Diagnosticabilité et diagnosticabilisation de motifs temporels dans les réseaux de Petri temporels

Camille Coquand

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)*

Présentée et soutenue le *04/12/2023* **par :**

Camille Coquand

Diagnosticabilité et diagnosticabilisation de motifs temporels dans les réseaux de Petri temporels

---

JURY

Étienne André	Professeur d'Université	Examinateur
Olivier (H.) Roux	Professeur d'Université	Rapporteur
Mohamed Ghazel	Directeur de Recherche	Rapporteur
Dimitri Lefebvre	Professeur d'Université	Président du Jury
Audine Subias	Professeure d'Université	Directrice de thèse
Yannick Pencolé	Chargé de Recherche	Co-encadrant de thèse

École doctorale et spécialité :
EDSYS : Informatique 4200018
Double mention :
EDSYS : Automatique 4200046
Unité de Recherche :
Laboratoire d'analyse et d'architechture des systèmes LAAS-CNRS (UPR 8001)
Directeur(s) de Thèse :
Audine Subias et *Yannick Pencolé*
Rapporteurs :
Olivier (H.) Roux et *Mohamed Ghazel*

Remerciements

Comme l'a dit un jour un grand Homme, *si la vie est un long fleuve tranquille, le doctorat se compare aux chutes du Niagara*. Cette citation image très bien les différents états émotionnels par lesquels un doctorant passe durant ces trois années de recherche. Je souhaite remercier l'ensemble des personnes qui m'ont accompagné dans toutes ces émotions.

Pour commencer j'aimerai remercier ma directrice de thèse : Audine. Le choix a été dur au départ, mais je n'ai pas regretté un seul instant d'avoir choisi ce sujet. Merci pour ces nombreuses discussions scientifiques, pour avoir suivi les directions que j'ai pu prendre, et merci pour avoir supporté mon caractère durant ces trois années.

J'aimerai ensuite m'adresser à Yannick. Merci pour toutes ces discussions, et ces moments interminables passés au tableau. Tu m'as appris énormément depuis mon stage de M2, et tu es une grande source d'inspiration pour mon hypothétique future carrière dans la science.

Je voudrais remercier les membres de mon jury d'avoir accepté de juger ma thèse et d'avoir fait le déplacement lors de la soutenance. Merci Mohamed, Étienne, Olivier et Dimitri. Les discussions lors de la soutenance ont été un réel plaisir pour moi. J'espère qu'il y en aura d'autres dans le futur.

Merci à Élodie, Didier et Gwendoline de m'avoir fait profiter de leur expérience d'enseignement : je saurai mettre cet apprentissage à profit dans le futur. Merci à l'équipe DISCO de m'avoir accueilli, et plus particulièrement merci à Euriell pour m'avoir guidé en conférence, et à Soheib pour avoir été un compagnon fidèle de pause café même pendant les périodes de COVID où les couloirs étaient déserts.

J'aimerai adresser un merci tout particulier à Pauline. Discuter avec toi a toujours été un plaisir, merci pour ta disponibilité. Un merci tout particulier à Fred pour ces nombreuses discussions sur l'automatique et sur l'enseignement. Vous êtes tous les deux des sources d'inspiration pour moi.

J'aimerai également remercier les membres de mon équipe d'adoption durant la fin de ma thèse : l'équipe VERTICS. Merci à Silvano et François pour leur écoute et les nombreuses discussions qui m'ont permi d'avancer dans ce travail. Les résultats que j'ai obtenu n'auraient pas été les mêmes sans vous. Vous êtes de grandes sources d'inspiration pour le jeune chercheur que je suis.

Merci à Claire, Alexandre, Ibis, Nicola, Manon, Mathieu, Le Toan, Léonie, Rafael, et à l'ensemble des doctorants de DISCO et MAC avec qui nous avons pu nous soutenir mutuellement durant ces années de doctorat. Un merci tout particulier à Éric et Yoni pour leur humour qui a bien souvent aidé à détendre l'atmosphère morose des mauvais jours.

Merci à mon camarade de rédaction de manuscrit : Nicolas. C'est dans la douleur que se forgent les grandes amitiés, la notre sera très grande. Je te souhaites de devenir un grand chercheur, tu as tout ce qu'il faut pour le devenir.

Enfin, je souhaite remercier mon collocataire de bureau/grand frère de labo, celui qui m'a accueilli quand je n'étais qu'un stagiaire, celui avec qui j'ai refait le monde tant de fois, celui dont la créativité n'a d'égal que sa bonne humeur à toute épreuve : docteur Adrien Dorise. Je n'aurai pu réver de meilleur compagnon durant la période de pandémie.

Ces remerciements ne seraient pas complets sans citer les deux dames qui ont partagées ma vie durant ces trois ans : Micaiah, mon chat qui par ses passages intenpestifs devant la caméra durant les réunions Zoom en confinement ont beaucoup amusé mes encadrants, et Axelle, ma splendide épouse qui a eu la patience de m'écouter raconter mes divagations mathématiques et qui m'a soutenu sans faille.

Résumé

La diagnosticabilité est la propriété d'un système de produire suffisamment d'observations (par l'intermédiaire de capteurs) pour différencier les comportements nominaux des comportements fautifs. Il est possible pour un système diagnosticable pour une faute donnée (ou défaut) de produire un diagnostic certain, les informations observables retournées par le système étant différentes selon si une faute a eu lieu ou non. Vérifier une telle propriété sur un système est donc une étape importante dans le processus de construction d'algorithmes de détection et d'isolation de telles fautes. Ce problème a fait l'objet de multiples études dans les Systèmes à Événements Discrets (SED) atemporels (STL, réseaux de Petri, etc.) et dans les modèles SED temporels (automates temporisés, réseaux de Petri temporels, etc.), que ce soit pour des fautes simples (modélisées par l'occurrence d'un unique événement) ou pour des comportements plus complexes appelés motifs de faute.

Dans ce travail nous proposons une méthode d'analyse de diagnosticabilité de réseaux de Petri temporels pour un nouveau type de défaut appelé motif temporel. Les motifs temporels consistent en l'occurrence de différents événements contraints temporellement. Cette méthode se base sur une abstraction des préfixes du langage d'un réseau de Petri temporel sous forme de polyèdres qui constitue la première contribution de cette thèse. La seconde contribution de ce travail est une méthode d'analyse de diagnosticabilité, qui permet de vérifier si un système est diagnosticable pour un motif temporel, et de fournir des explications structurelles de non-diagnosticabilité le cas échéant. Cette méthode se base sur l'analyse des polyèdres issus de l'abstraction présentée, qui sont découpés en différentes zones partageant des propriétés de diagnostic communes. De cette analyse de zones est synthétisé un graphe regroupant les différentes sources d'ambiguïtés dans les différentes exécutions du système étudié. La dernière contribution de ce travail est une méthode de vérification pour le problème de diagnosticabilisation temporelle qui consiste à paramétrer les intervalles statiques du réseau de Petri temporel initial et à trouver des valeurs des paramètres pour lesquelles le réseau ainsi généré est diagnosticable pour le motif étudié. Nous proposons une méthode de vérification de cette propriété garantissant certaines propriétés sur le réseau de Petri engendré. Cette méthode de vérification se base sur l'abstraction polyédrique précédemment citée, et consiste en la construction d'ensembles de contraintes linéaires. La vérification de l'existence de solution peut être réalisée par la suite à l'aide d'un solveur SMT.

Mots clés :

Diagnosticabilité **Motifs temporels** **Diagnosticabilisation**
Réseau de Petri Temporel **Systèmes à événements discrets**

Abstract

Diagnosability is the property of a system to produce sufficient observations (via sensors) to differentiate nominal from faulty behavior. It is possible for a diagnosable system to produce a certain diagnosis for a given fault, since the observable information returned by the system differs according to whether or not a fault has occurred. Verifying such a property on a system is therefore an important step in the process of building fault detection and isolation algorithms. This problem has been the subject of numerous studies in atemporal Discrete Event Systems (DES) (STL, Petri nets, etc.) and in time DES models (timed automata, time Petri nets, etc.), whether for simple faults (modeled by the occurrence of a single event) or for more complex behaviors called fault patterns.

In this work, we propose a method for analyzing the diagnosability of time Petri nets for a new type of fault called time pattern. Time patterns consist of the occurrence of different temporally constrained events. The method is based on a polyhedral abstraction of time Petri net language prefixes, which is the first contribution of this thesis. The second contribution of this work is a diagnosability analysis method, which makes it possible to check whether a system is diagnosable for a time pattern, and to provide structural explanations for non-diagnosability. This method is based on the analysis of polyhedra of the abstraction presented, which are divided into different zones sharing common diagnosis properties. From this analysis of zones is synthesized a graph grouping together the various sources of ambiguity in the different executions of the system. The last contribution of this work is a verification method for the time diagnosabilization problem, which consists in ! parameterizing some bounds of the static time intervals of the initial time Petri net and finding values for these parameters for which the net thus generated is diagnosable for the pattern under study. We propose a method to check diagnosabilization that guarantees certain properties of the generated Petri net. This verification is based on the previously mentioned polyhedral abstraction, and consists in the construction of linear constraint sets. Verification of the existence of solutions can then be carried out using an SMT solver.

Table des matières

Table des figures

Notations et abréviations

Nous utiliserons dans ce manuscrit les notations suivantes :

$\mathbb{N}$	L'ensemble des entiers naturels
$\mathbb{N}^*$	L'ensemble des entiers naturels strictement positifs
$\mathbb{R}_+$	L'ensemble des réels positifs
$\mathbb{R}_+^*$	L'ensemble des réels strictement positifs
$\mathbb{Q}_+$	L'ensemble des rationels positifs.
$\mathbb{Q}_+^*$	L'ensemble des rationels strictement positifs
$I_{\mathbb{Q}_+}$	$= \mathbb{Q}_+ \times \mathbb{Q}_+$
Σ	Un alphabet
Σ^*	La clôture de Kleene de Σ
$\sigma_{\|i}$	La réduction de σ à ses i premiers éléments ($i < \|\sigma\|$)
$\|\sigma\|$	Le nombre de symboles dans la séquence σ
$pref(\rho)$	L'ensemble des préfixes de ρ
ε	Le mot vide
$\downarrow(I)/\uparrow(I)$	La borne inférieure (resp. supérieure) d'un intervalle I
Θ	Un système Θ
Ω	Un motif de faute temporel
$\rho \Subset \Omega$	La trace ρ satisfait le motif Ω
Π	Un polyèdre (appelé aussi ensemble de contraintes)
$\mathcal{S}(\Pi)$	L'ensemble des solutions d'un polyèdre Π
$\mathcal{R}_{x,y}$	La réduction d'un polyèdre à ses variables x et y
$\mathcal{C}_f(\Omega)$	L'ensemble des chemins fautifs de Θ pour le motif Ω
$\mathcal{C}_\Omega$	L'ensemble des chemins d'un motif Ω
Λ	Un ensemble de paramètres à valeur dans $\mathbb{R}_+$
λ	Un paramètre à valeurs dans $\mathbb{R}_+$
$\wedge$	Le *et* logique
$\vee$	Le *ou* logique
$\oplus$	Le *ou exclusif* logique
$\models$	L'*implication sémantique* logique
$(\mathbb{R}_+ \times \Sigma)^*$	L'ensemble des séquences de couples d'éléments de $\mathbb{R}_+$ et Σ
$(\mathbb{R}_+ \times \Sigma)^+$	$(\mathbb{R}_+ \times \Sigma)^* \setminus \varepsilon$

Nous utiliserons également les abréviations suivantes :

SED	Systèmes à Événements Discrets
RdP	Réseau de Petri
PT	réseau de Petri Temporel
GC	Graphe des Classes
PTP	réseau de Petri Temporel Paramétrique
PTPr	réseau de Petri Temporel Produit
STÉ	Système de Transitions Étiqueté
STT	Système de Transitions Temporel
AT	Automate Temporisé
ATH	Automate Temporisé avec Horloge unique
ATP	Automate Temporisé Paramétrique
PTÉS	réseau de Petri Temporel Étiqueté
GAT	Graphe d'Ambiguïtés Temporel
GATR	Graphe d'Ambiguïtés Temporel Réduit

Préambule

Le travail proposé dans ce manuscrit est principalement théorique, et s'inscrit selon moi dans deux thématiques générales : l'Automatique et l'Informatique de par la modélisation des problèmes étudiés, les développements théoriques proposés et les approches choisies pour répondre à certains problèmes.

Lorsque j'ai postulé à la bourse qui m'a permis de réaliser cette thèse, le sujet était le suivant :

Diagnostic certifié de motifs de fautes dans les systèmes à événements discrets temporels

Ce titre se décompose en deux parties : d'une part le *diagnostic certifié* ; et d'autre part le sujet d'étude de ce diagnostic, les *motifs de faute dans les systèmes à événements discrets temporels*. La partie diagnostic certifié évoque la synthèse d'algorithmes permettant de décider avec certitude de l'occurrence d'une faute dans le comportement d'un système. Ces algorithmes reproduisent généralement les raisonnements de leur auteur qui permettent de discriminer les observations retournées par un système en fonction de l'occurrence ou non de la faute dans le comportement étudié. Les systèmes à événements discrets temporels évoquent une structure logique sur laquelle il est possible de vérifier des propriétés (à l'aide de logiques temporelles par exemple [PS17]), à laquelle le domaine temporel a été ajouté. Ce domaine temporel étant continu, son intégration impose de ne plus raisonner sur des structures logiques uniquement, mais d'allier logique et raisonnement sur des formes géométriques qui sont une sous partie de $\mathbb{R}_+^n$ [BM83]. Ce sujet est étudié depuis plusieurs années par mes directeurs de thèse Audine Subias et Yannick Pencolé, et s'inscrit à la suite d'une quinzaine d'années de recherches. Leur dernier travail en date au démarrage de ma thèse est une méthode de vérification de diagnosticabilité de motifs de fautes dans les réseaux de Petri temporels [PS21]. Nos premières discussions ont étendu les motifs de faute [Jér+06] en y intégrant le temps.

De nature très curieuse, j'observe le monde qui m'entoure en me posant toujours ces deux questions : pourquoi ça marche et comment ça marche ? Ces deux questions ont orienté la manière dont je me suis approprié ce sujet.

Tout d'abord la notion de diagnostic certifié fait écho à la notion de *diagnosticabilité* [Sam+95], qui est la propriété d'un système vis-à-vis d'une faute donnée de toujours pouvoir différencier les comportements fautifs des comportements nominaux avec un nombre suffisant d'observations. La clé du diagnostic certifié se trouve dans cette notion. Ainsi j'ai voulu comprendre pourquoi un système est diagnosticable, et synthétiser de manière formelle cette explication [PCC02]. La première année de ma thèse a donc été passée à chercher à mieux comprendre le formalisme que j'ai utilisé pour modéliser les systèmes temporels : les *réseaux de Petri temporels* [Mer74]. La vision acquise durant

cette première année m'a permis durant ma deuxième année d'étudier la diagnosticabilité de réseaux de Petri temporels pour ces motifs *temporels*. Lorsque ces travaux ont abouti deux choix se sont offerts à moi :

— Implémenter une méthode de diagnosticabilité en se basant sur les résultats obtenus, et optimiser cette implémentation pour qu'elle passe l'épreuve des fautes multiples et que les temps de calcul soient acceptables pour des systèmes dont l'espace d'état est de taille conséquente.
— Paramétrer le modèle temporel (ses contraintes de temps) pour le rendre diagnosticable pour une faute donnée, piste suggérée dans le sujet initial. L'opération de rendre un modèle diagnosticable est appelée dans ce manuscrit *diagnosticabilisation*.

J'ai choisi cette deuxième option pour deux raisons. La première est la curiosité que m'inspiraient les modèles paramétrés. Je souhaitais donc comprendre l'impact de la modification d'un paramètre du modèle sur le travail que je venais de réaliser durant deux ans. La seconde est que je pense que ce type de méthode peut se combiner au pouvoir fascinant de modélisation des réseaux de Petri temporels. Ces méthodes peuvent donc être intégrées dans une démarche de conception ou de retour sur conception. Ainsi dans une logique de transmission de savoir, ces méthodes s'intègrent parfaitement dans des cours sur les réseaux de Petri temporels pour du contrôle, de l'estimation d'états ou du diagnostic. Ce raisonnement est cohérent avec ma volonté de devenir *enseignant chercheur*, dont les deux missions principales d'enseignement sont :

1. Former les étudiants à des raisonnements complexes sur différents domaines scientifiques afin que ces étudiants soient capables de produire de tels raisonnements dans leur vie professionnelle future
2. Fournir aux étudiants un corpus de connaissances (théoriques et pratiques) qu'ils puissent mobiliser dans leur vie professionnelle future.

J'en arrive donc au titre de cette thèse :

Diagnosticabilité et diagnosticabilisation de motifs temporels dans les réseaux de Petri temporels

Introduction

Positionnement thématique

Cette thèse s'inscrit à l'intersection de deux domaines scientifiques : le *Diagnostic* d'une part, et les *Systèmes à Événements Discrets* d'autre part.

Le *Diagnostic* rassemble les travaux s'intéressant aux raisonnements dont le but est d'identifier une anomalie en se basant sur un ensemble de symptômes retourné par l'objet d'étude de ce raisonnement, qu'il soit un système biologique dans le cas du diagnostic posé par un médecin, ou un système *cyber-physique* dans le cas du diagnostic posé par un automaticien. Dans le cas des systèmes cyber-physiques (que nous appelons *systèmes* dans la suite de ce manuscrit), ces symptômes se manifestent sous la forme d'informations (appelées *observations*) retournées par le système par l'activation de capteurs. Une seule de ses observations n'est en général pas suffisante pour raisonner et ainsi identifier une anomalie. Ainsi plus le nombre d'informations retourné est grand, plus importante est la chance de pouvoir produire un diagnostic correct.

Les *Systèmes à Événements Discrets* [CL08] sont une branche de l'*Automatique* dans laquelle le comportement d'un système est partitionné en un ensemble fini de sous-comportements distincts. La transition d'un comportement à l'autre est modélisée par l'occurrence d'un événement qui peut être observable (l'événement est alors généralement associé à un capteur présent sur le système) ou non observable (dans ce cas aucun capteur présent sur le système ne peut rendre compte de son occurrence). Dans ce type de modélisation ce n'est pas l'évolution d'un signal en fonction du temps (temps évoluant sur $\mathbb{R}_+$) qui est utilisée pour piloter un système, mais la satisfaction de fonctions booléennes par certaines de ces grandeurs depuis un état donné. Les modèles utilisés se découpent en deux grandes catégories : les modèles de type *machines à états* (automates, réseaux de Petri, etc.) dont l'étude emprunte des outils au domaine des *méthodes formelles* tels que la théorie des langages ou les logiques temporelles, et les modèles *max-plus linéaires* [Bac92] qui présentent des similitudes dans leur approche avec l'*automatique continue*.

Un problème de *Diagnostic de Systèmes à Événements Discrets* est donc un problème d'identification d'une anomalie dans des séquences d'événements observables retournées par un système. Étant donné une séquence d'observations ρ_o, on appelle *fonction de diagnostic* une fonction Δ qui associe à cette séquence d'observations l'une des étiquettes suivantes :

— $\Delta(\rho_o) = sauf$ si nous avons la garantie que l'anomalie n'a pas eu lieu pour cette séquence d'observations
— $\Delta(\rho_o) = certain$ si nous avons la garantie que l'anomalie a eu lieu pour cette séquence

d'observations
— $\Delta(\rho_o) = ambigu$ si la séquence d'observations ne permet pas de conclure quant à la présence de l'anomalie.

Il existe deux possibilités pour qu'une fonction de diagnostic retourne *ambigu* : soit il n'y a pas un nombre suffisant d'observations pour conclure (le système n'en a pas produit assez), dans ce cas attendre que le système produise de nouvelles observations permettra de préciser le diagnostic ; soit le modèle est lui-même ambigu, c.-à-d. que les observations sont placées sur le système d'une telle manière que certaines séquences d'observations ne permettent pas de différencier les comportements anormaux des comportements nominaux et quelles que soient les observations produites par la suite, il ne sera jamais possible les différencier. Un système pour lequel toute fonction de diagnostic peut retourner *ambigu* et ce quelle que soit la taille de la séquence d'observations est appelé système *non diagnosticable*. Un système est dit *diagnosticable* pour une anomalie donnée s'il est toujours possible de décider avec certitude pour un nombre suffisant d'observations si l'anomalie a eu lieu. Si un système est diagnosticable pour une anomalie donnée, alors il existe une fonction de diagnostic qui, avec suffisamment d'observations retourne nécessairement *certain* ou *sauf*. Le problème de vérification pour un système de présenter une telle propriété est appelé problème de *diagnosticabilité*. L'étude de cette propriété est le sujet principal de cette thèse.

Dans un problème de diagnostic de système à événements discrets, une anomalie est appelée faute et est dans un premier temps modélisée par un événement non observable [Sam+95] qui fait partie des événements du système étudié. Cette modélisation n'est pas assez fine lorsqu'il s'agit de modéliser des anomalies causées par la conjonction de plusieurs événements. Ces anomalies complexes sont modélisées par des modèles discrets appelés *motifs de supervision* [Jér+06]. Le diagnostic de motifs de supervision est résolu dans la littérature en utilisant des méthodes sujettes à des problèmes d'explosion combinatoire au vu de leur complexité. Pour certains comportements, l'occurrence des événements n'est pas une modélisation assez fine pour rendre compte de la réalité physique de certains phénomènes. L'ajout du domaine temporel permet d'affiner cette modélisation (modélisation d'un retard dans un processus complexe de livraison, incohérence temporelle entre l'acheminement de produits dans une usine et la planification des opérations, etc.).

Cette thèse modélise les systèmes étudiés avec des réseaux de Petri temporels [Mer74], qui sont des modèles de type machine à états auxquels des informations observables sous forme de contraintes de temps ont été ajoutées. Les réseaux de Petri temporels présentent donc deux caractéristiques fortes : la structure logique d'un modèle système à événements discrets d'une part, et l'introduction du temps dense (c.-à-d. le temps continu) à l'intérieur de cette structure d'autre part. Les anomalies étudiées sont une extension des motifs de supervision : des contraintes temporelles sur l'occurrence des événements d'un motif sont ajoutées. Nous appelons ces anomalies *motifs temporels*.

Nous proposons dans un premier temps une analyse de diagnosticabilité pour les motifs temporels dans les réseaux de Petri temporels. Cette analyse présente une condition nécessaire et suffisante de diagnosticabilité, qui montre que le problème est décidable dans notre cadre d'hypothèses. Si une analyse de diagnosticabilité répond que le système n'est pas diagnosticable pour le motif étudié, deux choix se présentent :

1. Synthétiser un diagnostiqueur non certain dont le diagnostic est le plus juste possible.
2. Modifier le modèle du système pour le rendre diagnosticable.

Nous étudions dans un second temps ce deuxième choix, en prenant le parti pris suivant : l'ajout

de capteurs est intrusif et peut être complexe à réaliser pour un nombre conséquent de systèmes, nous allons donc étudier comment rendre un système temporel diagnosticable en modifiant les contraintes temporelles qui lui sont imposées. N'ayant aucune garantie qu'il existe une modification des contraintes temporelles du système permettant de le rendre diagnosticable pour le motif temporel étudié, nous formalisons ce problème appelé *diagnosticabilisation temporelle* et proposons une première approche pour vérifier s'il existe une telle solution pour un motif temporel et un système temporel donné. Ce problème demande donc de comprendre les raisons de non diagnosticabilité d'un système d'une part, et de les expliquer d'autre part. Pour cela il est nécessaire d'avoir une abstraction des modèles étudiés qui permette de synthétiser de telles explications. La construction d'une telle abstraction est proposée comme travail préliminaire aux études de diagnosticabilité et diagnosticabilisation développées dans ce manuscrit.

Contributions

Trois contributions sont proposées dans cette thèse :

1. La première contribution est **une abstraction finie des préfixes d'une taille donnée du langage d'un réseau de Petri temporel**. Une première version de cette abstraction a été présentée au 13e colloque sur la *Modélisation des Systèmes Réactifs* (MSR'21) [CSP21].
2. La seconde contribution est **une condition nécessaire et suffisante et une méthode d'analyse de diagnosticabilité pour les réseaux de Petri temporels et les modèles temporels**. Un travail préliminaire à cette contribution a été présenté au *Workshop on Discrete Event Systems 2022* (WODES'22) [Coq+22].
3. La troisième contribution est **une méthode de vérification de la diagnosticabilisation temporelle d'un réseau de Petri temporel pour un motif temporel**. Un travail préliminaire à cette contribution a été présenté à l'*IFAC World Congress 2023* [CPS23].

Deux autres publications réalisées durant cette thèse présentées à l'*IFAC Symposium on Fault Detection, Supervision and Safety for Technical Process 2022* (SafeProcess'22) [CSP22] et au 33e workshop international sur *Principle of Diagnosis* (DX'22) [Lub+22] sont discutées dans les perspectives de ce manuscrit.

Articulation du document

Le travail réalisé est découpé en 5 chapitres qui abordent chacun un aspect différent du travail réalisé.

Le Chapitre 1 introduit le problème de diagnosticabilité et les prérequis théoriques nécessaires à sa formulation. Ce chapitre présente ensuite les travaux de la littérature traitant de la diagnosticabilité de faute simple pour les réseaux de Petri et les réseaux de Petri temporels, ainsi que les travaux traitant de la diagnosticabilité de motif de supervision. De cette étude bibliographique est tiré un ensemble de constats motivant les directions prises dans la suite des travaux.

Le Chapitre 2 présente formellement les modèles étudiés dans ce manuscrit : les réseaux de Petri temporels. Une abstraction de réseaux de Petri temporels appelée *graphe des classes* est ensuite présentée, et les limites de cette abstraction pour traiter le problème étudié sont mises en lumière. En partant de ces constats, la première contribution de cette thèse est développée : une abstraction finie de préfixes du langage d'un réseau de Petri temporel, appelée *chemin*.

Le Chapitre 3 modélise le problème de diagnostic traité dans le reste de ce manuscrit. Il pose les définitions d'un système, d'un motif temporel, et présente la méthode utilisée pour notifier l'occurrence du motif dans les exécutions du système.

Le Chapitre 4 traite le problème de diagnosticabilité de motif temporel en se basant sur les éléments introduits dans le Chapitre 3. Les exécutions fautives sont abstraites sous forme de chemins (introduits dans le Chapitre 2), dont l'enveloppe temporelle est partitionnée en trois ensembles distincts partageant une propriété commune de diagnostic. En utilisant ces partitions la deuxième contribution de cette thèse est présentée : une condition nécessaire et suffisante de diagnosticabilité pour un réseau de Petri temporel et un motif temporel. Une courte discussion sur une implémentation de méthode de vérification de diagnosticabilité pour des fautes simples temporelles s'appuyant sur ce résultat est proposée en fin de chapitre.

Le Chapitre 5 formalise le problème de diagnosticabilité temporelle en se basant sur une brève étude bibliographique des travaux étudiant deux modèles temporels paramétriques utilisés en vérification formelle : les *automates temporisés paramétriques* et les *réseaux de Petri temporels paramétriques*. Une méthode de vérification de diagnosticabilisabilité est ensuite développée. Cette méthode prend en entrée un réseau de Petri temporel auquel un ensemble de paramètres est ajouté. Cette méthode s'appuie sur l'existence de solutions d'une conjonction d'ensembles de contraintes. S'il existe une valuation de ces paramètres qui satisfait ces contraintes, le système engendré par cette valuation est diagnosticable pour le motif temporel étudié et partage certaines caractéristiques du réseau de Petri temporel initial.

Chapitre 1

Diagnosticabilité dans les réseaux de Petri

La diagnosticabilité est la propriété d'un système pour une faute donnée de ne contenir aucune ambiguïté dans ses observations vis-à-vis de l'occurrence de cette faute. Autrement dit, un comportement contenant cette faute et un comportement ne la contenant pas ne peuvent produire les mêmes observations de manière infinie. C'est une propriété intéressante en diagnostic car elle implique que tout comportement fautif va présenter une *signature* (sous forme d'observations) qui ne peut pas correspondre à un comportement nominal (ou non fautif). Ainsi il est possible pour un système diagnosticable pour une faute donnée de synthétiser un diagnostiqueur certifié dont le résultat avec un nombre suffisant d'observations sera certain si la faute a eu lieu, ou sauf si la faute n'a pas eu lieu.

Ce chapitre propose un historique des travaux publiés sur la diagnosticabilité de deux modèles de Systèmes à Événements Discrets (SED) que sont les Réseaux de Petri (RdP) atemporels et les réseaux de Petri Temporels (PT). Deux types de fautes sont considérés : les fautes simples qui consistent en l'occurrence d'un événement et les motifs qui consistent en l'occurrence de plusieurs événements contraints par leur ordre d'apparition. Ce choix est guidé par le raisonnement suivant : les objets d'étude de ce manuscrit sont les systèmes modélisés par des PT et les motifs temporels. Il est donc nécessaire d'étudier la littérature sur la diagnosticabilité de modèles de type RdP d'une part, et d'étudier les subtilités apportées par l'ajout du domaine temporel dans les RdP.

Il y a peu de travaux traitant de diagnosticabilité comparé au nombre de travaux traitant du diagnostic de RdP et de PT. Ces travaux pour les RdP peuvent être rassemblés en deux types d'approches :

— Les approches posant un problème de programmation linéaire en nombres entiers [BCD08]. Ces approches utilisent les outils proposés par l'algèbre linéaire (représentation matricielle) pour trouver des conditions nécessaires et suffisantes de diagnosticabilité.
— Les approches construisant un graphe permettant de vérifier la propriété [UOO98]. Dans ce type d'approches, les RdP sont vus non pas sous l'angle de leur représentation matricielle, mais comme des machines à états acceptant un langage formel. Ainsi les raisonnements proposés s'appuient sur des intersections de langages, et doivent relever le défi d'abstraire de manière finie les langages infinis générés par les systèmes étudiés. De telles approches vont également utiliser des outils de logique pour raisonner. Ces approches se rapprochent des raisonnements que l'on retrouve dans le domaine de la *vérification formelle*. Nous avons donc choisi de les présenter comme telles (page 15).

Lorsque l'on se penche sur les travaux de diagnosticabilité de PT, le peu de travaux proposés choisissent des approches du deuxième type, car l'ajout du temps dense met en défaut les raisonnements algébriques utilisés pour les RdP. L'étude d'un système à travers son langage temporel devient pour les PT un outil très efficace.

L'organisation du chapitre est la suivante. Dans un premier temps, nous définissons les éléments nécessaires pour décrire un problème de diagnosticabilité. Quelques prérequis de *vérification formelle* utiles pour les analyses développées par la suite sont introduits, et permettent de présenter le travail de référence qui a défini le problème de diagnosticabilité de fautes simples pour les SED, et deux travaux apportant une notion permettant de mieux comprendre pourquoi un système n'est pas diagnosticable : les *paires critiques*. Ensuite nous nous intéressons aux travaux traitant de la diagnosticabilité de fautes simples dans les RdP. Une étude de la littérature sur la diagnosticabilité de fautes simples dans les PT est ensuite proposée. Nous finissons cet état de l'art sur les travaux de diagnosticabilité d'un autre type de fautes, plus complexes, appelées *motifs*, et les difficultés que les motifs apportent lorsque l'on s'intéresse aux PT.

1.1 Introduction à la diagnosticabilité

La notion de diagnosticabilité a été introduite pour la première fois dans [Lin94]. Cette notion a ensuite été reprise dans [Sam+95] qui est l'article de référence pour la communauté des SED. Avant d'aborder cette notion, nous allons tout d'abord présenter quelques prérequis formels indispensables à la compréhension de ce chapitre et du reste de ce manuscrit.

1.1.1 Langages formels et Systèmes de transitions labellisés

Une part importante des études de systèmes s'intéresse aux langages générés par les modèles de ces systèmes. Nous proposons la définition suivante d'un langage formel :

Définition 1. Un langage $\mathcal{L}$ sur un alphabet Σ est un ensemble contenant des séquences de symboles de Σ. Les éléments d'un langage sont appelés des *mots*.

Notation : Par la suite, pour une séquence σ de taille $n \in \mathbb{N}^*$, $\sigma_{|k}$ représente la restriction de σ à ses k premiers éléments ($k < n$). Le mot vide est noté ε. Il correspond à l'élément neutre pour la concaténation pour un langage.

Exemple 1. Considérons l'alphabet $\Sigma = \{a, b\}$. L'ensemble $\mathcal{L}$ composé des éléments $a.b.a.b.b.b$, a, $a.a$, $a.b.b.b$ et b est un langage sur Σ. Parmi les langages sur Σ, notons le langage Σ^*, appelé clôture de Kleene de Σ, qui contient tous les assemblages finis de symboles de Σ possibles ainsi que le symbole vide.

Les langages utilisés par la suite ont la propriété d'être *clos par préfixe* [HU79] : informellement un langage est dit clos par préfixe si pour tout mot w du langage, tout préfixe de w est aussi un mot du langage.

Définition 2. $w_1 \in \Sigma^*$ est un préfixe de $w \in \mathcal{L}$ un langage sur un alphabet Σ si $\exists w_2 \in \Sigma^*$, $w = w_1.w_2$. Un langage $\mathcal{L}$ est clos par préfixe si pour tout préfixe w' de $w \in \mathcal{L}$, nous avons $w' \in \mathcal{L}$. L'ensemble des préfixes d'un mot w est noté $pref(w)$ dans la suite de ce manuscrit.

Un langage formel L est dit *vivant* si tout mot w de $\mathcal{L}$ est le préfixe d'un mot $w' \in L$ avec $w \neq w'$.

Un Système de Transitions Étiqueté [Gor17] (vocabulaire utilisé dans la communauté de la vérification formelle) est un modèle largement utilisé dans les SED (on parle en général de *machine à état* dans la communauté SED, nous choisissons ici de parler de STL, car les travaux de diagnosticabilité étudiés se concentrent sur les labels pour exprimer les problèmes traités et non sur la notion d'état). Un STL est un graphe orienté, et correspond à un automate dont les états sont tous accepteurs, c'est à dire un automate pour lequel chaque séquence menant de l'état initial à un autre état est un mot de son langage (son langage est clos par préfixe).

Définition 3. Un Système de Transition Étiqueté (STÉ) est un triplet $(S, \Sigma, \twoheadrightarrow)$ où S est un ensemble d'états, Σ est un alphabet fini et $\twoheadrightarrow \subset S \times \Sigma \times S$ est la relation de transition.

Remarque. Le langage d'une telle machine est un langage régulier [Aho+00].

Nous utilisons dans la suite de ce chapitre la notion de *structure de Kripke* [Cla+18], qui permet d'ajouter des propositions atomiques dans les états d'un STL.

Définition 4. Soit AP un ensemble de propositions atomiques. Une structure de Kripke labellisée est un quadruplet $(S, \Sigma, \twoheadrightarrow, L)$ où $(S, \Sigma, \twoheadrightarrow)$ est un STL et $L : S \to 2^{AP}$

Une structure de Kripke labellisée est un STL dont les états contiennent des propositions atomiques. L'intérêt d'une telle structure mathématique est de pouvoir vérifier si un modèle satisfait des propriétés dépendantes de la trajectoire empruntée, ce qui passe par une description des différentes transitions. Nous ajoutons des labels aux transitions qui correspondent à des événements du système étudié dans un modèle SED. Les propriétés vérifiées sont généralement exprimées en utilisant des logiques temporelles : *Linear-Tree Logic* aussi notée LTL qui s'intéresse aux événements d'une trajectoire particulière, et *Computational Tree Logic* aussi notée CTL qui s'intéresse à un ensemble de trajectoires. Certains travaux de diagnostic et de diagnosticabilité expriment des propriétés logiques à vérifier pour résoudre le problème traité, cela ne sera pas le cas dans ce manuscrit.

1.1.2 Diagnosticabilité - Genèse et vérification

L'article de référence sur la diagnosticabilité de faute simple dans les SED [Sam+95] a été publié au milieu des années 90. Dans ce travail la diagnosticabilité est associée à la notion de langage : on dit qu'un langage est diagnosticable. Les systèmes étudiés sont modélisés par des STL dont l'ensemble d'événements Σ est partitionné en deux ensembles, Σ_o l'ensemble des événements observables et Σ_u

l'ensemble des événements non observables (on parle alors de système partiellement observable). Les éléments du langage étudié sont appelés *traces*. On appelle projection observable l'opérateur $\mathbf{P}$ qui à une trace t associe les événements observables de t (c.-à-d. les éléments de Σ_o) et supprime les événements non observables (c.-à-d. les éléments de Σ_u). Dans le cas où tous les événements de t sont non observables, nous avons $\mathbf{P}(t) = \varepsilon$. La faute étudiée est modélisée par un événement f non observable ($f \in \Sigma_u$). Deux hypothèses sont posées dans ce travail qui sont indispensables pour pouvoir produire une analyse de diagnosticabilité :

1. Le système ne contient pas de cycles non observables, qui sont des séquences répétables à l'infini de transitions labellisées par des événements non observables. Si un système ne respecte pas cette hypothèse, il est possible qu'il ne produise plus d'observations, il n'est donc pas possible de produire de diagnostic dans un tel cas.
2. Le langage du système étudié est vivant. Les systèmes étudiés n'admettent pas de blocage (c.-à-d. qu'il n'existe pas d'évolution qui mène à un état où le système ne produit plus d'événements). Cette hypothèse, combinée à la première, impose que le système peut toujours produire des observations (on parle dans certains travaux de systèmes *ultimement observables* [Lub+20]).

Ces hypothèses sont utilisées dans la grande majorité des travaux de diagnosticabilité de SED.

Définition 5. Un langage L clos par préfixe et vivant est diagnosticable conformément à une projection $\mathbf{P}$ et à un événement f si $(\exists n \in \mathbb{N}^*)$, $\forall s \in L$, $s = \sigma.f$, $\forall t \in L/s$, $|t| \geq n \Rightarrow (\forall w \in L$, $\mathbf{P}(w)=\mathbf{P}(s.t) \Rightarrow w = w_1.f.w_2)$ où L/s est l'ensemble des séquences d'événements ρ de Σ telles que $s.\rho \in L$.

Informellement, cette définition indique qu'un langage clos par préfixe est diagnosticable pour une faute donnée (et une projection observable) s'il existe un entier n pour lequel, en considérant toute les traces s qui finissent sur la faute f, pour toute continuation t de s de taille supérieure à n, toute trace de L ayant la même projection observable que $s.t$ contient f.

Pour résoudre le problème de diagnosticabilité posé, les auteurs de [Sam+95] proposent de construire une structure de Kripke labellisée appelée *diagnostiqueur* (connu aujourd'hui sous le nom de *diagnostiqueur de Sampath*). Cette structure consiste en une version observable déterminisée du STL du système étudié.

— Dans un premier temps un *observateur* est construit (un STL dont le langage correspond à la projection observable du langage du système étudié). Les états accessibles par une transition labellisée par un événement non observable dans le STL initial sont supprimés, seuls les états accessibles par une trace dont le dernier événement est observable sont conservés dans l'observateur.
— Ensuite, cet *observateur* est déterminisé, c.-à-d. qu'il est transformé en STL déterministe, dont les états sont des ensembles d'états de *l'observateur*. Deux états de l'observateur sont dans un même état du *diagnostiqueur* si les traces permettant d'accéder à ces états dans *l'observateur* sont les mêmes.

Pour un état du diagnostiqueur (qui consiste en une collection d'états), chaque sous état (c.-à-d. l'ensemble des états de l'observateur contenus dans cet état du diagnostiqueur) est labellisé par

F si toute trace du STL initial menant à cet état contient l'événement de faute, N si aucune de ces traces ne contient cet événement, et A si au moins une trace menant à cet état contient la faute et une autre ne la contient pas. Un tel système est diagnosticable pour une faute f s'il n'existe pas de cycle indéterminé. Un cycle indéterminé est une boucle dont tous les états contiennent soit :

— un sous état labellisé F et un sous état labellisé N
— un sous état labellisé A

La complexité de l'algorithme de construction du diagnostiqueur est exponentielle en le nombre d'états du système étudié.

Remarque. L'opération qui permet la construction de l'observateur est appelée *clôture non observable* [Jér+08].

Par la suite nous parlons de système diagnosticable et non de langage diagnosticable, car les langages diagnosticables sont les langages reconnus par les systèmes.

Une autre approche a été proposée quelques années après pour vérifier la diagnosticabilité de tels systèmes et de telles fautes dans [Jia+01]. Dans ce travail il est toujours question de construire une structure de Kripke labellisée sur laquelle il va être possible de vérifier la diagnosticabilité en vérifiant si la structure satisfait une propriété, mais cette structure n'est pas obtenue par déterminisation. Dans ce travail, la structure de Kripke est obtenue par produit synchrone sur les événements observables du STL modélisant le système avec lui même. Les états obtenus sont donc des paires d'états accessibles par la même trace observable. Chaque état d'une paire est étiquetée F si la trace permettant d'y accéder contient l'événement de faute, et N si elle ne le contient pas. Cette structure est appelée *twin-plant* par la suite [PCC02] (ou *verifier* dans [YL02]). Un système est diagnosticable pour une faute donnée si le *twin-plant* associé au système ne contient pas de boucle ambigüe, c.-à-d. de boucle sur un état dont la paire est étiquetée différemment, F pour l'un des éléments et N pour l'autre. Cette méthode présente deux intérêts majeurs :

— Le premier intérêt de la méthode du *twin-plant* par rapport à la méthode du *diagnostiqueur* est sa complexité, qui est polynomiale en le nombre d'états du système étudié.
— Le second intérêt de ces travaux réside dans l'introduction de la notion de *paire critique*. Informellement, une paire critique est une paire de traces infinies (en nombres d'événements), la première contient l'événement de faute, la seconde ne le contient pas, et ces deux exécutions partagent la même trace observable (cette notion est donc indissociable de la faute étudiée). Les auteurs de [PCC02] ont démontré que l'absence de paire critique pour une faute dans les traces du langage d'un système est une condition nécessaire et suffisante de diagnosticabilité d'un système pour cette faute, et ont également démontré que la présence d'un cycle dans le *twin-plant* associé au problème étudié est équivalente à l'existence d'un paire critique.

Pour l'étude de diagnosticabilité proposée dans ce manuscrit, nous nous appuyons sur l'équivalence entre la condition proposée et la présence de paire critique dans les traces du système pour obtenir notre condition de diagnosticabilité.

En conclusion de cette section, nous avons vu que le problème de diagnosticabilité de fautes simples dans les STL peut s'exprimer sous la forme d'une étude du langage fautif observable et du langage non fautif observable d'un système. Malgré le caractère infini de ces langages, le problème de diagnosticabilité est décidable et peut être traité comme un problème de recherche de cycles

dans une structure de Kripke labellisée.

1.2 Réseaux de Petri

Nous allons maintenant nous intéresser aux travaux réalisés sur la diagnosticabilité de fautes simples dans des systèmes modélisés par RdP. Pour cela les RdP [Dia13] sont présentés de manière informelle, puis nous étudions les travaux de diagnosticabilité portant sur de tels modèles.

1.2.1 Présentation informelle des réseaux de Petri

Les RdP sont un formalisme utilisé pour représenter des systèmes concurrents. Introduits par Carl Adam Petri en 1962, les RdP sont utilisés pour leur pouvoir de modélisation par la communauté des SED, notamment pour modéliser des systèmes présentant des évolutions en parallèle, ainsi que pour la simplicité de leur représentation sous forme de graphe. Un RdP est composé d'un ensemble de places (représentées par des cercles) dans lesquelles se trouvent des jetons, et d'un ensemble de transitions (représentées par des carrés) permettant la création et la suppression de jetons dans les différentes places. On dit qu'une place est marquée si elle contient au moins un jeton. De même on appelle marquage une fonction qui à un ensemble de places associe des jetons (ce nombre est nécessairement un entier naturel). Le marquage initial, c'est-à-dire le placement des jetons dans les places avant toute évolution d'un RdP est noté M_0. Des arcs relient des places et des transitions entre elles pour représenter la consommation (lorsqu'un arc part d'une place vers une transition) ou la création de jetons (lorsqu'un arc part d'une transition vers une place). Ces arcs sont éventuellement pondérés par des entiers indiquant le nombre de jetons à consommer ou à créer dans la place concernée. Une transition t est dite sensibilisée si l'ensemble des places reliées à cette transition par un arc entrant dans t sont marquées et si le nombre de jetons dans chaque place est supérieur ou égal à la pondération de l'arc qui relie cette place à t. Lorsqu'une transition est sensibilisée, elle peut être tirée. Lorsqu'une transition est tirée, les jetons nécessaires à sa sensibilisation sont supprimés et les jetons produits par le tir de cette transition sont créés. L'ensemble des places reliées à t par un arc entrant dans t est appelé *précondition* de t et est noté $Pre(t)$. De même, l'ensemble des places reliées à t par un arc sortant de t est appelé *postcondition* de t et est noté $Post(t)$. Un RdP est dit *sauf* si au cours de son évolution aucune place ne contient plus d'un jeton.

Un marquage M est dit accessible s'il existe depuis M_0 une séquence de transitions $\sigma = t_0 \dots t_n$ telle que t_0 est tirable depuis M_0 et qu'il existe un ensemble de marquages $(M_i)_{i\in[1,n-1]}$ avec $M_0 \xrightarrow{t_0} M_1 \xrightarrow{t_1} \dots \xrightarrow{t_n} M$ (à lire *le tir de* t_0 *mène au marquage* M_1 et ainsi de suite). Pour un marquage accessible M, il n'y a pas *a priori* unicité de la séquence de tir permettant de l'atteindre.

D'un point de vue modélisation, les jetons peuvent aussi bien servir à représenter des ressources d'un système automatique (bruts pour une machine à outil, pièces dans un convoyeur, etc.) qu'à représenter l'ensemble des actionneurs d'un système qui sont actifs à un instant donné (dans ce cas les actionneurs du système modélisé sont associés aux différentes places et le marquage d'une place signifie que l'actionneur associé est actif). Un autre intérêt des RdPs est la capacité de leur représentation graphique de pouvoir exprimer explicitement des évolutions parallèles à l'intérieur d'un système. En effet, plusieurs places pouvant être marquées à un même instant, si deux transitions sont sensibilisées par un marquage, et que le tir de l'une ne désensibilise pas l'autre (le tir de la première ne supprime pas les jetons nécessaires au tir de l'autre), pour ce marquage ces deux

transitions évoluent indépendamment l'une de l'autre. Dans le cas contraire on dit que les deux transitions sont en conflit structurel. Les transitions sont associées à des événements subis par les systèmes modélisés. Il n'y a apriori aucune raison que ces événements soient en bijection avec l'ensemble des transitions du modèle associé au système (un même type de capteur peut être présent à deux endroits), ainsi à la structure classique de RdP sont ajoutés un alphabet (noté Σ) dont les symboles correspondent à des événements des systèmes étudiés et une fonction de labellisation (ou d'étiquetage) notée ℓ qui associe à chaque transition un événement (cette fonction est injective).

La Figure 1.1 présente l'exemple d'un système automatique modélisé par un RdP sauf. Ce système est composé de deux ensembles de convoyeurs, et d'un ascenseur reliant ces deux ensembles (places Up et $Down$). Les produits indicés i se déplacent sur les parties des convoyeurs indicées de la même manière, et deux produits $Prod_1$ et $Prod_2$ évoluent en parallèle sans interaction entre eux. Le système étant composé d'un unique ascenseur, l'acheminement des produits d'un ensemble de convoyeurs à un autre ne peut être réalisé simultanément. Il est nécessaire que l'ascenseur soit disponible pour descendre un produit ou l'autre (place $WaitUp$). L'ascenseur dans cette modélisation est une ressource partagée par les différents produits. L'ensemble des événements de ce RdP est l'ensemble des événements (en gras sur la figure) spécifiés sur les différentes transitions : {**Pr1**, **Pr2**, **P1**, **P2**, **EP1**, **EP2**, **U**, **D**, **Req1**, **Req2**, **EPReq1**, **EPReq2**, **ELReq1**, **ERReq2**, **ERReq1**, **ELReq2**}. Dans cet exemple, certaines places modélisent des phases d'attente des ressources ($WaitDown$, $Prod_1Avail_1$, etc.) et d'autres modélisent des actions ($Down$, Up, $Push_2$, etc.).

Lorsque l'on s'intéresse à l'évolution d'un RdP et non pas uniquement à l'accessibilité de certains états, les trajectoires permettant d'accéder à de tels états portent l'information sur la trajectoire suivie par le RdP. Une manière d'étudier les trajectoires est de chercher à représenter le *langage* d'un RdP, c.-à-d. des différentes transitions tirables en séquence. Une telle étude a été réalisée la première fois dans [Hac76].

Exemple 2. Considérons un RdP labellisé (un RdP avec un alphabet Σ et une fonction de labellisation ℓ). L'ensemble des séquences d'événements obtenues en prenant une séquence de transitions correspondant à une évolution du RdP depuis son marquage initial et en remplaçant les transitions par leur labels donnés par la fonction ℓ, ces séquences forment le langage du RdP labellisé.

Comme vu précédemment, il est nécessaire d'avoir une abstraction d'un RdP qui contienne à la fois les marquages du RdP ainsi que ses exécutions possibles. Une telle abstraction a été proposée, appelée *graphe des marquages accessibles* [Dia13] :

Définition 6. Le graphe des marquages accessibles, ou graphe des marquages d'un RdP N de marquage initial M_0 est le STL $(\mathcal{S}, M_0, \mathcal{M}, \twoheadrightarrow)$ tel que :

1. M_0 est le marquage initial
2. $\mathcal{M} : \mathcal{S} \to \mathbb{N}^P$ est la fonction qui associe à chaque état un marquage
3. $\forall (S, S') \in \twoheadrightarrow^2$, $\mathcal{M}(S) \xrightarrow{t} \mathcal{M}(S')$ (depuis le marquage de S, il existe une transition t sensibilisée telle que le tir de t amène au marquage de S')

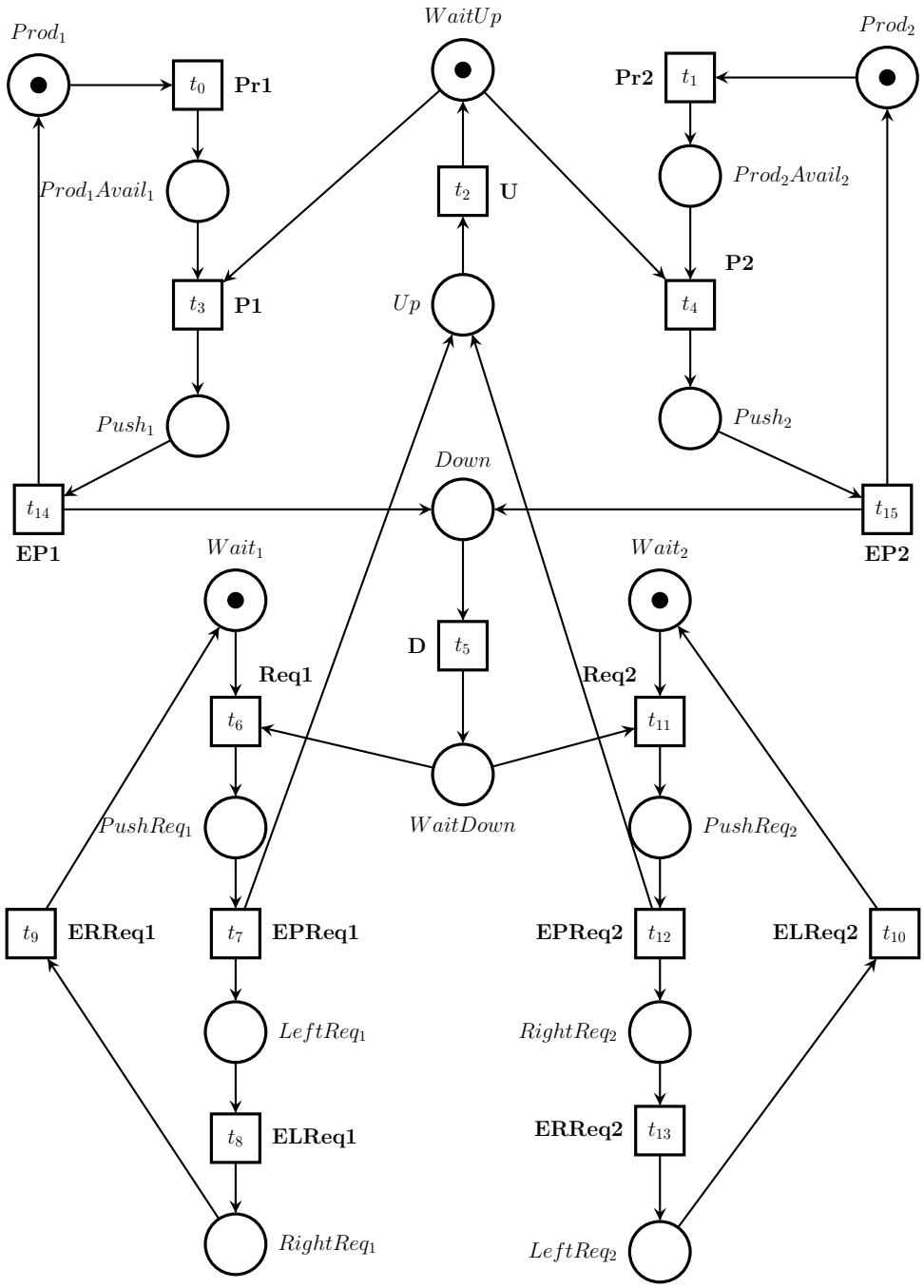


FIGURE 1.1 – Représentation graphique d'un RdP : réseau de transport [GPS17]

Un état du graphe des marquages correspond à un marquage accessible du RdP depuis M_0, c.-à-d. un marquage pour lequel il existe une séquence de transitions tirable qui partant de M_0 amène à ce marquage. Le langage d'une telle machine correspond à l'ensemble des séquences de transitions tirables dans le RdP. Comme indiqué précédemment, ce langage est clos par préfixe, ce qui implique que le langage reconnu par le graphe des marquages correspond à l'ensemble des exécutions du RdP.

Les RdP peuvent aussi être représentés sous forme matricielle. Pour cette représentation matricielle un marquage est représenté par une matrice contenant une colonne et autant de lignes qu'il y a de places dans le réseau, et les préconditions et postconditions (notées Pre et $Post$ généralement) sont représentées par des matrices ayant autant de lignes que de places et autant de colonnes que de transitions. Une matrice généralement notée C égale à $Post - Pre$ permet de calculer des marquages accessibles pour une séquence de transitions donnée. Nous ne développerons pas plus ici cette représentation qui ne sera pas utilisée dans les chapitres suivants de ce manuscrit car son utilisation n'est pas vraiment adaptée aux PT, mais une telle représentation est utilisée dans un nombre conséquent de travaux de diagnosticabilité de RdPs.

1.2.2 Diagnosticabilité de Réseaux de Petri

Le premier travail s'intéressant à la diagnosticabilité de fautes simples dans les RdP [UOO98] a été publié peu de temps après l'article introduisant le *diagnostiqueur de Sampath*. Ce travail propose de construire un diagnostiqueur équivalent pour les RdP en utilisant le *graphe des marquages*. À la différence de l'étude proposée dans [Sam+95], ce sont les marquages qui sont partiellement observables et non les transitions, mais cela ne change pas les idées générales permettant de résoudre le problème. Nous retrouvons les hypothèses de vivacité du langage du système étudié et d'absence de cycle non observable (adaptée sur les marquages pour ce travail). Une hypothèse supplémentaire est posée par rapport aux travaux modélisant leurs systèmes par des STL : le nombre de jetons dans les places des RdP étudiés est borné, c.-à-d. qu'il ne peut excéder un entier fixé. Cette hypothèse a comme conséquence que le graphe des marquages du RdP étudié est fini, le diagnostiqueur est donc calculable en temps fini. Par la suite nous concentrons notre étude sur les travaux dans lesquels ce sont les transitions qui sont partiellement observables.

Les méthodes de vérifications de diagnosticabilité se découpent en deux grandes catégories : les méthodes de programmation linéaire en nombres entiers [Lef14] (PLNE), et les méthodes de type *model-checking* (appelées aussi méthodes basées graphes dans [Bas14]) qui consistent à construire une structure de Kripke labellisée sur laquelle va être vérifiée une propriété suffisante ou nécessaire et suffisante pour la diagnosticabilité.

Pour les méthodes par PLNE, certaines se basent sur l'existence de T-invariants [WLJ05] qui sont des séquences de transitions menant d'un marquage à lui-même. En utilisant la représentation matricielle, un T-invariant est une séquence σ telle que $C.\bar{\sigma} = 0$ où $\bar{\sigma}$ est le vecteur dont les coefficients correspondent au nombre de fois où une transition t apparaît dans σ. Une autre approche utilisant les PLNE proposée dans [BCD08] propose une condition suffisante de diagnosticabilité s'appuyant sur la notion de *g-marquages*, qui sont des vecteurs de marquages admettant des coefficients négatifs représentant le fait que le marquage est accessible par le tir d'une transition non observable. Cela leur permet de définir des *explications non observables* sensibilisées par un g-marquage, qui sont à la base de leur condition suffisante. Les mêmes auteurs ont proposé quelques années plus tard un travail utilisant un PLNE pour résoudre le problème de K-diagnosticabilité

dans les RdPs [BCD12]. Dans ce travail l'objectif est de trouver s'il existe un entier K pour lequel la condition de diagnosticabilité est vérifiée pour toute trace fautive du système étudié. Un travail récent propose une méthode pour calculer la borne K minimale de diagnosticabilité [CGB23].

En parallèle de ces travaux d'autres approches de diagnosticabilité (de type *model-checking*) synthétisant des structures de Kripke ont été proposées. Nous pouvons citer en particulier [JB10] dans lequel les auteurs abstraient un RdP partiellement observable sous forme d'un automate déterministe (en opposition au graphe des marquages non déterministe dans ce travail car les transitions ont été remplacées par leurs labels) basé sur les principe d'explications minimales . La méthode du *verifier* a été également adaptée aux RdPs dans [Cab+09], en construisant le produit synchrone du système avec une copie de lui-même synchronisé sur les observations (appelé *verifier net*), et en raisonnant sur le graphe des marquages du *verifier net* ainsi créé. Ainsi dans ces deux travaux l'existence d'un cycle indéterminé dans l'une ou l'autre des structures calculées est équivalent à la non diagnosticabilité du système étudié. Récemment un travail [LKT20] propose des règles de réduction d'un RdP pour réduire dans certains cas le temps de calcul pour vérifier la diagnosticabilité avec différentes méthodes de l'état de l'art. Ces règles de réduction permettent de supprimer des transitions non observables, et certaines transitions observables (appelées *transitions observables labellisées exclusivement*) dont l'observation ne participe pas à créer d'ambiguïtés dans les exécutions du système étudié.

Un article de la littérature se propose de comparer les deux types d'approches [Bas+18] : l'approche PLNE proposée dans [BCD12], et une approche *basée graphe* appelée *diagnostiqueur semi-symbolique* utilisant les *diagrammes de décision binaire* pour représenter les marquages et permettant un calcul et une vérification de diagnosticabilité *à la volée* (le calcul de la structure complète hors ligne n'est pas nécessaire pour vérifier la diagnosticabilité avec cette méthode).

Enfin nous pouvons citer deux derniers travaux sur la diagnosticabilité de RdP proposant des approches ne rentrant pas dans la classification proposée précédemment : un travail démontrant la décidabilité de la diagnosticabilité de RdP non bornés [YL17] (les auteurs démontrent que la complexité du problème est EXSPACE-complet), et [Haa+03] dans lequel les auteurs définissent une notion de faible et forte diagnosticabilité, et trouvent une propriété équivalente basée sur les ordres partiels et les dépliages.

1.3 Diagnosticabilité de réseaux de Petri temporels

L'ajout du temps dense dans les systèmes amène une modélisation plus fine des systèmes étudiés. Mais le temps dense amène également une difficulté supplémentaire : pour une trace atemporelle, il existe une infinité de traces temporelles présentant les mêmes événements. Le premier article traitant de diagnosticabilité de modèles temporels a été publié en 2002 par Stravos Tripakis [Tri02]. Cet article s'intéresse à des systèmes modélisés par des automates temporisés [AD94] : un ensemble d'horloges est associé à un état et les différentes transitions contiennent des contraintes sur les valeurs de ces horloges (appelées *gardes*). Cet article définit la notion de Δ-diagnosticabilité (en écho de la K-diagnosticabilité) : un système temporel est Δ-diagnosticable si pour toute exécution fautive, les informations observables du système permettent de conclure que la faute a eu lieu au plus tard Δ unités de temps après l'occurrence de la faute. Une faute est modélisée par un événement non observable (les hypothèses de vivacité du langage du système et de clôture par préfixe ne sont pas formulées). La vérification de diagnosticabilité est effectuée sur un produit synchrone d'une copie du système avec une autre copie dans laquelle la faute est supprimée. Dans cet article apparaît

la notion de trace *Zénon* qui est une trace de durée finie dans laquelle une infinité de transitions sont exécutées (cette notion sera présente dans tous les travaux sur des systèmes temporels). La propriété prouvée équivalente à la diagnosticabilité est le fait que toute trace fautive du produit (c.-à-d. toute trace pour laquelle l'événement de faute apparaît dans la copie du système contenant cet événement) soit *Zénon*. Une preuve de complexité montre que le problème de Δ-diagnosticabilité dans les AT est *PSPACE-complet*. Pour faire le lien avec les travaux précédents, cette méthode est qualifiée de méthode de *model-checking*, car elle consiste à vérifier une propriété sur un produit synchrone qui est une structure de Kripke labellisée. Une étude concernant la décidabilité et la complexité de la diagnosticabilité pour certains types d'automates temporisés (AT) a ensuite été proposée dans [BCD05].

Les réseaux de Petri Temporels (PT) sont une extension des RdPs dans laquelle des intervalles de temps (appelés intervalles statiques) sont associés aux différentes transitions. Une transition sensibilisée peut être tirée si le temps écoulé depuis sa sensibilisation est supérieur à la borne inférieure de l'intervalle qui lui est associé, et doit être tirée si le temps écoulé depuis sa sensibilisation atteint la borne supérieure de son intervalle statique (une formalisation des PT sera présentée dans le Chapitre 2). Une telle sémantique est appelée *sémantique forte*. Une abstraction des PT appelée le Graphe des Classes (GC) a été introduite dans [BM83]. Le GC consiste en un STL pour lequel un polyèdre est associé à chacun de ses états (ces états sont appelés des classes pour le GC, ce qui explique l'origine du nom). Une telle structure est appelée *Système de Transitions Temporel* (STT). Ces polyèdres représentent le domaine de tir des différentes transitions sensibilisées (pour le marquage associé à la classe considérée), c.-à-d. les dates de tir possibles depuis cette classe pour une transition sensibilisée (une formalisation du GC est présentée dans le Chapitre 2). Le GC peut être considéré comme une structure de Kripke labellisée. Cela permet notamment de traiter le problème *d'accessibilité* dans les PT.

Les travaux traitant de la diagnosticabilité de fautes simples reposent tous sur la synthèse d'un diagnostiqueur en partant du GC du système, l'utilisation de la représentation matricielle associée n'étant plus aussi efficace. Nous pouvons citer un premier travail ne traitant pas directement de la diagnosticabilité de fautes simples pour des systèmes modélisés par des PT partiellement observables [GTY09], mais dont l'approche est assez similaire à celle pour traiter un tel problème. L'objectif premier de cet article n'est pas la diagnosticabilité, mais de construire un estimateur d'état. Cependant l'estimateur proposé peut être utilisé pour faire de la vérification de diagnosticabilité, et sa construction est caractéristique des travaux de diagnosticabilité de fautes simples dans des systèmes modélisés par des PT. Dans ce travail les auteurs proposent à partir du GC du système étudié de synthétiser un autre graphe ; appelé *estimateur*. Un état de cet estimateur est un ensemble de classes partitionné en deux : des classes appelées classes d'entrée qui sont des classes accessibles en tirant une transition observable (labellisée par un événement observable), et des classes appelées classes cachées qui sont accessibles depuis les classes d'entrée par le tir d'une ou plusieurs transitions non observables. L'obtention des classes cachées s'apparente à une opération de clôture non observable. Ainsi le langage accepté par cet estimateur correspond à un sous-ensemble du langage observable atemporel du système (c.-à-d. les séquences d'événements que l'on peut observer sur le système sans leurs temporisations), et les différents nœuds du graphe engendré correspondent aux classes dans lesquelles le système peut se trouver après avoir vu une séquence d'observations menant de l'état initial à l'état considéré. Cette approche peut être enrichie par un étiquetage $\{N, F\}$ des classes des différents nœuds de cet estimateur en posant un événement de faute f, ce qui le transforme en structure de Kripke labellisée et permet par la suite de vérifier l'existence de cycles indéterminés (attention vérifier si un nœud peut être atteint par le tir d'une

transition fautive revient à regarder si les nœuds qui le précèdent directement contiennent une classe cachée labellisée par F). Dans [LGT14] les auteurs définissent le *Graphe des Ensembles de Classes Augmenté* (GECA) qui peut être vu comme un diagnostiqueur (STL déterministe obtenu par clôture non observable) augmenté d'informations temporelles sur l'occurrence des événements. L'ensemble des états successeurs d'un état dépendent de l'événement observé ainsi que de ses dates d'occurrence possibles. Cela permet de raffiner le diagnostic en ajoutant le temps aux séquences d'observations. Une proposition similaire de synthèse de graphe a été faite dans [WMS15] où les auteurs définissent le *Graphe de Diagnostic de Fautes* (GDF). À la différence du GECA, les arcs du GDF sont étiquetés par les séquences de transitions menant d'un état à un autre (par clôture, deux états sont en relation dans ce graphe s'il existe une séquence de transitions non observables terminant par une transition observable menant d'un état à un autre dans le GC du système). Un algorithme est proposé pour calculer les dates de tir au plus tôt et au plus tard de la transition observable terminant une séquence, Cet algorithme calcule l'enveloppe temporelle de cette séquence sous forme d'un polyèdre, et réduit l'ensemble de contraintes associé pour obtenir les bornes des dates d'occurrence de la transition observable de la séquence. Il est important de noter que l'objectif du GDF est de faire du diagnostic et non de la diagnosticabilité, nous citons cette approche ici car la méthode proposée peut être adaptée pour résoudre un problème de diagnosticabilité. Le GECA et le GDF peuvent être vus comme des structures de Kripke labellisées. Le dernier travail sur la K et la Δ diagnosticabilité de fautes simples dans les PT a été présenté dans [BCS17]. Ce travail se base sur le *Graphe des Classes Modifié* [1] [BCS15a] (GCM). Le GCM est une copie du GC auquel ont été ajoutés sur chaque arc la transition concernée, son label, et son domaine de tir. La vérification de la K et de la Δ diagnosticabilité est réalisée en résolvant un PLNE pour chaque problème.

Les travaux présentés ici traitent le problème de diagnosticabilité en deux temps. En effet, nous avons pu voir que les méthodes proposées commencent par considérer une abstraction des PT sous forme d'un STL (ne considèrent que l'aspect STL de l'abstraction dans un premier temps, le GC cachant les aspects temporels à l'intérieur des classes), et rajoutent ensuite dans une seconde analyse une sur-couche temporelle. Une telle méthodologie est mise en défaut lorsque l'on intègre des contraintes de temps dans les comportements fautifs, car il devient nécessaire de *synchroniser* le système et la faute étudiée pour vérifier l'occurrence de l'événement fautif d'une part, et la satisfaction de la contrainte de temps imposée d'autre part.

1.4 Motifs de fautes

Les articles explorés dans les sections précédentes s'intéressent à la diagnosticabilité de fautes simples. Une faute est modélisée dans ces travaux par un événement non observable labellisant au moins une transition du système étudié. Des modèles de fautes plus complexes ont été étudiés dans certains travaux de la littérature : les *motifs de supervision* (que nous appelons *motifs* dans le reste de ce manuscrit). Introduits dans [Jér+06], les motifs sont des fautes complexes composées de plusieurs événements. Modélisés par des automates (un motif est donc associé à un langage dont les éléments sont des séquences d'événements non observables), un motif apparaît dans une trace (nous parlons d'occurrence du motif) s'il est possible d'extraire de cette trace une suite d'événements qui appartient au langage du motif.

Exemple 3. Considérons un motif Ω produisant le langage $L = \{w_1 = e_1.e_2, w_2 = e_1.e_1, w_3 =$

1. Une implémentation en Matlab est disponible ici : https://github.com/luigisvideos/wodes22 [BF22]

$e_1.e_2.e_3\}$, et considérons la séquence suivante : $\rho = e_1.o_1.o_2.e_1.o_1.e_2.o_1$. Nous pouvons extraire de ρ les deux suites d'événements $\rho_1 = e_1.e_2$ (obtenue en extrayant le premier et sixième événement de ρ) et $\rho_2 = e_1.e_1$ (obtenu en extrayant le premier et le quatrième événement de ρ). Nous avons $\rho_1 = w_1$, le motif Ω apparait donc dans la séquence ρ (de même $\rho_2 = w_2$). Considérons maintenant la séquence $\rho' = e_1.o_1.o_2.e_3.o_3$. Le motif Ω n'apparaît pas dans ρ', car il n'est pas possible d'extraire de suite d'événements formant un élément de L (aucune extraction n'est complète).

Remarque. L'exemple présenté ici permet de relever une hypothèse importante sur le langage d'un motif : ce langage ne contient pas la séquence vide ε. En effet si ε appartient au langage du motif, ε peut être extrait de toute trace, et ainsi la notion d'occurrence du motif n'a plus de sens.

La définition de diagnosticabilité pour les fautes simples est étendue aux motifs : un système est diagnosticable pour un motif donné si pour toute trace contenant le motif, il existe un entier naturel n pour lequel cette trace se distingue de toutes les traces ne contenant pas le motif au moins n événements après l'occurrence du motif dans cette trace. Dans [Jér+06], le système est modélisé par un STL. La solution proposée consiste à vérifier l'existence de cycles indéterminés sur une structure de Kripke labellisée obtenue en effectuant :

1. le produit synchrone du système avec le motif noté G_Ω
2. la clôture non observable de G_Ω notée $OBS(G_\Omega)$
3. le produit de $OBS(G_\Omega)$ avec lui même

1.4.1 Diagnosticabilité de motifs

Peu de travaux ont été réalisés sur la diagnosticabilité de motifs dans les RdP et les PT : nous en avons recensé deux. Le premier [GPS17] (un système est modélisé ici par un RdP) modélise les motifs sous la forme d'un RdP. Il introduit la notion de *matching* pour notifier l'occurrence du motif dans une trace (cette notion sera définie formellement dans le Chapitre 3), et propose de résoudre le problème de diagnosticabilité de motif en construisant une structure de Kripke suivant la logique de [Jér+06] :

1. Premièrement le produit synchrone du RdP du système et de celui du motif est réalisé, noté $\Theta \times \Omega$
2. Ensuite, le produit synchrone de $\Theta \times \Omega$ avec lui même synchronisé sur les événements observables est calculé (ce qui, transformé en STL correspond au twin-plant du problème).

Pour vérifier la diagnosticabilité les auteurs proposent une formule LTL qui est équivalente à une recherche de paires critiques dans le twin-plant du problème.

Le second travail [PS21] porte sur des systèmes modélisés par des PT. Les motifs sont ici modélisés de la même manière que dans le travail précédent : ce sont des RdP. Les systèmes modélisés n'admettent pas d'exécutions *Zénon*. Cet article soulève un problème de taille (et se trouve à l'origine des problèmes traités dans ce manuscrit de thèse) : le produit synchrone de deux PT n'est pas réalisable pour des intervalles dont l'intervalle statique n'est pas de la forme $[0, +\infty[$. Le problème est lié à la règle de sensibilisation des transitions : dans un PT, une transition est sensibilisée si l'ensemble des places la précédant sont marquées (modulo les pondérations des

arcs) ; à cet instant l'horloge associée à la transition est lancée et la transition peut être tirée si l'horloge atteint un temps qui appartient à son intervalle statique, et cette horloge est supprimée si la transition est tirée ou si la transition n'est plus sensibilisée. Ainsi les horloges associées aux différentes transitions ne sont pas synchronisables. Pour contourner ce problème les auteurs de l'article proposent un produit alternatif utilisant la notion de priorités [2] au niveau des transitions. En utilisant ce produit alternatif la démarche de l'article précédent est suivie : construire un twin-plant pour lequel une formule LTL permet de vérifier l'existence de paires critiques dans les traces du système. Les méthodes de ces deux articles ont été implémentées en utilisant le model-checker SE-LTL qui est un outil du logiciel TINA [BRV04] [3].

Au démarrage de cette thèse aucun article ne mentionnait un travail sur des *motifs temporels*. Un motif temporel est un motif pour lequel des contraintes de temps ont été rajoutées sur l'occurrence des événements de celui-ci. Le langage généré par un tel modèle est un langage temporel composé de séquences de paires (d, e) où d est un réel positif et e un événement non observable. Un travail a été présenté depuis le démarrage sur la diagnosticabilité de motifs temporels pour des AT [LLL23]. Cet article modélise un système par un AT avec Horloge unique (ATH) labellisé déterministe, dont le langage observable est vivant, et n'admettant pas d'exécutions *Zénon*. Les motifs temporels sont modélisés par des ATH déterministes. Les auteurs proposent de construire un diagnostiqueur en utilisant des *automates à tic* pour décomposer les contraintes de temps d'un ATH et ainsi pouvoir faire le produit synchrone entre le système et le motif. Vérifier la diagnosticabilité d'un tel système pour un motif temporel peut être réalisé en cherchant les cycles indéterminés dans le diagnostiqueur proposé.

En conclusion de ces lectures, les méthodes proposées pour la diagnosticabilité de motifs sont réalisées en faisant un produit synchrone d'un système avec le motif. Ce produit n'est pas réalisable pour deux PT, et demande l'introduction de priorités pour fonctionner. Ce manuscrit s'intéresse à des systèmes et des motifs modélisés par des PT : il n'est pas possible de résoudre notre problème en appliquant une méthode de l'état de l'art. Le besoin de proposer une alternative au produit synchrone s'impose.

1.4.2 TWINA

Récemment une proposition de produit de PT a été proposée. Le but de ce travail est de construire un PT dont le langage correspond à l'intersection du langage de deux PT. Ce nouveau type de PT, appelé *réseaux de Petri Temporels Produit* (PTPr) est réalisé en *forçant* deux transitions devant se synchroniser à être tirées en même temps [Lub+19]. Il a été appliqué pour calculer le produit synchrone entre un PT et un motif atemporel [Lub+20] (modélisé par un RdP), et produit un twin-plant de manière plus efficace que [PS21] La vérification de la diagnosticabilité est réalisée en vérifiant la validité d'une formule LTL dans le twin-plant : le produit du système PT avec le motif RdP, puis synchronisation de ce produit avec lui-même sur les observables. Un logiciel appelé TWINA [4] permet d'effectuer ce produit et de travailler avec des PTPr.

Le désavantage de TWINA est qu'il ne permet pas de faire le produit synchrone entre un système PT et un motif PT : les seuls motifs temporels traitables avec les PTPr sont ceux pour lesquels les contraintes de temps du motif sont réduites à des intervalles ponctuels.

2. Cette notion ne sera pas développée ici.
3. https://projects.laas.fr/tina/index.php
4. Vous pouvez retrouver TWINA ici :https://projects.laas.fr/twina/

1.5 Conclusion

Ce chapitre met en lumière que le problème de diagnosticabilité de fautes dans des systèmes modélisés par des SED peut être vu comme l'étude de l'intersection entre les langages fautifs observables (c.-à-d. la projection observable des traces qui contiennent la fautes) et du langage non fautif observable. Ce problème peut être résolu en recherchant des cycles dans une structure de Kripke labellisée, cette structure étant dans certains travaux un *diagnostiqueur*, et dans d'autres travaux un *twin-plant*. L'existence de cycles dans un twin-plant montre l'existence de *paires critiques* dans les traces d'un système.

Nous avons également vu que le problème de diagnosticabilité de motifs est traité en effectuant le produit synchrone entre un système et le motif étudié. Cependant ce produit n'est pas possible dans le cas général pour des motifs modélisés par des PT. Il est donc nécessaire de proposer une nouvelle méthode qui permet de contourner le produit synchrone de deux PT.

Il ressort de ce chapitre que les paires critiques sont des éléments qui intrinsèquement contiennent des explications de la non diagnosticabilité d'un système pour une faute. Un objectif de ce travail de thèse étant d'étudier les conditions selon lesquelles un système peut être rendu diagnosticable, il apparaît naturel que par la suite nous analysions les raisons de l'existence des paires critiques afin d'expliquer la non diagnosticabilité.

Chapitre 2

Réseaux de Petri temporels et abstractions

Le Chapitre 1 a mis en évidence le besoin de développer une approche liant à la fois la structure logique et la partie temporelle des modèles utilisés. Nous avons également mis en lumière la nécessité d'une abstraction permettant de travailler avec le langage temporel reconnu par un réseau de Petri temporel. Dans ce chapitre, nous allons tout d'abord présenter les réseaux de Petri temporels [Mer74]. Une abstraction des PT sous forme de Système de Transitions Temporel (STT) connue dans la littérature, appelée Graphe des Classes [BM83], sera ensuite présentée. Après avoir montré une limite de cette abstraction pour abstraire le langage temporel d'un PT, la première contribution de cette thèse sera développée, qui consiste en une abstraction des préfixes du langage temporel sous forme de polyèdres.

2.1 Réseaux de Petri Temporels

Les PT se démarquent des RdP par l'ajout du domaine temporel pour contraindre le comportement de la structure RdP initiale. Les PT ont été introduits dans [Mer74]. C'est une extension des RdP pour laquelle un intervalle de temps (appelé intervalle statique) est associé à chaque transition. Dans la modélisation d'un SED, chaque transition est associée à un événement du système modélisé. Pour cela la structure est également augmentée avec une fonction de labellisation qui associe à chaque transition un symbole.

2.1.1 Réseaux de Petri Temporels Sauf Étiquetés :

Dans ce manuscrit nous ne nous intéressons qu'aux PT saufs. En effet, le multi marquage impose une discussion complexe quant à la sémantique des PT qui n'est pas l'objectif premier de ce travail.

Définition 7. Un Réseau de Petri Temporel Étiqueté Sauf (PTÉS par la suite) est un 6-uplet $N = \langle P, T, A, \Sigma, \ell, I_s, M_0 \rangle$ où :

- P est un ensemble de places
- T est un ensemble de transitions
- $A \subset P \times T \cup T \times P$ est un ensemble d'arcs reliant les places et les transitions
- Σ est un alphabet

- $\ell : T \to \Sigma$ est une fonction de labellisation
- $I_s : T \to I_{\mathbb{Q}_+}$ est la fonction d'intervalle statique, qui associe à chaque transition un intervalle à bornes rationnelles positives (notées $\downarrow(I(t))$ pour la borne minimale et $\uparrow(I(t))$ pour la borne maximale)
- tout marquage M du réseau est une fonction de l'ensemble des places vers $\{0,1\}$ ($M : P \to \{0,1\}$)
- M_0 est le marquage initial du réseau

La sémantique utilisée par la suite est une sémantique forte (parfois appelée sémantique *à la Berthomieu* [BD91]), c.-à-d. que le tir des transitions suit une règle de tir *au plus tôt* (lorsqu'une borne supérieure d'un intervalle statique est atteinte, une transition doit être tirée).

Le *preset* d'une transition t est défini comme suit $pre(t) = \{p \in P | (p,t) \in A\}$. De manière similaire, le *postset* d'une transition est $post(t) = \{p \in P | (t,p) \in A\}$.

Pour un marquage M, une transition $t \in T$ est tirable à la date θ si et seulement si :

- t est sensibilisée ($\forall p \in pre(t), M(p) = 1$)
- $\theta \in I(t)$ où $I(t)$ correspond au dates de tir au plus tôt et au plus tard de t selon le temps écoulé depuis sa sensibilisation ou à $I_s(t)$ si t est nouvellement sensibilisée, et pour tout $t' \neq t$ sensibilisée par M, $\theta \leq \uparrow(I(t'))$

Pour un Réseau de Petri atemporel, la notion de marquage suffit à définir son état, en cela que la connaissance de la structure et des places marquées par des jetons est suffisante pour déterminer ses évolutions futures. Lorsque que l'on rajoute le domaine temporel, le marquage ne suffit plus. Il est nécessaire de représenter l'état temporel du réseau pour que l'information soit complète. Ainsi, un état d'un PTÉS est un couple $S = \langle M, I \rangle$, où M est le marquage courant et I est la fonction partielle d'intervalle de tir ($I : T \to I_{\mathbb{Q}_+}$) qui associe à chaque transition t un intervalle de tir de $I_{\mathbb{Q}_+}$ dans lequel t peut être tirée dès lors qu'elle est sensibilisée. $S_0 = \langle M_0, I_0 \rangle$ est l'état initial du réseau où pour toute transition t sensibilisée par M_0, $I_0(t) = I_s(t)$, et pour toute autre transition t', $I_0(t') = \emptyset$.

Le tir d'une transition t à une date θ $\langle M, I \rangle \xrightarrow{\theta t} \langle M', I' \rangle$ est tel que :

- $\forall p \in pre(t) \setminus post(t)$, $M'(p) = 0$, $\forall p \in post(t) \setminus pre(t)$, $M'(p) = 1$, sinon $M'(p) = M(p)$
- pour chaque transition $t' \in T, t' \neq t$ sensibilisée par M et toujours sensibilisée par M', $I(t') = [a,b] \Rightarrow I'(t') = [\max(0, a-\theta), b-\theta]$
- pour chaque transition t' sensibilisée par M', et pour chaque transition désensibilisée et ressensibilisée par le tir de t (dans le cas de boucles [1]), $I'(t') = I_s(t')$

Par la suite, on note :

- $(\mathbb{R}_+ \times \Sigma)^*$ l'ensemble des séquences de couples (d,e) où d est un réel positif et e un symbole de l'alphabet Σ
- $(0, \varepsilon)$ la séquence vide de durée nulle
- $(\mathbb{R}_+ \times \Sigma)^+ = (\mathbb{R}_+ \times \Sigma)^* \setminus \{(0,\varepsilon)\}$

Un état S est dit *accessible* dans un PTÉS s'il existe une séquence de transitions $t_0 \dots t_n$, un ensemble de dates $\{\theta_0 \dots \theta_n\}$ et un ensemble d'états $\{S_1, \dots S_n\}$ tels que $S_0 \xrightarrow{\theta_0 t_0} S_1 \xrightarrow{\theta_1 t_1}$

1. Il est possible que le tir d'une transition t conduise à un marquage ressensibilisant cette même transition t. Elle est donc sensibilisée dans les deux marquages, mais son tir réinitialise son horloge.

$\ldots S_n \xrightarrow{\theta_n t_n} S$. La séquence $r = \theta_0 t_0 \ldots \theta_n t_n$ est appelée *exécution* de N. L'ensemble des états accessibles d'un PTÉS N est noté $R(N, S_0)$.

De manière duale, une exécution $r = \theta_0 t_0 \ldots \theta_{n-1} t_{n-1}$ est dite *admissible* pour N s'il existe $\{S_1, \ldots S_n\} \in R(N, S_0)^n$ tels que $S_0 \xrightarrow{\theta_1 t_0} S_1 \xrightarrow{\theta_1 t_1} \ldots \xrightarrow{\theta_{n-1} t_{n-1}} S_n$. Une *trace* sur un alphabet Σ est une séquence de couples $(d_0, e_0) \ldots (d_n, e_n) \in (\mathbb{R}_+ \times \Sigma)^n$ où $d_{i, i\in[1,n]}$ correspond à la date d'occurrence de l'événement e_i relativement à l'occurrence de l'événement e_{i-1}. Pour un PTÉS, une trace correspond à une exécution pour laquelle chaque transition a été remplacée par son label. Autrement dit l'exécution $r = \theta_0 t_0 \ldots \theta_n t_n$ produit la trace $\rho = d_0 e_0 \ldots d_n e_n$ avec :

— $\forall i \in [0, n], \theta_i = d_i$
— $\forall i \in [0, n], e_i = \ell(t_i)$

Par la suite nous conservons la notation θ_i pour les exécutions et les traces. Une exécution produit ainsi une unique trace (la fonction de labellisation est injective), mais il est possible qu'une trace soit produite par plusieurs exécutions (un ensemble de transitions peut partager le même label). La définition de la fonction de labellisation ℓ est étendue de la manière suivante :

- $\ell(\theta_0 t_0 \ldots \theta_n t_n) = \theta_0 \ell(t_0) \ldots \theta_n \ell(t_n)$
- La définition est également étendue aux séquences de transitions non temporisée $t_0 \ldots t_n$: $\ell(t_0 \ldots t_n) = \ell(t_0) \ldots \ell(t_n)$

Remarque. La sémantique décrite est telle qu'il n'existe pas de blocage dû au temps dans un PTÉS, c.-à-d. qu'il n'existe pas de situation où une transition est sensibilisée par un marquage mais ne sera jamais tirable car le temps écoulé depuis sa sensibilisation est supérieur à sa date de tir au plus tard.

Un PTÉS est utilisé pour modéliser un système dans les chapitres suivants. Un système produit les événements qui sont associés aux différentes transitions, sachant qu'un événement (observable ou non) peut être associé à deux transitions (la fonction de labellisation n'est pas bijective). Cela implique que deux exécutions différentes produisant la même trace seront considérées d'un point de vue langage comme équivalentes. Ce choix est guidé par les méthodes de vérification qui s'appuient sur la comparaison de ces traces. Le langage d'un PTÉS est défini ici de la manière suivante :

Définition 8. Le langage $\mathcal{L}(N)$ d'un PTÉS N est l'ensemble des traces ρ telles qu'il existe une exécution $r = \theta_0 t_0 \ldots \theta_n t_n$ admissible de N avec $\rho = \theta_0 \ell(t_0) \ldots \theta_n \ell(t_n)$.

$$\mathcal{L}(N) = \{\rho | \exists (r, S) \in (\mathbb{R}_+ \times T)^n \times R(N, S_0), S_0 \xrightarrow{r} S \wedge \rho = \ell(r)\}$$

Toute trace w de N est un élément de $\mathcal{L}(N)$.

Notation : Dans ce document $|\rho|$ désigne la longueur de la séquence ρ (son nombre d'événements) et $time(\rho)$ la longueur temporelle de la séquence (le temps écoulé entre le début et la fin de la séquence), qui correspond à la somme des dates de tir : $\rho = \theta_0 e_0 \ldots \theta_n e_n \Rightarrow time(\rho) = \sum_{k=0}^{n} \theta_k$.

Pour vérifier l'occurrence de certaines séquences particulières d'événements, il est nécessaire d'extraire des événements de mots d'un PTÉS . Pour cela nous utilisons la notion de *sous-suite*

définie de la manière suivante :

Définition 9. Une *sous-suite* d'une trace $\rho = \theta_0 e_0 \dots \theta_n e_n$ est une séquence $\rho' = \theta'_0 e'_0 \dots \theta'_k e'_k$, $k \leq n$ telle que :

1. $k \leq n$
2. $\rho = \rho_0.\delta_0 e'_0.\rho_1.\delta_1 e'_1.\rho_2 \dots \rho_k.\delta_k e'_k.\rho_{k+1}$ (éventuellement $\rho_i = 0\varepsilon$)
3. $\forall i \in [0,k]$, $\theta'_i = time(\rho_i) + \delta_i$

De manière similaire, une sous-suite d'une trace atemporelle $w = e_0 \dots e_n$ est une séquence $w' = e'_0 \dots e'_k$ telle que :

1. $k \leq n$
2. $w = w_0.e'_0.w_1.e'_1 \dots w_k.e'_k.w_{k+1}$ (éventuellement $w_i = \varepsilon$)

L'ensemble des sous-suites d'une trace $\rho \in \mathcal{L}(N)$ est noté $\omega(\rho)$.

De manière informelle l'on peut décrire une sous-suite d'une trace temporelle comme la projection temporelle de cette trace ρ sur un sous-ensemble ordonné de ses symboles (ici de ses événements), c.-à-d. que l'on conserve le temps écoulé entre l'apparition des différents symboles.

Exemple 4. Considérons le mot $\rho = 2a.3b.1c.4b.2c.5a.0a$. La séquence $2a$ est une sous-suite de ρ de longueur 1 obtenue en posant $e'_0 = a$ (première occurrence de a dans ρ). De même, $5b$ et $12c$ sont des sous-suites de ρ obtenues en posant $e'_0 = b$ (première occurrence de b dans ρ) et $e'_0 = c$ (première occurrence de c dans ρ). $\rho' = 5b.5b.7a$ est une sous-suite de ρ en posant $e'_0 = b$ (première occurrence de b dans ρ), $e'_1 = b$ (deuxième occurrence de b dans ρ) et $e'_2 = a$ (deuxième occurrence de a dans ρ). On peut noter qu'il existe une autre manière d'obtenir ρ' à partir de ρ en posant e'_2 comme la troisième occurrence de a dans ρ (le temps écoulé entre la deuxième et troisième occurrence de a dans ρ étant nul).

De même, la séquence atemporelle $w' = e_0.e_2$ est une sous-suite de la séquence atemporelle $w = e_0.e_1.e_2.e_3$ (en posant $e'_0 = e_0$ et $e'_1 = e_2$).

Pour identifier des sous-suites dans une trace, il est nécessaire de pouvoir projeter une trace sur un sous-ensemble de ses événements. Pour cela la projection d'une trace sur un alphabet Σ_p est définie de la manière suivante :

— $\mathbf{P}_{\Sigma\to\Sigma_p}(\theta_1 e_1.\theta_2 e_2 \dots \theta_n e_n) = \theta_1 e_1.\mathbf{P}_{\Sigma\to\Sigma_p}(\theta_2 e_2 \dots \theta_n e_n)$ si $e_1 \in \Sigma_p$

— $\mathbf{P}_{\Sigma\to\Sigma_p}(\theta_1 e_1.\theta_2 e_2 \dots \theta_n e_n) = \mathbf{P}_{\Sigma\to\Sigma_p}((\theta_1+\theta_2) e_2 \dots \theta_n e_n)$ si $e_1 \notin \Sigma_p$

— $\mathbf{P}_{\Sigma\to\Sigma_p}(\theta e) = 0\varepsilon$ si $e \notin \Sigma_p$ où 0ε est l'élément neutre pour le langage.

2.1.2 Abstraction de réseau de Petri temporel - Le graphe des Classes :

Une abstraction des PT introduite par Berthomieu [BM83] étend le graphe des marquages accessibles en intégrant le temps dense sous forme de polyèdres. Cette abstraction repose sur la notion de classes, qui sont des classes de recouvrement englobant des ensembles infinis d'états

présentant le même marquage et un domaine temporel proche. L'ensemble des dates de tir de l'ensemble des transitions sensibilisées depuis une classe forment un polyèdre appelé *domaine de tir*. Nous proposons la définition suivante, adaptée de [GRR04] :

Définition 10. Le Graphe des Classes (GC par la suite) d'un PTÉS $N = \langle P, T, A, \Sigma, \ell, I_s, M_0 \rangle$ est le triplet (C,$\mathcal{C}_0, \rightarrow$) tel que :
- $\mathcal{C}_0 = (M_0, F_0)$, où $F_0 \in {I_{\mathbb{Q}_+}}^T$ est le domaine de tir initial de N. Pour toute transition t sensibilisée par M_0, la contrainte relative à t dans F_0 est donnée par $I_s(t)$.
- C$\in \{0,1\}^P \times {I_{\mathbb{Q}_+}}^T$ est l'ensemble des classes correspondant à des états accessibles de N
- $\rightarrow \in$ C$\times$ C est la relation définie comme suit : $(M, F) \xrightarrow{t} (M', F')$ si et seulement si
  - t est tirable depuis (M, F)
  - $M' = (M \setminus pre(t)) \cup post(t)$
  - $F' = next(F, t)$

où $next : {I_{\mathbb{Q}_+}}^T \times T \rightarrow {I_{\mathbb{Q}_+}}^T$ est la procédure de calcul du domaine de tir F' associé à un marquage accessible M' depuis un marquage M par le tir de la transition t qui est définie comme suit :

1. pour chaque transition $t' \neq t$ sensibilisée par M, établir le tir de t par l'ajout de la contrainte $\theta \leq \theta'$ et du changement de variable $\theta' = \theta + \theta'_{upd}$ (θ_{upd} est ici une variable de substitution)
2. éliminer les variables relatives à des transitions sensibilisées dans M et non dans M'
3. ajouter les contraintes relatives aux transitions nouvellement sensibilisées par M'
4. déterminer la forme canonique pour l'ensemble des contraintes de F'

Le graphe des classes est ce que l'on appelle en algèbre des processus un Système de Transitions Temporel. Il accepte un langage atemporel (les séquences de transitions tirables sans temporisation) et temporel (les séquences temporisées) tous deux clos par préfixe. Pour un PT labellisé, on considère par abus de langage que le langage du PTÉS est accepté par le graphe des classes en remplaçant les transitions par leur label. Nous ferons référence au langage d'un PT comme étant l'ensemble des séquences de transitions temporisées et au langage d'un PTÉS comme étant l'ensemble des séquences de labels.

Exemple 5. La Figure 2.2 et la Table 2.1 présentent respectivement le GC associé au réseau de la Figure 2.1 et le contenu des différentes classes. C'est un protocole de transfert de donnée avec perte. Les transitions t_1 et t_4 représentent les envois de paquets, les transitions t_7 et t_{10} modélisent la réception de ces paquets. Les transitions t_8 et t_{11} (resp. t_3 et t_6) modélisent l'envoi (resp. la réception) des accusés de réception. Les pertes dues à des délais écoulés sont représentées par les transitions t_{13}, t_{14}, t_{15} et t_{16} (il n'y a pas de places en sortie de ces transitions, le message est perdu). Les délais de retransmission des messages (transitions t_2 et t_5) sont tels qu'il ne peut y avoir au plus qu'un seul message dedans (ainsi le PT est sauf). Nous allons détailler ici le calcul du domaine de tir de la classe 3 atteinte après le tir de t_7 depuis la classe 1 ($\mathcal{C}_0 \xrightarrow{t_1} C_1 \xrightarrow{t_7} C_3$). Le domaine de tir de la classe 1 est le suivant : $\{5 \leq \theta_2 \leq 6, 0 \leq \theta_7 \leq 1, 0 \leq \theta_{13} \leq 1\}$.

1. mise à jour des contraintes des transitions sensibilisées dans le marquage de la classe 1 : θ_2 (variable relative au tir de t_2) est mise à jour. On ajoute les contraintes suivantes au

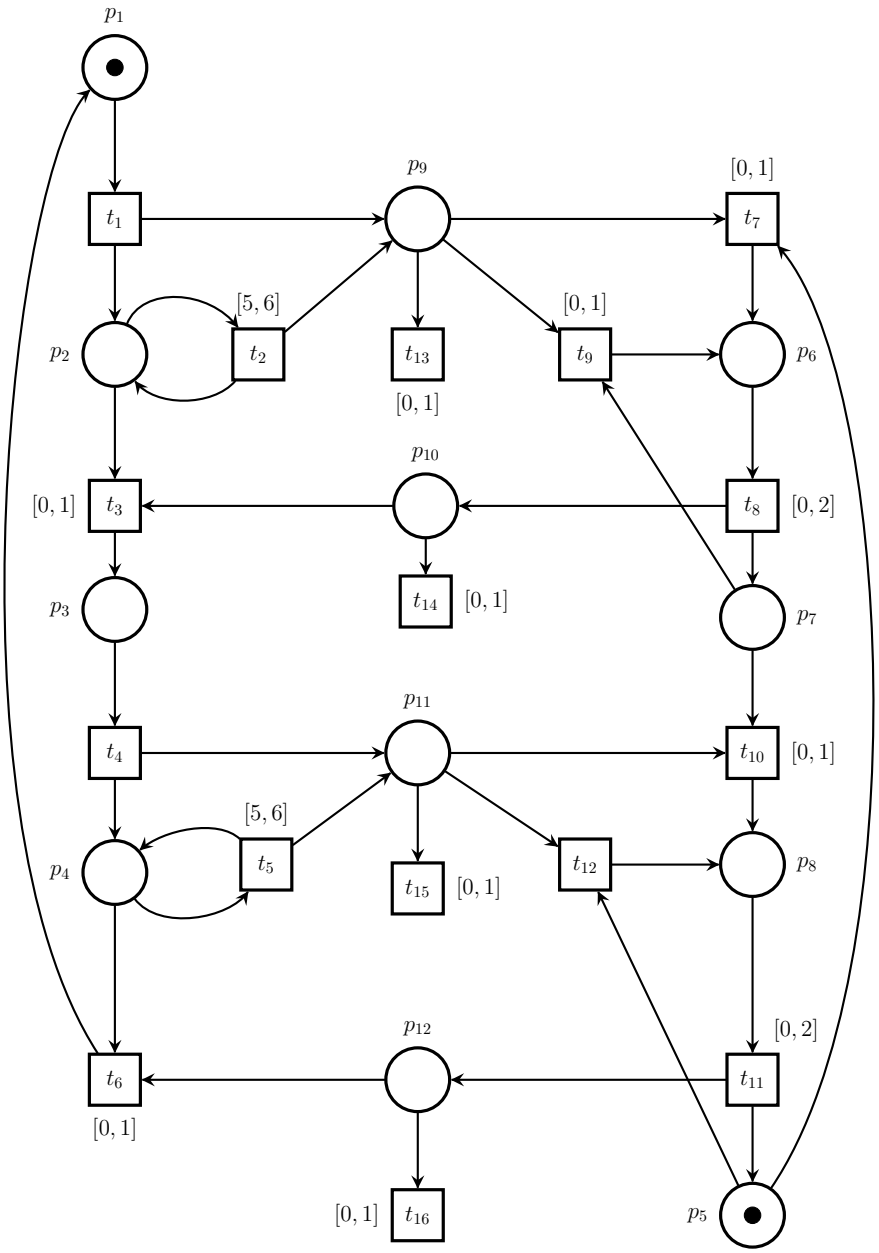


FIGURE 2.1 – Un PT sauf pour modéliser le protocole de bits alterné [Dia13]

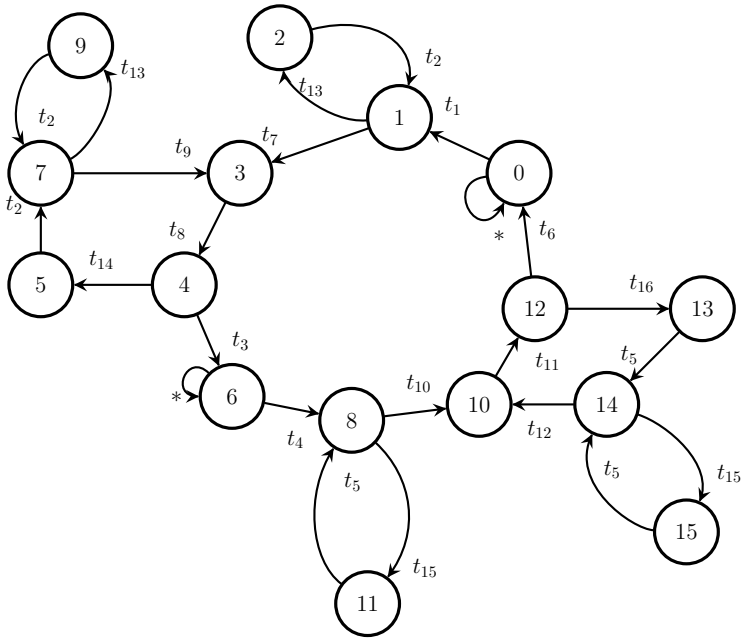


FIGURE 2.2 – Graphe des classes du PT présenté Figure 2.1

domaine de tir $\theta_7 \leq \theta'_2$, $\theta'_2 = \theta_2 + \theta_{2,upd}$, et $0 \leq \theta_{2,upd} \leq 1$ (les autres contraintes étant supprimées par l'étape suivante, nous ne développerons pas ici cette étape pour elles).

2. suppression des contraintes relatives à des transitions désensibilisées par le tir de t_7 : t_2 et t_{13} ne sont plus sensibilisées, leurs contraintes seront supprimées du nouveau domaine de tir
3. ajout des transitions nouvellement sensibilisées : t_8 est nouvellement sensibilisée. La contrainte la concernant est déterminée par son intervalle statique, à savoir : $0 \leq \theta_8 \leq 2$.
4. mise à jour du domaine

Nous obtenons ainsi le domaine de tir suivant : $\{4 \leq \theta_2 \leq 6, 0 \leq \theta_8 \leq 2\}$.

Les différentes classes, détaillées ci-dessus, ont été obtenues à l'aide du logiciel TINA [2] [BRV04].

La forme des domaines de tir est la suivante, pour une classe C et $E(C)$ l'ensemble des transitions sensibilisées dans C :

$$\begin{cases} \alpha_i \leq t_i \leq \beta_i, t_i \in E(C) \\ t_k - t_j \leq \gamma_{kj}, (t_k, t_j) \in E(C)^2 \end{cases} \tag{2.1}$$

Dans cette représentation (en supposant qu'elle soit canonique) α_i et β_i représentent les dates de tir au plus tôt et au plus tard de t_i depuis la classe C. Ces coefficients sont calculés en fonction des bornes des intervalles statiques lors de la sensibilisation, et mis à jour en fonction du temps écoulé pour le tir d'une transition (de ce fait les coefficients α décroissent plus vite que les coefficients β,

2. https://projects.laas.fr/tina/index.php

class 0 marking p1 p5 domain 0 <= t1	class 1 marking p2 p5 p9 domain 0 <= t13 <= 1 5 <= t2 <= 6 0 <= t7 <= 1	class 2 marking p2 p5 domain 4 <= t2 <= 6	class 3 marking p2 p6 domain 4 <= t2 <= 6 0 <= t8 <= 2
class 4 marking p10 p2 p7 domain 0 <= t14 <= 1 2 <= t2 <= 6 0 <= t3 <= 1	class 5 marking p2 p7 domain 1 <= t2 <= 6	class 6 marking p3 p7 domain 0 <= t4	class 7 marking p2 p7 p9 domain 0 <= t13 <= 1 5 <= t2 <= 6 0 <= t9 <= 1
class 8 marking p11 p4 p7 domain 0 <= t10 <= 1 0 <= t15 <= 1 5 <= t5 <= 6	class 9 marking p2 p7 domain 4 <= t2 <= 6	class 10 marking p4 p8 domain 0 <= t11 <= 2 4 <= t5 <= 6	class 11 marking p4 p7 domain 4 <= t5 <= 6
class 12 marking p12 p4 p5 domain 0 <= t16 <= 1 2 <= t5 <= 6 0 <= t6 <= 1	class 13 marking p4 p5 domain 1 <= t5 <= 6	class 14 marking p11 p4 p5 domain 0 <= t12 <= 1 0 <= t15 <= 1 5 <= t5 <= 6	class 15 marking p4 p5 domain 4 <= t5 <= 6

TABLE 2.1 – Contenu des classes de la Figure 2.2

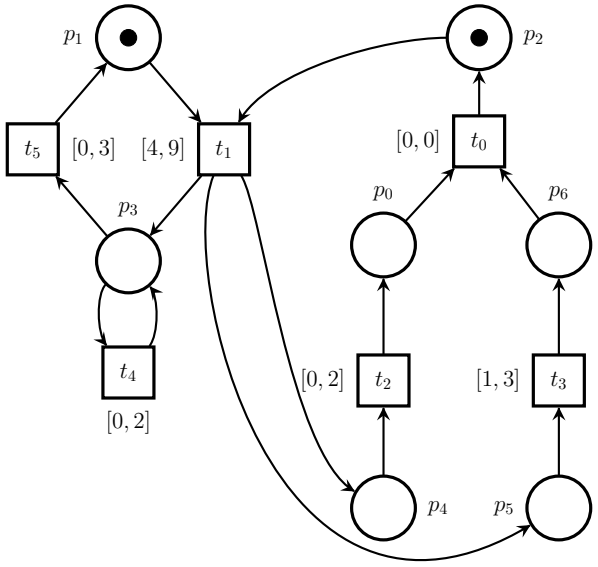


FIGURE 2.3 – Exemple de PTÉS sauf adapté de [BM83]. Dans cet exemple, depuis le marquage $p_4p_5p_3$, le tir d'une transition sensibilisée ne désensibilise pas deux autres transitions qui resteront persistantes pour la classe atteinte.

avec 0 comme limite basse). Les coefficients de type γ_{kj} se calculent comme la différence entre β_k et α_j lors de la sensibilisation de t_k ou t_j, et comme $\min(\gamma_{kj}^{precedent}, \beta_k - \alpha_j)$ lorsque les transitions t_k et t_j sont persistantes (c.-à-d. qu'elles ont été sensibilisées par un autre marquage que le marquage courant et le sont toujours dans le marquage courant). Dans le cas de transitions persistantes, ce calcul n'est pas trivial. La Figure 2.3 illustre à l'aide de certaines de ses classes ce phénomène. Le PTÉS est une adaptation de l'exemple présenté dans [BM83] (qui a été modifié pour être sauf). Les classes présentées dans le Tableau 2.2 (qui sont les classes du GC de la Figure 2.3) ont été obtenues à l'aide de TINA. Les seuls coefficients de type γ représentés sont ceux dont le calcul est non trivial. Nous pouvons ainsi remarquer que les paquets de transitions concernés par ces γ non triviaux sont les couples (t_2, t_3), (t_3, t_4) et (t_3, t_5) qui sont les paquets de transitions pouvant être persistantes après le tir de certaines transitions (ici t_2, t_3, t_4 ou t_5).

L'intérêt de cette représentation est que le signe des coefficients de type γ informe sur le fait qu'une transition soit tirable ou non depuis la classe courante [Lub21] :

Théorème 1. *Soit C une classe du GC d'un PTÉS N et soit $E(C)$ l'ensemble des transitions sensibilisées dans C. $t_i \in E(C)$ est tirable depuis $C \Leftrightarrow \forall t_j \in E(C) \setminus t_i$, $\gamma_{ji} \geq 0$.*

Démonstration. Soit C une classe, $E(C)$ l'ensemble des transitions sensibilisées dans C, et $t_i \in T$. $t_i \notin E(C) \Rightarrow t_i$ n'est pas tirable depuis C. Si $t_i \in E(C)$, elle est tirable si et seulement si aucune transition t_j n'est forcée d'être tirée avant t_i. t_j est forcée de tirer avant t_i si et seulement si sa date de tir au plus tard est plus petite que la date de tir au plus tôt de t_i, autrement dit $\beta_j - \alpha_i < 0$.

```
class 0
  marking
   p1 p2
  domain
   4 <= t1 <= 9
```

```
class 1
  marking
   p3 p4 p5
  domain
   0 <= t2 <= 2
   1 <= t3 <= 3
   0 <= t4 <= 2
   0 <= t5 <= 3
```

```
class 2
  marking
   p0 p3 p5
  domain
   0 <= t3 <= 3
   0 <= t4 <= 2
    0 <= t5 <= 3
    t4 - t3 <= 1
    t5 - t3 <= 2
```

```
class 3
  marking
   p3 p4 p6
  domain
   0 <= t2 <= 1
   0 <= t4 <= 1
   0 <= t5 <= 2
```

```
class 4
  marking
   p3 p4 p5
  domain
   0 <= t2 <= 0
   0 <= t3 <= 3
   0 <= t4 <= 2
   0 <= t5 <= 3
   t2 - t3 <= 1
```

```
class 5
  marking
   p1 p4 p5
  domain
   0 <= t2 <= 2
   0 <= t3 <= 3
   t2 - t3 <= 1
```

```
class 6
  marking
   p0 p3 p6
  domain
   0 <= t0 <= 0
   0 <= t4 <= 1
   0 <= t5 <= 2
```

```
class 7
  marking
   p0 p3 p5
  domain
   0 <= t3 <= 3
   0 <= t4 <= 2
   0 <= t5 <= 3
```

```
class 8
  marking
   p0 p1 p5
  domain
   0 <= t3 <= 3
```

```
class 9
  marking
   p3 p4 p6
  domain
   4 <= t2 <= 6
   0 <= t4 <= 2
   0 <= t5 <= 3
```

```
class 10
  marking
   p1 p4 p6
  domain
   0 <= t2 <= 1
```

```
class 11
  marking
   p2 p3
  domain
   0 <= t4 <= 1
   0 <= t5 <= 2
```

```
class 12
  marking
   p0 p3 p6
  domain
   0 <= t0 <= 0
   0 <= t4 <= 2
   0 <= t5 <= 3
```

```
class 13
  marking
   p0 p1 p6
  domain
   0 <= t0 <= 0
```

```
class 14
  marking
   p2 p3
  domain
   0 <= t4 <= 2
   0 <= t5 <= 3
```

TABLE 2.2 – Tableau présentant l'ensemble des classes du GC du PTÉS Figure 2.3

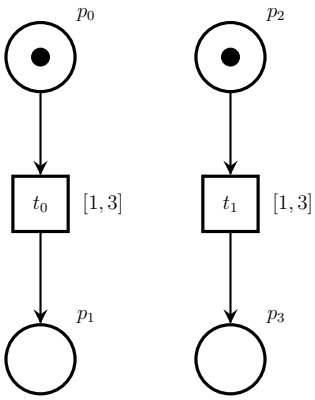


FIGURE 2.4 – PTÉS illustrant la perte d'information en explorant classe par classe le GC d'un PT

Pour un tel t_j, nous avons $\gamma_{ji} < 0$. t_i est donc tirable si et seulement si le minimum des γ_{ji} pour $t_j \in E(C)$ est positif. □

Il est important de noter que si une transition t est tirable depuis une classe C à une temporisation θ, pour toute exécution r' telle que $\mathcal{C}_0 \xrightarrow{r'} C$, il n'y a aucune garantie que $r = r'.\theta t$ soit une exécution admissible pour N (voir Exemple 6).

2.1.3 Limite du graphe des classes pour notre problème :

Le graphe des classes est une excellente abstraction de PT lorsque l'on s'intéresse à des problèmes d'accessibilité [HB10]. Cependant, comme relevé précédemment, le langage temporel accepté par le GC contient le langage du système, mais il accepte également des mots qui ne sont pas dans le langage du PT. Ces mots ne faisant pas partie du langage du PT sont acceptés car le domaine de tir d'une classe donnée représente l'ensemble des temporisations possibles pour une transition tirable depuis l'un des états de cette classe, et ce sans connaître la trajectoire particulière qui a mené à cette classe. Autrement dit pour toute temporisation d d'un domaine de tir d'une classe pour une transition donnée t tirable depuis cette classe, il existe une trajectoire permettant de tirer t à la date d, mais il n'y a aucune garantie que toute trajectoire accédant à cette classe permette de tirer t à la temporisation d. Lorsque l'on cherche à représenter tout ce qui est tirable depuis un domaine (autrement dit si l'on suit la philosophie du GC), on oublie le passé d'une séquence (comment elle est accessible), et l'on ne peut donc pas se baser uniquement sur les domaines de tir pour savoir si une séquence temporisée est une exécution admissible d'un PT. Il est nécessaire pour connaître l'ensemble des trajectoires pour lesquelles une temporisation est admissible de connaître le passé, c.-à-d. l'évolution du PT qui a mené à la classe considérée.

Exemple 6. Considérons le PT représenté Figure 2.4. Le GC de ce PT a 4 classes pour les marquages $p_0.p_2$ (classe C_0), $p_1.p_2$ (classe C_1), $p_0.p_3$ (classe C_2) et $p_1.p_3$ (classe C_4). Les domaines de tir de ces classes sont respectivement $\{1 \leq t_0 \leq 3, 1 \leq t_1 \leq 3\}$, $\{0 \leq t_1 \leq 2\}$, $\{0 \leq t_0 \leq 2\}$ et $\emptyset$. Considérons la séquence de tir $t_0.t_1$ tirable pour ce PT. Si l'on cherche un échéancier (c.-à-d. une temporisation) pour cette séquence de transition, la séquence $2t_0.2t_1$ respecte les domaines de tir

des classes que l'on visite lorsque l'on suit cette séquence de tir (C_0 et C_1 ici). Or cette séquence n'est pas admissible pour le PT.

Cette limite du GC nous pose problème par la suite, car nous cherchons à étudier une intersection de langages de PTÉS. Le graphe des classes n'est pas une abstraction satisfaisante de ce point de vue là. Nous allons présenter dans la section suivante comment utiliser le GC pour construire une abstraction finie des préfixes du langage d'un PTÉS.

2.2 Abstraction polyédrique du Réseau de Petri temporel

Nous avons vu dans la section précédente que le GC n'est pas une abstraction satisfaisante pour représenter strictement le langage temporel d'un PT, car le langage généré par une exploration du GC en choisissant des temporisations uniquement en se basant sur les domaines de tir est une sur-approximation du langage du PTÉS. Malgré cela le GC présente des avantages que nous allons exploiter pour construire une abstraction du langage d'un PTÉS. L'abstraction développée dans cette section doit satisfaire les propriétés suivantes :

P0 Abstraire les préfixes du langage d'un PTÉS de manière finie. Si une propriété de diagnostic est vérifiée pour un préfixe, alors elle sera vraie pour toute continuation de ce préfixe. Les langages des PTÉS considérés dans ce manuscrit sont clos par préfixe, il est donc naturel de chercher à les *regrouper* sous un même objet. De plus, pour éviter des problèmes d'indécidabilité, il faut pouvoir travailler sur des objets mathématiques en nombre fini. Le temps est continu, il y a pour une séquence atemporelle de transitions tirables dans un PTÉS une infinité de temporisations possibles dans le cas général (hors intervalles ponctuels). Nous cherchons donc à éviter l'énumération de toutes les séquences, et à représenter un ensemble fini de domaines temporels regroupant les temporisations proches (c.-à-d. appartenant à un même ensemble convexe).

P1 Ne pas être un sur ensemble du langage du PTÉS étudié. Nous souhaitons ici éviter l'existence d'exécutions non admissibles pour le PTÉS de départ et pourtant admissibles pour l'abstraction construite.

Nous proposons ici une abstraction des préfixes du langage du système sous forme de polyèdre. Pour une transition donnée dont l'intervalle statique n'est pas trivial (c.-à-d. n'est pas réduit à une seule valeur) il existe une infinité de dates de tir différentes pour cette transition. Pour une séquence de transitions donnée σ tirable depuis l'état initial, il existe une infinité d'exécutions s'appuyant sur σ. Malgré tout, ces exécutions ont énormément de caractéristiques communes. Partant de ce constat il nous est paru naturel de chercher à les abstraire en un seul objet mathématique. Cet objet doit contenir d'une part la séquence de transitions σ (que nous appelons *support*) ainsi que l'enveloppe temporelle, c.-à-d. l'enveloppe de toutes les traces admissibles pour σ.

Considérons un support $\sigma = t_0 \dots t_n$. Deux paramètres influent sur le domaine de tir d'une transition t_i dans ce support. Le premier est le temps écoulé entre la sensibilisation de t_i et son tir. Ce temps est contraint par l'intervalle statique de t_i. Le second paramètre est le minimum des dates de tir au plus tard de t_i parmi l'ensemble des transitions sensibilisées en même temps que t_i. D'après la sémantique établie, lorsque l'on atteint la borne supérieure du domaine de tir d'une transition parmi un ensemble de transitions sensibilisées, une transition doit être tirée. t_i ne peut donc pas être tirée après ce temps. Ce temps dépend de l'intervalle statique de la transition en

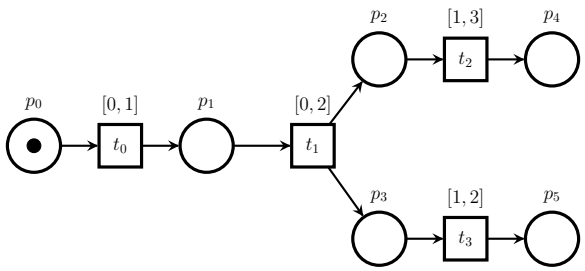


FIGURE 2.5 – PTÉS pour l'illustration de la notion de *chemin*

question ainsi que du temps écoulé depuis sa sensibilisation. Autrement dit, pour un support σ, son enveloppe temporelle peut s'écrire comme un ensemble de contraintes sur la date d'occurrence de ses différentes transitions en fonction de leur intervalle statique et de leur instant de sensibilisation. Un tel ensemble de contraintes définit un polyèdre sur $\mathbb{R}_+^{|\sigma|}$:

Définition 11. Un polyèdre P est un domaine de $\mathbb{R}^n$ défini par :

$$P = \{x \in \mathbb{R}^n, Ax \leq b\}$$

avec $A \in \mathbb{R}^{m \times n}$ et $b \in \mathbb{R}^m$. Un polyèdre P est dit *convexe* si $\forall (x, y) \in P$, $\forall \lambda \in [0, 1]$, $\lambda.x + (1 - \lambda).y \in P$.

Notation : Dans ce document nous notons $\mathbb{P}$ l'ensemble des polyèdres de dimension finie définis sur $\mathbb{R}_+$, et nous notons $\{\theta_1, \dots \theta_n\} \models \Pi$ pour dire qu'un ensemble de réels $\{\theta_1, \dots \theta_n\}$ satisfait les contraintes de définition d'un polyèdre $\Pi \in \mathbb{P}$.

Exemple 7. Intéressons nous au PTÉS de la Figure 2.5. Considérons la séquence de transitions suivante : $\sigma = t_0.t_1.t_2.t_3$. Cet ordre de tir est admissible pour ce PTÉS. Intéressons nous aux différentes dates relatives d'occurrence possibles pour ces transitions selon cette séquence de tir. Posons θ_0 la date d'occurrence de t_0, et pour toute autre transition de la séquence, θ_i représente le temps écoulé entre le tir de la transition t_{i-1} et le tir de t_i. Quelle que soit l'exécution s'appuyant sur σ, l'intervalle statique de t_0 impose que $0 \leq \theta_0 \leq 1$. Il en est de même pour le tir de t_1 : cette transition est sensibilisée par le tir de t_0, et c'est l'unique transition sensibilisée pour le marquage atteint par le tir de t_0. Nous avons donc $0 \leq \theta_1 \leq 2$, où les bornes de la contrainte sont données par l'intervalle statique de t_1. t_2 étant tirée directement après le tir de t_1, le temps écoulé entre le tir de t_1 et celui de t_2 est contraint d'une part par son intervalle statique (t_2 est sensibilisée par le tir de t_1), nous avons donc $1 \leq \theta_2 \leq 3$. De plus, t_3 est sensibilisée en parallèle de t_2 et est tirée après. Cela impose donc que le temps écoulé entre la sensibilisation de t_2 et son tir (ici θ_2) est inférieur à la borne supérieure de l'intervalle statique de t_3. Nous avons donc $\theta_2 \leq 2$. Intéressons nous maintenant au tir de t_3. Rappelons que t_3 est sensibilisée par le tir de t_1, et que θ_3 représente le temps écoulé entre le tir de t_2 et celui de t_3. Le temps écoulé entre la sensibilisation de t_3 et son tir est donc $\theta_2 + \theta_3$. D'après l'intervalle statique de t_3, nous avons donc $1 \leq \theta_2 + \theta_3 \leq 2$. En posant les changement de variables $y_0 = \theta_0$ et $\forall i \in [1, 3]$, $y_i - y_{i-1} = \theta_i$, nous pouvons réécrire les

contraintes précédentes sous la forme du système de différences suivant :

$$\begin{cases} 0 \leq y_0 \leq 1 \\ 0 \leq y_1 - y_0 \leq 2 \\ 1 \leq y_2 - y_1 \leq 3 \\ 1 \leq y_3 - y_1 \leq 2 \\ 0 \leq y_3 - y_2 \end{cases} \tag{2.2}$$

Cet ensemble peut se réécrire sous la forme $Ay \leq b$ avec :

$$A = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}, y = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} \text{ et } b = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 2 \\ 1 \\ 3 \\ 1 \\ 2 \\ 0 \end{pmatrix}.$$

Le changement de variable proposé est tel que pour un entier donné i, la valeur de y_i est donc $y_i = \theta_i + y_{i-1}$, et ainsi nous avons $y_i = \sum_{j=0}^{i} \theta_j$. y_i représente donc pour la transition t_i le temps écoulé entre l'origine des temps de σ et le tir de la transition t_i.

L'exemple 7 montre que l'enveloppe temporelle d'une séquence de transitions peut s'exprimer sous la forme d'un polyèdre. La variable y_i associée au tir de la transition t_i représente le temps écoulé entre l'origine du PTÉS et le tir de la transition t_i. Autrement dit, y_i représente le temps écoulé entre l'origine des temps du PTÉS et le tir de t_i. De cela découle trois propriétés des enveloppes temporelles :

1. Pour une enveloppe temporelle Π, son expression sous forme de polyèdre $\{y \in \mathbb{R}_+^n, A.y \leq b\}$ est telle que nous avons $A \in \{-1, 0, 1\}^{m \times n}$ et $b \in \mathbb{Q}_+^m$ (les bornes des intervalles statiques sont à valeurs dans $\mathbb{Q}_+$ d'après la Définition 7). En effet, une contrainte représente le temps écoulé entre le tir de deux transitions. Pour cela pour une contrainte donnée ces transitions n'apparaissent donc qu'une fois avec un coefficient de norme 1, ainsi les coefficients de la matrice A seront à valeur dans $\{-1, 0, 1\}$.

2. Pour $i \in [0, n]$, la valeur de y_i ne dépend que des variables $(y_j)_{j<i}$. En effet, y_i représente la date de tir absolue de la transition t_i dans une séquence σ. Le tir d'une transition dépend de sa sensibilisation. Lorsqu'elle a été tirée, elle n'est plus sensibilisée, ce qui se passe après son tir ne peut donc pas influencer sa date de tir.

3. Une enveloppe temporelle est convexe (intersection d'intervalles qui sont des ensembles convexes).

Définition 12. Un *chemin* d'un PTÉS N est une paire $\pi = (\sigma, \Pi) \in (T^* \times \mathbb{P})$ où σ est une séquence de transitions tirable depuis l'état initial et Π est l'enveloppe temporelle de σ, qui est un polyèdre convexe dont l'ensemble de contraintes s'exprime sous la forme d'un système de différences, pour lequel chaque variable $(y_i)_{i\in[1,|\sigma|]}$ est associée à une transition de σ (y_i est associée à la transition t_i) et représente la date de tir absolue de la transition t_i (c.-à-d. sa date de tir relativement à l'origine temporelle de N). Autrement dit, si $\sigma = t_0 \dots t_n$ et $r = \theta_0 t_0 \dots \theta_n t_n$ (noté $r \in \pi$), alors $(\{\theta_0, \theta_0 + \theta_1, \dots, \sum_{i=0}^{n} \theta_i\} \models \Pi) \Leftrightarrow (r$ est admissible pour $N)$.
Γ désigne l'ensemble des chemins d'un PTÉS N et Γ_k désigne l'ensemble des chemins de taille $k \in \mathbb{N}^*$.

Une conséquence directe de la Définition 12 est que tout chemin satisfait la propriété **P1**. Nous proposons ici une procédure de calcul pour déterminer Π pour une séquence de transitions σ donnée. Ce calcul se base sur une exploration du GC d'un PTÉS. La procédure de calcul est similaire à celle développée dans [WMS15], où l'objectif des auteurs est de calculer un GC modifié pour un système et une faute atemporelle f (qui peut être vue comme une réduction des arcs labellisés par des transitions non observables dans le système), et d'indiquer si une trace est fautive ou non.

Pour une séquence σ tirable depuis l'état initial, on utilise la séquence de classes visitée (c.-à-d. l'ensemble des classes atteintes par le tir des transitions de σ) pour calculer les contraintes temporelles associées aux différentes transitions. Deux types de contraintes différentes sont calculées :

- Contraintes d'ordre : ces contraintes modélisent l'ordre de tir des transitions dans σ. Ces contraintes sont de la forme $y_i - y_{i-1} \geq 0$ où y_i est la variable associée à la date de tir absolue (relative au démarrage du PTÉS) de la transition t_i (à la i-ème place dans σ).

- Contraintes structurelles : ces contraintes modélisent, pour une séquence donnée, les contraintes dues à la structure du réseau (intervalles statiques, parallélisme, conflits, etc.). Elles sont de la forme $\alpha_{ij} \leq y_i - y_j \leq \beta_{ij}$ où $(i, j) \in \mathbb{N}^* \times \mathbb{N}, j < i$, ou de la forme $y_i - y_j \leq \beta'_{ij}$. Pour la première forme, la contrainte peut se lire de la manière suivante : *la date de tir de la transition t_i relativement à sa sensibilisation (le tir de t_j sensibilise t_i) pour le support σ est borné entre α_{ij} et β_{ij} unités de temps.* Pour la deuxième forme, cela peut se lire ainsi : *la date de tir de t_i est nécessairement inférieure à la date de tir au plus tard des transitions sensibilisées en même temps mais qui ne sont pas tirées.*

L'Algorithme 1 présente une procédure de calcul de ces contraintes. Les lignes 1 à 4 synthétisent les contraintes d'ordre, et les lignes 5 à 14 synthétisent les contraintes de structure. Pour chaque transition, sa classe de sensibilisation C_j (ligne 6) dans s_c est récupérée. Ensuite, en utilisant le domaine de tir de C_j, on calcule les dates de tir au plus tôt et au plus tard de t_i (ligne 7). La contrainte structurelle imposée par la sensibilisation de t_i (c.-à-d. celle de la première forme) est ainsi synthétisée et intégrée à l'enveloppe temporelle Π (ligne 9 dans le cas où t_i est sensibilisée dans la classe initiale C_0, ligne 11 sinon). On va ensuite considérer l'ensemble des transitions t_k sensibilisées dans C_i (C_i correspond à la classe dans laquelle t_i est tirée) et qui ne seront pas tirées depuis C_i. Pour chaque transition t_k sensibilisée en même temps que t_i au moment de son tir (c.-à-d. dans la classe C_i), sa classe de sensibilisation est récupérée (ligne 14), et la contrainte structurelle associée (de la deuxième forme) est calculée (lignes 15) et intégrée dans l'enveloppe temporelle (ligne 17 si t_k est sensibilisée dans C_0, ligne 19 sinon).

Algorithme 1 Procédure de calcul de l'enveloppe temporelle d'un support σ

Entrées : Un support σ, $s_c = C_0 \dots C_{|\sigma|}$ la séquence de classes visitée associée
Sortie : L'enveloppe temporelle Π de σ

$\Pi \leftarrow \{0 \leq y_0\}$
pour $i \in [1, |\sigma| - 1]$ **faire**
  $\Pi \leftarrow \Pi \cup \{0 \leq y_i - y_{i-1}\}$
fin pour
pour $i \in [0, |\sigma| - 1]$ **faire**
  $C_j \leftarrow$ `classeSensibilisation`$(t_i, s_{c|i})$
  $(\alpha_{ij}, \beta_{ij}) \leftarrow$ `maximiseMinimise`(t_i, C_j)
  si $C_j == C_0$ **alors**
    $\Pi \leftarrow \Pi \cup \{\alpha_{ij} \leq y_i \leq \beta_{ij}\}$
  sinon
    $\Pi \leftarrow \Pi \cup \{\alpha_{ij} \leq y_i - y_{j-1} \leq \beta_{ij}\}$
  fin si
  pour $t_k \in TransitionsSensibilisees(C_i) \setminus t_i$ **faire**
    $C_k \leftarrow$ `classeSensibilisation`$(t_k, s_{c|i})$
    $\beta'_{ik} \leftarrow$ `maximiser`(t_k, C_k)
    si $C_k == C_0$ **alors**
      $\Pi \leftarrow \Pi \cup \{y_i \leq \beta'_{ij}\}$
    sinon
      $\Pi \leftarrow \Pi \cup \{y_i - y_{k-1} \leq \beta'_{ik}\}$
    fin si
  fin pour
fin pour
retourner Π

Exemple 8. Revenons sur le PTÉS de la Figure 2.5. Nous allons développer l'Algorithme 1 pour calculer l'enveloppe temporelle du support $\sigma = t_0.t_1.t_2.t_3$ présenté dans l'Exemple 7. Les variables associées aux différentes transitions sont notées y_0, y_1, y_2 et y_3. En considérant l'ordre de tir des transitions (t_0 puis t_1 puis t_2 puis t_3), en appliquant la ligne 3 nous pouvons écrire les contraintes d'ordre suivantes (à intégrer dans l'enveloppe temporelle Π) : $0 \leq y_0$, $y_1 - y_0 \geq 0$, $y_2 - y_1 \geq 0$, $y_3 - y_2 \geq 0$.

Intéressons nous maintenant pour chaque transition au calcul de ses contraintes structurelles. Pour les contraintes structurelles de t_0, cette transition est sensibilisée dans l'état initial. Ses dates de tir au plus tôt et au plus tard sont 0 et 1, nous insérons donc dans Π la contrainte suivante $0 \leq y_0 \leq 1$ (ligne 9). Aucune transition n'est sensibilisée en même temps que t_0, les lignes 11 à 13 ne sont donc pas exécutées. Concernant t_1, la contrainte liée à sa sensibilisation est $0 \leq y_1 - y_0 \leq 2$ (ligne 11), et aucune transition n'est sensibilisée en même temps qu'elle (les lignes 17 à 20 ne sont donc pas exécutées). Pour t_2, cette transition est sensibilisée en même temps que t_3. La contrainte structurelle due à sa sensibilisation est $1 \leq y_2 - y_1 \leq 3$ (ligne 11). De plus, t_3 étant sensibilisée en même temps, on ajoute une contrainte structurelle de la deuxième forme, c.-à-d. une contrainte représentant le fait que la date de tir de t_2 doit être inférieure à la date de tir au plus tard de t_3 relativement à sa sensibilisation (lignes 14, 15 et ligne 19). t_3 est sensibilisée par le tir de t_1, ainsi on obtient la contrainte structurelle suivante : $y_2 - y_1 \leq 2$ (ligne 19). Enfin, t_3 est sensibilisée par le tir de t_1 et aucune autre transition n'est sensibilisée en même temps qu'elle, on obtient donc la contrainte structurelle suivante pour t_3 : $1 \leq y_3 - y_1 \leq 2$ (ligne 11). Ainsi, l'enveloppe temporelle Π du support σ s'écrit :

$$\Pi = \begin{cases} 0 \leq y_0 \leq y_1 \leq y_2 \leq y_3 \\ 0 \leq y_0 \leq 2 \\ 0 \leq y_1 - y_0 \leq 2 \\ 1 \leq y_2 - y_1 \leq 3 \\ y_2 - y_1 \leq 2 \\ 1 \leq y_3 - y_1 \leq 2 \end{cases} \tag{2.3}$$

Notons qu'en règle générale cette enveloppe est écrite sous forme canonique, mais pour un souci de lisibilité relatif aux explications précédentes le choix a été fait ici de la présenter sous forme brute.

Avec un raisonnement similaire pour le support $\sigma' = t_0.t_1.t_3.t_2$, on obtient l'enveloppe Π' suivante :

$$\Pi' = \begin{cases} 0 \leq y_0 \leq y_1 \leq y_3 \leq y_2 \\ 0 \leq y_0 \leq 2 \\ 0 \leq y_1 - y_0 \leq 2 \\ 1 \leq y_3 - y_1 \leq 2 \\ y_3 - y_1 \leq 3 \\ 1 \leq y_2 - y_1 \leq 3 \end{cases} \tag{2.4}$$

Proposition 1. *Le polyèdre Π retourné par l'Algorithme 1 pour une séquence σ extraite d'un PTÉS est tel que (σ, Π) est un chemin de ce PTÉS.*

Démonstration. Considérons une exécution $r = \theta_0 t_0 \dots \theta_n t_n$ admissible d'un PTÉS N admettant $\sigma = t_0 \dots t_n$ comme support, et Π le polyèdre calculé par l'Algorithme 1 appliqué à σ et $s_c = C_0 \dots C_n$ la séquence de classes associée au parcours du GC de N en suivant σ. Montrons que le polyèdre calculé correspond à l'enveloppe temporelle de σ. Pour cela, nous allons montrer dans un premier temps que l'exécution r satisfait nécessairement Π (ainsi cela montrera que l'enveloppe temporelle du support pour lequel Π a été calculé est incluse dans Π). Ensuite, en considérant une solution de Π, nous montrerons que l'exécution engendrée par cette solution est admissible pour N (ainsi Π est inclus dans l'enveloppe temporelle de σ).

$\Rightarrow$: Regardons dans un premier temps les contraintes d'ordre. Pour chaque transition t_i du support σ, $\theta_i \geq 0$, donc $y_i - y_{i-1} = \theta_i \geq 0$ et r satisfait les contraintes d'ordre de Π ($y_0 = \theta_0 \geq 0$). Une exécution r s'appuyant sur σ satisfait donc l'ensemble des contraintes d'ordre de Π.

Intéressons nous maintenant aux contraintes structurelles :

- Soit t_i une transition du support, tirée depuis la classe C_i de s_c. Structurellement la date de tir de t_i est contrainte par son intervalle statique $I_s(t_i)$. Le calcul des dates min et max de tir dépend de la classe dans laquelle la transition est sensibilisée (notons C_j cette classe avec $j \leq i$). La transition t_{j-1} du support est la transition tirée pour atteindre C_j, le temps écoulé entre la sensibilisation de t_i et son tir s'exprime donc $y_i - y_{j-1}$. En posant α_{ij} et β_{ij} les bornes de la contrainte obtenue en réduisant le domaine de tir de C_j à la variable représentant t_i, nous obtenons $\alpha_{ij} \leq y_i - y_{j-1} \leq \beta_{ij}$ avec $\alpha_{ij} \geq \downarrow I_s(t_i)$ et $\beta_{ij} \leq \uparrow I_s(t_i)$. Ainsi $(y_i - y_{j-1}) \in I_s(t_i)$. En généralisant ce raisonnement à l'ensemble des transitions d'un support σ, une exécution satisfait donc les contraintes structurelles de la première forme de Π.
- Considérons de nouveau une transition t_i du support (t_i est tirée depuis la classe C_i). La date de tir de t_i relative à la dernière transition tirée (c.-à-d. θ_i) est nécessairement inférieure à la plus petite des bornes supérieures du domaine de tir de toutes les transitions $t_k \neq t_i$ sensibilisées dans C_i mais non tirées. Pour de telles transitions t_k, supposons que C_k est leur classe de sensibilisation dans s_c. Le temps écoulé depuis la sensibilisation de t_k et le tir de t_i correspond à $y_i - y_{k-1}$, ainsi si on pose u' la date au plus tard de t_k selon le domaine de tir de C_k, alors $y_i - y_{k-1} \leq u' \leq \uparrow I_s(t_k)$. Les contraintes structurelles de la deuxième forme sont donc aussi satisfaites.

Une exécution r satisfait donc l'ensemble des contraintes structurelles de Π. Ainsi l'enveloppe temporelle du support σ est incluse dans le polyèdre Π retourné par l'Algorithme 1.

$\Leftarrow$: Considérons un chemin d'un PTÉS de support σ et Π l'application de l'Algorithme 1 à σ. Montrons que Π est inclus dans l'enveloppe temporelle de σ. Considérons un ensemble de réels positifs $\{\theta_0, \dots \theta_n\} \in \mathbb{R}_+^{n+1}$ tel que $\{y_0, \dots y_n\} \models \Pi$ avec $\forall i \in [0, n], y_i = \sum_{k=0}^{i} \theta_k$. Montrons que $r = \theta_0 t_0 \dots \theta_n t_n$ est une exécution admissible. Considérons une transition t_i de σ, sensibilisée par le tir de $t_j, j \leq i$. Le temps écoulé entre la sensibilisation et le tir de t_i est supérieur à la borne inférieure de l'intervalle statique de t_i, et est nécessairement inférieur à sa date de tir au plus tard selon C_j (notons la β_{ij}. $y_i - y_{j-1}$ satisfait donc la contrainte $\alpha_{ij} \leq y_i - y_{j-1} \leq \beta_{ij}$ avec $\alpha_{ij} = \downarrow I_s(t_i)$ et $\beta_{ij} \leq \uparrow I_s(t_i)$. Ainsi la contrainte structurelle sur la sensibilisation de t_i (première forme) est satisfaite. De plus, pour toute transition t_k sensibilisée mais non tirée dans C_i, le temps écoulé entre leur sensibilisation et le tir de t_i est contraint par leur date de tir au plus tard selon C_i. Notons β'_{ik} cette borne supérieure : nous avons $y_i - y_{k-1} \leq \beta'_{ik}$. r est donc admissible pour N, et ainsi Π est compris dans l'enveloppe temporelle de σ.

Nous venons de montrer que Π est égal à l'enveloppe temporelle de σ (inclusion dans les deux sens). La procédure de calcul de Π décrite par l'Algorithme 1 satisfait donc les conditions de la Définition 12. □

Un autre intérêt de cette représentation est de pouvoir étudier le temps écoulé entre le tir de deux transitions d'un support σ. Pour cela nous proposons la notion de réduction d'une enveloppe temporelle (qui consiste en une réduction d'un ensemble de contraintes à un sous-ensemble de ses variables) :

Définition 13. Soit un chemin $\pi = (\sigma, \Pi)$. On appelle réduction de l'enveloppe temporelle Π à deux variables y_i et y_j $(i < n,\ j < i)$ le résultat de l'application suivante $\mathcal{R}_{y_j,y_i} : \mathbb{P} \to \mathbb{P}$ qui à un polyèdre $P \in \mathbb{P}$ associe la projection de P sur les variables y_i et y_j.

Remarque. L'application $\mathcal{R}_{y_j,y_i}$ est un projecteur. En effet $\mathcal{R}_{y_j,y_i} \circ \mathcal{R}_{y_j,y_i} = \mathcal{R}_{y_j,y_i}$.

La réduction est obtenue par changement de variable, nous en proposons un dans l'exemple suivant pour réaliser cette opération qui s'inspire de la méthode d'élimination de Fourier-Motzkin [Sch98].

Exemple 9. Revenons sur le chemin $\pi = (t_0.t_1.t_2.t_3, \Pi)$ du PTÉS Figure 2.5. Intéressons nous à l'application de $\mathcal{R}_{y_0,y_2}$ à Π. Tout d'abord, posons le changement de variable suivant pour les variables du système 2.3 : $y_2 = y_1 + \delta_{21}$. δ_{21} modélise le temps écoulé entre les tirs de t_1 et t_2. Les variables du système 2.3 sont y_0, y_1, y_2 et y_3. Les variables de $\mathcal{R}_{y_0,y_2}$ sont y_0 et y_2. Comme expliqué précédemment, une variable y_i d'une enveloppe temporelle ne dépend que des variables dont l'indice est inférieur à i. Les équations concernant des variables dont l'indice est supérieur à 2 n'influencent donc pas la réduction $\mathcal{R}_{y_0,y_2}$. Ainsi la contrainte $1 \leq y_3 - y_1 \leq 2$ est supprimée. En appliquant le changement de variable δ_{21}, nous obtenons le système suivant :

$$\begin{cases} 0 \leq y_0 \leq 1 \\ 0 \leq y_1 - y_0 \leq 2 \\ y_1 = y_2 - \delta_{21} \\ 1 \leq y_2 - y_1 \leq 3 \\ y_2 - y_1 \leq 2 \\ 1 \leq \delta_{21} \leq 3 \\ \delta_{21} \leq 2 \end{cases} \tag{2.5}$$

En éliminant y_1 et δ_{21} de ce système linéaire, nous obtenons le système suivant :

$$\begin{cases} 0 \leq y_0 \leq 1 \\ 1 \leq y_2 - y_0 \leq 4 \end{cases} \tag{2.6}$$

Nous pouvons donc conclure que pour toute exécution admissible de π, le temps écoulé entre le tir de t_0 et le tir de t_2 est compris entre 1 et 4 unités de temps.

Corollaire 1. *Soit $\Pi \in \mathbb{P}$ l'enveloppe temporelle d'un support σ.*

1. $\{x_0, \dots x_n\} \models \Pi \Rightarrow \{x_j, x_i\} \models \mathcal{R}_{y_j,y_i}(\Pi)$
2. $\{x, x'\} \models \mathcal{R}_{y_j,y_i}(\Pi) \Rightarrow \exists (x_0, \dots, x_{j-1}, x, \dots, x_{i-1}, x', \dots x_n) \in \mathbb{R}_+^{n+1}$, $\{x_0 \dots x_n\} \models \Pi$

L'énumération de tous les supports dont la taille est inférieure ou égale à un entier donné peut être réalisée par une procédure qui termine. Ainsi pour une taille de préfixe donnée nous avons la garantie qu'il existe un nombre fini de chemins dont le support est de taille fixée. Ainsi en combinant cet argument à la propriété **P1**, nous avons la propriété **P0**.

2.3 Conclusion

Nous avons présenté dans ce chapitre le formalisme qui sera utilisé tout au long de ce manuscrit : les réseaux de Petri temporels. Nous avons également présenté une abstraction utile pour traiter le problème d'accessibilité, le graphe des classes, et nous avons montré sa limite pour traiter un problème de comparaison de langages. Pour remédier à cela nous avons proposé une abstraction des préfixes du langage d'un réseau de Petri temporel sous forme de paires de séquences de transitions et de polyèdres appelée *chemin*. Les chemins présentent l'intérêt d'abstraire en un nombre fini de polyèdres l'ensemble des préfixes du langage d'une taille fixée de manière finie, et de plus de ne pas inclure d'exécutions non admissibles pour le réseau de Petri temporel étudié. Une telle abstraction permet d'étudier l'enveloppe temporelle de l'ensemble des exécutions s'appuyant sur une même séquence de transitions depuis l'état initial. Cette abstraction a fait l'objet de la première publication de ce travail de thèse [CSP21]. Le découpage de l'enveloppe en différentes zones satisfaisant des propriétés communes est à la base de nos travaux sur la diagnosticabilité proposés dans le Chapitre 4.

Le chapitre suivant est consacré à la modélisation du problème de diagnostic de motifs temporels dans les réseaux de Petri temporels. Nous définissons ce que sont un système, un motif temporel, et la méthode utilisée pour identifier l'occurrence du motif dans les exécutions du système.

Chapitre 3

Modélisation du problème de diagnostic

Le travail proposé dans ce manuscrit s'inscrit dans le domaine du *Diagnostic de Systèmes à Événements discrets*. Ce chapitre présente la modélisation de ce problème pour un type de SED temporels : les réseaux de Petri Temporels. Nous avons vu dans le Chapitre 2 l'outil de modélisation qui sera utilisé tout au long de ce manuscrit : les PTÉS. Un système et un motif de faute sont modélisés par des PTÉS avec des hypothèses qui vont être précisées dans ce chapitre. En utilisant les langages associés à chacun de ces éléments, nous allons présenter la reconnaissance de l'occurrence du motif dans une exécution du système sous forme d'un problème de *matching*.

3.1 Système et faute considérés

Nous considérons dans ce manuscrit des systèmes modélisés par des PTÉS $\Theta = \langle P^\Theta, T^\Theta, A^\Theta, \Sigma^\Theta, \ell^\Theta, I_s^\Theta, M_0^\Theta \rangle$ partiellement observables, c.-à-d. que l'alphabet du système est partitionné en deux ensembles : Σ_o l'alphabet des symboles associés à des événements observables du système, et Σ_u l'alphabet des symboles associés à des événements non observables du système. Par abus de langage, par la suite une transition observable t_o (resp. non observable t_u) sera une transition labellisée par un événement observable $\ell^\Theta(t_o) \in \Sigma_o$ (resp. par un événement non observable $\ell^\Theta(t_u) \in \Sigma_u$). M_0^Θ représente le marquage initial du système. On appellera *langage du système* Θ le langage $\mathcal{L}(\Theta)$ généré par Θ. La projection des mots du langage d'un système Θ sur l'alphabet observable sera notée par la suite $\mathbf{P}_{\Sigma_o}$.

Le problème de diagnosticabilité traité par la suite est traité comme l'étude du *langage observable* du système Θ :

Définition 14. Le langage observable d'un système est le langage temporel sur l'ensemble $(\mathbb{R}_+ \times \Sigma_o)$ tel que :

$$\mathcal{L}_{obs}(\Theta) = \{\rho_o \in (\mathbb{R}_+ \times \Sigma_o)^*, \exists \rho \in \mathcal{L}(\Theta), \mathbf{P}_{\Sigma_o}(\rho) = \rho_o\}$$

Les hypothèses suivantes sont posées sur les systèmes considérés :

A0 Les intervalles statiques sont fermés bornés, c.-à-d. $\forall t \in T^\Theta$, $I_s^\Theta(t) = [a, b]$, où $(a, b) \in \mathbb{Q}_+ \times (\mathbb{Q}_+^* \setminus \{+\infty\})$.

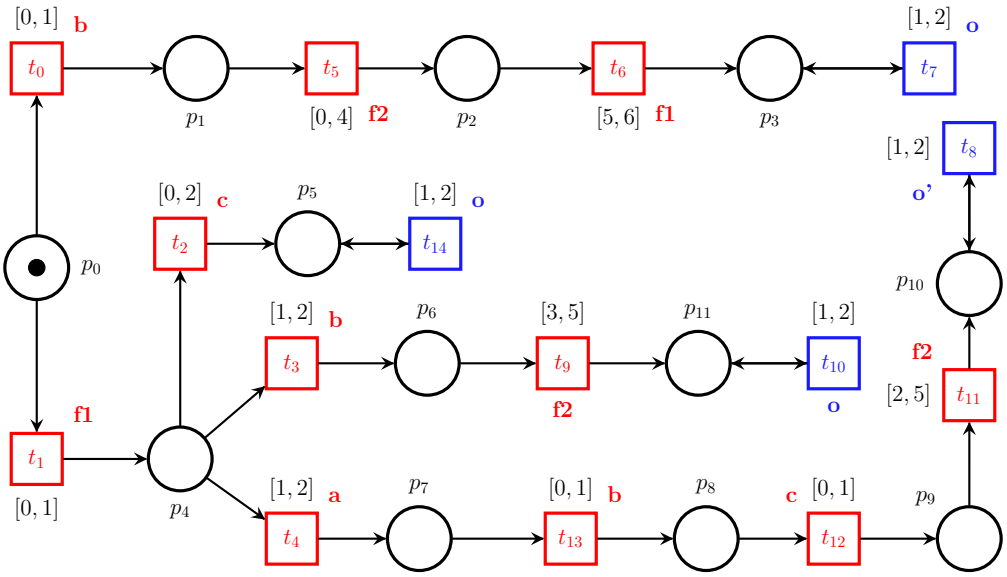


FIGURE 3.1 – Exemple de système partiellement observable Θ_1. Les transitions observables sont ici en bleu et les transitions non observables en rouge. Ce PTÉS est tiré de [Lub+22] et a été temporisé pour ce travail.

A1 Le GC d'un système ne contient pas de boucles non observables, qui sont des boucles de transitions labellisées exclusivement par des événements de Σ_u. De plus le système ne contient pas de marquage bloquant.

A2 Un système n'admet pas d'exécution *Zénon*, c.-à-d. d'exécution dans laquelle un nombre infini de transitions est tiré dans un temps fini.

A3 Aucun intervalle du système n'est ponctuel.

Les hypothèses **A0** et **A1** imposent au système de pouvoir toujours produire une observation en temps fini. Ce sont des hypothèses classiques dans un problème de diagnostic, en cela que pour produire un diagnostic il est nécessaire d'avoir suffisamment d'observations. Ces hypothèses peuvent être reformulées en terme de langage sous la forme suivante : *le langage observable du système est vivant* (Section 1.1.1 page 9). L'hypothèse **A2** évite des comportements qui ne peuvent se produire sur des systèmes réels. Il est à noter que l'existence d'exécutions *Zénon* pourrait poser des problèmes de décidabilité.

Remarque. Le choix ici d'imposer aux intervalles d'être fermés n'est pas une hypothèse inflexible. Nous pensons en effet que certains résultats développés par la suite restent vrais pour des intervalles seulement bornés, mais dans un soucis de simplicité des raisonnements présentés dans cette thèse nous avons posé cette hypothèse.

Comme dit précédemment, nous proposons d'étendre le modèle classique de motifs de fautes (ou *motifs de supervision* [Jér+06]) en intégrant dans celui-ci des contraintes temporelles. Les motifs de faute temporels sont donc des conjonctions d'événements non observables contraints temporellement et dont l'occurrence se vérifie en temps fini depuis le départ du système. À ce stade plusieurs choix s'offrent à nous quant à la définition formelle des motifs : les définir comme un ensemble de séquences temporelles d'événements (c.-à-d. les définir comme des langages) ou les définir comme un PTÉS

(choix fait dans [BPS20] pour permettre la composition du système et du pattern). Le nombre d'exécutions possible étant infini dans le cadre des motifs temporels, l'énumération sous forme de langage n'est pas un choix pertinent. Nous choisissons donc dans ce travail de les définir comme des PTÉS acycliques (c.-à-d. qu'ils ne contiennent pas de boucles), ce qui nous permettra par la suite d'abstraire le motif sous forme de chemins.

Définition 15. Un motif de faute temporel, est modélisé par PTÉS acyclique $\Omega = \langle P^\Omega, T^\Omega, A^\Omega, \Sigma^\Omega, \ell^\Omega, I_s^\Omega, M_0^\Omega \rangle$ et d'un ensemble de marquages Q^Ω où :

1. $\Sigma^\Omega \subset \Sigma_u$
2. $\forall t \in T^\Omega$, $I_s^\Omega(t) = [a, b]$, avec $(a, b) \in \mathbb{Q}_+ \times (\mathbb{Q}_+^* \setminus \{+\infty\})$
3. Q^Ω est l'ensemble des marquages finaux. Les marquages finaux sont des marquages bloquants (aucune transition n'est sensibilisée dans un tel marquage).
4. $M_0 \notin Q^\Omega$
5. Depuis tout marquage accessible de Ω, aucun événement $e \in \Sigma^\Omega$ ne labellise plus d'une transition sensibilisée.
6. Aucune exécution d'un motif n'est de durée nulle.

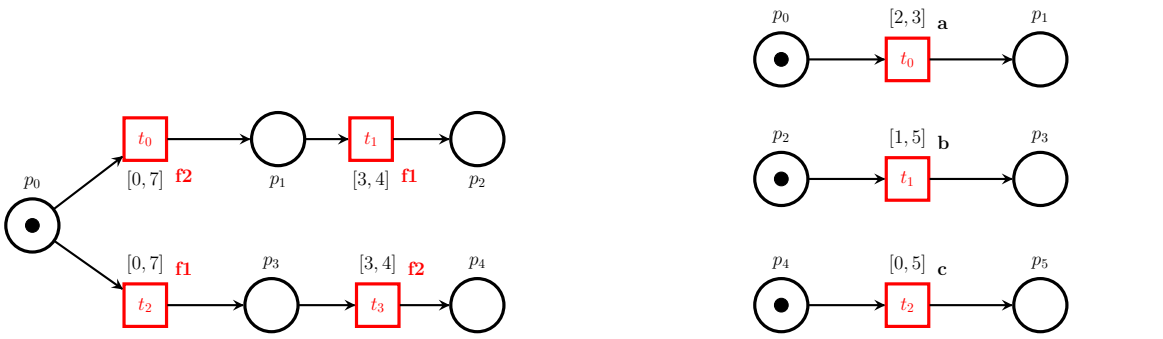
Les motifs considérés sont non observables. Le travail proposé s'adapte sans difficulté à des motifs observables ou partiellement observables, la difficulté du sujet reposant dans l'impossibilité d'observer l'occurrence d'un événement du motif. La condition 2 ainsi que l'hypothèse acyclique imposent au motif de s'exécuter dans une fenêtre de temps finie. La condition 3 impose que lorsque le motif est reconnu (c.-à-d. lorsqu'un marquage final est atteint) sa reconnaissance ne peut être annulée que par un redémarrage du système. Les motifs considérés sont déterministes (condition 5). Enfin, l'occurrence d'un motif ne peut être de durée nulle, cela ne correspondant à aucune réalité physique (condition 6).

Il est important de relever ici l'utilisation de la notion de *marquage final*. L'exécution d'un motif consiste en l'exécution complète de l'une de ses séquences de transitions tirables, c.-à-d. une exécution atteignant un état bloquant du PTÉS le modélisant. Une conséquence directe de cela est que le langage d'un motif ne correspond pas au langage généré par le PTÉS associé, car le langage associé au motif n'est pas clos par préfixe (l'occurrence d'un événement ou d'une partie des événements d'une exécution n'est pas considéré comme une exécution du motif).

Définition 16. Le langage d'un motif de faute temporel correspond à l'ensemble des traces pour lesquelles il existe une exécution menant de l'état initial S_0^Ω vers un état dont le marquage est un marquage final du motif $S^\Omega = \langle M_Q, \emptyset \rangle$, $M_Q \in Q^\Omega$.

$$\mathcal{L}(\Omega) = \{\rho_o^\Omega, \exists r \in (\mathbb{R}_+ \times T^\Omega)^*, S_0^\Omega \xrightarrow{r} \langle M_Q, \emptyset \rangle, M_Q \in Q^\Omega\}$$

Exemple 10. Les Figures 3.2a et 3.2b représentent deux motifs de fautes temporels. Le motif Figure 3.2a admet deux marquages finaux qui sont p_2 et p_4. Le motif Figure 3.2b admet un unique



(a) Motif Ω_1 f_1 suivi de f_2 entre 3 et 4 unités de temps (*t.u.*) au plus 7 *t.u.* après le démarrage du système, ou le contraire

(b) Motif Ω_2 a entre 2 et 3 *t.u.*, b entre 1 et 5 *t.u* et c entre 0 et 5 *t.u.*

FIGURE 3.2 – Exemples de motifs de fautes temporels

marquage final qui est $p_1p_3p_5$. Le langage de 3.2a satisfait le schéma suivant (appelé grammaire en théorie des langages) : $\mathcal{L} = \{(\theta_0 f_1 + \theta_1 f_2) \oplus (\theta_0 f_2.\theta_1 f_2), \theta_0 \in [0,7], \theta_1 \in [3,4]\}$ où $\oplus$ est le *ou exclusif* logique. Quant à 3.2b, toute séquence composée de a, b et c respectant les contraintes indiquées fait partie de son langage (par exemple $2b.1c.0a$ ou encore $2a.2c.1b$).

L'occurrence d'un motif dans une exécution du système est formulée comme un problème de *matching* [PS17]. Il s'agit d'identifier l'occurrence du motif dans une trace en projetant ladite trace sur un sous-ensemble de ses événements. La définition du *matching* peut se faire soit de manière récursive [GPS17], soit avec une approche langage (choix qui a été fait ici).

Définition 17. Une trace $\rho \in \mathcal{L}(\Theta)$ d'un système Θ satisfait un motif Ω noté $\rho \Subset \Omega$ si $\exists \rho' \in \mathcal{L}(\Omega)$ tel que :

1. $\rho' \in \omega(\rho)$
2. Soit ρ_1 le plus petit préfixe de ρ tel que $\rho' \in \omega(\rho_1)$ (pour tout préfixe de ρ_1 différent de lui-même, ρ' n'est pas une sous-suite de ce préfixe). Il n'existe pas de couple $(\rho_2, \rho'') \in ((pref(\rho_1) \setminus \{\rho_1\}) \times \mathcal{L}(\Omega))$ tel que ρ'' est une sous-suite de ρ_2.

Toute trace satisfaisant un motif sera aussi appelée trace *fautive*.

Concrètement une trace satisfait un motif s'il existe une projection de cette trace sur certains de ces événements telle que cette projection soit un mot du langage du motif. La deuxième condition impose que seule la première instance satisfaisant le modèle du motif reconnue est considérée. Cette considération est justifiée par le fait qu'un motif étant utilisé pour modéliser une faute, l'objectif est donc de savoir si la faute a eu lieu lors d'une exécution du système, et non combien de fois elle est apparue.

Exemple 11. Considérons le système Θ_1 présenté Figure 3.1. La trace $\rho_1 = 0f_1.1b.3f_2.2o.2o$ satisfait le motif Figure 3.2a car $\rho'_1 = 0f_1.4f_2$ est une sous-suite de ρ_1 et appartient à $\mathcal{L}(\Omega_1)$. Considérons maintenant la trace $\rho_2 = 0f_1.1b.3f_2.1o.0f_1.4f_2.3o$ (cette trace n'est pas une trace de Θ_1, elle est ici pour illustrer un autre aspect de la Définition 17). Nous avons $\rho_2 \Subset \Omega_1$ car ρ'_1 est

une sous-suite de ρ_2. La trace $\rho'_2 = 5f_1.4f_2$ est également une sous-suite de ρ_2, c'est également un élément de $\mathcal{L}(\Omega_1)$, cependant ρ'_2 ne peut être utilisée pour justifier que $\rho_2 \Subset \Omega_1$ car ρ'_1 satisfait les mêmes critères et ρ'_1 est une sous-suite d'un plus petit préfixe de ρ_2 que ρ'_2.

Avec un tel exemple nous pouvons être tentés de se dire que ρ'_2 n'est pas une justification du matching car $time(\rho'_2) > time(\rho'_1)$. Cette justification n'est pas suffisante. En effet considérons maintenant $\rho_3 = 0f_1.1b.3f_2.0f_2^2.1o.0f_1.4f_2.3o$ (l'exposant 2 est ici intégré pour pouvoir différencier les deux occurrences consécutives de f_2, mais cela ne change pas l'événement). $\rho'_3 = 0f_1.4f_2^2$ est une sous-suite de ρ_3, appartient au langage de Ω_1 et $time(\rho'_3) = time(\rho'_1)$. Pourtant ρ'_3 ne satisfait pas les conditions de la Définition 17, et ne peut donc pas être utilisée comme justification pour dire que $\rho_3 \Subset \Omega_1$.

3.2 Conclusion

Nous avons développé ici la modélisation de notre problème de diagnostic comme un problème de diagnostic dans des systèmes (modélisés par des réseaux de Petri Temporels) de motifs temporels (qui sont une extension des *motifs de supervision* de [Jér+06]) modélisés par des PTÉS admettant des marquages finaux. Le langage d'un motif a été défini en fonction de ces marquages finaux, et sa reconnaissance dans une exécution du système (ou *matching*) a été présentée comme l'existence d'une sous-suite de l'exécution du système appartenant au langage du motif.

Dans le Chapitre 4 nous définissons le problème de *diagnosticabilité de motifs temporels* en nous appuyant sur la notion de *matching* présentée dans ce chapitre. Nous utilisons les *chemins* (Chapitre 2 page 37) pour abstraire le système et le motif temporel, et ainsi proposer une condition nécessaire et suffisante de diagnosticabilité.

Chapitre 4

Diagnosticabilité de motifs temporels

Dans ce chapitre nous allons développer la deuxième contribution de cette thèse, qui consiste en une condition nécessaire et suffisante de diagnosticabilité de motif temporel pour un système modélisé par un PTLS. Des résultats préliminaires de cette analyse ont été publiés dans [Coq+22]. Nous considérons un système Θ et un motif temporel Ω. La définition de diagnosticabilité de [Sam+95] a été adaptée au problème du diagnostic de motifs dans les PTLS dans [PS21]. Cette définition s'inspire du travail de [Tri02] qui, pour des SED temporels, pose une définition de diagnosticabilité relative non pas au nombre d'événements arrivés après l'occurrence d'un événement faute (ou d'un motif), mais relative au temps écoulé depuis l'occurrence de cette faute.

Définition 18. Un système Θ est Ω-diagnosticable si et seulement si $\exists \tau \in \mathbb{R}_+$ tel que $\forall (\rho_1, \rho_2) \in \mathcal{L}(\Theta)^2$, $\rho_1 = \rho_1'.\rho_1''$, $time(\rho_1'') \geq \tau$, $\rho_1' \ni \Omega \wedge \mathbf{P}_{\Sigma_o}(\rho_2) = \mathbf{P}_{\Sigma_o}(\rho_1) \Rightarrow \rho_2 \ni \Omega$.

Informellement, un système est diagnosticable pour un motif temporel donné s'il n'existe pas d'ambiguïtés dans l'ensemble de ses traces, c.-à-d. qu'il n'existe aucune paire de trajectoires infinies partageant la même trace observable dont une des trajectoires contient une occurrence du motif et l'autre n'en contient aucune. L'existence de telles exécutions, appelées *paires critiques*, a été prouvée équivalente à la non diagnosticabilité d'un système pour une faute (un motif temporel étant une sous catégorie de fautes) dans [PCC02] :

Théorème 2. *Un système est Ω-diagnosticable pour un motif de faute temporel Ω si et seulement si l'ensemble de ses traces ne contient pas de paire critique.*

Dans ce chapitre une condition nécessaire et suffisante à l'existence de paires critiques est proposée. Cette condition se base sur la recherche d'ambiguïtés comme intersection de polyèdres représentant des comportements satisfaisant le motif d'une part, et ne le satisfaisant pas d'autre part. Pour vérifier l'existence d'ambiguïtés dans les traces observables de Θ, la première étape est d'identifier les exécutions fautives, c.-à-d. les exécutions où le motif a eu lieu. Nous allons donc dans un premier temps expliquer comment utiliser les chemins présentés dans le Chapitre 2 pour abstraire de manière finie les exécutions fautives du système. Ensuite en se basant sur cet ensemble de chemins *fautifs*, nous proposons une partition de l'enveloppe temporelle d'un chemin fautif pour

laquelle chaque élément de la partition abstrait des exécutions partageant une propriété de diagnostic commune du point de vue du chemin. Nous présentons enfin une condition nécessaire et suffisante pour la Ω-diagnosticabilité d'un système. Cette condition est basée sur la construction d'un graphe réalisant l'intersection entre les langages observables fautifs et non fautifs du système. Cette condition sera prouvée équivalente à l'existence de paires critiques dans les exécutions du système.

4.1 Identification des exécutions fautives

Une exécution est dite fautive si elle contient une occurrence du motif étudié. Identifier si une exécution est fautive revient donc à vérifier si la trace produite par cette exécution satisfait le motif (nous disons par la suite qu'une exécution satisfait le motif). Comme indiqué précédemment, énumérer l'ensemble des exécutions fautives d'un système n'est pas une option (cet ensemble n'étant pas dénombrable *a priori*). Il apparaît donc nécessaire de représenter de telles exécutions sous forme d'un ensemble de sous-ensembles d'exécutions partageant des caractéristiques communes.

Les motifs considérés ici s'exécutent en temps fini. Les systèmes ne contenant pas d'exécutions *Zénon*, l'occurrence du motif s'effectue donc en un nombre fini de transitions. Cela implique donc qu'en considérant l'ensemble des exécutions satisfaisant le motif, quelle que soit l'exécution, il y a un nombre fini de transitions entre le départ de l'exécution et la dernière transition réalisant l'occurrence du motif (c.-à-d. la transition pour laquelle tout préfixe de l'exécution ne contenant pas cette transition ne satisfait pas le motif, et tout préfixe de l'exécution contenant cette transition satisfait le motif). Autrement dit, il existe $n_0 \in \mathbb{N}^*$ tel que pour toute exécution r du système Θ, si r satisfait le motif ($r \ni \Omega$), alors pour tout r' préfixe de r, $|r'| > n_0 \Rightarrow r' \ni \Omega$. Ici $r' \ni \Omega$ est un abus de notation pour signifier $\ell(r') \ni \Omega$. Cette notation sera utilisée par la suite pour un soucis de simplicité.

Remarque. n_0 peut être vu comme le maximum de l'ensemble des tailles minimales pour lesquelles toute continuation d'une exécution fautive est fautive et tout préfixe ne l'est pas.

Le caractère *fautif* d'une exécution se transmet à toutes ses continuations. Autrement dit, une trace satisfait un motif s'il existe un de ses préfixes qui satisfait le motif. L'identification des exécutions fautives est donc ici un problème de recherche de traces de tailles finies (en nombre d'événements). De plus, les chemins présentés dans le Chapitre 2 permettent d'abstraire l'ensemble des exécutions admettant le même support. Le temps étant continu, pour un même support pour lequel il est possible de construire une exécution satisfaisant le motif, il existe plusieurs exécutions satisfaisant le motif. L'étude des exécutions fautives peut donc être réalisée en identifiant les chemins pour lesquels il existe au moins une exécution fautive parmi l'ensemble de leurs exécutions. Il est important de noter ici que pour un chemin donné il n'y a pour le moment aucune garantie que toutes ses exécutions satisfassent le motif. Pour un support donné, un chemin abstrait donc *a priori* des exécutions satisfaisant le motif et d'autres ne le satisfaisant pas.

Pour vérifier l'occurrence d'un motif sur une trace, il faut vérifier s'il existe une sous-suite de cette trace appartenant au langage du motif. Une sous-suite peut être vue comme une projection d'une trace sur un sous-ensemble de ses événements. Considérons une exécution r de Θ telle que $r \ni \Omega$. Il existe $\rho_p \in \omega(\ell(r))$ telle que $\rho_p \in \mathcal{L}(\Omega)$. Il existe donc une séquence de transitions du motif menant de son état initial à un marquage de Q^Ω et une temporisation de cette séquence telle que ρ' la trace produite par cette séquence et cette temporisation est égale à ρ_p. Posons π_Ω le chemin engendré par cette séquence et π_f le chemin du système dont r est extraite. Depuis π_f,

en considérant $(t_i)_{i\in I}$ (I désigne un ensemble d'indices ordonné) les transitions sur lesquelles r est projetée pour produire ρ_p (c.-à-d. les transitions à extraire pour obtenir la sous-suite ρ_p), vérifier si ρ_p est dans $\mathcal{L}(\Omega)$ revient à vérifier si les enveloppes temporelles de π_Ω et de la réduction de π_f aux variables associées aux $(t_i)_{i\in I}$ ont une intersection non vide (en supposant que les événements des $(t_i)_{i\in I}$ et ceux du support de π_Ω correspondent). D'un certain point de vue, cela revient à vérifier la synchronisation entre un chemin particulier du motif et la réduction d'un chemin du système réduit à un sous-ensemble de ses variables (dont le cardinal est égal à la taille du support du chemin du motif considéré). la vérification de l'existence d'une exécution fautive dans un chemin peut être réalisée en vérifiant la synchronisation entre un chemin et la réduction d'un autre.

Définition 19. L'ensemble des chemins du motif est l'ensemble des chemins

$$\mathcal{C}_\Omega = \{(\sigma_\Omega, \Pi_\Omega), C_0^\Omega \xrightarrow{\sigma_\Omega} C_{Q^\Omega}^\Omega\}$$

où $C_{Q^\Omega}^\Omega$ est une classe du GC de Ω dont le marquage appartient à Q^Ω. Les éléments de cet ensemble sont notés π_Ω et sont appelés *chemins du motif*.

Le langage d'un motif est l'ensemble des traces produites par des exécutions menant de l'état initial vers un état dont le marquage est un marquage final du motif (Définition 16). Il parait donc naturel d'abstraire les exécutions du motif avec des chemins dont les supports sont des séquences de transition menant dans le GC du motif de la classe initiale vers une classe *finale* (c.-à-d. une classe associée à un marquage final du motif).

Exemple 12. De retour sur les Figures 3.2a et 3.2b, nous avons :

— pour 3.2a, Il existe deux séquences de transitions différentes menant de la classe initiale vers une classe finale : $t_0.t_1$ et $t_2.t_3$. En associant à ces séquences leur enveloppe temporelle, nous obtenons $\mathcal{C}_\Omega^1 = \{(t_0.t_1, \{0 \leq y_0 \leq 7, 3 \leq y_1 - y_0 \leq 4\}), (t_2.t_3, \{0 \leq y_2 \leq 7, 3 \leq y_3 - y_2 \leq 4\})\}$

— pour 3.2b, l'ensemble des exécutions maximales du motif se découpe en 6 séquences de transitions possibles :

1. $\sigma_\Omega^1 = t_0.t_1.t_2$
2. $\sigma_\Omega^2 = t_0.t_2.t_1$
3. $\sigma_\Omega^3 = t_1.t_0.t_2$
4. $\sigma_\Omega^4 = t_1.t_2.t_0$
5. $\sigma_\Omega^5 = t_2.t_0.t_1$
6. $\sigma_\Omega^6 = t_2.t_1.t_0$

Ce PTLS est composé de trois transitions tirables en parallèle. Cela implique que l'occurrence d'une transition n'influe pas sur l'occurrence des autres structurellement : seul l'ordre de tir va contraindre les dates de tir des transitions entre elles. Quel que soit l'ordre de tir, les contraintes structurelles seront donc les mêmes. Nous avons donc les contraintes structurelles suivantes : $\Pi^{st} = \{2 \leq y_0 \leq 3,\ 1 \leq y_1 \leq 5 \text{ et } 0 \leq y_2 \leq 5\}$. Les contraintes d'ordre seront par contre différentes pour chaque support. Nous obtenons ainsi : $\mathcal{C}_\Omega^2 = \{(\sigma_\Omega^1, \Pi^{st} \cup \{0 \leq y_0 \leq y_1 \leq y_2\}), (\sigma_\Omega^2 \Pi^{st} \cup \{0 \leq y_0 \leq y_2 \leq y_1\}), (\sigma_\Omega^3, \Pi^{st} \cup \{0 \leq y_1 \leq y_0 \leq y_2\}), (\sigma_\Omega^4, \Pi^{st} \cup \{0 \leq y_1 \leq y_2 \leq y_0\}), (\sigma_\Omega^5, \Pi^{st} \cup \{0 \leq y_2 \leq y_0 \leq y_1\}), (\sigma_\Omega^6, \Pi^{st} \cup \{0 \leq y_2 \leq y_1 \leq y_0\})\}$.

Considérons maintenant un chemin π du système. Une occurrence du motif est considérée si elle correspond à une *première occurrence* (voir Définition 17). Il y a donc deux points importants à considérer ici :

1. Dans un chemin π, il peut exister plusieurs synchronisations avec un même chemin du motif, chacune correspondant à une exécution de π où la première occurrence du motif Ω est différente de celle des autres exécutions.
2. Dans un chemin π, il peut exister plusieurs exécutions fautives mais dont la synchronisation avec Ω ne se fait pas avec le même chemin du motif.

Exemple 13. Pour illustrer le paragraphe précédent, prenons un motif très simple, produisant les deux traces suivantes : $5e$ et $10e$. Considérons également le chemin du système suivant : $\pi_e = (t_e.t_e.t_e, \{0 \leq y_0 \leq 5, 0 \leq y_1 - y_0 \leq 5, 1 \leq y_2 - y_1 \leq 5\})$ où chaque transition t_e est labellisée par l'événement e. Considérons les trois traces suivantes correspondant à trois exécutions de ce chemin, $\rho_0 = 5e.3e.3e$, $\rho_1 = 3e.2e.3e$ et $\rho_2 = 1e.1e.3e$. Chacune de ces traces satisfait le motif $5e$, et pour chacune de ces traces l'extraction associée est différente : l'extraction qui engendre l'occurrence du motif dans ρ_0 est composée de la première transition t_e du support de π_e (sous-suite $5e$), l'extraction qui engendre l'occurrence du motif dans ρ_1 est composée de la deuxième transition t_e du support de π_e (sous-suite $(3+2)e$), et celle qui engendre l'occurrence du motif dans ρ_2 est composée de la troisième transition t_e du support de π_e (sous-suite $(1+1+3)e$). De même, il existe au moins une extraction du support de π_e permettant de satisfaire la seconde trace du motif $10e$ (une extraction composée de la troisième transition du support permet avec la trace $4e.5e.1e$ d'obtenir la sous-suite $(4+5+1)e$). Pour un chemin du système, il est donc possible de satisfaire avec différentes sous-suites le même chemin d'un motif, et même de satisfaire deux chemins différents d'un motif.

Définition 20. Soit $\pi = (\sigma, \Pi)$ un chemin de Θ et $\pi_\Omega = (\sigma_\Omega, \Pi_\Omega) \in \mathcal{C}_\Omega$. On appelle *extraction* de π qui satisfait π_Ω une sous-suite $\sigma_s = (t_i)_{i \in [0, |\sigma_\Omega|-1]} \in \omega(\sigma)$ telle que :

— $\ell^\Theta(\sigma_s) = \ell^\Omega(\sigma_\Omega)$

— $(\bigcup_{i=0}^{|\sigma_\Omega|-2} \mathcal{R}_{y_i, y_{i+1}}) \cap \Pi_\Omega \neq \emptyset$ où y_i est la variable associée à la date de tir absolue de t_i, pour $i \in [0, |\sigma_\Omega| - 1]$

Il apparait ici que si un chemin contient dans au moins l'une de ses exécutions l'occurrence du motif, alors pour cette exécution l'occurrence du motif est indissociable de l'extraction responsable de cette occurrence. En effet s'il existe pour un chemin deux manières différentes de réaliser l'occurrence d'un motif, les dates observées pour ces exécutions sont *a priori* différentes.

Avant de présenter les hypothèses que nous posons pour définir le cadre voulu pour ces chemins, nous définissons la notion de *transitions observables inévitables* pour une transition non observable t_u, tirée depuis une classe C. Cette notion correspond aux paquets de transitions observables dont le tir est directement impliqué par le tir de t_u depuis C. Informellement, une transition t_o appartient à

un paquet s'il existe une chaîne de transitions dont le tir d'une transition de la chaîne sensibilise la transition suivante, et dont t_o est la dernière transition de la chaîne et unique transition observable de cette chaîne. La première transition d'une chaîne est soit sensibilisée par le tir de t_u depuis C, soit sensibilisée en parallèle de t_u dans C.

Définition 21. Soit t_u une transition non observable, et C une classe depuis laquelle t_u est tirable. On appelle *transitions observables inévitables* de t_u depuis la classe C l'ensemble $\mathcal{O}(t_u, C)$ qui est composé de sous-ensembles de transitions observables telles que :

— Pour toute transition t_o dans $\mathcal{S}_o \in \mathcal{O}(t_u, C)$, il existe $(t_i)_{i\in[1,k]}$ un ensemble de transitions appelé *chaîne causale* telle que :

1. Soit le tir de t_u sensibilise t_1, soit t_1 est sensibilisée dans C en parallèle de t_u ;
2. Pour tout $j \in [1, k-1]$, le tir de t_j sensibilise la transition t_{j+1} ;
3. $t_k = t_o$ (ainsi dans le cas où $k = 1$, la chaîne causale est composée de t_o uniquement) ;

— Si deux transitions t_o et t'_o appartiennent au même paquet $\mathcal{S}_o$, alors il existe $(t_i)_{i\in[1,k]}$ et $(t'_j)_{j\in[1,l]}$ telles que :

1. Soit $(t_i)_{i\in[1,k]}$ et $(t'_j)_{j\in[1,l]}$ admettent au moins un préfixe commun. Le plus grand préfixe commun (notons sa taille m) est tel que les transitions t_{m+1} et t'_{m+1} ne sont pas en conflit ;
2. Soit $(t_i)_{i\in[1,k]}$ et $(t'_j)_{j\in[1,l]}$ n'ont pas de préfixe commun, et dans ce cas t_1 et t'_1 sont en parallèle. Dans ce cas, les deux chaînes de transitions sont telles que deux transitions de ces chaînes (pour la trajectoire partant de C où ces transitions vont être tirées) ne contiennent pas de conflits (nous dirons que les chaînes n'interfèrent pas).

Un paquet de transitions observables inévitables pour t_u et C correspond aux *premières transitions observables* dont la sensibilisation est soit relative au tir de t_u depuis C, soit relative à des transitions en parallèle de t_u dans C et qui peuvent donc être tirées dans une trajectoire contenant t_u tirée depuis C.

Proposition 2. *Soit t_u une transition non observable et C une classe dans laquelle t_u peut être tirée.*

1. *Il n'existe pas $(\mathcal{S}_o, \mathcal{S}'_o)$ deux éléments de $\mathcal{O}(t_u, C)$ tels que $\mathcal{S}_o \subset \mathcal{S}'_o$.*
2. *Soient $(\mathcal{S}_o, \mathcal{S}'_o)$ deux éléments de $\mathcal{O}(t_u, C)$. Alors il existe $t_o \in \mathcal{S}_o$ et $t'_o \in \mathcal{S}'_o$ tels que $t_o \notin \mathcal{S}'_o$ et $t'_o \notin \mathcal{S}_o$.*

Démonstration. Soient $(\mathcal{S}_o, \mathcal{S}'_o)$ deux éléments de $\mathcal{O}(t_u, C)$.

1. Montrons le premier point par l'absurde. Supposons que $\mathcal{S}_o \subset \mathcal{S}'_o$. Il existe donc $t'_o \in \mathcal{S}'_o$ tel que $t'_o \notin \mathcal{S}_o$. Comme $\mathcal{S}_o \subset \mathcal{S}'_o$, la chaîne causale permettant le tir de t'_o n'est donc pas en conflit avec les chaînes causales des éléments de $\mathcal{S}_o$. Il existe donc une trajectoire permettant de tirer les éléments de $\mathcal{S}_o$ et pas t'_o (sinon $\mathcal{S}_o$ n'existe pas). Comme il n'y a pas de conflit, t'_o n'est pas tirable pour cette trajectoire soit à cause d'un blocage ce qui est contredit par l'hypothèse **A1** (pas de marquages bloquants, Chapitre 2), soit la chaîne causale de

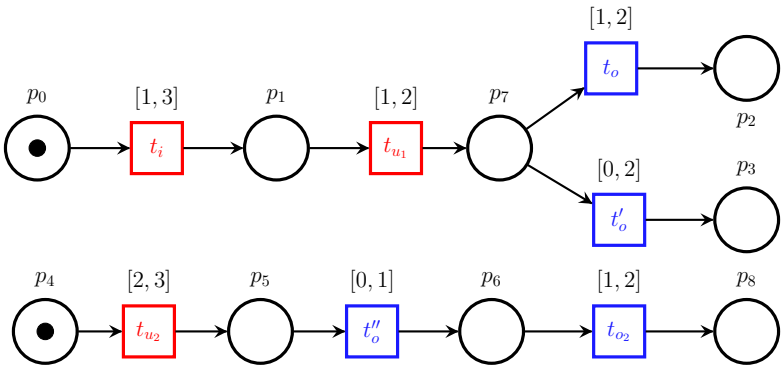


FIGURE 4.1 – Illustration de la notion de *transitions observables inévitables* pour la transition t_i (depuis la classe initiale dans cet exemple)

t'_o contient un nombre non borné de transitions non observables ce qui est contredit par l'hypothèse **A1** (pas de cycles non observables). Ainsi $\mathcal{S}_o = \mathcal{S}'_o$.

2. La seconde propriété découle directement de la première. Si deux ensembles ne sont pas égaux, l'un ne peut être inclus dans l'autre. Ainsi il existe au moins deux transitions qui n'appartiennent qu'à un seul des deux ensembles.

□

Une conséquence directe de la Proposition 2 est que pour une trajectoire du système et une occurrence du tir de t_u depuis C, il existe un unique paquet tiré. Pour se convaincre de cela supposons que pour un tir de t_u depuis C, il existe deux transitions de deux paquets différents qui sont tirées (ces deux transitions étant des premières observables). Les deux transitions étant tirées, leurs chaînes ne peuvent pas être en conflit, sinon l'une des deux transitions n'est pas sensibilisée comme première observable. Ainsi elles appartiennent au même paquet.

Exemple 14. Considérons la situation représentée Figure 4.1. Nous gardons le même code couleur, à savoir rouge pour les transitions non observables et bleu pour les transitions observables. Pour ce PTLS nous considérons la transition t_i et la classe initiale $C_0 = (p_0.p_4, \{t_i = 3, t_{u_2} = 3\})$. Nous allons construire $\mathcal{O}(t_i, C_0)$.

Premièrement, le tir de t_i sensibilise t_{u_1}, et t_{u_1} sensibilise t_o et t'_o. Les transitions t_o et t'_o sont en conflit, elles ne font donc pas partie du même paquet. Il y a donc au moins deux paquets dans $\mathcal{O}(t_i, C_0)$. De plus, t_{u_2} est sensibilisée en parallèle de t_i. Son tir sensibilise t''_o. Cette chaîne causale n'est pas en conflit avec les deux chaînes précédentes, t''_o est donc un paquet avec t_o, et un paquet avec t'_o. Ainsi il existe donc deux paquets de transitions observables inévitables qui sont $\{t_o, t''_o\}$ et $\{t'_o, t''_o\}$.

La transition t_{o_2} n'appartient à aucun paquet mais il existe une transition observable dans la chaîne causale qui la lie au tir de t_i depuis C_0 qui est t''_o.

Dans le cas où il existe une transition observable t_o pour laquelle sa contrainte de tir avec

une transition d'une extraction est ponctuelle (réduite à un seul élément), la notion de transition observable inévitable va permettre de trouver une transition observable qui satisfait cette condition (parmi l'ensemble des transitions observables pour lesquelles cette condition est satisfaite).

Pour avoir la garantie que toutes les occurrences du motif dans les différentes exécutions du système sont abstraites dans un même ensemble de chemins que nous appelons par la suite *chemins fautifs*, nous imposons les propriétés suivantes comme cadre pour la définition des chemins fautifs :

H0 Les chemins fautifs abstraient l'ensemble des exécutions fautives d'une séquence de transition donnée ;

H1 Les chemins fautifs admettent pour toute transition non observable au moins une transition observable qui lui succède dans le support ;

H2 Le support d'un chemin fautif est tel que pour toute transition t_i (non observable) de l'extraction associée, il existe un paquet de $\mathcal{O}(t_i, C_i)$ dont toutes les transitions apparaissent après t_i dans le support du chemin

H3 Les chemins fautifs ont le plus petit support satisfaisant les trois premières conditions.

L'hypothèse **H0** impose que pour un tel chemin π_f du système, un chemin π_Ω du motif et une extraction σ_s associée, σ_s est une sous-suite du support de π_f. L'hypothèse **H1** impose que les dates de chaque transition du support apparaissent dans la projection observable de toute exécution s'appuyant sur ce support. En effet, si une transition est non observable, sa date de tir n'est pas observable pour une exécution mais apparaitra dans la date de tir de la transition observable suivante. Pour se convaincre de cela revenons sur la définition de la projection observable. Si une trace finit par une transition non observable (par exemple $\rho = 2o.2u.1o.1u$), sa trace observable n'est pas différente de son préfixe terminant sur le dernier observable de cette séquence. Dans notre exemple, la trace observable de ρ est $2o.3o$, qui est la même que celle de $\rho' = 2o.2u.1o$, où ρ' est un préfixe de ρ. Ainsi nous avons une perte d'informations. Si la dernière transition du support d"un chemin est une transition de l'extraction associée à ce chemin, en regardant la trace observable il n'est pas possible de statuer sur l'occurrence ou non du motif. La dernière transition d'un chemin doit donc être observable pour garantir de prendre en compte toutes les dates de toutes les transitions d'une trace de ce chemin dans un raisonnement de diagnostic. L'hypothèse **H2** impose qu'il existe un paquet de transitions observables inévitables lié à chaque transition de l'extraction qui appartient nécessairement au support. Ainsi un tel support contient pour toute transition de l'extraction un paquet de transitions observables pour lesquelles nous pouvons construire une contrainte structurelle reliant ces transitions aux transitions de l'extraction. La présence de cette hypothèse est motivée dans la Section 4.2.3. Comme toute continuation d'une exécution satisfaisant un motif satisfait le motif, l'hypothèse **H3** impose de prendre le plus petit support qui satisfait **H0**, **H1** et **H2**. Un tel support finit donc sur une transition de l'un des paquets de transitions observables inévitables de l'une des transitions de l'extraction associée au chemin fautif.

Définition 22. Un *chemin fautif* est un chemin $\pi_f = (\sigma_f, \Pi_f)$ associé à un chemin d'un motif π_Ω et une extraction $\sigma_s = (t_i)_{i\in[0,|\sigma_\Omega|-1]}$ tel que :

1. $\sigma_f = \sigma_1.\sigma_2$
2. $\sigma_s \in \omega(\sigma_1)$
3. $(\bigcup_{i=0}^{|\sigma_\Omega|-2} \mathcal{R}_{y_i,y_{i+1}}) \cap \Pi_\Omega \neq \emptyset$

4. $\forall \sigma'_1 \in pref(\sigma_1) \setminus \{\sigma_1\}$, $\sigma_s \notin \omega(\sigma'_1)$
5. La dernière transition de σ_2 est observable ;
6. Pour toute transition t_i de σ_s, il existe un paquet de transitions de $\mathcal{O}(t_i, C_i)$ (où C_i est la classe de tir de t_i selon cette séquence de transitions dans le GC de Θ) dont les transitions apparaissent dans σ_f après l'occurrence de t_i ;
7. La dernière transition de σ_2 appartient à un paquet de transitions observables inévitables de l'une des transitions de l'extraction σ_s pour sa classe de tir, et cette transition appartient au paquet tiré dans σ_f pour cette transition.

L'ensemble des chemins fautifs associé à un motif Ω est noté $\mathcal{C}_f(\Omega)$.

Proposition 3. *La Définition 22 satisfait les hypothèses **H0**, **H1**, **H2** et **H3**.*

Démonstration. Considérons un chemin π_f satisfaisant la Définition 22.

— $\sigma_s \in \omega(\sigma_1)\ \Pi_\Omega \neq \emptyset \Rightarrow$ **H0**

— La dernière transition du support est la dernière transition de σ_2 qui est observable par définition, donc nous avons **H1**.

— En combinant les conditions 4, 5 et 6 de la Définition 22, σ_1 est le plus petit préfixe de σ_f dont σ_s est une sous-suite. La dernière transition de σ_2 est une transition observable inévitable de l'une des transitions de σ_s (condition 7). De plus la condition 6 impose que σ_f contient pour chaque transition de σ_s un paquet d'observables inévitables. Ce paquet étant unique, s'il existait un préfixe de σ_f différent de lui-même qui satisfasse la Définition 22, il manquerait au moins une transition d'un paquet à ce préfixe (la dernière transition de σ_2). Un tel préfixe ne satisferait donc pas la Définition 22, ce qui est absurde. Ainsi σ_f est le plus petit support satisfaisant les hypothèses **H0**, **H1** et **H2**, ce qui satisfait l'hypothèse **H3**.

□

Corollaire 2. *Toute exécution r telle que $r \ni \Omega$:*

— *Soit admet un préfixe r' tel qu'il existe un chemin fautif $\pi_f \in \mathcal{C}_f(\Omega)$ avec $r' \in \pi_f$;*

— *Soit est le préfixe d'une exécution d'un chemin fautif.*

Le Corollaire 2 met en lumière que quelle que soit r une exécution fautive ($r \ni \Omega$), soit il existe r' un préfixe de r et un chemin fautif π_f tel que $r' \ni \Omega$ et $r' \in \pi_f$, soit r appartient à un chemin dont le support ne contient pas assez de transitions pour satisfaire la définition de chemin fautif, ainsi r est le préfixe d'au moins une exécution r' appartenant à un chemin fautif.

Démonstration. Considérons une exécution r telle que $r \ni \Omega$. Comme $r \ni \Omega$, en posant σ la séquence de transitions support de r, il existe σ_s une extraction de σ et π_Ω un chemin du motif tels que la réduction r' de r aux transitions de son extraction présente la même séquence d'événements que le support de π_Ω et r' satisfait les contraintes de l'enveloppe temporelle de π_Ω.

Considérons $\pi = (\sigma, \Pi)$ le chemin contenant r (Π est l'enveloppe temporelle de la séquence de transitions support de r). π satisfait les points 2 et 3 de la Définition 22, car il existe une

extraction qui est une sous-suite de son support et la réduction de son enveloppe temporelle admet une intersection non nulle avec l'enveloppe temporelle du chemin du motif associé à cette extraction. De plus il existe deux séquences de transitions σ_1 et σ'_2 telles que $\sigma = \sigma_1.\sigma'_2$ et telles que tout préfixe de σ_1 différent de σ_1 n'admet pas σ_s comme sous-suite (point 4 de la Définition 22). Si σ'_2 ne satisfait pas les points 5 et 6 de la Définition 22 :

1. Soit σ'_2 contient trop de transitions pour satisfaire cette définition, et alors il existe un préfixe de σ'_2 qui satisfait la définition. En effet, si σ'_2 contient trop de transitions, alors σ'_2 contient toutes les transitions observables inévitables pour chaque transition de l'extraction. En prenant σ_2 le préfixe de σ'_2 qui contient ces transitions et finit sur l'une d'elles, σ_2 finit sur une transition observable, et ainsi le chemin de support $\sigma_1.\sigma_2$ satisfait la Définition 22.
2. Soit σ'_2 ne contient pas pour chaque transition de σ_s un paquet de transitions observables inévitables des transitions de σ_s (c.-à-d. qu'il existe au moins une transition t_i tirée depuis la classe C_i dans σ telle qu'il n'y a aucun élément $\mathcal{S}_o \in \mathcal{O}(t_i, C_i)$). Dans ce cas, comme le système n'admet pas de blocage, il existe une continuation σ_2 de σ'_2 qui contient toutes ces transitions observables et qui s'arrête sur l'une d'elles. Ainsi le chemin de support $\sigma_1.\sigma_2$ est un chemin fautif.

Nous venons de montrer qu'il est possible à partir de r de trouver un préfixe de r qui appartient à un chemin fautif (cas 1), ou sinon qu'il existe une continuation de la séquence support de r telle que le chemin admettant comme support cette continuation est un chemin fautif, et ainsi r est le préfixe d'une exécution r' appartenant à un chemin fautif (cas 2). Le dernier cas à traiter est celui où le chemin admettant pour support la séquence support de r est un chemin fautif, et dans ce cas, r étant un préfixe de r, r admet un préfixe qui appartient à un chemin fautif. □

4.2 Partition de l'enveloppe temporelle d'un chemin fautif

Le travail de la section précédente a permis d'identifier les chemins fautifs, c.-à-d. les chemins pour lesquels il existe une exécution fautive s'appuyant sur le support du chemin. Il n'y a cependant aucune garantie que l'ensemble des exécutions d'un chemin fautif satisfassent le motif. Nous allons donc étudier ces chemins afin d'exhiber les conditions pour qu'une exécution satisfasse le motif. De plus, pour vérifier l'existence d'ambiguïtés, il est nécessaire que cette identification puisse être effectuée sur les traces observables produites par ces exécutions. Il est important de noter ici que, contrairement au caractère fautif qui se transmet à toute continuation d'une exécution fautive, si le résultat de la fonction de diagnostic produit *ambigu* pour une trace observable donnée, ce résultat n'est pas nécessairement partagé par toute continuation de cette trace.

Nous proposons dans cette section de partitionner l'enveloppe temporelle d'un chemin fautif en un ensemble de polyèdres (c.-à-d. de contraintes temporelles portant sur les dates de tir des transitions observables) étiquetés certain (toute trace satisfaisant ces contraintes observables satisfait le motif), sauf (aucune trace satisfaisant ces contraintes ne satisfait le motif) et ambigu (il existe un couple de traces produisant une trace observable satisfaisant ces contraintes, l'une satisfaisant le motif mais pas l'autre). Dans cette section, nous considérerons un chemin fautif π_f de support σ_f et d'enveloppe temporelle Π_f, d'extraction $(t_i)_{i\in I}$, et π_Ω le chemin du motif associé à ce couple chemin/extraction.

Considérons un chemin fautif π_f. Il existe a priori deux types d'exécutions pour ce chemin : les exécutions fautives et les non-fautives. Pour qu'une exécution r de π_f soit fautive, il faut d'une part qu'elle respecte les contraintes imposées par Π_f (admissibilité pour Θ), et d'autre part que le temps écoulé entre le tir des différentes transitions respecte les contraintes imposées par Π_Ω. Autrement dit, une exécution $r \in \pi_f$ est fautive si :

— $r = \theta_0 t_0 \dots \theta_n t_n$

— $\{y_0, \dots y_n\} \models \Pi_f$ (avec $y_i = \sum_{j=0}^{i} \theta_j$)

— il existe une extraction $(t_i)_{i \in I}$ telle que les variables y_i associées aux transitions de cette extraction appartiennent au polyèdre Π_Ω

Une exécution r d'un chemin fautif π_f est fautive si et seulement si le temps écoulé entre le tir des différentes transitions de l'extraction associée satisfait l'ensemble de contraintes temporelles du chemin du motif associé à ce chemin fautif et à cette extraction (c.-à-d. π_Ω).

Posons Sub la substitution qui aux variables de Π_Ω associe les variables associées à leurs transitions *jumelles* dans σ_f, c.-à-d. les transitions placées au même rang dans la séquence et partageant le même label. En notant $\Pi_\Omega[Sub]$ l'ensemble Π_Ω dans lequel les variables sont substituées en suivant Sub (c.-à-d. les variables associées aux transitions jumelles), on a donc pour une exécution satisfaisant le motif :

$$\{y_0 \dots y_n\} \models (\Pi_f \wedge \Pi_\Omega[Sub])$$

Malgré tout $\Pi_\Omega[Sub]$, projeté sur ses variables observables, ne permet pas de différencier les traces observables fautives ou non. En effet, une trace satisfait un motif s'il existe un chemin du motif pour lequel les événements de son support sont reconnus et si les dates d'occurrence de ces événements sont cohérentes avec les contraintes de $\Pi_\Omega[Sub]$. Or, les contraintes de $\Pi_\Omega[Sub]$ ne concernant que des transitions non observables, et le diagnostic s'effectuant sur des traces observables, il n'est possible de vérifier une telle cohérence que sur la projection observable d'une trace, qui ne contient pas les dates d'occurrence des événements associés à l'occurrence du motif. Considérons c une contrainte de $\Pi_\Omega[Sub]$, et supposons que c contraint le temps écoulé entre le tir d'une transition t_{i-j} et une transition t_i [1].

$$\sigma_f = t_0 \dots t_{o_{i-j-1}} \dots t_{i-j} \dots t_i \dots t_{o_i} \dots t_n$$ [2]

1. Nous allons étudier l'intersection entre c et $\mathcal{R}_{y_{i-j},y_i}(\Pi)$ (Section 4.2.1). Afin de mettre en évidence pour le chemin π_f les conditions pour qu'une exécution satisfasse la contrainte c, il faut analyser l'intersection entre la contrainte c et la réduction $\mathcal{R}_{y_{i-j},y_i}$.
2. La deuxième analyse concerne la réduction $\mathcal{R}_{y_i,y_{o_i}}(\Pi)$ où t_{o_i} est la première transition observable succédant à t_i dans σ_f (Section 4.2.1). La réduction de Π_f à ces transitions représente le temps qui peut s'écouler entre le tir de t_i et celui de t_{o_i}. t_i n'étant pas observable, la date d'occurrence dans une trace observable de π qui informe sur la date d'occurrence de t_i sera celle de t_{o_i}.

1. ici t_{i-j} désigne soit une transition précédant t_i dans l'extraction, soit l'origine du système
2. Le cas $t_0 \dots t_i \dots t_{o_i} \dots t_n$ représente le cas où t_{i-j} et $t_{o_{i-j-1}}$ correspondent à l'origine du système

3. Enfin nous analysons la réduction $\mathcal{R}_{y_{o_{i-j-1}},y_{i-j}}(\Pi)$ où $t_{o_{i-j-1}}$ est la transition observable précédant t_{i-j} dans σ_f [3] (Section 4.2.2). Pour le cas de t_{i-j}, deux choix se présentaient : reporter c sur la dernière transition observable qui la succède, ou sur la première qui la précède. Le problème de la reporter sur l'observable qui lui succède est que nous n'avons a priori aucune garantie qu'il existe une transition observable entre t_{i-j} et t_i dans σ. Il y a donc un risque que les dates de tir de t_i et t_{i-j} soient reportées sur la même transition, ce qui rend la vérification de la satisfaction de c impossible. Nous faisons donc le choix ici de reporter t_{i-j} sur l'observable qui la précède.

L'idée de ce découpage se résume ainsi : $y_i - y_{i-j}$ (où y_i représente la date de tir absolue de t_i dans l'enveloppe temporelle de π_f) n'est pas un temps observable. Par une écriture semblable à la *relation de Chasles* pour les vecteurs de $\mathbb{R}^n$, nous pouvons écrire : $y_i - y_{i-j} = (y_{o_i} - y_{o_{i-j-1}}) - (y_{o_i} - y_i) - (y_{i-j} - y_{o_{i-j-1}})$. Or $(y_{o_i} - y_{o_{i-j-1}})$ est observable. Ainsi en étudiant les valeurs de $(y_{o_i} - y_i)$ et de $(y_{i-j} - y_{o_{i-j-1}})$ en fonction de la valeur de $(y_i - y_{i-j})$, il est possible de tirer des informations sur la satisfaction de c par $(y_i - y_{i-j})$ en observant $(y_{o_i} - y_{o_{i-j-1}})$.

Nous pouvons avant de réaliser cette analyse remarquer un point important : t_{i-j} et $t_{o_{i-j-1}}$ peuvent être pour certaines contraintes non pas une transition de σ_f mais l'origine des temps du système. C'est le cas pour t_{i-j} lorsque la contrainte c ne concerne qu'une variable. Dans ce cas c est une contrainte structurelle relative à une transition sensibilisée dans l'état initial du motif. C'est également le cas de $t_{o_{i-j-1}}$ lorsqu'il n'existe aucune transition observable entre le départ de la séquence σ_f et la transition t_{i-j} dans cette même séquence. Dans ce cas il faut bien penser à prendre en compte le temps écoulé entre l'origine du système et le tir de t_{i-j} pour nos raisonnements.

Cette étude va être réalisée en trois temps. Pour le premier temps nous posons l'hypothèse que pour un couple de transitions (t_{i-j}, t_i) de l'extraction du chemin fautif π_f, il n'existe aucune transition observable du support telle que le temps écoulé entre le tir de cette transition et le tir de t_{i-j} est unique (en posant $\alpha_{o_{i-j-1}}$ et $\beta_{o_{i-j-1}}$ les bornes de $y_{i-j} - y_{o_{i-j-1}}$, nous avons $\alpha_{o_{i-j-1}} < \beta_{o_{i-j-1}}$). De même il n'existe pas de transition observable telle que le temps écoulé entre le tir de t_i et celui de cette transition est fixé (en posant α_{o_i} et β_{o_i} les bornes de $y_{o_i} - y_i$, nous avons $\alpha_{o_i} < \beta_{o_i}$). Nous allons dans ce premier temps nous intéresser au temps écoulé entre le tir de t_{i-j} et le tir de t_{o_i} (points 1 et 2), puis nous étendrons l'analyse au tir de $t_{o_{i-j-1}}$ (point 3). et tel que $\alpha_{o_i} < \beta_{o_i}$ (resp. $\alpha_{o_{i-j-1}} < \beta_{o_{i-j-1}}$). Dans la Section 4.2.3 nous voyons les subtilités amenées par les cas $\alpha_{o_i} = \beta_{o_i}$. Enfin nous synthétisons une partition d'enveloppe temporelle d'un chemin fautif en trois polyèdres distincts.

4.2.1 Analyse de $\mathcal{R}_{y_{i-j},y_{o_i}}$

La Figure 4.2 représente le temps écoulé entre le tir de t_{i-j} et le tir de t_{o_i}. L'axe des abscisses représente la valeur de $y_i - y_{i-j}$. Les bornes α_c et β_c correspondent aux bornes inférieure et supérieure de c. Les bornes $y_i - y_{i-j} = \alpha_i$ et $y_i - y_{i-j} = \beta_i$ sont les bornes retournées par l'application de $\mathcal{R}_{y_{i-j},y_i}$ à Π_f. Les bornes $y_i - y_{i-j} = \alpha_c$ et $y_i - y_{i-j} = \beta_c$ représentent les bornes imposées par c à y_i et y_{i-j}. La zone **fautif** représente donc les comportements de π qui satisfont c, et les zones vertes les comportements pour lesquels c n'est pas satisfaite [4]. Sur l'axe des ordonnées est représentée la valeur de $y_{o_i} - y_i$, les bornes α_{o_i} et β_{o_i} sont les bornes retournées par l'application de $\mathcal{R}_{y_i,y_{o_i}}$ à

3. Dans le cas où t_{i-j} est l'origine du système, $t_{o_{i-j-1}}$ sera également l'origine du système
4. On représente ici le cas où $\alpha_i < \alpha_c$ et $\beta_i > \beta_c$. Un commentaire sur les autres cas sera proposé un peu plus loin.

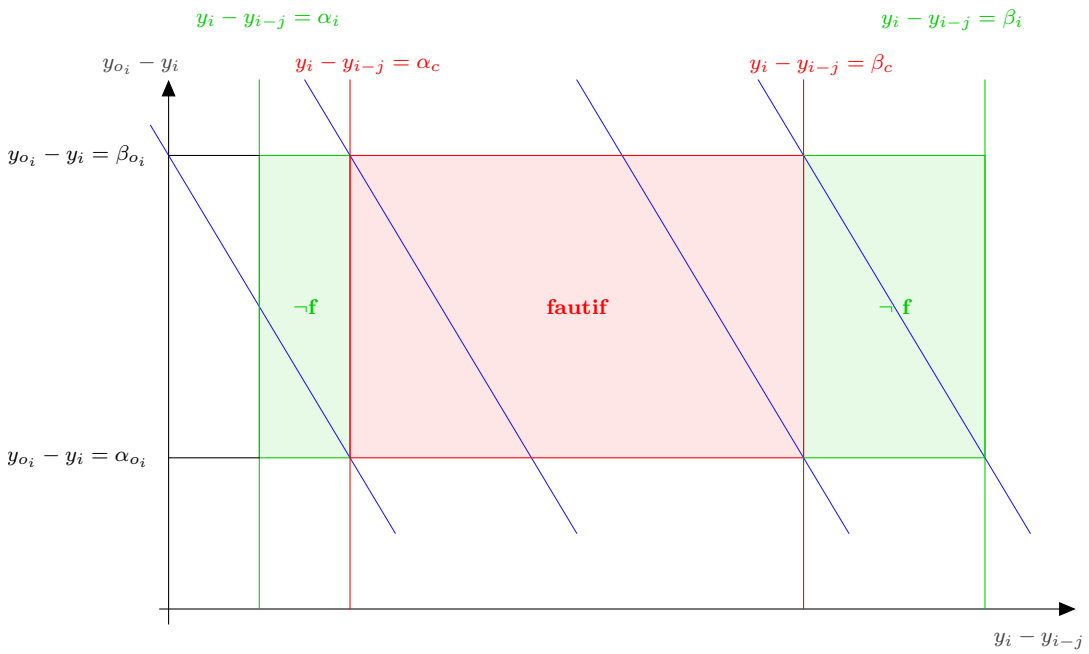


FIGURE 4.2 – Représentation graphique de la contrainte concernant t_i et t_{i-j} en abscisse, et le temps écoulé entre le tir de t_i et t_{o_i} en ordonnée

π. En considérant un point appartenant au polyèdre délimité par les bornes α_i, β_i, α_{o_i} et β_{o_i}, la somme d de son abscisse et de son ordonnée correspond à un comportement admissible du système pour lequel $y_{o_i} - y_{i-j} = d$. Les droites bleues sur la figure représentent des droites d'équation $y_{o_i} - y_{i-j} = d$, où $d \in \mathbb{R}_+$ (toutes ces droites sont parallèles). On remarque donc que selon la valeur de d, il existe 3 cas différents :

1. il existe des d pour lesquels quelle que soit la valeur de $y_i - y_{i-j}$, le comportement représenté ne satisfait jamais c. Cela se caractérise par le fait que la droite engendrée par une telle valeur de d ne coupe qu'une zone **¬f** sur la figure.
2. il existe des d pour lesquels quelle que soit la valeur de $y_i - y_{i-j}$, le comportement représenté satisfait nécessairement c. Cela se caractérise par le fait que la droite engendrée par une telle valeur de d ne coupe que la zone **fautif** sur la figure.
3. il existe des d pour lesquels il existe des valeurs de $y_i - y_{i-j}$ pour lesquelles le comportement représenté satisfait c et des valeurs de $y_i - y_{i-j}$ pour lesquelles le comportement représenté ne satisfait pas c. Cela se caractérise par le fait que la droite engendrée par une telle valeur de d coupe à la fois la zone **fautif** et une zone **¬f** sur la figure. Pour de tels comportements, en connaissant la valeur de $y_{o_i} - y_{i-j}$, il n'est pas possible de conclure si $y_i - y_{i-j}$ satisfait ou non c.

Nous pouvons tirer une première conclusion de cette analyse : il est possible de conclure dans certains cas sur la satisfaction d'une contrainte c sans nécessairement regarder directement la valeur de $y_i - y_{i-j}$.

Les résultats de l'analyse précédente sont synthétisés Figure 4.3 :

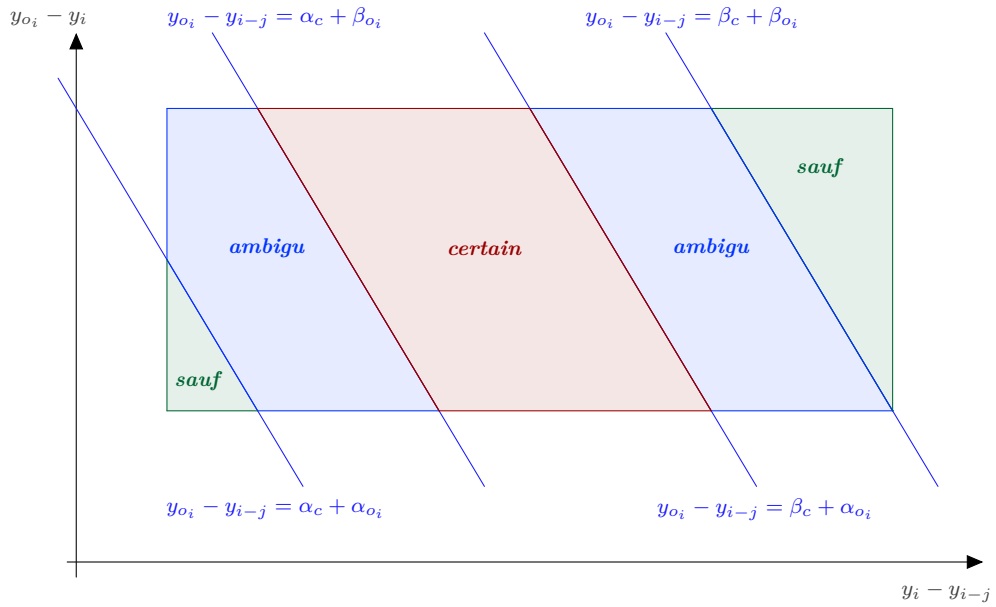


FIGURE 4.3 – Découpage en zones certain, sauf et ambigu de la Figure 4.2

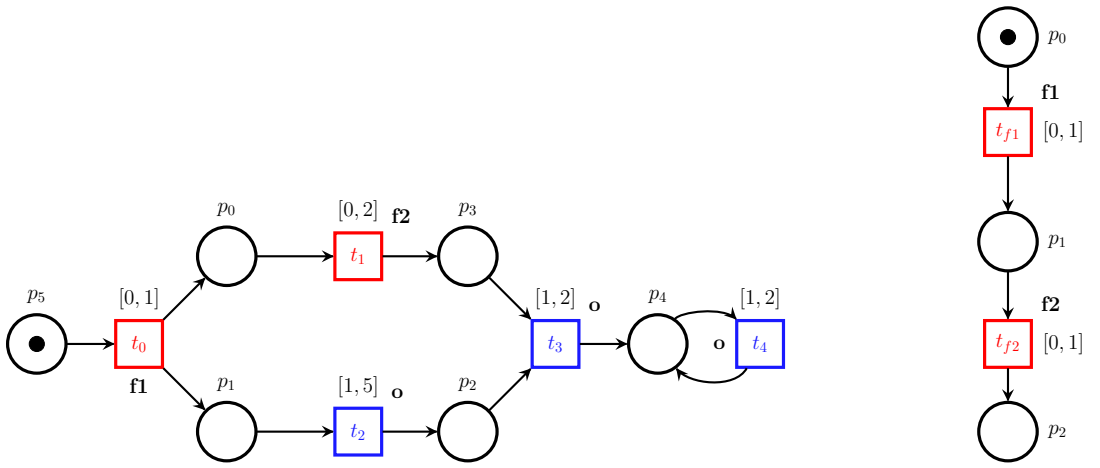
— Si $(y_{o_i} - y_{i-j} < \alpha_{o_i} + \alpha_c) \oplus (y_{o_i} - y_{i-j} > \beta_{o_i} + \beta_c)$, alors quelle que soit la valeur de $y_i - y_{i-j}$, c n'est pas satisfaite (zone **sauf**).
— Si $(\alpha_c + \beta_{o_i} \leq y_{o_i} - y_{i-j} \leq \beta_c + \alpha_{o_i})$, alors quelle que soit la valeur de $y_i - y_{i-j}$, c est satisfaite (zone **certain**).
— Si $(\alpha_c + \alpha_{o_i} \leq y_{o_i} - y_{i-j} < \alpha_c + \beta_{o_i}) \oplus (\beta_c + \alpha_{o_i} < y_{o_i} - y_{i-j} \leq \beta_c + \beta_{o_i})$, il existe des valeurs de $y_i - y_{i-j}$ satisfaisant c et d'autres non (zone **ambigu**).

où $\oplus$ dénote l'opérateur logique *ou exclusif*.

Remarque. Soulignons deux points importants :

1. Il est important de noter que dans ce travail les intervalles de tir du motif sont clos. Ainsi les limites de la zone certain appartiennent à la zone car la contrainte est vérifiée si $y_i - y_{i-j}$ prend la valeur des limites de cette zone. De même, si $y_i - y_{i-j}$ prend la valeur des limites de la zone ambigu *à droite* et *à gauche*, alors il existe une exécution satisfaisant le motif. Cela n'est vrai que parce que les intervalles du motif sont clos. Dans un cas plus général, une contrainte c s'écrirait sous la forme : $\alpha_c \ \#_c \ (y_i - y_{i-j})$, où $\#_c \in \{\leq, <, =, >, \geq\}$. Selon la valeur de $\#_c$ les limites seraient comprises (ou non) dans les différentes zones.

2. La situation représentée Figure 4.2 est telle que le polyèdre engendré par les contraintes de l'enveloppe temporelle de π_f portant sur $(y_i - y_{i-j})$, $(y_{o_i} - y_i)$ et $(y_{o_i} - y_{i-j})$ est un rectangle. Cela veut dire que les bornes α_{i-jo_i} et β_{i-jo_i} de la contrainte portant sur $(y_{o_i} - y_{i-j})$ sont telles que $\alpha_{i-jo_i} \leq (\alpha_i + \alpha_{o_i})$ et $\beta_{i-jo_i} \geq (\beta_i + \beta_{o_i})$. Dans le cas contraire, une droite couperait le coin bas gauche ou/ou haut droite du rectangle engendré. Une telle situation peut être engendrée par l'existence de transitions évoluant en parallèle dans le paquet de transitions



(a) Système Θ_2 (b) Motif Ω_2

FIGURE 4.4 – Un système et un motif illustrant le cas où pour une contrainte c, $\alpha_c \leq \alpha_i$

impliquées dans cette réduction (voir Chapitre 2 page 33), et est représentée Figure 4.4.

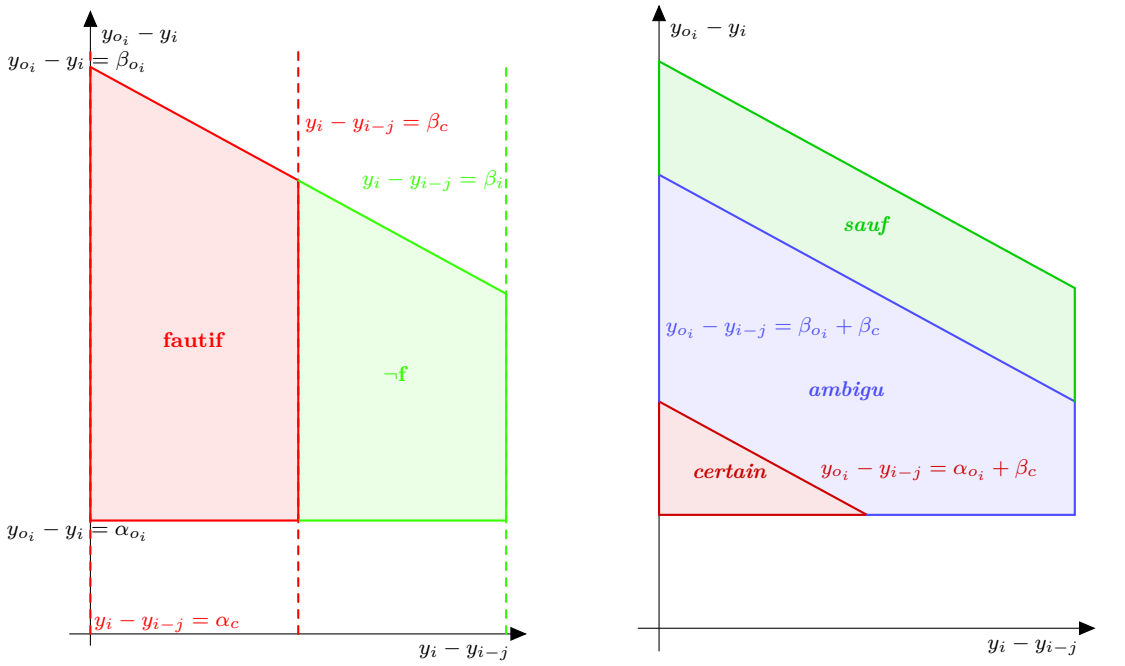
La situation représentée par les Figures 4.2 et 4.3 est le cas où, pour une contrainte c, nous avons $\alpha_i < \alpha_c \leq \beta_c < \beta_i$. Dans ce cas, nous observons la présence d'une zone ambigu et une zone sauf à droite et à gauche de la zone certain. La présence de ces zones est due au fait que pour ce chemin et cette contrainte c, il existe des exécutions pour lesquelles $y_i - y_{i-j}$ est inférieur à α_c. Si aucune exécution ne présente une telle caractéristique, les zones ambigu et sauf ne sont pas présentes.

Un tel cas est illustré par le système et le motif présentés Figure 4.4. Nous considérons le système Θ_2 (Figure 4.4a) et le motif Ω_2 (Figure 4.4b). Pour ce système et ce motif nous allons étudier le chemin fautif $\pi_f = (t_0.t_1.t_2, \{0 \leq y_0 \leq 1,\ 0 \leq y_1 - y_0 \leq 2,\ 1 \leq y_2 - y_0 \leq 5,\ y_2 - y_1 \geq 0\})$. L'unique chemin de Ω_2 est $\pi_{\Omega_2} = (t_{f1}.t_{f2}, \{0 \leq y'_0 \leq 1, 0 \leq y'_1 - y'_0 \leq 1\})$. En proposant la substitution de variables $Sub_2(y'_0) = y_0$ et $Sub_2(y'_1) = y_1$, nous allons nous intéresser à la contrainte suivante de $\Pi_{\Omega_2}[Sub_2]$, $c : \{0 \leq y_1 - y_0 \leq 1\}$. Cette contrainte concerne pour le chemin π_f le tir de t_1 relativement à celui de t_0. Dans ce cas, nous avons $(y_1 - y_0)_{min} = \alpha_1 = \alpha_c$. En effet pour ce système $(y_1 - y_0)_{min} = \downarrow(I(t_0)) = 0 = \downarrow(I(t_{f1}))$. Ainsi, pour toute date de tir proche de la date de tir au plus tôt de t_1, le motif est satisfait. La Figure 4.5 représente ce cas de manière générale en se basant sur la satisfaction de la contrainte c pour le chemin π_f [5]. Les valeurs numériques ayant été utilisées pour la construction sont données Figure 4.6. Comme dit précédemment, il n'existe pas d'exécution non fautive *à gauche*, ainsi nous n'observons ni zone ambigu ni zone sauf sur la Figure 4.5. Nous pouvons ainsi conclure que la présence des zones ambigu et sauf est fonction de α_c, α_i, β_c et β_i.

Quant à la présence de la zone certain, différents cas se présentent :

— Le premier est celui où il y a une zone ambigu et une zone sauf à droite et à gauche (c'est le cas de la Figure 4.3). Dans ce cas, la zone certain est déterminée par $\alpha_c + \beta_{o_i} \leq y_{o_i} - y_{i-j} \leq \beta_c + \alpha_{o_i}$ (dans le cas où $\alpha_c + \beta_{o_i} > \alpha_{o_i} + \beta_c$, cette zone est vide).
— Le second est celui où les zones ambigu et sauf n'existent que d'un côté. Dans ce cas, la zone

5. Le cas où $\beta_c \geq \beta_i$ est le symétrique de ce cas et ne sera donc pas développé ici, mais le résultat apparaitra par la suite.



(a) Satisfaction de c (b) Zones **certain**, **ambigu** et **sauf**

FIGURE 4.5 – Cas où $\alpha_c \leq \alpha_i$, tiré de l'analyse de la Figure 4.4

certain est déterminée par $\alpha_c + \beta_{o_i} \leq y_{o_i} - y_{i-j}$ ou $y_{o_i} - y_{i-j} \leq \beta_c + \alpha_{o_i}$.

— Le troisième cas est celui où $\alpha_c \leq \alpha_i \leq \beta_i \leq \beta_c$. Dans ce cas, toutes les exécutions de π_f satisfont c. Il n'y a donc pour c qu'une zone certain.

Nous pouvons faire la synthèse suivante de l'analyse vue dans cette section :

— Les zones ambigu sont définies par $((\alpha_{o_i} + \alpha_c \leq y_{o_i} - y_{i-j} < \beta_{o_i} + \alpha_c) \wedge (\alpha_c > \alpha_i)) \vee ((\alpha_{o_i} + \beta_c < y_{o_i} - y_{i-j} \leq \beta_{o_i} + \beta_c) \wedge (\beta_c < \beta_i))$.
— Les zones sauf sont définies par $((\alpha_{o_i} + \alpha_c > y_{o_i} - y_{i-j}) \wedge (\alpha_c > \alpha_i)) \vee ((y_{o_i} - y_{i-j} < \beta_{o_i} + \beta_c) \wedge (\beta_c < \beta_i))$.
— La zone certain est définie par $((\alpha_c > \alpha_i) \wedge (\beta_c < \beta_i) \wedge (\alpha_c + \beta_{o_i} \leq y_{o_i} - y_{i-j} \leq \beta_c + \alpha_{o_i})) \oplus ((\alpha_c \leq \alpha_i) \wedge (\beta_c < \beta_i) \wedge (y_{o_i} - y_{i-j} \leq \beta_c + \alpha_{o_i})) \oplus ((\beta_c \geq \beta_i) \wedge (\alpha_i < \alpha_c) \wedge (\alpha_c + \beta_{o_i} \leq y_{o_i} - y_{i-j})) \oplus ((\alpha_c \leq \alpha_i) \wedge (\beta_c \geq \beta_i))$.

4.2.2 Analyse de $\mathcal{R}_{y_{o_{i-j-1}}, y_{o_i}}$

De l'analyse précédente, nous avons pu conclure qu'il est possible, en considérant un chemin donné, d'identifier si certaines exécutions satisfont une contrainte ou non sans avoir l'ensemble des dates de tir de toutes les transitions du support du chemin (nous avons éliminé la date de tir de t_i). Cependant l'analyse précédente ne suffit pas pour résoudre un problème de diagnosticabilité, en cela que les transitions identifiées pour vérifier la satisfaction d'une contrainte c ne sont pas toutes observables. Il n'est donc pas possible en l'état de conclure sur l'occurrence ou non du motif dans une trace observable d'un chemin donné. Dans l'analyse précédente, t_{i-j} n'est pas observable. Nous

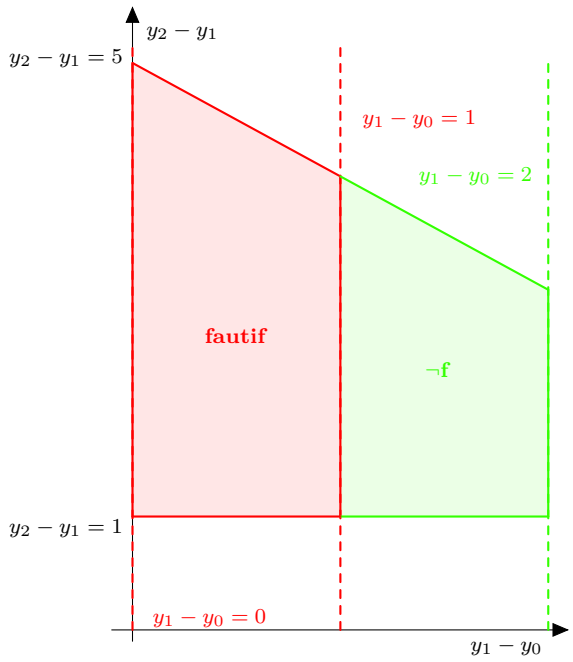


FIGURE 4.6 – Valeurs numériques de l'exemple Figure 4.4

allons nous intéresser au temps écoulé entre t_{i-j} et la transition observable la précédant $t_{o_{i-j-1}}$ (ou le cas échéant avec l'origine du système). En étudiant une trace observable, il est possible de vérifier le temps écoulé entre le tir de $t_{o_{i-j-1}}$ et t_{o_i}. De plus, nous avons déterminé des zones présentant des propriétés de diagnostic (pour un chemin fautif donné) en fonction de la valeur de $y_{o_i} - y_{i-j}$. Nous allons étendre le raisonnement des zones en incluant le temps $y_{i-j} - y_{o_{i-j-1}}$ et voir son influence sur la forme des zones. Nous allons donc étudier la valeur de $y_{o_i} - y_{o_{i-j-1}}$ et chercher à exhiber des zones dont l'ensemble des éléments partagent une même propriété. Dans la suite, $\alpha_{o_{i-j-1}}$ et $\beta_{o_{i-j-1}}$ représentent les bornes inférieure et supérieure du temps écoulé entre le tir de $t_{o_{i-j-1}}$ et le tir de t_{i-j} (ou entre l'origine du système et le tir de t_{i-j} le cas échéant).

La Figure 4.7 représente la contrainte c présentée Figure 4.5b. Dans ce cas l'on rappelle que $\alpha_c \leq \alpha_i$ et $\beta_c < \beta_i$, et que donc les zones sauf et ambigu ne sont présentes qu'à droite (c.-à-d. pour les dates d'occurrence les plus grandes possibles selon l'enveloppe du chemin). L'axe des abscisses représente la contrainte relative à $y_{i-j} - y_{o_{i-j-1}}$ dans $\mathcal{R}_{o_{i-j-1},i-j}(\Pi_f)$ (où Π_f est l'enveloppe temporelle de π_f). L'axe des ordonnées représente la contrainte relative à $y_{o_j} - y_{i-j}$ dans $\mathcal{R}_{i-j,o_i}(\Pi_f)$. Les droites en pointillés verte et rouge représentent les valeurs limites de $y_{o_i} - y_{i-j}$ pour les zones certain et sauf précédemment établies. Les droites en pointillés orange représentent les limites imposées par $\mathcal{R}_{o_{i-j-1},i-j}(\Pi_f)$. En reprenant la logique de l'analyse proposée Figure 4.3, nous étendons ces zones en incluant $y_{i-j} - y_{o_{i-j-1}}$. Nous étendons ainsi les différentes zones en raisonnant sur les droites d'équation $y_{o_i} - y_{o_{i-j-1}} = d_o$.

— La zone certain est déterminée selon le raisonnement suivant : *pour toute exécution satisfaisant cette zone, nous avons la garantie que cette exécution satisfait c.* Autrement dit, pour une valeur donnée $y_{o_i} - y_{o_{i-j-1}}$, nous avons la garantie que $y_i - y_{i-j}$ satisfait c pour une exécution produisant une telle valeur. La borne supérieure de la zone certain déduite

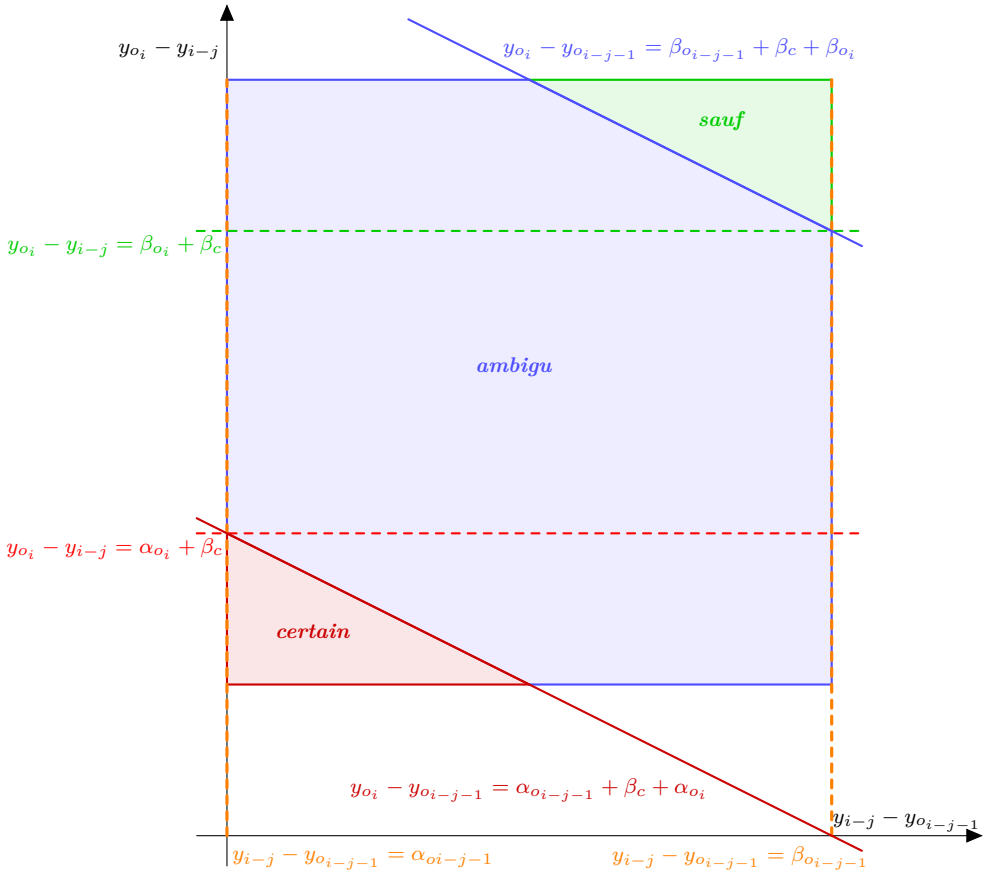


FIGURE 4.7 – Zones **certain, sauf** et **ambigu** pour une contrainte c en fonction de la valeur de $y_{o_i} - y_{o_{i-j-1}}$ (cas $\alpha_c \leq \alpha_i$)

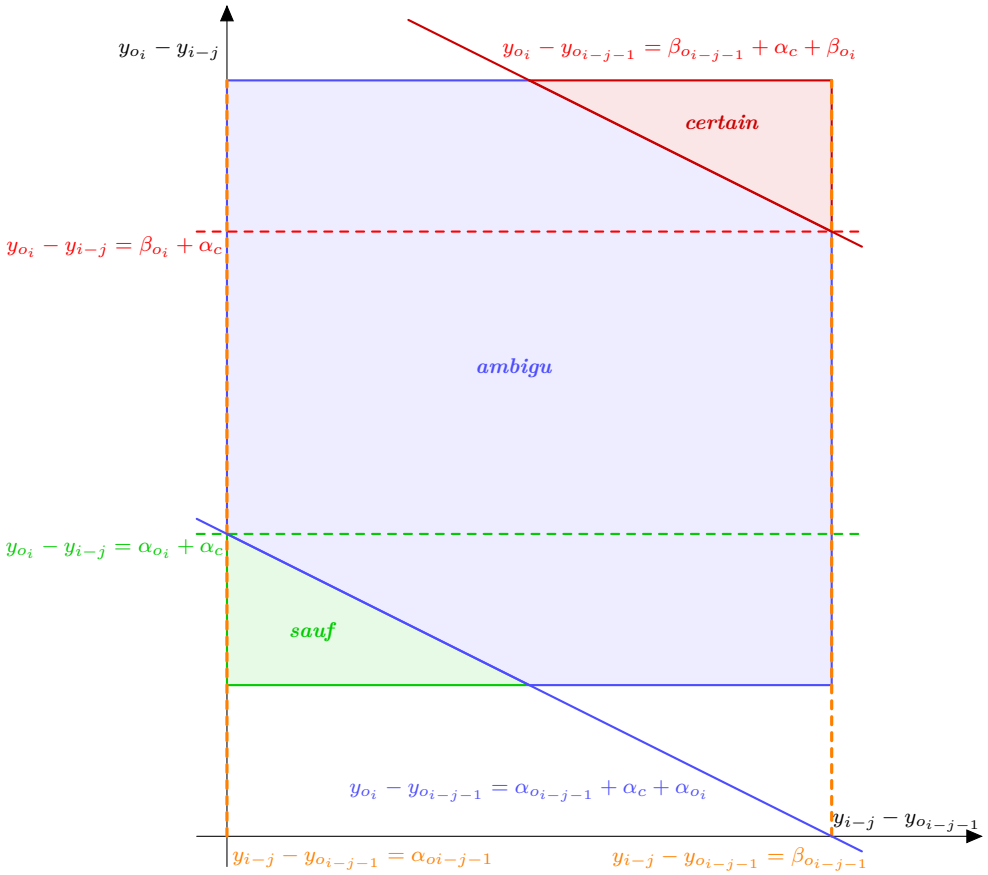


FIGURE 4.8 – Zones **certain, sauf** et **ambigu** pour une contrainte c en fonction de la valeur de $y_{o_i} - y_{o_{i-j-1}}$ (cas $\beta_c \geq \beta_i$)

pour le raisonnement précédent sur $y_{o_i} - y_i$ étant $y_{o_i} - y_{i-j} = \alpha_{o_i} + \beta_c$ (page 63), pour $y_{i-j} - y_{o_{i-j-1}} = \alpha_{o_{i-j-1}}$, nous avons la garantie que pour toutes les exécutions telles que $y_{o_i} - y_{o_{i-j-1}} \leq \alpha_{o_i} + \beta_c + \alpha_{o_{i-j-1}}$, la date d'occurrence de t_i satisfait c (cela permet de définir la borne inférieure de la zone certain pour $y_{o_i} - y_{o_{i-j-1}}$).

— La zone ambigu est déterminée selon le raisonnement suivant : *il existe une exécution satisfaisant c et une exécution ne satisfaisant pas c produisant une même valeur* $y_{o_i} - y_{o_{i-j-1}}$. Nous savons que pour toute valeur de $y_{o_i} - y_{o_{i-j-1}}$ avec $y_{o_i} - y_{i-j} = \beta_c + \beta_{o_i}$ (borne supérieure de la zone ambigu pour $y_{o_i} - y_{i-j}$), il existe au moins une exécution satisfaisant c et une autre ne satisfaisant pas c (page 63). Ainsi, la borne supérieure de la zone ambigu pour $y_{o_i} - y_{o_{i-j-1}}$ est donc $\beta_c + \beta_{o_i} + \beta_{o_{i-j-1}}$.

— La zone sauf est déterminée par toutes les valeurs strictement supérieures à la borne supérieure de la zone ambigu, c.-à-d. pour $y_{o_i} - y_{o_{i-j-1}} > \beta_c + \beta_{o_i} + \beta_{o_{i-j-1}}$.

Un raisonnement similaire au précédent (voir Figures 4.8 et 4.9) nous apporte les bornes des différentes zones pour les cas où $(\alpha_c > \alpha_i) \wedge (\beta_c \geq \beta_i)$ et où $(\alpha_c > \alpha_i) \wedge (\beta_c < \beta_i)$.

Remarque. La Figure 4.9 représente le cas où $\beta_{o_{i-j-1}} + \beta_{o_i} + \alpha_c = \alpha_{o_{i-j-1}} + \alpha_{o_i} + \beta_c$. Ce cas est un cas *particulier* dans le sens où il n'y a qu'une seule date caractérisant la zone certain (la contrainte

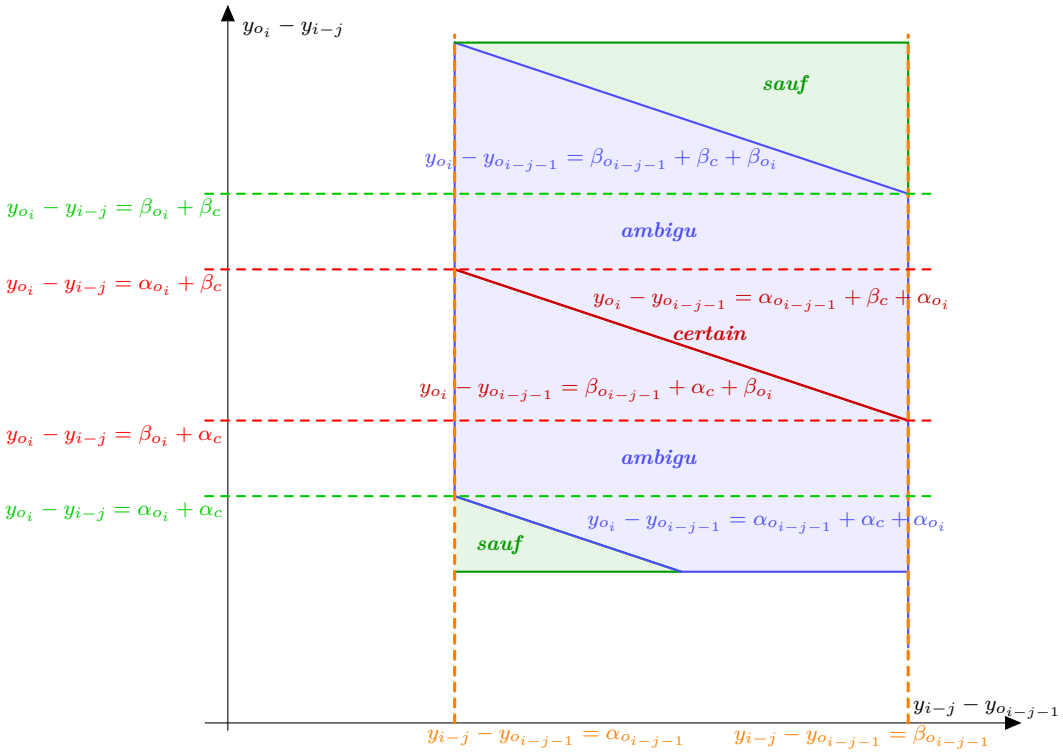


FIGURE 4.9 – Zones **certain, sauf** et **ambigu** pour une contrainte c en fonction de la valeur de $y_{o_i} - y_{o_{i-j-1}}$ (cas $(\beta_c < \beta_i) \wedge (\alpha_c > \alpha_i)$)

définissant cette zone est une égalité).

Nous pouvons donc définir pour une contrainte $c \in \Pi_\Omega[Sub]$ les zones suivantes :

Proposition 4. *Pour une contrainte c, les zones sauf, ambigu et certain sont définies par les contraintes suivantes :*

— *sauf :*

$$\begin{aligned} c^s = &((\alpha_c > \alpha_i) \wedge (y_{o_i} - y_{o_{i-j-1}} < \alpha_{o_i} + \alpha_c + \alpha_{o_{i-j-1}})) \\ \oplus\ &((\beta_c < \beta_i) \wedge (y_{o_i} - y_{o_{i-j-1}} > \beta_{o_i} + \beta_c + \beta_{o_{i-j-1}})) \end{aligned}$$

— *ambigu :*

$$\begin{aligned} c^a = &((\alpha_c > \alpha_i) \wedge (\alpha_{o_i} + \alpha_c + \alpha_{o_{i-j-1}} \leq y_{o_i} - y_{o_{i-j-1}} < \beta_{o_i} + \alpha_c + \beta_{o_{i-j-1}})) \\ \oplus\ &((\beta_c < \beta_i) \wedge (\alpha_{o_i} + \beta_c + \alpha_{o_{i-j-1}} < y_{o_i} - y_{o_{i-j-1}} \leq \beta_{o_i} + \beta_c + \beta_{o_{i-j-1}})) \end{aligned}$$

— *certain :*

$$\begin{aligned} c^c = &((\alpha_c > \alpha_i) \wedge (\beta_c < \beta_i) \wedge (\alpha_c + \beta_{o_i} + \beta_{o_{i-j-1}} \leq y_{o_i} - y_{o_{i-j-1}} \leq \beta_c + \alpha_{o_i} + \alpha_{o_{i-j-1}})) \\ &\oplus ((\alpha_c \leq \alpha_i) \wedge (\beta_c < \beta_i) \wedge (y_{o_i} - y_{o_{i-j-1}} \leq \beta_c + \alpha_{o_i} + \alpha_{o_{i-j-1}})) \\ &\oplus ((\beta_c \geq \beta_i) \wedge (\alpha_i < \alpha_c) \wedge (\alpha_c + \beta_{o_i} + \beta_{o_{i-j-1}} \leq y_{o_i} - y_{o_{i-j-1}})) \\ &\oplus ((\alpha_c \leq \alpha_i) \wedge (\beta_c \geq \beta_i)) \end{aligned}$$

Nous pouvons remarquer ici que ces contraintes sont incompatibles entre elles : $c^s \wedge c^a$, $c^s \wedge c^c$ et $c^a \wedge c^c$ sont nécessairement fausses.

En conclusion de cette analyse, pour une contrainte d'un chemin considéré, la structure temporelle du motif est génératrice d'ambiguïtés, ce qui rajoute une source d'ambiguïté par rapport à ce que l'on peut voir dans la littérature concernant les fautes et les motifs atemporels pour des systèmes labellisés par des réseaux de Petri temporels [BCS15b ; WMS15].

4.2.3 Cas de contraintes ponctuelles entre t_i et t_{o_i} ou entre $t_{o_{i-j-1}}$ et t_{i-j}

L'analyse proposée Section 4.2.1 et 4.2.2 est faite avec l'hypothèse que pour un couple de transitions (t_{i-j}, t_i) de l'extraction du chemin fautif π_f, il n'existe aucune transition observable du support telle que le temps écoulé entre le tir de cette transition et le tir de t_{i-j} est unique (la contrainte de $y_{i-j} - y_{o_{i-j-1}}$ est ponctuelle). De même il n'existe pas de transition observable t_{o_i} telle que le temps écoulé entre le tir de t_i et celui de cette transition est fixé.

Nous pouvons résumer les deux sections précédente ainsi : soient $\tau_{o_{i-j-1}}$ le temps écoulé entre $t_{o_{i-j-1}}$ et t_{i-j}, τ_i le temps écoulé entre les tir de t_{i-j} et de t_i et τ_{o_i} le temps écoulé entre t_i et t_{o_i}. La satisfaction de la contrainte c se vérifie en regardant la valeur de τ_i. En observant le système, nous avons accès au temps écoulé entre $t_{o_{i-j-1}}$ et t_{o_i}, qui s'exprime : $\tau_{o_{i-j-1}} + \tau_i + \tau_{o_i}$. Dans le cas précédent les valeurs de $\tau_{o_{i-j-1}}$ et τ_{o_i} n'étant pas fixées, il y a une incertitude concernant la valeur de τ_i qui donne donc les trois zones de l'analyse précédente.

Par contre, s'il existe une transition t_{o_i} telle que $\alpha_{o_i} = \beta_{o_i}$, alors dans ce cas nous connaissons donc avec certitude τ_{o_i} ($\tau_{o_i} = \alpha_{o_i}$). Il n'y a donc aucune incertitude quant à la valeur de τ_{o_i}, ce qui veut dire qu'il est possible de déterminer pour ce chemin la date de tir de t_i. La zone ambigu présentée en Section 4.2.1 n'est donc pas présente. Pour s'en convaincre, revenons sur la Figure 4.3. Si $\alpha_{o_i} = \beta_{o_i}$, alors le polyèdre représenté devient un segment. Il n'existe ainsi qu'une seule solution pour $y_{o_i} - y_{i-j} = d$, et ainsi aucune ambiguïté n'est générée par la valeur $y_{o_i} - y_{i-j}$. Il en est de même pour $t_{o_{i-j-1}}$. Si $\alpha_{o_{i-j-1}} = \beta_{o_{i-j-1}}$, alors il n'y a plus qu'une seule solution pour $y_{o_i} - y_{o_{i-j-1}} = d$, et ainsi pour cette contrainte aucune ambiguïté n'est amenée par le temps écoulé entre $t_{o_{i-j-1}}$ et t_{i-j}. Ceci se voit sur les expressions posées dans la Proposition 4. En effet, si $\alpha_{o_i} = \beta_{o_i}$ et $\alpha_{o_{i-j-1}} = \beta_{o_{i-j-1}}$, alors c^a se réécrit : $(\alpha_{o_i} + \alpha_c + \alpha_{o_{i-j-1}} \leq y_{o_i} - y_{o_{i-j-1}} < \alpha_{o_i} + \alpha_c + \alpha_{o_{i-j-1}}) \vee (\beta_{o_i} + \beta_c + \beta_{o_{i-j-1}} < y_{o_i} - y_{o_{i-j-1}} \leq \beta_{o_i} + \beta_c + \beta_{o_{i-j-1}})$. Ces deux contraintes sont nécessairement fausses dans ce cas. Cela valide la non présence de zone ambigu.

Cette situation justifie la formulation de l'hypothèse **H2** dans la Section 4.1, et de l'introduction de la notion de *transitions observables inévitables* (Définition 21). Si l'on choisit, pour une transition t_i d'une extraction, la transition t_{o_i} comme étant la première transition observable suivant t_i dans le support σ_f, l'analyse précédente peut définir une zone ambigu, alors qu'il est possible qu'il existe

une transition t'_{o_i} telle que pour t_i et t'_{o_i} nous avons $\alpha'_{o_i} = \beta'_{o_i}$. Dans une telle situation, avec t_{o_i} nous conclurions qu'il existe une zone ambigu non vide pour $y_{o_i} - y_i$, alors que cette zone est vide pour t'_{o_i}, ce qui veut donc dire qu'il n'y a pas d'incertitude quant à la date de tir de t_i.

Proposition 5. *Soient une transition t_i tirable depuis une classe C_i et t_{o_i} une transition observable tirable après t_i, et notons pour une trajectoire dans laquelle t_i est tirée depuis C_i et t_{o_i} est tirée après t_i, α_{o_i} et β_{o_i} les bornes minimale et maximale du temps qui peut s'écouler entre le tir de t_i et le tir de t_{o_i} pour cette trajectoire. $\alpha_{o_i} = \beta_{o_i}$ si et seulement si il existe t'_{o_i} une transition observable inévitable de $\mathcal{S}_o \in \mathcal{O}(t_i, C_i)$ telle que $\alpha'_{o_i} = \beta'_{o_i}$.*

Démonstration. Nous allons montrer ce résultat par double implication.

Premièrement, supposons qu'il existe une transition observable inévitable t'_{o_i} de t_i depuis la classe C_i telle que $\alpha'_{o_i} = \beta'_{o_i}$. Alors en posant $t_{o_i} = t'_{o_i}$, nous avons l'implication inverse.

Supposons maintenant qu'il existe une transition t_{o_i} telle que $\alpha_{o_i} = \beta_{o_i}$. Deux cas s'offrent à nous. Le premier est celui où t_{o_i} est une transition observable inévitable de t_i depuis C_i et alors nous avons l'implication souhaitée. Le second cas est celui où t_{o_i} n'est pas une transition observable inévitable (pour la trajectoire considérée). t_{o_i} est tirée après t_i. Il existe donc une chaîne de transitions dont le tir d'une transition de la chaîne sensibilise la suivante, et la dernière transition de la chaîne est t_{o_i}. Comme t_{o_i} n'est pas une transition observable inévitable pour la trajectoire considérée, il existe une transition observable qui appartient à cette chaîne. Considérons donc t'_{o_i} la première transition observable de cette chaîne. Nous avons par hypothèse $y_{o_i} - y_i = \alpha_{o_i}$. Comme la chaîne ne contient que des transitions reliées entres elles par une relation de sensibilisation (c.-à-d. une contrainte structurelle), la contrainte précédente peut se réécrire $y_{o_i} - y_i = y_{o_i} - y'_{o_i} + y'_{o_i} - y_i = \alpha_{o_i}$. Ainsi $y'_{o_i} - y_i$ est une contrainte ponctuelle. Nous avons donc trouvé une transition observable inévitable telle que la contrainte qui la lie à t_i est ponctuelle. □

Grâce à cette hypothèse, le support d'un chemin fautif contient toutes les transitions observables dont le tir est directement impliqué par une transition de l'extraction du chemin considéré. Ainsi le support d'un chemin fautif contient toutes les premières transitions observables susceptibles, si elles sont choisies pour être t_{o_i} pour une contrainte donnée, de présenter une contrainte ponctuelle du type $\alpha_{o_i} = \beta_{o_i}$.

4.2.4 Sélection des transitions t_{o_i} et $t_{o_{i-j-1}}$

Des paragraphes précédents nous pouvons donc conclure que s'il existe une transition observable t_{o_i} qui peut succéder à t_i pour un chemin donné, alors elle est nécessairement dans le support d'un chemin fautif par définition des chemins fautifs.

Nous allons maintenant discuter pour une contrainte c du choix de $t_{o_{i-j-1}}$ et de t_{o_i}. Nous cherchons ici à traiter un problème de diagnosticabilité. Autrement dit nous cherchons à savoir

s'il existe une zone ambigu pour une contrainte donnée c. Pour établir les différentes zones, nous choisissons donc t_{o_i} et $t_{o_{i-j-1}}$ ainsi :

1. S'il existe t_o succédant à t_i dans σ_f telle que $\alpha_{o_i} = \beta_{o_i}$, alors nous choisissons $t_{o_i} = t_o$.
2. De même, s'il existe t'_o précédant t_{i-j} telle que $\alpha_{o_{i-j-1}} = \beta_{o_{i-j-1}}$, alors nous choisissons $t_{o_{i-j-1}} = t'_o$.
3. Sinon, nous choisissons t_{o_i} le premier observable à succéder à t_i dans σ_f et $t_{o_{i-j-1}}$ comme le dernier observable à précéder t_{i-j}.

Une telle sélection permet, dans le cas où il est possible qu'une contrainte c ne présente pas de zone d'ambiguïté (moyennant un bon choix de t_{o_i} et $t_{o_{i-j-1}}$), de présenter $c^a = \bot$. Ensuite nous pouvons appliquer les formules présentées dans la Proposition 4. S'il existe une zone d'ambiguïté pour chaque observable de σ_f pour c, alors quelles que soient les contraintes calculées, il existe toujours des exécutions dont la trace observable est ambigüe. Comme nous traitons un problème de diagnosticabilité, c'est l'existence d'une zone ambigu qui nous intéresse, car l'existence d'une telle zone révèle l'existence d'ambiguïtés dans les exécutions du système. Ainsi la troisième règle de sélection est satisfaisante de ce point de vue. Nous allons voir dans la Section 4.3.1 l'implication de l'existence dune telle zone.

Si l'objectif était de construire un diagnostiqueur, le raisonnement présenté ci-dessus ne serait pas satisfaisant. En effet, dans le cas où il existe pour toute transition observable de σ_f pour la contrainte c une zone ambigu, la taille de ces zones et les traces observables satisfaisant leurs contraintes ne sont *a priori* pas les mêmes selon les transitions choisies pour t_{o_i} et $t_{o_{i-j-1}}$. Dans l'objectif de construire un diagnostiqueur, une étude de toutes ces zones et de leurs propriétés serait nécessaire, de manière à affiner le résultat produit par le diagnostiqueur.

Nous considérons dans la suite de ce chapitre que les transitions de type t_{o_i} et $t_{o_{i-j-1}}$ ont été sélectionnées selon la méthode proposée dans cette section. Le découpage des différentes contraintes est relatif à ces choix là. Nous allons maintenant synthétiser les résultat des analyses des sections précédentes pour partitionner l'enveloppe observable d'un chemin fautif.

4.2.5 Synthèse de la partition

Comme mentionné précédemment, une exécution est fautive si l'ensemble des contraintes de $\Pi_\Omega[Sub]$ sont satisfaites. Ainsi, pour une trace observable :

— s'il existe une contrainte c pour laquelle c^s est satisfaite, alors nous avons la garantie que le motif n'a pas eu lieu pour cette trace.
— s'il existe une contrainte c telle que c^a est satisfaite, alors nous avons la garantie que l'occurrence du motif est ambigüe pour cette trace.
— si une trace satisfait la contrainte $(\bigwedge_{c\in\Pi_\Omega[Sub]} c^c) = \top$, alors nous avons la garantie que le motif a eu lieu.

Ces informations peuvent être résumées par les ensembles de contraintes suivants pour un chemin fautif π_f, où Π^s_f correspond aux contraintes pour lesquelles nous avons la garantie que le motif n'a pas eu lieu, Π^c_f correspond aux contraintes pour lesquelles nous avons la garantie que le motif a eu lieu, et Π^a_f les contraintes pour lesquelles l'occurrence du motif est ambigüe :

$$\Pi_f^s = \bigvee_{c \in \Pi_\Omega[Sub]} (\Pi_f^o \wedge c^s) \tag{4.1}$$

$$\Pi_f^c = \bigwedge_{c \in \Pi_\Omega[Sub]} (\Pi_f^o \wedge c^c) \tag{4.2}$$

$$\Pi_f^a = \Pi_f^o \wedge \Big(\bigvee_{c \in \Pi_\Omega[Sub]} \big(c^a \wedge \bigwedge_{c' \neq c \in \Pi_\Omega[Sub]} \neg c'^s\big)\Big) \tag{4.3}$$

où Π_f^o désigne la réduction de Π_f à l'ensemble de ses variables associées à des transitions observables.

Lemme 1. *Π_f^s, Π_f^c et Π_f^a forment une partition de Π_f^o.*

$$\Pi_f^a = \Pi_f^o \setminus (\Pi_f^s \vee \Pi_f^c)$$

Démonstration.

$$\begin{aligned}
\Pi_f^o \setminus (\Pi_f^s \vee \Pi_f^c) &= \Pi_f^o \wedge \neg\Big(\big(\bigvee_{c \in \Pi_\Omega[Sub]} (\Pi_f^o \wedge c^s)\big) \vee \big(\bigwedge_{c \in \Pi_\Omega[Sub]} (\Pi_f^o \wedge c^c)\big)\Big) \\
&= \Pi_f^o \wedge \Big((\neg\Pi_f^o \vee \bigwedge_{c \in \Pi_\Omega[Sub]} \neg c^s) \wedge (\neg\Pi_f^o \vee \bigvee_{c \in \Pi_\Omega[Sub]} \neg c^c)\Big) \\
&= \Pi_f^o \wedge \Big(\big(\bigwedge_{c \in \Pi_\Omega[Sub]} \neg c^s\big) \wedge \big(\bigvee_{c \in \Pi_\Omega[Sub]} \neg c^c\big)\Big) \\
&= \Pi_f^o \wedge \Big(\bigvee_{c \in \Pi_\Omega[Sub]} \neg c^c \wedge \Pi_f^o \wedge \neg c^s \bigwedge_{c' \neq c \in \Pi_\Omega[Sub]} \neg c'^s\Big) \\
&= \Pi_f^o \wedge \Big(\bigvee_{c \in \Pi_\Omega[Sub]} c^a \bigwedge_{c' \neq c \in \Pi_\Omega[Sub]} \neg c'^s\Big) \\
&= \Pi_f^a
\end{aligned}$$

□

Exemple 15. Revenons sur le système Θ_1 représenté Figure 3.1 page 44, et considérons les motifs des Figures 3.2a et 3.2b. La séquence de transitions $\sigma_f = t_1.t_3.t_9.t_{10}$ est un support de chemin fautif, car il existe une trace s'appuyant sur ce support qui satisfait le motif $\Omega_1 : 0f_1.1b.3f_2.1o$.

Intéressons nous à la partition de son enveloppe temporelle qui est ici $\Pi_f = \{0 \leq y_1 \leq 1, 1 \leq y_3 - y_1 \leq 2, 3 \leq y_9 - y_3 \leq 5, 1 \leq y_{10} - y_9 \leq 2\}$. Le chemin du motif associé est : $\pi_{\Omega_1} = (t_2^\Omega.t_3^\Omega, \{c_0 : 0 \leq y_2^\Omega \leq 7, c_1 : 3 \leq y_3^\Omega - y_2^\Omega \leq 4\})$, et l'extraction est $t_1.t_9$. Il y a donc deux contraintes à satisfaire et potentiellement pour chaque contrainte les 3 types de zone à définir.

Concernant t_1 (la première transition de l'extraction), sa date de tir dans le système est bornée par 0 et 1. Ces bornes sont incluses dans les bornes de c_0, ainsi toute exécution de π_f satisfait nécessairement c_0. Il n'y a donc qu'une zone certain pour cette transition de l'extraction, soit $0 \leq y_1 \leq 1$.

Intéressons nous maintenant à t_9 (qui est la deuxième transition de l'extraction). t_9 est séparée de t_1 par la transition t_3. Ces trois transitions sont en série, leurs contraintes structurelles sont donc simples à calculer (elles sont données ici par les bornes de leurs intervalles statiques), la réduction aux variables y_9 et y_1 s'exprime ainsi comme la somme des bornes des intervalles de tir des transitions. Nous avons donc $\downarrow(I_s(t_3))+\downarrow(I_s(t_9)) \leq y_9 - y_1 \leq \uparrow(I_s(t_3))+\uparrow(I_s(t_9))$. Ainsi $\mathcal{R}_{y_1,y_9}(\Pi_f) = \{0 \leq y_1 \leq 1, 4 \leq y_9 - y_1 \leq 7\}$. L'intersection de c_1 avec cette réduction donne donc $y_9 - y_1 = 4$. t_1 n'étant pas observable, les contraintes seront projetées sur l'origine de Θ_1.

L'application de $\mathcal{R}_{o,10}$ à Π_f donne : $\mathcal{R}_{o,10}(\Pi) = \{5 \leq y_{10} \leq 10\}$ (t_9 et t_{10} sont en série, le calcul de la réduction suit donc le même raisonnement que pour y_9 et y_1), ainsi $\Pi_f^o = \{5 \leq y_{10} \leq 10\}$. c_0 étant toujours satisfaite, les zones de Π_f sont donc définies par les zones de c_1.

Appliquons les formules données par la Proposition 4 (page 67). Ici $t_{o_{i-j-1}}$ correspond à l'origine du système (il n'y a donc pas de variable associée), t_{o_i} correspond à t_{10} (variable y_{10}), t_{i-j} correspond à t_1 (variable y_1) et t_i correspond à t_9 (variable y_9). Nous avons ici $\alpha_9 = 4$, $\beta_{c_1} = 4$, $\beta_9 = 7$ et $\alpha_{c_1} = 3$ donc $\alpha_9 > \alpha_{c_1}$ et $\beta_9 > \beta_{c_1}$, ce qui implique qu'il n'y a ni zone ambigu ni zone sauf à gauche. Les contraintes seront donc fonction de la variable y_{10}. La contrainte correspondante est donc du type $((\alpha_c \leq \alpha_i) \wedge (\beta_c < \beta_i) \wedge (y_{o_i} \leq \beta_c + \alpha_{o_i} + \alpha_{o_{i-j-1}})$. Ainsi nous avons : $c_1^c = (y_{10} \leq \beta_{c_1} + \alpha_{10} + \alpha_1) = (y_{10} \leq 4+1+0)$, $c_1^a = (\beta_{c_1} + \alpha_{10} + \alpha_1 < y_{10} \leq \beta_{10} + \beta_{c_1} + \beta_1)$ $= (4 + 1 + 0 < y_{10} \leq 2 + 4 + 1)$ et $c_1^s = (y_{10} > \beta_{10} + \beta_{c_1} + \beta_1) = (y_{10} > 2 + 4 + 1)$.

Nous avons donc : $c_1^c : y_{10} \leq 5$, $c_1^a : 5 < y_{10} \leq 7$ et $c_1^s : 7 < y_{10}$.

$$\Pi_f^c = c_1^c \wedge \Pi_f^o = \{y_{10} = 5\}$$
$$\Pi_f^s = c_1^s \wedge \Pi_f^o = \{7 < y_{10} \leq 10\}$$
$$\Pi_f^a = c_1^s \wedge \Pi_f^o = \{5 < y_{10} \leq 7\}$$

Nous remarquons que dans ce cas $\Pi_f^c = c^c$, $\Pi_f^s = c^s$ et $\Pi_f^a = c^a$. Cela n'est dû qu'à la simplicité de l'exemple (la contrainte sur t_1 du motif est ici nécessairement satisfaite).

Pour un chemin dont le support n'admet aucun préfixe qui est le support d'un chemin fautif, la partition précédente se réécrit uniquement avec Π^s, car il n'existe pour de tels chemins aucune exécution satisfaisant le motif, et donc ni zone certaine ni zone ambigüe.

Nous avons vu tout au long de cette section que l'enveloppe temporelle d'un chemin fautif se partitionne en trois types d'ensembles de contraintes : les contraintes pour lesquelles nous avons la garantie en regardant la trace observable d'une exécution que le motif a eu lieu ; celles pour lesquelles nous avons la garantie qu'il n'a pas eu lieu ; et enfin celle pour lesquelles il n'est pas possible de conclure. Nous allons voir dans la section suivante comment utiliser cette partition pour vérifier si un système Θ est Ω-diagnosticable.

4.3 Théorème de diagnosticabilité

Nous avons vu précédemment que la vérification de diagnosticabilité pour une faute simple ou un motif atemporel peut être réalisée en vérifiant l'absence de cycles dont l'étiquetage des deux branches est différent dans le twin-plant associé au problème (voir Chapitre 1 page 9). Pour un système atemporel, la construction du twin-plant s'appuie sur l'énumération des paires de traces différentes

partageant la même trace observable. Ces traces sont associées à des séquences de transitions. Dans le cas temporel, une exécution ne peut pas être considérée comme le pendant d'une séquence de transitions du cas atemporel, car pour une séquence de transitions il existe une infinité d'exécutions. Cependant, la notion de chemin est un candidat intéressant pour servir d'équivalent aux séquences de transitions dans la construction d'un objet similaire au twin-plant atemporel. Nous allons par la suite construire une machine ayant la même fonction que le twin-plant pour tester l'existence de paires critiques. Nous montrons que notre construction est équivalente à une recherche de paires critiques.

Considérons deux chemins dont les supports partagent la même projection observable et terminent par une transition observable. Pour vérifier s'il existe une trace observable commune à ces deux chemins, nous pouvons regarder (moyennant une substitution de variables pertinente) si la projection observable de leurs enveloppes temporelles ont une intersection non vide. Si c'est le cas, il existe une solution commune à ces deux enveloppes (observables), il est donc possible pour ces deux chemins de construire une trace observable qui peut être produite par une de leurs exécutions. Nous avons ainsi un moyen de construire des paires de chemins ayant au moins une trace observable commune (dont la temporisation est une solution commune à la projection observable des enveloppes temporelles de ces deux chemins).

Vient ensuite la question de l'étiquetage des chemins. Cela peut être réalisé en utilisant la partition de l'enveloppe temporelle présentée en Section 4.2. Considérons deux chemins π_1 et π_2 satisfaisant les hypothèses précédentes. D'après la partition précédente, nous pouvons affirmer que l'ensemble des traces observables certaines de chaque chemin satisfont Π^c_1 ou Π^c_2, et les traces observables sauves satisfont Π^s_1 ou Π^s_2. En considérant les intersections $\Pi^c_1 \cap \Pi^s_2$ et $\Pi^s_1 \cap \Pi^c_2$, si ces intersections sont non vides, il existe une trace de chacun de ces chemins qui partagent la même trace observable, l'une correspondant à une exécution satisfaisant le motif, et l'autre non. Ainsi cette intersection non vide d'enveloppe temporelle témoigne de l'existence d'ambiguïtés.

Nous pouvons résumer l'objectif ainsi : *l'objet construit réalise l'intersection entre les langages observables fautif et non fautif.* De plus, pour un état donné de la machine construite nous souhaitons qu'il contienne deux séquences de transitions tirables depuis la classe initiale du GC du système, la première telle qu'il existe une exécution satisfaisant le motif s'appuyant sur cette séquence, et l'autre telle qu'il existe une exécution s'appuyant sur cette séquence ne satisfaisant pas le motif.

Dans la section précédente nous avons mis en évidence pour un chemin fautif la potentielle existence d'ambiguïtés à l'intérieur de ses exécutions (polyèdre Π^a_f). Nous allons donc dans un premier temps étudier pour un chemin fautif les solutions de Π^a_f, puis nous proposerons deux constructions pour une machine à états satisfaisant les critères énoncés dans le paragraphe précédent.

4.3.1 Diagnosticabilité : cas de Π^a_f

Soit $\pi_f \in \mathcal{C}_f(\Omega)$ un chemin fautif du système. Son enveloppe temporelle observable se partitionne en trois éléments distincts : Π^c_f, Π^s_f et Π^a_f. Pour une trace observable dont la temporisation satisfait les contraintes de Π^a_f, il existe deux exécutions la produisant, l'une satisfaisant le motif et l'autre ne le satisfaisant pas. Montrons qu'il est possible, pour une trace observable satisfaisant Π^a_f, de construire une paire critique. Soit $\mathcal{D}(\Theta, \Omega)$ le prédicat *le système* Θ *est* Ω*-diagnosticable.*

Proposition 6. $(\exists \pi_f \in \mathcal{C}_f(\Omega), \mathcal{S}(\Pi_f^a) \neq \emptyset) \Rightarrow \neg\mathcal{D}(\Theta, \Omega)$.

Démonstration. Soit $\pi_f \in \mathcal{C}_f(\Omega)$ tel que pour π_f nous avons $\mathcal{S}(\Pi_f^a) \neq \emptyset$. Nous allons montrer qu'il est possible de construire une paire critique admettant comme préfixes deux exécutions de π_f. $\mathcal{S}(\Pi_f^a) \neq \emptyset$, ce qui implique qu'il existe une contrainte $c \in \Pi_\Omega[Sub]$ (où Π_Ω est l'enveloppe temporelle du chemin du motif associé à π_f et où Sub est une substitution de variables qui associe aux variables de Π_Ω les variables des transitions de l'extraction de π_f qui l'associe au chemin d'enveloppe temporelle π_Ω) pour laquelle, en supposant que les transitions associées à c sont t_i et t_{i-j} (nous reprenons les notations de la section précédente ici), nous avons $\alpha_i < \alpha_c$ ou $\beta_c < \beta_i$. Nous allons démontrer que l'on peut construire une paire critique en supposant $\alpha_i < \alpha_c$ (le raisonnement pour le cas $\beta_c < \beta_i$ est le même et ne sera pas développé ici). Nous posons ici également l'hypothèse que $\alpha_{o_i} < \beta_{o_i}$.

Nous sommes dans le cas où $\mathcal{S}(\Pi_f^a) \neq \emptyset$. Nous reprenons ici les notations de la section précédente, c'est à dire que les transitions observables de référence sont notées $t_{o_{i-j-1}}$ et t_{o_i}, et les transitions relatives à la satisfaction de c sont t_{i-j} et t_i. Il existe donc une contrainte c dans l'enveloppe temporelle du chemin du motif associé à π_f telle que c^a est satisfiable. Cela implique que pour les transitions relatives à c pour laquelle nous n'avons pas $(\alpha_c \leq \alpha_i) \wedge (\beta_c \geq \beta_i) \vee (\alpha_{o_i} = \beta_{o_i}) \wedge (\alpha_{o_{i-j-1}} = \beta_{o_{i-j-1}})$. Aucune de ces deux conditions n'est donc vérifiée pour c. Pour cette démonstration, nous allons supposer que $(\alpha_i < \alpha_c)$ et $(\alpha_{o_i} < \beta_{o_i})$. Un raisonnement similaire permet de traiter l'ensemble des cas pour lesquels la proposition suivante est fausse.

L'articulation du raisonnement est la suivante : nous allons montrer que pour une valeur de $y_{o_i} - yi - j$ particulière, il existe deux exécutions produisant cette valeur telles que la première exécution satisfait c et la seconde ne satisfait pas c. Ainsi, comme nous sommes dans le cas où $\alpha_{o_i} < \beta_{o_i}$, comme t_{o_i} est une transition observable de référence choisie selon ce qui a été présenté dans la Section 4.2.3, il n'existe pas de transition observable telle que le temps écoulé entre t_i et cette transition soit fixé. Ainsi il existe un intervalle continu pour les valeurs possibles de ce temps quelle que soit l'observable reliée à t_{o_i}, et il est donc possible de construire un futur commun pour ces deux traces en s'appuyant sur le chemin π_f. Ainsi nous pouvons construire deux exécutions, l'une satisfaisant c, l'autre ne la satisfaisant pas, et telle que ces deux exécutions partagent la même trace observable. Nous avons ainsi le préfixe d'une paire critique.

Dans le cas où $\alpha_{o_i} = \beta_{o_i}$, nous avons nécessairement $\alpha_{o_{i-j-1}} < \beta_{o_{i-j-1}}$. Ainsi même si le temps entre t_i et t_{o_i} est fixé, le temps passé entre $t_{o_{i-j-1}}$ et t_i n'est pas fixé, et l'on peut donc construire deux exécutions partageant la même trace observable (d'une certaine manière on utilise le fait que le temps est continu pour synchroniser les deux exécutions à partir de t_{o_i}).

Revenons à la contrainte c ($\alpha_i < \alpha_c$ et $\alpha_{o_i} < \beta_{o_i}$). Pour c, nous avons $\alpha_i < \alpha_c$, il existe donc une exécution pour laquelle le temps écoulé entre le tir de t_{i-j} et celui de t_i est égal à α_c. Posons γ_{o_i} le temps écoulé entre le tir de t_i et celui de t_{o_i}. Considérons la droite d'équation $y_{o_i} - y_{i-j} = \gamma_{o_i} + \alpha_c$. Comme $\alpha_i < \alpha_c$ nous avons l'existence de dates pour $y_i - y_{i-j}$ ne satisfaisant pas c. Il existe donc un point de cette droite correspondant à une exécution admissible pour π_f et qui ne satisfait pas c. Posons $\delta = \frac{min(\alpha_c - \alpha_i, \beta_{o_i} - \gamma_{o_i})}{2}$ (cette quantité représente le milieu de la plus petite distance entre le point considéré et les bords du polyèdre, ceci pour s'assurer que l'exécution construite est bien admissible). Il existe une exécution admissible pour laquelle $y_i - y_{i-j} = \alpha_c - \delta$ et $y_{o_i} - y_i = \gamma_{o_i} + \delta$. Nous avons $y_{o_i} - y_{i-j} = \alpha_c + \gamma_{o_i}$, nous avons donc deux valeurs différentes pour $y_i - y_{i-j}$ et $y_{o_i} - y_i$ telles que leur somme donne la même valeur de $y_{o_i} - y_{i-j}$. Nous avons donc deux valeurs,

l'une correspondant à la satisfaction de c et l'autre correspondant à la non satisfaction de c. En suivant ce type de construction, il est possible de construire deux exécutions partageant la même trace observable et telles que pour l'une de ses exécutions c et satisfaite et pour l'autre c n'est pas satisfaite. En suivant le raisonnement énoncé au début de cette preuve, il est possible de construire une même continuation, et ainsi nous pouvons construire un préfixe de paire critique en prenant les mêmes valeurs pour tous les y_i, exceptés ceux pour que $y_i - y_{i-j}$ et $y_{o_i} - y_i$ satisfassent les valeurs proposées ici. Nous pouvons donc construire une paire critique si $\mathcal{S}(\Pi_f^a) \neq \emptyset$, et ainsi le système Θ n'est pas Ω-diagnosticable. □

Pour qu'un système soit diagnosticable, il faut donc que pour chacun de ses chemins fautifs, l'ensemble des solutions de Π_f^a soit vide. De plus nous venons de montrer qu'il est possible de construire une paire critique à partir de la continuation d'un chemin fautif pour lequel l'ensemble Π_f^a de la partition de son enveloppe temporelle est non vide. Nous allons utiliser ce résultat dans la section suivante.

4.3.2 Graphe d'ambiguïtés temporel

Nous venons de voir dans la section précédente que l'existence d'ambiguïtés au sein d'un même chemin engendre des paires critiques. Nous avons également expliqué la formation d'ambiguïtés en considérant deux supports : nous recensons donc deux types d'ambiguïtés. Le premier type regroupe les ambiguïtés induites par deux exécutions s'appuyant sur des séquences de transitions différentes (proche de ce que l'on rencontre dans le cas atemporel), et le second regroupe les ambiguïtés venant d'exécutions s'appuyant sur une même séquence de transitions (existence de solutions pour un chemin fautif de son ensemble Π_f^a). La présence d'une ambiguïté dans l'ensemble des exécutions d'un système (et ce quel que soit son type) témoigne de la non diagnosticabilité du système étudié pour le motif considéré. Nous cherchons donc un objet mathématique synthétisant l'ensemble de ces ambiguïtés de manière à pouvoir vérifier leur éventuelle existence dans les exécutions du système.

Notre proposition pour cela est le *Graphe d'Ambiguïté Temporel* (GAT), qui peut être vu comme un graphe répondant à un problème d'accessibilité. C'est un graphe orienté dont les nœuds contiennent une paire de séquences de transitions et l'intersection d'une partie de l'enveloppe observable des séquences de transitions considérées. La première séquence est le support d'une exécution fautive, la seconde est le support d'une exécution non fautive. Depuis un nœud $s_1 = ((\sigma_1, \sigma_1'), \Pi_1)$ vers un nœud $s_2 = ((\sigma_2, \sigma_2'), \Pi_2)$, il existe un arc entre les deux nœuds s'il existe deux continuations σ_{12} et σ_{12}' de σ_1 et σ_1' avec $\sigma_2 = \sigma_1.\sigma_{12}$ (resp. $\sigma_2' = \sigma_1'.\sigma_{12}'$), de même projection observable et telle que l'intersection entre leurs polyèdres certain et sauf soit non vide (soit Π^c et Π^s pour un chemin, soit Π^a lorsque $\sigma_2 = \sigma_2'$). L'arc sera labellisé par $\mathbf{P}_{\Sigma_o}(\sigma_{12})$. Les états reliés directement à l'état initial $s = ((\sigma_f, \sigma'), \Pi)$ sont tels qu'il existe un chemin fautif π_f admettant σ_f pour support (cela permet que pour chaque nœud du graphe, il existe nécessairement une exécution satisfaisant le motif).

Définition 23. Soit Θ un système et Ω un motif temporel. On appelle *Graphe d'Ambiguïté Temporel* le quadruplet $G(\Theta, \Omega) = (S, s_0, \twoheadrightarrow, \ell^a)$ défini comme suit :

- $S \subset T^{\Theta *2} \times \mathbb{P}$ (T^Θ est l'ensemble des transitions de Θ)
- $s_0 = ((\varepsilon, \varepsilon), \emptyset)$ où ε représente la séquence vide de $T^{\Theta *}$
- $(s_0, s_1) \in \twoheadrightarrow$ (noté $s_0 \twoheadrightarrow s_1$) si :

- $s_1 = ((\sigma_1, \sigma'_1), \Pi_{s_1})$ avec Π_1 (resp. Π'_1) l'enveloppe temporelle associée à σ_1 (resp. σ'_1)
- $\exists \pi_f = (\sigma_1, \Pi_f) \in \mathcal{C}_f(\Omega)$
- $\Pi_{s_1} \neq \emptyset$
- $\mathbf{P}_{\Sigma_o}(\sigma_1) = \mathbf{P}_{\Sigma_o}(\sigma'_1)$
- $\sigma'_1 = \sigma_f \Rightarrow \Pi_{s_1} = \Pi^a_f$, $\sigma'_1 \neq \sigma_f \Rightarrow \Pi_{s_1} = \Pi^c_f \cap \Pi'^s_1$

— $\forall s_1 \neq s_0$, $s_1 \twoheadrightarrow s_2$ si :

- $s_1 = ((\sigma_1, \sigma'_1), \Pi_{s_1})$, $s_2 = ((\sigma_2, \sigma'_2), \Pi_{s_2})$
- $\exists (\sigma_{12}, \sigma'_{12}) \in (T^{\Theta^{*2}}_{uo})$, $\exists (t_o, t'_o) \in T^{\Theta^2}_o$, $\sigma_2 = \sigma_1.\sigma_{12}.t_o$, $\sigma'_2 = \sigma'_1.\sigma'_{12}.t'_o$ et $\ell^\Theta(t_o) = \ell^\Theta(t'_o)$
- $\sigma_2 = \sigma'_2 \Rightarrow \Pi_{s_2} = \Pi^a_2$
- $\Pi_{s_2} \neq \emptyset$
- $\sigma_2 \neq \sigma'_2 \Rightarrow \Pi_{s_2} = \Pi^c_2 \cap \Pi'^s_2$

— $\ell^a(s_0, s_1) = \mathbf{P}_{\Sigma_o}(\sigma_1)$

— $\ell^a(s_1, s_2)_{s_1 \neq s_0} = \ell^\Theta(t_o)$ où $\sigma_2 = \sigma_1.\sigma_{12}.t_o$

où T^Θ_{uo} (resp. T^Θ_o) est l'ensemble des transitions du système labellisées par un événement non observable (resp. observable).

De cette définition découle les propriétés suivantes pour les états d'un GAT :

Proposition 7. *Soient Θ un système, Ω un motif et $G(\Theta, \Omega)$ le GAT associé.*

1. $s = ((\sigma, \sigma'), \Pi_s) \in S \Rightarrow \boldsymbol{P}_{\Sigma_o}(\sigma) = \boldsymbol{P}_{\Sigma_o}(\sigma')$
2. $s = ((\sigma, \sigma'), \Pi \cap \Pi') \in S \Rightarrow \exists (r, r') \in (\sigma, \Pi) \times (\sigma', \Pi')$, $r \Subset \Omega$, $r' \not\Subset \Omega$
3. $\forall s \in S \setminus \{s_0\}$, $\Pi_s \neq \emptyset$

Démonstration. 1 : Soit $s \in S$, $s = ((\sigma, \sigma'), \Pi_s)$. $s_0 \twoheadrightarrow s \Rightarrow \mathbf{P}_{\Sigma_o}(\sigma) = \mathbf{P}_{\Sigma_o}(\sigma')$ (cf Définition 23). Si s n'est pas en relation avec s_0, on montre par récurrence que σ et σ' partagent la même trace observable, en utilisant comme initialisation s_2 tel que $\exists s_1 \in S, s_0 \twoheadrightarrow s_1 \twoheadrightarrow s_2$. En effet pour s_2 en posant $\sigma_2 = \sigma_1.\sigma_{uo}.t_o$ et $\sigma'_2 = \sigma'_1.\sigma'_{uo}.t'_o$, on a $\mathbf{P}_{\Sigma_o}(\sigma_2) = \mathbf{P}_{\Sigma_o}(\sigma_1).\ell^\Theta(t_o)$ et $\mathbf{P}_{\Sigma_o}(\sigma'_2) = \mathbf{P}_{\Sigma_o}(\sigma'_1).\ell^\Theta(t'_o)$. Or par définition $\mathbf{P}_{\Sigma_o}(\sigma_1) = \mathbf{P}_{\Sigma_o}(\sigma'_1)$ et $\ell^\Theta(t_o) = \ell^\Theta(t'_o)$, d'où $\mathbf{P}_{\Sigma_o}(\sigma_2) = \mathbf{P}_{\Sigma_o}(\sigma'_2)$.

2 : Soit $s = ((\sigma, \sigma'), \Pi_s) \in S$. Si Π_s est de la forme $\Pi^c \cap \Pi'^s$, alors il existe une exécution admettant comme support σ dont la trace observable satisfait Π^c. De même, il existe une exécution admettant comme support σ' dont la trace observable satisfait Π'^s. Si $\Pi_s = \Pi^a$, alors pour toute trace observable satisfaisant Π^a il existe une exécution r fautive et une exécution r' non fautive s'appuyant sur σ produisant une telle trace. Nous avons $r \Subset \Omega$ et $r' \not\Subset \Omega$

3 : Vient directement de la Définition 23. □

Par construction, si un état est accessible depuis l'état initial, pour la paire de séquences de transitions de cet état, il existe une ambiguïté dans les exécutions du système s'appuyant sur ces deux séquences de transitions. En effet, si le polyèdre associé à l'état est non vide, cela implique qu'il existe une temporisation commune pour une trace observable produite par une exécution s'appuyant sur σ et pour une autre s'appuyant sur σ'. De plus, si $\sigma \neq \sigma'$, quelle que soit l'exécution s'appuyant sur σ, nous avons la garantie que cette exécution satisfait le motif (Π^c), et quelle que soit l'exécution s'appuyant sur σ', nous avons la garantie que cette exécution ne satisfait pas le motif (Π'^s). Le cas $\sigma = \sigma'$ a été traité dans la section précédente (Proposition 6). Ainsi, si le GAT contient un nombre infini d'états accessibles, il est possible de construire une paire critique. De

même si un système est Ω-diagnosticable, nécessairement le GAT associé admet un nombre d'états fini.

Il est important de noter que le GAT est tel qu'une paire critique admet nécessairement un préfixe parmi les traces du GAT, mais il ne permet pas d'énumérer toutes les paires critiques. Si l'on considère deux séquences de transitions différentes partageant la même projection observable et admettant un préfixe commun qui est le support d'un chemin fautif, le GAT ne calcule que les intersections entre les zones Π^c de l'un et Π^s de l'autre, le GAT ne calcule pas l'intersection entre un polyèdre Π^a et un polyèdre Π^c par exemple alors que cette intersection peut contenir des paires critiques. Cela ne pose pas problème dans une analyse de diagnosticabilité. En effet la présence d'un Π^a_f non vide pour un chemin fautif engendre une famille de paires critiques. Quelle que soit la quantité de paires critiques engendrées, l'information utile ici est de savoir s'il existe un tel Π^a_f pour un système et un motif donné, et non toutes les paires critiques engendrées.

Proposition 8. *$\mathcal{D}(\Theta, \Omega) \Leftrightarrow G(\Theta, \Omega)$ admet un nombre fini d'états accessibles.*

En pratique, construire le GAT pour un problème permet seulement dans le cas où la construction termine de conclure que le système est diagnosticable pour ce motif. Lorsque l'on s'intéresse à la construction du twin-plant [Jia+01], le problème du nombre d'états infini est évité par l'existence de boucles dans les états visités. Tels que sont définis les états d'un GAT, il n'existe pas de boucles dans ces différents états à proprement parler (les séquences de transitions et l'enveloppe temporelle grossissent à mesure que l'on continue d'ajouter des transitions). Malgré tout, les séquences de transitions d'un état du GAT peuvent représenter des parcours de cycles dans le GC du système étudié. Dans un GC, si une boucle est accessible, il existe une exécution qui boucle à l'infini entre certains états contenus dans les classes composant cette boucle (dans notre cadre d'hypothèse comme les intervalles statiques considérés sont clos, ce bouclage peut passer par les mêmes états). Cela veut donc dire que dans un GAT, si pour une paire (σ, σ') la séquence de paires de classes associée contient deux fois une même paire, il existe deux exécutions accessibles qui partagent la même trace observable et qui décrivent une boucle répétable *à l'infini*. Le nombre de classes du GC associé à un système étant fini, toute branche d'un GAT contenant un nombre infini d'états contient nécessairement de telles boucles. Ainsi si le nombre d'états du GAT est infini, il existe nécessairement une boucle dans les paires de classes visitées par la paire de séquences de transitions de l'un des états du GAT. Ainsi, nous avons un critère d'arrêt pour vérifier la Ω-diagnosticabilité d'un système en calculant son GAT.

Soit $s = ((\sigma, \sigma'), \Pi_s) \in S$. Par définition les deux séquences de transitions partagent la même trace observable. Il est donc possible de découper σ et σ'. Il existe donc un entier strictement positif n permettant le découpage suivant :

— $\sigma = \sigma_{o_1} \dots \sigma_{o_n}$
— $\sigma' = \sigma'_{o_1} \dots \sigma'_{o_n}$

où $\forall i \in [1, n]$, $\sigma_{o_i} = \sigma_{uo_i}.t_{o_i}$, $\sigma'_{o_i} = \sigma'_{uo_i}.t'_{o_i}$ avec $(\sigma_{uo_i}, \sigma'_{uo_i}) \in T_{uo}^{\Theta^{*2}}$ et $(t_{o_i}, t'_{o_i}) \in T_o^{\Theta^2}$.

De ce découpage nous pouvons considérer les séquences de classes atteintes. Soient donc $(C_1, \dots, C_n, C'_1, \dots, C'_n) \in \mathrm{C}^{2n}$ (C est l'ensemble des classes du GC de Θ) tels que :

— $\mathcal{C}_0 \xrightarrow{\sigma_{o_1}} C_1 \dots \xrightarrow{\sigma_{o_n}} C_n$
— $\mathcal{C}_0 \xrightarrow{\sigma'_{o_1}} C'_1 \dots \xrightarrow{\sigma'_{o_n}} C'_n$

Par définition des éléments de S, σ admet un préfixe qui est le support d'un chemin fautif. Nous savons donc que $\exists i_0 \in [1, n]$, il existe π_f un chemin fautif tel que $\sigma_{o_1} \dots \sigma_{o_{i_0}} = \sigma_f$. Supposons qu'il existe deux entiers i_1 et i_2 supérieurs à i_0 tels que $\sigma_{o_0} \dots \sigma_{o_{i_1}}$ (resp. $\sigma'_{o_0} \dots \sigma'_{o_{i_1}}$) mène au même état que $\sigma_{o_0} \dots \sigma_{o_{i_2}}$ (resp. $\sigma'_{o_0} \dots \sigma'_{o_{i_2}}$), c.-à-d. $C_{i_1} = C_{i_2}$ (resp. $C'_{i_1} = C'_{i_2}$). Depuis C_{i_2} et C'_{i_2}, $\sigma_{o_{i_1+1}} \dots \sigma_{o_{i_2}}$ et $\sigma'_{o_{i_1+1}} \dots \sigma'_{o_{i_2}}$ sont tirables. Ainsi nous avons deux séquences de transitions tirables à l'infini, partageant la même trace observable. De plus lors de la seconde exploration de cette boucle, les même classes sont explorées, et ainsi les contraintes de l'enveloppe temporelle sont les mêmes que lors du premier passage dans la boucle. Ainsi si $\Pi_{s_{o_{i_1}}}$ est non vide, quel que soit le nombre de fois où la boucle sera explorée l'intersection sera non vide.

Définition 24. Soit Θ un système et Ω un motif temporel. On appelle *Graphe d'Ambiguïtés Temporel Réduit* (GATR) le quintuplet $G_r(\Theta, \Omega) = (S_r, s_0, \twoheadrightarrow_r, \ell_r^a)$ qui correspond au GAT $G(\Theta, \Omega)$ auquel on rajoute la règle de construction suivante : pour tout $s_1 \neq s_0$, $s_1 \twoheadrightarrow_r s_2$ si $s_1 \twoheadrightarrow s_2$ et si, en considérant $(C_1, \dots, C_n)$ et $(C'_1, \dots, C'_n)$ les séquences de classes atteintes par le tir des transitions observables de σ_2 et σ'_2, il n'existe pas d'entier $i \in [1, n]$ tel que $(C_n, C'_n) = (C_i, C'_i)$. Si il existe un tel i (en notant s_i l'état correspondant), on construit un arc entre s_1 et s_i (et l'exploration s'arrête sur cette branche), et on l'étiquète avec le dernier événement observable de σ_2.

Remarque. Dans le cas où le GAT admet un nombre d'états fini, le GATR associé au problème est égal au GAT.

Théorème 3. $\mathcal{D}(\Theta, \Omega) \Leftrightarrow$ *le GATR associé au problème ne contient pas de boucles.*

Ainsi, nous avons donc un graphe fini pour lequel une propriété permet de vérifier si un système Θ est diagnosticable pour un motif temporel Ω.

Remarque. La taille du GATR est bornée par la taille du produit synchrone du GC avec lui-même synchronisé sur les observations, sa taille est donc en $O(n^2)$ où n est le nombre de classes du GC du système.

Exemple 16. Revenons sur le système Θ_1 représenté Figure 3.1. Ce système est diagnosticable pour le motif Ω_2 représenté Figure 3.2b, car l'unique chemin pour lequel le motif peut être satisfait est l'unique chemin à avoir l'observation o'.

Intéressons nous maintenant au motif Ω_1 présenté Figure 3.2a. Trois séquences de transitions présentent la même projection observable : $\sigma_f = t_1.t_3.t_9.t_{10}$ (qui est le support d'un chemin fautif), $\sigma_1 = t_0.t_5.t_6.t_7$ et $\sigma_2 = t_1.t_2.t_{14}$ (séquences de transitions pour lesquelles il n'existe aucune exécution satisfaisant Ω_1). Les enveloppes temporelles observables de ces séquences sont

les suivantes : $\Pi_1 = \Pi_1^s = \{6 \leq y_7 \leq 13\}$, $\Pi_2 = \Pi_2^s = \{1 \leq y_{14} \leq 5\}$, $\Pi_f^s = \{7 < y_{10} \leq 10\}$, $\Pi_f^c = \{y_{10} = 5\}$ et $\Pi_f^a = \{5 < y_{10} \leq 7\}$ (le calcul de l'enveloppe temporelle de σ_f a été présenté dans l'Exemple 15).

Intéressons nous aux sources de paires critiques de cet exemple. Tout d'abord, l'intersection entre Π_1^s et Π_f^c est vide, la paire (σ_f, σ_1) ne génèrera donc pas de branche dans le GATR $G_r(\Theta_1, \Omega_1)$. Par contre il existe une intersection non vide entre Π_f^c et Π_2^s. De plus, le tir du couple de transitions (t_7, t_{14}) boucle. Ainsi $G_r(\Theta_1, \Omega_1)$ contient un état différent de l'état initial, admettant une boucle sur lui-même : $s_1 = ((\sigma_f, \sigma_2), \{y_o = 5\})$ (ici la variable y_o représente la date d'occurrence de l'observation o). De plus $\mathcal{S}(\Pi_a^f)$ est non vide, et le tir de t_{10} après σ_f réalise une boucle, $G_r(\Theta_1, \Omega_1)$ contient donc un autre état admettant une boucle sur lui-même : $s_2 = ((\sigma_f, \sigma_f), \{5 < y_o \leq 7\})$. Étant donné que $G_r(\Theta_1, \Omega_1)$ contient au moins une boucle, nous pouvons en conclure que Θ_1 n'est pas Ω_1-diagnosticable.

Implémentation de la condition :

Une implémentation d'un algorithme de vérification de diagnosticabilité de motifs temporels a été réalisée pour des motifs plus contraints que ceux présentés dans ce manuscrit. L'implémentation proposée est en effet fonctionnelle pour les fautes simples temporelles (un événement fautif contraint d'arriver entre a et b unités de temps). Cette implémentation cherche à gagner en efficacité par rapport à une implémentation qui consisterait à calculer la totalité du GATR pour un couple système/motif temporel donné. En effet la Proposition 6 permet dans certains cas de conclure sans avoir à calculer $G_r(\Theta, \Omega)$, et lorsqu'une branche avec une boucle est trouvée, il n'est pas nécessaire de calculer les autres états du graphe. Cette implémentation est intégrée dans le logiciel DiaDES [6], (non disponible en ligne à ce jour) a été réalisée en C++ et implémente l'algorithme suivant :

Le calcul des chemins fautifs et des chemins non fautifs π_s est basée sur une évaluation paresseuse du graphe des classes qui permet de gagner du temps de calcul. L'algorithme `calculCheminsMotif` est semblable à un algorithme de reconnaissance de chroniques [DGG93], et se base sur l'exploration du graphe des classes du système. L'algorithme vérifie si les éléments de $\mathcal{C}_\Omega$ peuvent être synchronisés avec la branche explorée durant cette exploration. La difficulté de cet algorithme vient du fait que la découverte d'une extraction pour un chemin n'est pas suffisante comme condition d'arrêt d'exploration de ce chemin, il est nécessaire de vérifier s'il existe des exécutions pour lesquelles le motif n'est pas reconnu avec cette extraction. Dans ce cas il est nécessaire de continuer la recherche pour trouver l'ensemble des extractions réalisant l'occurrence du motif pour une séquence de transitions. La complexité de cette méthode est bornée par la complexité du calcul du graphe des classes.

4.4 Conclusion

Ce chapitre présente la contribution majeure de cette thèse : une condition nécessaire et suffisante de diagnosticabilité d'un motif temporel pour un système. Cette analyse se base sur la notion de *chemin fautif*, qui capture l'occurrence d'une faute sous forme d'un polyèdre en utilisant l'abstraction sous forme de chemins présentée dans le Chapitre 2. L'enveloppe temporelle des chemins fautifs peut être partitionnée en zones pour lesquelles nous avons la garantie que toutes les

6. https://homepages.laas.fr/ypencole/diades/html/

Algorithme 2 Algorithme de vérification de diagnosticabilité d'un système Θ pour une faute simple temporelle Ω

Entrées : Un système Θ et une faute simple temporelle Ω
Sortie : `Vrai` si Θ est Ω-diagnosticable, `Faux` dans le cas contraire

1: $\mathcal{C}_\Omega \leftarrow$ `calculCheminsMotif`(Ω)
2: $\mathcal{C}_f(\Omega) \leftarrow$ `calculCheminsFautifs`$(\Theta, \mathcal{C}_\Omega)$ (Section 4.1)
3: **pour** $\pi_f \in \mathcal{C}_f(\Omega)$ **faire**
4: `découpageEnveloppeTemporelle` (Section 4.2)
5: **si** $\mathcal{S}(\Pi_f^a) \neq \emptyset$ (Proposition 6) **alors**
6: **retourner** `Faux`
7: **fin si**
8: **fin pour**
9: **tant que** $\mathcal{C}_f(\Omega) \neq \emptyset$ (Section 4.3) **faire**
10: $\pi_f \leftarrow$ `élémentDeCf`
11: **pour** $\pi_s \neq \pi_f, \ell(\sigma_s) == \ell(\sigma_f)$ **faire**
12: $B \leftarrow$ `constructionBrancheGATR`(π_f, π_s) (Définition 24)
13: **si** `boucle`$(B) ==$ `Vrai` **alors**
14: **retourner** `Faux`
15: **fin si**
16: **fin pour**
17: $\mathcal{C}_f(\Omega) \leftarrow \mathcal{C}_f(\Omega) \setminus \pi_f$
18: **fin tant que**
19: **retourner** `Vrai`

exécutions produisant une trace observable satisfaisant les contraintes d'une zone satisfont nécessairement le motif temporel, ne le satisfont jamais ou sont ambigües. En se basant sur cette partition nous proposons un graphe appelé *Graphe d'Ambiguïtés Temporel Réduit* semblable à un twin-plant permettant de vérifier si le système est Ω-diagnosticable en vérifiant la non existence de boucles dans ce graphe.

Chapitre 5

Diagnosticabilisation de TPN

Le chapitre précédent a permis de présenter une analyse de diagnosticabilité de motifs temporels, mais également de mettre en évidence les origines de cette non diagnosticabilité : pourquoi un système n'est pas diagnosticable pour un motif donné. Ces origines sont synthétisées dans un graphe appelé le GATR, un graphe similaire à un twin-plant (ou un verifier [Cab+09]) admettant un nombre fini de nœuds. La présence de boucles à l'intérieur de ce graphe témoigne de l'existence d'exécutions infinies ambigües entre elles, et chaque boucle est le préfixe d'un ensemble de paires critiques. La structure logique et temporelle du système explique l'existence de telles boucles. Énumérer les boucles d'un GATR revient à énumérer les raisons pour lesquelles un système n'est pas diagnosticable pour un motif temporel donné. Supprimer ces raisons de non diagnosticabilité devrait permettre de rendre un système diagnosticable. Nous appelons cette opération la *diagnosticabilisation*. La diagnosticabilisation d'un système est l'opération, pour un système et un motif donné, de le rendre diagnosticable pour ce motif.

Dans le cas des systèmes temporels, cette opération peut être réalisée avec deux méthodes différentes. La première consiste à modifier le modèle du système en intégrant ou en supprimant des transitions observables, ce qui revient en pratique à ajouter ou supprimer des capteurs sur le système réel sous-jacent [CLS13 ; BDS15]. Cette méthode peut se révéler coûteuse si l'on rajoute des capteurs, et modifie potentiellement de manière importante le comportement du système. La seconde méthode consiste à modifier les intervalles statiques du système pour interdire certaines zones temporelles (les zones d'ambiguïtés) et ainsi satisfaire la propriété (ce qui revient à reprogrammer le système). Cette seconde méthode, moins coûteuse, peut également modifier de manière importante le comportement du système si certaines sécurités ne sont pas mises en place (sous forme de contraintes par exemple). La modification des intervalles statiques d'un modèle PT a été étudiée par les auteurs de [BCC15]. Dans ce travail, l'objectif est de réparer le modèle pour qu'il soit cohérent avec les exécutions du système réel en modifiant certains intervalles statiques si des exécutions retournées par le système (sous la forme de logs) ne sont pas prises en compte par le modèle.

Dans ce chapitre nous proposons une analyse structurelle des exécutions fautives du système initial dont nous tirons des contraintes que certaines bornes des intervalles statiques du système initial (paramétré par l'ajout de paramètres à la place des dites bornes) doivent satisfaire pour que le système engendré soit diagnosticable pour un motif temporel donné. Nous considérons un système pour lequel un ensemble de paramètres est posé sur certaines bornes de ses intervalles statiques. Ce choix est motivé par le fait que sur un système réel, il est possible de modifier

certaines parties de son fonctionnement, mais tout n'est pas modifiable. Ainsi, en connaissant certains degrés de liberté d'un système (les paramètres sur lesquels nous pouvons agir), nous voulons étudier si le système peut être rendu diagnosticable pour le motif temporel associé. Des résultats préliminaires de ce travail ont donné lieu à une publication [CPS23]. Contrairement au travail de [BCC15], les modifications du modèle ne sont pas déduites à partir de logs retournés par le système réel, mais viennent d'une analyse similaire à l'analyse de diagnosticabilité présentée dans le Chapitre 4. En remplaçant certaines bornes des intervalles statiques du système par des paramètres, des contraintes supprimant les ambiguïtés sont synthétisées. Le résultat de cette analyse est un ensemble de contraintes paramétrées. S'il existe une solution, alors le système engendré par ces valeurs de paramètres est diagnosticable pour le motif étudié.

L'organisation de ce chapitre est la suivante. Dans un premier temps, en se basant sur une étude brève de la littérature sur les modèles temporels paramétrés, nous définirons les modèles paramétrés utilisés dans ce chapitre : les réseaux de Petri Temporels Paramétrés (PTP). Ensuite nous définissons l'objet de notre étude : la propriété de *diagnosticabilisation temporelle*. Enfin, nous développons une méthode de vérification de cette propriété en imposant certaines conditions sur les systèmes engendrés.

5.1 Formalisation du problème de diagnosticabilisation

Après avoir exploré rapidement les travaux existants sur la paramétrisation de modèles discrets temporels, nous utilisons les conclusions de cette exploration pour formaliser le problème étudié dans la suite de ce chapitre. Modifier les intervalles statiques d'un modèle revient à considérer qu'un sous-ensemble des bornes de ces intervalles sont des paramètres et qu'une solution au problème de diagnosticabilisation est un ensemble de valeurs pour ces paramètres tel que le système résultant de l'application de ces valeurs aux paramètres est diagnosticable pour le motif temporel étudié. Pour formaliser un tel problème il est donc nécessaire d'étendre les PTÉS utilisés jusqu'à présent en intégrant un ensemble de paramètres ainsi que leur domaine de définition.

5.1.1 Paramétrisation de réseaux de Petri temporels

La question des modèles temporels paramétrés a été étudiée dans la littérature depuis les années 90 [AHV93 ; Hun+01]. Ces travaux ont défini les Automates Temporisés Paramétrés (ATP par la suite), qui sont des automates temporisés auxquels ont été ajouté un ensemble de paramètres qui apparaissent dans certaines gardes. Un problème de vérification pour un ATP $\mathcal{A}$ et une propriété ψ consiste à calculer l'ensemble des valeurs des paramètres pour lesquelles $\mathcal{A}$ satisfait ψ. Les auteurs de [Hun+02] introduisent les ATP bornés Min/Max, qui sont des ATP pour lesquels un paramètre apparaît soit en borne Min d'une contrainte, soit en borne Max, mais ne peut apparaître en borne Min et en borne Max d'une contrainte, car de nombreux problèmes non triviaux sont indécidables pour les ATP. Pour ce type d'ATP certains problèmes deviennent décidables (voir [And19 ; BL09] pour plus de détails). L'outil IMITATOR [1] permet de faire de la vérification d'ATP [And21].

Les réseaux de Petri Temporels Paramétrés (PTP par la suite) ont été introduits dans [VP99], et sont une extension des réseaux de Petri Temporels à arcs inhibiteurs [2] (ITPNs dans le texte [TLR08 ; Lim+09]). À la structure des PT est ajouté un ensemble de paramètres $\Lambda = \{\lambda_1, \ldots, \lambda_n\}$, un

1. https://www.imitator.fr/
2. Les arcs inhibiteurs ne faisant pas partie du cadre de notre étude nous les omettons pour la suite de ce travail.

domaine d'existence pour les paramètres $D_\Lambda \in \mathbb{Q}_+^n$ qui est un polyèdre convexe, et une fonction J_s qui à chaque transition du PT associe un intervalle paramétré. Pour une sous-classe de ces réseaux paramétrés appelée réseaux de PTP à bornes Min/Max (dans un tel modèle un paramètre ne peut apparaître que comme borne inférieure ou comme borne supérieure) le problème *d'accessibilité* est décidable [TLR09]. La notion de classe est étendue à ce type de réseaux sous la forme de classe paramétrée où les paramètres sont intégrés dans les domaines de tir comme peuvent l'être des bornes classiques d'intervalles statiques. L'outil ROMEO [3] permet d'étudier les PTP, et est équipé d'un model-checker prenant en entrée des formules TCTL (extension de CTL avec du temps) [Lim+09].

Dans notre approche un paramètre n'est pas partagé par deux intervalles différents, les modèles étudiés sont donc une sous-catégorie des PTÉS paramétrés à bornes Min/Max. Nous proposons la définition suivante pour les PTP qui seront utilisés dans la suite de ce travail :

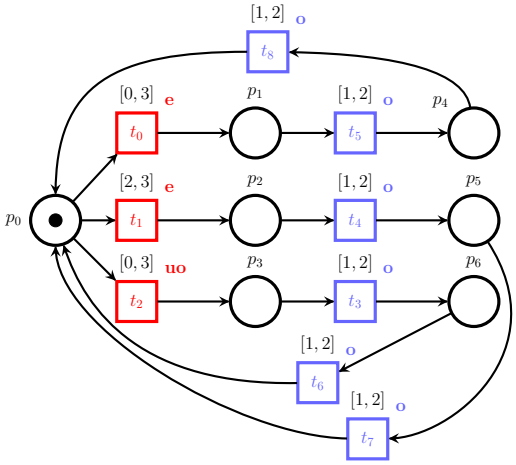
Définition 25. Un réseau de Petri Temporel Paramétré Étiqueté Sauf (P-PTÉS par la suite) est un quadruplet $N_\Lambda = \langle N, \Lambda, D_\Lambda, J \rangle$ où :
- N est un PTÉS
- $\Lambda = \{\lambda_i\}_{i \in I}$ est un ensemble fini de paramètres
- D_Λ est le domaine d'existence des paramètres de Λ. D_Λ est un polyèdre de $\mathbb{Q}_+^{card(\Lambda)}$ dont la dimension est égale au cardinal de Λ.
- D_Λ contient la valuation des éléments de Λ qui appliquée à N_Λ produit N (ainsi $D_\Lambda \neq \emptyset$).
- $J : T \times I_{\mathbb{Q}_+} \to Par_\Lambda(I_{\mathbb{Q}_+})$ est une fonction qui à un intervalle statique d'une transition associe un intervalle paramétré de $\mathbb{Q}_+$

Soit ν une valuation des éléments de Λ (ν est une fonction à valeurs dans D_Λ). On note N_Λ^ν le PTÉS obtenu en remplaçant les paramètres dans les intervalles paramétrés par leur valuation par ν (si le résultat est un PTÉS).

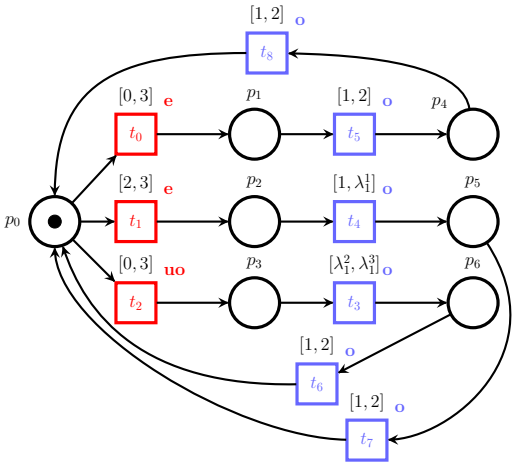
5.1.2 Diagnosticabilisation et Diagnosticabilisabilité temporelle

Comme nous l'avons expliqué précédemment, *diagnosticabiliser* un système est l'action de modifier la structure du modèle qui lui est associé dans le but de le rendre diagnosticable pour une faute ou un motif donné. Cette définition informelle n'est pas satisfaisante en cela qu'elle accepte des solutions très intrusives dans la structure du système. En effet, dans ce cadre il est possible qu'une solution empêche l'occurrence du motif dans le système (en empêchant le tir d'un sous-ensemble de transitions). En pratique cette solution n'a pas vraiment de sens de manière générale (elle pourrait en avoir selon le comportement étudié mais ce n'est pas vrai dans le cas général). Il est donc nécessaire de poser des limites pour écarter les solutions qui ne sont pas acceptables du point de vue pratique. De même, une solution qui ne satisferait pas les hypothèses **A1**, **A2** et **A3** (voir Chapitre 3 page 43) ne peut être considérée. Cela exclut notamment toute solution présentant des intervalles ponctuels (qui semblent être le type de solutions le plus simple pour diagnosticabiliser un système, au vu de leur pouvoir de différentiation des différentes trajectoires d'un système) et toute solution admettant des exécutions *Zénon*.

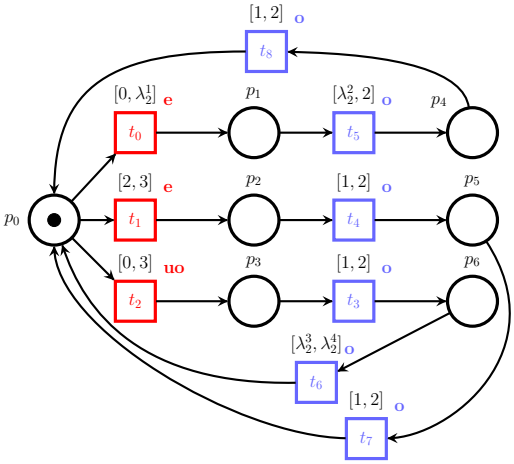
3. Cet outil est disponible ici : https://romeo.ls2n.fr/.



(a) Un système Θ



(b) Le système Θ paramétré par l'ensemble $\Lambda_1 = \{\lambda_1^1, \lambda_1^2, \lambda_1^3\}$ dont le domaine d'existence est $D_{\Lambda_1} = \mathbb{R}_+^3$



(c) Le système Θ paramétré par l'ensemble $\Lambda_2 = \{\lambda_2^1, \lambda_2^2, \lambda_2^3, \lambda_2^4\}$ dont le domaine d'existence est $D_{\Lambda_2} = \mathbb{R}_+^4$

FIGURE 5.1 – Le système Θ et deux versions différentes paramétrés avec les ensembles de paramètres Λ_1 et Λ_2

Soit Θ un système et Ω un motif temporel. Nous appelons système paramétré le P-PTÉS $\Theta_\Lambda = \langle \Theta, \Lambda, D_\Lambda, J_\Lambda \rangle$. Le problème de diagnosticabilisation consiste à chercher l'existence de valeurs des éléments de Λ pour lesquelles le système paramétré Θ_Λ est diagnosticable pour le motif Ω. Autrement dit, le problème de diagnosticabilisation temporelle est la recherche d'une valuation ν des éléments de Λ telle que Θ^ν_Λ est Ω-diagnosticable.

Définition 26. Une fonction de diagnosticabilisation temporelle est une fonction ν qui à Λ associe une valeur de D_Λ telle que le système Θ^ν_Λ est Ω-diagnosticable.

Pour un système Θ, un motif temporel Ω et un système paramétré Θ_Λ, il n'y a apriori aucune garantie qu'il existe une telle fonction de diagnosticabilisation temporelle ν. Avant de chercher une telle fonction, il est donc nécessaire de vérifier si une telle fonction existe. Vérifier l'existence d'une fonction de diagnosticabilisation temporelle est le problème de diagnosticabilisabilité temporelle (par la suite l'adjectif *temporelle* est omis pour éviter une certaine lourdeur dans la lecture mais est sous-entendu). Il est défini comme suit :

Définition 27. Un système Θ est (Ω, Λ)-diagnosticabilisable s'il existe une valuation à valeur dans D_Λ pour laquelle Θ est Ω-diagnosticable. Autrement dit, $\exists \nu, (\nu \models D_\Lambda) \Rightarrow \mathcal{D}(\Theta^\nu_\Lambda, \Omega)$. On note $\mathcal{D}^{ble}(\Theta, \Omega, \Lambda, D_\Lambda)$ la proposition *le système* Θ *est* (Ω, Λ)*-diagnosticabilisable pour l'ensemble de paramètres* Λ *défini sur* D_Λ.

Une conséquence directe de ces définitions est que tout système diagnosticable pour un motif est diagnosticabilisable pour ce motif quel que soit Λ.

Corollaire 3. $\mathcal{D}(\Theta, \Omega) \Rightarrow \mathcal{D}^{ble}(\Theta, \Omega, \Lambda, D_\Lambda)$.

Démonstration. Si Θ est Ω-diagnosticable, on pose ν telle que $\Theta^\nu_\Lambda = \Theta$ (existe car appartient à D_Λ par définition). □

Remarque. Nous cherchons dans ce travail à savoir si, pour un ensemble de paramètres donné, il existe une solution qui rend le système diagnosticable. Nous ne cherchons pas s'il existe un ensemble de paramètres qui permet de rendre le système diagnosticable.

Exemple 17. La Figure 5.1 présente un système et deux ensembles différents de paramètres pour ce système. Les motifs temporels considérés pour ce système sont $\Omega^1 = [0, 1]e$ et $\Omega^2 = [0, 3]e$. t_0 est l'unique transition réalisant l'occurrence de Ω^1 ou Ω^2 dans les exécutions de Θ. Θ n'est pas diagnosticable pour Ω^1 : le chemin fautif $(t_0.t_5, \{0 \leq y_0 \leq 3, 1 \leq y_5 - y_0 \leq 2\})$ admet un Π^a non vide, et ce chemin produit des ambiguïtés avec les séquences $t_1.t_4$ et $t_2.t_3$ qui partagent la même projection observable.

Les éléments présentés dans cet exemple vont servir de fil rouge tout au long de ce chapitre. L'ensemble de paramètres Λ_1 est choisi tel que Θ n'est ni (Ω^1, Λ_1)-diagnosticabilisable, ni (Ω^2, Λ_1)-diagnosticabilisable. Quant à Λ_2, il est tel que Θ n'est pas (Ω^1, Λ_2)-diagnosticabilisable, mais il est (Ω^2, Λ_2)-diagnosticabilisable. Une valuation ν des paramètres de Λ_2 telle que $\Theta^\nu_{\Lambda_2}$ est Ω^2-diagnosticable est présentée Figure 5.2.

Dans la suite de ce travail nous allons traiter un sous problème de la diagnosticabilisabilité, en cherchant des solutions préservant la structure du système initial, c.-à-d. que toute séquence de transition tirable dans Θ doit être toujours tirable dans Θ^{ν}_{Λ}. Pour cela, nous allons nous appuyer sur l'analyse de diagnosticabilité proposée dans le Chapitre 4, et tout particulièrement sur la partition de l'enveloppe temporelle des chemins fautifs. Nous avons pu voir dans le chapitre précédent qu'un système est diagnosticable pour un motif temporel donné s'il respecte un ensemble de conditions liées à ses chemins fautifs exprimables sous forme de contraintes ou d'ensemble de contraintes sur le temps écoulé entre le tir de deux transitions selon des séquences de transitions données. Nous allons par la suite exprimer cette propriété sous la forme d'un ensemble de contraintes paramétré dont les paramètres seront les éléments de Λ. Les contraintes sont synthétisées d'une part en exprimant la structure du système sous forme de contraintes sur les bornes des différents intervalles statiques des transitions (en se basant sur les domaines de tir des différentes classes du système de départ), et d'autre part en traduisant les conditions de diagnosticabilité vues dans le Chapitre 4. L'ensemble de contraintes synthétisé est tel que s'il existe une solution, alors Θ est diagnosticable pour le motif étudié et admet les mêmes séquences de transitions tirables (c.-à-d. le langage atemporel du système, le langage temporel sera lui différent).

5.2 Diagnosticabilisabilité temporelle

Dans cette section, nous considérons un système Θ, un motif Ω, un ensemble de paramètres Λ et un système paramétré Θ_{Λ}. Dans la recherche d'une valuation ν des éléments de Λ telle que Θ^{ν}_{Λ} est Ω-diagnosticable, il existe deux situations :

1. La première est celle où Θ est Ω-diagnosticable, et ainsi il existe déjà une telle valuation (voir Corollaire 3).
2. La seconde est celle où Θ n'est pas Ω-diagnosticable. Cette situation implique qu'il existe au moins une boucle dans $G_r(\Theta, \Omega)$ le GATR associé à Θ et Ω (voir page 78), et donc qu'il existe au moins un préfixe de paire critique identifié dans le Chapitre 4 pour Θ et Ω responsable de cette boucle. Une valuation ν rendant Θ^{ν}_{Λ} Ω-diagnosticable n'a aucune boucle dans $G_r(\Theta^{\nu}_{\Lambda}, \Omega)$. Une telle valuation satisfera donc nécessairement des contraintes qui suppriment les paires de chemins à l'origine de ces boucles.

Que ce soient les bornes des enveloppes temporelles ou les bornes des domaines de tir des classes du GC, ces bornes sont fonction des bornes des intervalles statiques. Modifier la valeur des bornes des intervalles statiques aura donc une répercussion sur les bornes des domaines de tir et des enveloppes temporelles.

Nous proposons ici de synthétiser des ensembles de contraintes fonction de ces différentes bornes. Ces ensembles sont prouvés équivalents à des propriétés de conservation d'éléments du système d'origine pour le système engendré par une valuation les satisfaisant. Ainsi si un système paramétré satisfait ces contraintes, il est diagnosticabilisable. Pour cela nous présentons dans un premier temps la synthèse de contraintes qui conservent la structure du système initial étudié. Ces contraintes permettront également d'assurer la satisfaction des hypothèses **A1**, **A2** et **A3** (page 43). Nous montrons ensuite la synthèse de contraintes assurant la préservation de l'ensemble des chemins fautifs (du point de vue des supports de ceux-ci). Pour de tels chemins, des contraintes assurant l'absence d'ambiguïtés dans leurs exécutions sont présentées. Pour assurer la diagnosticabilité du système produit, un dernier ensemble de contraintes assurant l'absence de paires critiques engendrées par deux chemins de supports différents est proposé.

5.2.1 Préservation de la structure d'un système

Nous sous-entendons par l'expression *préserver la structure* que nous voulons préserver les différents ordres de tir possibles des transitions du système initial, c.-à-d. que si t_i peut être tirée après t_j dans le système initial, il doit en être de même dans un système paramétré engendré. Cela se traduit par la conservation des séquences tirables dans le GC associé au système initial (nous cherchons à conserver le langage atemporel du GC du système initial). Autrement dit, pour une valuation considérée du système paramétré, le langage atemporel accepté par son GC est le même que le langage atemporel du GC du système de départ. En pratique, cela implique qu'un paramètre remplaçant une borne existante respecte la hiérarchie de sensibilisation des différentes transitions. Considérons une transition $t \in T$ dont la borne supérieure est remplacée par un paramètre λ dans Θ_Λ. Pour qu'une valuation préserve la structure, il faut que l'ensemble des transitions $t' \in T$ tirables en même temps que t le soient toujours. Autrement dit, il faut préserver les ensembles des transitions pouvant être sensibilisées en même temps.

Revenons sur le Théorème 1 présenté dans le Chapitre 2 (page 31). Ce théorème donne une condition pour le tir d'une transition depuis une classe donnée : une transition est tirable si et seulement si elle est sensibilisée et si le minimum des coefficients γ associés à cette transition est positif. Partant de ce résultat, l'idée est la suivante : si depuis la classe initiale le signe de l'ensemble des coefficients de type γ est préservé dans le système paramétré, alors nécessairement les classes accessibles depuis la classe initiale dans le GC de Θ (notons le G) et dans le GC de Θ^ν_Λ (notons le G^ν_Λ) présentent les mêmes marquages (et ainsi les mêmes transitions sensibilisées). Si depuis l'ensemble de ces classes nous conservons toujours le signe des coefficients de types γ, des classes admettant les mêmes marquages seront accessibles, et ainsi il semble possible en s'appuyant sur le signe des coefficients γ de préserver la même structure de G.

Comme cela a été vu précédemment, ces coefficients sont fonction des bornes des domaines de tirs (et donc des intervalles statiques). Notons α, β, et γ les coefficients de G et $\alpha^\nu(\Lambda)$, $\beta^\nu(\Lambda)$ et $\gamma^\nu(\Lambda)$ les coefficients dans G^ν_Λ. Considérons deux classes C et C' de G, où C' est accessible depuis C par le tir d'une transition t. En considérant deux transitions t_i et t_j sensibilisées dans C', γ'_{ij} s'écrit $\min(\beta'_i - \alpha'_j, \gamma_{ij})$. Ainsi, pour forcer la conservation du signe d'un coefficient γ_{ij} (avec t_i et t_j persistantes), une première remarque est que $\gamma_{ij} < 0 \Rightarrow \gamma'_{ij} < 0$. Les coefficients de type α décroissant plus rapidement que ceux de type β, nous avons également $\gamma_{ij} \geq 0 \Rightarrow \gamma'_{ij} \geq 0$ (soit $\gamma'_{ij} = \gamma_{ij}$, soit $\gamma'_{ij} = \beta'_i - \alpha'_j$ avec le coefficient α décroissant plus vite que les β). Pour tout γ_{ij} défini *la première fois* (c.-à-d. que l'une des transitions en jeu pour le calcul de ce coefficient est nouvellement sensibilisée dans la classe étudiée), ce coefficient est dépendant d'au moins une borne d'intervalle statique, conserver son signe s'exprime sous la forme $\beta'^\nu_i(\Lambda) - \alpha'^\nu_j(\Lambda) \#_\Lambda 0$ où l'opérateur $\#_\Lambda \in \{\leq, >\}$ est égal à l'opérateur $\#$ dans la contrainte donnant le signe de γ_{ij} : $\gamma_{ij} \# 0$. La conservation du signe s'exprime donc comme une fonction des coefficients définissant les domaines de tir des classes. De telles contraintes sont définissables pour l'ensemble des classes de G^ν_Λ. Ainsi un tel ensemble de contraintes défini pour toutes les classes de G permet de garantir que si une valuation ν le satisfait, alors G^ν_Λ est isomorphe à G.

Posons Π^{struct}_Λ l'ensemble de ces contraintes :

Théorème 4. *Si une valuation ν de Λ est telle que $\nu(\Lambda) \models \Pi^{struct}_\Lambda$, alors G est isomorphe à G^ν_Λ. Ces deux automates acceptent le même langage atemporel.*

Démonstration. Considérons une telle valuation ν. Nous allons montrer qu'il existe une bijection entre G et G^{ν}_{Λ}. Considérons la relation R définie sur les états de G et G^{ν}_{Λ} de la manière suivante : soient C et C' deux classes de G et G^{ν}_{Λ} respectivement. CRC' si :

— Les deux classes ont le même marquage.
— t_i est tirable depuis $C \Leftrightarrow t_i$ est tirable depuis C'.
— C est accessible depuis la classe initiale par la séquence de transitions $\sigma \Leftrightarrow C'$ est accessible depuis la classe initiale par la même séquence σ.

Montrons que R est une bijection.
Soient les deux classes initiales $\mathcal{C}_0$ et $\mathcal{C}'_0$ de G et G^{ν}_{Λ} respectivement :

— si pour toute transition sensibilisée dans $\mathcal{C}_0$, aucune borne d'intervalle statique n'est paramétrée, alors les domaines de tirs sont les mêmes, les marquages sont égaux, ainsi $\mathcal{C}_0 R \mathcal{C}'_0$
— si au moins une transition t_i tirable depuis $\mathcal{C}_0$ a son intervalle paramétré dans Θ_{Λ}, nous savons que $\forall t_j \in E(\mathcal{C}_0) \setminus t_i$, les signes de γ_{ij} et $\gamma^{\nu}_{ij}(\Lambda)$ sont les mêmes, et donc si t_i est tirable depuis $\mathcal{C}_0$ t_i est tirable depuis $\mathcal{C}'_0$ et inversement. Les marquages initiaux sont les mêmes, ainsi $\mathcal{C}_0 R \mathcal{C}'_0$.

Montrons que R est injective. Supposons maintenant qu'il existe C_1 et C_2 deux classes de G telles qu'il existe une classe C' de G^{ν}_{Λ} avec $C_1 R C'$ et $C_2 R C'$. C_1 et C_2 admettent le même marquage, les mêmes transitions tirables, et sont accessibles avec la même séquence. G étant déterministe, nous avons donc $C_1 = C_2$. R est une relation injective.

Montrons par l'absurde que R est surjective. Supposons maintenant qu'il existe une classe C' de G^{ν}_{Λ} telle qu'il n'existe pas de classe C de G avec CRC'. Quelle que soit C dans G, au moins une des conditions de R n'est donc pas satisfaite. Θ et Θ^{ν}_{Λ} ont les mêmes ensembles de places, de transitions et d'arcs. Soit σ' la séquence définie par $\mathcal{C}'_0 \xrightarrow{\sigma'} C'$. Supposons que σ' est aussi tirable dans G. La classe atteinte C par σ' admet le même marquage que C'. ν satisfait Π^{struct}_{Λ} donc C' et C partagent les mêmes transitions tirables. Cela implique donc que CRC', ce qui est absurde. Pour respecter notre hypothèse, cela veut donc dire que σ' n'est pas tirable dans G. Il existe donc un entier $i < |\sigma'|$ tel que $\sigma'_{|(i-1)}$ est tirable dans G et $\sigma'_{|i}$ n'est pas tirable. Posons $\sigma'_{|i} = \sigma'_{|(i-1)}.t_i$ et C_i et C'_i les classes atteintes en tirant $\sigma'_{|(i-1)}$ dans G et G^{ν}_{Λ} respectivement. Cela veut donc dire que t_i est tirable depuis C'_i et non tirable depuis C_i. G et G^{ν}_{Λ} étant déterministes et les ensembles de places, transitions et arcs étant égaux, C_i et C'_i partagent le même marquage. t_i étant tirable depuis C'_i elle est sensibilisée par son marquage, t_i est donc sensibilisée dans C_i. Pour que t_i ne soit pas tirable depuis C_i, il faut donc qu'il existe t_j sensibilisée dans C_i telle que pour C_i, $\gamma_{ij} < 0$. Or ν satisfait Π^{struct}_{Λ}, ainsi cela implique que $\gamma'^{\nu}_{ij}(\Lambda) < 0$, ce qui est impossible car t_i est tirable depuis C'_i (Théorème 1). Ainsi t_i est nécessairement tirable depuis C_i, et il n'existe donc pas de classe C' de G^{ν}_{Λ} telle qu'il n'existe aucune classe C de G en relation R avec C'. Ainsi R est surjective.

R est donc bijective. G et G^{ν}_{Λ} sont donc isomorphes. Les deux GC admettent donc le même langage atemporel. □

Exemple 18. Revenons sur la Figure 5.1c. Depuis l'état initial, trois transitions sont sensibilisées et tirables : t_0, t_1 et t_2. Conserver le signe des coefficients γ s'écrit pour ce système de la manière

suivante :

$$\begin{cases} \gamma_{10} :\uparrow (I(t_1)) - \downarrow (I(t_0)) \geq 0 \\ \gamma_{20} :\uparrow (I(t_2)) - \downarrow (I(t_0)) \geq 0 \\ \gamma_{12} :\uparrow (I(t_1)) - \downarrow (I(t_2)) \geq 0 \\ \gamma_{02} :\uparrow (I(t_0)) - \downarrow (I(t_2)) \geq 0 \\ \gamma_{21} :\uparrow (I(t_2)) - \downarrow (I(t_1)) \geq 0 \\ \gamma_{01} :\uparrow (I(t_0)) - \downarrow (I(t_1)) \geq 0 \end{cases} \tag{5.1}$$

En considérant l'ensemble de paramètres Λ_2 et le système paramétré Θ_{Λ_2}, cet ensemble se réécrit :

$$\begin{cases} \gamma_{10} : \top(3 - 0 \geq 0) \\ \gamma_{20} : \top(3 - 0 \geq 0) \\ \gamma_{12} : \top(3 - 0 \geq 0) \\ \quad \gamma_{02} : \lambda_2^1 - 0 \geq 0 \\ \gamma_{21} : \top(3 - 2 \geq 0) \\ \quad \gamma_{01} : \lambda_2^1 - 2 \geq 0 \end{cases} \tag{5.2}$$

Ainsi, nous avons :

$$\begin{cases} \quad \gamma_{02} : \lambda_2^1 \geq 0 \\ \gamma_{01} : \lambda_2^1 - 2 \geq 0 \end{cases} \tag{5.3}$$

Remarque. Il est important d'écrire la condition précédente en utilisant les coefficients de type γ et non seulement en se basant sur les coefficients α et β. Dans l'exemple présenté il semble qu'une condition sur les bornes α et β des domaines de tir soit suffisante. Malgré tout il existe certains cas (par exemple lorsque deux transitions sont persistantes) où le calcul de γ pour deux transitions n'est pas trivial (c.-à-d. différent de $\beta_i - \alpha_j$), il est nécessaire de poser les contraintes en utilisant les coefficients de type γ pour s'assurer de conserver les conditions de tir des transitions.

Par la suite nous ajoutons à Π_Λ^{struct} les contraintes suivantes :

- Pour toute transition t, nous imposons $\downarrow (I_s(t)) <\uparrow (I_s(t))$
- Pour tout cycle $\sigma^{cycl} = t_0 \dots t_n$ de G (c.-à-d. qu'il existe une classe C de G telle que $C \xrightarrow{\sigma^{cycl}} C$), nous imposons que pour toute temporisation r^{cycl} de σ^{cycl}, nous avons $time(r^{cycl}) > 0$. Cette expression implique que si l'on pose Π^{cycl} l'enveloppe temporelle de σ^{cycl}, nous pouvons écrire :

$$(\min_{r^{cycl} \in (\sigma^{cycl}, \Pi^{cycl})} time(r^{cycl})) > 0$$

Avec les conditions précédentes, nous imposons qu'une valuation satisfaisant Π_Λ^{struct} est tel qu'aucun intervalle n'est ponctuel, et nous imposons aussi qu'une valuation satisfaisant Π_Λ^{struct} n'admette pas d'exécution *Zénon*. Ainsi si une valuation satisfait Π_Λ^{struct}, Θ_Λ^ν satisfait les hypothèses **A2** et **A3** (page 43). Avec une telle condition aucune exécution de boucle n'est de durée nulle et donc il n'y a pas d'exécution infinie de durée nulle non plus.

Exemple 19. Revenons sur le système présenté Figure 5.1a. Ce système contient trois séquences de transitions qui engendrent des boucles dans son graphe de classes : $t_0.t_5.t_8$, $t_1.t_4.t_7$ et $t_2.t_3.t_6$. Pour empêcher les exécutions *Zénon*, il faut donc imposer $\downarrow(I(t_0))+\downarrow(I(t_5))+\downarrow(I(t_8)) > 0$, $\downarrow(I(t_1))+\downarrow(I(t_4))+\downarrow(I(t_7)) > 0$ et $\downarrow(I(t_2))+\downarrow(I(t_3))+\downarrow(I(t_6)) > 0$. En considérant le système paramétré Θ_{Λ_2}, nous obtenons les contraintes suivantes : $0+\lambda_2^2+1>0$, $\top$ (4>0) et $\top$ (2>0).

Comme expliqué précédemment, pour rendre un système diagnosticable, une idée est de chercher à supprimer l'ensemble de ses paires critiques. Une telle solution a été étudiée dans [HDY22]. Dans ce travail, les auteurs considèrent un système modélisé par un automate atemporel et une faute simple telle que le système considéré n'est pas diagnosticable pour cette faute. En se basant sur le twin-plant du problème, leur idée est d'intégrer du temps dans les branches ambigües sous forme de *blocs de retard* pour différencier les exécutions non pas seulement en observant des événements mais en rajoutant une horloge.

Le Chapitre 4 a permis d'exhiber pour les motifs temporels et les systèmes étudiés deux origines de paires critiques de nature différente : la première origine est intrinsèque à la structure temporelle d'un chemin fautif qui implique pour une même séquence de transitions l'existence d'exécutions fautives et non fautives produisant la même trace observable, et la seconde origine vient de l'existence de séquences de transitions différentes portant les mêmes observations (et qui dans le cas de chemins fautifs peuvent générer des ambiguïtés). La section précédente a permis de présenter une garantie pour une valuation d'un système paramétré de conserver la même structure de GC, et ainsi de conserver les séquences de transitions tirables. Autrement dit une telle valuation permet d'assurer que les mêmes séquences d'événements sont présentes dans le système initial et dans sa version paramétrée engendrée par une telle valuation (les dates d'occurrences des événements peuvent être différentes). Ainsi si l'on étudie le système initial et que l'on souhaite conserver les comportements dans lesquels le motif a lieu, il est possible de s'appuyer sur l'ensemble de chemins fautifs $\mathcal{C}_f(\Omega)$ du système initial (nous noterons $\mathcal{C}_f^{\nu}(\Omega)$ l'ensemble des chemins fautifs de la version paramétrée). L'étude de cet ensemble sera d'autant plus nécessaire pour supprimer les origines de paires critiques du premier type. Enfin, supprimer les sources de paires critiques du second type revient à s'assurer que les ensembles certains et saufs des enveloppes temporelles de deux séquences de transitions partageant la même projection observable ont une intersection vide. Les séquences étant préservées dans le système paramétré, supprimer ces origines de paires critiques peut s'exprimer sous forme de condition sur les enveloppes temporelles uniquement. Nous considérerons donc dans cette section que toutes les valuations évoquées satisfont Π_{Λ}^{Struct} (les valuations ne satisfaisant pas ces contraintes ne nous intéressant pas).

5.2.2 Conserver les chemins fautifs du point de vue des supports

Conserver les chemins fautifs se traduit sous la forme de deux conservations distinctes : tout chemin non fautif doit rester non fautif et tout chemin fautif doit rester fautif. Nous pouvons partitionner l'ensemble des chemins non fautifs en deux ensembles distincts :

1. L'ensemble des chemins non fautifs dont la séquence d'événements n'est cohérente avec aucune séquence d'événements d'aucun chemin du motif : cela se traduit par le fait qu'il

n'existe aucun chemin du motif dont la séquence d'événement associée à son support est une sous-suite de la séquence d'événements d'un chemin non fautif de cet ensemble.

2. L'ensemble des chemins non fautifs pour lesquels il existe une séquence d'événements cohérente avec une séquence d'événement d'au moins un chemin du motif, mais dont la synchronisation temporelle n'est pas possible : cela se traduit par l'existence d'au moins une contrainte du chemin du motif qui ne peut être satisfaite par ce chemin non fautif.

Quelle que soit une valuation ν telle que ($\nu \models \Pi_\Lambda^{struct}$) le premier ensemble de chemins non fautif est conservé, car une valuation n'affecte pas les séquences d'événements. Il n'est donc pas nécessaire de synthétiser de contraintes pour assurer sa conservation.

Quant au deuxième ensemble (notons le $\mathcal{C}_{\neg f}(\Omega)$), une valuation ν affecte la structure temporelle du système et peut donc faire apparaitre des occurrences du motif incongrues. Pour interdire un tel phénomène, il faut assurer que pour chaque chemin de $\mathcal{C}^\nu_{\neg f}(\Omega)$, il n'existe pas de préfixe du support qui associé avec son enveloppe temporelle forme un chemin fautif. Le motif s'exécutant en temps fini, nous avons vu dans le Chapitre 4 (page 50) que l'ensemble des chemins fautifs est fini. En appliquant un raisonnement similaire, nous pouvons conclure que l'ensemble de ces préfixes est fini, et que la conservation du caractère non fautif de ces chemins est un problème fini.

Définissons $\mathcal{C}_{\neg f}(\Omega)$ l'ensemble de chemins $\pi_{\neg f}$ tels quel :

1. $\sigma_{\neg f} = \sigma.\sigma'$
2. $\exists \pi_\Omega \in \mathcal{C}_\Omega,\ \ell^\Omega(\sigma_\Omega) \in \omega(\ell^\Theta(\sigma))$
3. $\forall (\sigma_1, \sigma_2) \in (T^* \setminus \emptyset)^2,\ \sigma_1.\sigma_2 = \sigma \Rightarrow \nexists \pi_\Omega \in \mathcal{C}_\Omega,\ \ell(\sigma_\Omega) \in \omega(\ell(\sigma_1))$
4. $\sigma' = \sigma''.t_o,\ \ell(\sigma'') \in \Sigma^*_{uo}$
5. $\pi_{\neg f} \notin \mathcal{C}_f(\Omega)$

Ces conditions sont similaires à celles posées dans le Chapitre 4 pour définir les chemins fautifs, à l'exception des conditions 4 et 5. Cette définition permet de se concentrer directement sur les préfixes des chemins non fautifs du deuxième ensemble.

Proposition 9. *Soit ν une valuation satisfaisant Π_Λ^{struct}. Pour tout chemin du système π, $\pi \in \mathcal{C}_{\neg f}(\Omega) \Rightarrow \pi^\nu(\Lambda) \in \mathcal{C}^\nu_{\neg f}(\Omega) \Leftrightarrow$ Pour toute paire de chemins $(\pi^\nu(\Lambda) = (\sigma, \Pi^\nu(\Lambda)), \pi_\Omega = (\sigma_\Omega, \Pi_\Omega)) \in \mathcal{C}^\nu_{\neg f}(\Omega) \times \mathcal{C}_\Omega$ telle que $\ell^\Omega(\sigma_\Omega) \in \omega(\ell^\Theta(\sigma))$, alors il existe une contrainte $c \in \Pi_\Omega[Sub]$ telle que $(\alpha_i^\nu(\Lambda) > \beta_c) \vee (\beta_i^\nu(\Lambda) < \alpha_c)$.*

Cette proposition exprime qu'un chemin π dans le système initial Θ appartenant à $\mathcal{C}_{\neg f}(\Omega)$ appartient à $\mathcal{C}^\nu_{\neg f}(\Omega)$ dans le système Θ^ν_Λ si et seulement si pour tout chemin du motif π_Ω dont les événements du support sont une sous-suite des événements du support de $\pi^\nu(\Lambda)$ (le chemin engendré par la valuation ν des paramètres Λ de π), il existe au moins une contrainte c de l'enveloppe temporelle de π_Ω qui ne peut être satisfaite par les variables de $\Pi^\nu(\Lambda)$.

Démonstration. $\Rightarrow$: Soit $\pi_{\neg f} \in \mathcal{C}_{\neg f}(\Omega)$. $\nu \models \Pi_\Lambda^{struct}$, il existe donc au moins une sous-suite σ_s de $\ell(\sigma_{\neg f})$ pour lequel il existe $\pi_\Omega \in \mathcal{C}_\Omega$ avec $\ell(\sigma_\Omega) = \sigma_s$. Pour que $\pi^\nu_{\neg f}(\Lambda) \notin \mathcal{C}^\nu_f(\Omega)$, nécessairement l'extraction $(t_i)_{i \in I}$ associée à ω dans $\sigma_{\neg f}$ est telle qu'au moins une contrainte c dans $\Pi_\Omega[Sub]$ n'est

pas satisfaite (où Sub est la substitution qui aux variables de l'extraction $(t_i)_{i\in I}$ associe les variables associées à ω dans Π_Ω). Ainsi, $\exists c \in \Pi_\Omega[Sub]$, $(\alpha_c > \beta_i^\nu(\Lambda)) \vee (\beta_c < \alpha_i^\nu(\Lambda))$.

$\underline{\Leftarrow}$: Si pour tout chemin $\pi_{\neg f}^\nu(\Lambda)$ il n'existe aucun chemin du motif dont toutes les contraintes sont satisfaites par quelque exécution de $\pi_{\neg f}^\nu(\Lambda)$ que ce soit, alors ce chemin n'est pas un chemin fautif, et ainsi $\pi_{\neg f}^\nu(\Lambda) \in \mathcal{C}_{\neg f}^\nu(\Omega)$. □

Ainsi, pour s'assurer qu'aucun chemin fautif supplémentaire n'a été créé, il faut satisfaire la Proposition 10. Cette condition s'écrit sous forme d'une conjonction de disjonctions de contraintes : en notant k_{π_Ω} le cardinal de $\Pi_\Omega[Sub]$ pour chaque chemin du motif π_Ω, nous pouvons écrire pour un chemin $\pi_{\neg f}^\nu(\Lambda)$ de $\mathcal{C}_{\neg f}^\nu(\Omega)$: $\bigwedge_{\pi_\Omega\in\mathcal{C}_\Omega}(\bigvee_{j=1}^{k_{\pi_\Omega}}((\alpha_{c_j} > \beta_j^\nu(\Lambda)) \vee (\beta_{c_j} < \alpha_j^\nu(\Lambda))))$. Nous notons par la suite $\Pi_\Lambda^{\neg f}$ l'ensemble de toutes ces contraintes. Posons $\mathcal{N}_f(\Lambda,\nu)$ le prédicat suivant :

$$\mathcal{N}_f(\Lambda,\nu) : \forall \pi \in \Gamma, \pi \in \mathcal{C}_{\neg f}(\Omega) \Rightarrow \pi^\nu(\Lambda) \in \mathcal{C}_{\neg f}^\nu(\Omega)$$

Ce prédicat est vrai si tout chemin non fautif dans le système initial est toujours non fautif pour le système paramétré produit par une valuation ν.

Proposition 10. $(\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f}) \Leftrightarrow \mathcal{N}_f(\Lambda,\nu)$

Démonstration. Application directe de la Proposition 9. □

Intéressons nous maintenant à la conservation des chemins fautifs. Informellement un chemin est fautif s'il existe une synchronisation de ce chemin avec un chemin du motif. Il est cependant important de ne pas oublier qu'il peut exister plus d'un chemin du motif synchronisable avec un même chemin fautif. Ces différents chemins du motif sont donc associés à des exécutions du chemin fautif différentes qui correspondent à une *première occurrence du motif* pour ces exécutions. Nous cherchons ici à préserver l'ensemble des synchronisations d'un chemin fautif. Pour qu'un chemin fautif π_f dans Θ associé à une extraction σ_s soit fautif dans Θ_λ^ν pour cette même extraction σ_s, il faut qu'avec la valuation ν le chemin du motif π_Ω associé à π_f et σ_s soient synchronisables, donc que toute contrainte c de l'enveloppe temporelle de π_Ω soit satisfiable par des réductions de l'enveloppe temporelle de π_f. Posons le prédicat suivant :

$$\mathcal{F}(\Lambda,\nu) : \forall \pi \in \Gamma, \pi \in \mathcal{C}_f(\Omega) \Rightarrow \pi^\nu(\Lambda) \in \mathcal{C}_f^\nu(\Omega)$$

Proposition 11. *Soit une valuation ν telle que $\nu \models \Pi_\Lambda^{struct}$. $\mathcal{F}(\Lambda,\nu) \Leftrightarrow$ Pour tout chemin fautif $\pi_f^\nu(\Lambda) = (\sigma_f, \Pi^\nu(\Lambda)) \in \mathcal{C}_f^\nu(\Omega)$, pour tout chemin du motif $\pi_\Omega = (\sigma_\Omega, \Pi_\Omega) \in \mathcal{C}_\Omega$ tel que $(\ell(\sigma_\Omega) \in \omega(\ell(\sigma_f))) \wedge (\forall c \in \Pi_\Omega[Sub], (\alpha_c \leq \beta_i) \vee (\beta_c \geq \alpha_i))$ alors pour toute contrainte $c \in \Pi_\Omega[Sub]$, $(\alpha_c \leq \beta_i^\nu(\Lambda)) \vee (\beta_c \geq \alpha_i^\nu(\Lambda))$.*

Cette proposition exprime qu'un chemin fautif π_f dans Θ est un chemin fautif de Θ_Λ^ν si et seulement si pour tout chemin du motif π_Ω synchronisable avec π_f dans Θ, π_Ω est synchronisable avec $\pi_f^\nu(\Lambda)$. Autrement dit un chemin fautif π_f de Θ est fautif dans Θ_Λ^ν si et seulement si $\pi_f^\nu(\Lambda)$ est synchronisable avec les mêmes chemins π_Ω que π_f.

Démonstration. $\underline{\Rightarrow :}$ Soit ν une valuation satisfaisant Π_Λ^{struct}. $\mathcal{F}(\Lambda,\nu)$ implique que pour tout chemin fautif $\pi_f \in \mathcal{C}_f(\Omega)$, le chemin $\pi_f^\nu(\Lambda)$ est aussi fautif. Pour tout chemin $\pi_\Omega \in \mathcal{C}_\Omega$ tel qu'il existe une extraction $(t_i)_{i\in I}$ telle que $\forall c \in \Pi_\Omega[Sub], (\alpha_c \leq \beta_i) \vee (\beta_c \geq \alpha_i)$, π_f associé à cette extraction est encore un chemin fautif après paramétrisation, ainsi il existe au moins une exécution de π_f telle que toutes les contraintes de $\Pi_\Omega[Sub]$ sont satisfaites. Ainsi nous avons $\forall c \in \Pi_\Omega[Sub], (\alpha_c \leq \beta_i^\nu(\Lambda)) \vee (\beta_c \geq \alpha_i^\nu(\Lambda))$.

$\underline{\Leftarrow :}$ Si pour tout π_Ω associé avec un chemin fautif $\pi_f \in \mathcal{C}_f(\Omega)$ l'ensemble des contraintes de son enveloppe temporelle sont satisfaites par au moins une exécution dans le système paramétré avec la valuation ν, alors $\pi_f^\nu(\Lambda) \in \mathcal{C}_f^\nu(\Omega)$. □

Ainsi pour s'assurer que l'on conserve l'ensemble des chemins fautifs (du point de vue des supports) il faut satisfaire la Proposition 11. Cette condition s'écrit comme un conjonction de contraintes pour tout couple *chemin fautif/extraction* pour qu'il existe une exécution fautive associée à ce couple. Pour un tel couple et le chemin du motif associé π_Ω (en supposant que l'enveloppe temporelle de π_Ω contient k_{π_Ω} contraintes), cela s'écrit : $\bigwedge_{j=1}^{k_{\pi_\Omega}}((\alpha_{c_j} \leq \beta_j^\nu(\Lambda)) \vee (\beta_{c_j} \geq \alpha_j^\nu(\Lambda)))$. On note par la suite Π_Λ^f l'ensemble de toutes ces contraintes.

Théorème 5. $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^f \wedge \Pi_\Lambda^{\neg f} \Leftrightarrow (\mathcal{F}(\Lambda,\nu) \wedge \mathcal{N}_f(\Lambda,\nu))$.

Démonstration. D'après le Théorème 4, $\nu \models \Pi_\Lambda^{struct}$, alors les graphes G et G_Λ^ν sont isomorphes. D'après la Proposition 10, $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f}$ si et seulement si tout chemin non fautif dans le système Θ est un chemin non fautif pour Θ_Λ^ν. D'après la Proposition 11, $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^f$ si et seulement si tout chemin non fautif de Θ est un chemin non fautif de Θ_Λ^ν. Ainsi $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^f$ si et seulement si $\mathcal{N}_f(\Lambda,\nu)$ et $\mathcal{F}(\Lambda,\nu)$. □

Exemple 20. Considérons le motif Ω^1 et le système paramétré Θ_{Λ_2} (Exemple 17 page 85). Pour ce motif il existe un chemin fautif dans Θ, qui dans Θ_{Λ_2} s'exprime ainsi : $\pi_f(\Lambda_2) = \{(t_0.t_5, \{y_0 \leq \lambda_2^1, \lambda_2^2 \leq y_5 - y_0 \leq 2\})\}$, et un chemin dans $\mathcal{C}_{\neg f}(\Omega)$ qui dans Θ_{Λ_2} s'exprime ainsi : $\pi = \{(t_1.t_4, \{2 \leq y_1 \leq 3, 1 \leq y_4 - y_1 \leq 2\})\}$. Pour conserver $\mathcal{C}_{\neg f}(\Omega)$, il faut donc imposer que $\pi(\Lambda_2)$ soit un chemin de $\mathcal{C}_{\neg f}^\nu(\Omega)$. L'enveloppe temporelle de π ne contient pas de paramètres, donc $\pi(\Lambda_2) \in \mathcal{C}_{\neg f}^\nu(\Omega)$. Pour préserver $\mathcal{C}_f(\Omega)$ il est nécessaire d'imposer $\lambda_2^1 \geq 0$, ce qui est déjà imposé par son domaine d'existence (qui est $\mathbb{R}_+$).

Il en est de même pour le couple Ω^2 et le système paramétré Θ_{Λ_2}. Pour ce couple, le chemin π (présenté précédemment dans cet exemple) est un chemin fautif, et les mêmes contraintes s'appliquent.

5.2.3 Interdire les ambiguïtés intrinsèques pour un chemin fautif

Pour résumer les sections précédentes, nous avons une condition sur des contraintes permettant d'assurer que G_Λ^ν est isomorphe à G (Théorème 4), et d'assurer que l'ensemble des chemins fautifs et non fautifs sont conservés (Théorème 5). Nous sommes donc capables de préserver le comportement général du système initial. La partie *conservation* est maintenant terminée, nous allons nous intéresser à la suppression des origines de paires critiques.

Nous avons vu que, pour un chemin fautif, l'existence de solutions à son polyèdre ambigu est source de non diagnosticabilité (Proposition 6). Nous allons donc établir une condition permettant d'assurer qu'un chemin fautif admet un polyèdre ambigu vide.

Considérons donc $\pi_f \in \mathcal{C}_f(\Omega)$ un chemin fautif de Θ pour lequel il existe une solution à la partie ambigüe de son enveloppe temporelle. Nous rappelons l'expression de $\Pi^a_f = \Pi^o_f \wedge (\bigvee_{c\in\Pi_\Omega[Sub]}(c^a \wedge \bigwedge_{c'\neq c\in\Pi_\Omega[Sub]} \neg c'^s))$ (page 71). Pour que Π^a_f soit un polyèdre vide il faut que quelle que soit la contrainte $c \in \Pi_{\Omega[Sub]}$, c^a soit une formule non satisfaite. Rappelons pour une telle contrainte c et les transitions de σ_f impliquées t_i, t_{o_i} et $t_{o_{i-j-1}}$, c^a s'exprime ainsi (page 67) :

$$c^a = ((\alpha_{o_i} + \alpha_c + \alpha_{o_{i-j-1}} \leq y_{o_i} - y_{o_{i-j-1}} < \beta_{o_i} + \alpha_c + \beta_{o_{i-j-1}}) \wedge (\alpha_c > \alpha_i))$$
$$\oplus((\alpha_{o_i} + \beta_c + \alpha_{o_{i-j-1}} < y_{o_i} - y_{o_{i-j-1}} \leq \beta_{o_i} + \beta_c + \beta_{o_{i-j-1}}) \wedge (\beta_c < \beta_i))$$

Comme nous l'avons dit précédemment nous ne considérons pas les solutions supprimant l'occurrence du motif dans les chemins fautifs du système, pour rendre c^a non satisfaite. Une solution pour s'assurer que c^a est non satisfaite est d'imposer $(\alpha^\nu_i(\Lambda) \geq \alpha_c)$ et $(\beta^\nu_i(\Lambda) \leq \beta_c)$. $\alpha^\nu_i(\Lambda)$ et $\beta^\nu_i(\Lambda)$ sont des fonctions des bornes des intervalles statiques des transitions du système, nous pouvons donc en tirer la proposition suivante :

Proposition 12. *Soit $\pi_f \in \mathcal{C}_f(\Omega)$ un chemin fautif de Θ, $(t_i)_{i\in I}$ une extraction associée et $\pi_\Omega \in \mathcal{C}_\Omega$ le chemin du motif associé à ce couple* chemin fautif/extraction. *Soit une valuation ν satisfaisant Π^{struct}_Λ, Π^f_Λ, $\Pi^{\neg f}_\Lambda$. Si pour Θ^ν_Λ nous avons $\forall c \in \Pi_{\Omega[Sub]}, (\alpha_c \leq \alpha^\nu_i(\Lambda)) \wedge (\beta^\nu_i(\Lambda) \leq \beta_c)$ alors $\mathcal{S}(\Pi^a_f(\Lambda)) = \emptyset$.*

Démonstration. Soient de tels π_f, $(t_i)_{i\in I}$ et π_Ω. Si quel que soit c dans $\Pi_\Omega[Sub]$ nous avons $(\alpha_c \leq \alpha^\nu_i(\Lambda))$ et $(\beta_c \geq \beta^\nu_i(\Lambda))$, alors c^a est toujours faux. Ainsi l'ensemble des solutions de $\mathcal{S}(\Pi^a_f(\Lambda))$ est vide.

□

Exemple 21. Revenons sur le couple $(\Omega^1 = [0,1]e, \Theta_{\Lambda_2})$ (page 85). Pour ce motif et ce système paramétré, nous avions trouvé un chemin fautif s'exprimant pour le système paramétré : $\pi_f(\Lambda_2) = (t_0.t_5, \{y_0 \leq \lambda^1_2, \lambda^2_2 \leq 2\})$. Pour imposer $\mathcal{S}(\Pi^a_f(\Lambda_2)) = \emptyset$, nous imposons (ici $\alpha_c = 0$ et $\beta_c = 1$) $\alpha_0 \geq \alpha_c$ et $\beta_0 \leq \beta_c$, ainsi nous obtenons la contrainte suivante : $0 \leq \lambda^1_2 \leq 1$. Nous pouvons ici remarquer que cette contrainte est contradictoire avec une contrainte à imposer pour conserver la structure : $\lambda^1_2 - 2 \geq 0$. Pour cette raison, le système Θ ne peut pas être rendu diagnosticable pour le motif Ω^1 en utilisant l'ensemble de paramètres Λ_2.

Pour le couple $(\Omega^2 = [0,3]e, \Theta_{\Lambda_2})$, il existe un second chemins fautifs. En effet, en considérant le motif Ω^2, le chemin π présenté dans l'Exemple 20 est un chemin fautif (il appartient à $\mathcal{C}_f(\Omega^2)$). Nous rappelons son expression dans le système paramétré Θ_{Λ_2} : $\pi(\Lambda_2) = (t_1.t_4, \{2 \leq y_1 \leq 3, 1 \leq y_4 - y_1 \leq 2\})$. L'enveloppe temporelle de $\pi(\Lambda_2)$ n'étant pas influencée par la valeur des paramètres de Λ_2, il n'est pas possible pour ce chemin fautif de synthétiser des contraintes assurant que $\mathcal{S}(\Pi^a) = \emptyset$. Or π ne présente pas de zone ambigu dans son enveloppe temporelle (le tir de t_1 satisfait nécessairement Ω^2), nous avons donc $\mathcal{S}(\Pi^a(\Lambda_2) = \emptyset$. Concernant $\pi_f(\Lambda_2)$, son enveloppe temporelle est influencée par la valeur des paramètres de Λ_2. Pour s'assurer que $\mathcal{S}(\Pi^a_f(\Lambda_2)) = \emptyset$, il faut donc imposer : $\lambda^1_2 \leq 3$.

Revenons sur les Figures 4.2 et 4.3 du Chapitre 4. Si pour une contrainte c nous avons $(\alpha_c \leq \alpha_i) \wedge (\beta_c \geq \beta_i)$, les zones non fautives de la Figure 4.2 (zones vertes) ne sont pas présentes. Ainsi toutes les exécutions de ce chemin satisfont nécessairement c, et ainsi sur la Figure 4.3 il n'y a ni zone ambigu ni zone sauf. Cela généralisé à l'ensemble des contraintes de $\Pi_\Omega[Sub]$, cela implique que quelle que soit l'exécution d'un tel chemin, elle satisfait nécessairement $\Pi_\Omega[Sub]$, et ainsi toute exécution d'un tel chemin satisfait le motif temporel.

La Proposition 12 peut se généraliser en posant :

$$\Pi_\Lambda^{\neg a} = \bigwedge_{\pi_f(\Lambda) \in \mathcal{C}_f^\nu(\Omega)} \Big(\bigwedge_{\pi_\Omega \in \mathcal{C}_\Omega, Sy(\pi_f(\Lambda), \pi_\Omega)} \Big(\bigwedge_{c \in \Pi_\Omega[Sub]} ((\alpha_c \leq \alpha_i^\nu(\Lambda)) \wedge (\beta_i^\nu(\Lambda) \leq \beta_c)) \Big) \Big)$$

où Sy est l'application booléenne qui a un couple chemin fautif/chemin du motif associe $\top$ s'il existe une extraction σ_s permettant de synchroniser π_f et π_Ω (et pour laquelle il existe une exécution de π_f où σ_s correspond à la première occurrence du motif).

Corollaire 4. $\nu \models \Pi_\Lambda^{\neg a} \Rightarrow \nu \models \Pi_\Lambda^f$

En effet, satisfaire Π_Λ^f revient à dire qu'il existe une intersection non nulle entre la contrainte imposée dans $\Pi_\Omega[Sub]$ et celle imposée par l'enveloppe temporelle d'un chemin. Satisfaire $\Pi_\Lambda^{\neg a}$ revient à dire que la contrainte imposée par l'enveloppe temporelle est inclue dans la contrainte imposée par le chemin du motif temporel, ce qui implique nécessairement de satisfaire Π_Λ^f. Nous avons donc $\Pi_\Lambda^f \wedge \Pi_\Lambda^{\neg a} = \Pi_\Lambda^{\neg a}$. Posons la proposition suivante :

$$\mathcal{P}_a(\Lambda, \nu) : \forall \pi_f^\nu(\Lambda) \in \mathcal{C}_f^\nu(\Omega), \mathcal{S}(\Pi_f^{a,\nu}(\Lambda)) = \emptyset$$

Théorème 6. $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^{\neg a} \Rightarrow \mathcal{N}_f(\Lambda, \nu) \wedge \mathcal{F}(\Lambda, \nu) \wedge \mathcal{P}_a(\Lambda, \nu)$

Démonstration. D'après le Théorème 4, si $\nu \models \Pi_\Lambda^{struct}$, alors G et G_Λ^ν sont isomorphes. D'après le Théorème 5, $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^f$ est équivalent à conserver les chemins fautifs et non fautifs. D'après le Corollaire 4, si $\nu \models \Pi_\Lambda^{\neg a}$ alors $\nu \models \Pi_\Lambda^f$. En appliquant la Proposition 12 à l'ensemble des éléments de $\mathcal{C}_f^\nu(\Omega)$, nous avons $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^{\neg a}$. □

Proposition 13. $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^{\neg a} \Rightarrow \forall \pi_f^\nu(\Lambda) \in \mathcal{C}_f^\nu(\Omega),\ \Pi_f^o = \Pi_f^c$

Démonstration. Soit ν une telle valuation. Le système paramétré produit Θ_Λ^ν admet les mêmes structures et langages atemporels (du point de vue du GC) que Θ, il y a donc préservation des ensembles des chemins fautifs et des chemins non fautifs (Théorème 6). Soit $\pi_f \in \mathcal{C}_f(\Omega)^\nu$. $\nu \models \Pi_\Lambda^{\neg a}$, donc en considérant π_Ω le chemin du motif associé à π_f et Sub la substitution, nous avons $\forall c \in \Pi_\Omega[Sub]$, $(\alpha_c \leq \alpha_i^\nu(\Lambda)) \wedge (\beta_c \geq \beta_i^\nu(\Lambda))$, et ainsi c^c est vrai pour toute exécution. Ainsi pour tout $c \in \Pi_\Omega[Sub]$ aucune exécution ne satisfait ni c^a ni c^s, les ensembles Π_f^s et Π_f^a admettent donc des ensembles de solution vides, et ainsi comme Π_f^a, Π_f^s et Π_f^c forment une partition de Π_f^o la projection observable de l'enveloppe temporelle de π_f (Lemme 1), nous avons $\Pi_f^o = \Pi_f^c$. □

À ce stade de notre analyse, nous pouvons vérifier si une valuation produit un système paramétré dont le graphe des classes accepte le même langage atemporel que celui du système initial, préserve l'ensemble des chemins fautifs et non fautifs (du point de vue des supports), et est tel que pour tout chemin fautif (et toute extraction associée à un tel chemin), l'ensemble de contraintes ambiguës n'admet pas de solutions. Autrement dit le système produit ne contient pas de source d'ambiguïtés intrinsèques à la structure des chemins fautifs.

5.2.4 Empêcher l'existence de paires critiques engendrées par deux séquences de transitions

Il manque une dernière étape pour assurer qu'une valuation produise un système diagnosticable : assurer qu'il n'existe aucune ambiguïté produite par deux chemins différents partageant une trace observable commune. Cela implique qu'il n'existe pas de boucles dans le GATR associé au problème avec le système paramétré. Considérons une valuation ν satisfaisant $\Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^{\neg a}$ (nous avons donc $\mathcal{P}_a(\Lambda, \nu)$). Le GC du système paramétré Θ_Λ^ν admet la même structure et le même langage atemporel que celui de Θ. Cependant les domaines de tirs des différentes classes sont a priori différents. Il n'y a donc aucune garantie que la structure du GATR généré par le couple $(\Theta_\Lambda^\nu, \Omega)$ soit la même structure que celle du GATR généré par (Θ, Ω). En particulier il est possible pour un couple de chemins (σ_f, σ') de produire une boucle dans $G_r(\Theta, \Omega)$ le GATR de Θ pour le motif Ω, mais que ce ne soit pas le cas dans $G_r(\Theta_\Lambda^\nu, \Omega)$ le GATR de Θ_Λ^ν pour Ω.

Il est nécessaire pour éviter de telles boucles d'identifier l'ensemble des paires de séquences de transitions susceptibles de produire de telles boucles, et de synthétiser des contraintes sur les dates d'occurrence des transitions observables de chaque séquence de manière à rendre vide l'intersection des enveloppes certaines et sauves de ces deux séquences. Nous cherchons donc à abstraire l'ensemble des supports de paires critiques sous la forme de paire de séquences de transitions de taille finie partageant la même projection observable. Considérons deux séquences de transitions, l'une admettant un préfixe qui est le support d'un chemin fautif et l'autre non, partageant la même trace observable, et telles que leur projection observable se termine par une boucle d'observations, c.-à-d. une séquence qui peut être répétée. Alors si l'intersection des enveloppes observables de ces deux séquences s'intersectent, toute continuation de ces séquences s'appuyant sur cette boucle est telle que l'intersection des enveloppes observables sera toujours non vide. Nous recherchons donc de telles séquences en nombre fini.

Les paires recherchées sont semblables à une séquence d'un twin-plant, où la structure parcourue correspond au produit synchrone de G avec lui même synchronisé sur les transitions observables (on utilise ici uniquement la partie STÉ de la structure de G pour réaliser ce produit).

Soit C_p l'ensemble des couples (σ_1, σ_2) tels que :

1. $\exists \sigma_f \in pref(\sigma_1)$, $\pi_f \in \mathcal{C}_f(\Omega)$
2. $\nexists \sigma_f \in pref(\sigma_2)$, $\pi_f \in \mathcal{C}_f(\Omega)$
3. $\mathbf{P}_{\Sigma_o}(\sigma_1) = \mathbf{P}_{\Sigma_o}(\sigma_2)$
4. $\exists (\sigma_1', \sigma_1'', \sigma_2', \sigma_2'') \in T^{\Theta *4}$, $\exists (t_{1o}, t_{1o}', t_{2o}, t_{2o}') \in T_o^{\Theta 4}$ tels que :
   - $\sigma_1 = \sigma_1'.t_{1o}.\sigma_1''.t_{1o}'$
   - $\sigma_2 = \sigma_2'.t_{2o}.\sigma_2''.t_{2o}'$
   - $\ell(t_{1o}) = \ell(t_{2o})$, $\ell(t_{1o}') = \ell(t_{2o}')$
   - $\sigma_1''.t_{1o}'$ (resp. $\sigma_2''.t_{2o}'$) est tirable après σ_1 (resp. σ_2) dans G
5. $\nexists (\sigma_3, \sigma_4) \in T^{\Theta *2}$, $\sigma_1''.t_{1o}' = \sigma_3.\sigma_3$, $\sigma_2''.t_{2o}' = \sigma_4.\sigma_4$, et $\mathbf{P}_{\Sigma_o}(\sigma_3) = \mathbf{P}_{\Sigma_o}(\sigma_4)$

La condition 1 impose que l'ensemble des exécutions s'appuyant sur σ_1 satisfont le motif temporel. La condition 2 impose que l'ensemble des exécutions s'appuyant sur σ_2 ne satisfont jamais le motif temporel. La condition 3 impose que σ_1 et σ_2 admettent la même projection observable. La condition 4 impose que la dernière transition de σ_1 et σ_2 sont observables, et qu'il est possible pour σ_1 et σ_2 de répéter la sous-séquence terminant ces séquences (autrement dit σ_1 et σ_2 terminent sur une séquence qui peut être répétée indéfiniment). La condition 5 impose qu'il n'existe pas de sous séquence qui peut être répétée à l'infini pour la paire (σ_1, σ_2). Ainsi une paire (σ_1, σ_2) est donc composée de deux séquences de taille finie. De plus, il existe un nombre fini de couples satisfaisant cette définition. Pour se convaincre de cela, nous savons que le produit synchrone de deux STÉ admet un nombre fini d'états. σ_1 et σ_2 sont de taille finie, et telles qu'il n'est pas possible d'extraire de boucles à l'intérieur de $\sigma'_1.t'_{1o}$ et $\sigma'_2.t'_{2o}$. Il y a un nombre d'états fini dans le produit synchrone, et donc un nombre fini de boucles, et donc un nombre fini d'assemblages de boucles. C_p est donc de cardinal fini.

Proposition 14. *Soit $c_p = (\rho_1, \rho_2)$ une paire critique de Θ formée par une trace d'un chemin fautif ρ_1 et une trace d'un chemin non fautif ρ_2. Soit (σ_1, σ_2) les supports infinis de ρ_1 et ρ_2. Il existe $(\sigma'_1, \sigma'_2) \in C_p$ tel que $\sigma'_1 \in pref(\sigma_1)$ et $\sigma'_2 \in pref(\sigma_2)$.*

Démonstration. ρ_1 est de taille infinie, σ_1 l'est donc aussi, de même pour σ_2. Il existe un nombre fini de classes dans G, et un nombre fini de transitions dans T^{Θ}. σ_1 et σ_2 contiennent donc une boucle se répétant à l'infini. Les deux séquences partagent la même projection observable, cette projection contient donc une boucle qui se répète à l'infini. Considérons σ_f un préfixe de σ_1 qui est le support d'un chemin fautif, et considérons $\sigma_{\neg f}$ un préfixe de σ_2 partageant la même projection observable que σ_f et terminant sur une transition observable. Considérons deux suites σ'_1 et σ'_2 de σ_f et $\sigma_{\neg f}$ respectivement dans σ_1 et σ_2, telles que :

— $\mathbf{P}_{\Sigma_o}(\sigma'_1) = \mathbf{P}_{\Sigma_o}(\sigma'_2)$
— Les dernières transitions de σ'_1 et σ'_2 sont observables
— Il n'existe pas (σ_3, σ_4) avec $\mathbf{P}_{\Sigma_o}(\sigma_3) = \mathbf{P}_{\Sigma_o}(\sigma_4)$, $\sigma'_1 = \sigma_3.\sigma_3$ et $\sigma'_2 = \sigma_4.\sigma_4$

Comme σ_1 et σ_2 sont infinies (avec un nombre fini de transitions), il existe des boucles se répétant à l'infini. De plus, ces deux séquences partagent la même projection observable. En prenant parmi les boucles partant de σ_f et $\sigma_{\neg f}$ la plus petite, nous avons bien l'existence de σ'_1 et σ'_2. La paire de séquences $(\sigma_f.\sigma'_1, \sigma_{\neg f}.\sigma'_2)$, cette paire satisfait les conditions 1, 2, 3 et 5 de la définition de C_p. De plus, en posant $\sigma_f = \sigma''_1.t_{1o}$ et $\sigma_{\neg f} = \sigma''_2.t_{2o}$, nous avons bien $\ell^{\Theta}(t_{1o}) = \ell^{\Theta}(t_{2o})$. De plus σ'_1 et σ'_2 sont des boucles de même projection observables, elles sont donc tirables depuis $\sigma_f.\sigma'_1$ et $\sigma_{\neg f}.\sigma'_2$. Ainsi la condition est respectée, et nous avons construit une paire de préfixes de support de paire critique qui appartient à C_p. □

C_p est donc l'ensemble de paires de séquences de transitions qui sont à l'origine des paires critiques de Θ, excepté les paires critiques dont l'origine est l'existence d'un Π^a_f non vide.

Proposition 15. *Soit $(\sigma_1, \sigma_2) \in C_p$, et considérons Π_1 et Π_2 les enveloppes temporelles de σ_1 et σ_2. Soit ν telle que $\nu \models \Pi^{struct}_{\Lambda} \wedge \Pi^{\neg f}_{\Lambda} \wedge \Pi^{\neg a}_{\Lambda}$. Les enveloppes temporelles observables de σ_1 et σ_2 sont telles que : $\Pi^{o,\nu}_1(\Lambda) = \Pi^{c,\nu}_1(\Lambda)$ et $\Pi^{o,\nu}_2(\Lambda) = \Pi^{s,\nu}_2(\Lambda)$. Autrement dit, toute exécution de support σ_1 satisfait le motif, et toute exécution de support σ_2 ne satisfait pas le motif.*

Démonstration. Revenons sur la définition de C_p, et rappelons que dans cette section nous considérons une valuation ν telle que $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^{\neg a}$. D'après le Théorème 6, alors pour tout chemin fautif $\pi_f^\nu(\Lambda)$, nous avons pour toute les contraintes du chemin du motif π_Ω qui lui est associé $\alpha_c \leq \alpha_i^\nu(\Lambda)$ et $\beta_c \geq \beta_i^\nu(\Lambda)$. Ainsi toute exécution d'un chemin fautif satisfait les contraintes c de $\Pi_\Omega[Sub]$, l'enveloppe observable temporelle d'un chemin fautif est ainsi égale à son polyèdre certain.

Soit $(\sigma_1, \sigma_2) \in C_p$. D'après la définition de C_p σ_1 admet un préfixe qui est le support d'un chemin fautif. Comme toute exécution d'un chemin fautif est fautive avec la valuation ν, alors toute exécution de support σ_1 satisfait le motif, d'où $\Pi_1^{o,\nu}(\Lambda) = \Pi_1^{c,\nu}(\Lambda)$. σ_2 n'admet aucun préfixe qui est le support d'un chemin fautif, les exécutions de support σ_2 sont donc nécessairement ne satisfont pas le motif, d'où $\Pi_2^{o,\nu}(\Lambda) = \Pi_2^{s,\nu}(\Lambda)$. □

Les valuations ν considérées sont telles qu'elles satisfont $\mathcal{N}_f(\Lambda, \nu)$, $\mathcal{F}(\Lambda, \nu)$ et $\mathcal{P}_a(\Lambda, \nu)$ (Théorème 6), ainsi les seules sources possibles de paires critiques sont les paires critiques du second type (c.-à-d. une paire de chemins, l'un étant l'extension d'un chemin fautif, pour lesquels l'intersection des polyèdres certains et sauf est non vide). Ainsi pour un tel ν, C_p contient l'ensemble des potentielles origines de paires critiques pour le système paramétré Θ_Λ^ν. Si ν est telle que pour tout élément de C_p, l'intersection des enveloppes temporelles observables est vide, alors il n'existe pour Θ_Λ^ν aucune source de paire critique.

Considérons un couple $(\sigma_1, \sigma_2) \in C_p$, et Sub' la substitution qui fait correspondre les variables des enveloppes temporelles observables de σ_1 et σ_2. Montrons le résultat suivant :

$$(\Pi_1^o \wedge \Pi_2^o[Sub'] \models \bot) \Leftrightarrow (\exists (c_1, c_2) \in \Pi_1^o \times \Pi_2^o[Sub'], (c_1 \wedge c_2 \models \bot))$$

Tout d'abord, Π_1^o et $\Pi_2^o[Sub']$ sont les enveloppes temporelles observables de σ_1 et σ_2, elles sont donc satisfiables. Si $\Pi_1^o \wedge \Pi_2^o[Sub'] \models \bot$, il existe au moins deux contraintes $c_1 \in \Pi_1^o$ et $c_2 \in \Pi_2^o[Sub']$ incompatibles. Si c_1 et c_2 ne concernent pas les mêmes variables, la conjonction est satisfiable. Si pour tout couple c_1 et c_2 admettant au moins une variable en commun, la conjonction est satisfiable, alors $\Pi_1^o \wedge \Pi_2^o[Sub']$ est satisfiable, ce qui est incohérent avec notre hypothèse. Ainsi il existe $(c_1, c_2) \in \Pi_1^o \times \Pi_2^o[Sub']$ tels que $(c_1 \wedge c_2 \models \bot)$.

Supposons maintenant que $\exists (c_1, c_2) \in \Pi_1^o \times \Pi_2^o[Sub']$ tels que $(c_1 \wedge c_2 \models \bot)$. Nécessairement $\Pi_1^o \wedge \Pi_2^o[Sub'] \models \bot$.

Nous venons donc de voir que l'intersection des enveloppes temporelles de deux chemins dont les supports partagent la même projection observable est vide si et seulement si il existe deux contraintes donc la conjonction n'est pas satisfiable. D'après la Proposition 15, pour toute paire (σ_1, σ_2) de C_p, les enveloppes temporelles observables de σ_1 et σ_2 correspondent à leurs polyèdres certain et sauf respectivement. Ainsi, pour que (σ_1, σ_2) ne soit pas une origine de paire critique, il faut que l'intersection de leurs enveloppe temporelle soit vide.

Proposition 16. *Soit* $(\sigma_1, \sigma_2) \in C_p$. *L'ensemble des solutions* $\mathcal{S}(\Pi_1^{o,\nu}(\Lambda) \wedge \Pi_2^{o,\nu}[Sub'](\Lambda))$ *est vide si et seulement si* $\exists (c_1(\Lambda), c_2(\Lambda)) \in \Pi_1^{o,\nu}(\Lambda) \times \Pi_2^{o,\nu}[Sub'](\Lambda)$, $c_1 \wedge c_2 \models \bot$.

Comme cela a été vu avec $\Pi_\Lambda^{\neg f}$, un tel prédicat s'exprime sous la forme d'une disjonction sur l'ensemble des contraintes. Par la suite nous noterons $\Pi_\Lambda^{\neg Cp}$ l'ensemble de ces contraintes pour tout les éléments de C_p.

Exemple 22. Considérons le couple $(\Omega^2, \Theta_{\Lambda_2})$. Nous avons vu précédemment que $\mathcal{C}_f(\Omega^2) = \{\pi_f, \pi\}$ (présentés dans l'Exemple 21 page 94). Nous allons calculer un élément de C_p avec chacun de ces chemins fautifs. Ces deux chemins admettent des supports qui partagent la même projection observable (à savoir o). Si un chemin non fautif forme avec le premier chemin fautif un élément de C_p, ce même chemin non fautif forme donc nécessairement avec le second chemin fautif un autre élément de C_p. Nous pouvons donc exhiber les deux paires de séquences suivantes : cp_1=$(t_0.t_5.t_8.t_0.t_5, t_2.t_3.t_6.t_2.t_3)$ et cp_2=$(t_1.t_4.t_7.t_1.t_4, t_2.t_3.t_6.t_2.t_3)$. Certaines transitions apparaissant deux fois, la variable associée à leur seconde apparition présente un exposant 2 pour éviter toute ambiguïté (les contraintes restent linéaires). En écrivant leurs enveloppes temporelles observables, nous obtenons pour cp_1 :

— $t_0.t_5.t_8.t_0.t_5 : \{\lambda_2^2 \leq y_5 \leq \lambda_2^1 + 2, 1 \leq y_8 - y_5 \leq 2, \lambda_2^2 \leq y_5^2 - y_8 \leq \lambda_2^1 + 2\}$
— $t_2.t_3.t_6.t_2.t_3 : \{1 \leq y_3 \leq 5, \lambda_2^3 \leq y_6 - y_3 \leq \lambda_2^4, 1 \leq y_3^2 - y_6 \leq 5\}$

De même, nous obtenons pour cp_2 :

— $t_1.t_4.t_7.t_1.t_4 : \{3 \leq y_1 \leq 5, 1 \leq y_4 - y_1 \leq 2, 3 \leq y_4^2 - y_7 \leq 5\}$
— $t_2.t_3.t_6.t_2.t_3 : \{1 \leq y_3 \leq 5, \lambda_2^3 \leq y_6 - y_3 \leq \lambda_2^4, 1 \leq y_3^2 - y_6 \leq 5\}$

Pour assurer que les enveloppes temporelles observables de ces paires ne s'intersectent pas, il faut donc d'après la Proposition 16 qu'il existe au moins deux contraintes (relatives à la même observation, c.-à-d. le même événement à la même place dans la séquence d'observations) dont la conjonction ne peut être satisfaite. Pour cela, la borne inférieure de l'une doit être strictement supérieure à la borne supérieure de l'autre. Nous obtenons donc les contraintes suivantes :

$$cp_1 : \begin{cases} \lambda_2^2 > 5 \\ \vee \\ 1 > \lambda_2^1 + 2 \\ \vee \\ 1 > \lambda_2^4 \\ \vee \\ \lambda_2^3 > 2 \\ \vee \\ \lambda_2^2 > 5 \\ \vee \\ 1 > \lambda_2^1 + 2 \end{cases} \tag{5.4}$$

$$cp_2 : \begin{cases} \bot(3 > 5) \\ \vee \\ \bot(1 > 5) \\ \vee \\ 1 > \lambda_2^4 \\ \vee \\ \lambda_2^3 > 2 \\ \vee \\ \bot(3 > 5) \\ \vee \\ \bot(1 > 5) \end{cases} \tag{5.5}$$

Nous pouvons remarquer ici que les deux dernières contraintes de chaque ensemble sont les mêmes que les deux premières. Cela vient du fait que le système boucle est réinitialisable, et que pour satisfaire les conditions de C_p les deux premières transitions de chaque chemin doivent être intégrées une deuxième fois (pour cet exemple ci).

Nous posons également la proposition suivante :

$$\mathcal{C}_p(\Lambda, \nu) : \forall(\sigma_1, \sigma_2) \in C_p, \Pi_1^o(\Lambda) \wedge \Pi_2^o[Sub'](\Lambda) \models \bot$$

5.3 Théorème de Diagnosticabilisabilité

Nous avons vu tout au long de ce chapitre comment construire des ensembles de contraintes permettant que le système paramétré généré par une valuation satisfaisant ces contraintes présente des propriétés de conservation d'une part (graphe des classes isomorphes à celui du système initial, ensemble de chemins fautifs et non fautifs), et des propriétés de non ambiguïté dans les exécutions (absence d'ambiguïtés intrinsèques à des exécutions d'un chemin fautif, absence de paires critiques).

En assemblant ces différents ensembles de contraintes, nous pouvons donc énoncer le résultat

de diagnosticabilisabilité suivant :

Théorème 7. $\nu \models \Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^{\neg a} \wedge \Pi_\Lambda^{\neg C_p} \Rightarrow \mathcal{N}_f(\Lambda,\nu) \wedge \mathcal{F}(\Lambda,\nu) \wedge \mathcal{P}_a(\Lambda,\nu) \wedge \mathcal{C}_p(\Lambda,\nu)$

Démonstration. D'après le Théorème 6, en considérant une valuation ν satisfaisant ce théorème, nous avons $\mathcal{N}_f(\Lambda,\nu) \wedge \mathcal{F}(\Lambda,\nu) \wedge \mathcal{P}_a(\Lambda,\nu)$. Autrement dit, le système engendré Θ_Λ^ν est tel qu'aucun chemin fautif de Θ_Λ^ν ne contient d'ambiguïtés à l'intérieur de ses exécutions. De plus, si ν satisfait la Proposition 16 (c'est à dire que $\nu \models \Pi_\Lambda^{\neg C_p}$), alors pour tous les couples de C_p, leurs enveloppes temporelles présentent une intersection vide. Ainsi $\mathcal{C}_p(\Lambda,\nu)$ est vrai. □

Nous avons donc synthétisé un ensemble de contraintes qui est tel que s'il est satisfiable, alors le système paramétré produit par une valuation satisfaisant cet ensemble est diagnosticable pour le motif temporel étudié. Ainsi si cet ensemble est satisfiable, le système est diagnosticabilisable temporellement pour ce motif, cet ensemble de paramètres, et ce domaine d'existence des paramètres.

Théorème 8. *Si l'ensemble des solution* $\mathcal{S}(\pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^{\neg a} \wedge \Pi_\Lambda^{\neg C_p})$ *est non vide, alors le système* Θ *est diagnosticabilisable pour l'ensemble de paramètres* Λ *défini sur* D_Λ *:* $\mathcal{D}^{ble}(\Theta,\Omega,\Lambda,D_\Lambda)$.

Démonstration. Supposons qu'il existe ν satisfaisant $\Pi_\Lambda^{struct} \wedge \Pi_\Lambda^{\neg f} \wedge \Pi_\Lambda^{\neg a} \wedge \Pi_\Lambda^{\neg C_p}$. D'après le Théorème 6, en considérant une valuation ν satisfaisant ce théorème, le système engendré Θ_Λ^ν est tel qu'aucun chemin fautif de Θ_Λ^ν ne contient d'ambiguïtés à l'intérieur de ses exécutions. De plus, si ν satisfait la Proposition 16 (c'est à dire que $\nu \models \Pi_\Lambda^{\neg C_p}$), alors pour tous les couples de C_p, leurs enveloppes temporelles présentent une intersection vide. En appliquant la Proposition 14, comme pour tout couple de C_p il n'est pas possible de construire une trace observable commune, il n'existe donc pas de paires critiques dont les supports sont des séquences de transitions différentes. Ainsi le système engendré est Ω diagnosticable. Si un tel ν existe, le système est donc (Ω,Λ)-diagnosticabilisable. □

Exemple 23. Le couple $\Omega^2, \Theta_{\Lambda_2}$ admet une solution ν telle que $\Theta_{\Lambda_2}^\nu$ est Ω^2-diagnosticable. En effet, en posant $\nu(\lambda_2^1) = 3$, $\nu(\lambda_2^2) = 1$, $\nu(\lambda_2^3) = 3$ et $\nu(\lambda_2^4) = 4$, nous obtenons un système diagnosticable pour Ω^2. Cette solution est représentée Figure 5.2.

Nous avons proposé une méthode permettant dans notre cadre d'hypothèse de vérifier si un système est diagnosticabilisable pour un motif et un ensemble de paramètres donné avec une solution qui n'est pas très intrusive pour le système. Ainsi une solution de ces contraintes conserve la structure générale du système, et ne modifie que certaines bornes des intervalles statique du système, et ce sans modifier l'ordre de ses différentes opérations. Une telle méthode n'est qu'un premier résultat et demande des améliorations avant d'être appliquée. Cette méthode est très sensible aux problèmes d'explosion combinatoire, ce qui pose un problème de taille si le système étudié est conséquent et contient beaucoup de paramètres. Malgré tout cette méthode est une première solution de vérification, et doit pouvoir s'adapter à des problèmes d'estimabilité et de commandabilité.

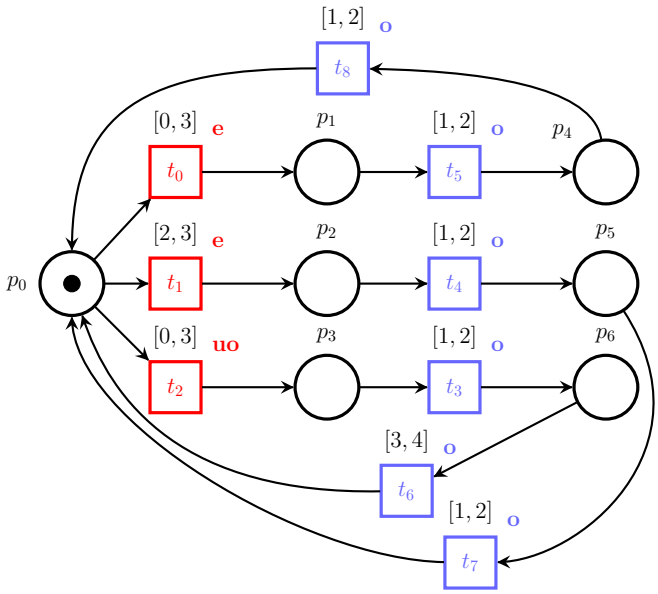


FIGURE 5.2 – $\Theta^{\nu}_{\Lambda_2}$ est un système Ω^2-diagnosticable

5.4 Conclusion

Dans ce chapitre la dernière contribution de ce travail a été développée. Cette contribution consiste en la définition du problème de diagnosticabilisation temporelle, et en une méthode de vérification de cette propriété pour un système en assurant la conservation du langage atemporel du système. La méthode proposée développe la synthèse de contraintes portant sur les bornes des intervalles statiques du système initial permettant si elles sont satisfiables d'obtenir un système paramétré conservant la même structure que le système initial, et où seulement certaines bornes des intervalles statiques des transitions du système initial sont modifiées.

Vérifier un tel système demande l'utilisation de solveur de type solveur SAT Modulo Theory (SMT) comme le solveur Z3 [MB08] [4]. L'analyse proposée ne s'intéresse ni à la complexité de la méthode ni au nombre de variables de l'ensemble de contraintes synthétisé. L'étude de cette perspective est nécessaire pour vérifier l'applicabilité de cette méthode. Enfin, chercher des heuristiques pour trouver des solutions de manière plus efficaces serait une piste intéressante à étudier.

4. https://github.com/Z3Prover/z3

Conclusion et Perspectives

Conclusion

Pour conclure ce manuscrit, revenons sur les contributions proposées dans celui-ci. La première contribution présentée est une abstraction finie des préfixes de taille fixée du langage temporel d'un réseau de Petri temporel sauf labellisé. Cette abstraction appelée *chemin* permet notamment d'étudier l'intersection entre les préfixes de deux langages temporels de taille n fixée.

La seconde contribution proposée est une condition nécessaire et suffisante de diagnosticabilité pour les motifs temporels dans les réseaux de Petri temporels sauf labellisés. Cette condition s'appuie sur un découpage de l'enveloppe temporelle des chemins abstrayant les exécutions satisfaisant le motif en fonction de propriétés de diagnostic pour les exécutions satisfaisant ces zones.

La dernière contribution proposée consiste en la définition du problème de diagnosticabilisation temporelle. Pour ce problème, des paramètres sont posés sur certaines bornes des intervalles statiques du système, et l'on cherche à savoir s'il existe une valuation de ces paramètres telles que le système engendré est diagnosticable pour le motif étudié. Une méthode de vérification est ensuite proposée. Cette méthode repose sur la synthèse de contraintes qui, si elles sont satisfaites, assurent des propriétés de conservation et de diagnostic vis-à-vis du motif étudié et du système initial.

Perspectives

Le travail présenté dans ce manuscrit met en lumière quelques continuations théoriques et de nouveaux problèmes à étudier.

La première piste est celle de l'implémentation. Comme cela a été mentionné précédemment, une implémentation pour la diagnosticabilité de fautes simples temporelle a été réalisée en C++ dans DiaDES. Il faut maintenant étendre cette implémentation aux motifs temporels d'une part, et y intégrer les résultats de diagnosticabilisation d'autre part. Pour implémenter la diagnosticabilisation des optimisations de la méthode proposée sont très certainement à prévoir. L'utilisation de solveur SMT semble une très bonne option pour vérifier la satisfiabilité de l'ensemble de contraintes construit.

Une deuxième piste à explorer est d'étudier l'impact de l'intégration d'intervalles ponctuels pour un système. Si une transition possède un intervalle ponctuel, il n'y a pas d'incertitude quant à son tir. D'un certain point de vue, un intervalle ponctuel déterminise la date de tir d'une transition.

Cependant l'intégration de tels intervalles impose de changer la manière que nous avons de raisonner sur des réseaux de Petri temporels. Il serait donc intéressant d'étudier de tels systèmes.

Une autre piste à explorer à la lumière des résultats présentés dans le Chapitre 4 est d'exhiber une condition de K-diagnosticabilité et de Δ-diagnosticabilité à partir GATR. En effet si un système Θ est diagnosticable pour un motif donné Ω, alors $G_r(\Theta, \Omega)$ le GATR associé au problème ne contient pas de boucles. Ainsi en considérant la longueur de la plus longue séquence d'observations permettant d'accéder à un état bloquant, nous obtenons le K minimal pour lequel le système est $K-\Omega$-diagnosticable. De même, en considérant la date de tir au plus tard de la dernière transition de toutes les branches la plus grande, nous obtenons le plus petit Δ tel que Θ est $\Delta-\Omega$-diagnosticable.

Il serait intéressant de se pencher sur une propriété appelée *prédictabilité*. La prédictabilité d'un système Θ pour un motif temporel Ω donné est la propriété du système de permettre de toujours pouvoir décider avec certitude si le motif va avoir lieu dans la continuation d'une exécution. Cette propriété a été étudiée pour les réseaux de Petri et les motifs atemporels dans [Lub+22], dans lequel le problème est résolu en construisant une structure de Kripke par clôture non observable et déterminisation sur un produit du système avec le motif. Le produit est réalisé avec TWINA, puis le model checker SE-LTL est appelé pour vérifier une propriété LTL utilisant des opérateurs du μ-calcul. Une condition similaire à la condition nécessaire et suffisante de diagnosticabilité peut être synthétisée en utilisant les chemins fautifs. En effet, s'il est toujours possible de prévoir l'occurrence du motif avant qu'il ai lieu, alors il n'existe pas de couple de chemins non fautif et fautif synchronisables. Un motif temporel arrivant nécessairement en temps fini, le problème de prédictabilité est plus simple à vérifier que le problème de diagnosticabilité, car le problème de diagnosticabilité demande une condition sur les séquences infinies alors que les traces observables à étudier pour vérifier la prédictabilité sont toutes de taille finie.

Les motifs temporels étudiés dans cette thèse présentent deux restrictions importantes. La première est leur caractère acyclique. La seconde est leur temps d'exécution qui est nécessairement fini. Les motifs étudié dans [LLL23] ne présentent pas ces limites. Ainsi il serait intéressant de voir si les résultats proposés s'adaptent à des motifs contenant des boucles ou ne présentant pas de borne supérieure pour leur date d'occurrence.

Quant au problème de diagnosticabilisation temporelle, il suppose dans ce manuscrit que le système est déjà paramétré. Une piste intéressante à considérer serait de poser le problème suivant : considérant un système non diagnosticable pour un motif temporel, quel est l'ensemble des paramètres minimaux à poser sur ce système pour pouvoir le rendre diagnosticable ? Une piste de résolution de ce problème est de poser un problème d'optimisation.

Enfin, la dernière perspective vient du titre original du sujet de thèse traité ici : la synthèse d'un diagnostiqueur certifié pour un motif temporel. Le travail proposé permet de vérifier si un système est diagnosticable, et dans le cas où il ne l'est pas propose une méthode pour vérifier s'il peut le devenir. En considérant maintenant un système diagnosticable pour un motif temporel, il est possible de synthétiser un diagnostiqueur en utilisant les chemins fautifs. Un tel diagnostiqueur a été proposé dans [CSP22], où nous avons développé un objet appelé *Observable Simple Temporal Network* (OSTN) qui consiste en un ensemble d'événements observables dont l'occurrence est soumise à un ensemble de contraintes. Pour un système diagnosticable, toute séquence d'observations non ambigüe peut être abstraite par un chemin. En considérant les événements observables de ce chemin et en utilisant son enveloppe temporelle observable pour obtenir les contraintes de leurs dates de tirs, nous obtenons un OSTN. Le problème d'un OSTN est qu'il contient toutes les observations

du support du chemin. Dans de nombreux cas toutes ses observations ne sont pas pertinentes. Il serait intéressant de supprimer les observations qui ne particularisent pas le chemin considéré et de l'abstraire sous la forme d'une *chronique*.

References

[Aho+00] Alfred V Aho, Ravi Sethi, Jeffrey D Ullman et Pierre Boullier. *Compilateurs : principes, techniques et outils : cours et exercices*. Dunod, 2000.

[AD94] Rajeev Alur et David L. Dill. « A theory of timed automata ». In : *Theoretical Computer Science* 126.2 (1994), p. 183-235. issn : 0304-3975. doi : https://doi.org/10.1016/0304-3975(94)90010-8. url : https://www.sciencedirect.com/science/article/pii/0304397594900108.

[AHV93] Rajeev Alur, Thomas A Henzinger et Moshe Y Vardi. « Parametric real-time reasoning ». In : *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*. 1993, p. 592-601.

[And19] Étienne André. « What's decidable about parametric timed automata ? » In : *Int. J. Softw. Tools Technol. Transf.* 21.2 (2019), p. 203-219. doi : 10.1007/s10009-017-0467-0. url : https://doi.org/10.1007/s10009-017-0467-0.

[And21] Étienne André. « IMITATOR 3 : Synthesis of Timing Parameters Beyond Decidability ». In : *Computer Aided Verification - 33rd International Conference, CAV 2021, Virtual Event, July 20-23, 2021, Proceedings, Part I*. Sous la dir. d'Alexandra Silva et K. Rustan M. Leino. T. 12759. Lecture Notes in Computer Science. Springer, 2021, p. 552-565. doi : 10.1007/978-3-030-81685-8_26. url : https://doi.org/10.1007/978-3-030-81685-8%5C_26.

[Bac92] F. Baccelli. *Synchronization and Linearity : An Algebra for Discrete Event Systems*. Probability and Statistics Series. Wiley, 1992. isbn : 9780471936091. url : https://books.google.fr/books?id=l8FnQgAACAAJ.

[BPS20] Johanne Bakalara, Yannick Pencolé et Audine Subias. « How to use Model Checking for Diagnosing Fault Patterns in Petri nets ». In : *IFAC-PapersOnLine* 53.4 (2020), p. 269-274.

[BF22] F. Basile et L. Ferrara. « A Matlab toolbox implementing MSCG computation ». In : *IFAC-PapersOnLine* 55.28 (2022). 16th IFAC Workshop on Discrete Event Systems WODES 2022, p. 303-308. issn : 2405-8963. doi : https://doi.org/10.1016/j.ifacol.2022.10.358. url : https://www.sciencedirect.com/science/article/pii/S2405896322023953.

[Bas14] Francesco Basile. « Overview of fault diagnosis methods based on Petri net models ». In : *2014 European Control Conference (ECC)*. 2014, p. 2636-2642. doi : 10.1109/ECC.2014.6862631.

[BCS15a] Francesco Basile, Maria Paola Cabasino et Carla Seatzu. « State Estimation and Fault Diagnosis of Labeled Time Petri Net Systems With Unobservable Transitions ». In : *IEEE Transactions on Automatic Control* 60.4 (2015), p. 997-1009. doi : 10.1109/TAC.2014.2363916.

[BCS15b] Francesco Basile, Maria Paola Cabasino et Carla Seatzu. « State Estimation and Fault Diagnosis of Labeled Time Petri Net Systems With Unobservable Transitions ». In : *IEEE Transactions on Automatic Control* 60.4 (2015), p. 997-1009. doi : 10.1109/TAC.2014.2363916.

[BCS17] Francesco Basile, Maria Paola Cabasino et Carla Seatzu. « Diagnosability Analysis of Labeled Time Petri Net Systems ». In : *IEEE Transactions on Automatic Control* 62.3 (2017), p. 1384-1396. doi : 10.1109/TAC.2016.2588736.

[BCC15] Francesco Basile, Pasquale Chiacchio et Jolanda Coppola. « Model repair of Time Petri Nets with temporal anomalies ». In : t. 48. Mai 2015, p. 85-90. doi : 10.1016/j.ifacol.2015.06.477.

[BCD12] Francesco Basile, Pasquale Chiacchio et Gianmaria De Tommasi. « On K - diagnosability of Petri nets via integer linear programming ». In : *Automatica* 48.9 (2012), p. 2047-2058.

[BCD08] Francesco Basile, Pasquale Chiacchiot et Gianmaria De Tommasi. « Sufficient conditions for diagnosability of Petri nets ». In : *2008 9th international workshop on discrete event systems*. IEEE. 2008, p. 370-375.

[BDS15] Francesco Basile, Gianmaria De Tommasi et Claudio Sterle. « Sensors selection for K-diagnosability of Petri nets via Integer Linear Programming ». In : *2015 23rd Mediterranean Conference on Control and Automation (MED)*. IEEE. 2015, p. 168-175.

[Bas+18] Francesco Basile, Gianmaria De Tommasi, Claudio Sterle, Abderraouf Boussif et Mohamed Ghazel. « Efficient diagnosability assessment via ILP optimization : a railway benchmark ». In : *2018 IEEE 23rd international conference on emerging technologies and factory automation (ETFA)*. T. 1. IEEE. 2018, p. 441-448.

[BD91] Bernard Berthomieu et Michel Diaz. « Modeling and verification of time dependent systems using time Petri nets ». In : *IEEE transactions on software engineering* 17.3 (1991), p. 259.

[BM83] Bernard Berthomieu et Miguel Menasche. « An enumerative approach for analyzing time Petri nets ». In : *Proceedings IFIP*. Elsevier Science Publishers, 1983, p. 41-46.

[BRV04] Bernard Berthomieu, P-O Ribet et François Vernadat. « The tool TINA – Construction of abstract state spaces for Petri nets and time Petri nets ». In : *International journal of production research* 42.14 (2004), p. 2741-2756.

[BCD05] Patricia Bouyer, Fabrice Chevalier et Deepak D'Souza. « Fault diagnosis using timed automata ». In : *International Conference on Foundations of Software Science and Computation Structures*. Springer. 2005, p. 219-233.

[BL09] Laura Bozzelli et Salvatore La Torre. « Decision problems for lower/upper bound parametric timed automata ». In : *Formal Methods in System Design* 35.2 (2009), p. 121-151.

[CLS13] Maria Paola Cabasino, Stéphane Lafortune et Carla Seatzu. « Optimal sensor selection for ensuring diagnosability in labeled Petri nets ». In : *Automatica* 49.8 (2013), p. 2373-2383. issn : 0005-1098. doi : https://doi.org/10.1016/j.automatica.2013.04.041. url : https://www.sciencedirect.com/science/article/pii/S0005109813002665.

[Cab+09] Maria Paola Cabasino, Alessandro Giua, Stéphane Lafortune et Carla Seatzu. « Diagnosability analysis of unbounded Petri nets ». In : *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*. IEEE. 2009, p. 1267-1272.

[CL08] Christos G Cassandras et Stéphane Lafortune. *Introduction to discrete event systems*. Springer, 2008.

[CGB23] Amira Chouchane, Mohamed Ghazel et Abderraouf Boussif. « K-diagnosability analysis of bounded and unbounded Petri nets using linear optimization ». In : *Automatica* 147 (2023), p. 110689.

[Cla+18] Edmund M Clarke, Thomas A Henzinger, Helmut Veith, Roderick Bloem et al. *Handbook of model checking*. T. 10. Springer, 2018.

[CPS23] Camille Coquand, Yannick Pencolé et Audine Subias. « Diagnosabilization of Time Petri net for timed fault ». In : *22nd World Congress of the International Federation of Automatic Control*. 2023.

[CSP21] Camille Coquand, Audine Subias et Yannick Pencolé. « Signature of timed patterns in time Petri nets : a formal characterization ». In : *Modélisation des Systèmes Réactifs (MSR'21)*. Paris, France, nov. 2021.

[CSP22] Camille Coquand, Audine Subias et Yannick Pencolé. « Observable Simple Temporal Network synthesis for the diagnosis of time patterns in time Petri nets ». In : *IFAC-PapersOnLine* 55.6 (2022). 11th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS 2022, p. 539-544. issn : 2405-8963.

[Coq+22] Camille Coquand, Audine Subias, Yannick Pencolé et Éric Lubat. « Critical pairs based diagnosability analysis of timed fault in Time Petri Nets ». In : *IFAC-PapersOnLine* 55.28 (2022). 16th IFAC Workshop on Discrete Event Systems WODES 2022, p. 297-302.

[Dia13] Michel Diaz. *Petri nets : fundamental models, verification and applications*. John Wiley & Sons, 2013.

[DGG93] Christophe Dousson, Paul Gaborit et Malik Ghallab. « Situation recognition : Representation and algorithms ». In : *IJCAI*. T. 93. 1993, p. 166-172.

[GRR04] Guillaume Gardey, Olivier H Roux et Olivier F Roux. « Using zone graph method for computing the state space of a time Petri net ». In : *Formal Modeling and Analysis of Timed Systems : First International Workshop, FORMATS 2003, Marseille, France, September 6-7, 2003. Revised Papers 1*. Springer. 2004, p. 246-259.

[GTY09] Mohamed Ghazel, Armand Toguyéni et Pascal Yim. « State observer for DES under partial observation with time Petri nets ». In : *Discrete Event Dynamic Systems* 19 (2009), p. 137-165.

[Gor17] Roberto Gorrieri. « Labeled Transition Systems ». In : *Process Algebras for Petri Nets : The Alphabetization of Distributed Systems*. Cham : Springer International Publishing, 2017, p. 15-34. ISBN : 978-3-319-55559-1. DOI : 10.1007/978-3-319-55559-1_2. URL : https://doi.org/10.1007/978-3-319-55559-1_2.

[GPS17] HE. Gougam, Y. Pencolé et A. Subias. « Diagnosability analysis of patterns on bounded labeled prioritized Petri nets ». In : *Discrete Event Dyn Syst 27* (2017), p. 143-180. DOI : 10.1007/s10626-016-0234-5.

[Haa+03] S. Haar, A. Benveniste, E. Fabre et C. Jard. « Partial order diagnosability of discrete event systems using petri net unfoldings ». In : *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*. T. 4. 2003, 3748-3753 vol.4. DOI : 10.1109/CDC.2003.1271732.

[Hac76] Michel Henri Theódore Hack. « Petri net language ». In : (1976).

[HB10] Rachid Hadjidj et Hanifa Boucheneb. « Efficient reachability analysis for time Petri nets ». In : *IEEE Transactions on Computers* 60.8 (2010), p. 1085-1099.

[HDY22] Lulu He, Philippe Dague et Lina Ye. « Using Delay Blocks to Make Non-Diagnosable Discrete Event Systems Diagnosable ». In : *33rd International Workshop on Principle of Diagnosis – DX 2022*. LAAS-CNRS-ANITI. Toulouse, France, sept. 2022. URL : https://hal.archives-ouvertes.fr/hal-03773712.

[HU79] John E. Hopcroft et Jeff D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley Publishing Company, 1979.

[Hun+01] Thomas Hune, Judi Romijn, Mariëelle Stoelinga et Frits Vaandrager. « Linear parametric model checking of timed automata ». In : *Tools and Algorithms for the Construction and Analysis of Systems : 7th International Conference, TACAS 2001 Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2001 Genova, Italy, April 2–6, 2001 Proceedings 7*. Springer. 2001, p. 189-203.

[Hun+02] Thomas Hune, Judi Romijn, Mariëlle Stoelinga et Frits Vaandrager. « Linear parametric model checking of timed automata ». In : *The Journal of Logic and Algebraic Programming* 52-53 (2002), p. 183-220. ISSN : 1567-8326. DOI : https://doi.org/10.1016/S1567-8326(02)00037-1. URL : https://www.sciencedirect.com/science/article/pii/S1567832602000371.

[Jér+06] T. Jéron, H. Marchand, S. Pinchinat et M.-O. Cordier. « Supervision patterns in discrete event systems diagnosis ». In : *2006 8th International Workshop on Discrete Event Systems*. 2006, p. 262-268. DOI : 10.1109/WODES.2006.1678440.

[Jér+08] Thierry Jéron, Hervé Marchand, Sahika Genc et Stéphane Lafortune. « Predictability of sequence patterns in discrete event systems ». In : *IFAC Proceedings Volumes* 41.2 (2008), p. 537-543.

[Jia+01] Shengbing Jiang, Zhongdong Huang, Venu ch et Ratnesh Kumar. « A Polynomial Algorithm for Testing Diagnosability of Discrete Event Systems ». In : *IEEE Transactions on Automatic Control* 46 (juill. 2001). DOI : 10.1109/9.940942.

[JB10] George Jiroveanu et René K Boel. « The diagnosability of Petri net models using minimal explanations ». In : *IEEE Transactions on Automatic Control* 55.7 (2010), p. 1663-1668.

[Lef14] Dimitri Lefebvre. « Fault diagnosis and prognosis with partially observed Petri nets ». In : *IEEE Transactions on Systems, Man, and Cybernetics : Systems* 44.10 (2014), p. 1413-1424.

[LLL23] Dimitri Lefebvre, Zhiwu Li et Ye Liang. « Diagnosis of timed patterns for discrete event systems by means of state isolation ». In : *Automatica* 153 (2023), p. 111045. issn : 0005-1098. doi : https://doi.org/10.1016/j.automatica.2023.111045. url : https://www.sciencedirect.com/science/article/pii/S0005109823002005.

[LKT20] Ben Li, Manel Khlif-Bouassida et Armand Toguyéni. « Reduction Rules for Diagnosability Analysis of Complex Systems Modeled by Labeled Petri Nets ». In : *IEEE Transactions on Automation Science and Engineering* 17.2 (2020), p. 1061-1069. doi : 10.1109/TASE.2019.2933230.

[Lim+09] Didier Lime, Olivier H. Roux, Charlotte Seidner et Louis-Marie Traonouez. « Romeo : A Parametric Model-Checker for Petri Nets with Stopwatches ». In : *Tools and Algorithms for the Construction and Analysis of Systems, 15th International Conference, TACAS 2009, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2009, York, UK, March 22-29, 2009. Proceedings*. Sous la dir. de Stefan Kowalewski et Anna Philippou. T. 5505. Lecture Notes in Computer Science. Springer, 2009, p. 54-57. doi : 10.1007/978-3-642-00768-2_6. url : https://doi.org/10.1007/978-3-642-00768-2%5C_6.

[Lin94] Feng Lin. « Diagnosability of discrete event systems and its applications ». In : *Discrete Event Dynamic Systems* 4 (1994), p. 197-212.

[LGT14] Baisi Liu, Mohamed Ghazel et Armand Toguyéni. « Diagnosis of labeled time Petri nets using time interval splitting ». In : *IFAC Proceedings Volumes* 47.3 (2014), p. 1784-1789.

[Lub+19] É. Lubat, S. Dal Zilio, D. Le Botlan, Y. Pencolé et A. Subias. « A State Class Construction for Computing the Intersection of Time Petri Nets Languages ». In : *Springer, Cham* 11750 (2019).

[Lub21] Eric Lubat. « Synchronous Product of Time Petri Nets and its Applications to Fault-Diagnosis ». Thèse de doct. INSA de Toulouse, 2021.

[Lub+20] Éric Lubat, Silvano Dal Zilio, Didier Le Botlan, Yannick Pencolé et Audine Subias. « A New Product Construction for the Diagnosability of Patterns in Time Petri Net ». In : *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE. 2020, p. 104-109.

[Lub+22] Éric Lubat, Camille Coquand, Yannick Pencole et Audine Subias. « Diagnosability and Predictability of pattern in Labelled Petri Nets ». In : *33rd International Workshop on Principle of Diagnosis (DX 2022)*. 2022.

[Mer74] Philip M Merlin. « The Time-Petri-Net and the Recoverability of Processes ». In : (1974).

[MB08] Leonardo de Moura et Nikolaj Bjørner. « Z3 : An Efficient SMT Solver ». In : *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*. T. 4963. LNCS. Springer, 2008. doi : 10.1007/978-3-540-78800-3_24.

[PCC02] Charles Pecheur, Alessandro Cimatti et Ro Cimatti. « Formal verification of diagnosability via symbolic model checking ». In : *Workshop on Model Checking and Atificial intelligence (MoChArt-2002), Lyon, France*. 2002.

[PS17] Yannick Pencolé et Audine Subias. « Diagnosis of supervision patterns on bounded labeled Petri nets by Model Checking ». In : *28th International Workshop on Principles of Diagnosis (DX'17), Brescia, Italy, September 26-29, 2017*. Sous la dir. de Marina Zanella, Ingo Pill et Alessandro Cimatti. T. 4. Kalpa Publications in Computing. EasyChair, 2017, p. 184-199. URL : www.easychair.org/publications/paper/n2NN.

[PS21] Yannick Pencolé et Audine Subias. « Diagnosability of event patterns in safe labeled time Petri nets : A model-checking approach ». In : *IEEE Transactions on Automation Science and Engineering* (2021), p. 1-12. DOI : 10.1109/TASE.2020.3045565.

[Sam+95] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen et D. Teneketzis. « Diagnosability of discrete-event systems ». In : *IEEE Transactions on Automatic Control* 40.9 (1995), p. 1555-1575. DOI : 10.1109/9.412626.

[Sch98] Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[TLR08] Louis-Marie Traonouez, Didier Lime et Olivier H. Roux. « Parametric Model - Checking of Time Petri Nets with Stopwatches Using the State-Class Graph ». In : *Formal Modeling and Analysis of Timed Systems, 6th International Conference, FORMATS 2008, Saint Malo, France, September 15-17, 2008. Proceedings*. Sous la dir. de Franck Cassez et Claude Jard. T. 5215. Lecture Notes in Computer Science. Springer, 2008, p. 280-294. DOI : 10.1007/978-3-540-85778-5_20. URL : https://doi.org/10.1007/978-3-540-85778-5%5C_20.

[TLR09] Louis-Marie Traonouez, Didier Lime et Olivier H. Roux. « Parametric Model - Checking of Stopwatch Petri Nets ». In : *J. Univers. Comput. Sci.* 15.17 (2009), p. 3273-3304. DOI : 10.3217/jucs-015-17-3273. URL : https://doi.org/10.3217/jucs-015-17-3273.

[Tri02] Stavros Tripakis. « Fault Diagnosis for Timed Automata ». In : *Formal Techniques in Real-Time and Fault-Tolerant Systems*. Sous la dir. de Werner Damm et Ernst - Rüdiger Olderog. Berlin, Heidelberg : Springer Berlin Heidelberg, 2002, p. 205-221. ISBN : 978-3-540-45739-8.

[UOO98] Toshimitsu Ushio, Isao Onishi et Koji Okuda. « Fault detection based on Petri net models with faulty behaviors ». In : *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*. T. 1. IEEE. 1998, p. 113-118.

[VP99] Irina B Virbitskaite et EA Pokozy. « Parametric behaviour analysis for time petri nets ». In : *International Conference on Parallel Computing Technologies*. Springer. 1999, p. 134-140.

[WMS15] Xu Wang, Cristian Mahulea et Manuel Silva. « Diagnosis of Time Petri Nets Using Fault Diagnosis Graph ». In : *IEEE Transactions on Automatic Control* 60.9 (2015), p. 2321-2335. DOI : 10.1109/TAC.2015.2405293.

[WLJ05] YuanLin Wen, ChunHsi Li et MuDer Jeng. « A polynomial algorithm for checking diagnosability of Petri nets ». In : *2005 IEEE International Conference on Systems, Man and Cybernetics*. T. 3. IEEE. 2005, p. 2542-2547.

[YL17] Xiang Yin et Stéphane Lafortune. « On the decidability and complexity of diagnosability for labeled Petri nets ». In : *IEEE Transactions on Automatic Control* 62.11 (2017), p. 5931-5938.

[YL02] Tae-Sic Yoo et Stéphane Lafortune. « Polynomial-time verification of diagnosability of partially observed discrete-event systems ». In : *IEEE Transactions on automatic control* 47.9 (2002), p. 1491-1495.