements défaillants sur le système ; A l'aide d'une fonction d'étiquetage des états par mode de fonctionnement et d'une fonction

de transition d'états par événement et par date, on arrive sur le modèle du pronostiqueur à identifier l'ensemble des séquences qui se terminent par un événement de défaillance permettant ainsi, sur la propriété de pronosticabilité de vérifier parmi ces séquences celles avec un événement de défaillance pronosticable. Nous allons démontrer qu'avec l'introduction de l'aspect temporel dans le modèle du pronostiqueur, certaines séquences jugées non pronosticables avec un pronostiqueur logique, le seront avec la nouvelle représentation du pronostiqueur. Un algorithme à base d'un système d'inéquations, va nous permettre de calculer la durée d'exécution de chacune des séquences pronosticables. A partir de cette durée la date d'occurrence au plus tôt de l'événement de défaillance sera déterminée.

3.2.2 Description de l'approche du pronostic

Le pronostic de défaillance repose sur la prédiction du comportement du système qui n'est pas conforme aux spécifications. Pour modéliser de manière plus synthétique le comportement d'un système, il est important de représenter ces dynamiques au travers de différents modes de fonctionnement. Un mode de fonctionnement rassemblera alors ses dynamiques interprétées dans des espaces communs d'évolution. Par mesure de cohérence, un seul mode est actif à la fois.

Nous proposons une modélisation dans un contexte de gestion de modes, limitée à 3 modes de fonctionnement (nominal, dégradé et critique) réduisant dans un premier temps tout risque d'une explosion combinatoire. Le premier mode représente le fonctionnement nominal du système (le système accomplit la fonction pour laquelle il a été conçu), le système peut basculer de ce mode vers un mode dégradé s'il y a occurrence de certains événements ou l'absence d'autres, ou bien à cause d'un non-respect des contraintes temporelles nécessaires pour la transition entre des états. Le système opère alors avec une dégradation ; accompagnée inéluctablement de performances amoindries par rapport au comportement nominal, sans pour autant que le système soit arrêté ou endommagé. L'occurrence d'un événement de défaillance fait basculer le système dans mode critique où l'état de panne est inévitable (le système ne remplit plus sa fonction et délivre un service incorrect), c'est ce que nous présentons comme mode critique dans la [figure 3.2](#). Les cycles d'exécution des modes ne seront pas pris en considération dans le cas du pronostic d'une défaillance au plus tôt. Ils seront repris plus tard dans la [section 3.3](#).

L'évolution du système est alors consécutive à des transitions entre ces modes. Ainsi, la transition du système d'un mode nominal vers un mode dégradé par exemple est représentée par un changement d'états (un passage d'un état nominal vers un état dégradé par exemple).

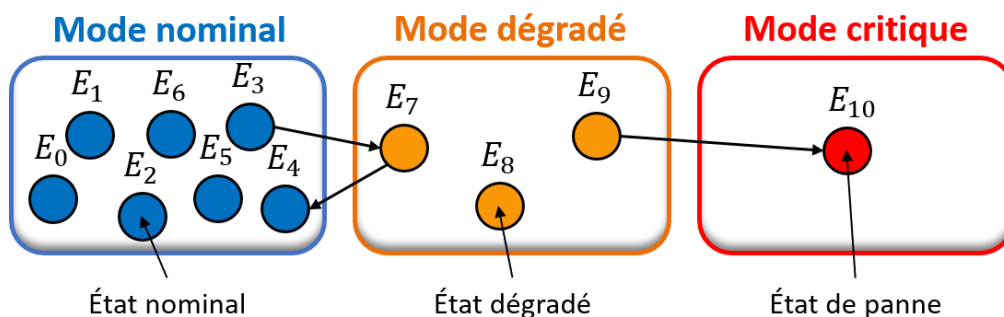


Figure 3.2: États et modes de fonctionnement.

Le mode nominal regroupe l'ensemble des états nominaux, le mode dégradé regroupe les états dégradés et le mode critique regroupe les états critiques conduisant inéluctablement aux états de pannes. A partir du mode dégradé des récupérations et reprise sont toujours possibles

alors que pour le mode critique un retour vers l'état dégradé n'est pas possible. L'ensemble des états du système sont finis et distincts ; un état du système n'appartient qu'à un et un seul mode de fonctionnement.

La [figure 3.3](#) rappelle que le rôle du pronostic est d'agir en signalant l'occurrence future d'une défaillance, avant que le système n'évolue dans un mode de fonctionnement critique, contrairement au diagnostic qui n'intervient qu'après. Dans notre approche un observateur sera nécessaire pour faciliter la détermination du système d'inéquations lors de la paramétrisation du modèle, afin de déterminer la date au plus tôt avant l'occurrence de l'événement de défaillance et signaler en conséquence à l'avance tout disfonctionnement futur.

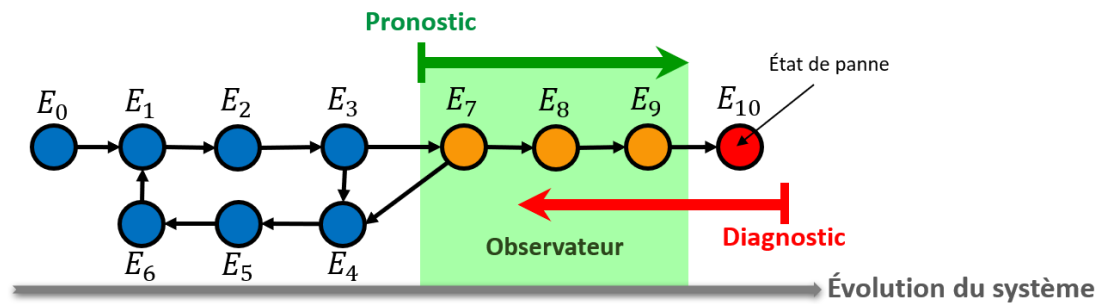


Figure 3.3: Le pronostic et le diagnostic par rapport à l'évolution du système.

La [figure 3.4](#) décrit la démarche suivie dans le pronostic pour déterminer à l'avance la date d'occurrence d'un événement de défaillance. Comme indiqué dans la [section 3.2.1](#), l'expressivité du modèle comportemental du système (en termes d'états et transition d'états) est primordiale pour garantir le pronostic des événements de défaillance ; un modèle moins expressif, laisse une ambiguïté entre des séquences de transitions, réduisant ainsi la possibilité de faire le pronostic. Nous avons explicitement dissimulé les événements et l'aspect temporel dans les [figures 3.2](#) et [3.3](#), pour dire que la séquentialité des états est très importante, mais elle n'est pas suffisante pour faire le pronostic. Nous avons choisi un modèle à base de RdPTL, pour exploiter la séquentialité des événements dans l'identification d'incohérence avec le comportement du système. L'utilité de l'aspect temporel ne sera mise en évidence qu'à partir de l'étude de la pronosticabilité ([section 3.2.5](#)), où le problème d'ambiguïté entre les séquences de transitions sera résolu. Mais avant de vérifier la propriété de pronosticabilité, nous avons introduit un modèle pronostiqueur pour identifier l'ensemble des séquences défaillantes sur le modèle.

Le modèle comportemental du système (voir [section 3.2.2](#)) doit être expressif et avec un espace d'états non explosif. Pour ce faire, nous supposons que le RdPTL est sauf ; seule l'évolution d'un composant est simulée sur le modèle. Expressivité ; notre objectif est d'identifier l'ordre et la date d'occurrence des événements, ces deux aspects (événements/dates) doivent être représentés sur le modèle pour pouvoir effectuer à la fois une analyse quantitative (ordre d'occurrence des événements) et qualitative (date d'occurrence des événements). L'explosion de l'espace d'états est un risque non négligeable dans les RdP temporels, limiter ce risque serait alors nécessaire. Nous allons présenter une méthode de paramétrisation d'états qui n'exploite que des valeurs entières pour représenter le tir des transitions dans les intervalles temporels associés *i.e.* utilise un système d'inéquations pour déterminer la date de tir des transitions. Ce qui réduit considérablement le problème d'explosion d'états pour les RdP temporels.

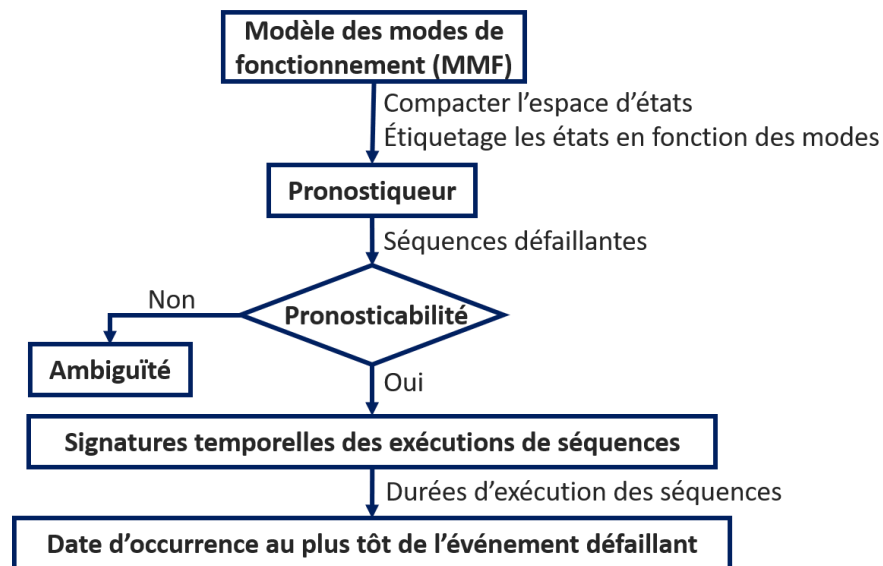


Figure 3.4: Approche de pronostic.

Le pronostiqueur :

Le pronostiqueur est un modèle construit à partir de l'arbre d'accessibilité du MMF (voir [section 3.2.3](#)). C'est un modèle compact, qui offre une représentation de l'ensemble des séquences du système d'une façon plus expressive pour identifier facilement celles qui se terminent avec un événement de défaillance. Nous associons à chaque sommet du pronostiqueur une étiquette qui caractérise son appartenance à un mode de fonctionnement. Ainsi pour un sommet avec marquage ne contenant que des places nominales, nous associerons l'étiquette N, pour un sommet avec marquage ne contenant que des places dégradées, nous associerons l'étiquette D et pour un sommet avec marquage ne contenant que des places Critiques, nous associerons l'étiquette C. Un sommet du pronostiqueur ne peut avoir des places de modes différents.

La pronosticabilité :

La pronosticabilité est une propriété nécessaire et suffisante indiquant que le modèle peut être exploité pour le pronostic. Avant de pronostiquer un événement de défaillance dans une séquence, il est nécessaire de vérifier la possibilité de faire ce pronostic ou pas. La vérification de la pronosticabilité consiste à déterminer si l'exécution d'une séquence est certaine ou incertaine (voir [section 3.2.4](#)). Une séquence est dite certaine s'il existe une seule séquence d'événements possible qui conduit vers l'état de panne. Une séquence est dite incertaine s'il y a une ambiguïté entre l'exécution de deux séquences ou plus qui conduisent vers le même état de panne. Etant donné que les séquences du modèle RdPTL sont datées, la propriété de pronosticabilité se base sur la comparaison de leurs signatures logique (ordre d'occurrence) et temporelles (date d'occurrence) pour distinguer celles qui sont certaines des autres.

Le système d'inéquations est calculé à partir des séquences pronosticables, où le temps d'exécution minimal de chaque séquence est déterminé ; chaque séquence est caractérisée par des transitions datées et son temps d'exécution minimal est calculé à partir des valeurs minimales associées aux variables d'horloge déterminées par le système d'inéquations (voir [section 3.2.5](#)).

La séquence avec le plus petit temps d'exécution, donnera la date d'occurrence minimale d'un événement de défaillance dès le franchissement de la première transition de cette séquence.

3.2.3 Modèle des modes de fonctionnement

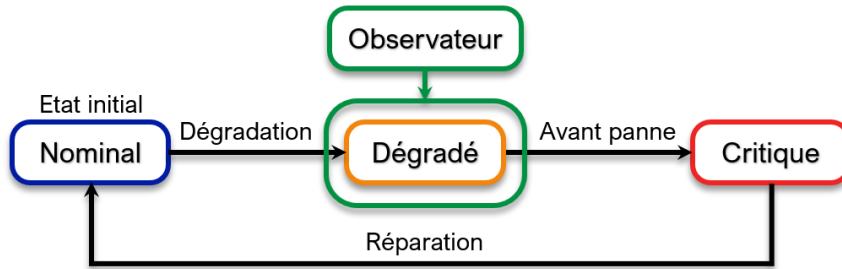


Figure 3.5: Structure et modes de fonctionnement d'un MMF.

Pour représenter les états du système et leurs modes de fonctionnement, nous avons choisi un modèle générique (pour pouvoir l'exploiter dans des systèmes multi-composant). Pour ce faire, il nous faut une représentation intuitive, facile à appréhender et donc à utiliser. Dans le cas de notre étude, la représentation du comportement du système n'est pas réduite à une expression d'états. Nous rappelons que l'objectif est de prédire la date d'occurrence d'un événement. Donc le modèle comportemental du système, doit représenter à la fois les événements et leurs dates d'occurrence. Les RdPTL sont plus adaptés pour décrire les aspects dynamiques et offrent une représentation compacte. Nous avons choisi un exemple didactique (voir figure 3.6) basé sur un RdPTL pour accompagner nos explications tout au long de ce chapitre. Le modèle de la figure 3.5 est modélisé selon les modes de fonctionnement décrits dans la figure 3.5, chaque mode de fonctionnement est caractérisé par ses propre états (places) et leurs transitions. L'observateur est représenté par une place p_{obs} , une transition t_{obs} et deux arcs, l'un deux est un arc inhibiteur.

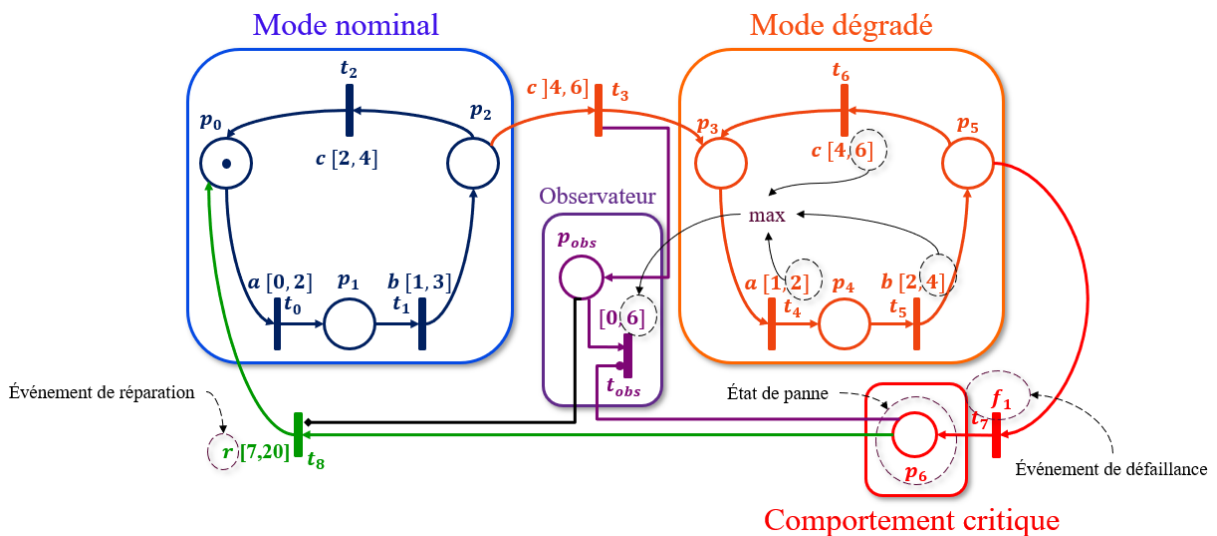


Figure 3.6: Exemple 1 d'un modèle RdPTL.

L'intégration de l'observateur n'influence pas le MMF, son rôle consiste à marquer la place p_{obs} lors du franchissement de la transition t_3 (l'évolution dans un mode dégradé) et de maintenir ce marquage jusqu'à ce que la place p_6 soit marquée (l'évolution dans un mode critique). Avec le marquage de la place p_{obs} , l'arc inhibiteur rend la transition t_{obs} franchissable tant que la place p_6 n'est pas marquée. Ce marquage de l'observateur sera exploité dans le système d'inéquations pour déterminer la durée d'exécution des séquences défaillantes (voir section 3.2.5).

Les transitions du modèle sont labellisées par des événements de l'ensemble des événements Σ , où l'occurrence de chaque événement est déterminée dans un intervalle de franchissement de I .

On suppose que :

- Une seule transition est franchie à la fois ;
- Un seul mode est actif à la fois ;
- Le RdPTL est saut ;
- Le franchissement des transitions est immédiat ; il n'y a pas de délai de franchissement ;
- Il est possible de labelliser deux transitions distinctes avec le même événement ;
- Tous les événements du RdPTL sont observables.
- Pas de retour possible du mode dégradé au mode nominal.
- La sémantique de tir est une sémantique faible » une transition peut ne pas être franchie même si l'on atteint la borne maximale de l'intervalle. C'est l'occurrence de l'événement dans l'intervalle de tir associé qui permet le franchissement.

On rappelle qu'un RdPTL est défini par $R_TL = \langle P, T, pré, post, m_0, \Sigma, \mathcal{L}, I \rangle$.

- P est l'ensemble des places du réseau avec $P = P_n \cup P_{deg} \cup P_{cri} \cup P_{obs}$ avec P_n l'ensemble des places nominales, P_{deg} l'ensemble des places dégradées, P_{cri} l'ensemble des places critiques et P_{obs} est la place d'observation de passage du mode nominal au mode dégradé.
- T est l'ensemble des transitions du réseau avec $T = T_n \cup T_{deg} \cup T_{cri} \cup T_{obs} \cup T_{rep}$
- Σ est l'ensemble des labels (événements) associés aux transitions,
- $\mathcal{L}$ est la fonction de labellisation des transitions
- I est la fonction qui associe un intervalle de temps statique à chaque transition.

si $\exists p_i \in (P_n \cup \{p_{cri}\})$ et $t_j \in (T_n \cup \{t_{cri}\})$, tel que : $m(p_i) \geq pré(p_i, t_j)$, alors $m(p_{obs}) = 0$ et t_{obs} n'est pas sensibilisée, sinon, $m(p_{obs}) = 1$ et t_{obs} est sensibilisée, i.e. Le marquage de la place p_{obs} , signale le passage du système à un mode de fonctionnement dégradé. C'est la raison pour laquelle $m(p_{obs}) = 0$ dans un état de fonctionnement nominal ou critique du système. $\mathcal{L}(t_{cri})$ est un événement de défaillance. Dans le cas de l'exemple de la [figure 3.6](#) :

$$P = \{p_0, p_1, p_2, p_3, p_4, p_5, p_6, p_{obs}\};$$

$$T = \{t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_{obs}\};$$

$$m_0 = (1, 0, 0, 0, 0, 0, 0)^T;$$

$\Sigma = \{a, b, c, f, r\}$, avec: f l'événement de défaillance et r l'événement de réparation.

Le mode critique est représenté par $p_{cri} = p_6$ et $t_{cri} = t_7$.

La transition t_7 est la transition de défaillance, donc : $t_7 \in T_{cri}$ et, $\mathcal{L}(t_7) = f$.

La transition t_8 est la transition de réparation, donc : $t_8 \in T_{rep}$ et, $\mathcal{L}(t_8) = r$.

Une transition sensibilisée par un marquage m du RdPTL est franchissable ssi : l'occurrence de l'événement associé à cette transition a eu lieu entre la borne minimale et maximale de son intervalle de tir $I(t)$. Nous notons l'ensemble des transitions sensibilisées par le marquage m par $TS(m)$.

La dynamique des RdPTL est cohérente avec l'approche de pronostic proposée dans ce mémoire. La représentation des labels et des intervalles dans un RdPTL permet d'identifier à la fois l'ordre et les dates d'occurrence des événements. Il serait alors nécessaire de définir les séquences d'événements datés, pour pouvoir analyser le comportement du système.

Définition 3.1 : séquence d'événements datés : Timed Event Sequence : T.E.S

Soit $\sigma = t_1 t_2 \dots t_n$, une séquence de franchissements dans le RdPTL et soit $\sigma(x) = x_0 t_1 x_1 t_2 \dots x_{n-1} t_n x_n$ sa séquence datée réalisable.

La séquence $s(\sigma) = e_1 e_2 \dots e_{n-1} e_n$ est la séquence logique d'événements associés aux transitions de la séquence de franchissement σ .

La fonction de labellisation est étendue aux séquences de tir datées $\sigma(x)$ comme suit :

- i. $\mathcal{L}(x_q t_q) = x_q \mathcal{L}(t_q) = x_q e_q$, où $e_q \in \Sigma$, $\mathcal{L}(t_q) = e_q$, $t_q \in T$ de même :
 $\mathcal{L}(t_j x_j) = x_j \mathcal{L}(t_j) = e_j x_j$
- ii. $\mathcal{L}(\sigma(x) (t_q x_q)) = \mathcal{L}(\sigma(x)) \mathcal{L}(t_q) x_q = s(x)$.

La séquence $s(x) = x_0 e_1 \dots x_{n-1} e_n x_n$ est une séquence d'événements datés (Timed Event Sequence T.E.S). Il s'agit d'une séquence datée d'événements associée à la séquence datée possible $\sigma(x)$. Ainsi, $s(x) = \mathcal{L}(\sigma(x))$.

Définition 3.2 : langage temporel

Soit un réseau de Petri temporel labellisé noté R_{TL} . Le langage temporel généré par R_{TL} , noté $L(R_{TL})$ est défini par l'ensemble des T.E.S $s(x)$ générées par R_{TL} depuis le marquage initial m_0 . $L(R_{TL}) = \{s(x) | m_0[\sigma(x) > \text{tel que } \mathcal{L}(\sigma(x)) = s(x)]\}$ où $\sigma(x)$ est la séquence datée réalisable dans R_{TL} .

A partir de la [définition 3.2](#) nous pouvons en déduire que pour calculer la date d'occurrence d'un événement défaillant, il faut déterminer à partir d'un marquage m_0 , la ou les séquences datées réalisables conduisant à cet événement.

3.2.4 Etats et séquences paramétriques

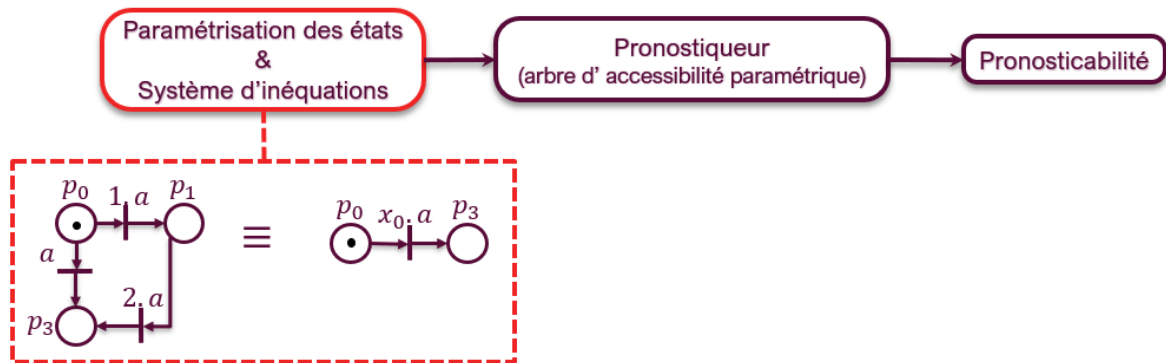


Figure 3.7: Paramétrisation d'états.

Nous avons mis en avant dans le chapitre précédent, la complexité de la construction d'un graphe d'états pour un RdP avec un aspect temporel ; le franchissement d'une transition dans un RdPTL est possible à tout instant dans son intervalle de tir, ce qui génère une infinité d'états. Une première solution restreinte au modèle temporel borné avec une évolution du temps continu a été proposée par [75]. Le but est de contracter cet espace d'états par un graphe de classes fini. Il définit une classe d'états comme un couple $C = (m, D)$ où D représente l'ensemble des vecteurs de date de tir ou domaine de tir. Ce domaine de tir est l'ensemble des solutions de systèmes d'inéquations linéaires avec une variable associée à chaque transition sensibilisée (voir la [section 2.4.2.21](#)).

La deuxième solution proposée par [76] est dédiée aux RdP avec une évolution discrète du temps. Les états obtenus de la discrétisation du temps sont regroupés dans un seul état. Cet état est défini par un couple $z = (m, h)$ avec h est le vecteur d'horloge des transitions. Chaque horloge de transition représente l'instant de sensibilisation de la transition dédiée. Les horloges de transition non sensibilisées sont représentées par $\$$. L'intervalle temporel associé à une transition avec une borne maximum égale à l'infini, génère une infinité d'états. Pour diminuer ce risque, [76] propose de limiter les horloges des transitions sensibilisées aux bornes inférieures si la borne supérieure tend vers l'infini.

Les classes d'états proposés par [75] représentent une abstraction de l'espace d'états, dans lesquels les propriétés temporelles sont vérifiées sur des domaines de tir. Ces domaines de tir ne conservent pas avec précision les valeurs des horloges ce qui rend difficile l'analyse temporelle des séquences, voir [112]. À l'inverse, les états proposés par [76] conservent avec précision dans chaque séquence ces valeurs des horloges.

Le temps d'exécution de chaque séquence (signature temporelle) est calculé en appliquant l'algorithme qui suit la définition 3.3. L'objectif est de trouver toutes les valeurs minimales solutions du système d'inéquations. Ces valeurs constitueront la date au plus tôt de l'occurrence de l'événement de défaillance. Après cette date, l'occurrence de l'événement de défaillance est certaine.

La définition 3.3, est une extension de la définition 2.5, qui permet, à partir d'un RdPTL, de déterminer récursivement l'état paramétrique (voir figure 3.7) et la séquence paramétrique conduisant à un événement de défaillance, et ainsi la génération d'un système d'inéquations composé des contraintes obtenues à partir des intervalles associés à chaque transition autorisée à partir d'une place du RdPTL.

Avant de présenter la définition 3.3, il convient de reconsidérer un ensemble de transitions possibles à partir d'un marquage m .

Soit m un marquage d'un RdP. Nous définissons V_m l'ensemble des transitions t_i sensibilisées à partir de m comme suit : $V_m = \{t_i \in T \mid m \geq \text{pré}(\bullet, t_i)\}$,

Définition 3.3 : (états et séquences paramétriques d'un RdPTL)

Soit $R_{TL} = \langle P, T, \text{pré}, \text{post}, m_0, \Sigma, \mathcal{L}, I \rangle$ un RdPTL, m un p_marquage et h un t_marquage contenant le temps associé à chaque transition validée de m . $\sigma := t_1 \dots t_n$ est une séquence de transitions de tir dans R_{TL} et $\sigma(x) := x_0 t_1 \dots x_{n-1} t_n x_n$ sa séquence datée possible (réalisable). V_{m_σ} est l'ensemble des transitions possibles à partir du marquage m_σ (marquage actuel) obtenu par le tir de la séquence de transition σ . La séquence paramétrique $(\sigma(x), B_\sigma)$ de σ et l'état paramétrique (q_σ, B_σ) dans R_{TL} sont déterminés par l'algorithme suivant :

Algorithme 1 :

Début

Si $\sigma = \varepsilon$, donc $\sigma(x) = x_0$ et $s(x) = x_0 \varepsilon$;

alors $q_\sigma = (m_\sigma, h_\sigma)$ et B_σ sont donnés par:

- 1- $m_\sigma := m_0$,
- 2- $V_{m_\sigma} = V_{m_0} = \{t \mid m_0 \geq \text{pré}(\bullet, t)\}$
- 3- $h_\sigma(t) = h_0(t) = \begin{cases} x_0 & \text{si } t \in V_{m_0} \\ \$ & \text{sinon} \end{cases}$
- 4- $B_\sigma := \{0 \leq h_\sigma(t) \leq \max(t) \mid t \in T \wedge t \in V_{m_0}\}$

sinon

répéter

On suppose que E_σ et B_σ sont déjà définis pour la séquence de transitions $\sigma := t_1 \dots t_n$.

alors $\sigma(x) := x_0 t_1 \dots x_{n-1} t_n x_n$. Sa T.E.S correspondante est $s(x) = x_0 e_1 \dots x_{n-1} e_n x_n$ pour $\sigma := t_1 \dots t_n t_{n+1} = \gamma t_{n+1}$, et $\mathcal{L}(\sigma) = \mathcal{L}(\gamma) \cdot \mathcal{L}(t_{n+1}) = s(\gamma) \cdot \mathcal{L}(t_{n+1})$

1. $m_\sigma := m_\gamma + \Delta t_{n+1}$, avec $\Delta t_{n+1} := post(\bullet, t_{n+1}) - pré(\bullet, t_{n+1})$
2. $V_{m_\sigma} = \{t \mid m_\sigma \geq pré(\bullet, t)\}$
3.
$$h_\sigma(t) = \begin{cases} \$ & \text{si } t \notin V_{m_\sigma} \\ h_\gamma(t) + x_{n+1} & \text{si } t \in V_{m_\sigma} \wedge t \in V_{m_\gamma} \wedge pré(t_{n+1}) \cap pré(t) = \emptyset \wedge t \neq t_{n+1} ; \\ x_{n+1} & \text{sinon} \end{cases}$$
4. $B_\sigma := B_\gamma \cup \{\min(t_{n+1}) \leq h_\gamma(t_{n+1})\} \cup \{0 \leq h_\sigma(t) \leq \max(t) \mid t \in T \wedge pré(\bullet, t) \leq m_\sigma\}$
Tant que $(\mathcal{L}(t_{n+1}) \neq ef)$

Fin

Dans [76], l'intervalle de temps associé à chaque transition t d'un RdPT est de la forme $[\min(t), \max(t)]$. Or, dans certains cas comme le nôtre, l'intervalle de temps n'est pas toujours fermé, d'où la nécessité d'introduire quelques notions sur les intervalles de temps que nous utiliserons plus tard.

Un intervalle de temps peut être défini par l'intersection de deux semi-intervalles.

Définition 3.4 : [96]

Soit I_t un intervalle temporel associé à une transition t d'un RdPT.

- Un semi-intervalle gauche (noté $g(I_t)$) est défini par :
 - o $] \min(t) = \{x \in Q^+ \mid x > \min(t), \min(t) \in Q^+\}$ où
 - o $[\min(t) = \{x \in Q^+ \mid x \geq \min(t), \min(t) \in Q^+\}$
- Un semi-intervalle droit (noté $d(I_t)$) est défini par :
 - o $\max(t)] = \{x \in Q^+ \mid x \leq \max(t), \max(t) \in Q^+\}$ où
 - o $\max(t)[= \{x \in Q^+ \mid x < \max(t), \max(t) \in Q^+\}$

Un semi-intervalle gauche $g(I_t)$ d'une transition peut être séparé en :

Un semi-intervalle gauche ouvert (noté $go(I_t)$) défini par $] \min(t)$ et,
Un semi-intervalle gauche fermé (noté $gf(I_t)$) défini par $[\min(t)$.

De la même manière, un semi-intervalle droit $d(I_t)$ d'une transition peut être séparé en :

Un semi-intervalle droit fermé (noté $df(I_t)$) défini par $\max(t)]$ et
Un semi-intervalle droit ouvert (noté $do(I_t)$) défini par $\max(t)[$.

Exemple : soit t une transition d'un RdPT à laquelle on associe l'intervalle temporel $]2, 3]$ de la forme $] \min(t), \max(t)] = go(I_t), df(I_t)$, ou alors un intervalle $[2, 3[$ de la forme $[\min(t), \max(t)[= gf(I_t), do(I_t)$.

De plus, dans la définition 3.3, si nous considérons des intervalles de temps à bornes ouvertes avec des intervalles à bornes fermées dans le même modèle, nous risquons de trouver des systèmes d'inéquations impossibles à résoudre surtout au moment du changement de mode de fonctionnement. Pour résoudre ce problème et considérer des intervalles de temps aussi bien fermés qu'ouverts dans le même modèle, nous introduisons la [définition 3.5](#).

Définition 3.5 :

Soient $R_T = \langle P, T, \text{pré}, \text{post}, m_0, I \rangle$ un RdPT, $\sigma = t_1 \dots t_n$ une séquence de franchissements dans R_T et $\sigma(x) = x_0 t_1 x_1 t_2 \dots x_{n-1} t_n x_n$ et V_{m_σ} l'ensemble de transitions sensibilisées à partir du marquage m_σ .

Alors, l'exécution paramétrique $(\sigma(x), B_\sigma)$ de σ et l'état paramétrique (q_σ, B_σ) dans R_T sont déterminés par :

Algorithme 2 :

Début

* si $\sigma = \varepsilon$, i.e., $\sigma(x) = x_0$. $\text{etiquette}(m_\sigma) = N$;

Alors $q_\sigma = (m_\sigma, h_\sigma)$ et B_σ sont donnés par :

- 1- $m_\sigma := m_0$;
- 2- $V_{m_\sigma} = V_{m_0} = \{t | m_0 \geq \text{pré}(\bullet, t)\}$
- 3- Tant que $V_{m_\sigma} \neq \emptyset$
 $h_\sigma(t) = \begin{cases} x_0 & \text{si } t \in V_{m_0} \\ \$ & \text{sinon} \end{cases}$,
fin
- 4- $B_\sigma := \{0 \leq h_\sigma(t) \sim \max(t) | t \in T \wedge t \in V_{m_0}\}$, avec $\sim = \begin{cases} \leq & \text{si } d(I_t) = \text{df}(I_t) \\ < & \text{si } d(I_t) = \text{do}(I_t) \end{cases}$;

sinon

Répéter

* on suppose que q_σ et B_σ sont déjà définis pour la séquence $\sigma = t_1 \dots t_n$.

pour $\sigma = t_1 \dots t_n t_{n+1} = \gamma t_{n+1}$, $\sigma(x) = x_0 t_1 x_1 t_2 \dots x_{n-1} t_n x_n t_{n+1} x_{n+1}$, $\mathcal{L}(\sigma) = \mathcal{L}(\gamma) \cdot \mathcal{L}(t_{n+1}) = s(\gamma) \cdot \mathcal{L}(t_{n+1})$ on pose :

si $t_{n+1} \in T_n$ alors $\text{etiquette}_{m_\sigma} = N$;

si $t_{n+1} \in T_{deg}$ alors $\text{etiquette}_{m_\sigma} = D$;

si $t_{n+1} \in T_{cri}$ $\text{etiquette}_{m_\sigma} = C$;

- 5- $m_\sigma := m_\gamma + \Delta t_{n+1}$,
- 6- $h_\sigma(t) = \begin{cases} \$ & \text{si } t \notin V_{m_\sigma} \text{ "donc } t \text{ n'est pas sensibilisée par } m_\sigma" \\ h_\gamma(t) + x_{n+1} & \text{si } t \in V_{m_\sigma} \wedge t \in V_{m_\gamma} \wedge \bullet t_{n+1} \cap \bullet t = \emptyset \wedge t \neq t_{n+1} \\ & \text{"t était sensibilisée pour } m_\gamma \text{ et reste sensibilisée pour } m_\sigma" \\ x_{n+1} & \text{sinon "car } t \text{ est nouvellement sensibilisée"} \end{cases}$
- 7- si $\text{etiquette}_{m_\sigma} \neq \text{etiquette}_{m_\gamma}$ "Il y a alors un changement de mode"
 $B_\sigma := B_\gamma \cup \{\min(t_{n+1}) \sim h_\gamma(t_{n+1}) \simeq d(I_{t_{n+1}})\} \cup \{0 \leq h_\sigma(t) \simeq \max(t) | t \in T \wedge t \in V_{m_\sigma}\}$.
sinon
 $B_\sigma := B_\gamma \cup \{\min(t_{n+1}) \sim h_\gamma(t_{n+1})\} \cup \{0 \leq h_\sigma(t) \simeq \max(t) | t \in T \wedge t \in V_{m_\sigma}\}$.
avec $\sim := \begin{cases} \leq & \text{si } g(I_t) = \text{gf}(I_t) \\ < & \text{si } g(I_t) = \text{go}(I_t) \end{cases}$ et $\simeq := \begin{cases} \leq & \text{si } d(I_t) = \text{df}(I_t) \\ < & \text{si } d(I_t) = \text{do}(I_t) \end{cases}$

Tant que $(\mathcal{L}(t_{n+1}) \neq ef)$

Fin

Exemple :

Prenons l'exemple 1 de la [figure 3.6](#). Supposons que l'on souhaite calculer le temps d'exécution de la séquence $\sigma = t_0 t_1 t_3 t_4 t_5 t_7$. L'événement défaillant f est associé à la transition t_7 . Le but est de calculer sa date d'occurrence future au plus tôt.

$$\begin{aligned}\sigma &= \varepsilon; \sigma(x) = x_0; \\ m_0 &= (p_0); \text{etiquette}_{m_0} = N; \\ V_{m_0} &= \{t_0\}; \\ h(t) &= (x_0, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T; \\ B_0 &= \{0 \leq x_0 \leq 2\};\end{aligned}$$

A partir de m_0 , on franchit $t_0 : \sigma = t_0$;

$$\begin{aligned}m_1 &= (p_1); \text{etiquette}_{m_1} = N; \\ V_{m_1} &= \{t_1\}; \\ \mathcal{L}(t) &= \{a\}; \\ h(t) &= (\$, x_1, \$, \$, \$, \$, \$, \$, \$, \$)^T; \\ B_1 &= \begin{cases} 0 \leq x_0 \leq 2 \\ 0 \leq x_1 \leq 3 \end{cases};\end{aligned}$$

A partir de m_1 , on franchit $t_1 : \sigma = t_0 t_1$;

$$\begin{aligned}m_2 &= (p_2); \text{etiquette}_{m_2} = N; \\ V_{m_2} &= \{t_2, t_3\}; \\ \mathcal{L}(t) &= \{b\}; \\ h_2 &= (\$, \$, x_2, x_2, \$, \$, \$, \$, \$, \$)^T; \\ B_2 &= \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 0 \leq x_2 \leq 4 \end{cases};\end{aligned}$$

A partir de m_2 , on franchit $t_2 : \sigma = t_0 t_1 t_2$;

$$\begin{aligned}m'_0 &= (p_0); \text{etiquette}_{m'_0} = N; \\ V_{m'_0} &= \{t_0\}; \\ \mathcal{L}(t) &= \{c\}; \\ h'_0 &= (x'_0, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T; \\ B'_0 &= \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 2 \leq x_2 \leq 4 \\ 0 \leq x'_0 \leq 4 \end{cases};\end{aligned}$$

A partir de m_2 , on franchit $t_3 : \sigma = t_0 t_1 t_3$;

$$\begin{aligned}m_3 &= (p_3 p_{obs}); \text{etiquette}_{m_3} = D; \text{ "il y a changement de mode"} \\ V_{m_3} &= \{t_4, t_{obs}\}; \\ \mathcal{L}(t) &= \{c\}; \\ h_3 &= (\$, \$, \$, \$, x_3, \$, \$, \$, \$, x_3)^T; \\ B_3 &= \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 0 \leq x_3 \leq 2 \end{cases};\end{aligned}$$

A partir de m_3 , on franchit $t_4 : \sigma = t_0 t_1 t_3 t_4$;

$$\begin{aligned}m_4 &= (p_4 p_{obs}); \text{etiquette}_{m_4} = D; \\ V_{m_4} &= \{t_5, t_{obs}\};\end{aligned}$$

$$\begin{aligned}\mathcal{L}(t) &= \{a\}; \\ h_4 &= (\$, \$, \$, \$, \$, x_4, \$, \$, \$, x_3 + x_4)^T; \\ B_4 &= \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 0 \leq x_4 \leq 4 \\ x_3 + x_4 \leq 12 \end{array} \right\};\end{aligned}$$

A partir de m_4 , on franchit $t_5 : \sigma = t_0 t_1 t_3 t_4 t_5$; etiquette $_{m_5} = D$;

$$\begin{aligned}m_5 &= (p_5); \\ V_{m_5} &= \{t_6, t_7, t_{obs}\}; \\ \mathcal{L}(t) &= \{b\}; \\ h_5 &= (\$, \$, \$, \$, \$, \$, x_5, \$, \$, x_3 + x_4 + x_5)^T; \\ B_5 &= \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 2 \leq x_4 \leq 4 \\ 0 \leq x_5 \leq 6 \\ x_3 + x_4 \leq 12 \\ x_3 + x_4 + x_5 \leq 12 \end{array} \right\};\end{aligned}$$

A partir de m_5 , on franchit $t_6 : \sigma = t_0 t_1 t_3 t_4 t_5 t_6$;

$$\begin{aligned}m'_6 &= (p_3, p_{obs}); \text{ etiquette}_{m'_6} = D; \\ V_{m'_6} &= \{t_4, t_{obs}\}; \\ \mathcal{L}(t) &= \{c\}; \\ h'_6 &= (\$, \$, \$, \$, x'_6, \$, \$, \$, \$, x_3 + x_4 + x_5 + x'_6)^T; \\ B'_6 &= \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 2 \leq x_4 \leq 4 \\ 4 \leq x_5 \leq 6 \\ 0 \leq x'_6 \leq 2 \\ x_3 + x_4 \leq 12 \\ x_3 + x_4 + x_5 \leq 12 \\ x_3 + x_4 + x_5 + x'_6 \leq 12 \end{array} \right\};\end{aligned}$$

On franchit $t_7 : \sigma = t_0 t_1 t_3 t_4 t_5 t_7$;

$$\begin{aligned}m_6 &= (p_6); \text{ etiquette}_{m_6} = C; \\ V_{m_6} &= \{t_8\}; \\ \mathcal{L}(t) &= \{f\}; \text{ "l'événement de défaillance a lieu, on arrête l'algorithme"} \\ h_6 &= (\$, \$, \$, \$, \$, \$, \$, \$, x_6, \$)^T;\end{aligned}$$

$$B_6 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 2 \leq x_4 \leq 4 \\ 6 \leq x_5 \leq 7 \\ 0 \leq x_6 \leq 12 \\ x_3 + x_4 \leq 12 \\ x_3 + x_4 + x_5 \leq 12 \end{array} \right\} ;$$

L'exécution minimale de la séquence $\sigma = t_0 t_1 t_3 t_4 t_5 t_7$ est alors représentée par une séquence datée $S(x) = 0. t_0. 1. t_1. 5. t_3. 2. t_4. 1. t_5. 0. t_7$, où la date d'occurrence de l'événement f est égale à 3 unités de temps à partir du franchissement de la transition t_3 , tout en respectant le système $0 \leq x_3 + x_4 \leq 12$ et $x_3 + x_4 + x_5 \leq 12$.

Dans cet exemple nous avons montré qu'à partir de la paramétrisation des états d'une séquence et d'un système d'inéquations, il est possible de calculer la durée d'exécution d'une séquence. Ceci permettant ainsi d'en déduire, la date d'occurrence au plus tôt du dernier événement de cette séquence. Dans la section suivante nous allons proposer un arbre d'accessibilité temporel regroupant l'ensemble des séquences paramétriques du MMF. A partir de cet arbre, il sera possible d'identifier toute séquence conduisant à un événement de défaillance et lever ainsi toute ambiguïté entre ces séquences. Le but est de générer de cet arbre d'accessibilité un pronostiqueur avec un système d'inéquations pour chaque état afin de vérifier la propriété de pronosticabilité *i.e.* discerner les séquences pronosticables de celles qui ne le sont pas.

3.2.5 Module pronostiqueur

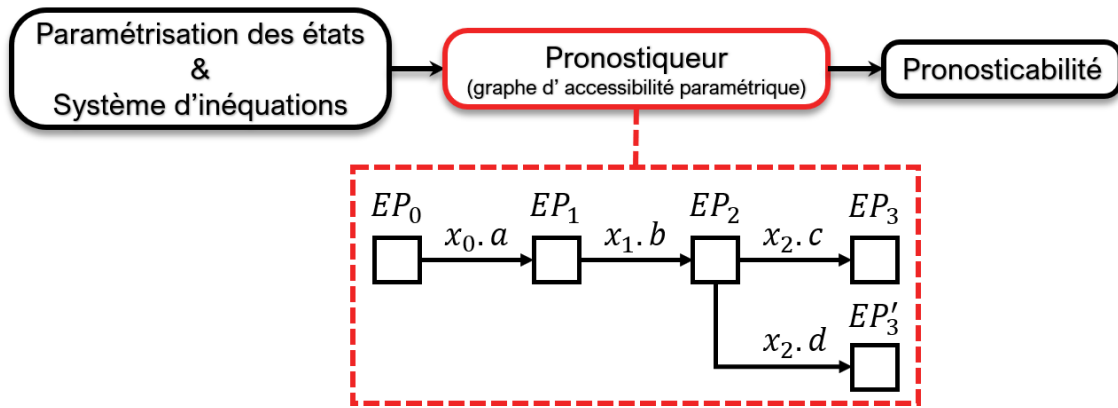


Figure 3.8: Génération du pronostiqueur.

Nous proposons deux méthodes de génération du pronostiqueur (figure 3.8) basées sur la paramétrisation des états du MMF. La première méthode est une méthode intuitive et consiste à déterminer le graphe d'accessibilité logique du modèle RdPLT à partir duquel, nous isolons toutes les séquences qui conduisent à un état de panne. Chacune de ces séquences est paramétrée afin de déterminer la date d'occurrence au plus tôt de l'événement de défaillance. La seconde méthode est une méthode plus expressive et permet de déterminer l'arbre d'accessibilité paramétrique à partir duquel, nous générons un module pronostiqueur. Le module pronostiqueur permet de suivre l'évolution du système et signale à l'avance, toute dégradation susceptible de l'endommager dans le futur. Il est généré à partir de l'arbre d'accessibilité paramétrique (figure 3.9) du MMF. Chaque sommet du pronostiqueur est étiqueté soit par N indiquant que les places marquées dans ce sommet sont des places

appartenant au mode nominales, D pour les places marquées dans ce sommet appartenant au mode dégradées et C pour les places marquées dans ce sommet appartenant au mode critique. L'évolution d'un sommet vers un autre est conditionnée par l'occurrence d'un événement dans un intervalle de temps prédéfini. Si une transition entre deux sommets est conditionnée par l'occurrence d'un événement sans intervalle, cela signifie que la transition entre les sommets amont et aval est immédiate. L'arbre arrête d'évoluer dès que l'on arrive à un état de panne.

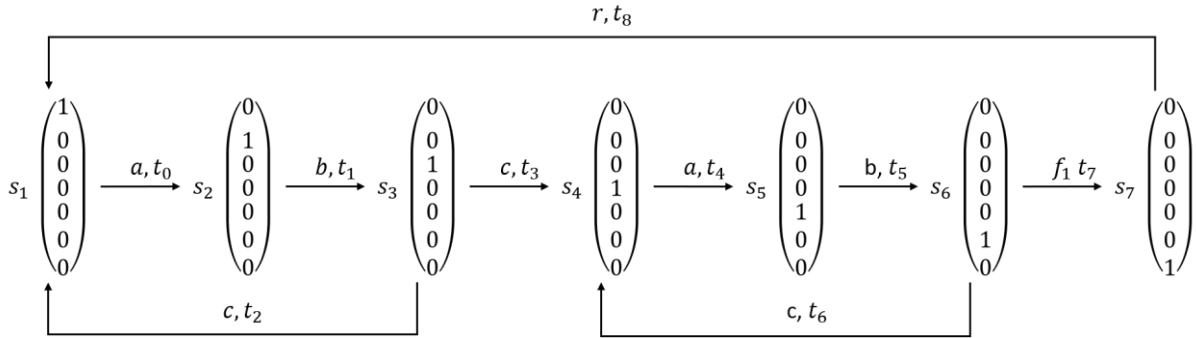


Figure 3.9: Graphe d'accessibilité du MMF de la figure 3.6.

Définition 3.6 : Arbre d'accessibilité paramétrique

Soit $R_{TL} = (P, T, Pre, Post, m_0, \Sigma, \mathcal{L}, I)$ un RdPTL.

L'arbre d'accessibilité paramétrique du R_{TL} , noté GAP_{TL} , est un arbre dont l'ensemble des sommets est S , l'ensemble des arcs est A . cet arbre est construit par l'algorithme suivant :

Algorithme 3 :

Début

$Q := \{EP_0\}$ avec $EP_0 = \{etiquette_{m_0}, E_0, B_0\}$ et $E_0 = (m_0, h_0)$ $S := \emptyset$; $A := \emptyset$. Avec :

$etiquette_{m_0} = N$, panne := false;

$V_{m_0} = \{t \in T / pré(\bullet, t) \leq m_0\}$ est l'ensemble des transitions sensibilisées à partir de m_0 .

Tant que $(Q \neq \emptyset)$ faire :

Choisir $EP = (etiquette_m, E, B)$ à partir de Q ; $Q := Q - \{EP\}$

si $EP \notin S$ alors $S := S \cup \{EP\}$;

$V'_m := V_m - \{t_{obs}\}$. $etiq := etiquette_m$;

si $\exists (t_i, t_j) \in V'_m \mid t_i \neq t_j \wedge t_i \in \text{mode } i \wedge t_j \in \text{mode } j$, alors $etiq := \emptyset$;

Sinon $etiq := etiquette_m$;

Tant que $V'_m \neq \emptyset$ faire :

On franchit une transition $t \in V'_m$; on atteint un état EP' ;

si $t \in T_n$ alors $etiquette_m = N$;

si $t \in T_{deg}$ alors $etiquette_m = D$;

si $t \in T_{cri}$ alors $etiquette_m = C$;

$V'_m := V'_m - \{t\}$;

$m' := m + \Delta t$ où $\Delta t = -\text{pré}(\bullet, t) + \text{post}(\bullet, t)$

$\text{enabled}_{m'} = \{\hat{t} \in T / \text{pré}(\bullet, \hat{t}) \leq m'\};$

$$h'(\hat{t}) = \begin{cases} \$ & \text{si } \hat{t} \notin \text{enabled}_{m'} \\ h(\hat{t}) + \tilde{x} & \text{si } \hat{t} \in \text{enabled}_{m'} \wedge \hat{t} \in \text{enabled}_m \wedge \hat{t} \cap \bullet t = \emptyset \wedge \hat{t} \neq t; \\ \tilde{x} & \text{si non} \end{cases}$$

si $(\text{etiquette}_m \neq \text{eti}q \text{ ou } \text{eti}q = \emptyset)$ alors "il y aura alors un changement de mode"

$B' := B \cup \{g(It) \sim h(t) \simeq d(It) \cup \{0 \leq h'(\hat{t}) \simeq d(I\hat{t}) | \hat{t} \in T \wedge \hat{t} \in \text{enabled}_{m'}\}.$

sinon

$B' := B \cup \{g(t) \sim h(t)\} \cup \{0 \leq h'(\hat{t}) \simeq d(I\hat{t}) | \hat{t} \in T \wedge \hat{t} \in \text{enabled}_{m'}\}.$

si $\mathcal{L}(t) = \text{ef}$, alors panne := true; "test pour arrêter l'arborescence en cours dès que l'on arrive à l'état de panne"

$A := A \cup \{EP, (\mathcal{L}(t), x, t), EP'\};$ "x est la variable d'horloge associée à la transition t"

si $EP' \notin S$ et m' n'est pas un marquage existant et panne = false, alors $Q := Q \cup \{EP'\};$

fin si

si m est un marquage existant ou panne = true alors $S := S \cup \{EP'\};$

fin si

fin

fin

Exemple :

Début

$Q = \{EP_0\}, S := \emptyset, A := \emptyset,$
 $m_0 = (p_0), V_{m_0} = \{t_0\};$
 $h_0 = (x_0, \$, \$, \$, \$, \$, \$, \$, \$)^T,$
 $B_0 = \{0 \leq x_0 \leq 2\};$

On choisit EP_0 :

$Q := Q - \{EP_0\} := \emptyset, S := S \cup \{EP_0\} := \{EP_0\},$
 $V'_{m_0} := V_{m_0} := \{t_0\}, \text{etiquette} := N, \text{eti}q := \text{etiquette};$

A partir de EP_0 , on franchit t_0 et on atteint $EP_1, \mathcal{L}(t_0) = a$, alors panne := false ;

$t_0 \in T_n$ alors $\text{etiquette} := N, V'_{m_0} := V'_{m_0} - \{t_0\} := \emptyset,$
 $m_1 = (p_1), V_{m_1} := \{t_1\};$
 $h_1 = (\$, x_1, \$, \$, \$, \$, \$, \$, \$)^T,$
 $\text{eti}q := \text{etiquette};$
 $B_1 = \begin{cases} 0 \leq x_0 \leq 2 \\ 0 \leq x_1 \leq 3 \end{cases};$
 $Q = \{EP_1\}, A := A \cup \{EP_0, (a, x_0, t_0), EP_1\};$

On choisit EP_1 :

$Q := Q - \{EP_1\} := \emptyset, S := S \cup \{EP_1\} := \{EP_0, EP_1\}, m_1 = (p_1),$
 $V'_{m_1} := V_{m_1} := \{t_1\}, etiq := etiquette, avec etiquette := N,$

A partir de EP_1 , on franchit t_1 et on atteint $EP_2, \mathcal{L}(t_1) = b$, alors panne := false ;

$t_1 \in T_n$ alors etiquette := N, $V'_{m_1} := V'_{m_1} - \{t_1\} := \emptyset,$

$m_2 = (p_2), V_{m_2} = \{t_2, t_3\};$

$h_2 = (\$, \$, x_2, x_2, \$, \$, \$, \$, \$)^T,$

etiq := etiquette ;

$$B_2 = \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 0 \leq x_2 \leq 4 \end{cases};$$

$Q = \{EP_2\}, A := A \cup \{EP_1, (b, x_1, t_1), EP_2\};$

On choisit EP_2 :

$Q := Q - \{EP_2\} := \emptyset, S := S \cup \{EP_2\} := \{EP_0, EP_1, EP_2\}, m_2 = (p_2),$

$V'_{m_2} := V_{m_2} := \{t_2, t_3\}, t_2 \in T_N$ et $t_3 \in T_{deg}$ etiq := $\emptyset$, etiquette := N,

A partir de EP_2 , on franchit t_2 et on atteint $EP_{0\ dup}, \mathcal{L}(t_2) = c$, alors panne := false ;

$t_2 \in T_n$ alors etiquette := N, $V'_{m_2} := V'_{m_2} - \{t_2\} := \{t_3\},$

$m_{0\ dup} = (p_0),$ "c'est un marquage existant $m_{0\ dup} = m_0$, Q ne sera pas étendu".

$enabled_{m_{0\ dup}} = \{t_0\};$

$h_{0\ dup} = (x'_{0\ dup}, \$, \$, \$, \$, \$, \$, \$, \$)^T,$

etiq := $\emptyset$;

$$B_{0\ dup} = \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 2 \leq x_2 \leq 4 \\ 0 \leq x_{0\ dup} \leq 2 \end{cases};$$

$Q = EP_{0\ dup}, S = \{EP_0, EP_1, EP_2, EP_{0\ dup}\}; A := A \cup \{EP_2, (c, x_2, t_2), EP_{0\ dup}\};$

A partir de EP_2 , on franchit t_3 et on atteint $EP_3, \mathcal{L}(t_3) = c$, alors panne := false ;

$t_3 \in T_{deg}$ alors etiquette := D, $V'_{m_2} := V'_{m_2} - \{t_3\} := \emptyset,$

$m_3 = (p_3, p_{obs}), V_{m_3} = \{t_4, t_{obs}\};$

$h_3 = (\$, \$, \$, \$, x_3, \$, \$, \$, x_3)^T,$

etiq := $\emptyset$ et etiquette := D, donc etiq $\neq$ etiquette ;

$$B_3 = \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 0 \leq x_3 \leq 2 \end{cases};$$

"dans l'algorithme de base on obtiendrait $4 < x_2 \leq 4$ ce qui est impossible".

$Q = \{EP_3\}; A := A \cup \{EP_2, (c, x_2, t_3), EP_3\};$

Le franchissement de la transition t_{obs} ne sera pas traité, vu qu'il n'affecte pas le fonctionnement du système.

On choisit EP_3 :

$Q := Q - \{EP_3\} := \emptyset, S := S \cup \{EP_3\} := \{EP_0, EP_1, EP_2, EP_{0\ dup}, EP_3\}, m_3 = (p_3, p_{obs}),$

$V'_{m_3} := V_{m_3} := \{t_4\}, etiq := etiquette, avec etiquette := D,$

A partir de EP_3 , on franchit t_4 et on atteint EP_4 , $\mathcal{L}(t_4) = a$, alors panne := false ;

$t_4 \in T_{deg}$ alors $etiquette := D$, $V'_{m_3} := V'_{m_3} - \{t_4\} := \emptyset$,

$m_4 = (p_4, p_{obs})$, $V_{m_4} = \{t_5, t_{obs}\}$;

$h_4 = (\$, \$, \$, \$, \$, \$, \$, \$, \$, x_3 + x_4)^T$;

$etiq := etiquette$;

$$B_4 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 0 \leq x_4 \leq 4 \\ 0 \leq x_3 + x_4 \leq 8 \end{array} \right\} ;$$

$Q = \{EP_4\}$; $A := A \cup \{EP_3, (a, x_3, t_4), EP_4\}$;

On choisit EP_4 :

$Q := Q - \{EP_4\} := \emptyset$, $S := S \cup \{EP_4\} := \{EP_0, EP_1, EP_2, EP_{0\ dup}, EP_3, EP_4\}$, $m_4 = (p_4, p_{obs})$,

$V'_{m_4} := V_{m_4} := \{t_5\}$, $etiq := etiquette$, avec $etiquette := D$,

A partir de EP_4 , on franchit t_5 et on atteint EP_5 , $\mathcal{L}(t_5) = b$, alors panne := false ;

$t_5 \in T_{deg}$ alors $etiquette := D$, $V'_{m_4} := V'_{m_4} - \{t_5\} := \emptyset$;

$m_5 = (p_5, p_{obs})$, $V_{m_5} = \{t_6, t_7, t_{obs}\}$;

$h_5 = (\$, \$, \$, \$, \$, \$, \$, \$, \$, x_3 + x_4 + x_5)^T$;

$etiq := etiquette$;

$$B_5 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 2 \leq x_4 \leq 4 \\ 0 \leq x_5 \leq 0 \\ 0 \leq x_3 + x_4 \leq 8 \\ 0 \leq x_3 + x_4 + x_5 \leq 8 \end{array} \right\} ;$$

$Q = \{EP_5\}$; $A := A \cup \{EP_4, (b, x_4, t_5), EP_5\}$;

On choisit EP_5 :

$Q := Q - \{EP_5\} := \emptyset$, $S := S \cup \{EP_5\} := \{EP_0, EP_1, EP_2, EP_{0\ dup}, EP_3, EP_4, EP_5\}$,

$m_5 = (p_5, p_{obs})$, $V'_{m_5} := V_{m_5} := \{t_6, t_7\}$, $t_6 \in T_{deg}$ et $t_7 \in T_{cri}$ alors $etiq := \emptyset$, $etiquette := D$,

A partir de EP_5 , on franchit t_6 et on atteint EP_6 , $\mathcal{L}(t_6) = f$

$t_6 \in T_{deg}$ alors $etiquette := D$, $V'_{m_5} := V'_{m_5} - \{t_6\} := \{t_7\}$;

$m_6 = (p_3, p_{obs})$, "c'est un marquage existant $m_6 = m_3$, Q ne sera pas étendu".

$V_{m_6} = \{t_4, t_{obs}\}$;

$h_6 = (\$, \$, \$, \$, \$, \$, \$, \$, \$, x_3 + x_4 + x_5 + x_6)^T$;

$etiq := etiquette$;

$$B_6 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 2 \leq x_4 \leq 4 \\ \text{Impossible} \\ 4 \leq x_5 \leq 0 \\ 0 \leq x_6 \leq 2 \\ 0 \leq x_3 + x_4 \leq 8 \\ 0 \leq x_3 + x_4 + x_5 \leq 8 \\ 0 \leq x_3 + x_4 + x_5 + x_6 \leq 8 \end{array} \right\}; \text{"on ne franchira pas } t_6 \text{ dans ce cas particulier".}$$

A partir de EP_5 , on franchit t_7 et on atteint EP_7 , $\mathcal{L}(t_7) = f$, alors panne := true ;

$t_7 \in T_{cri}$ alors etiquette := C, $V'_{m_5} := V'_{m_5} - \{t_7\} := \emptyset$;

$m_7 = (p_6, p_{obs}), V_{m_7}(t) = \{t_8\}$;

$h_7 = (\$, \$, \$, \$, \$, \$, \$, \$, x_7, \$)^T$;

etiq = D et etiquette = C; etiq $\neq$ etiquette ;

$$B_7 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 2 \leq x_4 \leq 4 \\ 0 \leq x_5 \leq 0 \\ 0 \leq x_7 \leq 8 \\ 0 \leq x_3 + x_4 \leq 8 \\ 0 \leq x_3 + x_4 + x_5 \leq 8 \end{array} \right\};$$

$S := S \cup \{EP_7\}$, donc $S = \{EP_0, EP_1, EP_2, EP_3, EP_4, EP_5, EP_7\}$; $A := A \cup \{EP_5, (f, x_5, t_7), EP_7\}$;

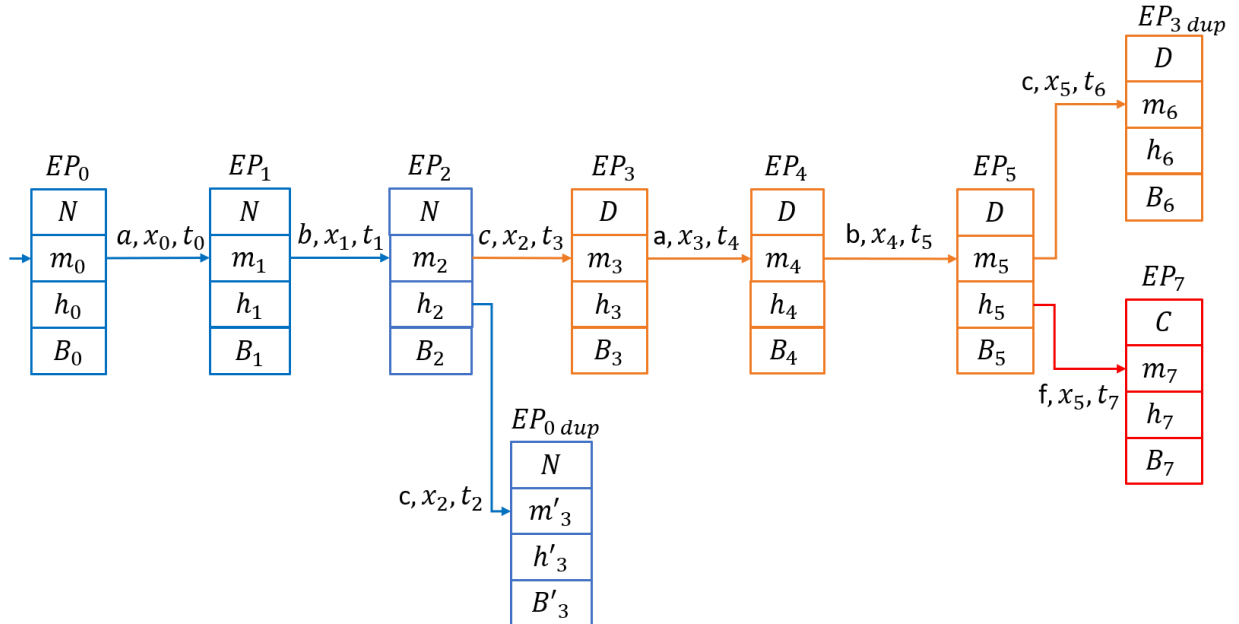


Figure 3.10: Le modèle pronostiqueur avec la première approche.

La figure 3.10 représente le pronostiqueur correspondant au MMF de la figure 3.6. Un modèle expressif qui inclus dans chaque sommet le résultat du système d'inéquations et paramétrise les états dans les fonctions de transition pour réduire l'espace d'états.

La figure 3.7 représente l'arbre d'accessibilité réduit (n'inclus pas les transitions de l'observateur) du MMF de la figure 3.6. Pour optimiser le temps de calcul du système d'inéquations, nous proposons une seconde version du pronostiqueur basée sur le graphe d'accessibilité logique. On paramétrise uniquement les séquences qui se terminent par un événement de défaillance f sur l'arbre d'accessibilité. Si nous considérons l'exemple de la figure 3.6, il existe une seule séquence $\sigma = t_0 t_1 t_3 t_4 t_5 t_7$ à laquelle nous appliquerons l'algorithme 2 :

$$\begin{aligned} \text{Pour } \sigma = \varepsilon, \sigma(x) &= x_0, \mathcal{L}(\sigma(x)) = \varepsilon x_0 ; \\ m_\sigma &= m_0 = (p_0), V_{m_0} = \{t_0\}, \\ h_0 &= (x_0, \$, \$, \$, \$, \$, \$, \$, \$), \\ B_0 &= \{0 \leq x_0 \leq 2\} ; \end{aligned}$$

On franchit t_0 :

$$\begin{aligned} \sigma &= t_0, \sigma(x) = x_0 t_0, \mathcal{L}(\sigma(x)) = x_0 a ; \\ m_1 &= (p_1), V_{m_1} = \{t_1\}, \\ h_1 &= (\$, x_1, \$, \$, \$, \$, \$, \$, \$), \\ B_1 &= \begin{cases} 0 \leq x_0 \leq 2 \\ 0 \leq x_1 \leq 3 \end{cases} ; \end{aligned}$$

On franchit t_1 :

$$\begin{aligned} \sigma &= t_0 t_1, \sigma(x) = x_0 t_0 x_1 t_1, \mathcal{L}(\sigma(x)) = x_0 a x_1 b ; \\ m_2 &= (p_2), V_{m_2} = \{t_2, t_3\}, \\ h_2 &= (\$, \$, x_2, x_2, \$, \$, \$, \$, \$), \\ B_2 &= \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 0 \leq x_2 \leq 4 \end{cases} ; \end{aligned}$$

On franchit t_3 :

$$\begin{aligned} \sigma &= t_0 t_1 t_3, \sigma(x) = x_0 t_0 x_1 t_1 x_3 t_3, \mathcal{L}(\sigma(x)) = x_0 a x_1 b x_3 c ; \\ m_3 &= (p_3, p_{obs}), V_{m_3} = \{t_4, t_{obs}\}, \\ h_3 &= (\$, \$, \$, \$, x_3, \$, \$, \$, x_3), \\ B_3 &= \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 0 \leq x_3 \leq 2 \end{cases} ; \end{aligned}$$

On franchit t_4 :

$$\begin{aligned} \sigma &= t_0 t_1 t_3 t_4, \sigma(x) = x_0 t_0 x_1 t_1 x_3 t_3 x_4 t_4, \mathcal{L}(\sigma(x)) = x_0 a x_1 b x_3 c x_4 a ; \\ m_4 &= (p_4, p_{obs}), V_{m_4} = \{t_5, t_{obs}\}, \\ h_4 &= (\$, \$, \$, \$, \$, x_4, \$, \$, x_3 + x_4), \\ B_4 &= \begin{cases} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 0 \leq x_4 \leq 4 \\ 0 \leq x_3 + x_4 \leq 8 \end{cases} ; \end{aligned}$$

On franchit t_5 :

$$\begin{aligned} \sigma &= t_0 t_1 t_3 t_4 t_5, \sigma(x) = x_0 t_0 x_1 t_1 x_3 t_3 x_4 t_4 x_5 t_5, \mathcal{L}(\sigma(x)) = x_0 a x_1 b x_3 c x_4 a t_5 b ; \\ m_5 &= (p_5, p_{obs}), V_{m_5} = \{t_6, t_7, t_{obs}\}, \end{aligned}$$

$$h_5 = (\$, \$, \$, \$, \$, \$, x_5, x_5, \$, x_3 + x_4 + x_5),$$

$$B_5 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 2 \leq x_4 \leq 4 \\ 0 \leq x_5 \leq 0 \\ 0 \leq x_3 + x_4 \leq 8 \\ 0 \leq x_3 + x_4 + x_5 \leq 8 \end{array} \right\};$$

On franchit t_7 :

$$\sigma = t_0 t_1 t_3 t_4 t_5 t_7, \sigma(x) = x_0 t_0 x_1 t_1 x_3 t_3 x_4 t_4 x_5 t_5 x_7 t_7, \mathcal{L}(\sigma(x)) = x_0 a x_1 b x_3 c x_4 a x_5 b x_7 f;$$

$$m_6 = (p_6, p_{obs}), V_{m_6} = \{t_8\},$$

$$h_6 = (\$, \$, \$, \$, \$, \$, \$, \$, x_7, \$),$$

$$B_6 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 2 \\ 1 \leq x_1 \leq 3 \\ 4 < x_2 \leq 6 \\ 1 \leq x_3 \leq 2 \\ 2 \leq x_4 \leq 4 \\ 0 \leq x_5 \leq 0 \\ 0 \leq x_7 \leq 8 \\ 0 \leq x_3 + x_4 \leq 8 \\ 0 \leq x_3 + x_4 + x_5 \leq 8 \end{array} \right\};$$

La figure 3.11 représente le pronostiqueur généré à partir du graphe d'accessibilité logique et intégrant le système d'inéquations de la séquence qui se termine par un événement de défaillance f .

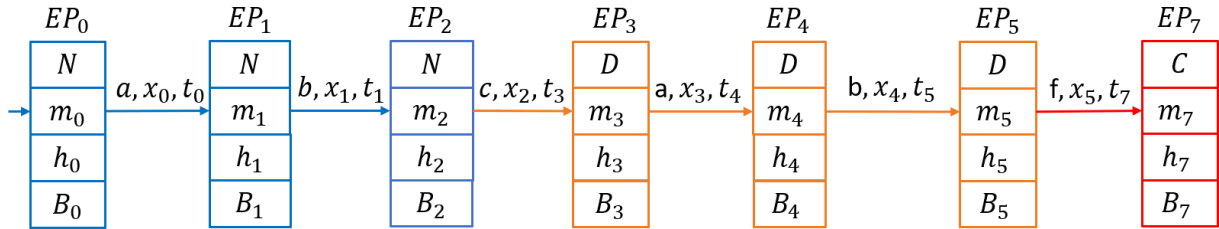


Figure 3.11: Modèle pronostiqueur avec la deuxième approche.

Une version réduite du modèle pronostiqueur qui permet l'exploitation de l'approche du pronostic pour des MMF avec un espace d'états beaucoup plus important. Cette méthode permet de réduire, le temps de calcul du système d'inéquations pour l'ensemble des sommets du pronostiqueur. Cependant elle ne permet pas de vérifier la pronosticabilité pour l'ensemble du modèle RdPTL.

3.2.6 Propriété de pronosticabilité

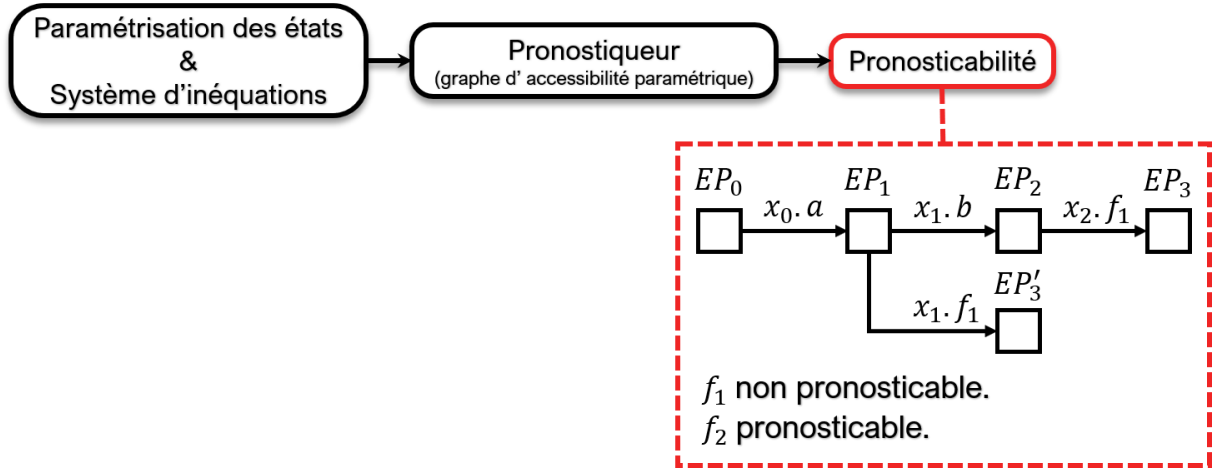


Figure 3.12: Vérification de la pronosticabilité.

Dans le cadre de cette approche, la propriété de pronosticabilité (figure 3.12) désigne la capacité de pronostiquer la date d'occurrence d'un événement défaillant dans une séquence donnée. Avant de discuter des conditions nécessaires et suffisantes pour vérifier cette propriété, il est important d'introduire d'abord la pronosticabilité dans le cas logique.

Selon Xiang Yin [113], la pronosticabilité appliquée aux RdPL est donnée par la définition suivante :

Définition 3.7 : (pronosticabilité) [110]

Soit $R_L = \langle P, T, \text{pré}, \text{post}, m_0, \Sigma, \mathcal{L} \rangle$ un RdPL, nous disons que R_L est pronosticable par rapport à t_f (transition avec un événement de défaillance) si :

- 1) $(\forall \alpha \in L(R_L, m_0) : t_f \in \alpha) (\exists \beta \in \bar{\alpha} : t_f \notin \beta)$; $\bar{\alpha}$ préfixe clos de α ,
- 2) $\forall \theta \in L(R_L, m_0) : \mathcal{L}(\theta) = \mathcal{L}(\beta) \wedge t_f \notin \theta, (\exists k \in \mathbb{N}) (\forall \gamma \in L(R_L, m_0)) ([\gamma] \geq k \wedge t_f \in \gamma)$;

Intuitivement la pronosticabilité est utilisée pour savoir si l'apparition de la défaillance dans le système peut être prévue sans ambiguïté ; donc toute séquence fautive doit avoir un préfixe ne contenant pas de défaillance pour laquelle l'occurrence de la défaillance est sûre dans k étapes.

Exemple :

Nous utilisons les deux modèles du RdPL de la figure 3.13 comme exemple, pour vérifier la propriété de pronosticabilité.

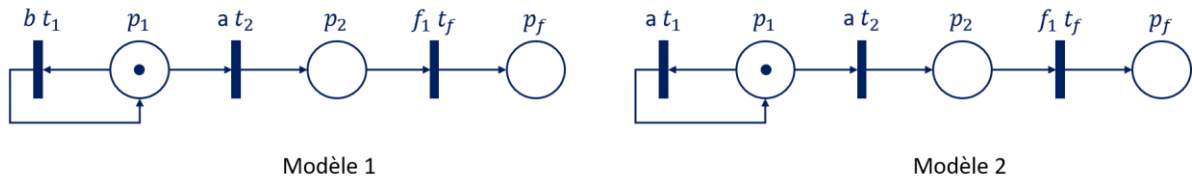


Figure 3.13: Pronosticabilité (modèles RdPL sans observateur).

Pour le modèle 1 de la figure 3.13

- 1) $\alpha = t_2 t_f, \bar{\alpha} = \{\varepsilon, t_2, t_2 t_f\}; \beta = \{t_2\} \in \bar{\alpha}$ et $t_f \notin \beta$,
- 2) soit $\theta = t_2, \mathcal{L}(\theta) = \mathcal{L}(\beta) = a$ et $t_f \notin \theta$,

$\exists k = 1$ tel que $\gamma = t_f$ et $\theta\gamma = t_2 t_f$, $\theta\gamma \in L(R_L, m_0)$ et $|\gamma| = 1$ avec $t_f \in \gamma$, donc le modèle 1 de la figure 3.13 est pronosticable par rapport à t_f .

Pour le modèle 2 de la figure 3.13 :

- 1) $\alpha = t_2 t_f$, $\beta = \{t_2\} \in \bar{\alpha}$ et $t_f \notin \beta$,
- 2) soit $\theta = t_1$, $\mathcal{L}(\theta) = \mathcal{L}(\beta) = \alpha$ et $t_f \notin \theta$, et pour tout $k \in \mathbb{N}$ une séquence non fautive $\gamma = t_1 t_1^*$ est définie dans $L(R_L, m_0)$ i.e. $\exists k \in \mathbb{N}$ tel que $\forall \theta\gamma \in L(R_L, m_0)[|\gamma| \geq k \wedge t_f \notin \gamma]$, donc le modèle 2 de la figure 3.13 n'est pas pronosticable par rapport à t_f .

[110] introduit les notions de marquage limite et de marquage non indicateur pour définir l'horizon (ou la limite) de son pronostic.

Définition 3.8 : (marquage limite et non indicateur)

Un marquage $m \in \mathbb{N}^*$ est dit :

- Un marquage limite si $\exists t_f$ tel que $m[t_f >$;
- Un marquage non indicateur si $(\forall k \in \mathbb{N})(\exists \sigma \in T_n)m[\sigma > \wedge |\sigma| \geq k$,

Intuitivement, un marquage limite est un marquage à partir duquel une transition fautive peut être immédiatement franchie. Un marquage non indicateur est un marquage à partir duquel une longue et arbitraire séquence non fautive est possible.

La caractérisation de la pronosticabilité en termes de marquages limites et marquage non indicateur est donnée par le lemme suivant :

Définition 3.9 : (marquage indicateur et non indicateur)[110]

Un RdPL $(R_L, m_0, \mathcal{L})$ est non pronosticable par rapport à t_f , si et seulement si, deux séquences non fautives $\sigma_1, \sigma_2 \in T_N^*$ tel que :

- 1) m_1 est non indicateur, où $m_0[\sigma_1 > m_1$;
- 2) m_2 est un marquage indicateur, où $m_0[\sigma_2 > m_2$;
- 3) $\mathcal{L}(\sigma_1) = \mathcal{L}(\sigma_2)$;

Dans notre cas, l'objectif est de vérifier si l'événement de défaillance est pronosticable τ unités de temps à l'avance. Autrement dit, on doit trouver un préfixe clos non défaillant à partir duquel l'occurrence d'une défaillance est sûre dans τ unités de temps à l'avance.

Nous proposons de modifier la définition 3.6.

Rappelons que $\sigma(x)$ est une séquence temporelle d'une séquence logique $\sigma, \sigma(x) = \text{timed}(\sigma)$, réciproquement $\sigma = \text{logic}(\sigma(x))$ et $s(\sigma) = \mathcal{L}(\sigma)$.

Soit $\sigma = t_1 t_2 \dots t_n$ et $\sigma(x) = x_0 t_1 x_1 \dots x_{n-1} t_n x_n$

$\text{delay}(\sigma(x)) = x_0 x_1 \dots x_{n-1} x_n$;

Définition 3.10 : (pronosticabilité temporelle)

Soit $R_{TL} = \langle P, T, \text{pré}, \text{post}, m_0, \Sigma, \mathcal{L}, I \rangle$ un RdPTL, avec $T = T_N \cup T_{deg} \cup T_{cri} \cup T_{rep}$. Un R_{TL} est dit pronosticable par rapport à t_f si :

- 1) $(\forall \alpha(x) \in G(R_{TL}) : t_f \in \alpha) (\exists \beta \in \bar{\alpha} : t_f \notin \beta)$;

- 2) $\forall \theta(x) \in G(R_{TL}) : \mathcal{L}(\theta) = \mathcal{L}(\beta) \wedge t_f \notin \theta, (\exists \tau \in \mathbb{R})(\forall \theta(x)\gamma(x) \in G(R_{TL})) \wedge \text{delay}(\gamma(x)) \geq \tau \wedge t_f \in \gamma]$;

Prenons cette fois-ci, deux modèles RdPTL pour vérifier la pronosticabilité avec des contraintes temporelles [figure 3.14](#). Le $G(R_{TL})$ est donné si après ([figure 3.15](#)).

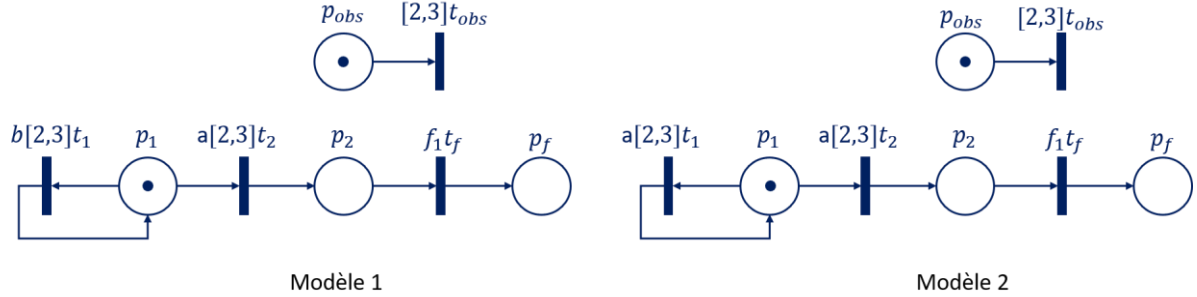


Figure 3.14: Pronosticabilité (modèles RdPTL avec observateur).

- 1) $\alpha(x) = x_0 t_2 x'_1 t_f$; $\alpha = t_2 t_f$; $\bar{\alpha} = \{\varepsilon, t_2, t_2 t_f\}$; $\beta = t_2$; $t_f \notin \beta$;
- 2) $\theta(x) = x_0 t_2$; $\theta = t_2$; $\mathcal{L}(\theta) = \mathcal{L}(\beta) = a$;

On prend $\theta(x)\gamma(x) = x_0 t_2 x'_1 t_f$ avec $\gamma(x) = x'_1 t_f$ et $\text{delay}(\gamma) = x'_1$ avec $2 \leq x'_1 \leq 3$ et $0 \leq x_0 + x'_1 \leq 3$ d'après le pronostiqueur de la [figure 3.15](#).

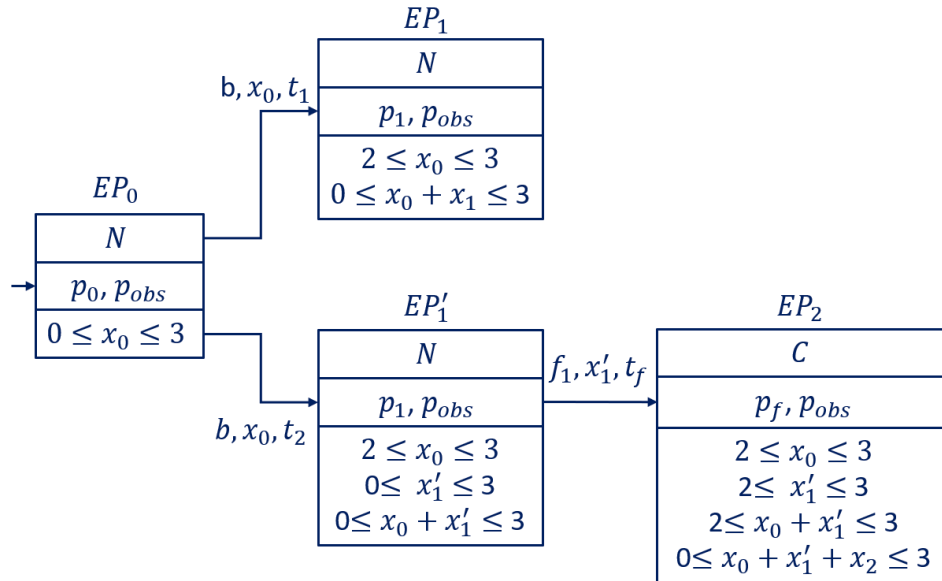


Figure 3.15: $G(R_{TL})$ Graphe d'accessibilité.

R_{TL} est pronosticable par rapport à t_f et l'occurrence de l'événement de défaillance f est prédite 2 unités de temps à l'avance.

Prenons comme exemple le modèle 2 de la [figure 3.14](#).

- 1) $\alpha(x) = x_0 t_2 x'_1 t_f$; $\alpha = t_2 t_f$; $\beta = t_2$; $t_f \notin \beta$;
- 2) $\theta(x) \in x_0 t_1$; $\theta = t_1$; $\mathcal{L}(\theta) = \mathcal{L}(\beta) = a$;

On prend $\theta(x)\gamma(x) = x_0 t_1 x'_1 t_1$ avec $\gamma(x) = x'_1 t_1$ et $\text{delay}(\gamma) = x'_1$, $2 \leq x_0 \leq 3$, $0 \leq x_0 + x'_1 \leq 3$ et $t_f \notin \gamma$. Si on répète cette boucle plusieurs fois, il n'y aura jamais occurrence de l'événement de défaillance f . Donc R_{TL} n'est pas pronosticable par rapport à t_f .

Remarque :

La propriété de pronosticabilité temporelle présentée dans la [définition 3.10](#) ne prend pas en considération les modes de fonctionnement. En effet, dans notre démarche nous ne commençons de pronostiquer que lorsque le système passe du mode nominal au mode dégradé. Ce qui n'est pas le cas de la [définition 3.10](#). Ainsi, nous proposons de modifier cette définition pour qu'elle prenne en considération les modes de fonctionnement.

Définition 3.11 : pronosticabilité temporelle étendue

Soit $R_{TL} = \langle P, T, pré, post, m_0, \Sigma, \mathcal{L}, I \rangle$ un RdPTL, avec $T = T_n \cup T_{deg} \cup T_{cri} \cup T_{rep}$. Un R_{TL} est dit pronosticable par rapport à t_f si :

- 1) $(\forall \alpha(x) \in A(R_{TL}) : t_f \in \alpha) (\exists \beta = t_0 t_1 \dots t_{j-1} \mid \beta \in \bar{\alpha} : t_f \notin \beta) \wedge t_0^*, t_1^* \dots t_{j-1}^* \in P_n \wedge t_j^* \in P_{deg}$
- 2) $\forall \theta(x) \in A(R_{TL}) : \mathcal{L}(\theta) = \mathcal{L}(\beta) \wedge t_f \notin \theta, (\exists \tau \in \mathbb{R}) (\forall \theta(x) \gamma(x) \in G(R_{TL})) \wedge delay(\gamma(x)) \geq \tau \wedge t_f \in \gamma]$;

Prenons l'exemple du modèle pronostiqueur de la [figure 3.10](#) pour vérifier la propriété de pronosticabilité selon la [définition 3.11](#) :

- 1) $\alpha(x) = x_0 t_0 x_1 t_1 x_2 t_3 x_3 t_4 x_4 t_5 x_5 t_7$;
 $\alpha = t_0 t_1 t_3 t_4 t_5 t_7$;
 $\bar{\alpha} = \{\varepsilon, t_0, t_0 t_1, t_0 t_1 t_3, t_0 t_1 t_3 t_4, t_0 t_1 t_3 t_4 t_5, t_0 t_1 t_3 t_4 t_5 t_7\}$;
 $\beta = t_0 t_1 ; t_7 \notin \beta$; avec $t_7 \in T_{cri}$;
- 2) $\theta(x) = x_0 t_0 x_1 t_1 ; \theta = t_0 t_1 ; \mathcal{L}(\theta) = \mathcal{L}(\beta) = a b$;

On prend $\theta(x) \gamma(x) = x_0 t_0 x_1 t_1$ avec $\gamma(x) = x_2 t_3 x_3 t_4 x_4 t_5$ et $delay(\gamma) = x_2 + x_3 + x_4 + x_5$ selon le système d'inéquation du sommet EP_5 :

$$4 < x_2 \leq 6, 1 \leq x_3 \leq 2, 2 \leq x_4 \leq 4, 0 \leq x_5 \leq 0 ;$$

Le MMF est pronosticable par rapport à t_7 et l'occurrence de l'événement de défaillance f est prédite 7 unités de temps à l'avance, après le franchissement de la transition t_3 . (l'événement est pronosticable aussi à partir de la transition t_0 et son pronostic est de 8 UT).

Maintenant pour caractériser la pronosticabilité par les RdPTL, nous allons utiliser les notions de marquage critique et marquage non indicateur.

Définition 3.12 : marquage critique et non indicateur

Un marquage $m \in \mathbb{N}^*$ est dit :

- Un marquage est critique si $\exists t_f$ tel que $m[t_f >$;
- Un marquage est non indicateur si $(\forall \tau \in \mathbb{R}) (\exists \sigma(x) \mid t \in T_n \cup T_{deg}) \wedge m[\sigma(x) > \wedge delay(\sigma(x)) \geq \tau$;

Intuitivement, un marquage critique est un marquage à partir duquel le franchissement d'une transition fautive est sûr. Un marquage non indicateur est un marquage à partir duquel le franchissement arbitraire d'une séquence, peut durer longtemps sans qu'il n'y ait occurrence d'un événement de défaillance.

La [figure 3.16](#) décrit l'intérêt de chaque étape de notre démarche de pronostic. La modélisation en modèle de modes de fonctionnement réduite à 3 modes séquentiels, permet

l'exécution d'un seul mode à la fois évitant toute incohérence due à une exécution parallèle des modes. L'explosion combinatoire d'états dans les RdPTL a été palliée par une paramétrisation des états où les valeurs des horloges sont calculées selon un système d'inéquations (voir algorithme 1 et algorithme 2). Ainsi, le franchissement des transitions ne peut être considéré à tout instant. Le résultat du système d'inéquations déterminera la date au plus tôt et au plus tard de l'occurrence de chaque événement. Sur la base de l'arbre d'accessibilité du MMF un pronostiqueur est généré. Les sommets du pronostiqueur seront réduits en fonction des états paramétriques similaires. La séquentialité des états paramétriques dans le pronostiqueur, sera la base de la vérification de la propriété de pronosticabilité ; tout séquence ayant une séquence similaire avec le même ordre logique des événements et avec le même système d'inéquations dans ces états est dites non pronosticable. A partir des séquences pronosticable et sur la base des résultats du système d'inéquations, la date au plus tard de l'occurrence d'un événement de défaillance est déterminée.

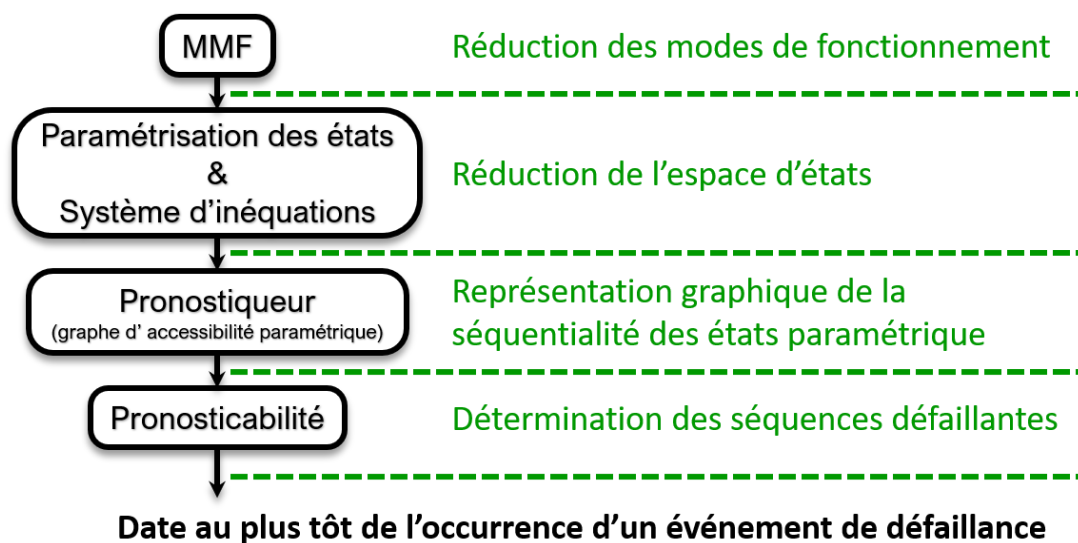


Figure 3.16: Démarche de pronostic.

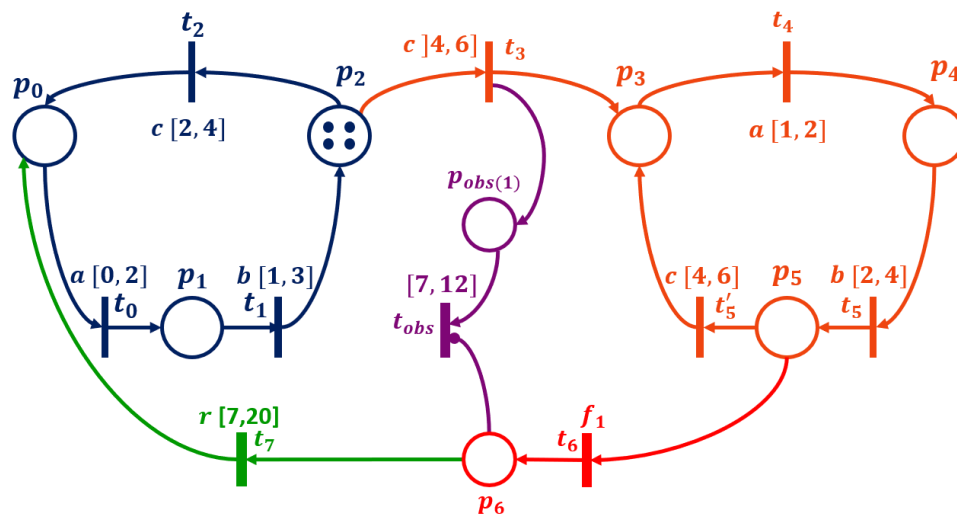
3.3. Discussions

3.3.1. Multi-composant

Après avoir décrit et vérifié l'approche du pronostic sur un modèle RdPTL sauf (voir la figure 3.4), nous nous intéressons dans cette section à étendre l'approche à un modèle multi-jeton. En attribuant un jeton par composant, le but est d'établir un pronostic sur un système multi-composant. Nous considérons que le MMF du système n'est plus sauf et qu'une seule transition est franchie à la fois. Nous proposons deux interprétations du problème du pronostic multi-composant :

La première interprétation ne fait pas de distinction entre les jetons, pas de priorité lors du franchissement d'une transition et considère la première évolution de l'un des jetons dans le mode dégradé, comme une menace d'un dysfonctionnement futur. Dans ce type de système la réparation d'un seul composant est souvent complexe et nécessite un temps d'intervention important, ce qui oblige l'opérateur à remplacer l'ensemble du système redouté par un nouveau. L'opérateur a besoin dans ce cas, d'un module de pronostic qui signale la date au plus tôt de l'occurrence d'une défaillance, sans avoir nécessairement une information sur le composant du système qui sera défaillant.

Considérons un système réparti en plusieurs sous-systèmes (composants), dont le fonctionnement est identique. Soit l'exemple de la [figure 3.17](#), un RdPTL qui n'est pas sauf, représentant le MMF du système. Chaque jeton du modèle, représente un composant et son évolution interprète l'état du composant. Nous supposons qu'une seule transition est franchie à la fois et nous utilisons un seul observateur pour faire le pronostic.



Dans ce mémoire on s'intéresse à la première interprétation où il n'y a ni distinction, ni priorité entre les jetons. Le premier franchissement de la transition t_3 par l'un des jetons présents sur la place p_2 , déclenche le processus du pronostic. Le modèle pronostiqueur (voir [figure 3.7](#)) du MMF de la [figure 3.17](#) restera identique à celui du MMF de la [figure 3.5](#). Ainsi, les résultats du pronostic seront les mêmes. L'événement de défaillance f_1 est pronosticable et son occurrence est prévue 3 unités de temps après le franchissement de la transition t_3 . En cas de franchissement d'un second jeton au niveau de la transition t_3 , c'est l'évolution du premier jeton dans le mode dégradé qui sera prise en considération pour faire le pronostic. Une autre interprétation pourrait être faite, cette démarche appartient aux perspectives.

Lors de l'étude, nous avons supposé plusieurs hypothèses; nous avons supposé qu'une seule défaillance est possible sur un RdPTL alors que dans le cas réel on peut avoir plusieurs événements de défaillance qui ne sont pas forcément observables. Ces événements de défaillance sont indépendants *i.e.* un événement de défaillance n'entraîne pas d'autre. Dans le cas où deux événements de défaillance sont successifs (dépendants); un événement de défaillance f_1 est suivi par un autre événement de défaillance f_2 , il faut distinguer f_1 de f_2 . Revoir le formalisme de la propriété de pronosticabilité dans ce cas est nécessaire. Néanmoins, nous avons proposé comme perspective dans le dernier chapitre de ce mémoire, des pistes qui peuvent être exploitées dans des futurs travaux de recherche orientés pronostic.

La proposition d'un module de pronostic pour un système multi-composant est limité dans notre approche aux composants avec comportement identique, *i.e.* leur mode de fonctionnement évolue de la même façon. Il serait alors intéressant de vérifier l'approche sur des systèmes multi-composant avec un comportement différent, pour vérifier le parallélisme, la priorité et déterminer à quel niveau il est possible d'agir pour signer l'occurrence future d'un événement de défaillance. Nous avons considéré que l'ensemble des événements sont observables, alors que dans le mode de fonctionnement réel, ce n'est pas toujours le cas. L'inobservabilité des événements affecte les résultats du pronostic, si les projections et les signatures temporelles de certaines séquences du RdPTL sont identiques et conduisent à une ambiguïté dans le module de pronostic. Sans la paramétrisation des états, la génération d'un graphe d'accessibilité aurait pu être complexe. Cependant, ce problème n'est pas résolu dans son intégralité, *i.e.* pour des architectures beaucoup plus complexes, la génération de l'arbre d'accessibilité et l'outil de simulation actuelle, nécessitent un temps important pour le générer. Dans la section suivante, nous avons choisi INA comme outil de simulation.

3.3.3.Simulation INA

Pour vérifier les résultats obtenus par le système d'inéquations et déterminer la date d'occurrence au plus tôt d'un événement de défaillance, nous avons cherché parmi les différents outils d'analyse des systèmes formalisés en réseaux de Petri (TINA [114], WoPeD [115], Romeo [116] ...); Il fallait trouver un outil de simulation cohérent approximativement (puisque'il n'existe pas un outil adapté à 100%) avec l'analyse que l'on souhaite établir. Un outil qui permet d'interpréter à la fois l'aspect logique (ordre d'occurrence) et temporel (date d'occurrence) d'un système pour conclure sur le temps d'exécution minimal d'une séquence et précisément la séquence qui se termine avec un événement de défaillance. Nous avons choisi l'outil INA (Integrated net analyzer) ([annexe 2](#)) développé par le professeur Peter H. Starke [117], non seulement pour sa capacité de répondre à des questions relatives à la vivacité, l'atteignabilité, la réversibilité ou la nature bornée, mais aussi pour ses résultats de calcul des cycles d'exécution min et max dans le modèle, qui se réfère au formalisme proposé par Popova [118] et qui a été la base de nos travaux.

INA ne permet pas de représenter graphiquement un modèle RdP et nécessite sa transformation en une description textuelle, qui est souvent source d'erreurs. Nous avons remarqué que le fichier généré par INA lors de l'enregistrement du code descriptif est d'une extension ".pnt". Cette extension est la même générée lors de l'enregistrement d'un modèle graphique sur TINA. Nous avons décidé alors de réaliser le modèle graphique sur TINA et charger le fichier généré ensuite sur l'outil INA pour l'analyser.

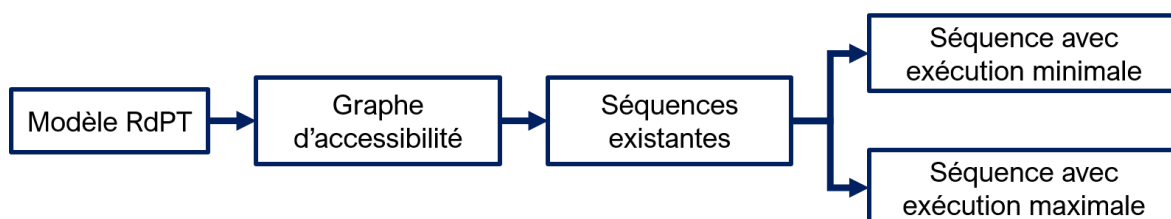


Figure 3.18: Etape de simulation d'un modèle RdP

La [figure 3.18](#) illustre les étapes à suivre pour obtenir les séquences d'exécution minimale et maximale d'un modèle RdP. Après la réalisation du modèle RdP sur TINA, il faut charger le fichier sur l'outil INA pour effectuer l'analyse du système. Lors de cette analyse, le graphe d'accessibilité est généré pour pouvoir déterminer l'ensemble des séquences possibles. A partir de ces séquences de la version réduite du pronostiqueur (exemple [figure 3.11](#)) INA identifie celles avec une exécution minimale et maximale. (Voir [Annexe 1](#)). Le graphe d'accessibilité généré par INA est très grand (espace d'états élevé) et nécessite un temps de

traitement important. Cependant, avec le modèle pronostiqueur généré par la deuxième méthode, il serait possible de déterminer le temps d'exécution minimal de la ou les séquences défaillante(s). Développer un outil à base du système d'inéquation proposé est envisageable comme perspective, pour réduire le temps d'analyse temporel des RdP.

3.4. Conclusion

Dans ce chapitre, nous avons donné une description sur l'approche de pronostic de la date au plus tôt de l'occurrence d'un événement de défaillance. Dans la première étape, nous avons modélisé le comportement du système avec des RdPTL pour pouvoir effectuer une analyse à la fois logique (occurrence des événements) et temporelle (dates d'occurrence des événements) sur le modèle. Sur la base de ce modèle, nous avons construit un modèle pronostiqueur; une représentation compacte du modèle des modes de fonctionnement (MMF) du système, qui permet de discerner les séquences qui se terminent par un événement de défaillance de celles qui ne le sont pas. Après l'identification de ces séquences nous avons introduit la propriété de pronosticabilité, qui vérifie la possibilité de faire le pronostic ou non. Nous avons proposé un algorithme de pronostic qui permet de déterminer la date d'exécution minimale de la séquence pronosticable et calculé la date d'occurrence d'un événement de défaillance au plus tôt. Nous avons conclu le chapitre sur les hypothèses et limites et par une présentation de l'outil de simulation qui sera utilisé dans le chapitre suivant pour vérifier les résultats du pronostic sur un benchmark.

CHAPITRE 4 : Benchmark

4. Benchmark

4.1. Introduction

Dans ce chapitre, nous avons choisi comme benchmark, la cellule d'une batterie de voiture électrique. Le but est de signaler au conducteur à l'avance la date au plus tôt d'un dysfonctionnement total de la batterie. Ce benchmark est basé sur l'approche de pronostic proposée dans le chapitre 3. Nous allons commencer par décrire le système, son fonctionnement et son interaction avec l'environnement. Pour identifier les sources de dégradation de la cellule et ses pannes, nous allons déterminer les relations de causes à effets entre ses états de fonctionnement. Pour la vérification de notre approche de pronostic, nous allons construire un modèle de modes de fonctionnement d'une seule cellule basée sur un RdPTL. À partir de l'arbre d'accessibilité de ce modèle, nous allons générer un modèle pronostiqueur qui va nous permettre de discerner l'ensemble des séquences d'événements susceptibles d'altérer le fonctionnement de la cellule (des séquences qui se terminent avec un événement de défaillance). Nous allons effectuer une paramétrisation des états de ces séquences i.e. introduire une horloge et un système d'inéquations des horloges pour chaque état du système. Ensuite, nous allons vérifier si les événements de défaillance sont pronosticables ou non à partir d'un sommet du modèle pronostiqueur. Le système d'inéquations, va nous permettre de déterminer la date d'occurrence au plus tôt des événements de défaillance dans les séquences pronosticables et d'en déduire la date au plus tôt de l'occurrence d'un événement de défaillance future.

4.2. Description du système : Les cellules de batterie

Nous allons donner dans cette section un aperçu sur l'utilisation et l'évolution des batteries, en commençant par la pile d'Alessandro Volta jusqu'aux batteries lithium.

En 1799, Alessandro Volta inventa la première Pile électrochimique qui produit de l'énergie électrique, basée sur l'empilement du zinc et du cuivre intercalé de carton humidifié avec un acide [119]. Un fois déchargée, il fallait humecter les cartons d'acide et remplacer les plaques de cuivre pour une nouvelle utilisation. En 1859, Gaston Planté proposa une nouvelle version de batterie rechargeable ; une batterie à électrodes d'acide-plomb, la plus utilisée jusqu'à nos jours dans l'industrie automobile (plus de 160 ans), connue pour sa capacité de décharge élevée et sa fabrication simplifiée. Cependant, leur volume et poids important ne les adaptent pas pour une utilisation sur certains équipements électriques (les télécommandes, les montres, ...). C'est à la fin des années 1950, que la pile alcaline a vu le jour pour donner naissance à des équipements électriques sans fils. Pendant les années 1970, plusieurs batteries électrochimiques ont apparues, basées sur le nickel-carbone, le zinc-argent, le zinc-manganèse, le zinc-mercure et le lithium. Ce n'est qu'au début des années 1990, que les premières cellules lithium ont été mises sur le marché, contribuant ainsi à l'apparition des ordinateurs portables, téléphones cellulaires et autres équipements qui nécessitent une énergie beaucoup plus importante. La première version de cellule à base de lithium est non rechargeable alors que la seconde est rechargeable avec une appellation lithium-ion. Étant très abondant et pas nocif pour l'environnement son utilisation a été étendue pour des applications de petite et grande puissance (voitures électriques).

La plupart des travaux réalisés sur les batteries et particulièrement ceux des voitures électriques qui fonctionnent avec le Lithium, s'intéressent à la quantité de l'énergie électrique stockée et à la juxtaposition des cellules pour analyser leur état de santé. Une énergie électrique est emmagasinée dans la cellule sous format chimique entre deux électrodes

(l'anode et la cathode) séparées par un électrolyte. L'électrolyte assure le déplacement de la composante ionique résultante de la réaction chimique à l'intérieur de la cellule et la force à sortir de la batterie. Il en résulte un courant électrique de décharge (I) et une tension (V) pendant un temps (Δt). Cette réaction chimique doit être réversible pour une batterie rechargeable sous l'application d'une tension de charge. Parmi les principaux paramètres critiques d'une batterie rechargeable on peut citer : la sécurité, la densité d'énergie qui peut être stockée dans une puissance d'entrée et une récupération de sortie spécifiques, le nombre de cycles charge/décharge, la durée de vie, l'efficacité du stockage et le coût de fabrication.

La figure 4.1 est une présentation simplifiée de la structure des cellules dans une batterie Lithium. Les cellules sont interconnectées entre elles par des liaisons séries parallèles, pour assurer le stockage d'un maximum de l'énergie électrique. Une énergie emmagasinée pour une utilisation ultérieure, afin d'alimenter une charge (tableau de bord, démarreur, climatiseur, éclairage, signalisation ...). Plusieurs familles de batteries ont été proposées par la communauté scientifique, la plus récente est la batterie Métal-air. La plupart de des travaux s'intéressent principalement à augmenter la capacité de stockage de l'énergie soit par la réduction de la taille des batteries, soit par le changement de matériaux constituant la batterie, modifiant ainsi la réaction chimique à l'intérieur des cellules ; 4 chimies sont utilisées en 2020 : Lithium métal oxide, Lithium nickel manganèse cobalt oxide (la plus utilisée), Lithium iron Phosphate et Lithium nickel cobalt aluminium oxide. D'autres travaux s'intéressent aux défaillances et aux modes de fonctionnement des cellules de la batterie [120]. La première version de batterie Lithium ion par exemple, s'explode dans un environnement de fonctionnement avec des températures élevées ou bien si l'état de charge des cellules n'est pas le même, provoquant la surchauffe de la batterie. Ce qui met non seulement l'état des équipements en danger, mais aussi la vie des utilisateurs. C'est la raison pour laquelle, les constructeurs limitent l'utilisation des batteries sur une plage de capacité entre 25 et 85%. De cette façon, la batterie ne sera par chargée plus de 85%, et la voiture s'arrête quand sa capacité de charge est à 25%. En conséquence, 40% de la capacité de la batterie restent inexploitable.

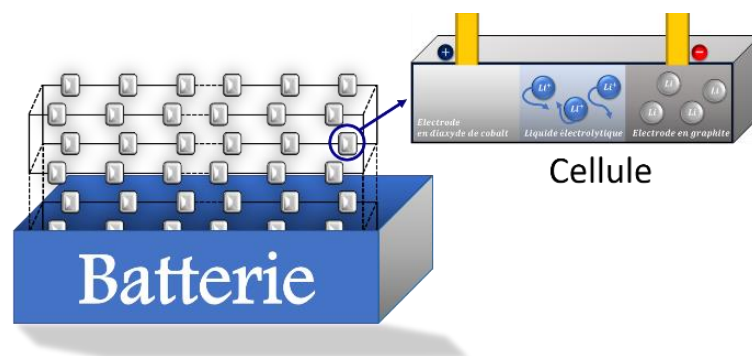


Figure 4.1: Représentation simplifiée d'une batterie Lithium.

Indépendamment de la technologie utilisée et sans se baser sur une étude comparative entre les différents types de batteries, puisqu'ils ne font pas l'objet de cette thèse, nous nous intéressons particulièrement dans ce chapitre, aux modes de fonctionnement d'une cellule de batterie, afin de prédire toute défaillance possible à l'avance. Chaque cellule est caractérisée par des paramètres qui permettent de justifier le choix d'utilisation d'une batterie :

- **Capacité d'une cellule** : quantité de charge électrique que contient une cellule;
- **Capacité instantanée** : quantité de charge électrique contenue dans une cellule à un instant donné;

- **SoC** : état de charge d'une cellule, mesurant la quantité de charge restante par rapport à sa capacité maximale.
- **Durabilité** : temps pendant lequel une cellule pleinement chargée peut fournir du courant sous son régime nominal de décharge sans être rechargée.
- **Cyclabilité** : nombre de cycles de décharge et de recharge que peut subir une cellule avant d'être considérée comme trop âgée.

Le nombre de cycles théoriques de chaque type de batterie :

- Lithium Titanate ($\text{Li}_4\text{Ti}_5\text{O}_{12}$) - triplette - 3000–7000
- Lithium Manganese Oxide (LiMn_2O_4) - Leaf 1, Kangoo 1, Zoé 22 - 300–700,
- Lithium Nickel Manganese Cobalt Oxide (LiNiMnCoO_2 or NMC) - Zoé 40 - 1000–2000,
- Lithium Iron Phosphate (LiFePO_4) - Mia - 1000–2000,
- Lithium Nickel Cobalt Aluminum Oxide (LiNiCoAlO_2 or NCA) - 500.
- **Durée de vie** : donnée constructeur désignant la cyclabilité que peut subir une cellule avant d'être considérée comme trop âgée.
- **Durée de vie restante** : le nombre de cycles que peut encore subir une cellule avant d'être considérée comme altérée.

Les défaillances dans une cellule sont classifiées en deux catégories :

- Internes :
 - Température
 - Durée de vie
 - Surcharge
 - Décharge à vide
 - Vieillesse cyclique
 - Vieillesse calendaire
- Externes :
 - Emplacement de la cellule
 - La résistance en série
 - Température

Nous nous intéressons dans notre étude aux sources de défaillance internes et particulièrement à la surcharge de la cellule, sa décharge à vide et son vieillissement cyclique. L'analyse de ces sources de défaillances nécessite des informations sur la capacité de la cellule, sa capacité instantanée, sa durabilité et sa cyclabilité. La composition série parallèle des cellules permet d'avoir un dimensionnement énergétique de la batterie. Le calcul de la durée de vie d'une cellule dans cette composition série parallèle a un impact sur l'exploitation de la batterie et par ailleurs, l'utilisation des systèmes de pilotage de composition série parallèle de la batterie.

4.3. Méthode de pronostic

Rappelons que le Framework (figure 3.5) de notre approche de pronostic est composé d'un modèle de modes de fonctionnement (MMF) basé sur un RdPTL ; il représente le

comportement du système sous forme de trois modes de fonctionnement possibles (nominal, dégradé et critique). Il est composé aussi d'un module de pronostic, constitué d'un pronostiqueur, généré à partir de l'arbre d'accessibilité du MMF ; il représente l'ensemble des séquences du modèle et principalement celles qui se terminent par un événement de défaillance. La paramétrisation des états est introduite dans le modèle du pronostiqueur en utilisant un système d'inéquations pour optimiser l'espace d'états de l'arbre d'accessibilité. La propriété de pronosticabilité est vérifiée à partir de ce pronostiqueur afin de déterminer les événements de défaillance qui sont pronosticables (la possibilité de calculer la date au plus tôt de l'occurrence de l'événement de défaillance). Ainsi, à partir de la séquence avec une durée d'exécution minimale, nous pouvons déterminer la date au plus tôt d'occurrence d'un événement de défaillance.

4.3.1 Modèle RdPTL du système

Nous construisons le MMF du benchmark à partir de la description fonctionnelle de la cellule (voir la [annexe 1](#)). Nous avons représenté son fonctionnement en 4 états ([figure 4.2](#)) : chargé, en charge, déchargé, en déchargement. Les transitions d'états sont conditionnées par des événements et des intervalles de franchissement. Comme nous l'avons indiqué avant, nous supposons que l'ensemble des événements sont observables et le franchissement des transitions est conditionné par l'occurrence des événements dans leurs intervalles de tir (sans l'occurrence de l'événement le franchissement de la transition est impossible). Le tir des transitions est immédiat.

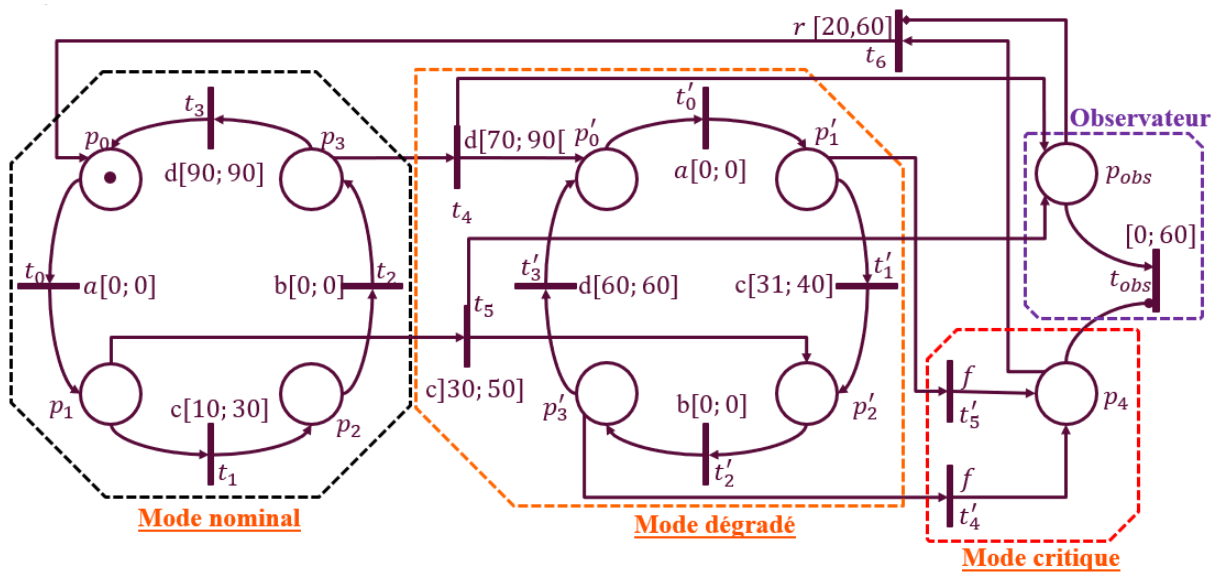


Figure 4.2: MMF d'une cellule de batterie électrique.

Les événements du MMF de la cellule sont :

- a: Début du chargement
- b: Début du déchargement
- c: Niveau maximum de charge
- d: Niveau minimum de décharge
- r: événement de réparation du système
- f: événement de défaillance

Les places du MMF sont :

- p_0 : état déchargé dans le mode nominal
- p_1 : état en chargement dans le mode nominal
- p_2 : état chargé dans le mode nominal
- p_3 : état en déchargement dans le mode nominal
- p'_0 : état déchargé dans le mode dégradé
- p'_1 : état en chargement dans le mode dégradé
- p'_2 : état chargé dans le mode dégradé
- p'_3 : état en déchargement dans le mode dégradé
- p_4 : état de panne dans le mode critique
- p_{obs} : place de l'observateur

Le passage du mode de fonctionnement nominal au mode dégradé est représenté par deux transitions t_4 et t_5 . La transition t_4 est franchissable si la cellule est déchargée entre 70 et 89 unités de temps (UT) ; ce qui représente un déchargement dégradé de la cellule (déchargement inférieur à 90 UT). Cependant, le franchissement de la transition t_5 est effectué si la cellule se charge entre 31 et 50 UT ; ce qui représente une surcharge de la cellule i.e. le chargement d'une cellule en fonctionnement nominal s'effectue avant 30 UT. La place et la transition de l'observateur (p_{obs} et t_{obs}) n'influencent pas la représentation du comportement du système sur le MMF. Elles permettent simplement, de suivre l'évolution d'un jeton dans le mode dégradé (on rappelle que le RdPTL est sauf). L'occurrence d'un événement de défaillance f bascule le système vers un mode de fonctionnement critique et par conséquent vers un état de panne. Deux transitions permettent ce changement de mode : les transitions t'_4 et t'_5 . Le franchissement de l'une de ces transitions est conditionné par l'occurrence de l'événement f et aucun intervalle de tir n'est défini (franchissement à tout instant de l'occurrence de f). L'intervalle $[0, 60]$ de t_{obs} , garde la transition de l'observateur sensibilisée pendant le fonctionnement dégradé (sensibiliser t_{obs} déclenche le calcul du système d'inéquations). Le marquage de la place p_4 désensibilise la transition t_{obs} avec l'arc inhibiteur (arrêt du calcul du système d'inéquations) .

4.3.2 Modèle pronostiqueur

Rappelons que dans le chapitre 3 nous avons proposé deux versions de pronostiqueurs. La première version, génère un graphe d'accessibilité paramétrique, où nous calculons pour chaque sommet le système d'inéquations. Ce qui permet d'avoir un modèle plus expressif (en termes de représentation d'états et transitions d'états) et avec un espace d'états réduit ; l'utilisation d'une variable d'horloge (dont les valeurs sont dans $\mathbb{N}$) et le calcul du système d'inéquations, pour palier au problème d'explosion d'états dans les Rdp temporels qui considèrent que le franchissement des transitions est possible à tout moment.

Chaque sommet du pronostiqueur est étiqueté de façon à ce qu'il représente le mode de fonctionnement dans lequel se trouve le système. Ainsi, les sommets ($EP_0, EP_1, EP_2, EP_3, EP_{0\ dup}$) sont étiquetés par N pour le mode de fonctionnement nominal, les sommets ($EP'_0, EP'_1, EP'_2, EP'_3, EP'_{0\ dup}, EP'_{2\ dup}$) sont étiquetés par D pour le mode de fonctionnement dégradé et les sommets (EP_4, EP'_4) sont étiquetés par C pour le mode de fonctionnement critique. Comme nous l'avons mentionné dans le précédent chapitre, la

fonction d'étiquetage se base sur l'appartenance des places du sommet à l'un des modes de fonctionnement du MMF (figure 4.2) i.e. le sommet EP_0 par exemple, est composé d'un marquage initial $m_0 = (p_0)$ et d'un système d'inéquations B_0 , de h_0 et l'étiquette N. Rappelons qu'un sommet ne peut contenir des places appartenant à deux modes de fonctionnement différents. Vu que la place p_0 appartienne au mode de fonctionnement nominal dans le MMF, le sommet EP_0 sera alors étiqueté par N. Le système d'inéquations B_0 détermine les valeurs possibles de l'horloge x_0 pour basculer au sommet EP_1 suite à l'occurrence de l'événement a .

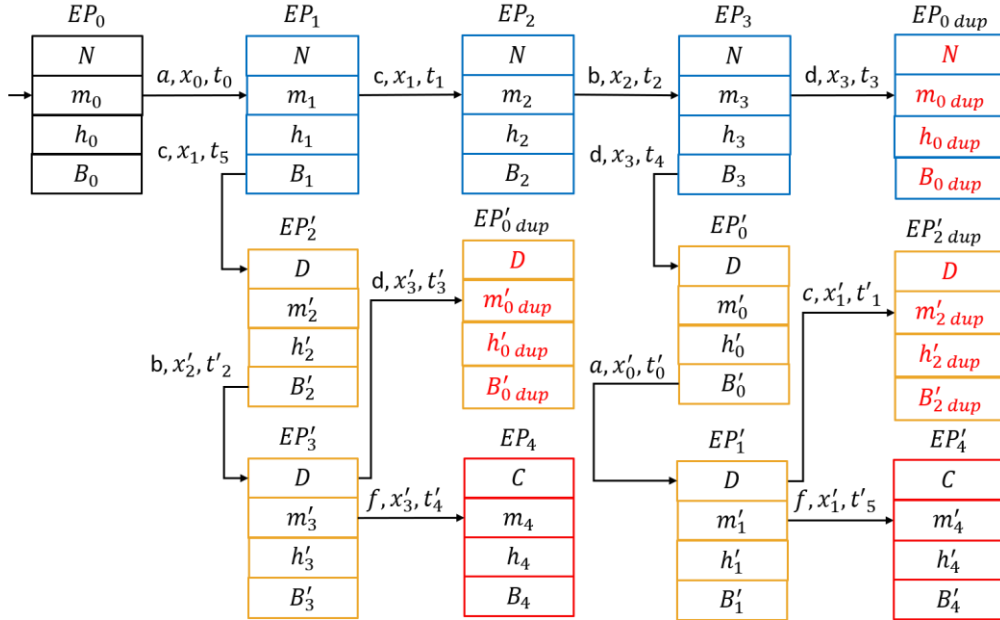


Figure 4.3: Modèle pronostiqueur d'une cellule de Batterie électrique (Approche 1).

Le pronostiqueur du MMF (figure 4.2) est représenté sur la figure 4.3. Chaque sommet du pronostiqueur est défini selon l'algorithme 3 de la section 3.2.5. Pour simplifier le modèle du pronostiqueur, nous n'avons pas représenté les sommets générés par le franchissement de la transition t_{obs} .

L'application de l'algorithme 3 du développement du pronostiqueur (figure 4.3) au modèle de l'exemple de la figure 4.2 donne le résultat suivant

Au début

$Q = \{EP_0\}$, $S := \emptyset$, $A := \emptyset$;
 $m_0 = (p_0)$, $V_{m_0} = \{t_0\}$;
 $h_0 = (x_0, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T$,
 etiquette = N,
 $B_0 = \{0 \leq x_0 \leq 0\}$;

On choisit EP_0 :

$Q := Q - \{EP_0\} := \emptyset$, $S := S \cup \{EP_0\} := \{EP_0\}$,
 $V'_{m_0} := V_{m_0} := \{t_0\}$, etiquette := N, etiq := etiquette;

A partir de EP_0 , on franchit t_0 et on atteint EP_1 ,

$t_0 \in T_N$ alors etiquette := N, $V'_{m_0} := V'_{m_0} - \{t_0\} := \emptyset$,
 $m_1 = (p_1)$, $V_{m_1} := \{t_1, t_5\}$;
 $h_1 = (\$, x_1, \$, \$, \$, x_1, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T$;
 etiq = etiquette ;

$B_1 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 0 \leq x_1 \leq 30 \end{array} \right\};$
 $\mathcal{L}(t_0) = a$, alors panne := false ;
 $A := \{EP_0, (a, x_0, t_0), EP_1\}$, $Q := Q \cup \{EP_1\} = \{EP_1\}$;

On choisit EP_1 :

$Q := Q - \{EP_1\} := \emptyset$, $S := S \cup \{EP_1\} := \{EP_0, EP_1\}$, $m_1 = (p_1)$
 $V'_{m_1} := V_{m_1} := \{t_1, t_5\}$, etiq := etiquette, avec etiquette := N,

A partir de EP_1 , on franchit t_1 et on atteint EP_2 :

$t_1 \in T_N$ alors etiquette := N, $V'_{m_1} := V'_{m_1} - \{t_1\} := \{t_5\}$,
 $m_2 = (p_2)$, $V_{m_2} := \{t_2\}$;
 $h_2 = (\$, \$, x_2, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T$;
 etiq = etiquette ;
 $B_2 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \end{array} \right\};$
 $\mathcal{L}(t_1) = c$, alors panne := false ;
 $A := \{EP_1, (c, x_1, t_1), EP_2\}$, $Q := Q \cup \{EP_2\} = \{EP_2\}$, $S := \{EP_0, EP_1, EP_2\}$;

A partir de EP_1 , on franchit t_5 et on atteint EP'_2 :

$t_5 \in T_{deg}$ alors etiquette := D, $V'_{m_1} := V'_{m_1} - \{t_5\} := \emptyset$,
 $m'_2 = (p'_2, p_{obs})$, $V_{m'_2} := \{t'_2, t_{obs}\}$;
 $h'_2 = (\$, \$, \$, \$, \$, \$, \$, x'_2, \$, \$, \$, x'_2)^T$;
 etiq = N $\neq$ etiquette = D, donc etiq $\neq$ etiquette alors ;
 $B'_2 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 31 < x_1 \leq 50 \\ 0 \leq x'_2 \leq 0 \end{array} \right\};$
 $\mathcal{L}(t_5) = c$, alors panne := false ;
 $A := \{EP_1, (c, x_1, t_5), EP'_2\}$, $Q := Q \cup \{EP'_2\} = \{EP_2, EP'_2\}$, $S := \{EP_0, EP_1, EP'_2\}$;

On choisit EP_2 :

$Q := Q - \{EP_2\} := EP'_2$, $S := S \cup \{EP_2\} := \{EP_0, EP_1, EP_2\}$, $m_2 = (p_2)$
 $V'_{m_2} := V_{m_2} := \{t_2\}$, etiq := etiquette, avec etiquette := N,

A partir de EP_2 , on franchit t_2 et on atteint EP_3 :

$t_2 \in T_N$ alors etiquette := N, $V'_{m_2} := V'_{m_2} - \{t_2\} := \emptyset$,
 $m_3 = (p_3)$, $V_{m_3} := \{t_3, t_4\}$;
 $h_3 = (\$, \$, \$, x_3, x_3, \$, \$, \$, \$, \$, \$, \$)^T$;
 etiq := etiquette ;
 $B_3 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 0 \leq x_3 < 90 \end{array} \right\};$
 $\mathcal{L}(t_2) = b$, alors panne := false ;
 $A := \{EP_2, (b, x_2, t_2), EP_3\}$, $Q = \{EP'_2, EP_3\}$;

On choisit EP'_2 :

$Q := Q - \{EP'_2\} := \{EP_3\}$, $S := S \cup \{EP'_2\} := \{EP_0, EP_1, EP_2, EP'_2\}$, $m'_2 = (p'_2, p_{obs})$,
 $V'_{m'_2} := V_{m'_2} := \{t'_2, t_{obs}\}$, $etiq := etiquette$, avec $etiquette := D$,

A partir de EP'_2 , on franchit t'_2 et on atteint EP'_3 :

$t'_2 \in T_{deg}$ alors $etiquette := D$, $V'_{m'_2} := V'_{m'_2} - \{t'_2\} := \{t_{obs}\}$,

$m'_3 = (p'_3, p_{obs})$, $V_{m'_3} := \{t'_3, t'_4, t_{obs}\}$;

$h'_3 = (\$, \$, \$, \$, \$, \$, \$, \$, \$, x'_3, x'_3, \$, \$, x'_2 + x'_3)^T$;

$etiq := etiquette$;

$$B'_3 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 31 < x_1 \leq 50 \\ 0 \leq x'_2 \leq 0 \\ 0 \leq x'_3 \leq 0 \\ 0 \leq x'_2 + x'_3 \leq 60 \end{array} \right\} ;$$

$\mathcal{L}(t'_2) = d$, alors $panne := false$;

$A := \{EP'_2, (b, x'_2, t'_2), EP'_3\}$, $Q = \{EP_3, EP'_3\}$;

Le franchissement de la transition t_{obs} ne sera pas traité, vu qu'il n'affecte pas le fonctionnement du système.

On choisit EP_3 :

$Q := Q - \{EP_3\} := \{EP'_3\}$, $S := S \cup \{EP_3\} := \{EP_0, EP_1, EP_2, EP'_2, EP_3\}$, $m_3 = (p_3)$

$V'_{m_3} := V_{m_3} := \{t_3, t_4\}$, alors que $t_3 \in T_N$ et $t_4 \in T_{deg}$ $etiq := \emptyset$,

A partir de EP_3 , on franchit t_3 et on atteint $EP_{0\ dup}$:

$t_3 \in T_N$ alors $etiquette := N$, $V'_{m_3} := V'_{m_3} - \{t_3\} := \{t_4\}$,

$m_{0\ dup} = m_0 = (p_0)$; "c'est un marquage existant $m_{0\ dup} = m_0$, Q ne sera pas étendu".

$V_{m_{0\ dup}} := \{t_0\}$;

$h_{0\ dup} = (x_{0\ dup}, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T$;

$etiq := \emptyset$;

$$B_{0\ dup} = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 90 \leq x_3 \leq 90 \\ 0 \leq x_{0\ dup} \leq 0 \end{array} \right\} ;$$

$\mathcal{L}(t_3) = d$, alors $panne := false$;

$S := S \cup \{EP_{0\ dup}\} = \{EP_0, EP_1, EP_2, EP'_2, EP_3, EP_{0\ dup}\}$, $A := \{EP_3, (d, x_3, t_3), EP_{0\ dup}\}$,

A partir de EP_3 , on franchit t_4 et on atteint EP'_0 :

$t_4 \in T_{deg}$ alors $etiquette := D$, $V'_{m_3} := V'_{m_3} - \{t_4\} := \emptyset$,

$m'_0 = (p'_0, p_{obs})$, $V_{m'_0} := \{t'_0, t_{obs}\}$;

$h'_0 = (\$, \$, \$, \$, \$, \$, \$, \$, \$, x'_0, \$, \$, \$, x'_0)^T$;

$etiq \neq etiquette$;

$$B'_0 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 70 \leq x_3 < 90 \\ 0 \leq x'_0 \leq 0 \end{array} \right\} ;$$

$\mathcal{L}(t_4) = d$, alors $panne := false$;

$$A := \{EP_3, (d, x_3, t_4), EP'_0\}, Q = \{EP'_3, EP'_0\};$$

On choisit EP'_3 :

$$Q := Q - \{EP'_3\} := \{EP'_0\}, S := S \cup \{EP'_3\} := \{EP_0, EP_1, EP_2, EP'_2, EP_3, EP_{0\text{ dup}}, EP'_3\}, m'_3 = (p'_3, p_{obs}), \text{etiquette} := D, \\ V'_{m'_3} := V_{m'_3} := \{t'_3, t'_4, t_{obs}\}, \text{or } t'_3 \in T_{deg} \text{ et } t'_4 \in T_{cri} \text{ alors etiq} := \emptyset,$$

A partir de EP'_3 , on franchit t'_3 et on atteint $EP'_{0\text{ dup}}$:

$$t'_3 \in T_{deg} \text{ alors etiquette} := D, V'_{m'_3} := V_{m'_3} - \{t'_3\} := \{t'_4, t_{obs}\}, \\ m'_{0\text{ dup}} = m'_0 = (p'_0, p_{obs}); \text{"c'est un marquage existant, } Q \text{ ne sera pas étendu".} \\ V'_{m'_{0\text{ dup}}} := \{t'_0, t_{obs}\}; \\ h'_{0\text{ dup}} = (\$, \$, \$, \$, \$, \$, x'_{0\text{ dup}}, \$, \$, \$, \$, \$, x'_2 + x'_3 + x'_{0\text{ dup}})^T; \\ \text{etiq} := \emptyset \text{ alors}$$

$$B'_{0\text{ dup}} = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 31 < x_1 \leq 50 \\ 0 \leq x'_2 \leq 0 \\ 0 \leq x'_3 \leq 0 \\ 60 \leq x'_{0\text{ dup}} \leq 60 \\ 0 \leq x'_2 + x'_3 \leq 60 \\ 0 \leq x'_2 + x'_3 + x'_{0\text{ dup}} \leq 60 \end{array} \right\};$$

$$\mathcal{L}(t'_3) = d, \text{ alors panne} := \text{false}; \\ S := S \cup \{EP'_{0\text{ dup}}\} := \{EP_0, EP_1, EP_2, EP'_2, EP_3, EP_{0\text{ dup}}, EP'_3, EP'_{0\text{ dup}}\}, \\ A := \{EP'_3, (d, x'_3, t'_3), EP'_{0\text{ dup}}\};$$

A partir de EP'_3 , on franchit t'_4 et on atteint EP_4 :

$$t'_4 \in T_{cri} \text{ alors etiquette} := C, V'_{m'_3} := V_{m'_3} - \{t'_4\} := \{t_{obs}\}, \\ m_4 = (p_4, p_{obs}), V_{m_4} := \{t_6\}; \\ h_4 = (\$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, x_6, \$)^T; \\ \text{etiq} := \emptyset \text{ alors}$$

$$B_4 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 31 < x_1 \leq 50 \\ 0 \leq x'_2 \leq 0 \\ 0 \leq x'_3 \leq 0 \\ 0 \leq x_4 \leq 60 \\ 0 \leq x'_2 + x'_3 \leq 60 \end{array} \right\};$$

$$\mathcal{L}(t'_4) = f, \text{ alors panne} := \text{true}; \\ S := S \cup \{EP_4\} := \{EP_0, EP_1, EP_2, EP'_2, EP_3, EP_{0\text{ dup}}, EP'_3, EP'_{0\text{ dup}}, EP_4\}; \\ A := \{EP'_3, (f, x'_3, t'_4), EP_4\};$$

On choisit EP'_0 :

$$Q := Q - \{EP'_0\} := \emptyset, m'_0 = (p'_0, p_{obs}), \\ V'_{m'_0} := V_{m'_0} := \{t'_0, t_{obs}\}, \text{etiq} := \text{etiquette, avec etiquette} := D,$$

A partir de EP'_0 , on franchit t'_0 et on atteint EP'_1 :

$t'_0 \in T_{deg}$ alors $etiquette := D$, $V'_{m'_3} := V'_{m'_3} - \{t'_0\} := \{t_{obs}\}$,

$m'_1 = (p'_1, p_{obs})$, $V_{m'_1} := \{t'_1, t'_5, t_{obs}\}$;

$h'_1 = (\$, \$, \$, \$, \$, \$, \$, \$, x'_1, \$, \$, \$, x'_1, \$, x'_0 + x'_1)^T$;

$etiq = etiquette$;

$$B'_1 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 70 \leq x_3 < 90 \\ 0 \leq x'_0 \leq 0 \\ 0 \leq x'_1 \leq 0 \\ 0 \leq x'_0 + x'_1 \leq 60 \end{array} \right\} ;$$

$\mathcal{L}(t'_0) = a$, alors $panne := false$;

$A := \{EP'_0, (a, x'_0, t'_0), EP'_1\}$, $Q = \{EP'_1\}$;

On choisit EP'_1 :

$Q := Q - \{EP'_1\} := \emptyset$, $S := \{EP_0, EP_1, EP_2, EP'_2, EP_3, EP_{0\ dup}, EP'_3, EP'_{0, dup}, EP_4, EP'_1\}$,

$m'_1 = (p'_1, p_{obs})$,

$V'_{m'_1} := V_{m'_1} := \{t'_1, t'_5, t_{obs}\}$, or $t'_1 \in T_{deg}$ et $t'_5 \in T_{cri}$ alors $etiq := \emptyset$,

A partir de EP'_1 , on franchit t'_1 et on atteint $EP'_{2\ dup}$:

$t'_1 \in T_{deg}$ alors $etiquette := D$, $V'_{m'_1} := V'_{m'_1} - \{t'_1\} := \{t'_5, t_{obs}\}$,

$m'_{2\ dup} = m'_2 = (p'_2, p_{obs})$; "c'est un marquage existant, Q ne sera pas étendu".

$V_{m'_{2\ dup}} := \{t'_2, t_{obs}\}$;

$h'_{2\ dup} = (\$, \$, \$, \$, \$, \$, \$, \$, x'_{2\ dup}, \$, \$, \$, \$, x'_0 + x'_1 + x'_{2\ dup})^T$;

$etiq := \emptyset$ alors

$$B'_{2\ dup} = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 70 \leq x_3 < 90 \\ 0 \leq x'_0 \leq 0 \\ 31 \leq x'_1 \leq 40 \\ 0 \leq x'_{2\ dup} \leq 0 \\ 0 \leq x'_0 + x'_1 \leq 60 \\ 0 \leq x'_0 + x'_1 + x'_{2\ dup} \leq 60 \end{array} \right\} ;$$

$\mathcal{L}(t'_1) = c$, alors $panne := false$;

$S = \{EP_0, EP_1, EP_2, EP'_2, EP_3, EP_{0\ dup}, EP'_3, EP'_{0, dup}, EP_4, EP'_1, EP'_{2\ dup}\}$,

$A := \{EP'_1, (c, x'_1, t'_1), EP'_{2\ dup}\}$;

A partir de EP'_1 , on franchit t'_5 et on atteint $EP_{4\ dup}$:

$t'_5 \in T_{cri}$ alors $etiquette := C$, $V'_{m'_1} := V'_{m'_1} - \{t'_5\} := \{t_{obs}\}$,

$m_{4\ dup} = (p_4, p_{obs})$, $V_{m_{4\ dup}} := \{t_6\}$;

$h_{4\ dup} = (\$, \$, \$, \$, \$, \$, \$, \$, x_{4\ dup}, \$, \$, \$, \$, \$)^T$;

$etiq \neq etiquette$;

$$B_{4 \text{ dup}} = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 70 \leq x_3 < 90 \\ 0 \leq x'_0 \leq 0 \\ 0 \leq x'_1 \leq 0 \\ 0 \leq x_{4 \text{ dup}} \leq 60 \\ 0 \leq x'_0 + x'_1 \leq 60 \end{array} \right\};$$

$\mathcal{L}(t'_5) = f$, alors $\text{spanne} := \text{true}$;

$S = S \cup \{EP_{4 \text{ dup}}\}$,

$S = \{EP_0, EP_1, EP_2, EP'_2, EP_3, EP_{0 \text{ dup}}, EP'_3, EP'_{0 \text{ dup}}, EP_4, EP'_1, EP'_{2 \text{ dup}}, EP_{4 \text{ dup}}\}$,

$A := \{EP'_1, (f, x'_1, t'_5), EP_{4 \text{ dup}}\}$;

$Q = \emptyset$, l'algorithme s'arrête, nous obtenons alors le pronostiqueur qui est représenté dans la figure 4.7 qui permet de calculer la date au plus tôt, où la date d'occurrence de l'événement f est égale à 31 UT s'il y a franchissement de la transition t_5 et 80 UT s'il y a franchissement de la transition t_4 .

La deuxième approche génère à partir du MMF, le graphe d'accessibilité logique et identifie l'ensemble des séquences qui se terminent avec un événement de défaillance. La paramétrisation ne concernera que les séquences dites défaillantes. Selon le graphe d'accessibilité, il existe deux séquences d'événements $s_1 = a \ c \ b \ d \ a \ f$ et $s_2 = a \ c \ b \ f$.

Soit le modèle de la figure 4.2. En appliquant l'algorithme 2 on obtient :

Pour la séquence d'événements $s_1 = a \ c \ b \ d \ a \ f$, on a $\sigma_1 = t_0 t_1 t_2 t_4 t'_0 t'_5$.

Au début

$m_0 = (p_0)$, $V_{m_0} = \{t_0\}$;

$h_0 = (x_0, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T$,

etiquette = N,

$B_0 = \{0 \leq x_0 \leq 0\}$;

On choisit EP_0 :

A partir de EP_0 , on franchit t_0 et on atteint EP_1 ,

$t_0 \in T_N$ alors etiquette := N,

$m_1 = (p_1)$, $V_{m_1} := \{t_1, t_5\}$;

$\mathcal{L}(t_0) = a$,

$h_1 = (\$, x_1, \$, \$, \$, x_1, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T$;

$B_1 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 0 \leq x_1 \leq 30 \end{array} \right\}$;

On choisit EP_1 :

A partir de EP_1 , on franchit t_1 et on atteint EP_2 :

$t_1 \in T_N$ alors etiquette := N,

$m_2 = (p_2)$, $V_{m_2} := \{t_2\}$;

$\mathcal{L}(t_1) = c$, $h_2 = (\$, \$, x_2, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T$;

$$B_2 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \end{array} \right\};$$

On choisit EP_2 :

A partir de EP_2 , on franchit t_2 et on atteint EP_3 :

$$\begin{aligned} t_2 &\in T_N \text{ alors } \text{etiquette} := N, \\ m_3 &= (p_3), V_{m_3} := \{t_3, t_4\}; \\ \mathcal{L}(t_2) &= b, \\ h_3 &= (\$, \$, \$, x_3, x_3, \$, \$, \$, \$, \$, \$, \$, \$)^T; \\ B_3 &= \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 0 \leq x_3 < 90 \end{array} \right\}; \end{aligned}$$

On choisit EP_3 :

A partir de EP_3 , on franchit t_4 et on atteint EP'_0 :

$$\begin{aligned} t_4 &\in T_{deg} \text{ alors } \text{etiquette} := D, \\ m'_0 &= (p'_0, p_{obs}), V_{m'_0} := \{t'_0, t_{obs}\}; \\ \mathcal{L}(t_4) &= d, h'_0 = (\$, \$, \$, \$, \$, \$, \$, \$, x'_0, \$, \$, \$, x'_0)^T; \\ B'_0 &= \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 70 \leq x_3 < 90 \\ 0 \leq x'_0 \leq 0 \end{array} \right\}; \end{aligned}$$

On choisit EP'_0 :

A partir de EP'_0 , on franchit t'_0 et on atteint EP'_1 :

$$\begin{aligned} t'_0 &\in T_{deg} \text{ alors } \text{etiquette} := D, \\ m'_1 &= (p'_1, p_{obs}), V_{m'_1} := \{t'_1, t'_5, t_{obs}\}; \\ \mathcal{L}(t'_0) &= a, \\ h'_1 &= (\$, \$, \$, \$, \$, \$, \$, x'_1, \$, \$, \$, x'_1, \$, x'_0 + x'_1)^T; \\ B'_1 &= \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 70 \leq x_3 < 90 \\ 0 \leq x'_0 \leq 0 \\ 0 \leq x'_1 \leq 0 \\ 0 \leq x'_0 + x'_1 \leq 60 \end{array} \right\}; \end{aligned}$$

On choisit EP'_1 :

A partir de EP'_1 , on franchit t'_5 et on atteint $EP_{4 \text{ dup}}$:

$$\begin{aligned} t'_5 &\in T_{cri} \text{ alors } \text{etiquette} := C, \\ m_{4 \text{ dup}} &= (p_4, p_{obs}), V_{m_{4 \text{ dup}}} := \{t_6\}; "m_{4 \text{ dup}} = m_4 \text{ mais la transition franchie } (t'_5) \text{ pour} \\ &\text{atteindre } EP_{4 \text{ dup}} \text{ est diff  rente de celle franchie } (t'_4) \text{ pour atteindre } EP_4 \\ \mathcal{L}(t'_5) &= f; \end{aligned}$$

$$h_{4 \text{ dup}} = (\$, \$, \$, \$, \$, \$, \$, \$, x_{4 \text{ dup}}, \$, \$, \$, \$, \$)^T ;$$

$$B_{4 \text{ dup}} = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 10 \leq x_1 \leq 30 \\ 0 \leq x_2 \leq 0 \\ 70 \leq x_3 < 90 \\ 0 \leq x'_0 \leq 0 \\ 0 \leq x'_1 \leq 0 \\ 0 \leq x_{4 \text{ dup}} \leq 60 \\ 0 \leq x'_0 + x'_1 \leq 60 \end{array} \right\} ;$$

On s'arrête car $\mathcal{L}(t'_5) = f$ un événement de défaillance. La date d'occurrence de l'événement f est égale à 80 UT s'il y a franchissement de la transition t_4 .

Pour la séquence d'événements $s_2 = a \ c \ b \ f$, on a $\sigma_2 = t_0 t_5 t'_2 t'_4$.

Au début

$$m_0 = (p_0), V_{m_0} = \{t_0\};$$

$$h_0 = (x_0, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$)^T,$$

$$\text{etiquette} = N,$$

$$B_0 = \{0 \leq x_0 \leq 0\};$$

On choisit EP_0 :

A partir de EP_0 , on franchit t_0 et on atteint EP_1 ,

$$t_0 \in T_N \text{ alors } \text{etiquette} := N,$$

$$m_1 = (p_1), V_{m_1} := \{t_1, t_5\};$$

$$\mathcal{L}(t_0) = a,$$

$$h_1 = (\$, x_1, \$, \$, \$, x_1, \$, \$, \$, \$, \$, \$)^T ;$$

$$B_1 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 0 \leq x_1 \leq 30 \end{array} \right\} ;$$

On choisit EP_1 :

A partir de EP_1 , on franchit t_5 et on atteint EP'_2 :

$$t_5 \in T_{deg} \text{ alors } \text{etiquette} := D,$$

$$m'_2 = (p'_2, p_{obs}), V_{m'_2} := \{t'_2, t_{obs}\};$$

$$\mathcal{L}(t_5) = c,$$

$$h'_2 = (\$, \$, \$, \$, \$, \$, \$, x'_2, \$, \$, \$, x'_2)^T ;$$

$$B'_2 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 31 < x_1 \leq 50 \\ 0 \leq x'_2 \leq 0 \end{array} \right\} ;$$

On choisit EP'_2 :

A partir de EP'_2 , on franchit t'_2 et on atteint EP'_3 :

$$t'_2 \in T_{deg} \text{ alors } \text{etiquette} := D,$$

$$m'_3 = (p'_3, p_{obs}), V_{m'_3} := \{t'_3, t'_4, t_{obs}\};$$

$$\mathcal{L}(t'_2) = d;$$

$$h'_3 = (\$, \$, \$, \$, \$, \$, \$, x'_3, x'_3, \$, \$, x'_2 + x'_3)^T ;$$

$$B'_3 = \left\{ \begin{array}{l} 0 \leq x_0 \leq 0 \\ 31 < x_1 \leq 50 \\ 0 \leq x'_2 \leq 0 \\ 0 \leq x'_3 \leq 0 \\ 0 \leq x'_2 + x'_3 \leq 91 \end{array} \right\};$$

Le franchissement de la transition t_{obs} ne sera pas traité, vu qu'il n'affecte pas le fonctionnement du système.

On choisit EP'_3 :

A partir de EP'_3 , on franchit t'_4 et on atteint EP_4 :

$t'_4 \in T_{cri}$ alors $etiquette := C$,

$$m_4 = (p_4, p_{\text{obs}}), V_{m_4} := \{t_6\};$$
$$\mathcal{L}(t'_4) = f;$$
$$h_4 = (\$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, \$, x_6, \$)^T ;$$

$$B_4 = \left\{ \begin{array}{c} 0 \leq x_0 \leq 0 \\ 31 < x_1 \leq 50 \\ 0 \leq x'_2 \leq 0 \\ 0 \leq x'_3 \leq 0 \\ 0 \leq x_4 \leq 0 \\ 0 \leq x'_2 + x'_3 \leq 60 \end{array} \right\};$$

On s'arrête car $\mathcal{L}(t'_4) = f$ est un événement de défaillance. La date d'occurrence de l'événement f est égale à 31 UT s'il y a franchissement de la transition t_5 .

Le nouveau pronostiqueur ne prend en considération que les deux séquences. Le nouveau pronostiqueur est alors réduit par rapport au précédent et garde le même système d'inéquations pour les sommets représentés (voir [figure 4.4](#)).

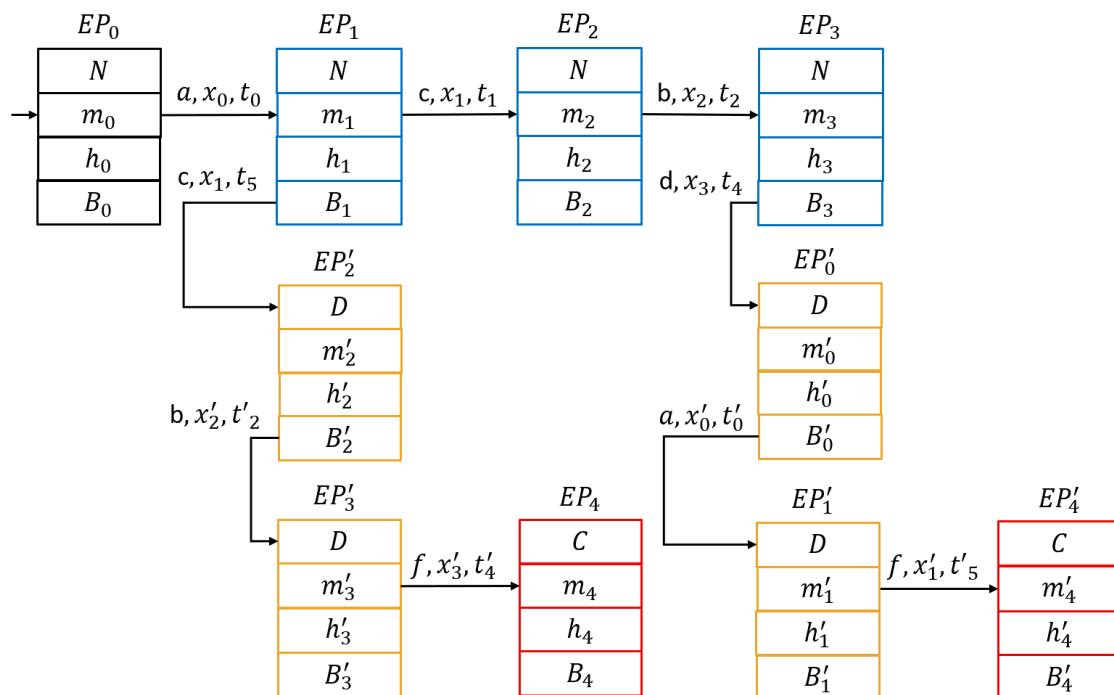


Figure 4.4: Modèle pronostiqueur d'une cellule de Batterie électrique (Approche 2).

4.3.3 Propriété de pronosticabilité

La propriété de pronosticabilité comme évoqué dans le chapitre 3, vérifie la possibilité de pronostiquer la date d'occurrence d'un événement de défaillance du MMF. Rappelons que cette propriété se base sur l'ensemble des séquences du MMF (défaillante et non défaillante) pour juger la pronosticabilité ou non d'un événement de défaillance. Son utilisation dans le cas de notre benchmark, nous permet de vérifier si les séquences qui conduisent le système à l'état de panne sont pronosticables. Un événement de défaillance non pronosticable, présente un risque pour le système. Dans ce cas, il faut reprendre le MMF pour qu'il prenne en considération l'ambiguïté détectée lors de la vérification de la propriété de pronosticabilité.

Si on prend en considération le modèle pronostiqueur de la figure 4.4, selon la définition 3.9 de la section 3.2.6, on prend :

- 1) Pour $\sigma_1(x) = x_0 t_0 x_1 t_5 x_2' t_2' x_4 t_4'$;
 $\sigma_1 = t_0 t_5 t_2' t_4'$;
avec $t_4' \in T_{cri}$,
 $\overline{\sigma}_1 = \{t_0, t_0 t_5, t_0 t_5 t_2', t_0 t_5 t_2' t_4'\}$;
 $\beta = t_0$; $t_4' \notin \beta$;
- 2) $\theta(x) = x_0 t_0$; $\theta = t_0$; $\mathcal{L}(\theta) = \mathcal{L}(\beta) = a$;
On prend $\theta(x)\gamma(x) = x_0 t_0$ avec $\gamma(x) = x_1 t_5 x_2' t_2' x_4 t_4'$ et $\text{delay}(\gamma) = x_1 + x_2' + x_4$ avec :
 $31 \leq x_1 \leq 50, 0 \leq x_2' \leq 0, 0 \leq x_3' \leq 0, 0 \leq x_4 \leq 0, 0 \leq x_2' + x_3' \leq 91$;

R_{TL} est pronosticable, l'occurrence de l'événement de défaillance f est prédite 31 unités de temps à l'avance après le franchissement de la transition t_5 .

- 1) Pour $\sigma_2(x) = x_0 t_0 x_1 t_1 x_2 t_2 x_3 t_4 x_0' t_0' x_1' t_5'$;
 $\sigma_2 = t_0 t_1 t_2 t_4 t_0' t_5'$; avec $t_5' \in T_{cri}$,
 $\overline{\sigma}_2 = \{\varepsilon, t_0, t_0 t_1, t_0 t_1 t_2, t_0 t_1 t_2 t_4, t_0 t_1 t_2 t_4 t_0', t_0 t_1 t_2 t_4 t_0' t_5'\}$;
 $\beta = t_0 t_1 t_2$; $t_5' \notin \beta$;
- 2) $\theta(x) = x_0 t_0 x_1 t_1 x_2 t_2$; $\theta = t_0 t_1 t_2$; $\mathcal{L}(\theta) = \mathcal{L}(\beta) = a c b$;
On prend $\theta(x)\gamma(x) = x_0 t_0 x_1 t_1 x_2 t_2 x_3 t_4 x_0' t_0' x_1' t_5'$ avec $\gamma(x) = x_3 t_4 x_0' t_0' x_1' t_5'$ et $\text{delay}(\gamma) = x_3 x_0' x_1'$ avec $80 \leq x_3 \leq 89, 0 \leq x_0' \leq 0, 0 \leq x_1' \leq 0$ et $0 \leq x_0' + x_1' \leq 91$.

R_{TL} est pronosticable, l'occurrence de l'événement de défaillance f est prédite 80 unités de temps à l'avance après le franchissement de la transition t_4 .

La prise en considération des cycles dans un MMF, donne une estimation sur les exécutions possible des modes de fonctionnement du système. La figure 4.5, est un exemple d'un MMF d'une cellule avec une place p_5 marquée par 2000, ce nombre de jetons désigne le nombre de charges et de décharges possibles de la cellule. Dans un cycle d'exécution, qu'il soit en mode nominal ou dégradé, existe un événement de charge c et un événement de décharge d . A chaque occurrence d'un événement c ou d , la place p_5 perd un jeton. Ce qui nous donne, un nombre de cycles d'exécution égale à 1000 cycles.

Avec l'introduction du nombre de cycles dans le pronostic, il ne serait plus nécessaire de détecter un changement du mode de fonctionnement du système (passage d'un mode nominal en mode dégradé) pour faire le pronostic. Il serait possible de pronostiquer la date d'occurrence d'un événement de défaillance, à partir de n'importe quel marquage du MMF (y compris le nominal). Ainsi que de déterminer un horizon au pronostic, à partir duquel la panne est certaine i.e. déterminer la date au plus tard de l'occurrence d'un événement de défaillance.

Cependant, pour pouvoir faire ce pronostic, il serait nécessaire de construire un pronostiqueur intégrant ce nombre de cycles d'exécution. Dans notre démarche de pronostic, nous n'avons pas pris en considération ces cycles, vu que l'intérêt est de pronostiquer la date d'occurrence d'un événement de défaillance au plus tôt. Le pronostiqueur que nous avons construit est généré à partir de l'arbre d'accessibilité du MMF. En revanche, un arbre d'accessibilité ne peut représenter des cycles, d'où la nécessité de construire un pronostiqueur à partir du graphe d'accessibilité du MMF.

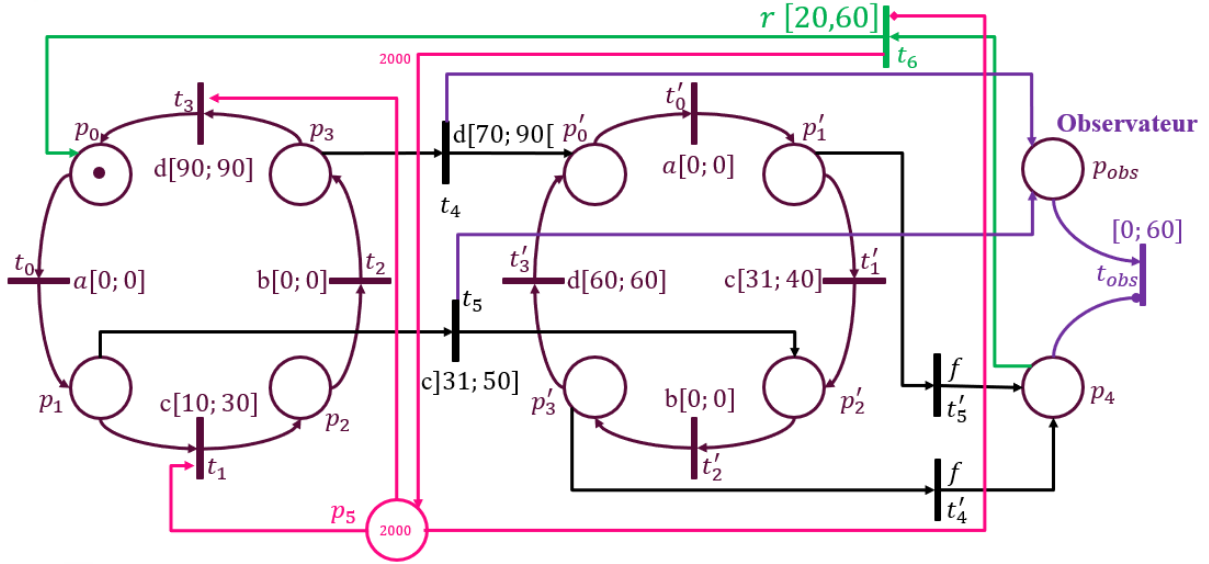


Figure 4.5: MMF d'une cellule avec cycles d'exécution.

L'exploitation de l'algorithme 3 ne sera possible pour générer les sommets du pronostiqueur. La figure 4.6 représenterait le pronostiqueur équivalent du MMF de la figure 4.5. NC représente le nombre de cycles possible. Au début il faut initialiser le n par NC . Ce n est décrémenté à chaque évolution d'un sommet vers un autre par occurrence d'un événement c ou d .

Soit $\sigma_{deg} = t'_0 t'_3 t'_2 t'_1$, une séquence dégradée du MMF de la figure 4.5 et sa séquence paramétrique $\sigma_{deg}(x) = x'_0 t'_0 x'_3 t'_3 x'_2 t'_2 x'_1 t'_1$. A chaque exécution de la séquence σ_{deg} , deux jetons sont retirés de la place p_5 . Pour calculer la date au plus tard de l'occurrence de f , il faut introduire les cycles d'exécution possibles du mode dégradé dans le pronostiqueur. Le nombre de cycle restant après l'évolution dans un mode de fonctionnement dégradé sera égale à $NR = (NC - n)/2$; avec NC le nombre de cycle initial et n le nombre de cycle actuelle. Sur la base des résultats du système d'inéquations de l'algorithme 3 la durée d'exécution minimale de la séquence σ_{deg} est $delay(\sigma_{deg}) = x'_0 x'_3 x'_2 x'_1$. Une durée qui correspond à la date au plus tôt de l'occurrence de f . En conséquence et avec la prise en considération des cycles, la date au plus tard de l'occurrence de f sera $Dmax = NR * delay(\sigma_{deg})$. Ainsi, il serait possible de pronostiquer à l'avance la date au plus tôt et au plus tard d'un événement de défaillance.

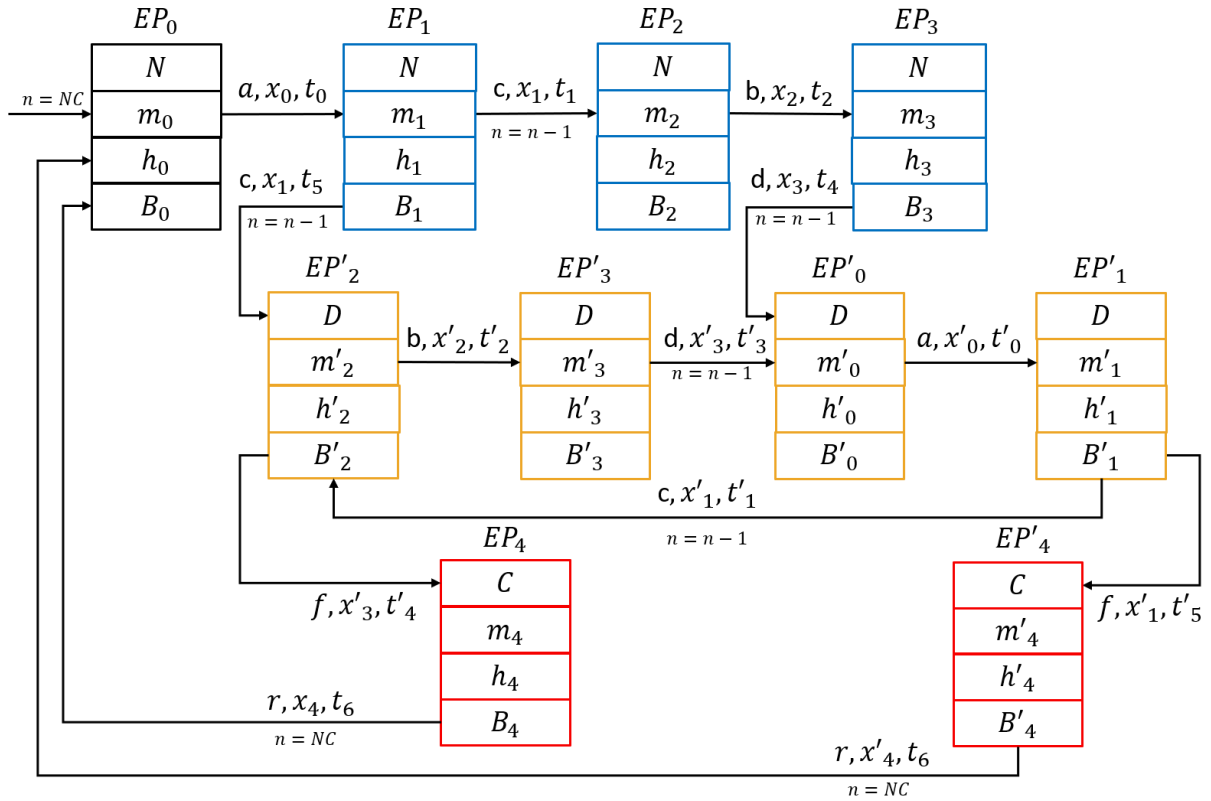


Figure 4.6: Pronostiqueur du MMF avec cycles d'exécution.

4.4. Simulation

Avec l'outil de simulation INA ([annexe 2](#)), nous avons généré le graphe d'accessibilité du MMF. La première étape de la simulation consiste à configurer l'environnement de développement pour intégrer les contraintes temporelles lors de la modélisation. Nous rappelons que la modélisation sur INA s'effectue d'une façon textuelle et non pas graphique. La génération du MMF sera lors sous le format (num_place nbr_jeton post,pré). Ainsi, pour la place p_0 par exemple nous écrivons : 0 1 3 6,0. Ce qui signifie que la P_0 a 1 jeton et comme post conditions (t_3, t_6) et comme pré condition t_0 . Le programme correspondant au MMF [figure 4.2](#) sera alors :

```
P M PRE,POST
0 1 3 6,0
1 0 0,1 5
2 0 1,2
3 0 2,3 4
4 0 11 12,13
5 0 4 10,7
6 0 7,8 12
7 0 5 8,9
8 0 9,10 11
```

Avec la génération de l'arbre d'accessibilité nous avons obtenu 117915 états. Un nombre important d'états pour le MMF d'une seule cellule, ce qui nous donne une idée sur la complexité d'analyse temporelle d'un système (ensemble de cellules). Lors de la génération du pronostiqueur, nous avons choisi de ne pas faire évoluer les sommets redondants, vu que nous n'avons pas pris en considération les cycles d'exécution des modes. Ce qui a réduit

considérablement le nombre d'états et par conséquent la complexité d'analyse du MMF. L'outil INA n'est pas adapté à 100 % à l'analyse de notre modèle. Nous avons prévu dans le cadre de la continuité de nos travaux de thèse, de développer un outil de simulation adéquat. INA nous a permis en plus de la génération de l'arbre d'accessibilité du MMF, d'identifier l'ensemble des séquences possibles et de déterminer celle avec une exécution minimale et maximale. Il serait intéressant d'introduire un outil (ou étendre INA) afin d'identifier l'ensemble des séquences qui amènent à un événement de défaillance et de déterminer la durée d'exécution max et min de ces séquences.

4.5. Conclusion

Le présent chapitre est un applicatif de l'approche proposée du pronostic. Nous l'avons commencé par une description du benchmark "Cellule d'une batterie", donnant ainsi une idée sur le mode de fonctionnement d'une cellule et les différentes sources de défaillances possibles. Chose qui nous a permis, de modéliser le comportement de la cellule selon un modèle de modes de fonctionnement (MMF) basé sur les réseaux de Petri temporels labellisés. Etant donné que notre intérêt est de pronostiquer au plus tôt la date d'occurrence d'un événement de défaillance, nous nous sommes basés sur les approches développées dans le chapitre 3 pour analyser le MMF. Pour effectuer cette analyse nous avons généré un modèle pronostiqueur à partir de l'arbre d'accessibilité du MMF. Un pronostiqueur qui nous a permis d'analyser à la fois le comportement logique et temporel du système. Généralement les RdP temporelle sont source d'explosion d'états ce qui rend leur analyse complexe. Nous avons alors, proposé une paramétrisation des états dans le modèle du pronostiqueur et une limitation d'évolution pour les sommets redondants. Ce qui a réduit considérablement l'espace d'états et par conséquent l'analyse temporelle et logique du MMF. Pour déterminer la possibilité ou non de pronostiquer la date d'occurrence d'un événement, nous avons vérifié la propriété de pronosticabilité, deux notions ont été évoquées : le marquage indicateur et le marquage limite. Actuellement il n'existe pas un outil de simulation adapté à 100% à notre approche. Cependant, Nous avons choisi d'utiliser l'outil INA pour pouvoir générer le graphe d'accessibilité et déterminer la (ou les) séquence(s) avec une exécution minimale dans le modèle.

CHAPITRE 5 : Conclusion & perspectives

Les travaux proposés dans ce mémoire de thèse, s'intéressent à l'analyse comportementale du système pour déterminer à l'avance la date d'occurrence d'un événement de défaillance future. Ce type d'événement provoque des arrêts (arrêt partiel ou total) avec des conséquences souvent désastreuses et coûteuses. Des applications de supervision de type SCADA ont été développées pour aider les ingénieurs de contrôle à déterminer la causalité des défaillances et établir le plan d'intervention curatif et préventif. La communauté scientifique des systèmes à événements discrets (SED), a proposé des approches réduisant l'impact des risques pouvant affecter le bon fonctionnement du système. Les relations de causes à effets entre les états de ce type de système, relèvent une dynamique avec un espace d'états discret, vu que l'évolution entre les états se fait sur l'occurrence d'un événement. Une dynamique cohérente avec l'évolution des SED qui est conditionnée par l'occurrence d'événements discrets.

Nous nous sommes basés dans notre étude sur un système soumis à des événements de défaillance. Notre analyse du comportement du système est à la fois logique (ordre d'apparition) et temporelle (date d'apparition), pour pouvoir déterminer la date d'occurrence d'un événement de défaillance. Sachant que l'analyse temporelle d'un modèle est confrontée généralement à un espace d'états important, nous avons proposé pour circonscrire cet espace d'états, une modélisation du système limitée à trois modes de fonctionnements (nominal, dégradé et critique), que nous avons nommés modèle de modes de fonctionnement (MMF). Ce MMF est basé sur les réseaux de Petri temporels labellisés (RdPTL), où les transitions sont conditionnées par des événements (labels) et leurs dates d'occurrences respectives. Nous avons généré à partir de l'arbre d'accessibilité du MMF un modèle pronostiqueur, qui offre une représentation de l'ensemble des séquences du MMF d'une façon plus expressive afin d'identifier facilement celles qui se terminent avec un événement de défaillance.

Toujours dans un souci d'une explosion de l'espace d'états, dû au fait que le franchissement d'une transition temporelle dans un RdPTL est possible à tout instant dans son intervalle de tir, nous avons proposé une paramétrisation des états du pronostiqueur i.e. l'introduction d'une horloge et un système d'inéquations des horloges pour chaque état du système. Les états obtenus de la discrétisation du temps sont alors regroupés dans un seul état (appelé état paramétrique) et le système d'inéquations déterminera les valeurs des horloges. Nous avons proposé deux approches pour la génération du pronostiqueur applicables selon la complexité (espace d'états et difficulté de construction de l'arbre d'accessibilité) du système :

La première approche introduit les variables d'horloges dans la fonction de transition du pronostiqueur et calcule pour chaque sommet du pronostiqueur un système d'inéquations correspondant.

La deuxième approche considère un pronostiqueur basé sur le graphe d'accessibilité logique, à partir duquel nous déterminons la ou les séquences d'événements qui se terminent par un événement de défaillance. A partir de ces seules séquences nous introduisons l'aspect temporel et déduisons le système d'inéquations temporelles, associé.

Le choix de l'une ou l'autre dépend de la complexité du système modélisé (taille du MMF, cycles d'exécution des modes). La première option est plus expressive (puisqu'elle calcule le système d'inéquations pour l'ensemble des sommets du pronostiqueur), alors que la seconde approche optimise le temps de calcul du système d'inéquations en ne s'intéressant qu'aux séquences d'événements pronosticables, ce qui favorise son utilisation pour les modèles complexes.

Etant conscients que le pronostic d'un événement de défaillance, ne peut être toujours assuré, nous avons alors établi la propriété de pronosticabilité, pour discerner les séquences qui sont pronosticables (possibilité de pronostiquer l'événement de défaillance) de celles qui ne le sont pas. Généralement, l'ambiguïté ou l'incertitude dans le pronostic est due au manque d'observations sur le système, pour des raisons économiques ou technologiques. Pour pallier ce problème, il faut donc introduire dans le MMF un label temporel distinctif i.e. en ayant des intervalles différents. A partir des séquences pronosticables et en se basant sur le système d'inéquations du pronostiqueur, nous calculons la durée d'exécution minimale de chaque séquence pronosticable et nous déterminons celle avec une durée d'exécution minimale (DEM). Cette durée représente la date au plus tôt d'occurrence de l'événement de défaillance après le franchissement de la première transition de la séquence avec DEM.

Nous avons choisi comme benchmark pour vérifier notre approche de pronostic, la cellule d'une batterie de voiture électrique. La composition série parallèle des cellules permet d'avoir un dimensionnement énergétique de la batterie. Le calcul de la durée de vie d'une cellule dans cette composition série parallèle a un impact sur l'exploitation de la batterie et par ailleurs, l'utilisation des systèmes de pilotage de composition série parallèle de la batterie. La durée de vie d'une cellule dépend de ses cycles de charge et décharge, ce qui augmente d'avantage la complexité de déterminer la date d'occurrence d'une défaillance au plus tôt et au plus tard. Certes avec la technique de chien de garde il serait possible de signaler tout dépassement anormal dans la durée de charge ou de décharge. Néanmoins la complexité de calcul de ces dates augmente sur des modèles avec des cycles d'exécution, comme il est présenté dans le benchmark. D'où l'intérêt de l'utilisation d'un système d'inéquations qui permet de déterminer les valeurs minimales et maximales des horloges du système pour chaque cycle d'exécution. Le MMF de la cellule ainsi que son pronostiqueur ont été construits à base d'un RdPTL. De part une modélisation formelle, on aboutit à un système d'inéquations permettant de décider de la pronosticabilité du système avec comme hypothèse (modèle RdP sauf, une seule défaillance et indépendante (une défaillance n'entraîne pas l'autre), l'ensemble des événements sont observables, pas de cycles d'exécution des modes). Le système d'inéquations a permis de déterminer les dates minimales d'occurrence des événements. Après vérification de la pronosticabilité des séquences, nous avons calculé la date au plus tôt de l'occurrence d'un événement de défaillance susceptible de se produire dans le futur. Avec l'outil de simulation INA, nous avons généré le graphe d'accessibilité du MMF. L'intérêt est de montrer que l'espace d'états du pronostiqueur proposé dans notre approche est compact par rapport au graphe d'accessibilité généré par INA. Le simulateur a permis aussi de déterminer la séquence d'exécution minimale en se basant sur le système d'inéquations de la séquence pronosticable.

Lors du développement du pronostic, plusieurs hypothèses ont été posées. Nous avons supposé que sur le RdPTL une seule défaillance est possible alors que dans le cas réel on peut avoir plusieurs événements de défaillance qui ne sont pas forcément observables. Ces événements de défaillance sont supposés indépendants dans notre approche i.e. un événement de défaillance n'entraîne pas l'autre. Alors qu'un événement de défaillance f_1 par exemple pourra être suivi par un autre événement de défaillance f_2 . Dans ce cas il ne serait pas possible de distinguer f_1 de f_2 . Pour surmonter cette hypothèse d'indépendance, il est nécessaire de revoir le formalisme la propriété de pronosticabilité. Cependant, au-delà de ces hypothèses, il y a des limites que nous n'avons pas pu traiter durant nos travaux. Elles seront reprises comme perspectives de travaux futurs.

Nous avons développé une méthode de pronostic pour un système mono-composant afin de simplifier la génération du modèle des modes de fonctionnement et du modèle pronostiqueur. L'intérêt est d'étendre l'approche sur des systèmes multi-composant de comportement identique ou non. Sans la paramétrisation des états, la génération d'un graphe d'accessibilité temporel serait compliquée. Cependant, ce problème n'est pas résolu dans son intégralité, i.e. pour des architectures plus complexes lors de la génération du modèle des modes (espace d'états important qui rend difficile la modélisation du système et par conséquent la génération du pronostiqueur), la construction de l'arbre d'accessibilité et les outils de simulations actuels nécessitent un temps important. Envisager une modélisation modulaire ou décentralisée peut réduire l'explosion combinatoire du système temporel. Il y a déjà des travaux récents dans ce sens pour le pronostic des systèmes complexes avec des méthodes stochastiques [102], [107], [121]. Le système d'inéquations proposé, peut être exploité comme solution pour les systèmes stochastiques, dans lesquels les probabilités des dates au plus tôt et au plus tard, peuvent être confrontées à des conflits. Une modélisation basée sur des réseaux bayésiens est bien adaptée à ce type de systèmes. En exploitant les probabilités conditionnelles, il serait alors possible de déterminer les probabilités des dates d'occurrence d'un événement de défaillance au plus tôt et au plus tard. L'approche peut être exploitée aussi sur des RdP pour des systèmes multi-composant avec une modélisation basée sur des RdP temporels labellisés colorés. Un formalisme serait alors nécessaire pour définir le MMF et générer le pronostiqueur associé. Avec le second modèle du pronostiqueur, la propriété de pronosticabilité ne s'applique pas car nous avons choisis uniquement les séquences fautives, déterminer une ambiguïté avec d'autres séquences saines n'est pas possible.

Parmi les perspectives proposées :

Introduire les cycles d'exécution des modes dans l'algorithme du pronostic ; les algorithmes proposés dans ce mémoire, ne prennent pas en considération les cycles d'exécution des modes. L'arborescence dans le pronostiqueur est limitée par des sommets avec des marquages identiques aux existants. Cependant, pour un modèle pronostiqueur avec des cycles d'exécution, l'arborescence des sommets doit prendre en considération le nombre de cycles possibles. Une telle représentation permettrait un pronostic de la date au plus tard d'occurrence d'un événement de défaillance. Après cette date la panne est certaine.

La formalisation d'un MMF multi-composant basée sur les RdPTL colorés, où chaque composant est représenté par un jeton coloré. Le modèle pronostiqueur généré à partir de ce MMF doit distinguer l'évolution des jetons afin d'analyser le comportement de chaque composant. Avec la vérification de la propriété de pronosticabilité il serait possible de déterminer les séquences pronosticables. Toute évolution d'un jeton dans l'une des séquences dites pronosticables fera signe d'une occurrence future d'un événement de défaillance. A partir des résultats du système d'inéquations, les dates d'occurrence au plus tôt et au plus tard seront déterminées.

Exploiter le système d'inéquations dans des systèmes stochastiques. Le modèle pronostiqueur proposé dans ce mémoire, permet une analyse à la fois temporelle et logique de l'occurrence des événements. Formaliser un modèle stochastique temporel qui prend en considération des probabilités conditionnelles, permettra de vérifier la certitude d'occurrence d'un événement dans une date au plus tôt et ou au plus tard.

Bibliographie

- [1] C. G. Cassandras et S. Lafortune, « Discrete event systems: The state of the art and new directions », in *Applied and computational control, signals, and circuits*, Springer, 1999, p. 1–65.
- [2] A. Pagès et M. Gondran, *Fiabilité des systèmes*, vol. 39. Eyrolles, 1980.
- [3] M. R. Zemouri, « Contribution à la surveillance des systèmes de production à l'aide des réseaux de neurones dynamique », *Appl. À E-Maintenance Thèse Dr. Univ. Franche-Comté*, 2003.
- [4] J. J. Gertler, M. Costin, X. Fang, R. Hira, Z. Kowalczyk, et Q. Luo, « Model-Based On-Board Detection and Diagnosis for Automotive Engines », in *Proceedings of IFAC/IMACS Symposium SAFEPROCESS 91*, 1991, p. 241–246.
- [5] M. Combacau, P. Berruet, E. Zamai, P. Charbonnaud, et A. Khatib, « Supervision and monitoring of production systems », *IFAC Proc. Vol.*, vol. 33, n° 17, p. 849–854, 2000.
- [6] P. Ribot, « Vers l'intégration diagnostic/pronostic pour la maintenance des systèmes complexes », phdthesis, Université Paul Sabatier - Toulouse III, 2009.
- [7] J. Zaytoon et S. Lafortune, « Overview of fault diagnosis methods for discrete event systems », *Annu. Rev. Control*, vol. 37, n° 2, p. 308–320, 2013.
- [8] F. Lin et W. M. Wonham, « On observability of discrete-event systems », *Inf. Sci.*, vol. 44, n° 3, p. 173–198, 1988.
- [9] R. Saddem, A. Toguyeni, et T. Moncef, « Algorithme d'interprétation d'une base de signatures temporelles causales pour le diagnostic en ligne des systèmes à événements discrets », Bordeaux, France, 2012.
- [10] A. K. A. Toguyeni, « Surveillance et diagnostic en ligne dans les ateliers flexibles de l'industrie manufacturière », PhD Thesis, Lille 1, 1992.
- [11] R. Ammour, « Contribution au diagnostic et pronostic des systèmes à événements discrets temporisés par réseaux de Petri stochastiques », phdthesis, Normandie Université, 2017.
- [12] A. Boufaied, « Contribution à la surveillance distribuée des systèmes à événements discrets complexes », PhD Thesis, Paul Sabatier - Toulouse III, 2003.
- [13] D. Gucik-Derigny, « Contribution au pronostic des systèmes à base de modèles: théorie et application », *L'Université Paul Cézanne Aix-Marseille III Dr. Diss.*, 2011.
- [14] R. Valette, J. Cardoso, et D. Dubois, « Monitoring manufacturing systems by means of Petri nets with imprecise markings », in *IEEE International Symposium on Intelligent Control*, 1989, vol. 2526.
- [15] J.-C. Laprie, « Dependability: Basic concepts and terminology », in *Dependability: Basic Concepts and Terminology*, Springer, 1992, p. 3–245.
- [16] M. Combacau, « Commande et surveillance des systèmes à événements discrets complexes: application aux ateliers flexibles », PhD Thesis, Toulouse 3, 1991.
- [17] K. Godary-Dejean, « LPT: Little Parametric Tool, outil pour la validation d'une borne temporelle paramétrée », 2008.
- [18] M. Sampath, « A discrete event systems approach to failure diagnosis », *PhD Univ. Mich. Mich. USA*, 1995.
- [19] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, et D. C. Teneketzis, « Failure diagnosis using discrete-event models », *IEEE Trans. Control Syst. Technol.*, vol. 4, n° 2, p. 105–124, 1996.
- [20] M. K. Shahzad, « Exploitation dynamique des données de production pour améliorer les méthodes DFM dans l'industrie Microélectronique », PhD Thesis, Grenoble, 2012.
- [21] D. Marquez, M. Neil, et N. Fenton, « Solving dynamic fault trees using a new hybrid Bayesian network inference algorithm », in *2008 16th Mediterranean Conference on Control and Automation*, 2008, p. 609–614.

- [22] J.-C. Laprie, B. Courtois, M.-C. Gaudel, et D. Powell, *Sûreté de fonctionnement des systèmes informatiques*. Dunod-Bordas, 1989.
- [23] J. C. Laprie, « Sûreté de fonctionnement et tolérance aux fautes: concepts de base », *Rapp. Tech. LAAS-88287 Lab. D'automatique D'analyse Syst. Toulouse*, 1988.
- [24] S. Hubac et E. Zamaï, *Politiques de maintenance équipement en flux de production stressant*, Techniques de l'ingénieur AGC. 2013.
- [25] S. Li et X. Li, « Study on generation of fault trees from Altarica models », *Procedia Eng.*, vol. 80, p. 140–152, 2014.
- [26] V. T. Tràn, « Machine fault diagnosis and condition prognosis using adaptive Neuro-Fuzzy inference system and classification and regression trees », PhD Thesis, Pukyong National University, 2009.
- [27] R. E. Neapolitan, *Contemporary artificial intelligence*. CRC press, 2012.
- [28] S. Khomfoi et L. M. Tolbert, « Fault diagnostic system for a multilevel inverter using a neural network », *IEEE Trans. Power Electron.*, vol. 22, n° 3, p. 1062–1069, 2007.
- [29] B. T. Pham, T.-A. Hoang, D.-M. Nguyen, et D. T. Bui, « Prediction of shear strength of soft soil using machine learning methods », *Catena*, vol. 166, p. 181–191, 2018.
- [30] K. Chabir, « Diagnostic de défauts des systèmes contrôlés via un réseau », PhD Thesis, Henri Poincaré - Nancy 1, 2011.
- [31] A. Philippot, P. Marangé, F. Gellot, J.-F. Pétin, et B. Riera, « Fault tolerant control for manufacturing discrete systems by filter and diagnoser interactions », 2014.
- [32] A. M. Ali, C. Join, et F. Hamelin, « A robust algebraic approach to fault diagnosis of uncertain linear systems », in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, 2011, p. 1557–1562.
- [33] D. Lefebvre, « On-line fault diagnosis with partially observed Petri nets », *IEEE Trans. Autom. Control*, vol. 59, n° 7, p. 1919–1924, 2013.
- [34] S. Genc et S. Lafortune, « Distributed diagnosis of place-bordered Petri nets », *IEEE Trans. Autom. Sci. Eng.*, vol. 4, n° 2, p. 206–219, 2007.
- [35] S. Genc et S. Lafortune, « Distributed diagnosis of discrete-event systems using Petri nets », in *International Conference on Application and Theory of Petri Nets*, 2003, p. 316–336.
- [36] D. T. Nguyen, Q. B. Duong, E. Zamaï, et M. K. Shahzad, « Bayesian network model with dynamic structure identification for real time diagnosis », in *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, 2014, p. 1–8.
- [37] X. Xu *et al.*, « Error diagnosis of cloud application operation using bayesian networks and online optimisation », in *2015 11th European Dependable Computing Conference (EDCC)*, 2015, p. 37–48.
- [38] S. Abdelwahed, G. Karsai, N. Mahadevan, et S. C. Ofsthun, « Practical implementation of diagnosis systems using timed failure propagation graph models », *IEEE Trans. Instrum. Meas.*, vol. 58, n° 2, p. 240–247, 2008.
- [39] J. Chen et R. Kumar, « Stochastic Failure Prognosability of Discrete Event Systems », *IEEE Trans. Autom. Control*, vol. 60, n° 6, p. 1570–1581, juin 2015, doi: 10.1109/TAC.2014.2381437.
- [40] S. Takai, « Robust prognosability for a set of partially observed discrete event systems », *Automatica*, vol. 51, p. 123–130, janv. 2015, doi: 10.1016/j.automatica.2014.10.104.
- [41] H. R. Berenji et Y. Wang, « Wavelet neural networks for fault diagnosis and prognosis », in *2006 IEEE International Conference on Fuzzy Systems*, 2006, p. 1334–1339.
- [42] L. V. Kirkland, T. Pombo, K. Nelson, et F. Berghout, « Avionics health management: Searching for the prognostics grail », in *2004 IEEE Aerospace Conference Proceedings (IEEE Cat. No. 04TH8720)*, 2004, vol. 5, p. 3448–3454.
- [43] S. Zabi, P. Ribot, et E. Chanthery, « Health monitoring and prognosis of hybrid systems », présenté à Annual Conference of the Prognostics and Health Management Society (PHM), Nouvelle Orléans, 2013, [En ligne]. Disponible sur: <https://hal.archives-ouvertes.fr/hal-01027538/document>.
- [44] R. Gouriveau et K. Medjaher, *Prognostics. Part: Industrial Prognostic-An Overview*. 2011.

- [45] M. Javed, A. Akhtar, I. Ahmed, et R. Faisal, « A Novel Method for Seizure Detection in Intracranial Eeg Recordings », in *2015 International Conference on Computational Intelligence and Communication Networks (CICN)*, 2015, p. 237–241.
- [46] M. J. Roemer, J. Dzakowic, R. F. Orsagh, C. S. Byington, et G. Vachtsevanos, « Validation and verification of prognostic and health management technologies », in *2005 IEEE Aerospace Conference*, 2005, p. 3941–3947.
- [47] A. Khoumsi et H. Chakib, « Multi-decision decentralized prognosis of failures in discrete event systems », in *2009 American Control Conference*, juin 2009, p. 4974-4981, doi: 10.1109/ACC.2009.5159960.
- [48] R. Kumar et S. Takai, « Decentralized prognosis of failures in discrete event systems », *IEEE Trans. Autom. Control*, vol. 55, n° 1, p. 48–59, 2009.
- [49] F. Basile, P. Chiacchio, et G. De Tommasi, « Fault diagnosis and prognosis in Petri Nets by using a single generalized marking estimation », *IFAC Proc. Vol.*, vol. 42, n° 8, p. 1396–1401, 2009.
- [50] A. Madalinski et V. Khomenko, « Predictability verification with parallel LTL-X model checking based on Petri net unfoldings », *IFAC Proc. Vol.*, vol. 45, n° 20, p. 1232–1237, 2012.
- [51] D. Lefebvre, « Fault diagnosis and prognosis with partially observed Petri nets », *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 44, n° 10, p. 1413–1424, 2014.
- [52] R. Ammour, E. Leclercq, E. Sanlaville, et D. Lefebvre, « Faults prognosis using partially observed stochastic Petri nets », in *2016 13th International Workshop on Discrete Event Systems (WODES)*, mai 2016, p. 472-477, doi: 10.1109/WODES.2016.7497890.
- [53] P. J. Ramadge et W. M. Wonham, « The control of discrete event systems », *Proc. IEEE*, vol. 77, n° 1, p. 81–98, 1989.
- [54] H. A. Watson, « Launch control safety study », *Bell Labs*, 1961.
- [55] W. E. Vesely, F. F. Goldberg, N. H. Roberts, et D. F. Haasl, « Fault tree handbook », Nuclear Regulatory Commission Washington DC, 1981.
- [56] G. Merle et J.-M. Roussel, « Algebraic modelling of Fault Trees with Priority AND gates », *IFAC Proc. Vol.*, vol. 40, n° 6, p. 31–36, 2007.
- [57] J. Cornfield, « Bayes theorem », *Rev. Inst. Int. Stat.*, p. 34–49, 1967.
- [58] H. Boudali, *A temporal Bayesian network reliability modeling and analysis framework*, ProQuest Dissertations Publishing. University of Virginia, 2005.
- [59] A. V. Kozlov et J. P. Singh, « Sensitivities: an alternative to conditional probabilities for Bayesian belief networks », *ArXiv Prepr. ArXiv13024969*, 2013.
- [60] A. Mosallam, K. Medjaher, et N. Zerhouni, « Integrated bayesian framework for remaining useful life prediction », in *2014 International Conference on Prognostics and Health Management*, 2014, p. 1–6.
- [61] M. K. Tay, K. Mekhnacha, C. Chen, M. Yguel, et C. Laugier, « An efficient formulation of the Bayesian occupation filter for target tracking in dynamic environments », *Int. J. Veh. Auton. Syst.*, vol. 6, n° 1-2, p. 155–171, 2008.
- [62] C. Laugier, « Embedded Sensor Fusion and Perception for Autonomous Vehicle », 2019.
- [63] S. Conrady et L. Jouffe, « Introduction to bayesian networks & bayesialab », *Bayesia SAS*, 2013.
- [64] P. Fuster-Parra, A. García-Mas, F. J. Ponseti, P. Palou, et J. Cruz, « A Bayesian network to discover relationships between negative features in sport: a case study of teen players », *Qual. Quant.*, vol. 48, n° 3, p. 1473–1491, 2014.
- [65] R. Alur et D. Dill, « The theory of timed automata », in *Workshop/School/Symposium of the REX Project (Research and Education in Concurrent Systems)*, 1991, p. 45–73.
- [66] C. A. Petri, « Communication par les automates », PhD Thesis, thèse de doctorat de l'université technologique de Darmstadt, Allemagne, 1962.
- [67] P. M. Merlin, « The Time-Petri-Net and the Recoverability of Processes », *AFIPS 75*, 1974, [En ligne]. Disponible sur: <https://escholarship.org/uc/item/7rq1j2vz>.
- [68] B. Berthomieu et M. Menasche, « An enumerative approach for analyzing time Petri nets », 1983.

- [69] B. Berthomieu et M. Diaz, « Modeling and verification of time dependent systems using time Petri nets », *IEEE Trans. Softw. Eng.*, vol. 17, n° 3, p. 259, 1991.
- [70] L. Popova-Zeugmann, *On Time Invariance in Time Petri Nets*. Professoren des Inst. für Informatik, 1994.
- [71] L. Popova-Zeugmann, *Essential states in time Petri nets*, Informatik-Berichte., vol. 96. Humboldt-Univ. zu Berlin, 1998.
- [72] G. Berthelot, H. Boucheneb, et U. Alger, « Occurrence graphs for interval timed coloured nets », in *International Conference on Application and Theory of Petri Nets*, 1994, p. 79–98.
- [73] W. M. P. Van der Aalst, « Interval timed Petri nets and their analysis », *Comput. Sci. Notes*, vol. 9109, 1991.
- [74] F. Basile, P. Chiacchio, et J. Coppola, « Identification of labeled time Petri nets », in *2016 13th International Workshop on Discrete Event Systems (WODES)*, 2016, p. 478–485.
- [75] B. Berthomieu, « La méthode des classes d'états pour l'analyse des réseaux temporels », in *3e congrès Modélisation des Systemes Réactifs (MSR'2001)*, 2001, p. 275–290.
- [76] L. Popova-Zeugmann, « Time petri nets », in *Time and Petri Nets*, Springer, 2013, p. 31–137.
- [77] B. Berthomieu, D. Lime, O. H. Roux, et F. Vernadat, « Problèmes d'accessibilité et espaces d'états abstraits des réseaux de Petri temporels à chronomètres », *J. Eur. Syst. Autom.*, vol. 39, n° 1/3, p. 223, 2005.
- [78] F. Liu et P. Yang, « Verification for the predictability of decentralized discrete event systems with a polynomial complexity », in *2017 36th Chinese Control Conference (CCC)*, juill. 2017, p. 2367–2372, doi: 10.23919/ChiCC.2017.8027712.
- [79] F. Basile, M. P. Cabasino, et C. Seatzu, « Marking estimation of Time Petri nets with unobservable transitions », in *2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA)*, 2013, p. 1–7.
- [80] M. A. Marsan, « Stochastic Petri nets: an elementary introduction », in *European workshop on applications and theory in Petri nets*, 1988, p. 1–29.
- [81] K. Bousson, « Raisonnement causal pour la supervision de processus basée sur des modèles », PhD Thesis, Toulouse, INSA, 1993.
- [82] R. Milne et al., « TIGER: real-time situation assessment of dynamic systems », *Intell. Syst. Eng.*, vol. 3, n° 3, p. 103–124, 1994.
- [83] Y. Pencolé et A. Subias, « A Chronicle-based Diagnosability Approach for Discrete Timed-event Systems: Application to Web-Services. », *J UCS*, vol. 15, n° 17, p. 3246–3272, 2009.
- [84] B. Guerraz et C. Dousson, « Chronicles construction starting from the fault model of the system to diagnose », in *International Workshop on Principles of Diagnosis (DX04)*, 2004, p. 51–56.
- [85] Y. Pencolé, « Diagnostic décentralisé de systèmes à événements discrets: application aux réseaux de télécommunications », PhD Thesis, Université Rennes 1, 2002.
- [86] C. Dousson, « Extending and unifying chronicle representation with event counters », in *Proceedings of the 15th European Conference on Artificial Intelligence*, 2002, p. 257–261.
- [87] S. H. Zad, R. H. Kwong, et W. M. Wonham, « Fault diagnosis in discrete-event systems: Incorporating timing information », *IEEE Trans. Autom. Control*, vol. 50, n° 7, p. 1010–1015, 2005.
- [88] S. H. Zad, R. H. Kwong, et W. M. Wonham, « Fault diagnosis in timed discrete-event systems », in *Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No. 99CH36304)*, 1999, vol. 2, p. 1756–1761.
- [89] A. Boussif, B. Liu, et M. Ghazel, « An experimental comparison of three diagnosis techniques for discrete event systems », Brescia, Italy, 2017, p. 8, [En ligne]. Disponible sur: <https://hal.archives-ouvertes.fr/hal-01647904>.
- [90] Y. Pencolé, G. Steinbauer, C. Mühlbacher, et L. Travé-Massuyès, « Diagnosing Discrete Event Systems Using Nominal Models Only. », in *DX*, 2017, vol. 4, p. 169–183.
- [91] N. Malki, « Contribution au diagnostic des Systèmes à Événements Discrets par modèles temporels et distributions de probabilité. », PhD Thesis, Reims, 2013.

- [92] R. Saddem et A. Philippot, « Causal temporal signature from diagnoser model for online diagnosis of discrete event systems », in *2014 International Conference on Control, Decision and Information Technologies (CoDIT)*, 2014, p. 551–556.
- [93] R. Ammour, E. Leclercq, E. Sanlaville, et D. Lefebvre, « Fault prognosis of timed stochastic discrete event systems with bounded estimation error », *Automatica*, vol. 82, p. 35–41, août 2017, doi: 10.1016/j.automatica.2017.04.028.
- [94] F. Nouioua et D. Kayser, « A Norm-Based Approach to Causal Reasoning », présenté à International Multidisciplinary Workshop on Causality, Université de Toulouse-Le Mirail, 2009.
- [95] F. Basile, P. Chiacchio, et G. De Tommasi, « On K-diagnosability of Petri nets via integer linear programming », *Automatica*, vol. 48, n° 9, p. 2047–2058, 2012.
- [96] B. Liu, M. Ghazel, et A. Toguyeni, « Évaluation de la volée de la diagnosticabilité des systèmes à événements discrets temporisés », *J. Eur. Syst. Autom. Ed. Spéc. MSR*, vol. 13, p. 227–242, 2013.
- [97] G. S. Viana, M. V. Moreira, et J. C. Basilio, « Codiagnosability analysis of discrete-event systems modeled by weighted automata », *IEEE Trans. Autom. Control*, vol. 64, n° 10, p. 4361–4368, 2019.
- [98] G. S. Viana et J. C. Basilio, « Codiagnosability of discrete event systems revisited: A new necessary and sufficient condition and its applications », *Automatica*, vol. 101, p. 354–364, 2019.
- [99] F.-J. Lian et S.-L. Shu, « Online Prognosis of Stochastic Discrete Event Systems », 2016.
- [100] S. Genc et S. Lafortune, « Predictability in discrete-event systems under partial observation 1 », *IFAC Proc. Vol.*, vol. 39, n° 13, p. 1461–1466, 2006.
- [101] A. Khoumsi, « Supervisory control for the conformance of real-time discrete event systems », *IFAC Proc. Vol.*, vol. 37, n° 18, p. 39–44, 2004.
- [102] A. Khoumsi, « Alternative Inference-Based Decentralized Prognosis of Discrete Event Systems », in *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, avr. 2019, p. 250–255, doi: 10.1109/CoDIT.2019.8820588.
- [103] S. Takai et R. Kumar, « Distributed failure prognosis of discrete event systems with bounded-delay communications », *IEEE Trans. Autom. Control*, vol. 57, n° 5, p. 1259–1265, 2011.
- [104] J. Chen et R. Kumar, « Failure prognosability of stochastic discrete event systems », in *2014 American Control Conference*, juin 2014, p. 2041–2046, doi: 10.1109/ACC.2014.6858775.
- [105] T. Jéron, H. Marchand, S. Genc, et S. Lafortune, « Predictability of sequence patterns in discrete event systems », *IFAC Proc. Vol.*, vol. 41, n° 2, p. 537–543, 2008.
- [106] S. Genc et S. Lafortune, « Predictability of event occurrences in partially-observed discrete-event systems », *Automatica*, vol. 45, n° 2, p. 301–311, 2009.
- [107] R. Kumar et S. Takai, « Comments on “Predictability of Failure Event Occurrences in Decentralized Discrete-Event Systems and Polynomial-Time Verification” », *IEEE Trans. Autom. Sci. Eng.*, vol. 16, n° 4, p. 1988–1989, 2019.
- [108] B. Benmessahel, M. Touahria, et F. Nouioua, « Predictability of fuzzy discrete event systems », *Discrete Event Dyn. Syst.*, vol. 27, n° 4, p. 641–673, 2017.
- [109] A. Grastien, « Interval Predictability in Discrete Event Systems », *CoRR*, 2015.
- [110] X. Yin et S. Li, « Robust Diagnosability and Robust Prognosability of Discrete-Event Systems Revisited », in *2018 IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, juill. 2018, p. 302–307, doi: 10.1109/CYBER.2018.8688259.
- [111] R. J. Barcelos, M. A. Corrêa, et J. C. Basilio, « Predictability of Discrete-Event Systems with Cycles of States Connected with Unobservable Events », *J. Control Autom. Electr. Syst.*, vol. 31, n° 4, p. 842–849, 2020.
- [112] G. Gardey, D. Lime, et M. Magnin, « Romeo: A tool for analyzing time Petri nets », in *International Conference on Computer Aided Verification*, 2005, p. 418–423.
- [113] X. Yin, « Verification of Prognosability for Labeled Petri Nets », *IEEE Trans. Autom. Control*, vol. 63, n° 6, p. 1828–1834, juin 2018, doi: 10.1109/TAC.2017.2756096.

- [114] B. Berthomieu, F. Vernadat, et S. D. Zilio, « The tool TINA – construction of abstract state spaces for Petri nets and time Petri nets », *International Journal of Production Research* 42(14):2741-2756.
- [115] T. Freytag, « Woped–workflow petri net designer », *Univ. Coop. Educ.*, p. 279–282, 2005.
- [116] O. H. Roux et A.-M. Déplanche, « A t-time Petri net extension for real time-task scheduling modeling », *Inst. Rech. En Commun. Cybernétique Nantes*, vol. 36, 2002.
- [117] P. H. Starke et S. Roch, « INA: integrated net analyzer », *Comput. Sci.*, 1999.
- [118] L. Popova-Zeugmann, « Time Petri nets state space reduction using dynamic programming », *Control Cybern.*, vol. 35, n° 3, p. 721–748, 2006.
- [119] P. Mayé, *Générateurs électrochimiques: Piles, accumulateurs et piles à combustible*. Dunod, 2010.
- [120] C. Savard, « Modélisation par un graphe de flots d’une architecture alternative pour les systèmes de stockage multi-cellulaire de l’énergie électrique », présenté à JCGE, Arras, France, 2017, [En ligne]. Disponible sur: <https://hal.archives-ouvertes.fr/hal-01534706>.
- [121] B. Benmessahel, M. Touahria, F. Nouioua, J. Gaber, et P. Lorenz, « Decentralized prognosis of fuzzy discrete-event systems », *Iran. J. Fuzzy Syst.*, vol. 16, n° 3, p. 127–143, 2019.

Annexe 1 : fonctionnement d'une cellule

Pour évaluer les risques posés par le stockage de l'électricité dans une cellule en lithium, il est très utile de connaître son fonctionnement. Il faut savoir qu'il n'existe pas une cellule en Lithium. Il existe une variété de cellules dans lesquelles le lithium est utilisé à l'état pur. La cellule Lithium ou Lithium-ion comprend une anode qui représente l'électrode positive, et la cathode pour l'électrode négative séparées par un électrolyte conducteur d'ions. Ce dernier permet le transport des ions lithium entre les électrodes pendant le processus de charge (figure 6.1) ou de décharge.

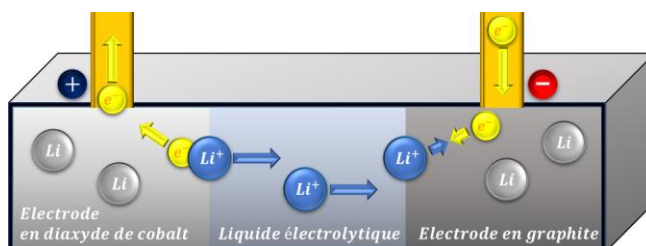


Figure 5.1: Chargement d'une cellule

Ce séparateur empêche le contact direct entre les deux électrodes et évite un court-circuit. L'anode d'une cellule chargée (figure 6.2) est gorgée de lithium, prêt à se transformer en ion, et sa cathode est prête à recevoir les ions. Une fois la cellule connectée à une charge électrique (figure 6.3), elle débitera du courant tant qu'il y a des ions disponibles à l'anode avec une cathode qui les accueille. Comme il est impossible de connaître l'état des réactifs une fois la cellule scellée dans un emballage, les fabricants donneront les courbes de décharges et les valeurs de tensions limites pour lesquelles ils savent que les réactifs commencent à manquer. Ils indiquent que la réaction d'oxydoréduction de cette cellule pourra se faire sans manque de réactifs jusqu'à ce que la cellule affiche une valeur val en Volt à ses bornes. Le fabricant indiquera aussi la vitesse maximale à laquelle la réaction de décharge peut réagir, indiquant alors un courant maximal pouvant entrer ou sortir de la cellule.

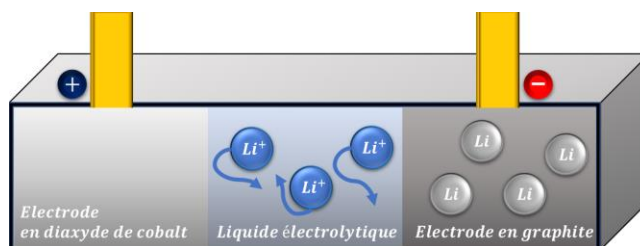


Figure 5.2: Cellule chargée.

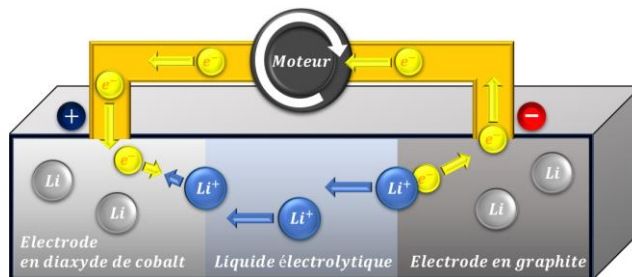


Figure 5.3 :Déchargement d'une cellule.

Si la cellule Lithium-ion n'est pas dans des conditions de fonctionnement appropriée ou si elle présente des défauts techniques de fabrication, elle peut poser un risque de sécurité important et provoque souvent des incendies dévastateurs. Le danger réside dans le contact entre les électrolytes inflammables et les matériaux à haute densité énergétique. Parmi les causes majeures de ces incendies :

La surcharge de la cellule :

La cellule surchauffe si elle est surchargée ou exposée à une température trop élevée appelée "emballement thermique". L'électrolyte s'évapore à cause de l'augmentation de l'énergie thermique, provoquant ainsi une chaleur importante et des gaz combustibles. Lorsque cette chaleur dépasse la température d'inflammation du gaz, celui-ci s'enflamme et enflamme le lithium réactif. En quelques minutes la cellule explose.

La décharge complète de la cellule :

La non-utilisation d'une cellule pendant une longue durée, cause sa décharge complète et provoque ainsi la décomposition de l'électrolyte, ce qui forme des gaz facilement inflammables. La tentative du chargement d'une cellule déchargée complètement ne peut aboutir en raison du manque du fluide électrolytique qui ne permet pas la conversion de l'énergie fournie. Ce qui cause un court-circuit et par conséquent un incendie.

Dommmages mécaniques :

La déformation d'une cellule entraîne un court-circuit, en plus des impuretés qui ne peuvent être exclues à 100 % lors de sa production. Cependant, l'incendie n'est pas l'unique risque d'une cellule lithium-ion ; des substances nocives telles que l'acide fluorhydrique ou chlorhydrique qui sont fuis à l'intérieur de la cellule risquent d'être sécrétées sous forme de vapeur ou se diluer et s'infiltrer dans le sol et provoquent des dégâts environnementaux.

La consigne de sécurité principale pour éviter ces risques est de contrôler la charge et la décharge de la cellule. Le but est de limiter la surcharge ou la décharge complète de la cellule. C'est dans ce cadre que nous essayons d'exploiter notre approche de pronostic, afin de déterminer la date au plus tôt qui permet à l'opérateur d'intervenir avant le disfonctionnement de la cellule.

Parmi les chimies utilisées en 2020 dans les batteries de voitures électriques (figure 4.2), nous citons :

- LMO (Lithium Métal Oxide),
- NMC (Lithium Nickel Manganese Cobalt Oxide)
- LFP (Lithium Iron Phosphate)
- NCA (Lithium Nickel Cobalt Aluminium Oxide)

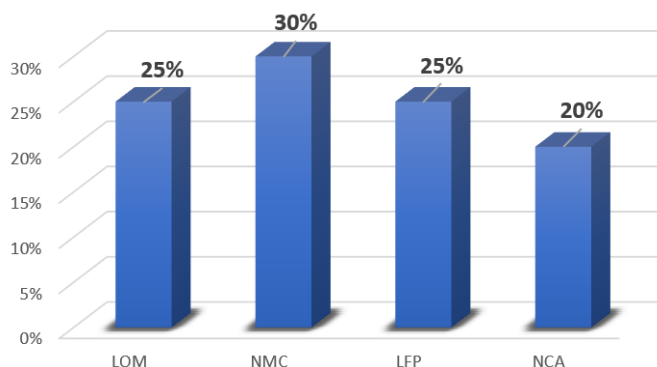
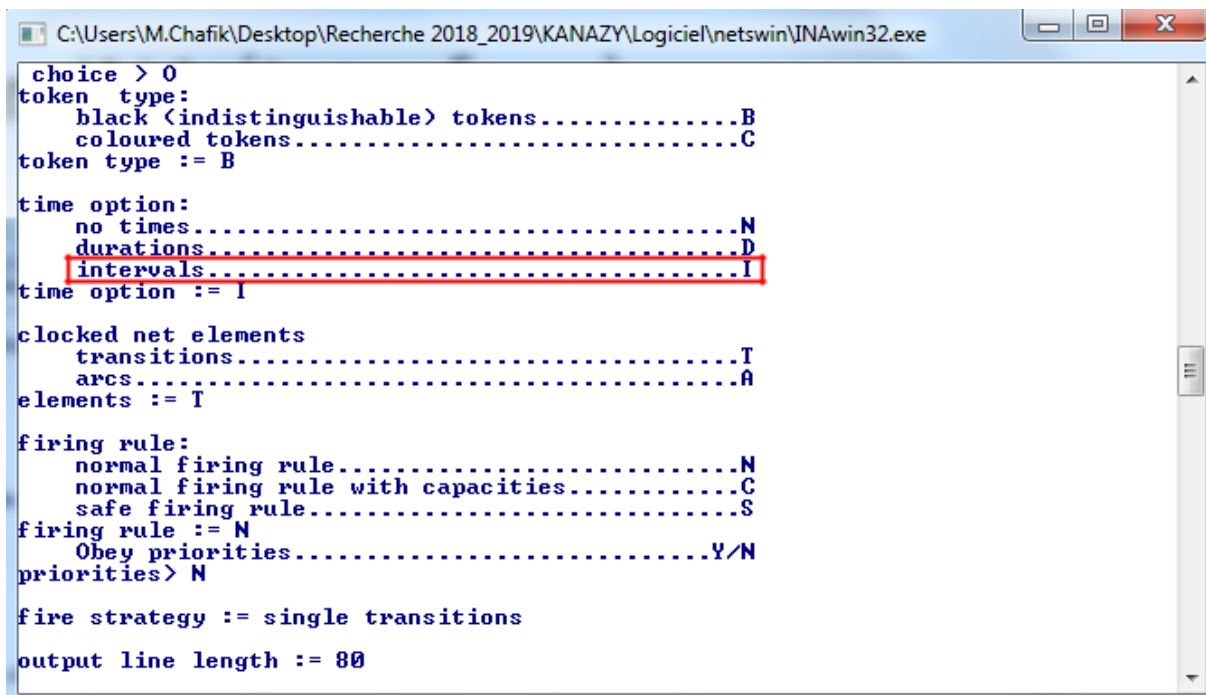


Figure 5.4: Taux d'utilisation des chimies pour l'alimentation des VE en 2020.

Dans la nouvelle configuration, il faut prendre en considération l'aspect temporel. C'est la raison pour laquelle nous avons choisi l'option intervals dans time option (figure 5.7).



```

C:\Users\M.Chafik\Desktop\Recherche 2018_2019\KANAZY\Logiciel\netswin\INAwin32.exe

choice > 0
token type:
  black <indistinguishable> tokens.....B
  coloured tokens.....C
token type := B

time option:
  no times.....N
  durations.....D
  intervals.....I
time option := I

clocked net elements
  transitions.....T
  arcs.....A
elements := I

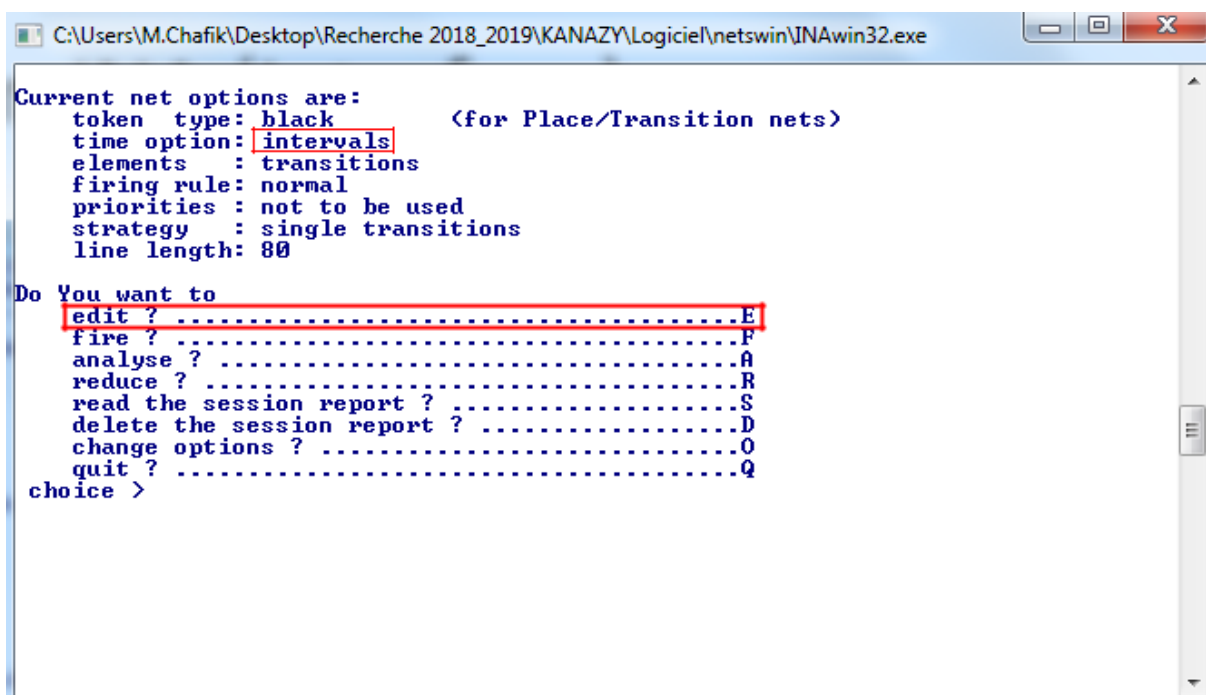
firing rule:
  normal firing rule.....N
  normal firing rule with capacities.....C
  safe firing rule.....S
firing rule := N
  Obey priorities.....Y/N
priorities> N

fire strategy := single transitions

output line length := 80
  
```

Figure 5.7 : Configuration de l'environnement INA

Pour éditer un modèle INA, il faut saisir le caractère "E" (figure 5.8)



```

C:\Users\M.Chafik\Desktop\Recherche 2018_2019\KANAZY\Logiciel\netswin\INAwin32.exe

Current net options are:
  token type: black      <for Place/Transition nets>
  time option: intervals
  elements : transitions
  firing rule: normal
  priorities : not to be used
  strategy : single transitions
  line length: 80

Do You want to
edit ? .....E
fire ? .....F
analyse ? .....A
reduce ? .....R
read the session report ? .....S
delete the session report ? .....D
change options ? .....O
quit ? .....Q
choice >
  
```

Figure 5.8 : Environnement INA après paramétrage.

Deux méthodes sont possibles pour écrire l'interprétation textuelle du modèle RdP (figure 5.9) ; une à base fichier avec une extension (*.pnt) et l'autre sur le terminal. Nous avons choisi la méthode qui permet de générer un fichier pour pouvoir l'exploiter dans d'autre environnement d'analyse des RdP.

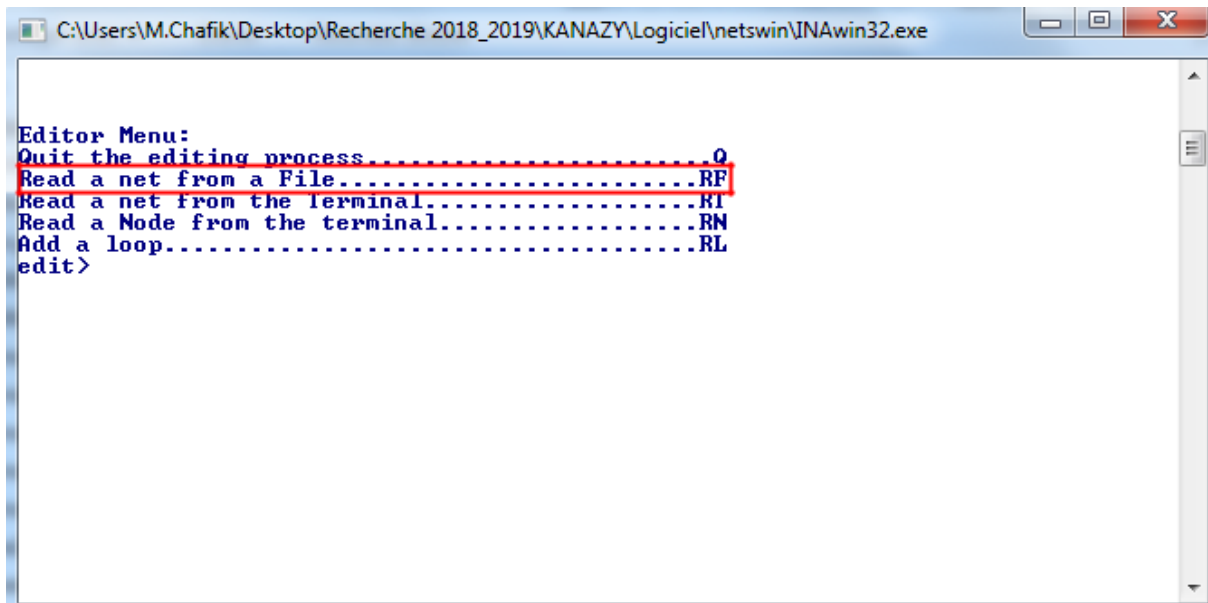


Figure 5.9 : Menu Editor INA

Pour écrire le code il faut saisir “WF” (figure 5.10).

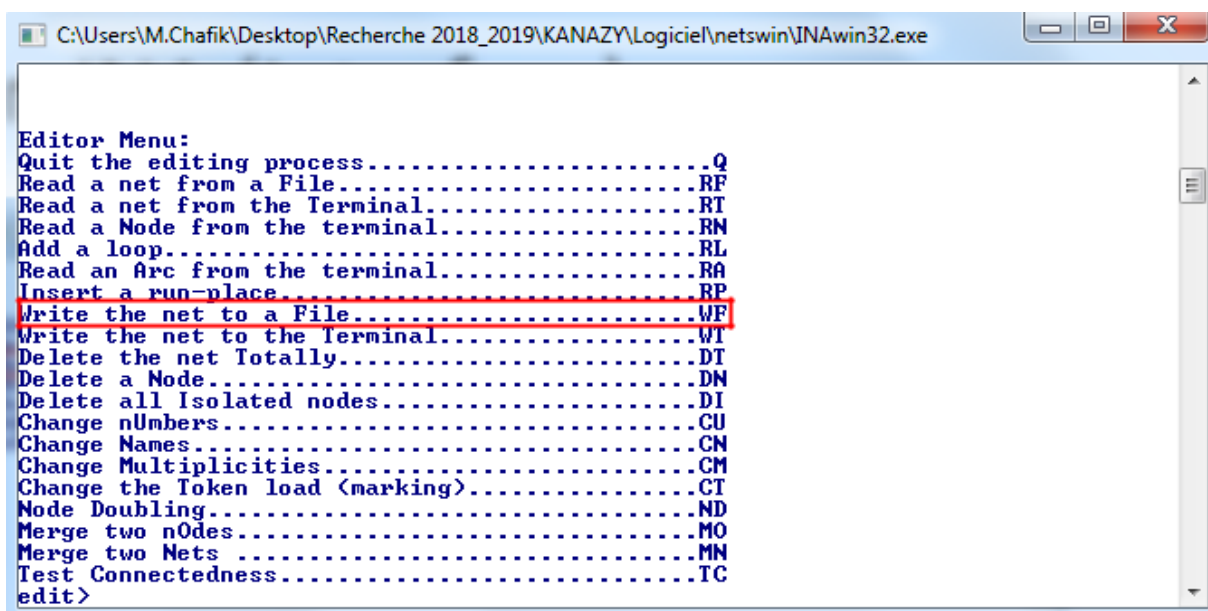


Figure 5.10 : Menu Editor INA et génération de fichier npt

La syntaxe de programmation d'un modèle RdP est la suivante :

Chaque place du RdP est décrite sur une ligne i.e. numéro de place suivi pas un espace ensuite le nombre de marquage dans la place, si cette place a les transitions d'entrée et de sortie sont représentées comme des listes séparées par des virgules. La structure du code est $p_j \text{ nb_jeton_}p_j [T_pré], [T_post]$.

Exemple :

Soit le modèle de la [figure 4.2](#) du 4^{ème} chapitre. Le programme équivalent sous INA est :

```
P M PRE,POST
0 1 3 6, 0
1 0 0, 1 5
2 0 1, 2
3 0 2, 3 4
4 0 11 12, 13
5 0 4 10, 7
6 0 7, 8 12
7 0 5 8, 9
8 0 9, 10 11
```

Pour procéder à l'analyse du modèle RdP, il faut saisir "A" dans le menu principal ([figure 5.5](#)). Un menu sera alors présenté ([figure 5.11](#)) permettant à l'utilisateur de déterminer le type d'analyse qu'il souhaite effectuée et affiche les résultats demandés concernant les propriétés du modèle.

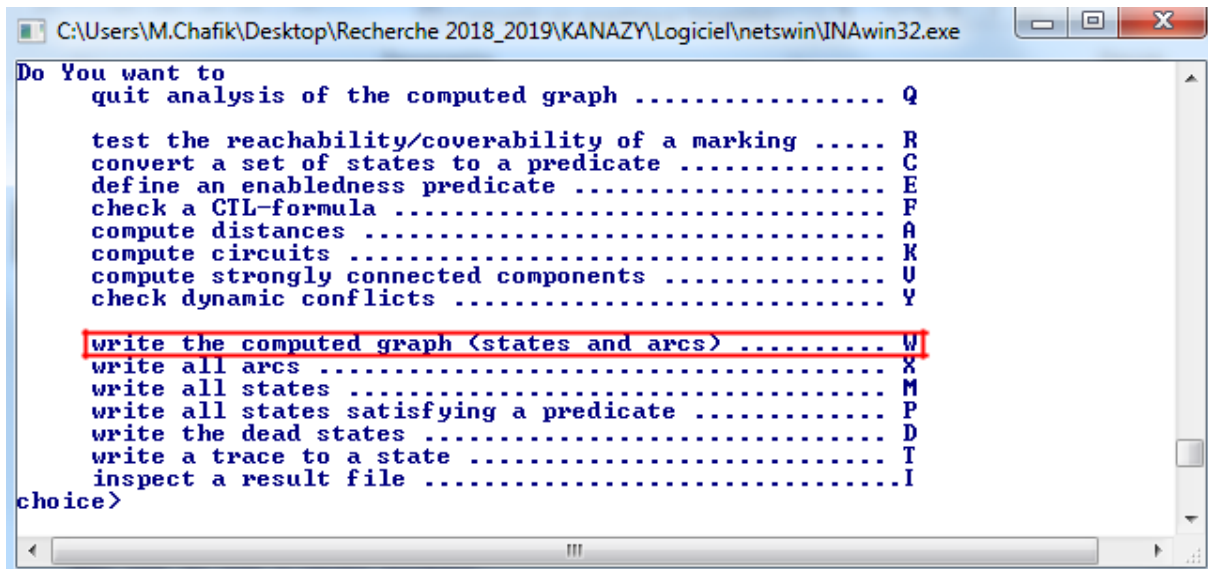


Figure 5.11 : Menu d'analyse INA

Annexe 3 : Publications

Titre : Prognosis of Failure Events Based on Labeled Temporal Petri Nets

Date de publication : 18 juin 2020

Description de la publication : Advances in Science, Technology and Engineering Systems Journal

Abstract : To reduce the risk of accidental system shutdowns, we propose to control system developers (supervisor, SCADA) a prediction tool to determine the occurrence date of an imminent failure event. The existing approaches report the rate of occurrence of a future failure event (stochastic method), but do not provide an estimation date of its occurrence. The date estimation allows to define the system repair date before a failure occurs. Thus, provide visibility into the future evolution of the system. The approach consists in modelling the operating modes of the system (nominal, degraded, failed); the goal is to follow the evolution of the system to detect its degradation (switching from nominal to degraded mode). When degradation is reported, a prognoser is generated to identify all possible sequences and more precisely those ending with a failure event. then it checks among the sequences (with failure event) which ones are prognosable. The last step of the approach is carried out in two parts: the first part consists in calculating the execution time of the so-called prognosable sequences (by optimizing the number of possible states and resolving an inequalities system). The second part makes it possible to find the minimum execution (the earliest occurrence of a failure event).

Auteurs : Redouane KANAZY, Samir CHAFIK, Éric NIEL

Titre : Failure prognosis in discrete events systems based on extended Time petri nets: example of an electric car battery cell

Date de publication : 1 oct. 2019

Description de la publication : SYSTOL19 IEEE (Conférence)

Abstract : The complexity of critical systems does not tolerate an unplanned stop of attempted functionalities, which destabilizes their operation and causes material and human damage. This makes failure prognosis a requirement that scientific communities are constantly responding to through developing new techniques that reduce the impact of failures on the system. The control of failure events and the system repair date are very important in discrete event systems (DES) area to meet this requirement. Therefore, in the first part of this paper we propose a prognosis approach based on extended Petri Nets (EPN). A technique which makes it possible to determine the occurrence of a future failure event τ -time units in advance. For multi-resource prognosis, we will use several clocks.

Auteurs : Redouane KANAZY, Samir CHAFIK, Éric NIEL, Mohammed ZOUAGUI

Titre : Failure prognosis of discrete events systems based on extended Petri Nets.

Date de publication : 16 Juin. 2018

Description de la publication : ESREL18 (conférence)

Abstract : Fault prognosis has become a major scope for complex and interconnected systems. Such significant events as fault events can cause partial or total stop of attempted functionalities. Prevention failure events are an issue to preserve performance, availability and safety of both operators and equipment. The aim of prognosis is to prevent fault events before their occurrence. Fault/repair management refers to event control, and so it is relevant to the domain of Discrete Events Systems (DES), for which stochastic finite state automaton and Petri Nets (PN) have been used to prognosticate fault state. They are based on predictions of fault event at least m -steps in advance. The proposal is based on the time notion, which is crucial for fault prognosis. Indeed, one can give the remaining time before the occurrence of fault event. The goal is to prevent the occurrence of a fault event at τ – timeunits in advance. This approach is based on labeled and T-temporal Petri Net, which has the advantage of a formal character for the assessment of properties and of sufficiently generic in order to apprehend a high level of complexity.

Auteurs : Redouane KANAZY, Samir CHAFIK, Éric NIEL

Titre : Contribution onto Prognosability based on discrete events systems.

Date de publication : oct. 2018

Description de la publication : ICCAD18 IEEE (conférence)

Abstract : Developing a method of prognosis, allows to avoid any accidental stop of the system, by predicting the future failure occurrence. Which makes it possible to preserve performance, availability and safety of both operators and equipment. To strengthen this technique, various research works have developed the prognosability, as a necessary and sufficient condition for prediction of a failure without missed detection and false alarm. This paper is a continuation of the works that has been realized on the prognosis of failure on extended Petri Net (PN) based on discrete events systems (DES); a prognostic method based on time, to predict the future failure occurrence at the earliest and at the latest. The aim is to prove that prognosability is possible on extended PN, while respecting these necessary and sufficient conditions.

Auteurs : Redouane KANAZY, Samir CHAFIK, Éric NIEL

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : KANAZY**DATE de SOUTENANCE :** 15/12/2020

(avec précision du nom de jeune fille, le cas échéant)

Prénoms : Redouane**TITRE :** Pronostic des événements de défaillance basé sur les réseaux de Petri temporels labellisés**NATURE :** Doctorat**Numéro d'ordre :** 2020LYSEI132**Ecole doctorale :** EEA - 160**Spécialité :** Automatique**RESUME :**

L'utilisation des outils d'aide à la décision accroît les exigences en termes d'efficacité, d'agilité et de qualité pour réduire les coûts relatifs au maintien du bon fonctionnement des systèmes supervisés. Un arrêt accidentel ou intentionnel, qu'il soit partiel ou total du système, provoque parfois des conséquences désastreuses et coûteuses. La communauté scientifique opérant sur les systèmes à événements discrets (SED), s'est intéressée au comportement du système et notamment aux relations de causes à effets entre ses états, pour proposer une solution d'aide à la décision, qui répond à cette problématique.

Nos travaux s'insèrent dans le cadre d'un pilotage d'un système soumis à des événements de défaillances. Nous avons développé une approche de pronostic à base de modèle, où tout comportement possible du système est supposé être connu et explicitement inclus dans le modèle. Notre analyse du modèle comportemental du système est basée sur la séquentialité et la date d'occurrence des événements, c'est la raison pour laquelle nous avons choisi de modéliser le système par les réseaux de Petri temporels labellisés (RdPTL). L'intérêt de l'approche est de prédire la date au plus tôt de l'occurrence d'un événement de défaillance. Nous avons représenté les dynamiques du système au travers une modélisation dans un contexte d'analyse de modes, limitée à 3 modes de fonctionnement (nominal, dégradé et critique). En première approximation et à des fins de démonstration, nous avons supposé que le modèle RdPTL est saut, une seule transition est tirée à la fois, son tir est immédiat, il n'y a pas de cycles d'exécution et un seul mode de fonctionnement est actif à la fois. Afin d'assurer la surveillance du système, nous avons construit un pronostiqueur à partir du graphe d'accessibilité du modèle RdPTL. Ce pronostiqueur permet de repérer l'ensemble des séquences d'événements qui se terminent par un événement de défaillance. Pour pallier la difficulté liée au franchissement d'une transition dans un RdPTL est possible à tout instant dans son intervalle de tir, nous avons utilisé une paramétrisation des états du pronostiqueur i.e. l'introduction d'une horloge et un système d'inéquations des horloges pour chaque état du système. Les états obtenus de la discrétisation du temps sont alors regroupés dans un seul état et le système d'inéquations déterminera les valeurs des horloges.

Nous avons choisi comme benchmark la cellule d'une batterie de voiture électrique. Nous avons utilisé le modèle des modes de fonctionnement et le système d'inéquations basé sur le modèle pronostiqueur pour déterminer la date au plus tôt de l'occurrence d'un événement de défaillance. Nous avons choisi INA comme outil de simulation pour générer le graphe d'accessibilité et déterminer la durée d'exécution minimale des séquences pronosticables.

MOTS-CLÉS :

Pronostic, diagnostic, systèmes à événements discrets, pronosticabilité, diagnosticabilité, réseaux de Petri temporels labellisés

Laboratoire (s) de recherche : Ampère**Directeur de thèse:** Pr Éric NIEL**Président de jury :** Pr Oulaid KAMACH**Composition du jury :**

Armand TOGUYENI, Zineb SIMEU-ABAZI, Oulaid KAMACH, Éric NIEL, Samir CHAFIK