# Modélisation Hybride, Analyse et Vérification Quantitative des Grands Réseaux de Régulation Biologique

Louis Fippo Fitime

UNIVERSITE BRETAGNE LOIRE

Centrale Nantes

# Thèse de Doctorat

Louis FIPPO FITIME

*Mémoire présenté en vue de l'obtention du*
***grade de Docteur de l'École centrale de Nantes***
*sous le sceau de l'Université Bretagne Loire*

**École doctorale : Sciences et technologies de l'information et mathématiques**

**Discipline : Informatique**
**Unité de recherche : Institut de Recherche en Communications et Cybernétique de Nantes**

**Soutenue le 28/11/2016**

# Modélisation Hybride, Analyse et Vérification Quantitative des Grands Réseaux de Régulation Biologique

## JURY

| | |
|---|---|
| Présidente : | **M^me^ Anne POUPON**, Directrice de recherche CNRS, INRA de Tours |
| Rapporteurs : | **M^me^ Christine FROIDEVAUX**, Professeur des universités, Université de Paris-Sud<br>**M. Simon DE GIVRY**, Chargé de recherche (HdR) INRA, INRA de Toulouse |
| Examinateurs : | **M. Sylvain SOLIMAN**, Chargé de recherche INRIA, INRIA Paris<br>**M^me^ Carito GUZIOLOWSKI**, Maître de conférences, École centrale de Nantes |
| Directeur de thèse : | **M. Olivier ROUX**, Professeur des universités, École centrale de Nantes |

# Remerciements

Ce travail de thèse n'aurait pas été possible, ou aurait été très différent, sans la présence de nombreuses personnes autour de moi que je souhaite ici remercier du fond du coeur. Tout d'abord, je remercie chaleureusement les membres de mon jury d'avoir accepté d'être dans mon jury de thèse. Tout particulièrement *Anne Poupon* d'avoir présidé ce jury et de l'avoir rendu conviviale. Je remercie *Christine Froideveaux* et *Simon De Givry* pour avoir rapporté mon manuscrit. Les commentaires, les remarques et questions ont été plus qu'utiles et ont permis d'améliorer la qualité de ce travail. Un grand merci également à *Sylvain Soliman* pour son enthousiasme et ses commentaires qui m'encouragent pour la suite.

Merci mille fois à mon directeur de thèse, *Olivier Roux*, et mon encadrante, *Carito Guziolowski*, de m'avoir toujours soutenu et poussé pour aller plus loin. Merci pour la confiance sans faille qu'ils m'ont accordé durant toute cette thèse et au-delà. Merci également pour l'orientation, les discussions, les conseils et l'investissement personnel qu'ils ont consenti dans mon travail.

Cette thèse a bénéficié des collaborations et je tiens à remercier vivement *Loïc Paulevé*, de m'avoir accueilli quelques jours au LRI, ce qui a permis le démarrage d'une collaboration très fructueuse sur l'indentification des bifurcations dans les réseaux biologiques. Je remercie également le centre allemand de lutte contre le cancer et en particulier *Peter Angel*, *Christian Schuster*, pour leur collaboration et la mise à disposition des données de série temporelles.

Je remercie grandement *Anne Poupon* et *Damnien Eveillard*, d'avoir accepté d'être membre du commité de suivi de cette thèse. Les réunions de suivi m'ont aider à prendre le recul nécessaire et à avoir des avis externes qui m'ont été très utiles dans les choix que j'ai eu à effectuer.

Pour l'accueil formidable qui m'a été reservé à l'IRCCyN, je tiens à remercier chaleureusement *Michel Malabre* pour les facilités qu'il a mis à ma disposition, *Virginie Dupont* et *Isabelle Favreau* pour l'assistance dans les démarches administratives, enfin toutes l'équipe administrative et technique. Merci également aux membres de l'équipe MeForBio en particulier à *Morgan Magnin* pour ses conseils toujours pertinents. Merci infiniment à *Maxime Folschette*, qui a su m'accueillir dans le bureau et l'équipe. Merci *Maxime* de m'avoir aidé à prendre en main les outils necessaires pour la mise en œuvre de mes travaux. Merci également à *Courtney Chancellor* pour les bons moments passés dans le bureau. Merci à *Emna* et *Bertrand* pour la bonne ambiance au bureau, les provisions alimentaires et les impressions du manuscrit.

Très amicalement merci aux autres thèsards (dont certains sont désormais docteur) avec lesquels nous avons eu des moments agréables d'échange, de partage, de discussions aux nombreuses sorties (restaurants, lasers game,...) et pendant les pauses café. Ainsi, je dis un grand merci à *Toussaint, Louis-Marie, Adrien B., Adrien Q., Taghreed, Marie, Lauriane, Julien, Johan, Armel, Saab, Lila, Elaheh, Misbah, Benjamin, Jha* et tous ceux que je n'ai pas pu citer ici mais qui sauront que j'ai bien évidement pensé à eux en ce moment.

Je remercie du fond du coeur mes parents, mes frères et soeurs et ma famille toute entière pour leur confiance, soutient et encouragements. Merci infiniment à mes oncles *Martin*, et *Jean-Louis* d'avoir fait le déplacement. Merci grandement *Ghislaine* pour le soutien et le coup de main logistique.

Enfin, je ne pourrai jamais assez te remercier, *Coriane*, d'avoir été à mes côtés pendant ces trois années de thèse, de m'avoir soutenu même pendant mes nombreuses absences au nom de la thèse alors que toi même tu te prépares à devenir docteur...

# Table des matières

# Chapitre 1

# Introduction

## 1.1 Contexte & Motivations

L'étude et la compréhension des systèmes complexes, en particulier des systèmes biologiques requièrent le développement des méthodes et des modèles mathématiques et informatiques innovants. Les systèmes complexes sont largement connus comme un assemblage (souvent très grand) d'éléments simples (souvent hétérogènes) dont les interactions produisent des comportements complexes. Dans le cas particulier des systèmes biologiques (et bien sûr c'est valable pour d'autres types de systèmes complexes), notamment les cellules, les tissus et les organismes, la biologie des systèmes, ambitionne de faire comprendre comment leurs fonctions sont dérivées des interactions entre les composants du système. Ceci nécessite donc une étude globale de la structure et des dynamiques des interactions de l'ensemble des composants du système. Cette étude peut être conduite suivant deux grandes approches dans le domaine de la biologie des systèmes moderne. Une première approche dite *de bas en haut* qui consiste à partir du particulier pour aller au général, en spécifiant les comportements de chaque composant et en intégrant ces différentes spécifications généralement comme un système mécanique afin de prédire le comportement de l'ensemble du système. La deuxième approche est dite *de haut en bas* et se caractérise par l'utilisation du large potentiel des ensembles des données *omics* afin d'identifier les structures ou les modules des réseaux d'interactions, les corrélations, etc.

Indépendamment de l'approche utilisée, les mathématiques et l'informatique doivent proposer des méthodes et des modèles efficaces pour faire face aux nombreux défis suscités par l'étude et la compréhension des systèmes biologiques. En effet, les systèmes biologiques sont intrinsèquement complexes. Aussi, en fonction du degré de complexité des interactions et de leur taille, ils vont générer de nombreux défis. L'un des plus importants est l'explosion combinatoire de l'espace d'états du système. En effet, que l'on soit dans une représentation continue ou discrète, l'espace d'états croit très rapidement avec le nombre de composants du système. Un autre défi est la puissance de calcul nécessaire pour mettre en œuvre certains raisonnements sur les modèles étudiés. Il est évident que certains raisonnements sont intrinsèquement complexes, rendant leur mise en œuvre impossible pour les grands systèmes.

Les méthodes d'analyse dites « statiques »(Cousot & Cousot, 1977) tentent d'apporter une réponse à ces deux défis. La caractéristique d'une analyse statique est sa capacité à dériver des propriétés d'un modèle sans l'exécuter dans sa forme complètement déployée. Cette caractéristique permet de fait à la méthode de contourner la potentielle explosion combinatoire des comportements décrits. Typiquement, les méthodes par analyse statique

produisent des approximations supérieures et inférieures du résultat recherché. Dans certains cas, elles peuvent se révéler non-concluantes selon le cas traité. La construction, la mise en œuvre et l'efficacité d'une analyse statique sont fortement liées à la propriété recherchée et fortement dépendent de la sémantique du langage formel sur lequel l'analyse s'effectue. De nombreuses techniques permettent la mise au point d'une analyse statique. Nous pouvons citer l'analyse topologique du modèle, l'analyse du flot de contrôle, l'utilisation de contraintes, ou encore l'interprétation abstraite pour ne citer que celles-ci.

Développer de telles méthodes pour la biologie des systèmes est fondamental parce qu'elles vont apporter des outils aux acteurs de ce domaine interdisciplinaire, afin de faire face aux défis informatiques et de modélisations auxquels ils sont confrontés.

Aussi, l'objectif de cette thèse est d'apporter une contribution dans ce domaine en proposant une démarche qui permette de modéliser, d'analyser et de vérifier de façon efficace les systèmes biologiques. Nous représentons ces systèmes comme des Réseaux de Régulation Biologique (RRB).

Les RRB modélisent l'évolution des composants (gènes, protéines, complexes, etc.) au sein d'un système biologique en fonction des influences (positives, négatives, associations, dissociations, etc.) qu'ils exercent entre eux. Ils permettent de modéliser et comprendre les différentes fonctions biologiques au sein d'une cellule. Ils sont notamment utilisés pour comprendre les dynamiques de l'expression des gènes et de production des protéines au sein des cellules, les mécanismes de cycle cellulaire et de prolifération cellulaire, le processus conduisant à la mort cellulaire (apoptose), les opérations de détection et de réparation de l'acide désoxyribonucléique (ADN), et cette liste ne se veut pas exhaustive.

La disponibilité des données expérimentales d'expression des gènes à grande échelle, sur les systèmes biologiques étudiés permet d'envisager une approche de modélisation qui combine les deux approches déjà citées (de haut en bas et de bas en haut). Un type des données peut être les séries temporelles qui renseignent sur l'évolution des composants dans le temps. Ces mesures permettent d'envisager un raffinement de la dynamique des RRB en ajoutant des délais aux influences ou aux régulations. Cette intégration de nouvelles données a pour conséquence immédiate que les modèles qui en découlent sont des modèles dit *quantitatifs*. Leur analyse doit donc prendre en compte de nouveaux paramètres et permettre de répondre à des questions qui concernent une évaluation quantitative (probabilité d'observer ou non un comportement, délai pour observer un comportement, etc.).

Nous introduisons dans ce chapitre dans la section 1.2 les systèmes biologiques d'un point de vue de leurs modélisations et de leurs analyses. La section 1.3 introduit les données des séries temporelles et la problématique de leur intégration dans les modèles. La section 1.4 expose les motivations des différentes contributions de cette thèse. Enfin nous présentons le contenu et le plan général de cette thèse à la section 1.5.

## 1.2 Les systèmes biologiques : modélisation & analyse

La modélisation et l'analyse des systèmes biologiques se sont imposées comme un enjeu majeur dans la biologie des systèmes. De nombreuses bases de données (Schaefer, Anthony, Krupa, Buchoff, Day, Hannay & Buetow, 2009*a*; Kanehisa, Sato, Kawashima, Furumichi & Tanabe, 2015) regroupent une quantité importante d'informations sous forme de réseaux. Ces réseaux peuvent être des réseaux de signalisation, des réseaux métaboliques, des réseaux de régulation biologique. Dans cette thèse, nous nous intéressons à la modélisation des systèmes biologiques comme des réseaux de régulation biologique. Les réseaux

de régulation biologique décrivent les fonctions biologiques ou les combinaisons de ces fonctions généralement à l'échelle cellulaire. Du fait de la complexité des systèmes biologiques, la taille des RRB qui les modélisent est importante et nécessite donc des approches innovantes pour faire face à ces problèmes de taille dans le processus de modélisation.

Le processus de modélisation doit trouver un compromis entre le niveau de détail à prendre en compte pour reproduire au mieux les dynamiques et les précisions qu'il est possible de dériver des propriétés des modèles. Cette contrainte a constamment accompagné les scientifiques engagés dans le développement des modèles. Ainsi, quand les systèmes modélisés sont de très petite taille, une des approches est d'avoir recours aux équations différentielles ordinaires qui modélisent les dynamiques en décrivant l'évolution des concentrations des composants (protéines par exemple)(Tyson & Othmer, 1978). Ce cas de figure se produit lorsqu'on s'intéresse à des régulations géniques spécifiques. Cependant, la résolution analytique ou numérique des équations différentielles est parfois très complexe quand elle n'est pas impossible à cause : (1) de la difficulté d'exprimer les solutions analytiques de la plupart des équations différentielles ; (2) du très grand nombre de paramètres à estimer pour les solutions numériques.

Dans beaucoup des cas, les systèmes modélisés font intervenir plusieurs voies métaboliques et sont donc de tailles plus importantes que les régulations spécifiques. La figure 1.1 illustre le fonctionnement d'une cellule. On peut y découvrir les différentes influences permettant de réaliser les fonctions biologiques majeures. Nous pouvons citer comme exemple les facteurs de prolifération cellulaire, l'expression des gènes, la différenciation cellulaire, la mort cellulaire. Pour chacune de ces fonctions, des dizaines, voire des centaines de composants interviennent pour sa réalisation. Les composants interagissent les uns sur les autres à travers les influences ou les régulations. L'ensemble formé des éléments biologiques et de leurs interactions peut être représenté par un graphe d'interactions qui décrit les informations qualitatives sur les régulations entre les entités. Les nœuds du graphe représentent les entités biologiques (gène, ARN, protéine) ou une abstraction de plusieurs entités, et les arcs représentent les influences (positives ou négatives des uns sur les autres). Les graphes d'interactions abstraient l'évolution de la concentration des composants en fonction de leurs régulateurs.

À la fin des années '60 Stuart A. Kauffman (1969) et René Thomas (1973) ont introduit une modélisation discrète des RRB : la concentration ou le niveau d'expression des composants est quantifié en un nombre fini de niveaux discrets ($\{0, 1, 2\}$ par exemple), et l'évolution d'un composant est déterminée en fonction du niveau de ses régulateurs. L'évolution des composants se détermine à travers les paramètres discrets que nous détaillerons dans le chapitre 2. Cette approche a l'avantage d'abstraire les valeurs des seuils de concentration, qui ne sont pas toujours bien connues mais qui permet de représenter le niveau de concentration à partir duquel le composant influence un autre composant. Ainsi, à chaque niveau d'expression d'un composant est associé un ensemble de régulations effectuées sur d'autres composants.

Les modélisations hybrides introduisent des aspects continus (temps, probabilité) qui gouvernent les transitions entre les états discrets du RBB. Cette introduction des dimensions continues permet de reproduire avec plus de précision les dynamiques des RRB. C'est avec cette motivation que Loïc Paulevé, Magnin & Roux (2011*a*) ont introduit les Frappes de Processus en y intégrant les paramètres stochastiques et temporels. Ces modélisations mettent en avant des aspects quantitatifs sur l'apparition de certains comportements et servent à apprécier l'importance de certains paramètres temporels ou stochastiques sur l'évolution du système. *Dans cette thèse, nous nous plaçons dans le cadre de la modéli-*

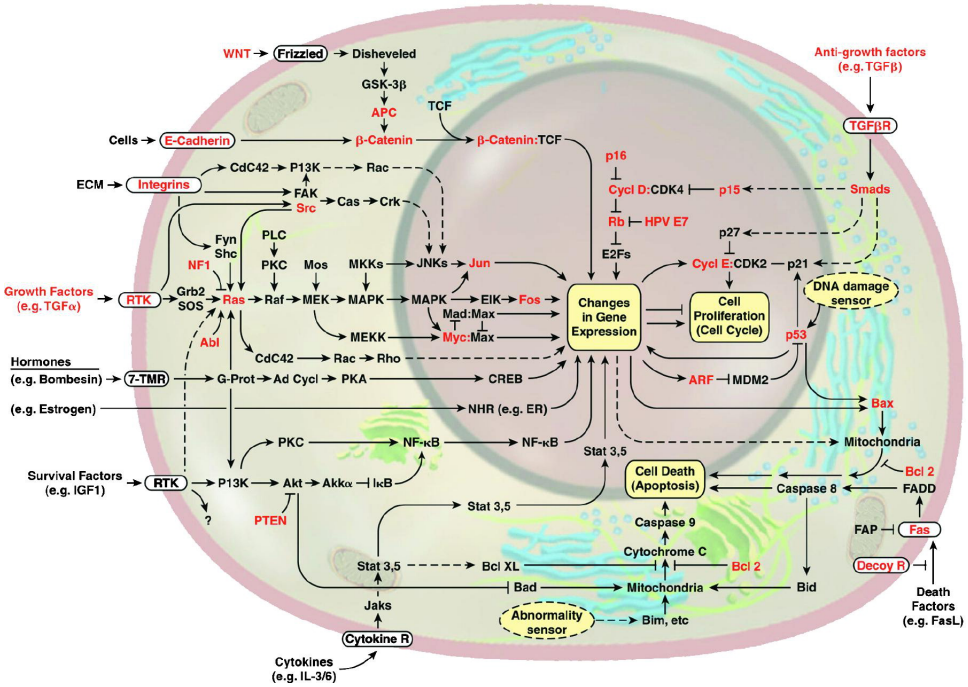



Figure 1.1 : Description du fonctionnement d'une cellule.
(Hanahan & Weinberg, 2000)

*sation hybride des systèmes biologiques. Les éléments biologiques sont modélisés comme des composants à états discrets et la dynamique est continue avec l'introduction des paramètres temporels et stochastiques.*

Définir les paramètres temporels et stochastiques n'est pas toujours facile, en particulier pour les très grands systèmes. Aussi, la production et la mise à disposition des données expérimentales, en particulier des séries temporelles, s'avère être une source très riche pour l'estimation des paramètres. Nous présentons dans la section suivante les données de séries temporelles.

## 1.3 Intégration des donnée de séries temporelles : un pas vers la modélisation chronométrique

Les technologies « omiques » ont révolutionné les recherches en génomique fonctionnelle en modifiant l'échelle des données analysables et la forme des protocoles de recherche scientifique. Elles permettent de générer des quantités massives de données à des niveaux biologiques multiples. Du séquençage des gènes à l'expression des protéines et des structures métaboliques, ces données peuvent couvrir une grande part des mécanismes impliqués dans les variations qui se produisent dans les réseaux cellulaires et qui influencent le fonctionnement des systèmes organiques dans leur ensemble.

En plus d'augmenter le débit et le nombre des données, les technologies « omiques » ont fondamentalement modifié les procédures de recherche. Ce qui permet une compréhension rapide des évolutions transcriptionnelles produites dans les cellules en réponse aux perturbations internes ou externes ou aux programmes de développement propres à la cellule. Ce potentiel ouvre de manière inédite la porte à de nombreuses découvertes sur les mécanismes des maladies, sur la compréhension des facteurs qui influencent l'efficacité

et la toxicité des médicaments, ou encore sur la manière dont notre organisme répond et réagit aux médicaments et à l'alimentation. Cependant, malgré l'énorme quantité de données produites, l'accès libre à ces données reste difficile. De plus, dans le processus de génération des données, il arrive que certaines d'entre elles se perdent. Ce qui peut conduire à la génération des données incomplètes. Enfin, la multiplicité des technologies « omiques » a pour effet de produire des données de différents types qui ne sont pas toujours adaptées selon l'étude qui est à mener.

*Dans cette thèse, nous nous intéressons aux séries temporelles en particulier*. En effet, elles décrivent l'évolution des quantités des composants biologiques dans le temps. Une analyse fine de cette dynamique peut permettre d'identifier les niveaux d'activité des composants en fonction du temps. Ce qui permet d'estimer les paramètres temporels (qui peuvent être des délais) et stochastiques pour les changements de niveaux discrets. Ce paramétrage des transitions entre niveaux discrets avec des valeurs continues (délais) permet de mettre en œuvre la notion de chronométrie. On peut aussi aller plus loin que la simple prise en compte de l'ordre des évènements (chronologie). Précisément, nous avons voulu introduire au moins partiellement cette amorce de prise en compte des aspects continus pour raffiner des modélisations purement discrètes (et donc éloignées de la réalité). Pour autant, nous avons cherché à maîtriser les dimensions de modèles afin de conserver la possibilité de les analyser.

## 1.4 Contributions

L'objectif de cette thèse est de modéliser, d'analyser et de vérifier les grands RRB à l'aide des réseaux d'automates, en prenant en compte la notion de *chronométrie* par l'intégration des paramètres quantitatifs (délais et aléatoire) estimés des données de séries temporels.

Cette thèse présente trois contributions principales :

- La modélisation de très grands RRB à l'aide des formalismes des Frappes de Processus et des réseaux d'automates stochastiques. La spécificité de notre modélisation est qu'elle permet un raffinement de la dynamique des actions (transitions) grâce à des paramètres temporels et stochastiques estimés des données de séries temporelles. Suite à la modélisation, une analyse statistique des comportements générés par les composants du modèle est faite grâce à l'analyse statistique des traces issues des simulations stochastiques.
- La vérification formelle des propriétés quantitatives (probabilité et délai) par l'analyse statique des réseaux d'automates stochastiques, c'est-à-dire par la construction des structures (Graphes de causalité quantifié) qui permettent les approximations des limites inférieures et supérieures des probabilités et des délais des propriétés d'accessibilité.
- L'analyse statique des « bifurcations » qui correspondent à des transitions telles qu'une fois ces transitions franchies, le système ne peut plus atteindre un état donné. Cette analyse statique s'appuie sur les approximations supérieures et inférieures des propriétés d'accessibilité. Elle est mise en œuvre grâce à une implémentation efficace en *Answer Set Programming*.

Les principales contributions précitées permettent de décliner les apports de cette thèse en cinq points : la formalisation des réseaux type RSTC (multi Layer Receptor Transcription

Celle State) ; l'intégration des données de séries temporelles ; la simulation et la validation par analyse des traces ; l'analyse statique à très grande échelle des propriétés quantitatives (probabilité et délai) d'accessibilité dans les réseaux d'automates stochastiques ; et l'analyse statique des bifurcations dans les réseaux d'automates asynchrones à grande échelle.

**Formalisation en Frappes de Processus et réseaux d'automates stochastiques des RRB type RSTC (multi Layer Receptor Transcription Celle State).**

Dans le but de formaliser les RRB type RSTC, spécifiés partiellement ou totalement, en vue de la construction de la dynamique généralisée et de la simulation stochastique des RRB, nous nous appuyons sur deux formalismes. Les Frappes de Processus dédiées à la modélisation de grands systèmes complexes et les réseaux d'automates stochastiques qui sont une généralisation des Frappes de Processus. Les Frappes de Processus (Paulevé, 2011) regroupent un ensemble fini de processus répartis dans des ensembles appelés des sortes. À tout instant, un et un seul processus de chaque sorte est actif ; un processus peut être remplacé par un autre processus de la même sorte par la *frappe* d'un autre processus présent. Pour les réseaux d'automates, on parle plutôt d'un ensemble fini d'états locaux répartis dans des ensembles appelés des automates. Contrairement aux Frappes de Processus, la présence de plusieurs états locaux d'automates distincts (pré-condition de frappes ou de transitions) permet le déclenchement d'une transition. C'est ce qui permet d'exprimer plus aisément les combinaisons logiques de condition de franchissement de transitions et de n'avoir pas à se préoccuper des délais nécessaires pour la mise à jour des sortes coopératives. Nous proposons une **détection automatique des motifs minimaux** (que nous définirons dans le chapitre 3) dans les RRB type RSTC que nous transformons en Frappes de Processus puis en réseaux d'automates asynchrones.

**Intégration des données de séries temporelles.**

Afin de prendre en compte la notion de chronométrie qui permet d'introduire une dimension temporelle dans le processus de modélisation des RRB, nous présentons une méthode qui permet d'**estimer les paramètres temporels et stochastiques des séries temporelles**. La méthode consiste à déterminer les seuils d'activité des composants pour lesquels nous avons les mesures de la dynamique à partir des séries temporelles. Puis, par une projection sur l'axe du temps, il est possible d'estimer les délais pour appliquer les actions dans les Frappes de Processus ou activer les transitions dans les réseaux d'automates. Le paramètre d'absorption de stochasticité est choisi en fonction de la variabilité que nous souhaitons pour chaque action ou transition.

Les séries temporelles que nous avons utilisé dans le cadre des applications de cette thèse ont été fournies par le Centre Allemand de Recherche contre le Cancer (DKFZ) [1].

**Simulation stochastique et validation des modèles par analyse statistique des traces.**

Dans le but de simuler et de valider les modèles générés dans les deux phases de contribution citées ci-dessus, nous avons utilisé le simulateur intégré dans le logiciel des Frappes de Processus Pint [2]. Ces simulations génèrent des traces qui reflètent la dynamique des composants. Du fait des interactions concurrentes, les traces peuvent être différentes d'une

[1] `http://www.dkfz.de/en/signal_transduction/`

[2] `http://process.hitting.free.fr`

simulation à l'autre. Aussi, nous avons choisi d'analyser un ensemble fini de traces issues des simulations. Nous avons pour cela défini les **traces acceptantes** en fonction de la dynamique attendue pour chaque composant. Puis, nous avons construit pour chacune des traces acceptantes un automate qui reconnaît les traces équivalentes aux traces acceptantes. Nous avons de cette façon analysé la cohérence d'un ensemble de simulations.

**Analyse statique à grande échelle des propriétés quantitatives (probabilité et délai) d'accessibilité dans les réseaux d'automates stochastiques.**

En partant des abstractions introduites dans le cadre de l'analyse statique des Frappes de Processus, permettant d'établir des approximations supérieures et des approximations inférieures des propriétés d'accessibilité successive de processus, nous définissons des abstractions quantitatives qui prennent en compte les fréquences de déclenchement des transitions. Ainsi, une propriété d'accessibilité donnée peut être récursivement et itérativement raffinée afin de construire ce que nous avons appelé *Graphe de Causalité Locale Quantifié (GCLQ)*. C'est un graphe qui est une variante du *Graphe de Causalité Locale (GLC)* introduit par (Paulevé, 2011) pour la vérification des propriétés d'accessibilité qualitatives. La spécificité du graphe de causalité locale quantifié par rapport au graphe de causalité locale est qu'il contient des quantités qui peuvent être soit des probabilités, soit des délais. Les probabilités et les délais sont calculés pendant un parcours du graphe de causalité locale quantifié où nous ignorons délibérément l'exploration des sous-graphes induits par les transitions concurrentes. Le résultat du calcul des probabilités et des délais est l'obtention d'une borne inférieure de la probabilité et/ou des délais d'accessibilité. La complexité théorique de cette analyse est limitée (polynomiale dans le nombre de composants et exponentielle dans le nombre d'états locaux pour chaque composant) : ceci promet son applicabilité à l'analyse de grands réseaux d'automates stochastiques et donc de grands RRB. Cette approche permet en plus d'extraire les nœuds et des chemins critiques (ayant des probabilités d'être traversé inférieures à un certain seuil fixé par un problème biologique spécifique) à la satisfaction de la propriété d'accessibilité donnée. Cette information peut **amener au contrôle de l'accessibilité dans le système réel** modélisé.

**Analyse statique à très grande échelle des bifurcations dans les réseaux d'automates asynchrones.**

Dans le but d'identifier les **transitions de bifurcations** (transitions après lesquelles l'accessibilité d'un état donné n'est plus possible), nous nous appuyons sur les approximations supérieures et inférieures des propriétés d'accessibilité dans les réseaux d'automates asynchrones et sur une implémentation efficace en ASP pour proposer une méthode d'identification des dites transitions. La méthode que nous avons conçue propose à la fois une sous- et sur-approximation des transitions de bifurcations. La méthode a été expérimentée sur les réseaux d'automates modélisant des RRB regroupant plus de cent composants. Le résultat sous la forme d'une liste de transitions de bifurcations est obtenu en quelques secondes. À notre connaissance, cette méthode est la première qui permette d'envisager un contrôle de la dynamique des grands RRB.

## 1.5 Organisation du manuscrit

La suite de ce manuscrit est organisée de la manière suivante :

Le chapitre 2 synthétise l'état de l'art d'une part sur les principales techniques de modélisation des RRB, d'autres part sur des techniques qui amènent une confrontation des modèles avec les données et enfin sur les principales analyses des propriétés (en particulier de la dynamique) développées sur modèles.

Le chapitre 3 propose une démarche qui construit des modèles formels (Frappes de Processus et réseaux d'automates stochastiques) en intégrant des données des séries temporelles. Il y est notamment présenté la détection des motifs, l'estimation des paramètres temporels et stochastiques et l'évaluation des modèles par l'analyse statistique des traces des simulations stochastiques. Ce travail a été effectué en collaboration avec le Centre Allemand de Recherche contre le Cancer (DKFZ) qui a fourni les données de séries temporelles qui ont guidé notre démarche pour l'estimation des paramètres.

Le chapitre 4 présente les analyses statiques efficaces par interprétation abstraite de la dynamique quantitative des réseaux d'automates stochastiques. Cette interprétation abstraite permet de proposer une version étendue du Graphe de Causalité Locale (GLC) en ajoutant des quantités pour obtenir le *Graphe de Causalité Locale Quantifié (GLCQ)*. Dans ce chapitre, nous présentons également comment du graphe de causalité locale quantifié, nous obtenons des bornes inférieures et supérieures des probabilités et des délais d'accessibilité. Enfin, nous montrons comment déterminer les nœuds et les chemins critiques dans le GLCQ pour une propriété d'accessibilité donnée.

Le chapitre 5 traite de l'identification des bifurcations par une combinaison de l'analyse statique et une implémentation efficace de la méthode développée par analyse statique en ASP. Nous présentons les approximations supérieures et inférieures des ensembles de transitions de bifurcation. Enfin, nous présentons la mise en œuvre en ASP des méthodes développées. Ce travail a été effectué avec la collaboration du Dr Loïc Paulevé.

Le chapitre 6 traite des différentes implémentations et la mise en applications effectuée au cours de cette thèse sur des systèmes biologiques réels.

Le chapitre 7 conclut cette thèse en discutant des contributions apportées et en donnant quelques perspectives principales.

## 1.6 Notations

**Nombres réels.** On note $\mathbb{R}$ l'ensemble des nombres réels. Si $i, j \in \mathbb{R}$, on note $[i\,;j] = \{x \in \mathbb{R} \mid i \leq x \leq j\}$ l'ensemble des nombres réels entre $i$ et $j$ compris.

**Entiers naturels.** On note $\mathbb{N}$ l'ensemble des entiers naturels, $\mathbb{N}^* = \mathbb{N} \setminus \{0\}$ l'ensemble des entiers naturels strictement positifs, et $\mathbb{N}^\bullet = \mathbb{N} \setminus \{0, 1\}$ l'ensemble des entiers naturels supérieurs ou égaux à 2. Si $i, j \in \mathbb{N}$, $i < j$, on note $[\![i\,;j]\!] = \{i, i+1, \ldots, j-1, j\}$ l'ensemble des entiers naturels entre $i$ et $j$ compris.

Pour tous entiers $i, j, k \in \mathbb{N}$, on note : $k < [\![i\,;j]\!] \overset{\Delta}{\Leftrightarrow} k < i$ et $k > [\![i\,;j]\!] \overset{\Delta}{\Leftrightarrow} k > j$. De plus, si $i_1, i_2, j_1, j_2 \in \mathbb{N}$, on note :

$$[\![i_1\,;j_1]\!] \leq_{[\![]} [\![i_2\,;j_2]\!] \overset{\Delta}{\Leftrightarrow} (i_1 \leq i_2 \wedge j_1 \leq j_2)$$

$$\text{et :} \qquad [\![i_1\,;j_1]\!] <_{[\![]} [\![i_2\,;j_2]\!] \overset{\Delta}{\Leftrightarrow} (i_1 < i_2 \wedge j_1 \leq j_2) \vee (i_1 \leq i_2 \wedge j_1 < j_2)$$

**Entiers relatifs.** On note $\mathbb{Z}$ l'ensemble des entiers relatifs.

Par ailleurs, la fonction signe est définie sur les entiers relatifs comme suit :

$$\text{signe} : \mathbb{Z} \to \{+, -, \varnothing\}$$
$$n \mapsto \begin{cases} + & \text{si } x > 0 \\ - & \text{si } x < 0 \\ \varnothing & \text{if } x = 0 \end{cases}$$

**Séquences.** Si $n \in \mathbb{N}$, on note $e_1 :: \ldots :: e_n$ la séquence finie formée des éléments $e_1, \ldots, e_n$ si $n \geq 1$, et on note $\varepsilon$ la séquence vide (si $n = 0$).

Pour toute séquence finie $E = e_1 :: \ldots :: e_n$, on note $|E| = n$ la longueur de cette séquence, et $\{1, \ldots, |E|\} = [\![1\,;|E|]\!]$ l'ensemble des indices de cette séquence. Pour tout $i \in \{1, \ldots, |E|\}$, on note $E_i = e_i$ le $i^{\text{e}}$ élément de $E$, et pour tout $i, j \in \{1, \ldots, |E|\}$, on note $E_{i..j} = e_i :: \ldots :: e_j$ la sous-séquence formée des éléments $i$ à $j$ de $E$ ; naturellement, $E_{i..j} = \varepsilon$ si $i > j$.

On note de plus : $e \in E \Leftrightarrow \exists i \in \{1, \ldots, |E|\}, e = E_i$

**Ensembles.** Le cardinal d'un ensemble $A$ est noté $|A|$ et son ensemble des parties est noté $\wp(A)$.

Si $A$ et $B$ sont deux ensembles, on note $A \cup B$ leur union, $A \cap B$ leur intersection et $A \times B$ leur produit cartésien.

Si $n \in \mathbb{N}$, et $\{A_i\}_{i \in [\![1;n]\!]}$ est un ensemble d'ensembles, on note $\bigcup_{i \in [\![1;n]\!]} A_i = A_1 \cup A_2 \cup \ldots \cup A_n$ leur union, $\bigcap_{i \in [\![1;n]\!]} A_i = A_1 \cap A_2 \cap \ldots \cap A_n$ leur intersection et $\bigotimes_{i \in [\![1;n]\!]} A_i = A_1 \times A_2 \times \ldots \times A_n$ leur produit cartésien. Cette définition peut naturellement être étendue à une séquence d'ensembles. De plus, par convention : $\bigcup_\emptyset = \bigcap_\emptyset = \bigotimes_\emptyset = \bigcup_\varepsilon = \bigcap_\varepsilon = \bigotimes_\varepsilon = \emptyset$.

Si $E = e_1 :: \ldots :: e_n$ est une séquence, on note $\widetilde{E} = \{e_1, \ldots, e_n\}$ l'ensemble correspondant. De même, si $T = (t_1, \ldots, t_n)$ est un $n$-uplet, on définit : $\widetilde{T} = \{t_1, \ldots, t_n\}$.

**Fonctions et plus petit point fixe.** Si $A$ et $B$ sont deux ensembles, on note $f : A \to B$ si $f$ est une fonction qui associe chaque élément de $A$ à un élément de $B$.

On note de plus $\mathbf{pppf}\{x_0\}\,(x \mapsto x')$ le plus petit point fixe plus grand que $x_0$ de la fonction $x \mapsto x'$, s'il existe.

**Recouvrement.** L'opérateur $\Cap$, qui désigne le recouvrement d'un état par un autre, est donné à la définition 2.11 en page 30. Il est ensuite étendu aux contextes à la définition 4.13 en page 88.

# Chapitre 2

# Modélisation et analyse des systèmes biologiques

Nous présentons dans ce chapitre un état de l'art lié à la modélisation, à l'analyse et à la vérification des systèmes biologiques. Cet état de l'art présente dans un premier temps les différentes abstractions couramment utilisées dans le processus de modélisation. Il y est donc présenté tour à tour le graphe des interactions, les modèles discrets et les modèles hybrides avec un accent particulier sur le formalisme des Frappes de Processus. Dans un second temps, nous présentons différentes techniques qui ont proposé une modélisation basée sur les données expérimentales. Le but étant soit de construire les modèles à partir des données, soit de confronter les deux (modèles et données) pour obtenir une validation des modèles ou une révision de ceux-ci. Enfin, les différentes méthodes d'analyse (statique) sont présentées : les méthodes par réduction des modèles, utilisation des diagrammes de décision, interprétation abstraite et vérification quantitatives.

## 2.1 Introduction

Une partie des systèmes biologiques peuvent être abstraits comme des Réseaux de Régulation Biologique (RRB) ceci en ne considérant que les effets régulateurs entre les composants. Nous parlons alors de régulation positive (resp. négative) quand la présence soutenue d'un composant $a$ participe à augmenter (resp. diminuer) la présence d'un composant $b$ ; $a$ est alors activateur (resp. inhibiteur) de $b$. Dans le but de comprendre les dynamiques de production de ces différents composants, plusieurs modélisations et formalismes ont été proposées. Dans le cadre de cette thèse, nous nous intéressons aux modélisations hybrides des RRB. Les modélisations hybrides considérées ici utilisent une modélisation discrète et asynchrone des RRB en intégrant une dimension continue ou quantitative pour la dynamique. La dimension continue dans ce contexte signifie rajouter des délais sous forme de temps continu aux changements d'états discrets du système.

Les modèles discrets permettent de représenter les RRB en abstrayant une partie de la dynamique contrairement aux équations différentielles qui représentent toute la dynamique. Cette abstraction permet d'avoir un modèle plus simple et donc plus facile à analyser, les systèmes d'équations différentielles étant souvent plus complexes pour permettre leur analyse. L'abstraction est obtenue en décomposant les différents niveaux d'expression d'un gène en intervalles pour obtenir les niveaux discrets. Les changements de niveaux se font suivant une dynamique asynchrone. L'aspect asynchrone des formalismes vient du fait qu'il est biologiquement très improbable que plusieurs entités d'un système évoluent en parfaite simultanéité. Le parallèle avec les systèmes d'équations différentielles est qu'il est rare d'observer plusieurs composants passer un seuil simultanément au cours d'une évolution continue de leur état. Ces hypothèses ont été théorisées par René Thomas (1973), qui en a dérivé le modèle qui porte son nom. À sa suite, plusieurs autres travaux ont permis d'enrichir ce modèle. Un bref aperçu de ces travaux et des résultats qui en découlent sera proposé en section 2.2. Les modélisations discrètes ont évolué vers les modèles hybrides par une prise en compte partielle des aspects continus. Cette prise en compte permet d'avoir des modèles discrets avec des possibilités de faire une évaluation semi-quantitative d'une partie de la dynamique. Nous présenterons rapidement quelques modélisations hybrides existantes avec une attention particulière aux Frappes de Processus introduite par Paulevé et al. (2011*a*).

En parallèle des développements théoriques et méthodologiques, l'évolution technologique et les expérimentations fournissent des données quantitatives de plus en plus abondantes et de meilleur qualité. Ces données rendent compte de l'évolution des quantités des composants des systèmes biologiques en environnement réel au travers des expérimentations. Dans un processus de modélisation classique, cette information est cruciale parce qu'elle permet de confronter l'idéalisation des modèles avec la réalité des données issues des expérimentations. Cette confrontation peut se faire de plusieurs façons. Soit en se servant des données pour construire des modèles, soit en se servant des données pour raffiner les modèles, soit enfin en se servant des données pour évaluer le comportement des modèles d'un point de vue de la dynamique. Nous présenterons et discuterons à la section 2.3 en page 40 les principales avancées dans ce champ de recherche très prometteur.

Confronter les modèles avec les données est une façon de les valider. Toutefois, il peut être, dans certaines situations (notamment des cas critiques comme par exemple éviter la mort cellulaire) utile d'apporter la preuve formelle de l'absence de comportement non souhaité dans les meilleurs des cas, d'évaluer les chances de voir ces comportements se produire ou pas. Aussi, de nombreuses approches de vérification ont été introduites pour la vérification des systèmes biologiques. Nous nous intéressons ici aux approches par analyse statique. Ces méthodes vont des analyses topologiques du graphe des interactions, permettant la dérivation de caractéristiques globales de la dynamique, aux méthodes de réduction de modèles, en passant par les manipulations algébriques sur les représentations symboliques. Dans cet état de l'art, nous présentons les méthodes de réduction des modèles, puis les opérations algébriques sur les digrammes de décision, les méthodes par interprétation abstraite et enfin les techniques introduites pour une vérification des propriétés quantitatives.

Le présent état de l'art est structuré comme suit. La section 2.2 présente un état de l'art des modélisations des réseaux de régulation biologique. Nous introduisons dans cette section le graphe des interactions à la section 2.2.1. Puis nous introduisons à la section 2.2.2 en page 22 la modélisation discrète, où nous présentons principalement le modèle de Thomas (1973). La section 2.2.3 en page 26 présente quelques travaux sur les

modélisations hybrides, nous mettrons un accent particulier sur le formalisme des Frappes de Processus. La section 2.3 en page 40 présente un état de l'art sur la prise en compte des données expérimentales dans le processus de modélisation. Dans un premier temps nous présentons à la section 2.3.1 en page 40 les travaux sur l'inférence des modèles à partir des données expérimentales. Puis à la section 2.3.2 en page 41 nous présentons les travaux sur la confrontation des modèles avec les données expérimentales. Enfin, nous présenterons à la section 2.4 en page 43 les techniques de vérification introduites pour la vérification des systèmes biologiques de grande taille.

## 2.2 État de l'art des modélisations des réseaux de régulation biologique

Nous exposons dans cette section les principales méthodes de modélisation des RRB. Ces méthodes vont des modélisations discrètes aux modélisations hybrides. Nous commençons par le graphe des interactions, qui nous permet d'introduire les modèles discrets en particulier le modèle de Thomas, puis nous introduisons d'autres modélisations hybrides introduites pour la modélisation des RRB. Nous finissons ce tour d'horizon par le formalisme des Frappes de Processus introduit par Paulevé (2011) et ses nombreux enrichissements apportés par Folschette (2014) dans sa thèse.

### 2.2.1 Graphe des Interactions

Le *graphe des interactions* d'un RRB offre une représentation simple et qualitative des régulations entre les composants. Dans cette représentation les composants sont représentés par des nœuds étiquetés par un nom (celui du composant : $a$, $b$, $c$, etc.) et les interactions par des arcs signés (ou orientés) positifs ou négatifs. Un arc signé positif (resp. négatif) de $a$ vers $b$ dénote que $a$ est un activateur (resp. inhibiteur) de $b$.

Selon le niveau de connaissance sur le système et les questions posées, le graphe d'interactions peut être agrémenté d'informations supplémentaires afin de mieux visualiser le rôle des régulations impliquées. Ainsi, on peut rajouter aux nœuds un plafond (son niveau d'expression maximum : $l_a$, $l_b$, $l_c$, etc.) et les arcs peuvent prendre la forme $a \xrightarrow{s,t} b$, c'est-à-dire étiquetés par un signe $s$ qui représente le type de régulation ($+$ pour une activation, $-$ pour une inhibition et $\circ$ pour une régulation plus complexe) et un entier $t$ qui représente le seuil de déclenchement de la réaction (c'est-à-dire le niveau d'expression du composant régulateur à partir duquel celui-ci a effectivement une influence sur le composant régulé). La notion de seuil (niveau) est présentée plus en détail à la section 2.2.2. Aussi la définition 2.1 propose une formalisation générale du *graphe des interactions*.

**Définition 2.1** (Graphe des interactions)**.** Un *graphe des interactions* est un couple $\mathcal{G} = (\Gamma\,;E)$ où $\Gamma$ est l'ensemble fini des *composants*, étiquetés par un nom et un *plafond*, et $E$ est l'ensemble fini des *régulations* entre deux nœuds, étiquetées par un *signe* et un *seuil* :

$$E \triangleq \{a \xrightarrow{s,t} b, \ldots \mid a, b \in \Gamma \wedge s \in \{+, -, \circ\} \wedge t \in [\![1\,; l_a]\!]\}$$

tel que chaque régulation de $a$ vers $b$ soit unique :

$$\forall a \xrightarrow{s,t} b \in E, \forall a \xrightarrow{s',t'} b \in E, s = s' \wedge t = t' \quad .$$

Étant donnée cette définition, on note $E_s \triangleq \{a \xrightarrow{s,t} b \in E\}$ pour $s \in \{+, -, \circ\}$. De plus, pour tout composant $b \in \Gamma$, on note $\mathcal{G}^{-1}(b)$ l'ensemble de ses *régulateurs* :

$$\mathcal{G}^{-1}(b) \triangleq \{a \in \Gamma \mid \exists a \xrightarrow{s,t} b \in E\}$$

La définition 2.1 propose une définition assez riche (signe, seuil) du graphe des interactions. Toutefois cette définition peut être dans certains cas plus simple en ommettant par exemple l'information sur les seuils, ou même dans des cas les signes. De nombreuses bases de données représentent les RRB sous forme des graphes d'interactions et elles utilisent pour cela un format (SMBL, SIF, BIOPAX, SBGN, KGML, etc.) qui est dans la plus part des cas interopérable pour permettre un plus grand partage des modèles. Pathway Interaction Database (PID) (Schaefer et al., 2009*a*) et Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa et al., 2015) sont des exemples des bases de données qui regroupent les RRB sous forme de graphes d'interactions. Ces bases de données ne contiennent pas les informations sur les seuils des interactions entre les composants. De nombreux exemples montrent que dans les RRB, de nombreuses régulations s'effectuent en coopération. Si nous supposons par exemple que $a$ et $b$ sont des activateurs de $c$, l'activation de $c$ n'est possible que si $a$ et à la fois $b$ sont présents. Cette coopération correspond pour certains systèmes biologiques à la formation des complexes. Les coopérations peuvent aussi s'effectuer entre les composants présents et les composants absents. De telles coopérations peuvent être spécifiées dans le graphe des interactions à travers l'introduction des multiplexes (Bernot, Comet & Khalis, 2008). L'introduction des multiplexes permet (1) une compréhension plus facile du système à la simple lecture du graphe des interactions et (2) un raffinement des dynamiques spécifiées et donc une réduction du champ de recherche des paramètres discrets permettant de spécifier complètement le comportement du système. Dans la suite nous allons nous intéresser aux modélisations discrètes.

*Exemple.* La figure 2.1(gauche) représente un graphe des interactions $(\Gamma\,; E)$ où $\Gamma = \{a, b, c\}$, avec $l_a = 2$ et $l_b = l_c = 1$, et :

$$E_+ = \{a \xrightarrow{+,2} a,\ a \xrightarrow{+,1} b\} \qquad E_\circ = \emptyset$$
$$E_- = \{b \xrightarrow{-,1} a\}$$

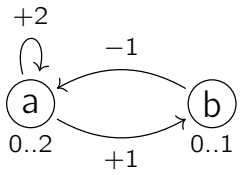
Ainsi :

$$\mathcal{G}^{-1}(a) = \{a, b\} \qquad \mathcal{G}^{-1}(b) = \{a\}$$

### 2.2.2 Modélisations Discrètes

Les modélisations discrètes affectent à chaque composant un ensemble fini et dénombrable de niveaux qualitatifs. Ces niveaux représentent souvent une concentration ce qui permet une abstraction d'une partie de la dynamique des composants. Elles ont été introduites dans un premier temps par Stuart A. Kauffman (1969) puis par René Thomas (1973) dans le cadre de formalismes booléens qui a ensuite étendu au multivalué.

L'activation (ou l'inhibition) effective de $b$ par $a$ dépend généralement du niveau de l'élément $a$ (par exemple le niveau de sa concentration). Tant qu'un certain seuil n'est pas




$$K_{a,\emptyset} = [\![0\,;0]\!] \qquad K_{b,\emptyset} = [\![0\,;0]\!]$$
$$K_{a,\{b\}} = [\![1\,;1]\!] \qquad K_{b,\{a\}} = [\![0\,;1]\!]$$
$$K_{a,\{a,b\}} = [\![2\,;2]\!]$$

Figure 2.1 : (gauche) Un exemple de graphe des interactions. Les composants sont représentés par les nœuds, comportant un nom et un ensemble de niveaux d'expression, tandis que les régulations sont représentées par des arcs étiquetés par un signe et un seuil. Par exemple, l'arc de $b$ vers $a$ étiqueté $-1$ représente la régulation $b \xrightarrow{-,1} a$. En d'autres termes, $b$ se comporte comme un inhibiteur de $a$ si son niveau d'expression est supérieur ou égal à 1, et se comporte comme un activateur sinon (c'est-à-dire si son niveau d'expression est égal à 0). (droite) Un exemple de paramétrisation du graphe des interactions de gauche. Nous définissons les paramètres $K_{a,\omega}$ à la définition 2.4.

atteint, la régulation peut être considérée comme ineffective ou inversée dans ce dernier cas $a$ devient alors un inhibiteur de $b$.

Aussi, pour tout composant $a$ régulant $b$, c'est-à-dire si $a \xrightarrow{s,t} b \in E$, on note $\mathsf{niveaux}(a \to b)$ (resp. $\overline{\mathsf{niveaux}}(a \to b)$) l'ensemble des niveaux d'expression de $a$ qui sont au-dessus (resp. en-dessous) du seuil $t$ (définition 2.2).

**Définition 2.2** (Niveaux effectifs ($\mathsf{niveaux}$))**.** Soit $\mathcal{G} = (\Gamma\,; E)$ un graphe des interactions. Si $a \xrightarrow{s,t} b \in E$, on définit :

$$\mathsf{niveaux}(a \to b) \triangleq [\![t\,; l_a]\!] \quad \text{et} \quad \overline{\mathsf{niveaux}}(a \to b) \triangleq [\![0\,; t-1]\!]$$

*Exemple.* Sur le graphe des interactions de la figure 2.1(gauche) on a notamment :

$$\mathsf{niveaux}(a \to b) = [\![2\,;2]\!] \qquad \overline{\mathsf{niveaux}}(a \to b) = [\![0\,;1]\!]$$

**Définition 2.3** (Ressources ($\mathsf{Res}$))**.** Soit $\mathcal{G} = (\Gamma\,; E)$ un graphe des interactions. Pour tout composant $a \in \Gamma$ et tout état $s \in \mathcal{S}$, on appelle *ressources de $a$ dans $s$* et on note $\mathsf{Res}_a(s)$ l'ensemble des régulateurs de $a$ dont le niveau dans $s$ est supérieur au seuil de la régulation qui les relie à $a$ :

$$\mathsf{Res}_a(s) \triangleq \{b \in \mathcal{G}^{-1}(a) \mid s[b] \in \mathsf{niveaux}(b \to a)\}$$

Un paramètre de René Thomas $K_{a,\omega}$ spécifie le niveau vers lequel évolue le composant $a$ soumis aux influences positives et négatives des composants dans $\omega$. En effet, un paramètre $K_{a,\omega}$ représente un ensemble de valeurs vers lesquelles le composant $a$ évolue dans tout état où l'ensemble de ses ressources est égal à $\omega$. Autrement dit, $a$ va évoluer vers la valeur de $K_{a,\omega}$ qui est la plus proche de son niveau d'expression courant. La définition 2.4 donne une formalisation du paramètre $K_{a,\omega}$.

**Définition 2.4** (Paramètre $K_{a,\omega}$ et paramétrisation $K$)**.** Soit $\mathcal{G} = (\Gamma\,; E)$ un graphe des interactions. Pour un composant $a \in \Gamma$ donné et $\omega \subset \mathcal{G}^{-1}(a)$ un sous-ensemble de ses régulateurs, le *paramètre* $K_{a,\omega} = [\![i\,;j]\!]$ est un intervalle non-vide tel que $0 \leq i \leq j \leq l_a$. La carte complète $K$ des paramètres sur un graphe des interactions $\mathcal{G}$ est appelée *paramétrisation de $\mathcal{G}$*.

Dans la suite, le couple formé d'un graphe des interactions et d'une paramétrisation est appelé modèle de *Thomas* et noté $(\mathcal{G}\,;K)$.

Au niveau de la dynamique, pour tout niveau d'expression de $a$ appartenant à $\mathsf{niveaux}(a \to b)$, $a$ est censé avoir une influence correspondant au signe $s$ sur $b$, c'est-à-dire être activateur si $s = +$, inhibiteur si $s = -$, ou avoir une influence indéterminée ou multiple si $s = \circ$ ; en revanche, pour tout niveau d'expression de $a$ appartenant à $\overline{\mathsf{niveaux}}(a \to b)$, l'influence opposée devrait être observée.

Cette hypothèse permet de modéliser la dégradation de $b$ en l'absence de l'activation de $a$ si $s = +$, ou l'activation de $b$ en l'absence de l'inhibition de $a$ si $s = -$.

### La dynamique du modèle de Thomas

La dynamique d'un modèle de *Thomas* $(\mathcal{G}\,;K)$ suit deux hypothèses formulées par *Thomas* lui-même.

- **Asynchrone** : un seul composant peut évoluer entre chaque état. En effet comme nous le disions en introduction, il est vraiment très peu probable que deux composants passent en même temps un seuil d'expression discret.
- **Unitaire** : chaque composant ne peut évoluer que d'un niveau discret à la fois.

Ces deux hypothèses permettent de conserver un certain nombre de propriétés propres aux systèmes d'équations différentielles dans lesquels chaque composant évolue de façon continue.

Aussi, la dynamique du modèle de *Thomas* avec des paramètres discrets est définie comme suit : il existe une transition d'un état $s$ vers un état $s'$ si et seulement si il existe un unique composant $a$ qui évolue entre ces deux états, d'exactement un niveau d'expression et vers le paramètre $K_{a,\mathsf{Res}_a(s)}$ (définition 2.5). Il faut cependant noter que $a$ ne peut pas évoluer si son niveau d'expression dans l'état $s$ appartient déjà à l'intervalle du paramètre $K_{a,\mathsf{Res}_a(s)}$.

**Définition 2.5** (Dynamique unitaire d'un modèle de Thomas $(\to)$)**.** Pour tout modèle de Thomas $\mathsf{T} = (\mathcal{G}\,;K)$, La dynamique de $\mathsf{T}$ est donnée par la relation de transition $\to \in \mathcal{S} \times \mathcal{S}$ définie par :

$$\forall s, s' \in \mathcal{S}, s \to s' \iff \exists a \in \Gamma, s[a] \notin K_{a,\mathsf{Res}_a(s)} \wedge s'[a] = s[a] + \delta^a(s)$$
$$\wedge\, \forall b \in \Gamma, b \neq a \Rightarrow s[b] = s'[b]$$

$$\text{avec : } \delta^a(s) = \begin{cases} +1 & \text{si } s[a] < K_{a,\mathsf{Res}_a(s)} \\ -1 & \text{si } s[a] > K_{a,\mathsf{Res}_a(s)} \end{cases}$$

Les symboles « $<$ » et « $>$ » de cette définition permettant de comparer un entier à un intervalle sont définis à la section 1.6 en page 16.

*Exemple.* La figure 2.1(droite) donne un exemple de paramétrisation du graphe des interactions de la figure 2.1(gauche), ce qui en fait un modèle de Thomas complet, dont la figure 2.2 donne l'espace des états. On note notamment la présence d'un état stable pour ce modèle, c'est-à-dire un état depuis lequel plus aucune évolution n'est possible : $\langle a_2, b_1 \rangle$, où $x_i$ représente le niveau d'expression $i$ pour le composant $x$.

On peut observer l'aspect unitaire de la dynamique d'un modèle de *Thomas* sur ce graphe. En effet, malgré le paramètre $K_{a,\{a,b\}} = [\![2\,;2]\!]$, le composant $a$ ne peut pas

directement passer de l'état $a_0$ à l'état $a_2$ en « sautant » l'état $a_1$. C'est pourquoi on observe les transitions $\langle a_0, b_0\rangle \to \langle a_1, b_0\rangle$ et $\langle a_1, b_0\rangle \to \langle a_2, b_0\rangle$.

Nous notons enfin que les paramètres sous forme d'intervalles permettent notamment de rendre un composant immobile. C'est par exemple le cas du paramètre $K_{b,\emptyset} = [\![0\,;1]\!]$, qui est pris en compte lorsque $a$ n'est pas au niveau 2. Dans une sémantique ne permettant que des paramètres unitaires, une auto-régulation de $b$ serait nécessaire.

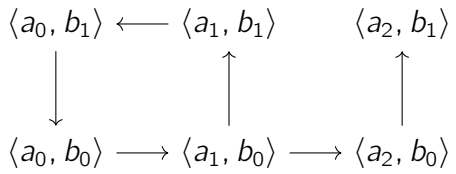



Figure 2.2 : Représentation de la dynamique du modèle de Thomas donné à la figure 2.1. Chaque état est représenté par un couple $\langle a_i, b_j\rangle$ où $i$ et $j$ représentent respectivement le niveau d'expression de $a$ et $b$ et une transition entre deux états est représentée par une flèche.

### La dynamique des réseaux discrets

Certaines contraintes propres aux modèles de Thomas peuvent être relâchées pour permettre des comportements supplémentaires. Ainsi, il est courant de représenter la dynamique des réseaux de régulation biologique sous la forme de *réseaux discrets asynchrones*. Ces réseaux sont aussi fondés sur un graphe des interactions, mais ils utilisent des fonctions d'évolution (définition 2.6) pour plus de permissivité, en lieu et place de paramètres discrets tels que précédemment formalisés à la définition 2.4 en page 23. Par ailleurs, l'hypothèse d'asynchronisme est conservée car un seul composant peut évoluer depuis chaque état, mais leur dynamique n'est pas unitaire dans le cas général car ce composant peut évoluer d'un nombre arbitraire de niveaux d'expression (définition 2.7).

**Définition 2.6** (Réseau discret asynchrone (RDA))**.** Si $\mathcal{G} = (\Gamma\,; E)$ est un graphe des interactions, un *réseau discret asynchrone* est un couple RDA $= (\mathcal{G}\,; F)$ avec $F = (f_x)_{x\in\Gamma}$, tels que $\forall x \in \Gamma, f_x : \mathcal{G}^{-1}(x) \to [\![0\,; l_x]\!]$.

**Définition 2.7** (Dynamique d'un réseau discret asynchrone ($\to_{\text{RDA}}$))**.** Pour tout réseau discret asynchrone RDA $= (\mathcal{G}\,; F)$, La dynamique non-unitaire de RDA est donnée par la relation de transition $\to_{\text{RDA}} \in \mathcal{S} \times \mathcal{S}$ définie par :

$$\forall s, s' \in \mathcal{S}, s \to_{\text{RDA}} s' \iff \exists a \in \Gamma, s[a] \neq f_a(s) \wedge s'[a] = f_a(s) \\ \wedge \forall b \in \Gamma, b \neq a \Rightarrow s[b] = s'[b]$$

Enfin, nous désignons par *réseau booléen asynchrone* tout réseau discret asynchrone dont les composants ne possèdent que deux niveaux discrets, c'est-à-dire tel que : $\forall x \in \Gamma, l_x = 1$.

### 2.2.3 Modélisations Hybrides

Les modèles peuvent être enrichis notamment en rajoutant une composante continue qui gouverne les transitions entre les états discrets, il devient donc possible d'explorer les propriétés quantitatives (probabilité d'observer un comportement, temps moyen, etc.). Les modélisations hybrides apportent cette richesse pour la compréhension des RRB qui sont des systèmes intrinsèquement aléatoires, et où la notion de délai peut jouer un rôle dans la caractérisation d'un comportement donné. Très souvent trois types de modélisations hybrides sont rencontrées : les modélisations ajoutant une composante stochastique, les modélisations ajoutant strictement une composante temporelle, et celles combinant les deux précédentes.

#### 2.2.3.1 Modèles Stochastiques

Dans le cadre des modélisations stochastiques, il est courant d'associer aux transitions des délais qui suivent généralement une distribution exponentielle. Cette distribution exponentielle permet de donner au système la propriété Markovienne. Ainsi dans le cadre des modélisations Markoviennes, nous pouvons citer de façon non exhaustive l'utilisation de Réseaux de Petri stochastiques (Heiner, Gilbert & Donaldson, 2008), du $\pi$-calcul stochastique (Maurin, Magnin & Roux, 2009), de $\kappa$ (Danos, Feret, Fontana & Krivine, 2007) ou encore de Biocham (Rizk, Batt, Fages & Soliman, 2008) pour la modélisation et l'analyse des systèmes biologiques. Le but principal des modélisations stochastiques est de permettre le calcul des probabilités d'observation de certains comportements.

#### 2.2.3.2 Modèles Temporels

Les modèles temporels se focalisent plutôt sur les délais pris généralement dans un intervalle de temps fixé ou suivant une certaine équation différentielle. Nous pouvons ainsi citer l'utilisation de Réseaux de Petri temporisés (Popova-Zeugmann, Heiner & Koch, 2005), d'automates temporisés (Siebert & Bockmayr, 2006), d'automates hybrides linéaires (Ahmad, Roux, Bernot, Comet & Richard, 2008) et d'automates hybrides non linéaires (Alur, Belta, Kumar, Mintz, Pappas, Rubin & Schug, 2002) pour la modélisation et l'analyse des systèmes biologiques. Les modélisations temporelles permettent un raffinement sur les dynamiques discrètes initiales. En effet, plusieurs possibilités émergent du fait de la modélisation temporelle : (1) du fait de la contrainte temporelle, il peut être désormais possible d'observer des comportements initialement interdits dans la dynamique discrète et de même, la contrainte temporelle peut aussi interdire certaines actions ; (2) la contrainte temporelle permet d'avoir des comportements plus précis sur la dynamique.

La section 2.2.3.3 présente le formalisme des Frappes de Processus qui est un formalisme introduit pour la modélisation des systèmes biologiques. Il présente l'avantage de permettre une modélisation hybride des systèmes. Ceci par une modélisation des composants comme des composants à états discrets et une dynamique continue qui permet à la fois la prise en compte du temps sous forme continue avec un comportement aléatoire.

#### 2.2.3.3 Les Frappes de Processus

Nous présentons ici les Frappes de Processus (standards) telles que introduites dans la thèse de Loïc Paulevé (2011) pour la modélisation de systèmes concurrents. C'est un formalisme qui se veut simple et qui peut être considéré comme une restriction de nombreux autres

formalismes existant avant son introduction. La principale motivation pour son introduction est basée sur l'intuition que sa simplicité engendre des modèles possédant une structure à partir de laquelle la dynamique sous-jacente peut être aisément comprise. C'est de fait un formalisme bien adapté pour la modélisation des RRB.

Les Frappes de Processus regroupent un ensemble fini de processus, divisés en sortes : un processus appartient à une et une seule sorte. À tout instant, un et un seul processus de chaque sorte est actif, indiquant l'état courant de la sorte à laquelle il appartient. Le changement de processus actif dans une sorte se fait à partir de la frappe du processus actif par au moins un autre processus courant. Un processus est noté $a_i$ où $a$ est la sorte et $i$ l'identifiant du processus au sein de la sorte $a$.

Les interactions concurrentes entres les processus sont définies par un ensemble d'*actions*. Ces actions permettent le remplacement d'un processus par un autre de la même sorte, conditionné par la présence d'au plus un autre processus de l'état courant des Frappes de Processus.

La définition 2.8 formalise les Frappes de Processus standards telles que introduites par Loïc Paulevé (2011).

**Définition 2.8** (Frappes de Processus standards)**.** Les *Frappes de Processus standards* sont définies par un triplet $\mathcal{PH} = (\Sigma\,;\mathcal{L}\,;\mathcal{H})$, où :

- $\Sigma \triangleq \{a, b, \ldots\}$ est l'ensemble fini et dénombrable des *sortes* ;
- $\mathcal{L} \triangleq \bigotimes_{a\in\Sigma} \mathcal{L}_a$ est l'ensemble fini des *états*, où $\mathcal{L}_a = \{a_0, \ldots, a_{l_a}\}$ est l'ensemble fini et dénombrable des *processus* de la sorte $a \in \Sigma$ et $l_a \in \mathbb{N}^*$, chaque processus appartenant à une unique sorte : $\forall (a_i\,;b_j) \in \mathcal{L}_a \times \mathcal{L}_b, a \neq b \Rightarrow a_i \neq b_j$ ;
- $\mathcal{H} \triangleq \{a_i \to b_j \Rsh b_k \mid (a\,;b) \in \Sigma\times\Sigma \wedge (a_i\,;b_j\,;b_k) \in \mathcal{L}_a \times \mathcal{L}_b \times \mathcal{L}_b \wedge b_j \neq b_k \wedge a = b \Rightarrow a_i = b_j\}$ est l'ensemble fini des *actions*.

On note $\mathbf{Proc} \triangleq \bigcup_{a\in\Sigma} \mathcal{L}_a$ l'ensemble de tous les processus. La sorte d'un processus $a_i$ est donnée par $\mathsf{sorte}(a_i) = a$ ; on définit aussi l'ensemble des sortes d'une action ou d'un ensemble de processus par :

$$\forall h \in \mathcal{H}, \mathsf{sortes}(h) = \{\mathsf{sorte}(\mathsf{frappeur}(h)), \mathsf{sorte}(\mathsf{cible}(h))\}$$
$$\forall A \subset \mathbf{Proc}, \mathsf{sortes}(A) = \{\mathsf{sorte}(p) \mid p \in A\}$$

Étant donné un état $s \in \mathcal{L}$, le processus de la sorte $a \in \Sigma$ présent dans $s$ est donné par $s[a]$, c'est-à-dire la coordonnée correspondant à $a$ dans l'état $s$. Si $a_i \in \mathcal{L}_a$, nous définissons la notation : $a_i \in s \overset{\Delta}{\Leftrightarrow} s[a] = a_i$ ; par extension, si $ps \subset \mathbf{Proc}$, on écrit alors : $ps \subseteq s \overset{\Delta}{\Leftrightarrow} \forall p \in ps, p \in s$. Pour toute action $h = a_i \to b_j \Rsh b_k \in \mathcal{H}$, $a_i$ est appelé le *frappeur*, $b_j$ la *cible* et $b_k$ le *bond* de $h$, et on note : $\mathsf{frappeur}(h) = a_i$, $\mathsf{cible}(h) = b_j$ et $\mathsf{bond}(h) = b_k$.

*Exemple.* La figure 2.3 illustre une représentation possible des Frappes de Processus standards. Le modèle $\mathcal{PH} = (\Sigma, \mathcal{L}, \mathcal{H})$ représenté comporte trois sortes : $\Sigma = \{a, c, f\}$. Chaque sorte comporte exactement deux processus :

$$\mathcal{L}_a = \{a_0, a_1\} \quad ; \quad \mathcal{L}_c = \{c_0, c_1\} \quad ; \quad \mathcal{L}_f = \{f_0, f_1\} \quad .$$

On peut notamment en déduire le nombre total d'états du système : $|\mathcal{L}| = |\mathcal{L}_a|\cdot|\mathcal{L}_c|\cdot|\mathcal{L}_f| = 2^3 = 8$. Cette grandeur n'est cependant donnée qu'à titre indicatif, car nous évitons de

construire explicitement l'espace des états dont la taille est exponentielle dans le nombre de sortes et de processus du modèle. Enfin, le modèle étudié comporte 7 actions :

$$\begin{aligned}\mathcal{H} = \{ \quad & c_1 \to a_1 \Rsh a_0 \quad , & & c_0 \to a_0 \Rsh a_1 \quad , \\ & c_1 \to c_1 \Rsh c_0 \quad , & & f_1 \to c_0 \Rsh c_1 \quad , \\ & f_1 \to a_0 \Rsh a_1 \quad , & & f_0 \to c_1 \Rsh c_0 \quad , \\ & f_1 \to f_1 \Rsh f_0 & & \}\end{aligned}$$

Dans cet exemple, les Frappes de Processus représentent un modèle simplifié du mécanisme de segmentation des métazoaires qui permet par exemple de décrire la production de rayures chez les drosophiles. Il a été originellement établi *in silico* par François, Hakim & Siggia (2007) à l'aide d'un formalisme à base d'équations différentielles, et le modèle présenté ici est tiré de la thèse de Folschette (2014) qui s'est inspiré du modèle proposé par Paulevé et al. (2011*a*).

Les trois sortes $a$, $c$ et $f$ de ce modèle représentent différents gènes du système, que nous qualifions d'*actifs* lorsqu'ils sont à l'état 1. La production de pigment est déclenchée par le produit du gène $a$, et une succession d'activations et de désactivations de celui-ci permet donc de produire des rayures. Pour que celles-ci soient régulières, il est donc nécessaire que la durée d'activation de $a$ soit constante, et que la durée entre deux activations le soit aussi. Ce mécanisme est réglé par le gène $c$ qui inhibe à la fois le gène $a$ à intervalles réguliers, et s'inhibe lui-même afin d'avoir le rôle d'une horloge. Enfin, le procédé complet est dirigé par le gène $f$ qui, lorsqu'il est actif, permet la progression d'un front au niveau duquel les pigments sont déposés. Ce gène peut aussi s'auto-inhiber après une certaine période, faisant cesser l'oscillation de l'horloge et ainsi la production de rayures.

Les séquences d'actions permettent d'abstraire une dynamique locale en se concentrant sur la conséquence plutôt que sur la cause. Ce qui leurs confère une importance particulière. C'est ce que nous verrons plus loin dans le développement des méthodes d'analyse statique chapitre 4 et chapitre 5.

Pour toute séquence d'actions $A$, on note $\mathsf{sortes}(A)$ l'ensemble des sortes dont au moins un processus figure dans $A$ en tant que frappeur, cible ou bond d'une action. De plus, pour toute sorte $a$, on note $\mathsf{premier}_a(A)$ le premier processus de $a$ référencé dans $A$, en tant que frappeur ou en tant que cible, et $\mathsf{dernier}_a(A)$ le dernier, en tant que frappeur ou en tant que bond. On note en conséquence $\sup A$ l'ensemble de tous ces premiers processus, et $\mathsf{fin}(A)$ l'ensemble de tous ces derniers processus.

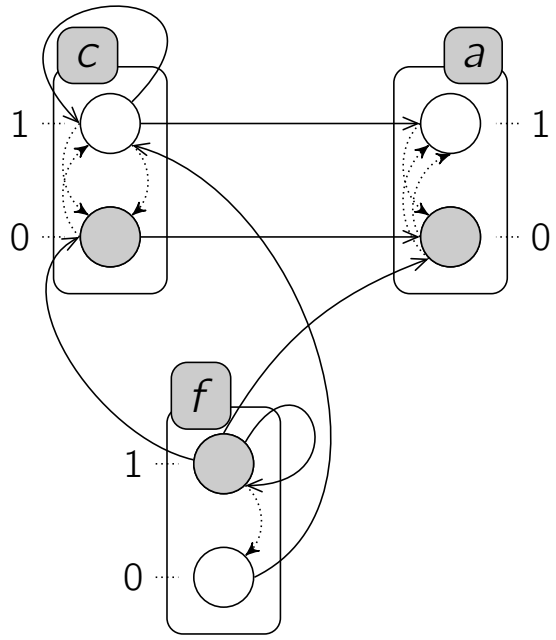



Figure 2.3 : Un exemple de Frappes de Processus standards. Les sortes sont représentées par des rectangles arrondis contenant des cercles représentant les processus. Ainsi, le processus $a_1$ est représenté par le cercle marqué « 1 » dans le rectangle étiqueté « $a$ », etc. Chaque action est de plus symbolisée par un couple de flèches, l'une en trait plein et l'autre en pointillés ; par exemple, l'action $c_1 \to a_1 \nearrow a_0$ est représentée par une flèche pleine entre les processus $c_1$ et $a_1$ suivie d'une flèche en pointillés entre $a_1$ et $a_0$. Enfin, les processus grisés représentent un état courant possible pour ces Frappes de Processus : $\langle a_0, c_0, f_1 \rangle$, ce peut être par exemple l'état initial pour ce modèle.

**Définition 2.9** (premier, dernier, support et fin)**.**

$$\mathsf{premier}_a(A) = \begin{cases} \varnothing & \text{si } a \notin \mathsf{sortes}(A) \\ \mathsf{frappeur}(A_m) & \text{si } m = \min\{n \in \{1,\dots,|A|\} \mid a \in \mathsf{sortes}(A_n)\} \\ & \qquad \wedge\ \mathsf{sorte}(\mathsf{frappeur}(A_m)) = a \\ \mathsf{cible}(A_m) & \text{sinon si } m = \min\{n \in \{1,\dots,|A|\} \mid a \in \mathsf{sortes}(A_n)\} \\ & \qquad \wedge\ \mathsf{sorte}(\mathsf{cible}(A_m)) = a \end{cases}$$

$$\mathsf{dernier}_a(A) = \begin{cases} \varnothing & \text{si } a \notin \mathsf{sortes}(A) \\ \mathsf{bond}(A_m) & \text{si } m = \max\{n \in \{1,\dots,|A|\} \mid a \in \mathsf{sortes}(A_n)\} \\ & \qquad \wedge\ \mathsf{sorte}(\mathsf{bond}(A_m)) = a \\ \mathsf{frappeur}(A_m) & \text{sinon si } m = \max\{n \in \{1,\dots,|A|\} \mid a \in \mathsf{sortes}(A_n)\} \\ & \qquad \wedge\ \mathsf{sorte}(\mathsf{frappeur}(A_m)) = a \end{cases}$$

$$\mathsf{support}(A) = \{p \in \mathbf{Proc} \mid \mathsf{sorte}(p) \in \mathsf{sortes}(A) \wedge p = \mathsf{premier}_{\mathsf{sorte}(p)}(\delta)\}$$
$$\mathsf{fin}(A) = \{p \in \mathbf{Proc} \mid \mathsf{sorte}(p) \in \mathsf{sortes}(A) \wedge p = \mathsf{dernier}_{\mathsf{sorte}(p)}(\delta)\}$$

La définition 2.10 établit la notion de sous-état sur un ensemble de sortes, c'est-à-dire

un ensemble de processus qui sont deux à deux de sortes différentes, ce qui permet de ne considérer qu'une partie d'un état complet. L'ensemble de tous les sous-états est noté $\mathcal{L}^\Diamond$ et nous constatons qu'un état est *a fortiori* un sous-état : $\mathcal{L} \subset \mathcal{L}^\Diamond$. De plus nous notons $\mathbf{Proc}^\Diamond$ l'ensemble des sous-états désordonnés, c'est-à-dire dont l'ordre entre les sortes a été oublié.

Le recouvrement d'un état $s$ par un processus $a_i$ est formalisé à la définition 2.11 par un état identique à $s$, sauf pour le processus de $a$ qui a été remplacé par $a_i$, ce qui permettra de définir la dynamique des Frappes de Processus à la définition 2.14. La définition de recouvrement est aussi étendue à un sous-état désordonné, autrement dit, à un ensemble de processus contenant au plus un processus par sorte.

**Définition 2.10** (Sous-états ($\mathcal{L}^\Diamond$))**.** Si $S \subset \Sigma$ est un ensemble de sortes, un sous-état sur $S$ est un élément de : $\mathcal{L}_S^\Diamond \triangleq \bigotimes_{a \in S} \mathcal{L}_a$. L'ensemble de tous les sous-états est noté : $\mathcal{L}^\Diamond \triangleq \bigcup_{S \in 2^{(\Sigma)}} \mathcal{L}_S^\Diamond$.
De plus, si $\sigma \in \mathcal{L}^\Diamond$ et $s \in \mathcal{L}$, on note alors :

$$\sigma \subseteq s \triangleq \forall a_i \in \mathbf{Proc}, a_i \in \sigma \Rightarrow a_i \in s \ .$$

Enfin, si $S \subset \Sigma$, on note : $\mathbf{Proc}_S^\Diamond = \{\widetilde{ps} \subset \mathbf{Proc} \mid ps \in \mathcal{L}_S^\Diamond\}$ et $\mathbf{Proc}^\Diamond = \{\widetilde{ps} \subset \mathbf{Proc} \mid ps \in \mathcal{L}^\Diamond\}$.

**Définition 2.11** (Recouvrement ($\Cap : \mathcal{L} \times \mathbf{Proc} \to \mathcal{L}$))**.** Étant donné un état $s \in \mathcal{L}$ et un processus $a_i \in \mathbf{Proc}$, $(s \Cap a_i)$ est l'état défini par : $(s \Cap a_i)[a] = a_i \wedge \forall b \neq a, (s \Cap a_i)[b] = s[b]$. On étend de plus cette définition à un ensemble de processus par le recouvrement de l'état par chaque processus, à condition que les processus de l'ensemble soient tous de sortes différentes : $\forall ps \in \mathbf{Proc}^\Diamond, s \Cap ps = s \underset{a_i \in ps}{\Cap} a_i$.

Nous présentons dans la suite les éléments nécessaires pour la dynamique en complément des actions dans les Frappes de Processus. Le premier élément va être la propriété de jouabilité introduit à la définition 2.12. Cette propriété permet de décrire la présence d'une configuration de processus actifs dans un état donné, ce qui permet de décrire la « jouabilité » d'une action. Aussi la définition 2.13 permet de définir l'*opérateur de jouabilité* des Frappes de Processus standards. Enfin la dynamique proprement dite est donnée à la définition 2.14. Elle est construite à partir de l'opérateur de jouabilité.

**Définition 2.12** (Propriété de jouabilité ($F$))**.** Une *propriété de jouabilité* est un élément du langage $F$ défini inductivement par :

- $\top$ et $\bot$ appartiennent à $F$ ;
- si $a \in \Sigma$ et $a_i \in \mathcal{L}_a$, alors $a_i$ appartient à $F$ et est appelé un *atome* ;
- si $P \in F$ et $Q \in F$, alors $\neg P$, $P \wedge Q$ et $P \vee Q$ appartiennent à $F$.

Si $P \in F$ est une propriété de jouabilité et $\sigma \in \mathcal{L}^{\Diamond}$ est un sous-état, on note $[P](\sigma)$ l'*évaluation* de $P$ dans $\sigma$ :

- si $P = a_i \in \mathcal{L}_a$ est un atome, avec $a \in \Sigma$, alors $[a_i](\sigma)$ est vraie si et seulement si $a_i \in \sigma$ ;
- si $P$ n'est pas un atome, alors $[P](\sigma)$ est vraie si et seulement si on peut l'évaluer récursivement comme vraie en utilisant la sémantique habituelle des opérateurs $\neg$, $\wedge$ et $\vee$ et des constantes $\top$ et $\bot$.

Une fonction $F : \mathcal{H} \to F$ associant à toute action une propriété de jouabilité est appelée un *opérateur de jouabilité*.

Étant donné que ce langage n'utilise que des opérateurs logiques classiques, les propriétés de la logique booléenne sont applicables aux propriétés de jouabilité, à savoir celles concernant la distributivité, l'associativité et la commutativité, ainsi que les lois de De Morgan concernant la négation.

Il en résulte notamment la propriété suivante, permettant d'évaluer la négation d'un atome, et qui dérive naturellement du fait que si un processus n'est pas actif dans un état donné, cela signifie alors qu'un autre processus de la même sorte l'est :

$$\forall a \in \Sigma, \forall a_i \in \mathcal{L}_a, \forall \sigma \in \mathcal{L}^{\Diamond}, [\neg a_i](\sigma) \Leftrightarrow \left[ \bigvee_{\substack{a_j \in \mathcal{L}_a \\ a_j \neq a_i}} a_j \right](\sigma)$$

L'opérateur de jouabilité $F$ donné à la définition 2.13 est propre aux Frappes de Processus standards. En revanche, la dynamique donnée à la définition 2.14 est générale à toutes les Frappes de Processus, et peut donc à fortiori être utilisée avec l'opérateur $F$ des Frappes de Processus standards.

**Définition 2.13** (Opérateur de jouabilité ($F : \mathcal{H} \to F$))**.** L'opérateur de jouabilité des Frappes de Processus est défini par :

$$\forall h \in \mathcal{H}, F(h) \equiv \mathsf{frappeur}(h) \wedge \mathsf{cible}(h) \ .$$

**Définition 2.14** (Dynamique des Frappes de Processus ($\to_{\mathcal{PH}}$))**.** Une action $h \in \mathcal{H}$ est dite *jouable* dans l'état $s \in \mathcal{L}$ si et seulement si : $[F(h)](s)$. Dans ce cas, $(s \cdot h)$ est l'état résultant du jeu de l'action $h$ dans $s$, et on le définit par : $(s \cdot h) = s \Cap \mathsf{bond}(h)$. De plus, on note alors : $s \to_{\mathcal{PH}} (s \cdot h)$.
Si $s \in \mathcal{L}$, un *scénario* $\delta$ dans $s$ est une séquence d'actions de $\mathcal{H}$ qui peuvent être jouées successivement depuis $s$. L'ensemble de tous les scénarios dans $s$ est noté $\mathbf{Sce}(s)$.

*Remarque.* Les Frappes de Processus standards possèdent une dynamique totalement asynchrone : entre deux états, une unique action peut être jouée. Ce choix de conception est largement inspiré du modèle de Thomas présenté à la section 2.2.2 en page 22, dont la

dynamique est aussi totalement asynchrone. Biologiquement, une telle hypothèse est cohérente avec le fait que la probabilité que deux composants franchissent simultanément un seuil d'expression est très faible. Cependant, cette sémantique asynchrone permet aussi de simplifier la dynamique des Frappes de Processus standards en assurant que deux états successifs ne varient que d'un seul processus (en fait : exactement d'un processus). Cela a notamment permis le développement de l'analyse statique évoquée plus loin, à la section 2.4.3.

*Exemple.* La dynamique des Frappes de Processus standards de la figure 2.3 est représentée à la figure 2.4. On peut notamment y observer le comportement stationnaire normal du modèle qui consiste en une oscillation alternée des processus actifs des sortes $a$ et $c$ :

$$\langle a_0, c_0, f_1\rangle \to_{\mathcal{PH}} \langle a_1, c_0, f_1\rangle \to_{\mathcal{PH}} \langle a_1, c_1, f_1\rangle \to_{\mathcal{PH}} \langle a_0, c_1, f_1\rangle \to_{\mathcal{PH}} \langle a_0, c_0, f_1\rangle \to_{\mathcal{PH}} \dots$$

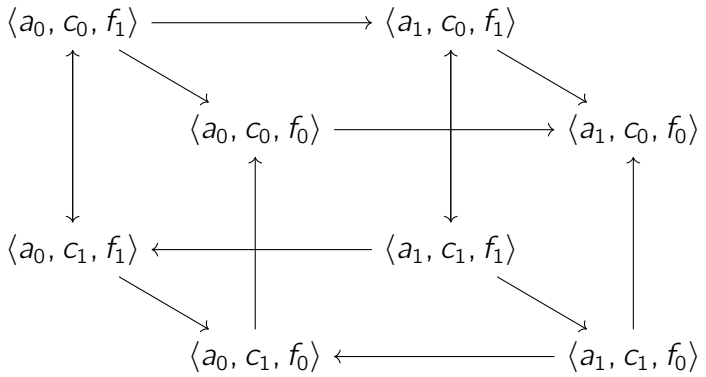



Figure 2.4 : Représentation de la dynamique du modèle de Frappes de Processus standards de la figure 2.3. Chaque état est représenté par un triplet $\langle a_i, c_j, f_k\rangle$ où $i$, $j$ et $k$ représentent respectivement le niveau d'expression de $a$, $c$ et $f$. Une transition entre deux états est représentée par une flèche.

### Utilisation des sortes coopératives

L'une des questions qui se posent en présence d'un formalisme totalement asynchrone comme les Frappes de Processus standards est la représentation des coopérations entre les différents composants. En effet, le bond d'un processus dans un modèle de Frappes de Processus standards ne peut se faire que par le jeu d'une action, elle-même déclenchée par la présence d'au plus deux processus : le frappeur et la cible (c'est-à-dire le processus qui va bondir vers un autre processus). Il n'est donc pas possible de conditionner le bond d'un processus par la présence de plusieurs processus de sortes différentes de celle de la cible. En effet, si on souhaite par exemple représenter l'activation d'un processus $c_1$ (d'une sorte $c$) uniquement lorsque deux autres processus $a_1$ et $b_1$ (de deux autres sortes $a$ et $b$) sont actifs, il n'est pas suffisant d'ajouter deux actions $a_1 \to c_0 \Rsh c_1$ et $b_1 \to c_0 \Rsh c_1$, car celles-ci permettent d'activer $c_1$ à la condition que $a_1$ soit actif ou que $b_1$ soit actif : il s'agit bien de deux interactions distinctes et non d'une coopération.

*Exemple.* Le modèle de Frappes de Processus de la figure 2.3 représente le mécanisme de segmentation métazoaire évoqué à la page 27. Dans ce modèle, la production de pigment

devrait uniquement être possible à la condition suivante : « $f$ est actif et $c$ n'est pas actif ». Or dans l'état courant du modèle, la désactivation du gène $f$ n'empêche pas la production de pigment, car depuis tout état contenant $f_0$, il est toujours possible d'activer $a$ à l'aide des actions $f_0 \to c_1 \not\Uparrow c_0$ et $c_0 \to a_0 \not\Uparrow a_1$.

Afin de représenter la coopération entre composants, et donc le raffinement de la dynamique des modèles, Paulevé et al. (2011*a*) ont proposé d'ajouter des sortes particulières appelées *sortes coopératives*, qui servent exclusivement à la modélisation. Une sorte coopérative permet de représenter l'état conjoint de plusieurs sortes dans le modèle. Pour cela, à chaque processus de la sorte coopérative correspond un sous-état des sortes qu'elle *représente*. Ainsi, il est possible de représenter les différents états combinés d'un ensemble de sortes, afin de ne jouer une action que dans une configuration particulière. Ces sortes ont l'avantage d'être des sortes standards, et donc de ne pas nécessiter d'enrichissement particulier de la sémantique. De plus, leur utilisation n'impacte pas les méthodes d'analyse de la dynamique développées : en effet, ces méthodes sont principalement impactées par le nombre de processus dans chaque sorte, et non le nombre total de sortes. Ainsi, à condition de limiter le nombre de processus dans les sortes coopératives, comme expliqué à la fin de cette section, il est possible de les utiliser en maintenant de bonnes performances d'analyse.

*Exemple.* Les Frappes de Processus standards de la figure 2.5 reprennent les trois sortes $f$, $c$ et $a$ du modèle de la figure 2.3 et comprennent en plus une sorte coopérative $fc$ permettant de détecter la présence simultanée de $f_1$ et $c_1$. Les processus de $fc$ décrivent les combinaisons possibles des états de $f$ et de $c$ : $fc_{00}$ correspond à $f_0$ et $c_0$, $fc_{01}$ correspond à $f_0$ et $c_1$, etc. Les actions en amont de cette sorte coopérative permettent de la mettre à jour, c'est-à-dire de changer son processus actif en fonction des évolutions du processus actif de $f$ et $c$. Par exemple, si $f_1$ est actif, les deux actions $f_1 \to fc_{00} \not\Uparrow fc_{10}$ et $f_1 \to fc_{01} \not\Uparrow fc_{11}$ effectuent cette mise à jour en faisant bondir le processus actif de $fc$ depuis un processus représentant la présence de $f_0$ ($fc_{00}$ ou $fc_{01}$) vers le processus correspondant représentant la présence de $f_1$ (respectivement $fc_{10}$ ou $fc_{11}$). Son action en aval, $fc_{11} \to c_0 \not\Uparrow c_1$, joue alors le rôle d'une coopération entre $f_1$ et $c_0$ pour frapper $a_0$ et le faire bondir en $a_1$.

*Exemple.* Afin de pallier partiellement l'absence de coopération entre les sortes $f$ et $c$ dans le modèle de la figure 2.3, il est possible d'intégrer la sorte coopérative décrite à la figure 2.5. La figure 2.6 propose un modèle corrigé de cette manière, avec une sorte coopérative $fc$ permettant de détecter la présence de $c_0$ et $f_1$. Les deux actions $c_0 \to a_0 \not\Uparrow a_1$ et $f_1 \to a_0 \not\Uparrow a_1$ sont alors remplacées par une action $fc_{10} \to a_0 \not\Uparrow a_1$ afin d'avoir une véritable coopération entre ces deux processus pour activer $a$.

Les sortes coopératives possèdent cependant un inconvénient, qui est lié au fait que les actions sont totalement indépendantes car le formalisme des Frappes de Processus standards est totalement asynchrone et indéterministe. En effet, les sortes coopératives ne sont pas nécessairement mises à jour immédiatement après un bond du processus actif d'une des sortes qu'elles représentent. Il peut ainsi exister un « décalage temporel » entre le changement de processus actif d'une sorte et la mise à jour des sortes coopératives. Ce décalage temporel permet alors de jouer des actions modélisant des coopérations dans des états où la coopération n'est plus possible, car même si l'un des processus modélisant la coopération a bondi, la sorte coopérative peut ne pas en avoir fait de même. Mais ce décalage peut aussi aboutir à des comportements indésirables. En effet, le processus actif

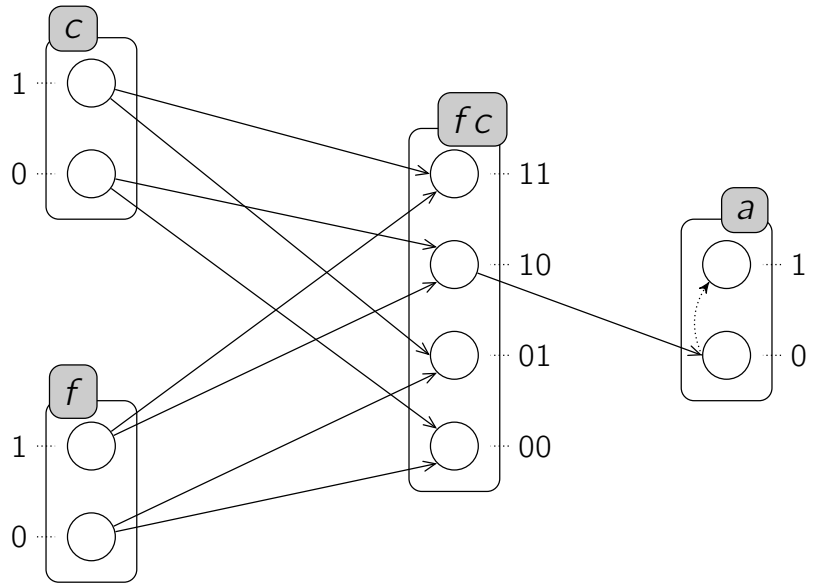



Figure 2.5 : Un exemple de Frappes de Processus standards avec une sorte coopérative $f\,c$. L'action $f\,c_{10} \rightarrow a_0 \Rsh a_1$ modélise une coopération entre $f_1$ et $c_0$ pour faire bondir le processus actif de $a$.

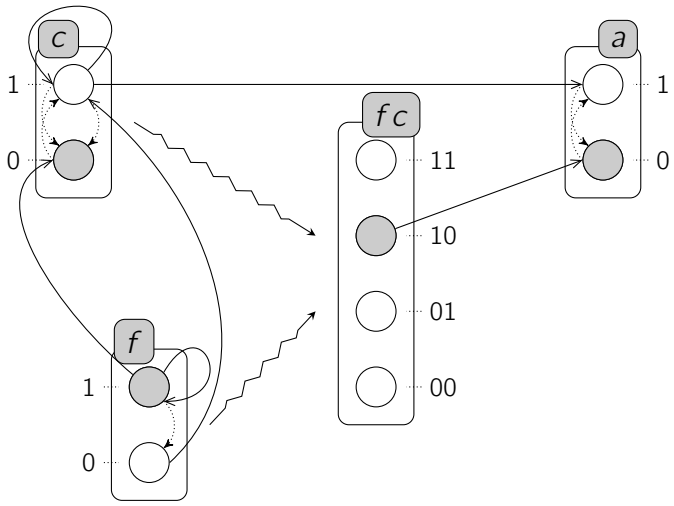



Figure 2.6 : Amélioration du modèle de Frappes de Processus de la figure 2.3 à l'aide de la sorte coopérative $f\,c$. Les processus de cette sorte représentent les différents sous-états formés par les deux sortes $f$ et $c$. Ainsi, $f\,c_{00}$ représente le fait que $f_0$ et $c_0$ sont actifs, etc. Les actions permettant la mise à jour de cette sorte coopérative n'ont pas été représentées explicitement ici mais sont symbolisées par les deux flèches en zigzag provenant de $f$ et $c$ et leur construction est immédiate.

d'une sorte coopérative ne correspond de fait pas à l'état courant des sortes représentées, mais uniquement à une combinaison d'états passés. Il est alors possible d'activer un processus de la sorte coopérative correspondant à un sous-état artificiel, c'est -à-dire non accessible aux sortes représentées. Ce mécanisme sera détaillé notamment au chapitre 3.

*Exemple.* Malgré l'ajout d'une sorte coopérative $fc$ dans le modèle de la figure 2.3, il faut noter que le comportement désiré n'est pas exactement représenté. En effet, l'ajout de cette sorte coopérative devait permettre d'éviter toute activation de $a$ lorsque $f$ devenait inactif, en permettant par exemple de jouer ce type de scénario depuis l'état initial $\langle a_0, c_0, f_1, fc_{10}\rangle$ :

$$f_1 \to f_1 \Rsh f_0 :: f_0 \to fc_{10} \Rsh fc_{00}$$

après lequel il n'est plus possible d'atteindre un état où $a_1$ est actif.

Cependant, il se trouve qu'il existe encore un cas particulier où $a$ peut être activé malgré la présence de $f_0$. Ce cas particulier relève du comportement mis en valeur ci-dessus, où une action a pour frappeur un processus de sorte coopérative qui ne devrait pas être actif si celle-ci avait été mise à jour. Il s'observe par exemple en jouant le scénario suivant depuis l'état initial $\langle a_0, c_0, f_1, fc_{10}\rangle$ :

$$f_1 \to f_1 \Rsh f_0 :: fc_{10} \to a_0 \Rsh a_1 \ ,$$

ce qui est possible parce que la sorte $fc$ n'a pas été mise à jour avant le jeu de l'action $fc_{10} \to a_0 \Rsh a_1$.

### Paramètres temporels & stochastiques

(Paulevé et al., 2011*a*) ont aussi proposé un enrichissement des Frappes de Processus standards à l'aide de paramètres stochastiques, l'objectif étant d'intégrer des données temporelles continues dans les modèles. Cet enrichissement est directement inspiré du *pi*-calcul stochastique (Priami, 1995). Cependant, la loi exponentielle utilisée pour la simulation stochastique possède une trop grande variabilité, l'approche a été raffinée par l'introduction d'un paramètre supplémentaire permettant de réduire l'intervalle de tir (Paulevé, Magnin & Roux, 2011*b*).

L'introduction de données dans les Frappes de Processus standards consiste à affecter un couple de paramètres stochastiques $(r\,;sa) \in \mathbb{N} \times \mathbb{R}$ à chaque action. La probabilité de tirer une action à un instant donné (sur un axe de temps continu) suit alors une loi d'Erlang (Evans, Hastings & Peacock, 2000) en fonction de ces deux paramètres, c'est-à-dire une somme de lois exponentielles. Le premier paramètre, appelé *taux*, indique le nombre de fois qu'une action peut être tirée par unité de temps. Le second paramètre est l'*absorption de stochasticité*, qui détermine le nombre de lois exponentielles sommées pour obtenir la distribution finale.

Tout couple de paramètres stochastiques $(r\,;sa)$ correspond à un intervalle de tir $[d\,;D]$, où $d, D \in \mathbb{R}$, pour un niveau de confiance $\alpha \in [0\,;1]$ donné, et inversement, qui peut être approximé (Paulevé, 2011, p. 72). Cette conversion permet de raisonner sur des intervalles de tirs plutôt qu'en termes de loi d'Erlang, ce qui permet notamment de définir des fenêtres de tir pour chaque action — une action devant être tirée dans sa fenêtre avec un niveau de confiance de $\alpha$. Au niveau de l'intervalle de tir, les deux paramètres stochastiques ont un rôle particulier :

- Le taux détermine la distance de l'intervalle de tir par rapport à l'origine de l'axe du temps ; autrement dit, une action avec un taux élevé sera tirée plus rapidement après sensibilisation.
- Le facteur d'absorption de stochasticité détermine la taille de l'intervalle de tir. Ainsi, augmenter ce facteur réduit la quantité $D - d$.

Enfin, il est aussi possible d'assigner un taux *infini* à une action ($r = \infty$), ce qui a pour conséquence de rendre le tir de l'action instantané : l'action doit être jouée dès sa sensibilisation. Si plusieurs actions de taux infini sont sensibilisées en même temps, le non-déterminisme est la règle et elles peuvent donc être jouées dans n'importe quel ordre. Une action de taux infini ne possède pas de facteur d'absorption de stochasticité, car ce deuxième paramètre ne changerait rien à sa condition de tir.

*Exemple.* Définissons les fenêtres de tir donnés à la figure 2.7, inspirées de (Paulevé et al., 2011*a*). Nous pouvons enrichir les Frappes de Processus standards décrites par la figure 2.6 en page 34 à l'aide de ces paramètres, de la façon donnée à la figure 2.8. Cette affectation nous permet notamment d'obtenir un comportement oscillant pour les sortes $a$ et $c$, et une auto-désactivation de $f$ après plusieurs oscillations. Les paramètres du couple **c** ont en effet été conçus de façon à ce que l'arrêt de l'horloge, provoqué par la désactivation de $f$, survienne après la formation des rayures (normalement 7).

Ces paramètres temporels et stochastiques permettent de simuler les modèles avec le simulateur Pint. Du fait de la nature indéterministe et probabiliste du modèle, plusieurs dynamiques sont possibles, et chaque simulation donne des résultats différents en termes de chemin d'exécution — ou, pour un même chemin d'exécution, en termes de temps de tir des actions. Il est donc possible d'effectuer un grand nombre de simulations pour obtenir des statistiques sur le comportement d'un modèle. Cette démarche sera utilisée dans le chapitre 3 et le chapitre 6 pour valider l'approche par intégration des données de séries temporelles.

Folschette (2014) a proposé de nombreux enrichissiments du formalisme des Frappes des Processus standards que nous présentons dans la suite.

**Frappes de Processus avec classes de priorités** Pour tout entier naturel $k$ non nul, les *Frappes de Processus avec k classes de priorités* sont des Frappes de Processus dont l'ensemble des actions est partitionné en $k$ ensembles, chacun étant associé à une classe de priorité distincte. Cela signifie qu'une action est jouable dans un état si et seulement si, en plus de la condition de la présence du frappeur et de la cible, il n'existe aucune autre action appartenant à une classe de priorité plus grande qui soit aussi jouable dans cet état. Il est à noter que les classes de priorités sont étiquetées de façon décroissante par les entiers de l'ensemble $[\![1\,;k]\!]$ en fonction de l'importance de la priorité ; autrement dit, la classe de priorités 1 contient les actions les plus prioritaires, ne pouvant jamais être préemptées, tandis que la jouabilité d'une action de la classe de priorité $k$ ne peut pas empêcher le jeu d'une autre action. Un exemple de ce type de modèle est donné par la figure 2.8, où les différentes priorités sont signifiées par des étiquettes numérotées sur les actions.

Cette modélisation permet notamment de distinguer les actions en fonction de différents critères comme leur vitesse d'exécution (les actions les plus rapides étant jouées en priorité), ou tout autre paramètre permettant de déterminer l'existence de la préemption d'une action en fonction de la possibilité d'en jouer une autre. L'application la plus

| | $\infty$ | **a** | **b** | **c** |
|---|---|---|---|---|
| $[d\,;D]$ | $[0\,;0]$ | $[0{,}742\,;1{,}29]$ | $[1{,}48\,;2{,}59]$ | $[23{,}9\,;35{,}4]$ |
| $(r\,;sa)$ | $r=\infty$ | $(1\,;50)$ | $(0{,}5\,;50)$ | $(0{,}034\,;100)$ |


Figure 2.7 : Exemples d'intervalles de tir et de paramètres stochastiques.

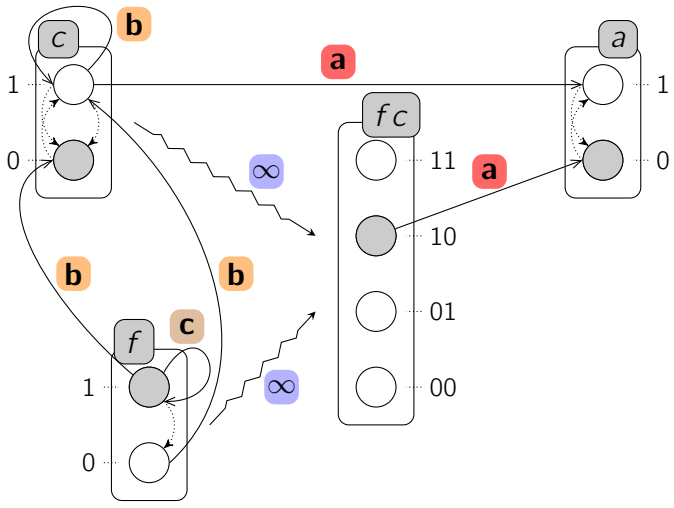



Figure 2.8 : Enrichissement des Frappes de Processus standards de la figure 2.6 à l'aide des paramètres stochastiques proposés à la figure 2.7. Chaque action est affectée à un couple de paramètres stochastiques repéré par une étiquette ($\infty$, **a**, **b**, **c**) placée contre l'action. Les ensembles d'actions mettant à jour la sorte coopérative $fc$, représentées par des flèches en zigzag, sont toutes affectées, possèdent toutes un taux infini ($r=\infty$).

poussée de cette utilisation consisterait à classer les actions d'un modèle en fonction d'un tel critère, et à attribuer à chaque classe de priorité une action unique en fonction de ce classement. De cette manière, les actions seraient arrangées selon un ordre total défini par leurs priorités.

De même, ces classes de priorités permettent de prendre en compte des comportements non biologiques inhérents à la modélisation. Il est par exemple possible de donner une priorité différente aux actions qui n'ont pas de sens biologique propre — mais dont ce sens émerge uniquement dans leur relation avec d'autres actions. L'application la plus immédiate de ce cas est celle des actions de mise à jour des sortes coopératives, comme cela est développé dans (Folschette, 2014), où une classe de priorités supérieure offre l'avantage de supprimer les effets d'entrelacement, et ainsi de simuler le comportement d'une véritable porte logique sans le problème de décalage temporel soulevé à la section 2.2.3.3.

Cette représentation basée sur des classes de priorités permet de modéliser un système dont les actions peuvent être distinguées en plusieurs classes en fonction de leur importance, de leur vitesse d'exécution, ou encore d'autres facteurs leur donnant prévalence sur d'autres. Par exemple dans les interactions entre composants d'un systèmes biologique, les interactions protéines-protéines n'ont pas les mêmes propriétés que les interactions protéines-gènes. Chaque action peut donc en préempter un ensemble d'autres en fonction de sa classe de priorité. Cela permet une représentation compacte des rapports de priorités entre actions ou, autrement dit, de leur ordonnancement, qui présente néanmoins quelques lacunes. Les phénomènes d'accumulation, notamment, n'y sont pas représentés ; un cycle d'actions prioritaires ne peut jamais être interrompu par une action moins prioritaire, menant à un cycle infini et pouvant contredire la réalité biologique. De plus, les classes de priorités définies pour un modèle sont invariables ; certains modèles pourraient cependant nécessiter l'évolution de certaines classes de priorités en fonction de la présence ou de l'absence d'un composant dans un état donné. Enfin, elles peuvent ne pas permettre la précision nécessaire à une représentation fidèle de certains modèles, notamment lorsqu'il est nécessaire de définir des préemptions ponctuelles comme le permettent les Frappes de Processus avec arcs neutralisants que nous ne présentons pas dans ce manuscrit mais qui sont abondamment discutées dans (Folschette, 2014). Cette extension des Frappes de Processus est bien adaptée pour la modélisation des réseaux de type RSTC. En effet, ces réseaux modélisent les composants différents (gènes, protéines, complexes, facteurs de transcription, etc.), situés à différents niveaux dans la cellule. Par exemple, les gènes sont localisés dans le noyau cellulaire, les protéines sont situées dans le cytoplasme. Ces composants interagissent entre eux et ces interactions peuvent être regroupées en classes. Certaines classes d'actions peuvent être prioritaires sur d'autres selon les choix et les considérations de modélisation. Aussi, nous nous servons de cette extension des Frappes de Processus dans le chapitre 3 pour la modélisation des réseaux de régulation biologique type RSTC.

**Frappes de Processus avec actions plurielles.** Afin de mieux prendre en compte un système au niveau de ses réactions biochimiques, c'est-à-dire des réactions entre les différents composants présents, Folschette (2014) a proposé un autre enrichissement des Frappes de Processus Standars qui sont les Frappes de Processus avec actions plurielles. Ces réactions peuvent avoir différentes formes (transformation, complexation, dissociation...), et il est fréquent qu'elles fassent intervenir plusieurs réactifs et plusieurs produits. Cette extension va dans le même sens que la sémantique booléenne de Biocham (Fages, Soliman

& Chabrier-Rivier, 2004) qui propose par exemple de modéliser un tel système de réactions biochimiques à l'aide d'un ensemble de règles de réaction de la forme : $X \xrightarrow{Y} Z$, ou encore : $X + Y \rightarrow Y + Z$, où $X$ est un ensemble de réactifs, $Y$ un ensemble de catalyseurs et $Z$ un ensemble de produits.

Les *Frappes de Processus avec actions plurielles* permettent de représenter de telles réactions mettant en jeu un nombre arbitraire de réactifs, de produits et de catalyseurs. Ainsi, une réaction de la forme : $X \xrightarrow{Y} Z$ peut être représentée à l'aide de l'action $A \rightarrowtail B$ où $A$ et $B$ sont deux ensembles des processus, $A$ regroupant tous les processus représentant les composants nécessaires à initier la réaction, et $B$ tous les processus qui ont évolué pendant la réaction. Une telle action peut donc être jouée dans un état contenant tous les processus de $A$ et fait évoluer celui-ci vers un état contenant tous les processus de $B$, les autres processus restant inchangés. Cela implique toutefois que pour tout processus de $B$, il existe un autre processus de la même sorte dans $A$. Les Frappes de Processus avec actions plurielles permettent donc de représenter un nombre arbitraire de bonds simultanés — autrement dit, de changements simultanés de processus actifs — déclenchés par un nombre arbitraire de prérequis — sous la forme de processus actifs dans l'état courant.

Un parallèle peut être tracé d'une part entre $A$ et l'ensemble des réactifs et catalyseurs, et d'autre part entre $B$ et l'ensemble des produits. Cependant, la modélisation par Frappes de Processus avec actions plurielles nécessite aussi de donner explicitement les composants qui sont absents. Par exemple, une réaction de complexation du type : $x + y \rightarrow c$ est représentée en Frappes de Processus avec actions plurielles à l'aide de trois sortes $x$, $y$ et $c$ contenant chacune deux processus et représentant respectivement les deux réactifs et le complexe produit, et par l'action $\{x_1, y_1, c_0\} \rightarrowtail \{x_0, y_0, c_1\}$.

Autrement dit, il est nécessaire de décomposer chaque élément en fonction de sa présence ($x_1$ et $y_1$) ou de son absence ($c_0$) au début comme à la fin de la réaction, et pas uniquement d'indiquer les composants présents en tant que réactifs ou produits.

On note ainsi qu'une réaction de la forme $\{a_0, b_0, c_0\} \rightarrowtail \{a_1, b_1\}$ ne peut pas être jouée si $a$ ou $b$ est déjà au niveau 1, comme c'est le cas par exemple dans l'état $\langle a_1, b_0, c_0 \rangle$. Un tel comportement a du sens lorsque les différents processus d'une sorte ($a_0$ et $a_1$, par exemple) représentent différents états d'une même molécule : la réaction ne peut alors pas être jouée pour des raisons de stœchiométrie. Cependant, si ces différents processus représentent plutôt des niveaux de concentration ($a_1$ représentant par exemple un niveau de concentration de la molécule $a$ plus élevé que $a_0$), cette restriction n'a plus de sens car une plus forte concentration d'une des entités ne devrait pas empêcher la réaction d'avoir lieu et de produire la seconde entité. Cela peut néanmoins être corrigé en ajoutant les actions $\{a_1, b_0, c_0\} \rightarrowtail \{a_1, b_1\}$ et $\{a_0, b_1, c_0\} \rightarrowtail \{a_1, b_1\}$, ou encore en séparant la production de $a_1$ et de $b_1$ en deux actions (ou ensemble d'actions) distinctes.

Cette forme des Frappes de Processus peut être aisément représentée à l'aide d'un réseau d'automates synchronisés, car chaque sorte possède un rôle similaire à celui d'un automate et chaque action pouvant être remplacée par un ensemble de transitions étiquetées avec le même libellé. Plus généralement, les Frappes de Processus avec actions plurielles peuvent être considérées comme une restriction des réseaux d'automates stochastiques introduits par (Plateau & Atif, 1991) pour la modélisation des systèmes parallèles. Nous proposons dans le chapitre 3 une formalisation des RRB en réseaux d'automates stochastiques. Le modèle que nous utilisons est centré transitions et mieux adapté pour de développement d'un raisonnement causal local. Ce qui permet de développer des méthodes d'analyse efficaces.

## 2.3 État de l'art de l'intégration des données quantitatives

Les modèles offrent la possibilité d'analyser, de simuler et de comprendre les systèmes. Dans le cas des systèmes biologiques, en fonction du niveau de précision, certains modèles rendent plus ou moins bien compte des propriétés des systèmes étudiés. D'autres modèles par contre, ont besoin d'être confrontés aux données et/ou enrichis des données pour être raffinés voire améliorés. En effet, la démarche habituelle dans la biologie des systèmes consiste à partir d'une base de connaissance, d'émettre de nouvelles hypothèses de travail, ces hypothèses vont conduire à effectuer des expérimentations. Les expérimentations génèrent des résultats qui sont dans la plupart des cas des données qui sont traitées afin d'identifier de façon fiable les éléments significatifs et les structures pour le phénomène biologique considéré. De ce traitement, les modèles sont construits en utilisant l'approche de modélisation qui correspond la mieux aux questions qui sont posées. Nous présenterons dans la section 2.3.1 quelques techniques de construction des modèles à partir des données expérimentales. Une fois les modèles générés, ils doivent être validés. Nous proposons une courte discussion sur la validation des modèles à la section 2.3.2. En plus d'être utilisées pour valider les modèles, les données peuvent également être utilisées pour enrichir les modèles et, de fait, offrir une opportunité pour une analyse plus raffinée. Cela passe généralement par un processus d'intégration des données dans les modèles. C'est l'objet du chapitre 3 de cette thèse. Nous présentons donc dans cette section un aperçu certainement pas exhaustif des travaux qui ont proposés une prise en compte des données expérimentales dans le processus de modélisation. Nous les avons regroupés en trois principales classes : les travaux qui permettent d'inférer les modèles à partir des données (section 2.3.1), ceux qui permettent de valider les modèles (section 2.3.2) et ceux qui permettent d'intégrer les données dans les modèles (chapitre 3).

### 2.3.1 Inférence des RRB à partir des données expérimentales

Plusieurs méthodes ont été proposées pour l'inférence des RRB à partir des données. C'est un processus qui nécessite au préalable (indépendamment de la méthode) de choisir le modèle d'architecture le plus adapté. Le modèle d'architecture décrit le comportement général des composants cibles en fonction de ses régulateurs. Ce n'est qu'à la fin de la définition du modèle d'architecture, qu'on peut définir la structure du modèle (les interactions entre les composants) et les paramètres du modèle (type et la force des interactions).

Dans le cadre de la définition du modèle d'architecture, plusieurs modèles d'architectures ont été proposés. Ils varient en fonctions des niveaux de simplifications et des différentes hypothèses émises pour la caractérisation des mécanismes moléculaires entre les composants. De façon générale, les nœuds du réseaux représentent les composants du système (les gènes, les protéines, les complexes, etc.). Les interactions entre les composants du système dépendent de comment sont abstraites les influences.

Ainsi donc, en fonction des choix de modélisation, l'inférence des RRB peut se faire par l'approche de la théorie de l'information en utilisant les réseaux de corrélations (Stuart, Segal, Koller & Kim, 2003) qui sont représentés par un graphe non orienté où les arrêtes sont pondérées avec les coefficients de corrélations. Aussi, il y aura un lien entre deux gènes si le coefficient de corrélations de leurs expressions est supérieur à un certain seuil. Ainsi, plus le seuil est élevé, plus le RRB inféré est clairsemé. À coté du coefficient de corrélation,

d'autres métriques existent pour prédire les relations entre les composants d'un RRB. La distance euclidienne et l'information mutuelle ont été appliquées dans (Steuer, Kurths, Daub, Weise & Selbig, 2002) pour détecter les dépendances des RRG.

De nombreux algorithmes ont été proposés pour ces méthodes dites de « reverse ingeniering ». Les avantages de ces méthodes sont leur simplicité et un faible coût de calcul. De plus, comme elles ne nécessitent pas un gros volume de données (Hecker, Lambeck, Toepfer, Van Someren & Guthke, 2009), elles sont adéquates pour inférer les grands RRB. Cependant, elles ne prennent pas en compte le fait que plusieurs composants peuvent participer à une même régulation. Elles ont de plus le désavantage d'être statiques c'est-à-dire qu'elles ne permettent pas de modéliser l'évolution du système en fonction du temps. Une conséquence immédiate est qu'il n'est pas possible de modéliser les boucles de rétrocontrôle (lorsque des variables d'un système interagissent entre elles de manière bouclée).

L'inférence des réseaux booléens (Kauffman, 1969; Thomas, 1973) nécessite d'avoir les données d'expression des gènes continues. Ces réseaux utilisent des variables binaires $x_i \in \{0, 1\}$ (comme nous l'avons présenté à la section 2.2.2 en page 22) qui définissent l'état du composant (gène, protéine). Pour chaque composant, sa courbe d'expression doit être discrétisée. La discrétisation peut s'effectuer soit par des méthodes de clustering ou par des méthodes de seuil. Les interactions entre les composants peuvent être représentées par des fonctions booléennes. Le défi est de déterminer ces fonctions booléennes tel que les observations des données d'expression des gènes expliquent le modèle. De nombreux algorithmes ont été proposés dans ce sens (Liang, Fuhrman & Somogyi, 1998).

L'inférence des réseaux bayésiens est particulière parce qu'elle permet de combiner différents types de données et de connaissances à priori dans le processus d'inférence. Les réseaux bayésiens reflètent la nature stochastique de la régulation des gènes. L'idée principale dans le processus d'inférence des réseaux bayésiens est de considérer les valeurs d'expression des gènes comme des variables aléatoires qui suivent une distribution de probabilité. Ainsi, les modèles bayésiens permettent de modéliser l'aléatoire et le bruit. De plus, il est possible de calibrer le modèle sur des données d'apprentissage et de mieux prendre en compte les données incomplètes et bruitées. Les méthodes pour l'apprentissage des réseaux bayésiens sont présentées en détail dans (Needham, Bradford, Bulpitt & Westhead, 2007; Heckerman, 1998).

Plusieurs autres méthodes et approches ont été utilisées pour inférer les modèles de régulations biologiques (statique ou dynamique) des protéines de signalisation ou des régulations géniques. Parmi ces méthodes, nous pouvons citer les travaux de (Gardner, Di Bernardo, Lorenz & Collins, 2003) (Pinna, Soranzo & de la Fuente, 2010), qui permettent d'inférer les réseaux de régulation statique à partir des états stables des ensembles d'expression des gènes. Ces méthodes se basent sur des modèles statistiques qui permettent de générer des modèles de petite taille ($\sim$ 10 composants) ou de taille moyenne (maximum 100 composants).

### 2.3.2 Valider les modèles à partir des données expérimentales

La validation des modèles consiste à déterminer la qualité du modèle conformément aux propriétés attendues et des données disponibles. Pour une validation quantitative, l'approche utilisée est la méthode de score. La méthode de score évalue le modèle conformément aux informations déjà utilisées pour générer le modèle (validation interne) et aux informations indépendantes (validation externe). En général, la qualité d'un modèle est déterminée en répondant aux questions suivantes :

– Est-ce que le modèle prédit correctement les comportements du système modélisé ?
– Est-ce que le modèle représente la structure réelle du système ?

La réponse à la première question peut se faire en comparant les résultats des simulations du modèle avec les données expérimentales du système réel. Nous mettrons cette approche en œuvre dans le chapitre 3 et dans le chapitre 6.

La réponse à la deuxième question est moins évidente. En effet, une réponse efficace à la deuxième question suppose de connaître la structure du système réel. Ce qui n'est pas toujours le cas. Dans bien des cas, les informations disponibles sur le système réel sont incomplètes, bruitées et pas toujours fiables. Une façon de contourner cette difficulté est d'utiliser les données synthétiques avec la conséquence que les performances de la méthode d'inférence du modèle sera fortement liée au modèle utilisé pour la construction des données artificielles.

Les méthodes suivantes accordent une importance à la cohérence du graphe d'interactions avec les données expérimentales. Ici, le graphe des interactions est très souvent connu sous le nom de PKNs (Prior Knowledge Networks). Les PKNs sont en partie collectés des bases de données différentes. Par exemple, KEGG (Kanehisa et al., 2015), Reactome (Joshi-Tope, Gillespie, Vastrik, D'Eustachio, Schmidt, de Bono, Jassal, Gopinath, Wu, Matthews et al., 2005), WikiPathways (Pico, Kelder, Van Iersel, Hanspers, Conklin & Evelo, 2008) dont la plupart sont accessibles via le portail Pathway Commons (Cerami, Gross, Demir, Rodchenkov, Babur, Anwar, Schultz, Bader & Sander, 2011) qui intègre beaucoup d'autres bases de données. Ces bases de données contiennent les informations provenant de la littérature, des résultats obtenus des publications basées sur des recherches expérimentales. Ce sont des réseaux de protéines ou de signalisations très utiles pour l'étude des propriétés topologiques (Ma'ayan, Jenkins, Neves, Hasseldine, Grace, Dubin-Thaler, Eungdamrong, Weng, Ram, Rice et al., 2005) des réseaux ou d'un mappage avec les données (Ideker & Sharan, 2008; Terfve & Saez-Rodriguez, 2012). Cependant les PKNs ne sont pas fonctionnels au sens où ils ne peuvent pas être simulés comme des processus de signalisation et de fait, comme une prédiction des sorties d'une expérience.

(Guziolowski, Bourdé, Moreews & Siegel, 2009) proposent de vérifier la cohérence d'un graphe des interactions avec des données expérimentales mesurant, à l'état stationnaire, la diminution ou l'augmentation des composants partant d'une solution initiale. Concrètement, les données obtenues sont de la forme : « le niveau de *a* est plus élevé, celui de *b* plus faible,etc. ». Les variations observées doivent alors s'expliquer par la topologie du graphe des interactions : par exemple si *b* diminue significativement, alors un de ses activateurs a été diminué ou un de ses inhibiteurs a été augmenté.

(Klamt, Saez-Rodriguez & Gilles, 2007) proposent des analyses similaires en utilisant une analyse de dépendance entre les composants afin de déterminer la cohérence d'un graphe des interactions avec des interactions des données expérimentales. Saez-Rodriguez, Alexopoulos, Epperlein, Samaga, Lauffenburger, Klamt & Sorger (2009) vont plus loin en proposant le logiciel CNO pour générer les modèles (booléens) logiques à partir des réseaux de signalisations et déterminer la topologie optimale des modèles générés à partir des données expérimentales. Cette démarche génère la structure optimale, mais ne garantit pas la structure optimale globale et de plus elle ne passe pas à l'échelle.

Une approche par programmation par contrainte est proposée dans (Videla, Guziolowski, Eduati, Thiele, Grabe, Saez-Rodriguez & Siegel, 2012) pour palier les limites de l'approche de (Saez-Rodriguez et al., 2009). L'approche proposée permet effectivement de générer les modèles optimaux contrairement à l'approche développée dans CNO et possède

une bonne complexité en terme de temps de calcul. Ainsi donc, il est possible d'inférer les réseaux booléens qui sont conformes au réseau de signalisation de protéines associé et de les confronter aux données phosphoprotéomiques.

Dans (Guziolowski, Videla, Eduati, Thiele, Cokelaer, Siegel & Saez-Rodriguez, 2013) l'outil CASPO est développé pour générer les modèles logiques (booléens) des signaux de transductions. Cette génération prend en compte les boucles de rétro contrôle.

Contrairement aux approches précédentes, qui regardent le système à deux instants de temps (le début et la fin), dans (Ostrowski, Paulevé, Schaub, Siegel & Guziolowski, 2015), l'idée est d'aller plus loin en généralisant les méthodes précédentes pour prendre en considération la dynamique complète ou du moins à plusieurs instants de temps. Le but est donc de générer les conditions nécessaires que doit vérifier la dynamique des réseaux booléens générés pour être cohérente avec les données expérimentales. Autrement dit, considérons les données de séries temporelles qui donnent la mesure d'une partie des composants biologiques pendant une expérimentation, il est question d'identifier tout les réseaux booléens qui ont une structure compatible avec le PKN d'entrée et qui reproduisent toutes les observations des données de séries temporelles.

Plus récemment, (Ben Abdallah, Ribeiro, Magnin, Roux & Katsumi, 2016) ont proposé une méthode originale pour la révision des modèles (qui peuvent être initialement vide) basée sur l'ASP et les données de séries temporelles. Son approche consiste à déterminer les délais des différents changements d'états des composants dans le système, et à proposer des actions (transitions) dans le modèle généré qui pourraient expliquer ces changements. Cette démarche produit, de fait, un ensemble de modèles cohérents avec les observations.

Malgré les efforts de modélisations qui permettent de valider/réviser les modèles à partir des données, très peu de méthodes se concentrent sur le raffinement de la dynamique des modèles en paramétrant les interactions avec les estimations issues des données expérimentales. C'est le but du chapitre 3 où nous présentons une contribution dans ce sens. Et puis, partant des modèles raffinés, nous validons par une comparaison avec les données de séries temporelles expérimentales.

## 2.4 État de l'art de la vérification des propriétés dans les modèles

Dans cette partie nous présentons une synthèse des résultats développés pour l'analyse et la vérification de la dynamique des RRB à grande échelle. Les principales techniques présentées sont des techniques par analyse statique qui permettent de proposer des réponses aux RRB de grandes taille. La section 2.4.1 présente quelques techniques de réduction des modèles introduites pour les RRB. La section 2.4.2 présente les opérations algébriques sur les diagrammes de décision. La section 2.4.3 présente les techniques par interprétation abstraite des dynamiques dans les RRB. Nous nous concentrerons dans cette section sur les propriétés d'accessibilité et leur vérification par analyse statique. Enfin, la section 2.4.4 en page 47 présente quelques travaux sur la vérification des propriétés quantitatives.

### 2.4.1 Réduction de Modèles

Une des approches naturelles lorsque nous faisons face à un grand modèle est de penser à le réduire dans un modèle plus petit et pouvant reproduire la même dynamique d'un point

de vue des propriétés recherchées. Les méthodes de réduction permettent d'extraire des propriétés sur les dynamiques du système. Elle peuvent également permettre de comprendre le rôle de certains composants en étudiant leur impact sur la dynamique globale du modèle.

Une approche pour la réduction des réseaux discrets à été proposée par (Naldi, Remy, Thieffry & Chaouiya, 2009). Dans cette approche, ils suppriment un composant $a$ dépourvu d'auto-régulation : les régulateurs de $a$ sont prolongés aux composants régulés par $a$ ; pour chaque composant régulé $b$, sa fonction discrète est modifiée en remplaçant toute occurrence de l'état de $a$ par le résultat de la fonction discrète $f^a$ (par exemple, $f^b(x) = x[c] \vee x[a]$ devient $f^b(x) = x[c] \vee f^a(x)$). Naldi et al. démontrent alors les propositions suivantes : les atteignabilités des composants conservés sont réduites (des transitions peuvent être supprimées par la réduction) ; les points fixes sont conservés ; les attracteurs cycliques sont conservés.

Dans le cadre des modélisations par équations différentielles des RRB (et plus généralement des systèmes de réactions), (Radulescu, Gorban, Zinovyev & Lilienbaum, 2008) établissent des méthodes de réductions en se basant sur la vitesse des réactions : selon les relations entre les réactions, celles ayant une vitesse faible ou élevée peuvent être enlevées sans impacts sur la dynamique globale. Cette méthode de réduction permet alors d'extraire les réactions critiques dont les vitesses peuvent influencer de manière notable la dynamique du système.

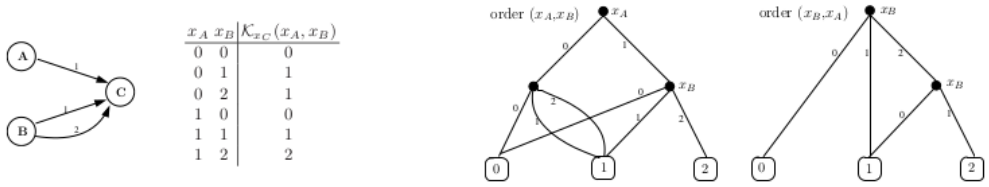
Nous citons également les travaux de (Gay, Soliman & Fages, 2010) permettant d'établir automatiquement des relations de réduction entre différents modèles en graphes bipartis décrivant un système de réactions (qui peut être considéré comme plus génériques que les réseaux discrets). Un modèle est alors considéré comme la réduction d'un autre si le premier peut être obtenu par un ensemble d'opérations de suppression et de fusion des réactions.

Enfin, nous citons les travaux de (Paulevé, 2016) qui propose une réduction des réseaux d'automates basé sur l'identification des transitions qui ne prennent pas part à une propriété d'accessibilité donnée et peuvent donc de fait être ignorées. Cette démarche conserve l'ensemble de traces minimales qui satisfont la propriété d'accessibilité. Pour cela, la méthode identifie les transitions qui ne sont pas concernées par la propriété d'accessibilité recherchée par une analyse statique des transitions causales dans chaque automate.

### 2.4.2 Opération algébrique sur les Diagrammes de Décision

Différentes analyses des Réseaux Discrets ont été proposées par (Naldi, Thieffry & Chaouiya, 2007) et (Hamez, Thierry-Mieg & Kordon, 2009) en utilisant les opérations algébriques sur les Diagrammes de Décision Multi-valués (DDM) afin de détecter efficacement les états stables de la dynamique et la fonctionnalité des circuits du graphe des interactions. Un DDM est un graphe orienté acyclique possédant une seule racine. Chaque nœud représente le test d'une variable et possède autant d'arcs sortants que de valeurs possibles : l'arc libellé par la valeur de la variable est alors sélectionné et amène au test d'une autre variable, jusqu'à arriver à une feuille, contenant alors la valeur de la décision. Étant donné un état $x$ définissant $n$ variables $x_1, \ldots, x_n$, il existe un unique chemin partant de la racine correspondant aux valeurs des variables et arrivant à une feuille.

Partant d'un Réseau Discret complètement paramétré, les paramètres de René Thomas de chaque composant sont encodés en DDM. Nous en avons une illustration à la figure 2.9 tirée de (Naldi et al., 2007)




| $x_A$ | $x_B$ | $\mathcal{K}_{x_C}(x_A, x_B)$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 0 | 2 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 2 | 2 |

Figure 2.9 : Illustration de l'utilisation des DDM pour la représentation des paramètres de René Thomas

### 2.4.3 Interprétation abstraite

Dans le domaine de l'interprétation abstraite où l'objectif principal est de fournir des analyses efficaces d'un modèle sans l'exécuter (Cousot & Cousot, 1977), des travaux ont été introduits pour permettre de comprendre les propriétés des systèmes biologiques. Parmi ces travaux, nous pouvons citer les travaux de (Danos, Feret, Fontana & Krivine, 2008) pour le modèle *kappa*.

Les travaux introduits dans la thèse de Loïc Paulevé (2011) et enrichi par Maxime Folschette (2014) proposent une approche très spécifique qui repose sur une interprétation abstraite des comportements concurrents des réseaux d'automates. À partir de la spécification du réseau d'automates, ils calculent des représentations abstraites de l'ensemble des comportements concernés par la propriété d'accessibilité recherchée. Ces représentations prennent la forme de graphes qu'ils ont appelés Graphes de Causalité Locale (GCL). Les abstractions faites oublient délibérément une partie de l'information sur l'ordre ou l'arité des transitions locales, résultant ainsi en des approximations supérieures et inférieures des comportements du modèle concret. Une analyse du GCL permet d'identifier les propriétés qui sont soit nécessaires, soit suffisantes à l'accessibilité étudiée.

Le principal avantage de cette méthode est une complexité très réduite comparée à une vérification formelle exacte : ce sont des approches exponentielles selon le nombre d'états au sein d'un seul automate, mais polynomiales selon le nombre d'automates. Ce qui permet de garantir leur applicabilité pour l'étude de la dynamique de très grands réseaux où chaque automate n'a que peu d'états locaux, ce qui est typiquement le cas des modèles qualitatifs des réseaux biologiques. Cependant, il existe un risque d'obtenir une réponse non concluante pour le modèle concret, nécessitant alors de raffiner l'analyse de la dynamique.

**Causalité locale.** Nous présentons ici de façon informelle, la démarche qui conduit à la construction d'un Graphe de Causalité Locale. En effet, il est possible de vérifier localement : (1) qu'un état local actif d'un automate peut bondir d'un niveau vers un autre en n'observant que les transitions locales à cet automate ; (2) que le jeu d'une transition est conditionné par au plus un état local d'un automate autre que celui qui contient la cible de la transition.

Aussi, du premier constat nous déduisons le fait que l'accessibilité d'un état local peut être résolue localement, en observant les transitions locales à cet automate. Une fois ce problème résolu localement pour cet automate, la deuxième constatation permet de déplacer le problème à d'autres automates afin d'activer les états locaux requis pour jouer chaque transition nécessaire. .

Cette démarche exhibe bien la construction récursive qui lie l'activation locale d'un

état local, les transitions requises pour le faire, et de nouveau l'activation des états locaux nécessaires pour jouer ces transitions, etc. Dans le but de réduire la complexité de la méthode, une approximation de la dynamique est effectuée à chaque étape : Pour chaque résolution d'accessibilité locale au sein de chaque automate, un ensemble d'états locaux requis appartenant à d'autres automates est produit en oubliant délibérément (par l'abstraction) l'ordre dans lequel ces états locaux sont nécessaires. Cela permet donc de déplacer l'accessibilité locale d'un état local à plusieurs accessibilités indépendantes dans d'autres automates.

**Sur- et Sous-approximation.** Cette approche permet de décliner une approximation supérieure (sur-approximation) et une approximation inférieure (sous-approximation) de l'ensemble de toutes les dynamiques possibles du modèle. La sur-approximation consiste à ne pas s'intéresser à l'ordre dans lequel les états locaux requis sont activables — et donc à l'ordre dans lequel les transitions résolvant l'accessibilité locale sont jouables. Cela autorise effectivement davantage de comportements, car en pratique un processus peut ne pas être activable après certaines transitions. La sous-approximation, à l'inverse, stipule que tous les états locaux requis doivent être activables dans tous les ordres possibles, bien qu'en pratique, tous les ordres ne soient pas intéressants pour la résolution.

Chacune de ces approximations est représentée à l'aide d'un *graphe de causalité locale*, qui est unique à chaque problème d'atteignabilité, et qui formalise les liens de causalité évoqués précédemment. Enfin, (Paulevé, Magnin & Roux, 2012) donnent une propriété qui, sous certaines conditions dépendant du graphe de causalité locale correspondant à la sous-approximation, stipule que l'état local donné est accessible depuis l'état donné ; de même, une propriété complémentaire est donnée pour le graphe de causalité locale correspondant à la sur-approximation, qui permet d'obtenir la conclusion inverse. Si aucune des deux propriétés n'est vraie, la méthode est dite non conclusive, et il est nécessaire de raffiner le problème ou le modèle.

Le calcul des deux graphes de causalité locales est polynomial dans le nombre de sortes et exponentielle dans le nombre de processus de chaque sorte des Frappes de Processus standards sur lesquelles la méthode est appliquée, et la vérification des deux propriétés l'est dans la taille des graphes obtenus. Ainsi, cette méthode est plus efficace que les approches par force brute car elle évite l'explosion combinatoire propre à l'analyse de la dynamique. Son implémentation produit des résultats en quelques dixièmes de seconde sur des modèles de plusieurs centaines de composants, et s'avère toujours conclusive sur la plupart des exemples étudiés. Toutefois, le rajout des priorités dans les actions des Frappes de Processus augmente les cas inconclusifs. Folschette (2014) dans sa thèse a exhibé quelques exemples et proposé une alternative à la condition suffisante énoncée dans Paulevé et al. (2012) qui permet de prendre en compte la séquentialité des objectifs plutôt que de les considérer simultanément, tel que cela est fait dans la version actuelle. Comme les objectifs sont pris en compte individuellement, une telle approche ne prend en compte qu'un sous-ensemble des scénarios possibles. Cependant, en se concentrant à chaque itération sur une plus petite partie du réseau, cette sous-approximation séquentielle peut s'avérer plus souvent conclusive.

Une partie de cette analyse statique a par la suite été exploitée dans le but d'approximer efficacement des ensembles de coupes, c'est-à-dire des ensembles d'états locaux nécessaires à une certaine accessibilité (Paulevé, Andrieux & Koeppl, 2013). Son utilisation dans ce cadre s'est avérée efficace sur des modèles de plusieurs milliers de composants.

Nous nous servons des résultats de cette analyse statique pour le développement de nouvelles méthodes d'analyse statique des propriétés quantitatives au chapitre 4 et pour l'identification des bifurcations au chapitre 5.

### 2.4.4 Vérification des propriétés quantitatives

Du fait du comportement non-déterministe des systèmes complexes, de nombreux travaux ont proposé des méthodes de vérification qui se veulent quantitatives (donner une probabilité, un délai) par opposition aux approches qualitatives qui répondent par oui/ non à une propriété donnée.

Dans ce sens on peut citer les travaux de (Bertrand, Bouyer, Brihaye & Markey, 2008), qui proposent de faire du model checking quantitatif sur les automates temporisés. Pour cela, ces travaux prennent avantage de la sémantique probabiliste est introduite dans (Baier, Bertrand, Bouyer, Brihaye & Größer, 2007) pour les automates temporisés. Cette nouvelle sémantique permet de calculer la probabilité d'une propriété régulière $\omega$ en corrigeant une abstraction par chaîne de Markov précédemment introduite dans (Baier, Bertrand, Bouyer, Brihaye & GröBer, 2008).

D'autres travaux ont été introduits pour faire du model checking quantitatif sur des systèmes abstraits sous forme de chaînes de Markov. Nous citons par exemple les travaux de (Hansson & Jonsson, 1994; Courcoubetis & Yannakakis, 1988) qui se sont intéressés aux systèmes à temps discret. Cependant, une grande majorité de systèmes sont à temps continu. Aussi (Zhang, Jansen, Nielson & Hermanns, 2011), ont introduit un formalisme CSL (Continuous Stochastic Logic) pour exprimer les propriétés des systèmes abstraits comme des chaînes de Markov à temps continu. Le principal résultat qu'ils ont apporté est la preuve que le problème de vérification des propriétés quantitatives est décidable. CSL est une logique inspirée par la logique temporelle CTL (Emerson, 1990) et ses extensions pour les systèmes stochastiques à temps discret (Hansson & Jonsson, 1994) et les systèmes non stochastiques à temps continu (Alur, Courcoubetis & Dill, 1990).

(Gao, Xu, Zhan & Zhang, 2013) proposent une vérification quantitative des propriétés CCSL qui sont une extension aux travaux de (Zhang et al., 2011), où ils prennent en compte les aspects concurrents des systèmes stochastiques. Cette extension permet de prendre en compte une plus grande classe de propriétés pour les chaînes de Markov à temps continu.

Afin de pouvoir appliquer ces méthodes à des systèmes de grandes taille, (Heiner, Rohr, Schwarick & Streif, 2010) proposent une méthode par analyse numérique des propriétés stochastiques. L'analyse évite d'explorer les états avec une probabilité inférieure à un certain seuil. Autrement dit, pendant l'exploration du système, si un état a une probabilité inférieure au seuil cet état n'est pas exploré.

Dans (Feret, Koeppl & Petrov, n.d.), les auteurs vont plus loin en proposant une réduction exacte des modèles stochastiques à base de règles. Ils proposent en effet une abstraction qui permet de réduire l'espace d'états pour les réseaux d'interaction protéine-protéine. L'approche utilisée est basée sur la construction des classes d'équivalences du réseau qui conservent la propriété markovienne. Si l'abstraction par classes d'équivalences conserve la propriété markovienne, on parle de *strong lumpability* (Kemeny, Snell et al., 1960) ou d'agrégation sûre. Ces travaux sont étendus dans (Feret, Henzinger, Koeppl & Petrov, 2012) pour une construction des classes d'équivalences qui conservent la propriété markovienne forte. Dans cette thèse, nous souhaitons aller plus loin dans la vérification des propriétés quantitatives des très grands systèmes. Pour cela, nous proposons de prendre

en compte dans l'analyse par interprétation abstraite des scénarios, les informations quantitatives.

## 2.5 Discussion

Le processus de modélisation des RRB propose plusieurs niveaux d'abstractions dans leur spécification en fonction des propriétés recherchées. Cela va des modélisations basées sur le graphe d'interactions simple, à un graphe intégrant (au fur et à mesure des rajouts des informations) des paramètres discrets, puis hybrides des dynamiques.

Le rajout des informations (paramètres) et la complexification des modèles est possible par la disponibilité des données expérimentales qui fournissent les informations nécessaires pour améliorer/raffiner le processus de modélisation. Cet enrichissement des modèles ouvre la voie à une étude plus détaillée des propriétés sur les dynamiques des RRB. Cela nécessite cependant un développement des méthodes d'analyse efficaces.

Obtenir des propriétés dynamiques très précises s'effectue par l'utilisation des méthodes de vérification formelle coûteuses et peu adaptées pour l'analyse des grands RRB à cause de l'explosion combinatoire des comportements.

Avec l'introduction du formalisme des Frappes de Processus et des développements des méthodes d'analyses statiques pour les propriétés d'accessibilité, Paulevé (2011) a ouvert la voie à l'analyse des grands RRB et notamment des propriétés dynamiques.

Nous nous appuyons donc sur ces travaux, notamment les Frappes de Processus et les réseaux d'automates qui en sont une généralisation pour proposer une modélisation discrète et hybride des grands RRB, où nous intégrons à la fois les paramètres temporels et stochastiques estimés des données de séries temporelles. Puis nous développons des méthodes d'analyses statiques originales pour une meilleure analyse des propriétés sur les dynamiques des grands RRB.

# Chapitre 3

# Intégration des séries temporelles dans les réseaux d'automates asynchrones

La modélisation et l'analyse fine des réseaux de régulations biologiques à large échelle est rendu possible par le développement de méthodes mathématiques et informatiques efficaces d'une part et l'évolution technologique d'autre part qui améliore d'un côté les performances des machines de calcul et qui fourni des quantités de données de plus en plus nombreuses de la dynamique des systèmes biologiques.
Nous proposons dans ce chapitre une approche innovante qui permet non seulement de formaliser les réseaux d'interactions comme des réseaux d'automates asynchrones et stochastiques, mais qui permet aussi de raffiner la dynamique de ces réseaux. Le raffinement de la dynamique est effectué aussi bien d'un point de vue qualitatif (par l'utilisation des sortes de synchronisation et les sortes coopératives), que d'un point de vue quantitatif en intégrant les informations stochastiques et temporelles estimées des séries temporelles. En plus du raffinement de la dynamique, nous proposons dans ce chapitre une analyse quantitative de la dynamique par le calcul des fréquences des *traces acceptantes* pour les composants du modèle.

L'intégration des séries temporelles dans les Frappes de Processus a été publié dans (Fitime, Schuster, Angel, Roux & Guziolowski, 2015).

## 3.1 Préliminaires

Dans ce chapitre, nous nous intéressons au raffinement de la dynamique des systèmes biologiques par l'intégration des données de séries temporelles. Les systèmes biologiques sont modélisés ici à l'aide des réseaux d'automates asynchrones. Les transitions qui gouvernent la dynamique des modèles sont raffinées par des paramètres temporels et stochastiques estimés des données de séries temporelles. En introduisant ces aspects stochastiques et

temporels dans les modèles, nous aspirons à reproduire avec plus de précision le comportement des systèmes réels.

Les systèmes biologiques sont un ensemble de composants qui interagissent pour réaliser une fonction biologique spécifique. Ces systèmes sont aujourd'hui représentés à l'aide des modèles et/ou des formalismes qui décrivent de façon plus ou moins formelle leurs propriétés. Les Réseaux de Régulation Biologique (RRBs) sont une modélisation des systèmes biologiques. Ils tendent à modéliser l'évolution des composants au sein d'un système biologique en fonction des interactions positives et négatives qu'ils exercent les uns sur les autres. Afin de représenter la structure d'un tel système, nous utilisons un graphe des interactions (définition 2.1 en page 21) qui représente un ensemble fini de composants se régulant entre eux. Les nœuds du graphe représentent les composants et les arcs (orientés et signés) leurs interactions mutuelles. La sémantique de ces régulations est fortement inspirée de celle de (Thomas, 1973). Ces composants sont donc décrits par un niveau d'expression discret qui caractérise leur état (taux d'activité pour un gène, concentration pour une protéine, taux de dissociation pour un complexe, etc.). Les Réseaux de Régulation Biologique que nous manipulons dans ce chapitre ont une structure particulière. Ils sont organisés en couche (ou strate). Les couches sont : une couche de récepteurs, une couche des protéines de signalisation, une couche des facteurs de transcription et une couche des gènes. Ce sont des réseaux dits RSTC (multi-layer Receptor Signaling Transcription Cell state) Cette structure permet de voir le système biologique comme un système qui a besoin d'une stimulation, et après un certain nombre d'interactions, détermine un comportement en particulier au niveau de l'expression de certains gènes et des nœuds représentant des processus biologiques. Afin de comprendre et d'analyser ces comportements, nous proposons une approche (illustrée à la figure 3.1) prometteuse qui vise à modéliser les réseaux de régulation avec la structure RSTC comme des réseaux d'automates en raffinant la dynamique des interactions par l'intégration des données expérimentales, qui sont dans notre cas, des séries temporelles.

La première contribution de notre approche est de proposer une définition formelle des *motifs minimaux* (définition 3.5 en page 57) et un algorithme efficace (algorithme 1) pour la détection de tels motifs dans les grands réseaux de régulation biologique. L'idée de l'identification des motifs est de considérer que chaque composant du système est influencé par les composants qui sont directement liés à lui. L'ensemble constitué du motif, des prédécesseurs directs et des influences associées constitue un motif minimal. L'algorithme de détection des motifs (algorithme 1) parcourt le graphe des interactions nœud par nœud afin de détecter tous les motifs minimaux qui constituent le graphe.

La disponibilité et l'abondance des données expérimentales permet d'envisager une meilleure compréhension et une meilleure connaissance des dynamiques des composants constituants un système biologique ainsi que le degré d'influence potentiel de ces composants les uns sur les autres. Notre deuxième contribution de ce chapitre est de partir des données de séries temporelles mesurées sur des systèmes biologiques réels, et de proposer un raffinement de la dynamique des modèles. Ce raffinement se fait en estimant les paramètres stochastiques et temporelles qui gouvernent la dynamique des interactions dans les réseaux d'automates asynchrones. L'idée est donc d'établir une relation entre la dynamique des séries temporelles et les paramètres stochastiques et temporels pour franchir les transitions dans les réseaux d'automates asynchrones. Établir cette correspondance nécessite de passer de l'espace continu des séries temporelles à l'espace discret des réseaux d'automates. Ce passage se fait à travers la discrétisation des niveaux dans les données des séries temporelles.

Enfin, dans le but d'évaluer la modélisation proposée, par rapport aux séries temporelles d'expression des gènes issues des expérimentations, nous avons proposé une analyse statistique des traces obtenues après la simulation stochastique. La simulation stochastique a été effectuée avec le simulateur Pint[1]. Cette analyse compare les traces des composants avec les données de séries temporelles discrétisées. Cette démarche permet ainsi de calculer la fréquence de traces acceptantes pour un composant après un nombre donné de simulations.

Ce chapitre est structuré comme suit, la section 3.2 donne une définition formelle des motifs biologiques dans les réseaux des interactions de type RSTC (définition 3.1) et un algorithme pour les identifier. Toujours dans la même section, nous présentons comment traduire ces motifs en d'autres motifs dans les réseaux d'automates asynchrones et les Frappes de processus. La section 3.3 détaille la démarche requise pour partir des données des séries temporelles au raffinement de la dynamique dans les modèles discrets tels que les réseaux d'automates et les Frappes de Processus. La section 3.4 présente la démarche pour l'évaluation des simulations par analyse statistique des traces. Les applications de ce chapitre sur le modèle de la différenciation des cellules de la peau sont présentées à la section 6.2 en page 134 du chapitre 6.

[1] `http://process.hitting.free.fr`

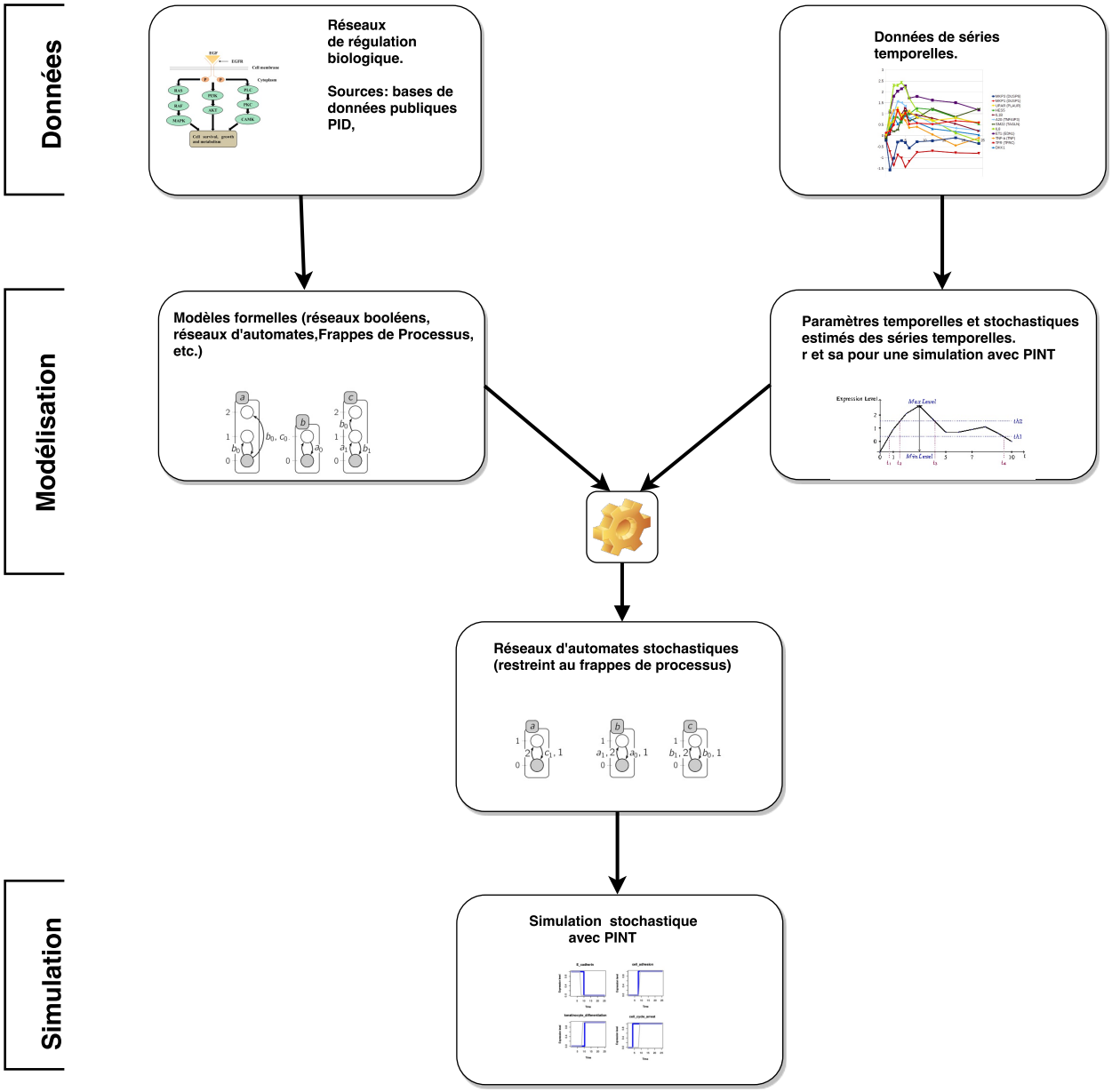



Figure 3.1 : Approche générale de la méthode que nous utilisons pour formaliser, raffiner les modèles par l'intégration des paramètres stochastiques et temporels, simuler et évaluer les modèles. La première couche représente les données d'entrée de la méthode. Les données les RRBs et les données de séries temporelles. À la deuxième couche nous proposons la formalisation des RRBs en réseaux d'automates asynchrones et en Frappes de Processus d'une part, puis d'autre part nous proposons d'estimer les paramètres temporels ($r$ et $sa$) qui sont intégrés dans les modèles pour une simulation à la troisième couche avec le simulateur Pint.

## 3.2 Identification des motifs dans les réseaux de régulation biologique

Dans cette section, nous présentons une définition des réseaux de régulation biologique. Cette définition s'inspire de celle introduite par (Thomas, 1973). Mais elle prend en compte un certain nombre d'informations additionnelles. En effet, les données disponibles sur les systèmes biologiques de nos jours, notamment dans les bases de données publiques (Schaefer et al., 2009*a*; Kanehisa et al., 2015) renseignent de plus en plus précisément sur la nature des interactions entre les composants des systèmes biologiques. Il est par exemple possible d'avoir des informations sur le type des composants (protéines, gènes, complexes, métabolites, etc.), sur la localisation du composant dans le système considéré (cytoplasme, noyau, paroi cellulaire, etc.), sur le nombre et la nature des interactions avec les autres composants du système.

Une connaissance plus précise des composants d'un système nous aide à déterminer dans le processus de modélisation le nombre de niveaux discrets des composants. D'un autre côté, une bonne connaissance des interactions du système nous aide à proposer une modélisation plus fine de la dynamique entre les composants du système. Nous proposons donc dans la suite une modélisation des réseaux type RSTC de façon formelle en réseaux d'automates asynchrones où nous prenons en compte de façon plus importante des informations disponibles afin de parvenir à un meilleur raffinement de la modélisation de la dynamique.

### 3.2.1 Définition des réseaux de régulation type RSTC

Nous présentons ici une définition formelle (définition 3.1) des réseaux de régulation type RSTC introduit par (Guziolowski, Kittas, Dittmann & Grabe, 2012). Nous nous appuyons sur cette configuration de réseau pour proposer une identification les *motifs minimaux* qui seront ensuite transformés en motifs équivalents dans les réseaux d'automates asynchrones stochastiques et en Frappes de Processus à la section 3.2.4 en page 59. Les réseaux type RSTC offrent une formalisation des graphes des interactions plus riche que celle offerte par le modèle de *Thomas*. Ce supplément de richesse vient du fait que, les différents types de composants sont clairement identifiés et non abstraits comme dans le modèle de *Thomas*. Il est donc possible avec les réseaux de type RSTC, de distinguer les protéines des gènes, les protéines des complexes, les gènes des facteurs de transcription pour ne citer que ces exemples. En plus de la richesse d 'expressivité pour les composants, il y a aussi une richesse d'expressivité au niveau des interactions. En plus des interactions comme définies dans le modèle de *Thomas* $a \xrightarrow{s,t} b$ section 2.2.1 en page 21, nous aurons dans les réseaux type RSTC, des interactions qui expriment explicitement des interactions complexes comme : les relations de complexation, de translocation, des conditions positives, etc. Nous présentons donc ici une définition des réseaux de régulation biologique type RSTC. Les réseaux de régulations biologiques type RSTC (multi layer Receptor Signaling Transcription Cell State) sont des réseaux de régulation qui regroupent un nombre fini de composants en interaction qui sont organisés en couches. Les principales couches de ce type de réseau sont : la couche *Receptor* qui regroupe les composants qui peuvent être stimulés grâce à une influence extérieure, la couche *Signaling* qui regroupe un ensemble généralement composé de protéines de signalisations, de complexes, et d'autres composants..., la couche *Transcription* qui regroupe un ensemble de facteurs de transcriptions

et les gènes, et enfin la couche *Cell State* qui regroupe les états cellulaires (prolifération, activation, différenciation, etc.). Plus formellement ces réseaux sont définis comme suit :

**Définition 3.1** (Réseau RSTC)**.** Un *réseau RSTC N* est un couple $(V, \mathcal{R})$, où :

- $V = V_T \bigcup V_I$ est l'ensemble fini des *nœuds* ; avec $V_T = \{v_{1t}, v_{2t}, \ldots, v_{n1t}\}$ est l'ensemble fini des nœuds terminaux ; $V_I = \{v_{1i}, v_{2i}, \ldots, v_{n2i}\}$ est l'ensemble fini des nœuds transitoires.
- $\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_m\}$ est l'ensemble fini des relations. $\mathcal{R}_i \subseteq (V_T \times V_T) \bigcup (V_T \times V_I) \bigcup (V_I \times V_T)$

*Exemple.*

La figure 3.2 présente un exemple de structure de réseau type RSTC. Le nœud $R_e$ représente un nœud de la couche Receptor. Il est à préciser que cette couche peut avoir plusieurs nœuds. L'ensemble $V_T$ pour la figure 3.2 est constitué du nœud $R_e$, des nœuds qui représentent les protéines de signalisations $ps_1, \cdots, ps_9$, des nœuds qui représentent les complexes $cplx_1, \cdots, cplx_6$, des états cellulaires $cs_1, cs_2$, des facteurs de transcription $tf_1, \cdots, tf_4$, des gènes $a, b, c, d$. Les nœuds transitoires sont représentés par des petits carrés sur la figure.

Les gènes et les facteurs de transcription appartiennent à la couche *Transcription*. Les états cellulaires appartiennent à la couche *Cell State*. Les complexes et les protéines de signalisations appartiennent à la couche *Signaling*.

$\mathcal{R}$ représente les interactions qui existent entre les nœuds du réseau. Ces relations sont des activations, des conditions positives, des inhibitions, des complexations, des dissociations de complexes, etc. Le tableau 3.1 repertorie quelques relations que nous avons identifiées dans les réseaux type RSTC. Conformément aux descriptions du tableau 3.1 et à l'exemple de la figure 3.2, les relations dans un réseau type RSTC sont interprétées de la manière suivante :

- $x\mathcal{R}_1 y$ signifie *x active y*. Par exemple, nous avons la relation $R_e(\mathcal{R}_1)cplx_2$ dans la figure 3.2 qui indique que le composant $R_e$ est un activateur du composant $cplx_2$.
- $x\mathcal{R}_2 y$ signifie *x est un inhibiteur de y*. Une illustration de cette relation dans la figure 3.2 est la relation $a(\mathcal{R}_2)ps_6$ qui indique que le composant $a$ est un inhibiteur du composant $ps_6$.

*Exemple.* Exemples des interactions possibles entre les composants du réseau de type RSTC de la figure 3.2

La figure 3.3 donne une représentation graphique de quelques exemples des interactions entre composants qui sont extraits de la figure 3.2.

### 3.2.2 Une définition des motifs dans les réseaux de régulations biologiques type RSTC

Si on s'intéresse aux influences que les composants ont les uns sur les autres, on peut concevoir un motif comme un ensemble de composants ayants une influence sur un autre ensemble de composants. De l'exemple de la figure 3.2 à la page ci-contre, nous pouvons extraire quelques motifs qui sont représentés à la figure 3.3 et à la figure 3.5. La définition 3.4 propose une définition formelle de la notion de *motif*. Cette définition est basée sur la notion de relation en mathématique. Plus loin, la définition 3.5 introduit la notion de

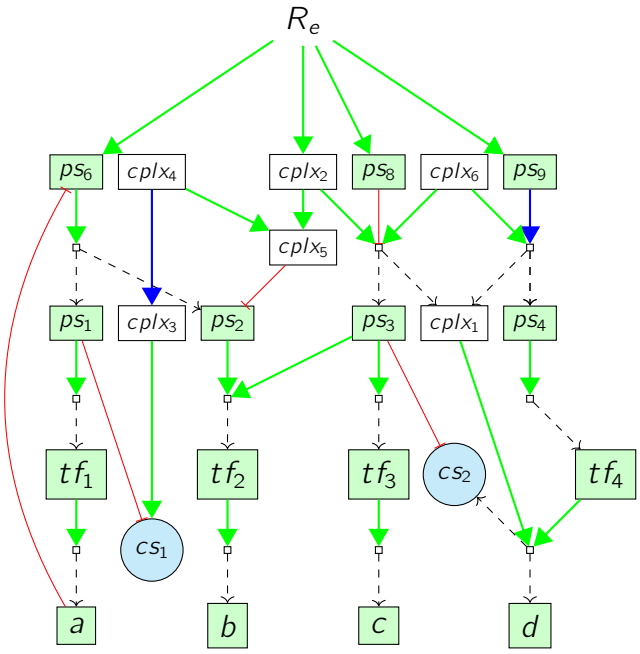



Figure 3.2 : Exemple de structure d'un réseau type RSTC. Le nœud $R_e$ est le récepteur, les rectangles claires représentent les complexes, les rectangles verts représentent les protéines de signalisation, les carrés représentent les gènes et les facteurs de transcription. Les cercles représentent les états cellulaires. Les flèches représentent les relations qui sont décrites dans le tableau 3.1.

Table 3.1 : Exemple de relation dans un réseau type RSTC. Ces relations indiquent le type ou encore la nature des interactions qu'il est possible de retrouver dans un réseau de type RSTC. La première colonne **Relations** est la colonne de la désignation formelle des relations telle que nous définissons dans ce manuscrit. La deuxième colonne **Symboles** contient les symboles utilisés pour les relations dans les réseaux type RSTC. Enfin la troisième colonne **Descriptions** donne une description de la relation.

| **Relations** | **Symboles** | **Descriptions** |
|---|---|---|
| $\mathcal{R}_1$ | → | Simple activation |
| $\mathcal{R}_2$ | ⊣ | Simple inhibition |
| $\mathcal{R}_3$ | - - - - → | Transition d'état |
| $\mathcal{R}_4$ | → | Condition positive |
| $\mathcal{R}_1 \circ \mathcal{R}_3$ | →- - - - → | Activation |

*motif minimal* dans les RSTC. L'identification des motifs dans les réseaux de type RSTC que nous présentons en section 3.2.3 est basée sur ce concept de motif minimal.

Ainsi, afin de proposer une caractérisation et une identification efficace des motifs minimaux dans les réseaux de type RSTC, nous introduisons un ensemble de concepts qui nous aident dans cette tâche. Nous définissons tout d'abord l'ensemble $\mathbf{Comp}_{\mathcal{R}}$ (définition 3.2) qui est l'ensemble des compositions des relations sur $\mathcal{R}$. En effet, cet ensemble contient toutes les compositions possibles qui peuvent être effectuées avec les relations de l'ensemble $\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_m\}$. Il est important de procéder de la sorte pour pouvoir

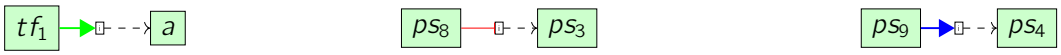



Figure 3.3 : **(Gauche) Simple activation.** Dans cette interaction le composant $tf_1$ active le composant $a$. **(Centre) Simple inhibition** Dans cette interaction le composant $ps_8$ inhibe le composant $ps_3$. **(Droite) Condition positive** Dans cette interaction le composant $ps_9$ est une condition positive pour le composant $ps_4$.

capturer les relations composées qui peuvent être observées dans les réseaux type RSTC. Ces réseaux mettent en relation les composants grâce à des relations qui sont des compositions des relations de bases de l'ensemble $\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_m\}$. La définition 3.2 donne donc une formalisation de l'ensemble des compositions des relations.

**Définition 3.2** ($\mathbf{Comp}_\mathcal{R}$)**.** $\mathbf{Comp}_\mathcal{R}$ est l'ensemble des compositions des relations de $\mathcal{R}$. $\mathbf{Comp}_\mathcal{R} = \{(\mathcal{R}_1 \circ \mathcal{R}_2 \ldots \circ \mathcal{R}_p) \mid \mathcal{R}_i \in \mathcal{R} \wedge p \leq m, avec \quad m = |\mathcal{R}|\}$

Les compositions de relation définies à la définition 3.2 peuvent mettre en relation aussi bien les éléments de $V_I$ que ceux de $V_T$. Cependant, les éléments de $V_I$ ne sont pas des entités biologiques et permettent juste de caractériser les interactions. Aussi, nous proposons à la suite de la définition 3.2 une définition de la composition des relations fonctionnelles $\mathbf{Compf}_\mathcal{R}$ sur $\mathcal{R}$ comme étant l'ensemble des compositions mettant en relation deux éléments de $V_T$. De cette façon, il est plus aisé de caractériser les sous-ensembles d'éléments ayant une réalité et un fonctionnement biologique.

**Définition 3.3** ($\mathbf{Compf}_\mathcal{R}$)**.** $\mathbf{Compf}_\mathcal{R}$ est l'ensemble des compositions des relations de $\mathcal{R}$ mettant en relation deux éléments de $V_T$. $\mathbf{Compf}_\mathcal{R} = \{(\mathcal{R}_1 \circ \mathcal{R}_2 \ldots \circ \mathcal{R}_p) \mid \mathcal{R}_i \in \mathcal{R} \wedge p \leq m \wedge (a(\mathcal{R}_1 \circ \mathcal{R}_2 \ldots \circ \mathcal{R}_p)b \Longrightarrow a, b \in V_T \quad et \quad m = |\mathcal{R}|\}$

Nous introduisons maintenant la définition de ce que c'est qu'un *motif*. Un *motif* est un composant biologique (protéine, gène, complexe, etc.) que nous notons ici par $b$, un ensemble de prédécesseurs (qui peuvent avoir une influence directe ou indirect) sur le composant, et un ensemble de composition de relation ayant $b$ comme deuxième opérande. L'ensemble des prédécesseurs est noté par $\mathbf{Infl}(b)$. Il est à préciser que cette définition est compatible avec la notion de motif couramment utilisée en biologie des systèmes en ce sens qu'elle permet de caractériser aussi bien les motifs connus que ceux inconnus. La définition 3.4 donne une expression plus formelle de cette définition.

Nous notons par $\mathbf{Infl}(b)$ l'ensemble des composants ayant une influence sur le composant $b$

$$\mathbf{Infl}(b) \triangleq \{a \mid \exists \mathcal{T} \in \mathbf{Compf}_\mathcal{R} \wedge a\mathcal{T}b\}$$

**Définition 3.4** (Motif)**.** Un motif est un 3-uplet de la forme $(b, \mathbf{Infl}(b), \{a\mathcal{T}b \mid \mathcal{T} \in \mathbf{Compf}_\mathcal{R}\})$ tel que :

- $b$ est le composant sur lequel s'exerce les influences.
- $\mathbf{Infl}(b)$ est l'ensemble des composants ayant une influence sur $b$.
- $\{a\mathcal{T}b \mid \mathcal{T} \in \mathbf{Compf}_\mathcal{R}\}$ est l'ensemble des compositions des relations fonctionnelles ayant $b$ comme deuxième opérande.

Une *plus petite composition fonctionnelle* est tout élément $\mathcal{T} \in \mathbf{Compf}_\mathcal{R}$ tel qu'il n'existe pas d'autre élément $\mathcal{T}' \in \mathbf{Compf}_\mathcal{R} \mid \mathcal{T}' \subseteq \mathcal{T}$. L'intérêt d'une *plus petite compo-*

*sition fonctionnelle* est qu'elle permet, lors de la détection des motifs, partant d'un nœud donnée, de remonter uniquement jusqu'à ses prédécesseurs directs appartenant à $V_T$. Nous désignons par $\mathbf{PPCompf}_{\mathcal{R}}$ l'ensemble des *plus petites compositions fonctionnelles*.

$$\mathbf{PPCompf}_{\mathcal{R}} \triangleq \{\mathcal{T} \in \mathbf{Compf}_{\mathcal{R}} \mid \nexists \mathcal{T}' \in \mathbf{Compf}_{\mathcal{R}} \wedge \mathcal{T}' \subseteq \mathcal{T}\}$$

Nous notons également par $\mathbf{Infld}(b)$ l'ensemble des composants ayant une influence directe sur le composant $b$

$$\mathbf{Infld}(b) \triangleq \{a \mid \exists \mathcal{T} \in \mathbf{PPCompf}_{\mathcal{R}} \wedge a\mathcal{T}b\}$$

**Définition 3.5** (Motif Minimal)**.** Un motif minimal est un 3-uplet de la forme $(b, \mathbf{Infld}(b), \{a\mathcal{T}b \mid \mathcal{T} \in \mathbf{PPCompf}_{\mathcal{R}}\})$ tel que :

- $b$ est le composant sur lequel s'exercent les influences.
- $\mathbf{Infld}(b)$ est l'ensemble des composants ayant une influence sur $b$.
- $\{a\mathcal{T}b \mid \mathcal{T} \in \mathbf{PPCompf}_{\mathcal{R}}\}$ est l'ensemble des compositions des relations fonctionnelles minimale ayant $b$ comme deuxième opérande.

*Exemple.* Exemples illustrant la notion de motif et de motif minimal dans les réseaux de type RSTC.

La figure 3.4 donne une représentation graphique des motifs extraits de la figure 3.2. La partie gauche de la figure est un motif (quelconque par opposition à minimal) qui comporte 7 composants dont 4 composants terminaux ($ps_6$, $ps_1$, $tf_1$, $a$). La partie gauche de la figure comporte aussi 7 relations. Ce motif (de la partie gauche) n'est pas minimal parce qu'il est possible d'extraire de ce motif des motifs minimaux qui sont représentés par les parties de la figure au centre et à droite.

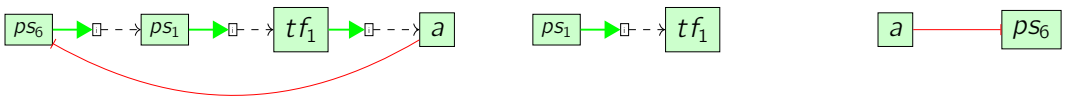



Figure 3.4 : **(Gauche) Motif dans un réseau type RSTC. (Centre) Motif Minimal.** Ce motif est extrait du motif de gauche selon la définition 3.5. **(Droite) Motif minimal.** Un autre motif extrait de la figure de gauche.

### 3.2.3 Identification des motifs minimaux

Nous introduisons dans cette section un algorithme de détection de motifs minimaux (algorithme 1) dans les réseaux de régulation biologique type RSTC. L'idée de l'algorithme est de détecter les relations telles que indiqué par la définition 3.5. Partant du principe qu'un motif est constitué d'un nœud (sur lequel s'exerce une influence), de ses prédécesseurs directs (les nœuds qui exercent l'influence) et du nombre et des types des relations entre les nœuds, l'algorithme parcourt le graphe représentant le réseau nœud par nœud. Pour chaque nœud, il identifie toutes les relations directes incidentes à ce nœud, et les nœuds associés à ces relations. Ainsi, en fonction du nombre de prédécesseurs, du nombre et du type de relation, on a un motif spécifique de notre réseaux de régulation biologique. L'algorithme 1 formalise cette démarche.

**Algorithm 1 : Algorithme pour la détection des motifs dans les réseaux type RSTC, fonction detectionMotif (*Net*, *n*)**

**Require:** *Net*, *n* {*Net* est un réseau type RSTC et *n* est le nœud courant sur lequel on va s'appuyer pour détecter un motif}

**Ensure:** Construit un motif minimal de la forme $M = (n, \mathbf{InfId}(n), \mathbf{PPCompf}^{n}_{\mathcal{R}} = \{a\mathcal{T}n \mid \mathcal{T} \in \mathbf{PPCompf}_{\mathcal{R}}\})$.

```
1: switch (n)
2: case n ∈ V_T:
3:    n ∈ M {n est la première composante du motif}
4:    switch (|InfId(n)|)
5:    case 1:
6:       for all p ∈ InfId(n) do
7:          switch (p)
8:          case p ∈ V_T:
9:             PPCompf^n_R ← PPCompf^n_R ∪ pTn ;
10:         case p ∈ V_I:
11:            effectuer la composition ;
12:            detectionMotif (Net, p) ;
13:         end switch
14:      end for
15:      Affecter un code à M ;
16:      renvoyer M ;
17:   case 2:
18:
19:   end switch
20: case n ∈ V_I:
21:   switch (|InfId(n)|)
22:   case 1:
23:      for all p ∈ InfId(n) do
24:         switch (p)
25:         case p ∈ V_T:
26:            PPCompf^n_R ← PPCompf^n_R ∪ pTn ;
27:         case p ∈ V_I:
28:            effectuer la composition ;
29:            detectionMotif (Net, p) ;
30:         end switch
31:      end for
32:      Affecter un code à M ;
33:      renvoyer M ;
34:   case 2:
35:
36:   end switch
37: end switch
```

L'algorithme 1 parcourt tous les nœuds du réseau un par un pour construire le motif minimal associé à chaque nœud. Pour chaque nœud $n$, il s'agit de déterminer l'ensemble $\mathbf{InfId}(n)$. Cette détection nécessite de parcourir un arbre enraciné en $n$ qui a une hauteur $h$

égale à $\mathcal{O}(|V|)$ dans le pire des cas et à $\log(|V|)$ dans le cas moyen. D'où nous établissons la proposition 3.1.

**Proposition 3.1** (Complexité)**.** *La Détection des motifs est effectuée avec une complexité de* $\mathcal{O}(|V|\log(h))$*, avec* $h = \log(|V|)$*. Où h est la hauteur moyenne des motifs.*

L'intérêt d'une méthode automatique de traitement des motifs dans les réseaux de régulation biologique est que nous pouvons traiter les réseaux biologiques de grandes tailles existant dans les bases de données comme PID pour notre cas d'étude.

Appliqué à l'exemple de la figure 3.2 en page 55, l'algorithme 1 produit les motifs de la figure 3.5 comme résultats. Ce qui permet de donner une idée du principe de fonctionnement de l'algorithme.

*Exemple.* Exemples de motifs.

La figure 3.5 donne une représentation graphique de quelques exemples de motifs qui sont extraits par l'application de l'algorithme 1.

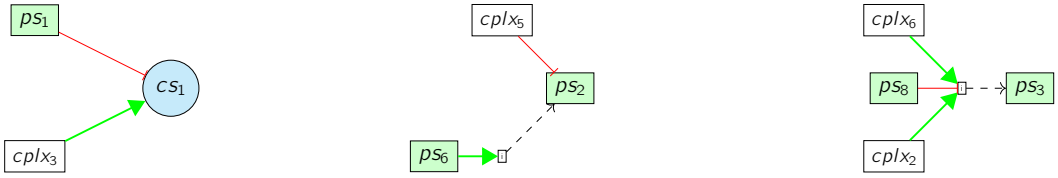



Figure 3.5 : **(Gauche) Simple activation.** Dans ce motif le composant $cplx_3$ active le composant $CS_1$ indépendemment du composant $ps_1$ qui inhibe le composant $CS_1$. **(Centre) Simple inhibition** Dans ce motif le composant $cplx_5$ inhibe le composant $ps_2$ pendant que $ps_6$ en est un activateur. **(Droite) Condition positive** Dans ce motif les composants $cplx_2$ et $cplx_6$ sont des activateurs pour le composant $ps_3$. Le composant $ps_8$ est un inhibiteur de $ps_3$.

### 3.2.4 Des réseaux de régulation biologique vers les réseaux d'automates asynchrones

Nous donnons dans cette section la démarche de traduction des motifs RSTC vers les réseaux d'automates asynchrones et les Frappes de Processus. Nous commençons par donner une définition des réseaux d'automates et des Frappes de Processus avec deux classes de priorités dont nous avons résumé les propriétés dans le chapitre 2. Puis nous illustrons comment il est possible de passer des motifs des réseaux RSTC vers les réseaux d'automates asynchrones. Pour ce faire nous nous appuyons sur trois exemples de motifs qui vont nous aider à donner l'intuition de notre démarche.

#### 3.2.4.1 Définition des modèles formels : les réseaux d'automates asynchrones et les Frappes de Processus

Nous définissons dans cette section les réseaux d'automates (définition 3.6) et les Frappes de Processus avec classes de priorité (définition 3.7). Ces deux formalismes permettent dans la suite à formaliser les réseaux de régulation biologique. Les réseaux d'automates sont un ensemble fini de composants appelés automates. Chaque automate $a$ possède un ensemble fini d'état locaux $S(a) = \{a_0, \ldots, a_{k_a}\}$. Nous notons **LS** l'ensemble de tous les états locaux. À chaque automate correspond un ensemble de transitions locales $T = \{a \mapsto$

$T_a \mid a \in \Sigma\}$. À chaque transition locale, on associe un taux $r$ qui indique le nombre de fois que la transition peut être franchie par unité de temps.

**Définition 3.6** (Réseau d'Automates $\mathcal{SAN} = (\Sigma, S, T)$)**.** Un *Réseau d'Automates* est défini par le tuple $\mathcal{SAN} = (\Sigma, S, T)$ où

- $\Sigma$ est l'ensemble fini d'automates ;
- Pour chaque $a \in \Sigma$, $S(a) = \{a_0, \ldots, a_{k_a}\}$ est l'ensemble fini des états locaux de l'automate $a$ ; $S \triangleq \prod_{a\in\Sigma} S(a)$ est l'ensemble fini des états globaux ;
  **LS** $\triangleq \bigcup_{a\in\Sigma} S(a)$ est l'ensemble de tous les états locaux.
- $T = \{a \mapsto T_a \mid a \in \Sigma\}$, où $\forall a \in \Sigma, T_a \subseteq S(a) \times 2^{\text{LS}\setminus S(a)} \times \mathbb{R} \times S(a)$ avec $(a_i, \ell, r, a_j) \in T_a \Rightarrow a_i \neq a_j$ et $\forall b \in \Sigma, |\ell \cap S(b)| \leq 1$, est une correspondance des automates vers l'ensemble des transitions locales. $r$ est le taux (plus de détails sont donnés sur le taux à la section 3.3.2 en page 68) de la transition.

Nous écrivons $a_i \xrightarrow{\ell,r} a_j \in T \triangleq (a_i, \ell, r, a_j) \in T(a)$ et $a_i \to a_j \in T \triangleq \exists \ell \in 2^{\text{LS}\setminus S(a)}, a_i \xrightarrow{\ell} a_j \in T$. Étant donné une transition $t = a_i \xrightarrow{\ell} a_j \in T$, nous notons $\text{orig}(t) \triangleq a_i$, et $\text{dest}(t) \triangleq a_j$, $\text{enab}(t) \triangleq \ell$, ${}^\bullet t \triangleq \{a_i\} \cup \ell$, et $t^\bullet \triangleq \{a_j\} \cup \ell$. La relation globale de transition $\to \subseteq S \times S$ est définie par :

$$s \to s' \triangleq \exists \ell \in \ : \forall a_i \in {}^\bullet\ell, s(a) = a_i \wedge \forall a_j \in \ell^\bullet, s'(a) = a_j \\ \wedge \forall b \in \Sigma, S(b) \cap {}^\bullet\ell = \emptyset \Rightarrow s(b) = s'(b).$$

À chaque instant, chaque automate est dans un et un seul état local. Cet état est l'état local actif. L'ensemble des états locaux actifs forment l'état global du réseau. Aussi l'ensemble des états globaux du système est noté $S$ et est égal à $\prod_{a\in\Sigma} S(a)$. Étant donné un état global $s \in S$, $s(a)$ est l'état local de l'automate $a$ dans $s$, c'est-à-dire la $a$ ième coordonnée de $s$.

Une transition locale $t = a_i \xrightarrow{\ell,r} a_j \in T$ est jouable dans un état global $s \in S$ quand $a_i$ est actif et tous les états locaux dans $\ell$ sont dans $s$. L'application de la transition locale notée par $s \cdot t$, remplace l'état local de $a$ par $a_j$ (définition 5.2).

Une transition globale du système est donnée par $s \xrightarrow{t} s'$ où $s' = s \cdot t$.

Dans toute cette thèse, nous supposons une sémantique *asynchrone* des réseaux d'automates : à chaque instant, une seule transition locale peut être appliquée. Ce qui implique que seul un automate change un état local pour chaque transition entre deux états globaux.

Dans cette sémantique, plusieurs transitions locales peuvent être jouées dans le même état. Ce qui peut potentiellement générer plusieurs dynamiques différentes. Le choix de la transition a jouer est *non-deterministe*. Nous nous servons de ce formalisme pour modéliser les réseaux de régulation biologique. C'est un formalisme centré sur les transitions par opposition aux formalismes centrés sur les fonctions (comme les modèles de Thomas) qui permet de construire des approximations efficaces que nous verrons plus loin dans le chapitre 4 et le chapitre 5.

La définition 3.7 présente un enrichissement du formalisme des Frappes de Processus avec l'ajout des classes de priorité. Cet enrichissement a été proposé par Paulevé (2011) et formalisé par Folschette (2014). Les classes de priorités permettent en effet de modéliser les systèmes où les dynamiques des composants peuvent être regroupées en classes. Par exemple, les processus cellulaires ont une dynamique entre les protéines de signalisation qui se trouvent dans le cytoplasme différente de la dynamique des interactions entre les facteurs

de transcription et l'ADN dans le noyau. Modéliser ces différents types d'interaction peut se faire à l'aide des Frappes de Processus avec les classes de priorité.

**Définition 3.7** (Frappes de Processus avec $k$ classes de priorités)**.** Si $k \in \mathbb{N}^*$, les *Frappes de Processus avec k classes de priorités* sont définies par un triplet $\mathcal{PH} = (\Sigma\,;\mathcal{L}\,;\mathcal{H}^{\langle k\rangle})$, où $\mathcal{H}^{\langle k\rangle} = (\mathcal{H}^{(1)}\,;\ldots;\mathcal{H}^{(k)})$ est un $k$-uplet, et :

- $\Sigma \triangleq \{a, b, \ldots\}$ est l'ensemble fini et dénombrable des *sortes* ;
- $\mathcal{L} \triangleq \bigotimes_{a\in\Sigma} \mathcal{L}_a$ est l'ensemble fini des *états*, où $\mathcal{L}_a = \{a_0, \ldots, a_{l_a}\}$ est l'ensemble fini et dénombrable des *processus* de la sorte $a \in \Sigma$ et $l_a \in \mathbb{N}^*$. Chaque processus appartient à une unique sorte : $\forall (a_i\,;b_j) \in \mathcal{L}_a \times \mathcal{L}_b, a \neq b \Rightarrow a_i \neq b_j$ ;
- pour tout $n \in [\![1\,;k]\!]$, $\mathcal{H}^{(n)} \triangleq \{a_i \to b_j \not\curvearrowright b_l \mid (a\,;b) \in \Sigma^2 \wedge (a_i\,;b_j\,;b_l) \in \mathcal{L}_a \times \mathcal{L}_b \times \mathcal{L}_b \wedge b_j \neq b_l \wedge a = b \Rightarrow a_i = b_j\}$ est l'ensemble fini des *actions de priorité n*.

On note $\mathcal{H} \triangleq \bigcup_{n\in[\![1;k]\!]} \mathcal{H}^{(n)}$ l'ensemble de toutes les actions et, pour tout $n \in \mathbb{N}^*$ et $h \in \mathcal{H}^{(n)}$, $\mathsf{prio}(h) \triangleq n$.

On réutilise de surcroît les notations définies à la section 2.2.3.3 concernant les états et l'extraction de la sorte d'un processus.

À l'instar de la section 2.2.3.3, il faut définir un opérateur de jouabilité pour déterminer la dynamique des Frappes de Processus avec $k$ classes de priorités. Cependant, à l'inverse de celui des Frappes de Processus standards (définition 2.13) il faut ici prendre en compte la possible présence d'actions jouables appartenant à des classes de priorités supérieures. Pour cela, il est suffisant de vérifier que le frappeur et la cible de toute action de priorité plus importante ne sont pas simultanément présents. En effet, prenons deux actions $g, h \in \mathcal{H}$ avec : $\mathsf{prio}(g) < \mathsf{prio}(h)$, et un état $s \in \mathcal{L}$ tel que $\mathsf{frappeur}(g) \in s \wedge \mathsf{cible}(g) \in s$ ; Deux cas de figures sont alors possibles :

- l'action $g$ est jouable dans $s$ — autrement dit, aucune autre action de priorité plus importante ne la préempte – et elle préempte $h$ en conséquence,
- l'action $g$ n'est pas jouable dans $s$, ce qui signifie qu'elle est préemptée par une action de priorité plus importante, qui préempte alors aussi l'action $h$.

Dans les deux cas, $h$ n'est pas jouable, ce qui montre qu'il est suffisant de n'observer que la présence simultanée du frappeur et de la cible de chaque action de priorité supérieure pour déterminer la jouabilité de $h$. Nous obtenons alors l'opérateur de jouabilité donné à la définition 3.8.

**Définition 3.8** (Opérateur de jouabilité ($F_p : \mathcal{H} \to F$))**.** L'opérateur de jouabilité des Frappes de Processus avec $k$ classes de priorités est défini par :

$$\forall h \in \mathcal{H}, F_p(h) \equiv \mathsf{frappeur}(h) \wedge \mathsf{cible}(h) \wedge \left( \bigwedge_{\substack{g\in\mathcal{H}^{(n)} \\ n<\mathsf{prio}(h)}} \neg\left(\mathsf{frappeur}(g) \wedge \mathsf{cible}(g)\right) \right)$$

### 3.2.4.2 Traduction des motifs vers les réseaux d'automates

Nous présentons ici la traduction des motifs biologiques en modèles formels notamment les réseaux d'automates stochastiques et les Frappes de Processus. Le principe de cette traduction est de reconnaître un *motif biologique* et de générer le modèle formel qui

l'encode en réseau d'automates. Ainsi, pour chaque motif, biologique, un identifiant unique lui est associé. En fonction de l'identifiant associé, le générateur en réseau d'automates ou Frappes de Processus génère le code associé. Cette méthode à l'inconvénient de ne pas pouvoir traduire les motifs non appris à priori par le détecteur et le traducteur de motif. Cependant elle évite de recourir à une recherche combinatoire pour détecter le type de chaque motif. Pour les applications en modélisation des systèmes biologiques, il est préférable de choisir d'apprendre les nouveaux motifs au besoin sachant que cela se produit rarement. Dans la suite de cette section, nous donnons des exemples de transformation des motifs biologiques en modèles formels.

*Exemple.* Simple activation. Nous présentons dans cet exemple la traduction d'un motif qui représente une activation d'un composant $b$ par un composant $a$. Le motif est donné par :

$$M_1 = (b, \mathbf{Infld}(b) = \{a\}, \mathbf{PPCompf}^{\mathrm{b}}_{\mathcal{R}} = \{a(\mathcal{R}_1 \circ \mathcal{R}_3)b\})$$

L'équivalent en Réseau d'automates asynchrones est donné par : $\mathcal{SAN}_1 = (\Sigma_1, S_1, T_1)$ avec $\Sigma_1 = \{a, b\}$, et les états locaux et les transitions locales définies comme suit :

$$S_1(a) = \{a_0, a_1\}, \quad S_1(b) = \{b_0, b_1\},$$
$$T_1(b) = \{t_1 = b_0 \xrightarrow{a_1} b_1, t_2 = b_1 \xrightarrow{a_0} b_0\}$$

L'équivalent en Frappes de Processus est donné par : $PH_1 = (\Sigma_1, \mathcal{L}_1, \mathcal{H}_1)$ avec $\Sigma_1 = \{a, b\}$, et les processus et les actions définies comme suit :

$$\mathcal{L}_{1a} = \{a_0, a_1\}, \quad \mathcal{L}_{1b} = \{b_0, b_1\},$$
$$\mathcal{H}_1 = \{a_1 \to b_0 \Rsh b_1, a_0 \to b_1 \Rsh b_0\}$$

La figure 3.6 donne une représentation graphique de notre traduction.

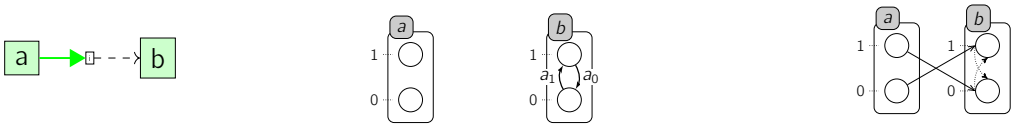



Figure 3.6 : **(Gauche) motif RSTC.** Les nœuds représentent les composants et les arcs les interactions. Dans ce motif, le composant $a$ active le composant $b$. **(Centre) Le motif équivalent en réseau d'automates asynchrones** avec deux automates : $a$ et $b$. **(Droite) Le motif équivalent en PH** avec deux sortes : $a$ et $b$.

*Exemple.* Simple inhibition. Nous présentons dans cet exemple la traduction d'un motif qui représente une inhibition d'un composant $b$ par un composant $a$. Le motif est donné par :

$$M_1 = (b, \mathbf{Infld}(b) = \{a\}, \mathbf{PPCompf}^{\mathrm{b}}_{\mathcal{R}} = \{a(\mathcal{R}_2 \circ \mathcal{R}_3)b\})$$

L'équivalent en réseau d'automates asynchrones est donné par : $\mathcal{SAN}_2 = (\Sigma_2, S_2, T_2)$

avec $\Sigma_2 = \{a, b\}$, et les états locaux et les transitions locales définies comme suit :

$$S_2(a) = \{a_0, a_1\}, \quad S_2(b) = \{b_0, b_1\},$$
$$T_2(b) = \{t_1 = b_0 \xrightarrow{a_1} b_1, t_2 = b_1 \xrightarrow{a_0} b_0\}$$

L'équivalent en Frappes de Processus est donné par : $PH_2 = (\Sigma_2, \mathcal{L}_2, \mathcal{H}_2)$ avec $\Sigma_2 = \{a, b\}$, et : les processus et les actions définies comme suit :

$$\mathcal{L}_{2a} = \{a_0, a_1\}, \quad \mathcal{L}_{1b} = \{b_0, b_1\},$$
$$\mathcal{H}_2 = \{a_1 \to b_1 \not\Uparrow b_0, a_0 \to b_0 \not\Uparrow b_1\}$$

La figure 3.7 donne une représentation graphique de notre traduction.

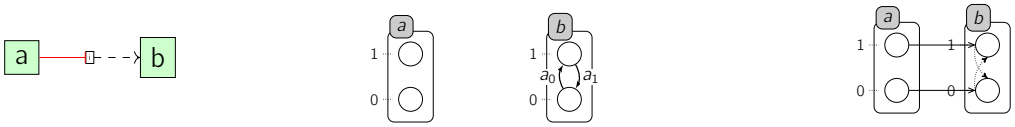



Figure 3.7 : **(Gauche) motif RSTC.** Les nœuds représentent les composants et les arcs les intercations. Dans ce motif le composant $a$ inhibe le composant $b$. **(Centre) Le motif équivalent en réseau d'automates asynchrones** avec deux automates : $a$ et $b$. **(Droite) Le motif équivalent en PH** avec deux sortes : $a$ et $b$.

*Exemple.* Activation & Inhibition. Nous présentons dans cet exemple la traduction d'un motif qui représente une activation d'un composant $c$ par un composant $a$ ou son inhibition par un composant $b$. Les deux influences sont indépendantes. Le motif est donné par :

$$M_3 = (c, \mathbf{Infld}(c) = \{a, b\}, \mathbf{PPCompf}^{\mathrm{b}}_{\mathcal{R}} = \{a(\mathcal{R}_1 \circ \mathcal{R}_3)b, a(\mathcal{R}_2 \circ \mathcal{R}_3)b\})$$

L'équivalent en réseau d'automates asynchrones est donné par : $\mathcal{SAN}_3 = (\Sigma_3, S_3, T_3)$ avec $\Sigma_3 = \{a, b, c\}$, et les états locaux et les transitions locales définies comme suit :

$$S_3(a) = \{a_0, a_1\}, \quad S_3(b) = \{b_0, b_1\}, \quad S_3(c) = \{c_0, c_1\},$$
$$T_3(b) = \{t_1 = c_0 \xrightarrow{a_1 \vee b_0} c_1, t_2 = c_1 \xrightarrow{a_0, b_1} c_0\}$$

L'équivalent en Frappes de Processus est donné par : $PH_3 = (\Sigma_3, \mathcal{L}_3, \mathcal{H}_3)$ avec $\Sigma_3 = \{a, b, c\}$, et : les processus et les actions définies comme suit :

$$\mathcal{L}_{3a} = \{a_0, a_1\}, \quad \mathcal{L}_{3b} = \{b_0, b_1\}, \quad \mathcal{L}_{3c} = \{c_0, c_1\},$$
$$\mathcal{H}_3 = \{a_1 \to c_0 \not\Uparrow c_1, a_0 \to c_1 \not\Uparrow c_0, b_0 \to c_0 \not\Uparrow c_1, b_1 \to c_1 \not\Uparrow c_0\}$$

La figure 3.8 donne une représentation graphique de notre traduction.

*Exemple.* Activation en synchronisation. Nous présentons dans cet exemple la traduction d'un motif qui représente une activation d'un composant $b$ par un composant $a$. Le motif est donné par :

$$M_4 = (c, \mathbf{Infld}(c) = \{a, b\}, \mathbf{PPCompf}^{\mathrm{b}}_{\mathcal{R}} = \{a(\mathcal{R}_1 \circ \mathcal{R}_3)b, a(\mathcal{R}_1 \circ \mathcal{R}_3)b\})$$

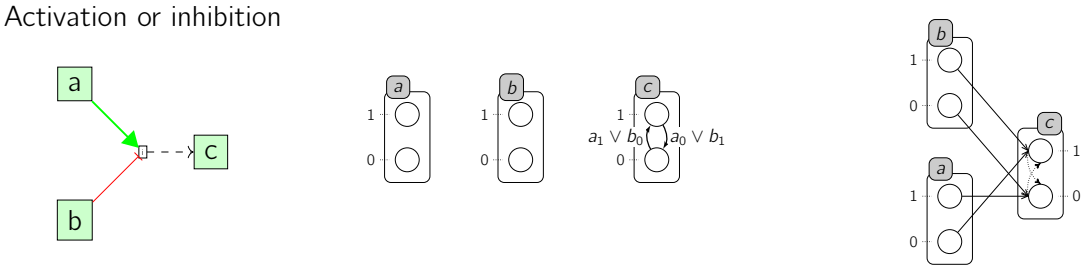



Figure 3.8 : **(Gauche) motif RSTC.** Les nœuds représentent les composants et les arcs les interactions. Dans ce motif les composants $a$ et $b$ active et inhibe le composant $c$. **(Centre) Le motif équivalent en réseau d'automates asynchrones** avec trois automates : $a$, $b$ et $c$. **(Droite) Le motif équivalent en PH** avec trois sortes : $a$, $b$ et $c$.

.
L'équivalent en Réseau d'automates asynchrones est donné par : $\mathcal{SAN}_4 = (\Sigma_4, S_4, T_4)$ avec $\Sigma_4 = \{a, b, c\}$, et les états locaux et les transitions locales définies comme suit :

$$S_4(a) = \{a_0, a_1\}, \quad S_4(b) = \{b_0, b_1\}, \quad S_4(c) = \{c_0, c_1\},$$
$$T_4(b) = \{t_1 = c_0 \xrightarrow{a_1 \vee b_1} c_1, t_2 = c_0 \xrightarrow{a_1 \vee b_0} c_1, t_3 = c_0 \xrightarrow{a_0 \vee b_1} c_1, t_4 = c_1 \xrightarrow{a_0, b_0} c_0\}$$

L'équivalent en Frappes de Processus est donné par : $PH_4 = (\Sigma_4, \mathcal{L}_4, \mathcal{H}_4)$ avec $\Sigma_4 = \{a, b, c, ab\}$, et : les processus et les actions définies comme suit :

$$\mathcal{L}_{4a} = \{a_0, a_1\}, \quad \mathcal{L}_{4b} = \{b_0, b_1\}, \quad \mathcal{L}_{4c} = \{c_0, c_1\}, \quad \mathcal{L}_{4ab} = \{ab_{00}, ab_{01}, ab_{10}, ab_{11}\}$$
$$\mathcal{H}_4 = \{a_1 \to ab_{00} \Rsh ab_{10}, b_1 \to ab_{00} \Rsh ab_{01}, \ldots, ab_{00} \to c_1 \Rsh c_0,$$
$$ab_{01} \to c_0 \Rsh c_1, ab_{10} \to c_0 \Rsh c_1, ab_{11} \to c_0 \Rsh c_1\}$$

La figure 3.9 donne une représentation graphique de notre traduction.

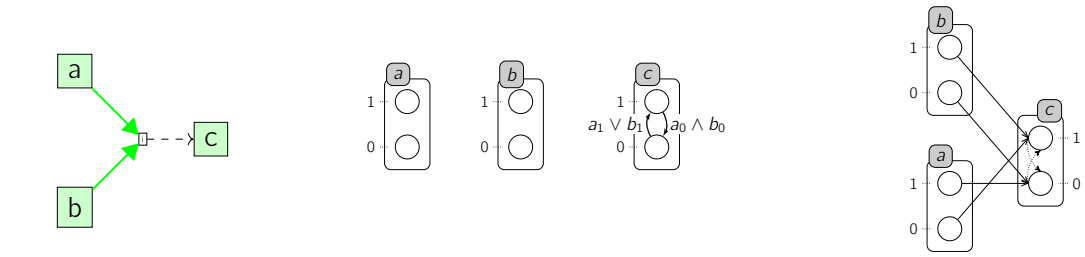



Figure 3.9 : **(Gauche) motif RSTC.** Les nœuds représentent les composants et les arcs les interactions. Dans ce motif, les composants $a$ et $b$ activent indépendemment le composant $c$. Par contre, il doivent être tous désactivés pour désactiver le composant $c$. Il faut noter ici que cette opération est différente d'une complexation. **(Centre) Le motif équivalent en réseau d'automates asynchrones** avec trois automates : $a$, $b$ et $c$. **(Droite) Le motif équivalent en PH** avec trois sortes : $a$, $b$, $c$ et une sorte coopérative $ab$.

### Utilisation des sortes de synchronisation

Nous avons vu à la section 2.2.3.3 en page 32 que les sortes coopératives permettent de modéliser la coopération entre deux composants qui doivent coopérer pour avoir une

influence sur un autre composant. Cette sorte coopérative permet de représenter l'état conjoint de plusieurs sortes dans le modèle. Pour cela, à chaque processus de la sorte coopérative correspond un sous-état des sortes qu'elle *représente*. Ainsi, il est possible de représenter les différents états combinés d'un ensemble de sortes, afin de ne jouer une action que dans une configuration particulière. De cette façon, la sorte coopérative permet un raffinement de la dynamique.

Cependant, nous avons des motifs où chaque composant (régulateur) a une influence indépendante sur un composant (régulé). C'est le cas par exemple dans la figure 3.9 où les composants $a$ et $b$ régulent positivement le composant $c$. En Frappes de Processus, si l'un des deux composants régulateurs ($a$ ou $b$) s'active (c'est-à-dire passe dans son processus $a_1$ ou $b_1$) et que l'autre composant reste inactif (c'est-à-dire reste dans son processus $a_0$ ou $b_0$), une compétition est créée entre les deux régulateurs. La compétition est due au fait que pendant que le composant actif active $c$, le composant inactif le désactive. Cette compétition crée au niveau de la dynamique du composant $c$ une *oscillation artificielle*. Une dynamique normale qu'on s'attendrait à avoir serait de voir le composant $c$ être activé par l'un de ses deux régulateurs ($a$ ou $b$), et rester actif tant que l'un de ses deux régulateurs est également actif. Ainsi, le composant $c$ devient inactif si tous ses deux régulateurs sont inactifs. Cet exemple illustre la nécessité d'effectuer une *synchronisation* pour certains motifs.

*La synchronisation* utilise des *sortes de synchronisation* qui sont des sortes comme les sortes coopératives. Elles permettent de synchroniser des évènements comme par exemple un composant qui ne peut être désactivé que si tous ses activateurs sont inactifs. Nous illustrons à la figure 3.10 le rôle d'une modélisation incluant la sorte de synchronisation. Cette figure montre que, dans le cas d'une modélisation qui n'utilise pas de sorte de synchronisation (à gauche de la figure), des oscillations sont produites. Par contre, dans le cas où le modèle utilise une sorte de synchronisation (à droite de la figure), il n'y a pas d'oscillations artificielles.

## 3.3 Intégration des séries temporelles

### 3.3.1 Les séries temporelles

Les séries temporelles, ou séries chronologiques correspondent à des observations régulièrement espacées dans le temps. Elles sont utilisées dans plusieurs domaines, parmi lesquels on peut citer l'astronomie, la météorologie, la théorie du signal, l'économie, la biologie, etc. Les séries temporelles étudient des données ou des observations d'évènements obtenues à un temps t spécifique, ou à des intervalles de temps au mieux identiques. L'une des particularité de ces séries temporelles est que la valeur de chaque observation est dépendante des valeurs qui la précèdent, il s'agit alors d'auto-corrélation. Dans la biologie des systèmes, les séries temporelles permettent d'obtenir des mesures sur l'évolution des quantités des composants biologiques au cours du temps. Elles permettent de mesurer l'expression d'un gène à travers la mesure de l'évolution de la quantité de protéines sécrétées au cours du temps. Ainsi, la dynamique des composants au cours du temps, nous apporte des renseignements sur le niveau d'activité du système et les niveaux discrets des composants. Il existe de plus en plus des données de séries temporelles disponibles pour la compréhension de la dynamique de certains systèmes biologiques. Nous allons montrer dans cette section comment il est possible de partir des données de séries temporelles d'expression des gènes

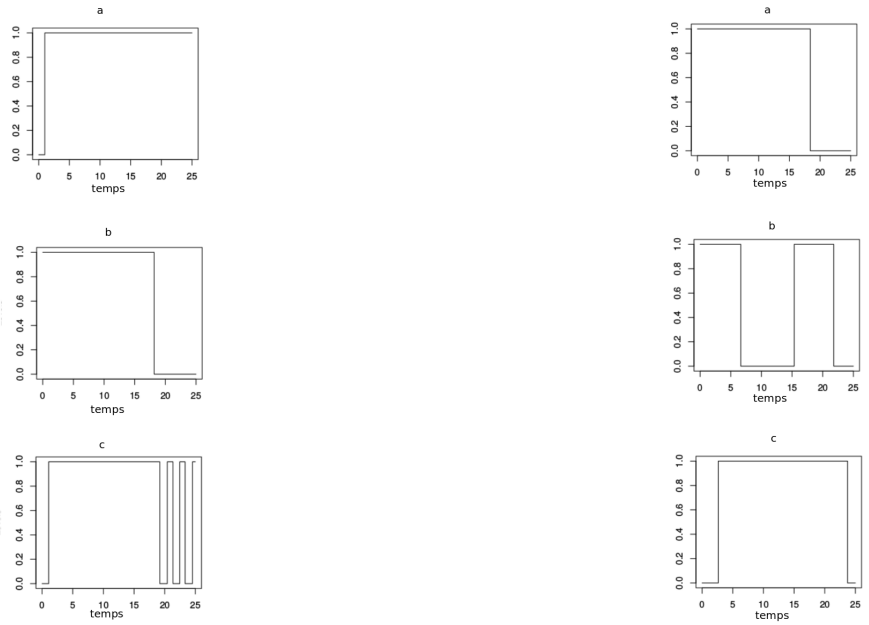



Figure 3.10 : Illustration de l'apport des sortes de synchronisation dans le raffinement de la dynamique des Frappes de processus. Le modèle étudié comporte trois composants $a$, $b$ (régulateurs) et $c$ (régulé). **(Gauche) la simulation sans sorte de synchronisation montre l'apparition des oscillations** chez le composant régulé ($c$) **(Droite) la simulation avec la sorte de synchronisation** montre que le composant $c$ n'est désactivé que quand à la fois $a$ et $b$ le sont aussi.

associées à un réseau type RSTC pour estimer les niveaux discrets des composants pour lesquels nous avons des séries temporelles et d'estimer les délais nécessaires pour observer les changements de niveau discret.

La figure 3.11 est un exemple de séries temporelles d'expression des gènes associé au processus de différenciation des cellules de la peau pour lequel nous proposons une application dans le chapitre des applications de cette thèse (voir section 6.2 en page 134). Ces données ont été obtenues de l'équipe du Professeur Peter Angel, du Centre Allemand de Recherche Contre le Cancer (DKFZ). On peut observer sur la figure que cette expérience est réalisée sur une période de $24h$ représentée par l'axe des abscisses. Les observations ont été prises à 10 instants différents. Les intervalles de temps ne sont pas réguliers et on observe qu'ils sont de plus en plus grands au fur et à mesure qu'on avance dans l'expérience. L'axe des ordonnées représentent les $\log_2(\frac{RNA}{RNA_{control}})$ de l'expression des composants qui ont pu être mesuré. Dans le cas de cette expérience, 12 gènes ont pu être mesuré ($MKP3$, $MKP1$, $UPAR$, $HES5$, $IL1B$, $A20$, $SM22$, $IL8$, $ET1$, $TNF_\alpha$, $TFR$ et $DKK1$). Le problème pour nous une fois que nous avons les mesures c'est de les traduire en comportement discret. Aussi nous allons donc être confronté au problème de la discrétisation des données des séries temporelles.

La première des choses va être de déterminer le nombre de niveaux discrets. Une fois le nombre de niveaux discrets déterminé, il va falloir choisir les seuils de discrétisation. Ces derniers peuvent être abstraits comme des délais pour passer d'un niveau discret à un autre. Déterminer le nombre de niveaux discrets d'un composant est une hypothèse de modélisation. Cette hypothèse est dans bien des cas inspirée par des expérimentations. C'est notamment le cas lorsque (Thomas, 1973) conjecture en observant la courbe d'expression

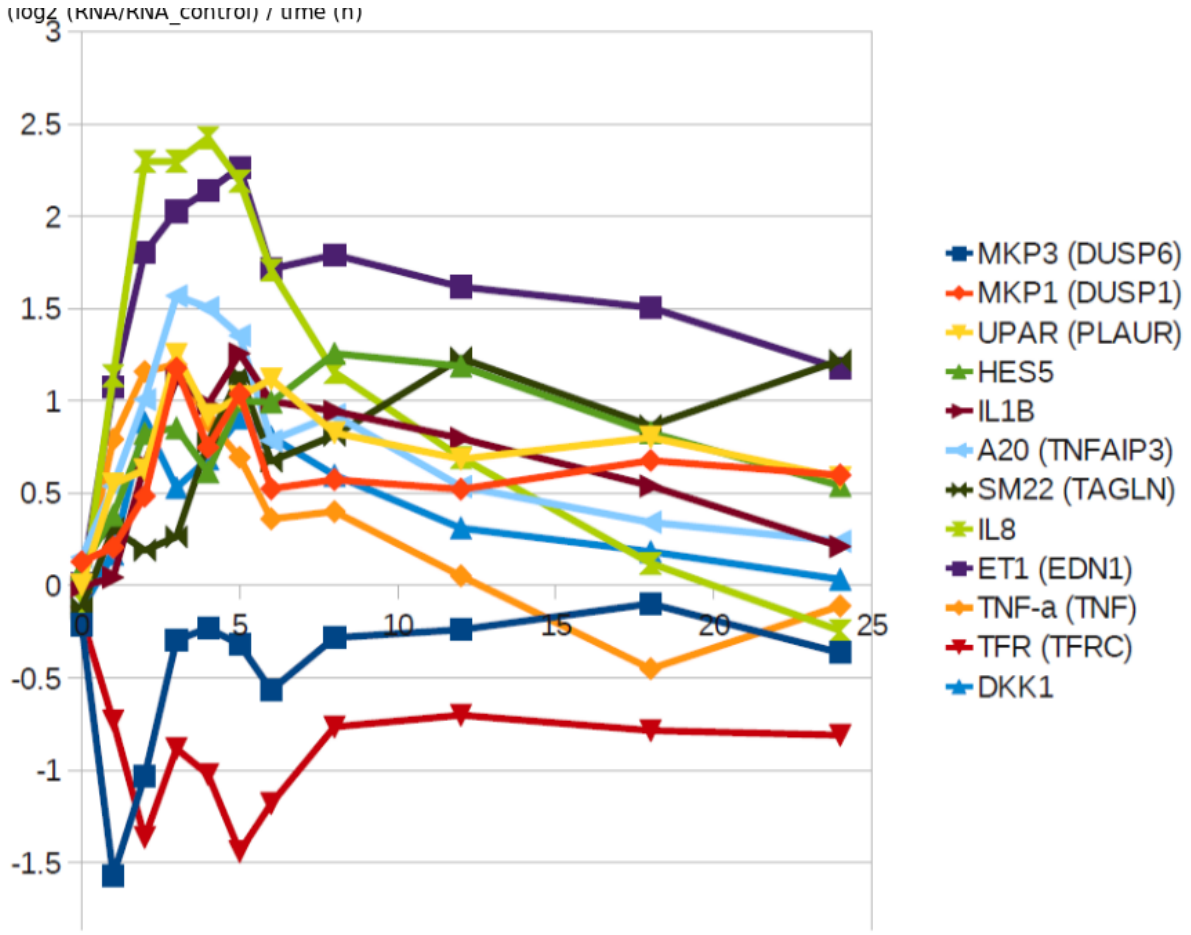



Figure 3.11 : **Données de séries temporelles.** Ces données représentent les mRNA différentiellement exprimés au cour d'une expérience où le nœud d'entré $E - cadherin$ du modèle de la différenciation des kératinocytes a été stimulé au calcium. Ces données sont observées sur $24h$ (axe des X) et elles représentent le $\log_2$ du niveau d'expression par rapport au contrôle (axe de Y).

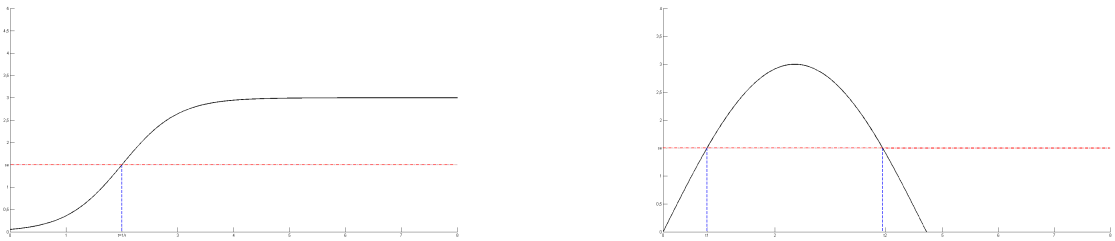


Figure 3.12 : Illustration de la notion de seuil, de délais et lien entre les deux notions

d'un gène représentée comme une sigmoïde (exemple figure 3.12) qu'en dessous d'un certain seuil, le gène n'a pas d'influence et qu'une fois ce seuil passé, le gène a une influence. Une telle observation conduit à déterminer deux niveaux discrets. Une observation fine de la figure 3.11 permet de constater que, pour la plupart des courbes, on observe deux pics d'activité. Le premier très fort et le second moins important que le premier. Pour confirmer cette observation, nous avons effectué un test statistique en utilisant le package R TTR

[2] qui permet de lisser les données de séries temporelles. En utilisant la fonction SMA avec le paramètre 2, nous avons plus de 50% des séries temporelles qui présentent deux seuils d'activités. Une façon de capturer ces deux niveaux d'activité dans la dynamique discrète est d'avoir deux seuils, de cette façon, les composants auront trois niveaux discrets.

Il est donc clair que les seuils permettent de fixer le nombre de niveaux discrets des composants. Ceci parce qu'ils fixent un niveau de quantité nécessaire pour avoir une certaine influence. De façon analogue, en faisant une projection du seuil sur l'axe du temps, on obtient une valeur qui est le délai nécessaire pour observer une quantité d'un composant donné.

Nous constatons donc à travers les deux cas mentionnés ci-dessus, que la dynamique des séries temporelles rend compte d'une réalité de la dynamique du système à travers la dynamique des composants mesurés. Cette réalité doit être prise en compte au mieux dans les choix de modélisation. La section suivante sera consacrée à montrer comment on peut partir des séries temporelles, avec les seuils et les délais et estimer les paramètres nécessaires pour raffiner la dynamique des automates asynchrones.

### 3.3.2 Raffinement de la dynamique dans les réseaux d'automates asynchrones

Nous présentons dans cette section une technique qui permet de partir des séries temporelles et d'estimer les paramètres temporels et stochastiques nécessaires pour une simulation stochastique des réseaux d'automates asynchrones. Dans le cadre du simulateur Pint (Paulevé, 2011), la simulation stochastique des Frappes de Processus se fait en utilisant une distribution d'Erlang pour le franchissement des transitions. Ce choix est différent de celui des simulateurs stochastiques usuels qui utilisent une distribution exponentielle pour les transitions. En effet la distribution exponentielle impose un fort lien entre la durée moyenne et la variance temporelle d'une transition. On peut noter par exemple que, plus la durée moyenne est élevée (c'est-à-dire le taux de franchissement est faible), plus la variance est forte. Fort de ce constat, (Paulevé, 2011) a fait le choix d'utiliser la distribution d'Erlang pour réduire la variance de la durée moyenne d'une action.

La distribution d'Erlang est généralement définie par deux paramètres : un paramètre de forme $k \in \mathbb{N}^*$ et un taux $\lambda \in \mathbb{R}_+^*$. La distribution d'Erlang est la distribution de la somme de $k$ variables aléatoires indépendantes suivant une distribution exponentielle de taux d'utilisation $\lambda$. Pour une raison de cohérence, nous reprenons la définition de la distribution d'Erlang telle que proposée par (Paulevé, 2011) comme étant la distribution de la somme de $sa$ variables aléatoires exponentielles suivant un taux d'utilisation $r.sa$, où $sa$ est l'absorption de stochasticité, et $r$ est le taux d'utilisation de la variable aléatoire exponentielle non-absorbée (c'est-à-dire sa=1). C'est une définition équivalente compte tenu des relations $k = sa$ et $\lambda = r.sa$. La densité de probabilité et la fonction de répartition d'une distribution d'Erlang suivant un taux $r$ et une absorption de stochasticité $sa$ sont définies par les équations 3.1 et 3.2, respectivement. Elles correspondent à la distribution

[2] `https://cran.r-project.org/web/packages/TTR/index.html`

d'Erlang usuelle avec une forme $sa$ et un taux $r.sa$ ((Evans et al., 2000))

$$f_{r,sa}(t) = \frac{(r.sa)^{sa} t^{sa-1} e^{-r.sa.t}}{(sa-1)!} \tag{3.1}$$

$$F_{r,sa}(t) = 1 - e^{-r.sa.t} \sum_{n=0}^{sa-1} \frac{(r.sa.t)^n}{n!} \tag{3.2}$$

Une transition qui suit une distribution d'Erlang requiert essentiellement deux paramètres : le taux que nous notons $r$ et l'absorption de stochasticité notée $sa$. Le taux est le paramètre qui permet d'estimer *le temps moyen* nécessaire pour franchir une transition. L'absorption de stochasticité est le paramètre qui permet de réduire la variance autour du temps moyen. En effet, la probabilité qu'une transition suivant un taux $r$ soit tirée dans un délai de $t$ unités de temps est donnée par $1 - e^{-r.t}$. La durée moyenne d'une telle transition est de $r^{-1}$ unités de temps avec une variance de $r^{-2}$. Étant données $x$ transitions, ayant respectivement les taux d'utilisation $r_1, r_2, \ldots, r_x$, la probabilité que la $y^e$ transition soit tirée est donnée par $\frac{r_y}{r_1+\ldots+r_x}$. Le simulateur Pint (Paulevé, 2011) prend également en compte le cas où nous avons des transitions instantanées, c'est-à-dire qui se joue immédiatement. Ces transitions ont un taux marqué comme étant infini $r_a = \infty$. Si deux transitions instantanées sont tirables, la transition à tirer est choisie de façon non-déterministe.

Après avoir présenté les paramètres nécessaires pour la simulation stochastique, nous allons maintenant montrer comment il est possible de partir des délais nécessaires pour observer les dépassements de seuil vers l'estimation du taux $r$ et l'absorption de stochasticité $sa$. Pour cela, supposons que nous avons une série temporelle représentant l'expression d'un composant $b$ du système régulé par un composant $a$. Le composant $b$ possède un ensemble de niveaux discrets. Modélisé avec les réseaux d'automates asynchrones (RAA) ou avec les Frappes de processus (PH), $b$ est donc respectivement appelé un automate ou une sorte et aura des états locaux (RAA) ou des des processus (PH) notés $b_0, b_1, \ldots, b_{n-1}$ pour n états discrets. Si nous faisons l'hypothèse simplificatrice qui consiste à considérer que le composant $a$ dans l'état local (processus) $a_j$, il a une influence positive sur $b$, donc tend à faire augmenter la quantité de $b$ et que de même, $a$ est dans l'état local (processus) $a_i$ il a une influence négative sur $b$, donc tend à faire diminuer la quantité de $b$, alors ces considérations se traduisent aisément au niveau de la dynamique par des transitions ou des actions suivantes (selon le cas) :

L'expression en Frappes de processus :
$a_j \to b_j \Rsh b_k \quad r_i @ sa,\ j < k$

L'expression en automates asynchrones :
$b_j \xrightarrow{a_j, r_i @ sa} b_k,\ j < k$

La question est donc de savoir comment nous obtenons les valeurs de $r$ et $sa$. Nous avons vu plus haut qu'avec un taux $r$ pour franchir une transition ou une action avec la loi d'Erlang, le temps moyen nécessaire pour franchir la transition ou l'action est égal à $\frac{1}{r}$. Ainsi si $t_i$ est le temps auquel on observe un dépassement de seuil, et $t_{i-1}$ le temps du précédent dépassement de seuil et $t_0$ l'instant initial, le taux $r_i$ de la transition ou de l'action qui permet un changement d'état local ou de processus au temps $t_i$ est donné par la formule suivante :

$$r_i = \frac{1}{t_i - t_{i-1}}$$

Le choix de la valeur de l'absorption de stochasticité dépend du niveau de raffinement que nous voulons avoir. En effet, la valeur de l'absorption de stochasticité détermine l'intervalle de tir de la transition ou de l'action. Ainsi, plus la valeur se rapproche de 1, plus

la variance autour du temps moyen est grande et donc l'intervalle de tir est aussi large. Par contre plus la valeur est grande, moins la variance autour du temps moyen est grande et donc l'intervalle de tir est plus petit.

Nous illustrons cette démarche par la figure 3.13. Nous considérons que le composant $b$ possède trois niveaux discrets (ce qui implique que sa courbe d'expression admet deux seuils de discrétisation) et le composant $a$ en possède deux. À chaque intersection entre la courbe d'expression et un seuil de discrétisation, nous avons la valeur correspondante sur l'axe du temps auquel une action doit être effectuée pour correspondre à un changement de niveau discret.

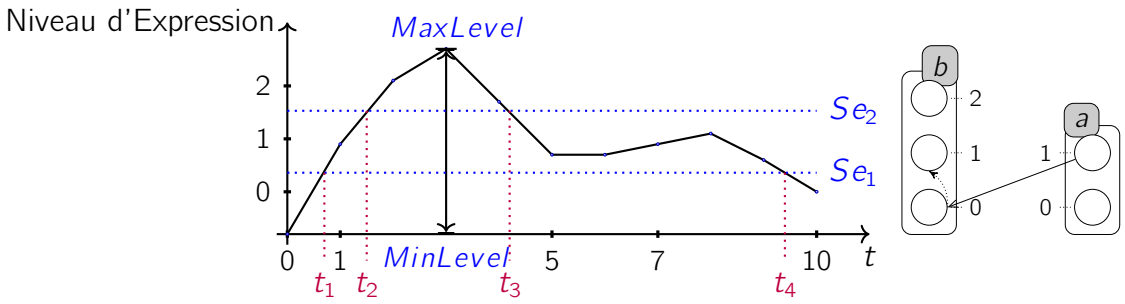



Figure 3.13 : **Estimation des paramètres temporels et stochastiques (taux et absorption de stochasticité) des séries temporelles :** Le taux pour effectuer une action qui permet un changement d'état est donné par $r_i = \frac{1}{t_i - t_{i-1}}$. $MaxLevel$ représente l'expression maximale alors que $MinLevel$ représente l'expression minimale du composant $b$ qui est régulé par le composant $a$. Les seuils $Se_1$ and $Se_2$ les niveaux discrets (états locaux ou processsus) (e.g. 0,1,2) d'un composant suivant sa courbe d'expression.

## 3.4 Évaluation par analyse statistique des traces

Dans cette section, nous présentons l'analyse statistique des traces (définition 3.9) issues des simulations. Cette démarche a pour but de pouvoir vérifier d'un point de vue quantitatif (statistiques sur les traces) la conformité des simulations avec les données de séries temporelles issues des expérimentations. En effet, la simulation stochastique et l'aspect concurrent des systèmes biologiques génèrent des traces différentes à chaque exécution du système. Ainsi, une façon de valider la dynamique d'un modèle consiste à analyser les traces générées par un ensemble de simulations et à déterminer pour chaque composant le pourcentage de traces correspondant à une trace de référence.

### 3.4.1 Définition de trace et de trace acceptante

Dans cette section, nous ne nous intéressons qu'à la dynamique d'un seul automate du système à la fois afin de déterminer quel a été son comportement pendant la simulation. Ainsi, à la suite de plusieurs simulations, nous pouvons déterminer un comportement fréquent. Plus spécifiquement, nous nous intéressons à un comportement-dit d'intérêt qui est représenté ici par la notion de trace acceptante (définition 3.10). Aussi, après la simulation de chaque composant $c_i$ du système génère, une trace $\mathsf{tr}(c_{ij})$ qui est associée au composant $c_i$. Dans la notation $\mathsf{tr}(c_{ij})$, $i$ est l'identifiant du composant et $j$ est l'identifiant de la simulation. Rappellons qu'un composant correspond à un automate fini stochastique.

D'après la définition 3.6 en page 60, chaque $c_i \in \Sigma$. Une trace pour chaque $c_i$ est une succession d'états locaux activés pendant la simulation. Ces états locaux sont activés en fonction des transitions locales franchissables dans $c_i$. La définition 3.9 formalise la notion de trace.

**Définition 3.9** (Trace $\mathsf{tr}(a)$ d'un automate $a$)**.** Soient un réseau d'automate $\mathcal{N} = (\Sigma, S, T)$ et un automate $a \in \Sigma$, la trace $\mathsf{tr}(a) = (a_0, a_1, \cdots, a_m)$ est la séquence des états locaux de l'automate $a$ ($a_i \in a$) obtenus après une simulation stochastique.

Par exemple, pour un composant $a$ ayant trois états locaux $\{a_0, a_1, a_2\}$ et les transitions locales $T(a) = \{t_1 = a_0 \xrightarrow{\emptyset} a_1, t_2 = a_1 \xrightarrow{\emptyset} a_0, t_2 = a_1 \xrightarrow{\emptyset} a_2\}$, un exemple de trace obtenue après la simulation de $a$ sachant que $a$ était initialement dans l'état $a_0$ peut être $(a_0, a_1, a_2)$. Il est aussi possible d'obtenir la trace suivante $(a_0, a_1, a_0, a_1, a_2)$.

Puisque les traces sont potentiellement différentes les unes des autres après chaque simulation et que notre objectif est de retrouver les traces qui correspondent à la courbe d'expression (discrète) du composant que nous observons, nous déterminons la trace acceptante à partir de la courbe d'expression du composant. En effet, pour les composants dont nous disposons d'une courbe d'expression, la trace de référence ou trace acceptante est obtenue en discrétisant la courbe d'expression du composant. Par exemple pour le composant $b$ de la figure 3.13 la trace acceptante est donnée par $(b_0, b_1, b_2, b_1, b_0)$.

**Définition 3.10** (Trace acceptante)**.** Une trace est acceptante pour un composant $a$ donné si elle correspond à la séquence d'états locaux obtenue après la discrétisation de la courbe d'expression du composant $a$.

### 3.4.2 Calcul des proportions de traces acceptantes

L'évaluation statistique des dynamiques des composants consiste à évaluer la proportion de traces acceptantes pour un ensemble de $N$ simulations. Notons par $w_{ij}$ la trace obtenue pour le composant $c_i$ après la simulation $j$, ce qui permet d'avoir $\mathsf{tr}(c_{ij}) = w_{ij}$. Notons également par $\mathcal{A}_{c_i}$ l'automate qui reconnaît les traces acceptantes pour le composant $c_i$. Nous avons :

$$\mathcal{A}_{c_i}(w_{ij}) = \begin{cases} 1 & \text{si } w_{ij}\text{est une trace acceptante pour} c_i \\ 0 & \text{sinon.} \end{cases}$$

Désignons également par $N_{c_i}$ le nombre de traces acceptantes pour le composant $c_i$ après $N$ simulations. La proportion $p_{c_i}$ de traces acceptantes pour le composant $c_i$ est alors donnée par :

$$p_{c_i} = \frac{N_{c_i}}{N}$$

.

la définition de trace que nous avons présentée à la section 3.4.1 peut s'avérer très restrictive au sens où la dynamique générée par un composant peut différer d'un ou d'au plus deux éléments dans la séquence composant la trace. Dans certains cas, les traces ne sont pas strictement équivalentes mais en faisant certaines considérations, il est tout à fait possible de les considérer comme telles. Par exemple, considérons la trace acceptante $tr_1 = (b_0, b_1, b_2, b_1, b_0)$ du composant $b$ de la figure 3.13. Il est tout à fait possible d'avoir une trace $tr_2 = (b_0, b_1, b_2, b_1, b_1)$ après une simulation. $tr_2$ diffère de $tr_1$ par le dernier composant de la séquence.

$$\begin{cases} tr_1 = & (b_0, b_1, b_2, b_1, b_0) \\ tr_2 = & (b_0, b_1, b_2, b_1, b_1) \end{cases}$$

En fonction des considérations qui sont faites lors d'une analyse, il est possible d'introduire des *tolérances*. Plusieurs critères peuvent être choisis pour ce faire. Un critère peut être par exemple le fait que la différence entre la trace acceptante et la trace obtenue soit au plus de 1. Cela revient théoriquement à utiliser une mesure de distance entre deux traces. Dans cette thèse, nous avons défini une tolérance dite de *type I*. Cette dernière admet qu'il y ait une différence entre la trace acceptante et la trace obtenue après simulation uniquement à la dernière position des deux séquences.

L'analyse statistique des traces définie dans cette section ne prend pas en compte l'aspect temporel et stochastique de la simulation. Une analyse complète consisterait à ajouter un intervalle de temps qui indique l'intervalle de temps dans lequel l'état local du composant devrait normalement être observé dans le cas de la trace acceptante et indique la période dans laquelle l'état local du composant est réellement observé dans le cas de la trace obtenue après simulation. De cette façon, la vérification ne se contente plus seulement de vérifier l'ordre dans lequel les états locaux sont apparus pendant la simulation, mais aussi à quels moments ils sont apparus. Avec cette nouvelle considération, la trace acceptante $tr_a$ et la trace réelle obtenue après simulation $tr_r$ pour un composant $b$ devrait ressembler à :

$$\begin{cases} tr_a = & (b_0\ [d_0, D_0], b_1\ [d_1, D_1], \cdots, b_n\ [d_n, D_n]) \\ tr_r = & (b_0\ t_0, b_1\ t_1, \cdots, b_n\ t_n) \end{cases}$$

Avec les intervalles $[d_i, D_i]$ qui représentent l'intervalle de temps dans lequel les $b_i$ devraient être activés. Les $t_i$ représentent, eux, l'instant où les $b_i$ ont été activés. Ainsi, une trace $tr_r$ est acceptée si la séquence des $b_i$ apparaît dans le bon ordre et si tous les $t_i \in [d_i, D_i]$.

## 3.5 Discussion

Dans ce chapitre, nous avons proposé une démarche pour modéliser et analyser plus finement la dynamique des grands réseaux de régulations biologiques. Cette démarche est structurée en trois phases. La première est la formalisation des réseaux de régulations biologiques comme des réseaux d'automates asynchrone ou des Frappes de Processus. La deuxième est le raffinement de la dynamique des modèles formels obtenus en intégrant les informations temporelles et stochastiques issues des séries temporelles. Enfin, la troisième phase est la simulation et l'analyse des modèles obtenus.

Pour la phase de formalisation, nous avons défini la notion de motif minimal dans le cadre de réseaux de régulations biologiques. Cette définition s'appuie sur la notion de relation en mathématique et l'opération de composition pour construire les motifs minimaux. Cette conception est mise en œuvre par un algorithme efficace qui permet de détecter les motifs minimaux dans les réseaux de régulations biologiques. Les motifs détectés sont ensuite traduits dans un modèle formel (réseau d'automates stochastiques ou Frappes de Processus). Toutefois, notre approche nécessite une connaissance à priori des motifs pour avoir une traduction en réseaux d'automates ou en Frappes de Processus. Il serait possible de traduire les motifs détectés vers d'autres formalismes (notamment le pi-calcul stochastique, les réseaux de Petri, etc.).

Le raffinement de la dynamique est acquis grâce à l'introduction des paramètres temporels et stochastiques (le taux et l'absorption de stochasticité). Le taux permet, comme nous l'avons vu, de définir le temps moyen nécessaire pour franchir une transition ou exécuter une action. L'absorption de stochasticité permet de réduire la variance. Avec ces deux paramètres, une spécification temporelle plus précise est possible dans le cadre des systèmes stochastiques. La grande innovation de notre approche est que ces paramètres sont estimés à partir des données de séries temporelles expérimentales qui représentent la dynamique des systèmes modélisés. Il est donc ainsi possible de prétendre modéliser et reproduire de façon assez précise la dynamique de grands (des centaines de composants) systèmes biologiques.

Une application sur un système biologique réel est présentée à la section 6.2 du chapitre 6. Il s'agit du processus de différenciation des cellules de la peau (les kératinocytes). Nous montrons dans cette application, comment il est possible de raffiner la dynamique des grands réseaux de régulation biologique. Ce raffinement est possible par des choix de modélisation structurels et l'estimation des paramètres temporels et stochastiques.

# Chapitre 4

# Analyse statique des propriétés quantitatives dans les réseaux d'automates stochastiques

Ce chapitre présente une méthode efficace pour la vérification quantitative des propriétés d'accessibilité dans les réseaux d'automates stochastiques. La méthode développée s'appuie sur l'analyse statique par interprétation abstraite des scénarios en séquences de transitions et en séquence d'objectifs (un objectif étant l'accessibilité d'un état local d'un automate depuis un autre état local du même automate) qui peuvent être résolus localement dans chaque automate. Partant de ces abstractions, la méthode dérive une structure qui prend la forme d'un graphe de causalité locale quantifié par des probabilités et des délais. De cette structure, il est possible de faire des estimations des probabilités et des délais de l'ensemble des scénarios et des scénarios critiques.
La construction du graphe de causalité locale quantifié se fait avec une complexité polynomiale selon le nombre d'automates et exponentielle selon le nombre d'états locaux par automate. Ce résultat permet ainsi d'éviter la construction du graphe d'états qui n'est pas possible pour les systèmes ayant un très grand nombre de composants.
Ce résultat est prometteur pour les systèmes biologiques et l'industrie pharmaceutique où la notion de chronométrie joue un rôle essentiel. Il peut notamment permettre d'estimer une borne inférieure de la probabilité ou du délai d'observer ou non un comportement après une influence externe (introduction d'un médicament par exemple).

## 4.1 Préliminaires

Dans les systèmes stochastiques, le temps et l'aléatoire peuvent être intégrés en imposant une durée aléatoire pour effectuer une transition. Il est courant que cette durée suive

une distribution exponentielle. Elle possède un unique paramètre appelé *taux* qui définit de manière informelle le nombre de fois qu'une transition peut être utilisée en l'espace d'une unité de temps. L'un des intérêts de ces systèmes est qu'ils permettent de modéliser et d'analyser plus finement les dynamiques des dits systèmes en prenant en compte la dimension aléatoire et temporelle inhérente à ces systèmes.

L'analyse des systèmes stochastiques peut se faire de façon efficace en les abstrayant comme des chaînes de Markov à temps continu (section 4.3 en page 82). Par une telle abstraction, il est alors possible en se basant sur les théories développées dans la cadre de l'étude des chaînes de Markov à temps continu, de connaître les propriétés stochastiques et temporelles des systèmes modélisés. Il est donc très facile de calculer par exemple la probabilité de partir d'un état du système pour atteindre un autre état, de connaître les distributions stationnaires (c'est-à-dire les probabilités à l'équilibre) du système. D'un point de vue des propriétés temporelles, il est possible d'avoir une estimation du temps passé dans un état, du temps de premier passage dans un état partant d'un autre état du système et bien d'autres propriétés encore.

Toutes ces propriétés sont des propriétés dites quantitatives en ce sens qu'elles permettent d'avoir des mesures sur le comportement du système. Elles sont d'un intérêt majeur pour de nombreux domaines d'applications. Nous pouvons citer par exemple la gestion des files d'attentes dans les banques et les centres commerciaux. Dans ce type de problème, on s'intéresse au temps moyen qu'un utilisateur passe dans la file, au temps moyen de service, etc. Dans les systèmes de production, on va par exemple s'intéresser au nombre moyen de produits défectueux sortant d'une chaîne de fabrication. Notre intérêt pour les systèmes stochastiques est qu'ils sont également d'une grande utilité pour les systèmes biologiques et par conséquent pour l'industrie pharmaceutique. Ils peuvent notamment permettre d'étudier les probabilités d'observer et/ou de ne pas observer certains comportements dans les systèmes biologiques. Une application dans l'industrie pharmaceutique pourrait par exemple consister à étudier d'un point de vue quantitatif l'effet d'une molécule sur un système biologique. On pourrait alors par exemple s'intéresser aux délais d'observer l'effet de la molécule sur le système biologique.

Du fait de la grande taille (des centaines de composants et d'interactions) des systèmes considérés, une analyse complète de la dynamique quantitative par les méthodes et outils de model-checking classique s'avère généralement impossible. En effet, ces problèmes sont connus pour être de complexité PSPACE-complet (Cheng, Esparza & Palsberg, 1995). Aussi le recourt à des méthodes alternatives s'avèrent plus que nécessaire.

L'analyse statique par interprétation abstraite (Cousot & Cousot, 1977) a pour but de fournir des analyses efficaces d'un modèle sans l'exécuter. Ceci est possible à travers une abstraction d'un modèle et de sa sémantique afin d'obtenir un système abstrait dont le comportement est plus simple à interpréter (comprendre et analyser) que le système concret. Soit une propriété sur la dynamique du modèle dont la validité doit être vérifiée, les méthodes par interprétation abstraite produisent des approximations supérieure (sup) et inférieure (inf) de cette validité. Ces approximations peuvent se révéler non concluantes parfois. Une approximation supérieure (sup) permet de garantir la non-validité de la propriété, tandis qu'une approximation inférieure (inf) garantit sa validité.

Les analyses statiques ont été mises en œuvre dans des travaux récents pour des systèmes mobiles (Feret, 2005), des systèmes séquentiels (Bouissou, 2008) et des systèmes biologiques à travers le langage $\kappa$ (Danos et al., 2008). Plus récemment (Feret et al., 2012) ont proposé une analyse statique des probabilités en réduisant l'espace des états grâce à la construction des classes d'équivalences de ces états.

(Paulevé et al., 2012) ont introduit la vérification de l'accessibilité dans les réseaux d'automates à travers une interprétation abstraite des comportements concurrents des réseaux d'automates. À partir de la spécification du réseau d'automates, ils calculent des représentations abstraites de l'ensemble des comportements concernés par la propriété d'accessibilité recherchée. Ces représentations prennent la forme de graphes appelés Graphes de Causalité Locale (GLC). Les abstractions construites oublient délibérément une partie de l'information sur l'ordre ou l'arité des transitions locales, résultant ainsi en des approximations supérieures et inférieures des comportements du modèle concret.

Afin de répondre efficacement au problème de l'analyse de propriétés quantitatives dans les réseaux d'automates stochastiques, nous prenons donc avantage des abstractions introduites par (Paulevé et al., 2012) dans le cadre de l'analyse des propriétés qualitatives d'accessibilité dans les réseaux d'automates asynchrones, pour proposer une analyse statique efficace des propriétés quantitatives d'accessibilité dans les réseaux d'automates stochastiques. L'approche est basée sur deux abstractions complémentaires des scénarios (succession d'applications des transitions) des réseaux d'automates stochastiques : en séquences d'objectifs et en séquences de transitions. Un objectif est l'accessibilité d'un état local depuis un autre état local du même automate. Une séquence d'objectifs peut représenter la spécification d'une propriété d'accessibilité. Notre principale contribution est de donner une mesure approchée de la probabilité et des délais de réalisation de la propriété d'accessibilité. Par le développement de raisonnements récursifs et itératifs, nous définissons les approximations des quantités requises.

Ce travail a de nombreuses applications. Par exemple, dans l'industrie pharmaceutique ou pour l'étude des systèmes biologiques, il peut contribuer à estimer une borne inférieure de la probabilité d'observer un comportement à la suite de l'administration d'un médicament quand le calcul de la probabilité exacte s'avère inaccessible par les méthodes classiques. Outre l'industrie pharmaceutique et l'étude des systèmes biologiques, ce travail peut également trouver de nombreuses applications dans la vérification des systèmes industriels.

Ce chapitre est structuré comme suit. La section 4.2 introduit quelques définitions notamment la définition du problème d'accessibilité dans les réseaux d'automates et définit les réseaux d'automates. Dans cette même section, nous présentons l'idée de l'analyse statique que nous avons développée pour l'analyse des propriétés quantitatives. La section 4.3 introduit quelques notions pour effectuer les analyses probabilistes et stochastiques dans les réseaux d'automates stochastiques. La section 4.4 propose une interprétation quantitative de l'abstraction des scénarios. La section 4.5 présente les structures abstraites (graphe de causalité locale et graphe de causalité locale quantifié) pour une vérification efficace des propriétés (quantitatives) d'accessibilité. Cette même section présente également le calcul des bornes inférieures des probabilités et des délais d'accessibilité. Enfin, la section 4.6 résume et discute les contributions de ce chapitre.

## 4.2 Définitions préliminaires

L'objectif de cette section est de définir le problème de l'*accessibilité* dans les réseaux d'automates stochastiques (aussi appelée *atteignabilité*) et de présenter l'idée et les contributions de ce chapitre. Nous y présentons notamment la démarche pour construire les approximations quantitatives de la propriété l'accessibilité, en particulier l'estimation d'une borne inférieure de la probabilité et du délai d'observer ou non un comportement.

### 4.2.1 Définition du problème d'accessibilité

Le problème de l'accessibilité dans les Réseaux d'Automates consiste à rechercher l'existence d'un scénario qui permette d'activer un ou plusieurs états locaux donné(s). Il peut se résumer à la question suivante : « Étant donné un état initial, existe-t-il un scénario partant de cet état et qui permette d'activer un état local donné ? » ou, de façon plus générale pour un ensemble d'états locaux : « Étant donné un contexte/état initial $\varsigma$ et une séquence d'états locaux $\omega$ donnés, existe-t-il un scénario $\delta$ jouable dans le contexte $\varsigma$ et permettant d'atteindre successivement chacun des états locaux de $\omega$ dans l'ordre du scénario ? »

Dans une configuration où les transitions sont jouées avec des paramètres quantitatifs (taux d'occurences, probabilité), le problème d'accessibilité ne se résume plus seulement à déterminer étant donné un contexte/état initial, s'il existe un scénario partant de cet état initial et qui permette d'atteindre un état local donné. Il consiste maintenant non seulement à dire s'il existe un scénario, mais aussi à évaluer la probabilité et/ou les délais d'observer ce scénario. Aussi, le problème d'accessibilité dans les réseaux d'automates stochastiques peut parfois consister à évaluer la probabilité et/ou les délais de tous les scénarios jouables dans un contexte/état initial qui permettent d'activer un objectif. Nous pouvons donc le formuler comme suit : « Etant donné un état initial $\varsigma$ et une séquence d'états locaux $\omega$ donnés, quelle est la probabilité d'activer chacun des états locaux de $\omega$ partant des scénarios jouables dans $\varsigma$ : $\mathcal{P}(\mathbf{Sce}(\varsigma \rightarrow^* \omega))$ ? Quel est le délai nécessaire pour activer chacun des états locaux de $\omega$ partant des scénarios jouables dans le contexte $\varsigma$ : $\mathcal{T}(\mathbf{Sce}(\varsigma \rightarrow^* \omega))$ ? »

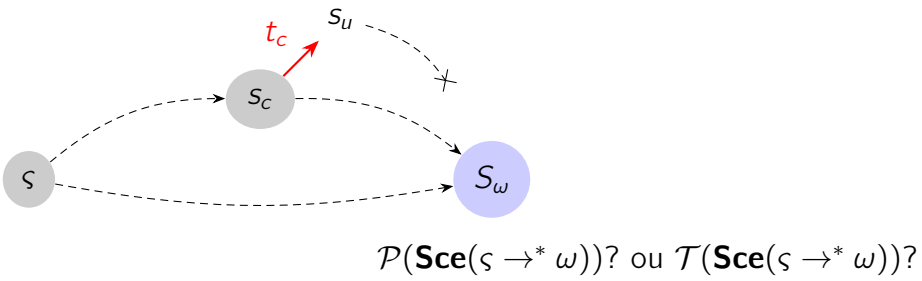



Figure 4.1 : Illustration générale du problème d'accessibilité quantitative. $\varsigma$ est l'état ou le contexte initial, $S_\omega$ est l'ensemble d'états qui contiennent le but état local à atteindre. Les flèches en tirets représentent une séquence (potentiellement vide) de transitions. La flèche rouge en trait plein est une transition concurrente d'un état global $s_c$ à $s_u$, et $t_c$ est la transition locale associée.

On rappelle que **Sce** désigne l'ensemble des scénarios tel que cela va être défini dans la section 4.4 en page 85. Ce problème d'accessibilité peut parfois être résolu à l'aide des outils de *model-checking* classiques. Cependant, de telles méthodes reposent généralement sur l'analyse de la dynamique complète du modèle. Pour de grands modèles, ces méthodes se heurtent donc à l'explosion combinatoire inhérente au calcul du graphe des états. La figure 4.1 illustre le problème d'accessibilité d'un point de vue quantitatif. La différence avec l'accessibilité qualitative est qu'ici c'est la probabilité ou le délai d'atteindre un état partant d'un contexte/état initial qui nous intéresse.

### 4.2.2 Définition d'un réseau d'automates stochastiques ($\mathcal{SAN}$)

Nous rappelons dans cette section la définition d'un réseau d'automates stochastiques que nous avons déjà introduit au chapitre 3. Un réseau d'automates stochastiques (définition 4.1) est formé par un ensemble d'automates finis où chaque transition au sein d'un automate est associée à un libellé. Un libellé peut être partagé entre plusieurs transitions d'automates différents. Ce libellé ($\ell$) définit alors une pré-condition sur l'état global du réseau : tout état partageant le libellé doit être dans l'état permettant la transition associée. Lorsqu'une telle condition est satisfaite, chaque automate change d'état en suivant les transitions associées ; l'état global du réseau contient alors tous les états locaux des automates résultants des transitions associées à $\ell$, référencés dans la post-condition. Lorsque plusieurs pré-conditions sont satisfaites par un même état global, seul un libellé est choisi, de manière non déterministe. Le délai moyen pour franchir une transition est donné par l'inverse du taux ($r$) d'utilisation de la transition.

**Définition 4.1** (Réseau d'Automates Stochastiques)**.** Un *Réseau d'Automates Stochastiques* est défini par le tuple $\mathcal{SAN} = (\Sigma, S, T)$ où

- $\Sigma$ est l'ensemble fini d'automates ;
- Pour chaque $a \in \Sigma$, $S(a) = \{a_0, \ldots, a_{k_a}\}$ est l'ensemble fini des états locaux de l'automate $a$ ; $S \triangleq \prod_{a\in\Sigma} S(a)$ est l'ensemble fini des états globaux ;
  **LS** $\triangleq \bigcup_{a\in\Sigma} S(a)$ est l'ensemble de tous les états locaux.
- $T = \{a \mapsto T_a \mid a \in \Sigma\}$, où $\forall a \in \Sigma, T_a \subseteq S(a) \times 2^{\mathrm{LS}\setminus S(a)} \times \mathbb{R} \times S(a)$ avec $(a_i, \ell, r, a_j) \in T_a \Rightarrow a_i \neq a_j$ et $\forall b \in \Sigma, |\ell \cap S(b)| \leq 1$, est une correspondance des automates vers l'ensemble des transitions locales. $r$ est le taux section 3.3.2 en page 68 de la transition.

Nous écrivons $a_i \xrightarrow{\ell,r} a_j \in T \triangleq (a_i, \ell, r, a_j) \in T(a)$.

La définition 4.2 formalise la sémantique et la relation binaire de transition globale entre deux états globaux d'un réseau d'automates stochastiques.

**Définition 4.2** (Sémantique des réseaux d'automates stochastiques)**.** $a_i \to a_j \in T \triangleq \exists \ell \in 2^{\mathrm{LS}\setminus S(a)}, a_i \xrightarrow{\ell} a_j \in T$. Etant donné une transition $t = a_i \xrightarrow{\ell} a_j \in T$, nous notons $\mathrm{orig}(t) \triangleq a_i$, et $\mathrm{dest}(t) \triangleq a_j$, $\mathrm{enab}(t) \triangleq \ell$, ${}^\bullet t \triangleq \{a_i\} \cup \ell$, et $t^\bullet \triangleq \{a_j\} \cup \ell$. La relation globale de transition $\to \subseteq S \times S$ est définie par :

$$s \to s' \triangleq \exists \ell \in \; : \forall a_i \in {}^\bullet\ell, s(a) = a_i \wedge \forall a_j \in \ell^\bullet, s'(a) = a_j \\ \wedge \forall b \in \Sigma, S(b) \cap {}^\bullet\ell = \emptyset \Rightarrow s(b) = s'(b).$$

Cette définition des réseaux d'automates diffère de la définition déterministe généralement utilisée dans les travaux sur les réseaux booléens ou multivalués (Richard, 2010) où chaque automate $a$ d'un réseau est défini par une fonction déterministe $f^a$ qui associe à l'état global du réseau l'état suivant de l'automate $a$. Elle diffère également de la définition introduite par (Plateau & Atif, 1991) pour la modélisation des systèmes parallèles parce qu'elle est centrée sur les transitions. Dans la définition non-déterministe et stochastique utilisée dans cet article, l'évolution de chaque automate est définie par un ensemble de transitions munies de pré-conditions sur l'état global du réseau. Il est ainsi possible d'avoir deux transitions avec la même pré-condition mais des post-conditions différentes.

La figure 4.2 à la page suivante représente un réseau d'automates stochastiques $\mathcal{SAN} =$

$(\Sigma, S, T)$ de 3 automates ($\Sigma = \{a, b, c\}$), avec $S(a) = \{a_0, a_1\}$, $S(b) = \{b_0, b_1\}$, $S(c) = \{c_0, c_1\}$, et 6 transitions locales définies comme suit :

$$T(a) = \{t_1 = a_0 \xrightarrow{\emptyset,2} a_1, t_2 = a_1 \xrightarrow{c_1,1} a_0\}$$
$$T(b) = \{t_3 = b_0 \xrightarrow{a_1,2} b_1, t_4 = b_1 \xrightarrow{a_0,1} b_0\}$$
$$T(c) = \{t_5 = c_0 \xrightarrow{b_1,2} c_1, t_6 = c_1 \xrightarrow{b_0,1} c_0\}$$

Chaque transition possède un libellé et un taux. Par exemple la transition $t_3 = b_0 \xrightarrow{a_1,2} b_1$ a comme libellé $a_1$ et pour taux 2. Pour rappel le taux désigne le nombre de fois par unité de temps avec lequel la transition est franchie quand la pré-condition est vérifiée.

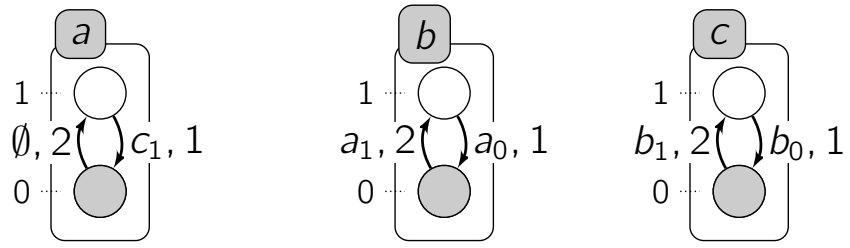



Figure 4.2 : Un exemple de réseaux d'automates stochastiques. Les automates sont représentés par les boîtes labélisées, les états locaux sont représentés par des cercles. Les états locaux sont associés à leur identifiant grâce aux pointillés. Par exemple, l'état local $a_0$ est le cercle identifié 0 dans la boîte $a$. Une transition est un arc orienté entre deux états locaux du même automate. Les transitions peuvent être labélisées par un ensemble d'états locaux des autres automates et des taux de transition. Enfin, les états locaux en gris représentent l'état global du système à un instant donné. Ici, l'état global du système est $\langle a_0, b_0, c_0 \rangle$.

### 4.2.3 Approche pour la construction de notre analyse statique

Soit un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$ pour lequel nous voulons vérifier une propriété d'accessibilité $\mathcal{R}$ d'un point de vue quantitatif $\mathcal{Q}(\mathcal{R})$ telle que définie à la section 4.2.1 en page 78. Ici $\mathcal{Q}(\mathcal{R})$ peut être soit la probabilité $\mathcal{P}(\mathcal{R})$, soit le délai $\mathcal{T}(\mathcal{R})$ nécessaire pour que propriété d'accessibilité soit réalisée. Nous présentons dans ce chapitre une approche qui permet d'obtenir une estimation efficace d'un intervalle contenant la probabilité et le délai recherchés. Cette approche est basée sur la construction d'une structure abstraite $\mathcal{G}$ proche de celles introduites par (Paulevé et al., 2012) pour la vérification efficace des propriétés d'accessibilité d'un point de vue qualitatif. La structure que nous proposons a la particularité de contenir à chaque nœud une quantité qui est soit une probabilité, soit un délai. De la structure $\mathcal{G}$, nous pouvons efficacement calculer les bornes inférieures et supérieures de $\mathcal{Q}(\mathcal{R})$ qui renseignent sur la probabilité et le délai de l'accessibilité recherchée.

Dans sa thèse, Paulevé (2011) a établi des propriétés $\mathcal{CN}$ et $\mathcal{CS}$ permettant respectivement l'approximation supérieure (sup.) et inférieure (inf.) de la propriété d'accessibilité qualitative pour les réseaux d'automates non-déterministes.

De façon intuitive : Si $\mathcal{SAN}$ n'a pas la propriété $\mathcal{CN}$, alors l'accessibilité de $\mathcal{R}$ est impossible (approximation sup. de $\mathcal{R}$).

Si $\mathcal{SAN}$ a la propriété $\mathcal{CS}$, alors l'accessibilité de $\mathcal{R}$ est possible (approximation inf. de $\mathcal{R}$).

Les propriétés $\mathcal{CN}$ et $\mathcal{CS}$ sont respectivement des conditions nécessaires et des conditions suffisantes de l'accessibilité recherchée $\mathcal{R}$. Elles se vérifient efficacement de manière statique sur $\mathcal{SAN}$.

Dans cette thèse, nous proposons de partir de la propriété $\mathcal{CS}$ (condition suffisante) et de proposer un encadrement de $\mathcal{Q}(\mathcal{R})$ par le calcul d'une borne inférieure $\mathsf{binf}_{\mathcal{Q}(\mathcal{R})}$ et d'une borne supérieure $\mathsf{bsup}_{\mathcal{Q}(\mathcal{R})}$.

Ainsi, dans le cas de $\mathcal{P}(\mathcal{R})$, cette démarche permet d'établir une borne inférieure $\mathsf{binf}_{\mathcal{P}(\mathcal{R})}$ et supérieure $\mathsf{bsup}_{\mathcal{P}(\mathcal{R})}$ à la probabilité d'accessibilité de $\mathcal{R}$. De plus, il est possible de dériver de la structure $\mathcal{G}$ le scénario le plus ou le moins probable.

Dans le cas de $\mathcal{T}(\mathcal{R})$, cette démarche permet d'établir une borne inférieure $\mathsf{binf}_{\mathcal{T}(\mathcal{R})}$ et une borne supérieure $\mathsf{bsup}_{\mathcal{T}(\mathcal{R})}$ au délai d'accessibilité. Il est aussi possible de conclure sur le scénario le plus ou le moins rapide par un parcours spécifique de $\mathcal{G}$.

Dans la formule 4.1, nous présentons comment les valeurs que nous calculons (dans le cas particulier des probabilités) se positionnent par rapport à la probabilité réelle de la propriété d'accessibilité.

$$0 \leq \mathsf{binf}_{\mathcal{P}(\mathcal{R})} \leq \mathcal{P}(\mathcal{R}) \leq \mathsf{bsup}_{\mathcal{P}(\mathcal{R})} \leq 1 \tag{4.1}$$

La construction et la taille de la structure $\mathcal{G}$ s'avère d'une complexité limitée. La structure $\mathcal{G}$ est dans sa mise en œuvre un graphe mettant en relation quatre types de nœuds : les *états locaux*, les *objectifs* (par exemple $a_i \rightsquigarrow a_j$) spécifiant l'accessibilité d'un état local $a_j$ depuis un état local initial $a_i$, les *solutions* ou pré-conditions décrivant un ensemble d'états locaux nécessaires à l'accessibilité de certains états locaux voulus et enfin les solutions simultanées qui correspondent à un ensemble de pré-conditions, qui sont des ensembles d'états locaux dont la présence simultanée est requise pour pouvoir appliquer la transition à l'état global du réseau. Aux nœuds, nous associons les quantités qui sont selon le cas, des probabilités ou des délais. Ces quantités associées aux relations entre les nœuds du graphe sont la principale avancée par rapport à la structure abstraite telle que introduite par (Paulevé et al., 2012) et enrichie par (Folschette, Paulevé, Magnin & Roux, 2015). Le traitement de notre structure $\mathcal{G}$ permet de répondre avec efficacité au problème d'accessibilité $\mathcal{R}$.

La structure $\mathcal{G}$ est obtenue à travers une construction récursive : partant des états locaux dont l'accessibilité est demandée par $\mathcal{R}$, les états locaux sont liés à l'objectif les atteignant depuis l'état initial (imposé par $\mathcal{R}$) et en y associant les probabilités ou les délais nécessaires ; ensuite, toutes les solutions d'un objectif sont calculées : une solution contient l'ensemble minimal des processus permettant, localement, de résoudre l'objectif.

Il faut noter que cette démarche ne construit pas les sous-graphes générés par les *objectifs concurrents* mais nous en tenons compte en prenant une majoration de leur influence sur la dynamique globale. Un objectif $a_i \rightsquigarrow a_k$ est dit concurrent à un objectif $a_i \rightsquigarrow a_j$ dit d'intérêt si $k \neq j$. Cependant, les objectifs concurrents ont en effet un impact sur la probabilité ou le délai de réaliser l'objectif d'intérêt. Ils sont en général des concurrents de l'objectif d'intérêt et réduisent la probabilité ou augmentent le délai de le voir réaliser. Nous prenons en compte cet impact lors du calcul de la probabilité de réaliser l'objectif (proposition 4.1 en page 87). De façon succincte, si nous calculons $\mathsf{binf}_{\mathcal{P}(\mathcal{R})}$, nous considérons que les objectifs concurrents vont certainement être réalisés. Cette hypothèse évite de parcourir le sous arbre induit par ces objectifs mais réduit en contrepartie

la probabilité $\mathcal{P}(\mathcal{R})$ pour avoir une estimation inférieure $\mathsf{binf}_{\mathcal{P}(\mathcal{R})}$. De même, nous pouvons faire l'hypothèse (irréaliste) qu'ils ne vont jamais être réalisés. Dans ce cas nous avons une sur-estimation $\mathsf{bsup}_{\mathcal{P}(\mathcal{R})}$ de $\mathcal{P}(\mathcal{R})$. Nous construisons un raisonnement analogue pour l'estimation des délais.

## 4.3 Une sémantique probabiliste pour la dynamique des réseaux d'automates stochastiques

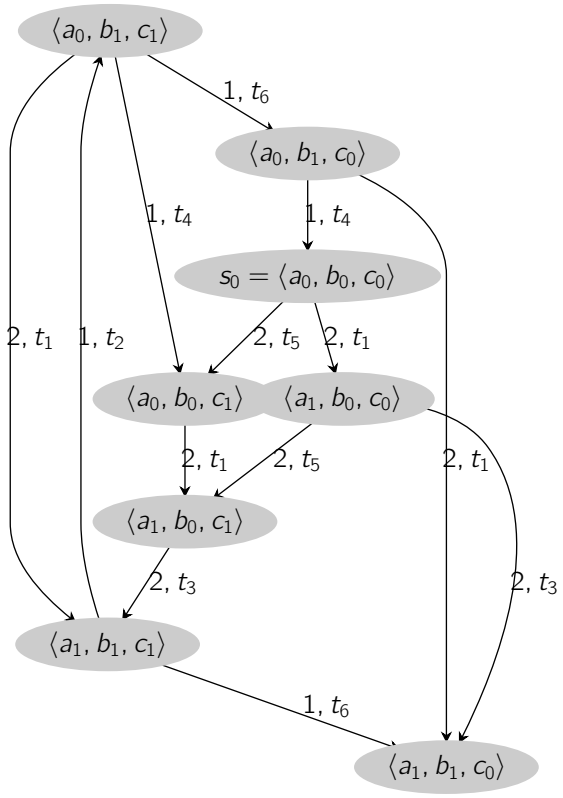



Figure 4.3 : Le graphe de transitions associé au réseau de la figure 4.2 avec pour état initial $s_0 = \langle a_0, b_0, c_0\rangle$. Les transitions sont labélisées avec leur vitesse respective

Dans cette section, nous introduisons les éléments de base nécessaires à l'analyse de la dynamique quantitative dans les réseaux d'automates stochastiques. Nous commençons par préciser la notion de *taux* qui correspond au nombre de fois par unité de temps qu'une transition est franchie. Puis nous définissons la notion d'activité d'un état global ou d'un état local. Ensuite, nous définissons le temps moyen d'activité et tous les éléments nécessaires à la compréhension de la dynamique dans les réseaux d'automates stochastiques.

Le temps et l'aléatoire sont assimilés en imposant une durée aléatoire pour effectuer une transition. Il est courant que cette durée suive une distribution exponentielle. Elle possède un unique paramètre appelé *taux*. Le *taux* $r$ d'une transition $(a_i, \ell, r, a_j) \in T(a)$ est le nombre moyen de fois que cette transition est franchie par unité de temps. C'est une valeur qui permet d'introduire un paramétrage quantitatif (probabilités/temps) sur les transitions d'un réseau d'automates stochastiques.

Une fois ce paramétrage quantitatif introduit, nous pouvons voir le réseau d'automates stochastiques comme une chaîne de Markov à temps continu à laquelle on peut associer

une matrice de taux que nous notons ici $\mathbf{R}$ et définie comme suit $\mathbf{R} : S \times S \to \mathbb{R}_{\geq 0}$.

Les questions que nous nous posons sont donc de savoir avec quelle probabilité ou quel délai on passe d'un état à un autre ou d'un groupe d'états à un autre groupe d'états. Nous introduisons donc la définition 4.3 qui caractérise les taux de transition entre états. Pour la suite de ce travail, nous faisons l'hypothèse que les transitions sont franchies suivant une distribution exponentielle.

**Définition 4.3** (Taux de transitions entre états)**.** Soit un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$, pour tous sous ensembles $A, B \subseteq S$.

- Le taux de transition d'un état $s \in S$ vers un sous ensemble d'états $A$ est défini par : $\mathbf{R}(s, A) := \sum_{s' \in A} \mathbf{R}(s, s')$. Où $\mathbf{R}(s, s')$ est le taux de transition de l'état $s$ à l'état $s'$.
- Le taux de transition entre deux sous ensembles d'états $A, B$ est défini par : $\mathbf{R}(B, A) := \sum_{s \in B} \mathbf{R}(s, A)$.
- Le taux de sortie d'un état $s \in S$ est noté $E(s)$ et est défini par $E(s) := \mathbf{R}(s, S)$.

De la définition 4.3, nous constatons que les taux régissent les transitions entre états ou entre sous-ensembles d'états. Dans la figure 4.3, nous observons que le taux de transition de l'état $\langle a_0, b_0, c_0 \rangle$ vers l'état $\langle a_1, b_0, c_0 \rangle$ est 2. De même, la transition de l'état $\langle a_1, b_1, c_1 \rangle$ vers l'état $\langle a_1, b_1, c_0 \rangle$ se fait avec un taux 1.

S'il n'est pas possible de sortir d'un état $s$, alors cet état est absorbant et $E(s) = 0$. Dans le cas où $E(s)$ est différent de 0, alors l'état est dit transitoire (définition 4.4).

**Définition 4.4** (État absorbant/transitoire)**.** Soit un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$, $\mathbf{R} : S \times S \to \mathbb{R}_{\geq 0}$ la matrice des taux. Un état $s \in S$ est dit absorbant si $E(s) = 0$ et transitoire sinon.

*Remarque.* Si $R(s, s') > 0$, on dit qu'une transition est possible de $s$ vers $s'$.

Après avoir défini le taux et donné une idée intuitive de la relation qui peut exister entre le taux et les transitions, nous donnons maintenant le lien qui existe entre les taux de transition et les probabilités de transition sachant que nous faisons l'hypothèse que les transitions sont franchies suivant une distribution exponentielle. Ainsi, étant donné un état $s$, la probabilité qu'une transition sera tirée depuis l'état $s$ est donnée par $1 - e^{-\lambda t}$ (définition 4.5). Ici, $\lambda$ est la somme des taux des transitions partant de $s$ vers n'importe quel autre état.

**Définition 4.5** (Probabilité de transition $\mathcal{P}_{(s,*)}(t)$)**.** Soit un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$, $\mathbf{R} : S \times S \to \mathbb{R}_{\geq 0}$ la matrice des taux. Soit un état $s \in S$, si $s$ est l'état courant, la probabilité qu'une transition soit franchie depuis l'état $s$ dans un délai de temps $t$ est donnée par : $\mathcal{P}_{(s,*)}(t) = 1 - e^{-\lambda t}$. Où $\lambda = \sum_{r \in R_s} r$ et $R_s = \{r : \exists (a_i, \ell, r, a_j) \in T(a) \wedge a_i \in s\}$.

Par exemple, la probabilité qu'une transition soit franchie depuis l'état $s_0 = \langle a_0, b_0, c_0 \rangle$ de la figure 4.3 dans un délai de temps $t = 2$ est donnée par :

$$\mathcal{P}_{(s_0,*)}(2) = 1 - e^{-(2t_1 + 2t_5) * 2} = 0.9996$$

Suite à la définition 4.5, nous introduisons la définition 4.6 à la page suivante qui donne la probabilité d'observer une transition depuis un état courant $s$ vers un autre état $s'$ dans un délai de temps $t$. Sachant que la définition 4.5 donne déjà la probabilité de sortir d'un état $s$, il suffit d'y rajouter une proportion qui indique combien de fois cette sortie sera

vers l'état $s'$ ce qui est donné par $\frac{R(s,s')}{E(s)}$. La définition 4.6 formalise la probabilité de jouer une transition d'un état $s$ vers un état $s'$.

**Définition 4.6** (Probabilité de jouer une transition d'un état $s$ vers un état $s'$ : $\mathcal{P}_{(s,s')}(t)$)**.** Soit un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$, $\mathbf{R} : S \times S \to \mathbb{R}_{\geq 0}$ la matrice des taux. Soient deux états $s, s' \in S$, si $s$ est l'état courant, la probabilité qu'une transition soit jouée de $s$ vers $s'$ dans un délai de temps $t$ est donnée par : $\mathcal{P}_{(s,s')}(t) = \frac{R(s,s')}{E(s)} \cdot \mathcal{P}_{(s,*)}(t)$.

Comme illustration, le calcul de la probabilité de partir de l'état $s_0 = \langle a_0, b_0, c_0 \rangle$ à l'état $s_1 = \langle a_1, b_0, c_0 \rangle$ en 2 unités de temps s'effectue comme suit :

$$\mathcal{P}_{(s_0,s_1)}(2) = \frac{2_{t_1}}{2_{t_1} + 2_{t_5}} \cdot \mathcal{P}_{(s_0,*)}(2) \tag{4.2}$$

$$\mathcal{P}_{(s_0,s_1)}(2) = 0.4998 \tag{4.3}$$

En faisant l'abstraction qui consiste à regarder un sous-ensemble d'états comme un seul état, nous pouvons étendre la définition 4.6 pour définir la probabilité d'observer une transition d'un état $s$ vers un sous-ensemble d'états $A$.

**Définition 4.7** (Probabilité de jouer une transition d'un état $s$ vers un sous-ensemble d'états $A$ : $\mathcal{P}_{(s,A)}(t)$)**.** Soit un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$, $\mathbf{R} : S \times S \to \mathbb{R}_{\geq 0}$ la matrice des taux. Soit un état $s \in S$ et un sous-ensemble d'états $A \subset S$, si $s$ est l'état courant, la probabilité qu'une transition soit jouée de $s$ vers un état $s' \in A$ dans un délai de temps $t$ est donnée par : $\mathcal{P}_{(s,A)}(t) = \frac{R(s,A)}{E(s)} \cdot \mathcal{P}_{(s,*)}(t)$.
De façon analogue, nous pouvons écrire que la probabilité qu'une transition soit jouée d'un sous ensemble d'états $A$ vers un état $s$ est donnée par :
$\mathcal{P}_{(A,s)}(t) = \frac{R(A,s)}{E(A)} \cdot \mathcal{P}_{(A,*)}(t)$.

En guise d'illustration, intéressons-nous à la probabilité d'effectuer une transition de l'état $s_3 = \langle a_0, b_1, c_1 \rangle$ en 2 unités de temps vers le sous-ensemble d'états constitué des états :

$A_1 = \{\langle a_1, b_1, c_1 \rangle, \langle a_0, b_0, c_1 \rangle\}$.

Par la définition 4.3 et la définition 4.7, cette probabilité est calculée comme suit :

$$\mathcal{P}_{(s_3,A_1)}(2) = \frac{R(s_3, A_1)}{E(s_3)} \cdot \mathcal{P}_{(s_3,*)}(2) \tag{4.4}$$

$$\mathcal{P}_{(s_3,A_1)}(2) = \frac{2_{t_1} + 1_{t_4}}{2_{t_1} + 1_{t_4} + 1_{t_6}} \cdot \mathcal{P}_{(s_3,*)}(2) \tag{4.5}$$

$$\mathcal{P}_{(s_3,A_1)}(2) = 0.7497 \tag{4.6}$$

La définition 4.7 est celle sur laquelle nous nous appuyons pour calculer la probabilité de passer d'un sous-ensemble d'états vers un autre sous-ensemble d'états. Elle est notamment utilisée pour le calcul de la probabilité de réaliser un objectif que nous présentons plus tard à la proposition 4.1.

Plus généralement et conformément à (Gao et al., 2013; Aziz, Sanwal, Singhal & Brayton, 2000), si nous supposons avoir une distribution initiale $\alpha$ sur un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$, le vecteur des probabilités de transition au temps $t$, noté par $\pi^{\mathcal{SAN}}(\alpha, t)$ est la distribution des probabilités à travers les états au temps $t$. La définition 4.8 à la page ci-contre formalise son expression.

**Définition 4.8** (Probabilités de transition)**.** Soit un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$, $\mathbf{R} : S \times S \to \mathbb{R}_{\geq 0}$ la matrice des taux. On définit les probabilités de transition sur $\mathcal{SAN}$ comme suit : partant d'une distribution initiale $\alpha$ sur $\mathcal{SAN}$, le vecteur des probabilités de transition au temps $t$, noté par $\pi^{\mathcal{SAN}}(\alpha, t)$, est la distribution de probabilité sur les états au temps $t$. A $t = 0$, $\pi^{\mathcal{SAN}}(\alpha, 0)(s') = \alpha(s')$. Pour $t > 0$, la probabilité de transition (Stewart, 1994) est donné par : $\pi^{\mathcal{SAN}}(\alpha, t) = \pi^{\mathcal{SAN}}(\alpha, 0)e^{Qt}$ où $Q := \mathbf{R} - diag(E)$ est la matrice génératrice $diag(E)(s, s) = E(s)$.

En plus des probabilités de transition entre états, nous nous intéressons également aux délais. Le calcul des délais de transition entre états est fortement lié au temps moyen passé dans un état. La définition 4.9 définit le temps d'activité en moyenne d'un état local, et plus généralement d'un état dans les réseaux d'automates stochastiques. Le temps d'activité d'un état correspond à la durée pendant laquelle cet état est actif.

**Définition 4.9** (Le temps d'activité moyen $\rho$)**.** Soit $\mathcal{SAN} = (\Sigma, S, T)$, un réseau d'automates stochastiques, soient $a_i \in \mathbf{LS}$ un état local appartenant à l'ensemble des états locaux (**LS**) et $s \in S$ un état global. Le temps d'activité moyen d'un état $s$ respectivement d'un état local $a_i$, est donné par $\rho(s) = \frac{1}{E(s)}$ respectivement $\rho(a_i) = \frac{1}{E(a_i)}$

L'analyse des propriétés quantitatives dans un système d'états de transition s'avère très compliqué en particulier quand le graphe des états est très grand. En effet, il n'est plus raisonnable de représenter le système comme une chaîne de Markov à temps continu (CTMC) et d'appliquer les propriétés que nous avons énoncées plus haut pour le calcul des probabilités et des délais. Face à cette difficulté, nous proposons de recourir à une interprétation abstraite des scénarios dans de tels systèmes. L'idée est de construire un raisonnement causal local, qui permet d'abstraire les scénarios en séquences de transitions et en séquences de transitions abstraites et de résoudre le problème d'accessibilité en ne se concentrant que sur la portion du réseau qui est directement impliquée dans la résolution du problème d'accessibilité. La prochaine section présente l'abstraction des scénarios d'un point de vue quantitatif c'est-à-dire en prenant en compte les taux des transitions entre les états. Cette démarche diffère de celle introduite par (Paulevé, 2011) parce qu'elle prend en compte les quantités (taux des transitions).

## 4.4 Interprétation Quantitative de l'Abstraction des Scénarios

Cette section montre comment les scénarios dans un réseau d'automates stochastiques sont abstraits en séquences de transitions (définition 4.15 en page 89) et en séquences d'objectifs (définition 4.11 en page 88). Une séquence d'objectifs décrit une succession de changements d'états locaux par automates (appelés objectifs), tandis que la séquence de transitions détaille les actions à appliquer pour la résolution de ces objectifs. Partant de ces abstractions, nous montrons comment il est possible de quantifier la « *concrétisabilité* » d'un scénario. Cela se fait par le calcul de la probabilité ou du délai de réaliser les abstractions (séquences d'objectifs ou séquences de transitions abstraites) de ces scénarios. Nous nous appuyons sur la figure 4.4 pour illustrer quelques notions que nous présentons dans cette section.

La figure 4.4 à la page suivante représente un réseau d'automates stochastiques $\mathcal{SAN} = (\Sigma, S, T)$ de 6 automates ($\Sigma = \{a, b, c, d, e, f\}$), avec $S(a) = \{a_0, a_1\}$, $S(b) = \{b_0, b_1\}$,

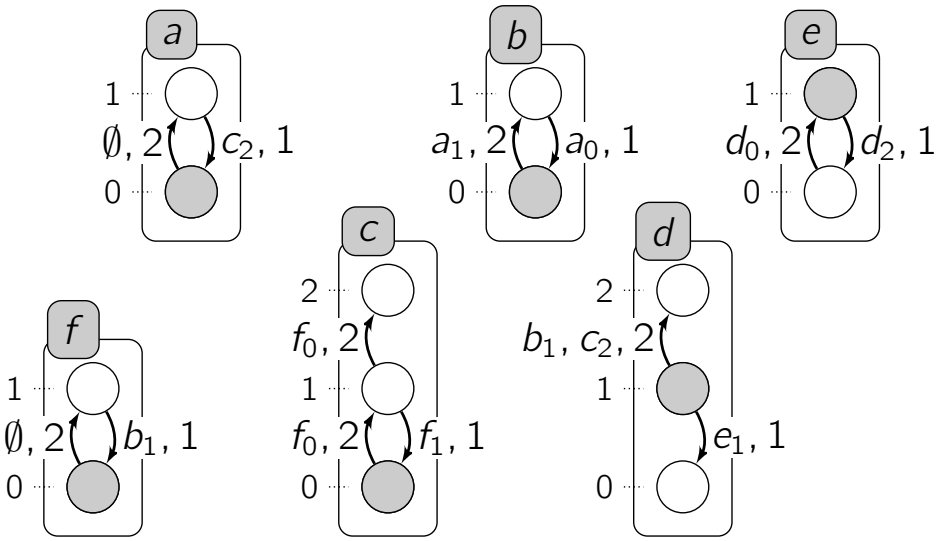



Figure 4.4 : Un exemple de réseaux d'automates stochastiques. Les automates sont représentés par les boîtes labélisées, les états locaux sont représentés par des cercles. Les états locaux sont associés à leur identifiant grâce aux pointillés. Par exemple, l'état local $a_0$ est le cercle identifié 0 dans la boîte $a$. Une transition est un arc orienté entre deux états locaux du même automate. Les transitions peuvent être labélisées par un ensemble d'états locaux des autres automates et des taux de transition. Enfin, les états locaux en gris représentent l'état global du système. $\langle a_0, b_0, c_0, d_1, e_1, f_0 \rangle$.

$S(c) = \{c_0, c_1, c_2\}$, $S(d) = \{d_0, d_1, d_2\}$, $S(e) = \{e_0, e_1\}$, $S(f) = \{f_0, f_1\}$ et 6 transitions locales définies comme suit :

$$
\begin{aligned}
T(a) &= \{t_1 = a_0 \xrightarrow{\emptyset,2} a_1, t_2 = a_1 \xrightarrow{c_2,1} a_0\} \\
T(b) &= \{t_3 = b_0 \xrightarrow{a_1,2} b_1, t_4 = b_1 \xrightarrow{a_0,1} b_0\} \\
T(c) &= \{t_5 = c_0 \xrightarrow{f_0,2} c_1, t_6 = c_1 \xrightarrow{f_0,2} c_2, t_7 = c_1 \xrightarrow{f_1,1} c_0\} \\
T(d) &= \{t_8 = d_1 \xrightarrow{e_1,1} d_0, t_9 = d_1 \xrightarrow{b_1,c_2,2} d_2\} \\
T(e) &= \{t_{10} = e_0 \xrightarrow{d_0,2} e_1, t_{11} = e_1 \xrightarrow{d_2,1} e_0\} \\
T(f) &= \{t_{12} = f_0 \xrightarrow{\emptyset,2} f_1, t_{13} = f_1 \xrightarrow{b_1,1} f_0\}
\end{aligned}
$$

### 4.4.1 Définitions & propriétés préliminaires

L'accessibilité d'un état local $a_j$ d'un automate $a$ donné depuis un autre état local $a_i$ du même automate est le fait, depuis un état où $a_i$ est actif, de pouvoir jouer un scénario menant dans un état où $a_j$ est actif. La question de l'existence d'un tel scénario possède naturellement un intérêt particulier dans la résolution d'une accessibilité locale ; c'est pourquoi on la représente sous la forme d'un *objectif*, noté $O = a_i \rightsquigarrow a_j$ (définition 4.10). De plus, on appelle *séquence d'objectifs* toute séquence dans laquelle la cible de chaque objectif est égale au bond de l'objectif précédent de la même sorte dans la séquence, s'il existe (définition 4.11).

**Définition 4.10** (Objectif (**Obj**))**.** Si $a \in \Sigma$, l'accessibilité d'un état local $a_j$ depuis un état local $a_i$ est appelé un *objectif*, noté $a_i \rightsquigarrow a_j$. L'ensemble de tous les objectifs est noté $\mathbf{Obj} \triangleq \{a_i \rightsquigarrow a_j \mid a \in \Sigma \wedge (a_i, a_j) \in S(a) \times S(a)\}$. Pour tout objectif $O = a_i \rightsquigarrow a_j \in \mathbf{Obj}$, on note $\mathsf{automate}(O) \triangleq a$ l'automate dans lequel l'objectif $O$ se réalise, $\mathsf{cible}(O) \triangleq a_i$ sa cible et $\mathsf{bond}(O) \triangleq a_j$ son bond. Enfin, $O$ est dit *trivial* si $a_i = a_j$.

La réalisation d'un objectif peut être quantifiée en ce sens qu'il est possible de calculer la probabilité ou le délai nécessaire d'accéder à un état local $a_j$ depuis un état local $a_i$. Dans le graphe des états, cela sa traduit par le passage d'un sous-ensemble des états contenant l'état $a_i$, à un sous-ensemble d'états contenant l'état $a_j$. La définition 4.7 permet de calculer les probabilités de passer d'un sous-ensemble d'états à un autre sous-ensemble d'états. Le calcul de cette probabilité nécessite de connaître uniquement les transitions qui partent du sous-ensemble d'états contenant $a_i$ au sous-ensemble d'états contenant l'état $a_j$. Retrouver ces transitions consiste à effectuer une recherche dans la liste des transitions telles que la source de la transition corresponde à $a_i$ et la destination à $a_j$. Les sous-ensembles des états sont une abstraction de l'ensemble des états du système en ensembles quotients obtenus en construisant des classes d'équivalence de la même façon que (Feret et al., 2012). Aussi, la proposition 4.1 donne la probabilité de réaliser un objectif.

**Proposition 4.1** (Probabilité de réaliser un objectif ($\mathcal{P}_t(a_i \rightsquigarrow a_j)$))**.** *La probabilité de réaliser un objectif $a_i \rightsquigarrow a_j$ dans un délai de temps $t$ est donné par $\mathcal{P}_t(a_i \rightsquigarrow a_j) = \mathcal{P}_{(A_{a_i}, A_{a_j})}(t)$ où $A_{a_i}$ (resp. $A_{a_j}$) représente l'ensemble des états contenant l'état local $a_i$ (resp. $a_j$).*

*Démonstration.* La preuve de la proposition 4.1 est directe par la définition 4.7. En effet, atteindre un objectif $a_j$ depuis un autre objectif $a_i$ consiste à transiter d'un sous-ensemble d'états contenant $a_i$ à un autre sous-ensemble d'états contenant $a_j$. La probabilité d'effectuer cette transition se calcule en se servant de la définition 4.7. □

Le calcul des délais de réalisation d'un objectif est plus sujet à discussion. Fondamentalement, un délai est crucial quand il peut être donné en terme de délai au plus tôt, ou de délai au plus tard, même si certaines fois les délais moyens peuvent avoir du sens. La proposition 4.2 propose le calcul du délai minimum nécessaire pour réaliser un objectif. En effet, un objectif est réalisé si au moins une transition qui connecte la cible et le bond est franchie.

**Proposition 4.2** (Délai pour la réalisation d'un objectif ($\mathcal{T}_t(a_i \rightsquigarrow a_j)$))**.** *Le délais minimum (au plus tôt) nécessaire pour réaliser un objectif $a_i \rightsquigarrow a_j$ est donné par $\mathcal{T}_t(a_i \rightsquigarrow a_j) = \min\{1/r : \exists (a_i, \ell, r, a_j) \in T(a)\}$.*

Dans la figure 4.4, si on s'intéresse à la probabilité d'accéder à l'état local $d_2$ depuis l'état local $d_1$. Cela revient à calculer la probabilité de réaliser l'objectif $d_1 \rightsquigarrow d_2$. Pour ce faire, il faut pouvoir calculer la probabilité de passer du sous-ensemble d'états contenant l'état local $d_1$ au sous-ensemble des états contenant l'état local $d_2$. Cependant, il faut noter que la transition $t_9 = d_1 \xrightarrow{b_1, c_2, 2} d_2$ est conditionnée par la présence à la fois des états locaux $b_1$ et $c_1$. Ce qui implique que les états du sous-ensemble d'états contenant l'état local $d_1$, doivent aussi contenir les états locaux $b_1$ et $c_2$. La proposition 4.1 prend aussi en compte l'objectif concurrent $d_1 \rightsquigarrow d_0$ qui peut contribuer à ce que l'objectif $d_1 \rightsquigarrow d_2$ ne se réalise pas ou à réduire sa probabilité de se réaliser. Cette prise en compte est effective

par la définition 4.7 qui dans son expression prend en compte au niveau du taux de sortie l'impact des transitions concurrentes.

**Définition 4.11** (Séquence d'objectifs (**Obj**))**.** Une *séquence d'objectifs* est une séquence $\omega = P_1 :: \ldots :: P_{|\omega|}$, où $\forall n \in \mathbb{I}^\omega, \omega_n \in \mathbf{Obj}$ et $\mathsf{cible}(\omega_n) = a_i \Rightarrow \mathsf{dernier}_a(\omega_{1\ldots n-1}) \in \{\varnothing, a_i\}$. L'ensemble des séquences d'objectifs est référé par **OS**. Les définitions de $\mathsf{premier}_a$, $\mathsf{dernier}_a$, $\mathsf{support}$ et $\mathsf{fin}$ (définition 2.9 en page 29) sont étendues aux séquences d'objectifs en omettant de spécifier le cas des frappeurs.

**Proposition 4.3** (Probabilité de réaliser une séquence d'objectifs indépendants (**Obj**))**.** *La probabilité de réaliser une séquence d'objectifs $\omega$ est donnée par :* $\mathcal{P}(\omega) = \prod_{i=1}^{|\omega|} \mathcal{P}(\omega_i)$.

La définition 4.12 introduit la notion de *contexte* qui étend celle d'état afin de pouvoir représenter un ensemble d'états initiaux possibles : plutôt que d'attribuer un seul état local actif à chaque automate, comme pour un état, un contexte permet d'en attribuer plusieurs. La notion de recouvrement, précédemment définie sur un état (définition 2.11 en page 30) est étendue au cas d'un contexte dans la définition 4.13. Il permettra à la définition 4.27 en page 96 de saturer le contexte initial d'analyse avec des états locaux supplémentaires.

**Définition 4.12** (Contexte (**Ctx**))**.** Un *contexte* $\varsigma$ associe à chaque automate dans $S$ un sous-ensemble non vide de ses états locaux : $\forall a \in \Sigma, \varsigma[a] \subseteq S(a) \wedge \varsigma[a] \neq \emptyset$. On note **Ctx** l'ensemble de tous les contextes.

**Définition 4.13** (Recouvrement ($\Cap : \mathbf{Ctx} \times \wp(\mathbf{LS}) \to \mathbf{Ctx}$))**.** Pour tout contexte $\varsigma \in \mathbf{Ctx}$ et tout ensemble d'états locaux $ps \subset \mathbf{LS}$, le recouvrement de $\varsigma$ par $ps$ est noté $\varsigma \Cap ps$ et est défini par :

$$\forall a \in S, (\varsigma \Cap ps)[a] \triangleq \begin{cases} \{p \in ps \mid \mathsf{automate}(p) = a\} & \text{si } \exists p \in ps, \mathsf{automate}(p) = a, \\ \varsigma[a] & \text{sinon.} \end{cases}$$

Pour tout contexte $\varsigma \in \mathbf{Ctx}$ et tout état local $a_i \in \mathbf{LS}$, on note : $a_i \in \varsigma \overset{\Delta}{\Leftrightarrow} a_i \in \varsigma[a]$, et pour tout état $ps \in S$ ou ensemble d'états locaux $ps \subset \mathbf{LS}$, on note : $ps \subseteq \varsigma \overset{\Delta}{\Leftrightarrow} \forall a_i \in ps, a_i \in \varsigma$. De plus, une séquence d'actions $\delta$ est *jouable* dans un contexte $\varsigma$ si et seulement si $\exists s \subseteq \varsigma, \delta \in \mathbf{Sce}(s)$ ; on note alors : $\delta \in \mathbf{Sce}(\varsigma)$, et l'exécution de la séquence $\delta$ dans $\varsigma$ est : $\varsigma \cdot \delta = \varsigma \Cap \mathsf{fin}(\delta)$.

La durée moyenne pendant laquelle un contexte $\varsigma$ reste actif est égale à la durée de vie de l'état local $a_i \in \varsigma$ ayant la plus petite durée de vie.

**Définition 4.14** (Durée d'activité moyenne d'un contexte ($\rho(\varsigma)$))**.** Pour tout contexte $\varsigma \in \mathbf{Ctx}$, la durée d'activité moyenne de $\varsigma$ est donnée par $\rho(\varsigma) = \min_{a_i \in \varsigma}\{\rho(a_i)\}$.

Finalement, une séquence de transitions sur un automate $a$ (définition 4.15) est une séquence d'actions dans $a$ dans laquelle la destination de chaque transition est égale à l'origine de la transition suivante, en ignorant donc la pré-condition (un ensemble d'états locaux requis pour appliquer la transition) de chaque transition. Les séquences de transitions sont utilisées pour trouver les solutions locales d'un objectif donné. Une séquence de transitions sur $a$ peut être abstraite par l'ensemble de toute les préconditions de ses transitions qui ne sont pas dans $a$ (définition 4.16). Cette abstraction permet de déplacer un objectif qui concerne un automate $a$ vers d'autres objectifs sur d'autres automates. On note dans la suite : $\mathbf{Sol} = \wp(\mathbf{LS})$ (définition 4.19).

**Définition 4.15** (Séquence de transitions (**TS**))**.** Une *séquence de transitions* $\zeta$ est une séquence d'actions telle que $\forall n \in \mathbb{I}^{\zeta}, n < |\zeta|, \text{dest}(\zeta_n) = \text{orig}(\zeta_{n+1})$. L'ensemble de toutes les séquences de transitions est appelé **TS**, et on note **TS**$(O)$ l'ensemble de toutes les séquences de transitions *résolvant* un objectif $O$, appelé **TS**$(O)$, qui est défini par :

$$\mathbf{TS}(a_i \rightsquigarrow a_j) \triangleq \{\zeta \in \mathbf{TS} \mid \text{orig}(\zeta_1) = a_i \wedge \text{dest}(\zeta_{|\zeta|}) = a_j\} \ .$$

On remarque que pour tout objectif $a_i \rightsquigarrow a_j \in \mathbf{Obj}$, $\mathbf{TS}(a_i \rightsquigarrow a_j) = \emptyset$ s'il n'existe aucun moyen d'atteindre $a_j$ depuis $a_i$. À l'inverse, la séquence vide appartient toujours à l'ensemble des séquences de transitions résolvant un objectif trivial : $\forall a_i \in \mathbf{Proc}, \varepsilon \in \mathbf{BS}(a_i \rightsquigarrow a_i)$.

**Définition 4.16** (Séquence de transitions abstraite ($\mathbf{TS}^{\wedge} : \mathbf{Obj} \to \wp(\mathbf{Sol})$))**.**

$$\mathbf{TS}^{\wedge}(O) \triangleq \{\zeta^{\wedge} \in \mathbf{Sol} \mid \zeta \in \mathbf{TS}(O), \nexists \zeta' \in \mathbf{TS}(O), \zeta'^{\wedge} \subsetneq \zeta^{\wedge}\} \ ,$$

où $\zeta^{\wedge} \triangleq \{\text{precond}(\zeta_n) \mid n \in \{1, \ldots, |\zeta|\} \wedge \text{automate}(\text{precond}(\zeta_n)) \neq \text{automate}(O)\}$.

On note $\gamma_{\varsigma}(\omega)$ l'ensemble des scénarios concrétisant une séquence d'objectifs $\omega$ dans le contexte $\varsigma$ (définition 4.17)

**Définition 4.17** ($\gamma_{\varsigma} : \mathbf{OS} \to \wp(\mathbf{Sce})$)**.** Pour toute séquence d'objectifs $\omega \in \mathbf{OS}$, $\gamma_{\varsigma}(\omega)$ est l'ensemble des scénarios minimaux concrétisant $\omega$ dans le contexte $\varsigma$. Il est défini comme le plus grand ensemble satisfaisant les conditions suivantes :

(i) $\forall \delta \in \gamma_{\varsigma}(\omega), \exists s \subseteq \varsigma, \delta \in \mathbf{Sce}(s)$,

(ii) $\forall \delta \in \gamma_{\varsigma}(\omega), \exists \phi : \mathbb{I}^{\omega} \to \mathbb{I}^{\delta}, (\forall n, m \in \mathbb{I}^{\omega}, n < m \Leftrightarrow \phi(n) \leq \phi(m)), \forall n \in \mathbb{I}^{\omega}, \text{bond}(\omega_n) \in \varsigma \cdot \delta_{1 \ldots \phi(n)}$,

(iii) $\forall \delta, \delta' \in \gamma_{\varsigma}(\omega), |\delta| \leq |\delta'| \Rightarrow \delta \neq \delta'_{1 \ldots |\delta|}$.

La figure 4.5 à la page suivante illustre les liens entre les différentes abstractions auxquelles nous pouvons avoir recours pour comprendre la dynamique des réseaux d'automates stochastiques. En effet, la dynamique des réseaux d'automates ($SAN$) peut être abstraite en ensemble de scénarios (**Sce**) ou en ensemble de séquences de transitions (**TS**). L'ensemble des scénarios peut être abstrait en séquence de transitions (**TS**) et en séquence d'objectifs (**OS**). Enfin, l'ensemble des séquences de transitions peut être abstrait en séquence de transitions abstraites ($\mathbf{TS}^{\wedge}$).

Dans la section 4.5.1, nous allons introduire les structures abstraites qui nous permettront donner des estimations des probabilités et des délais d'observer les scénarios nécessaires à la réalisation d'une séquence d'objectifs.

## 4.5 Approximations Inf et Sup de la probabilité et des délais d'accessibilité

L'objet de cette section est de partir des abstractions présentées dans la section précédente pour construire les structures sur lesquelles il est possible de calculer les bornes inférieures et supérieures de la probabilité et des délais d'atteindre un état $\omega$ partant d'un contexte initial $\varsigma$. Nous commençons par rappeler les structures introduites par (Paulevé, 2011) pour les approximations des propriétés d'accessibilité d'un point de vue qualitatif. Puis

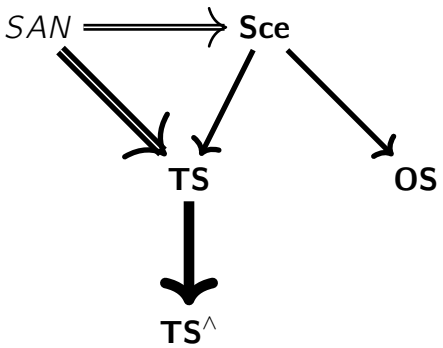



Figure 4.5 : Relation de dérivation entre les réseaux d'automates stochastiques ($\mathcal{SAN}$), les scénarios (**Sce**), les séquences de transitions (**TS**) et de transitions abstraites (**TS**$^{\wedge}$), les séquences d'objectifs (**OS**). Comme nous l'avons introduit dans cette section, il est possible d'abstraire le comportement d'un réseau d'automates stochastiques ($\mathcal{SAN}$) en scénarios (**Sce**) ou en séquences de transitions (**TS**). Ce qui permet de construire un raisonnement par étape en ne se focalisant à chaque étape que sur un seul automate à la fois. Selon l'abstraction proposée, la capacité à construire des raisonnements et à dériver des propriétés va être différente. C'est ce qui explique la différence au niveau des traits. Dans cette thèse, nous ne rentrerons pas dans les spécificités des différentes abstractions.

nous présentons la spécificité de la structure que nous introduisons pour le calcul des bornes inférieures et supérieures de la probabilité et des délais.

### 4.5.1 Structures abstraites pour l'évaluation quantitative

Soit un contexte $\varsigma$ et une séquence d'objectifs $\omega$, dans (Paulevé et al., 2012), il est prouvé que $\omega$ est concrétisable seulement si chaque objectif $\omega_n$, $n \in \mathbb{I}^{\omega}$ est concrétisable indépendamment dans le même contexte (proposition 4.4). Il est donc possible d'approximer la concrétisabilité d'une séquence d'objectifs en appliquant une procédure récursive qui extrait les objectifs de la séquence donnée et se sert d'un opérateur (de raffinement) qui permet d'étendre l'objectif en plusieurs séquences d'objectifs. L'opérateur de raffinement $\beta$ n'est pas formellement présenté dans cette thèse, mais il permet, étant donné un objectif $O$, une séquence de transitions permettant la concrétisation de l'objectif $O$, d'abstraire la séquence de transitions en séquence d'objectifs décrivant l'atteinte successive de chaque séquence d'objectifs concrétisant la séquence de transitions.

**Proposition 4.4.** $\gamma_{\varsigma}(\omega) \neq \emptyset \Longrightarrow \forall n \in \mathbb{I}^{\omega}, \gamma_{\varsigma}(\omega_n) \neq \emptyset$.

La probabilité de l'ensemble des scénarios $\gamma_{\varsigma}(\omega)$ qui permettent de concrétiser une séquence d'objectifs $\omega$ est une borne supérieure de la probabilité de concrétiser la séquence d'objectifs $\omega$.

Si la séquence d'objectifs $\omega$ est concrétisable (c'est-à-dire $\gamma_{\varsigma}(\omega) \neq \emptyset$), cela implique que la probabilité de concrétiser cette séquence d'objectifs est non nulle (proposition 4.5).

**Proposition 4.5.** $\gamma_{\varsigma}(\omega) \neq \emptyset \Longrightarrow \mathcal{P}(\gamma_{\varsigma}(\omega)) \neq 0$.

*Démonstration.* La preuve est immédiate. En effet, par la proposition 4.4, la séquence d'objectifs $\omega$ est concrétisable si chaque objectif l'est indépendamment et a donc une

probabilité non nulle. Par la proposition 4.3, on a bien $\mathcal{P}(\gamma_\varsigma(\omega)) \neq 0$. □

De la proposition proposition 4.5, nous déduisons qu'il existe un nombre réel non nul $p \in ]0,1]$ tel que $\mathcal{P}(\gamma_\varsigma(\omega)) \geq p$. Si nous considérons la propriété suivante $\mathcal{P}_{\geq p}(\gamma_\varsigma(\omega))$, nous avons la proposition 4.6 :

**Proposition 4.6.** $\mathcal{P}(\gamma_\varsigma(\omega)) \neq 0 \Longrightarrow \exists p \in ]0,1], \mathcal{P}_{\geq p}(\gamma_\varsigma(\omega))$ *est vérifiable.*

Un objectif Q est re-ciblé par rapport à un objectif O, si les deux objectifs ont le même bond et les cibles différentes. Soit un objectif O, l'ensemble des objectifs re-ciblés Q, avec cible(O) ≠ cible(Q) et bond(O) = bond(Q), qui sont toujours dérivés lors d'une application récursive de $\beta^\wedge(O, \mathbf{TS}^\wedge(O))$ et de la proposition 4.4 est $\text{minCont}_\varsigma(O)$ (définition 4.18).

La procédure récursive vérifiant les conditions nécessaires pour la concrétisabilité d'un objectif est alors résumée par la proposition 4.7 ci-dessous.

Si $O$ est concrétisable, alors il existe une exécution de cette procédure sans cycle, et où tous les objectifs testés possèdent au moins une solution (théorème 4.1).

**Définition 4.18** ($\text{minCont}_\varsigma : \mathbf{Obj} \mapsto \wp(\mathbf{Obj})$)**.**

$$\text{minCont}_\varsigma(\star \rightsquigarrow a_j) = \{a_k \rightsquigarrow a_j \mid a_k \neq a_j \wedge \forall a_i \in \varsigma[a], a_k \in \text{minCont}_\varsigma^{\text{Obj}}(a, a_i \rightsquigarrow a_j)\} \quad (4.7)$$

$$\text{minCont}_\varsigma^{\text{Obj}} : \Sigma \times \mathbf{Obj} \mapsto \wp(\mathbf{LS}) \quad (4.8)$$

$$\text{minCont}_\varsigma^{\text{Obj}}(a, O) = \emptyset \quad si \quad \mathbf{TS}^\wedge(O) = \emptyset, sinon, \quad (4.9)$$

$$\text{minCont}_\varsigma^{\text{Obj}}(a, O) = \{p \in \mathbf{LS} | \forall ps \in \mathbf{TS}^\wedge(O), \exists q \in ps, p \in \text{minCont}_\varsigma^{\text{LS}}(a, q)\} \quad (4.10)$$

$$\text{minCont}_\varsigma^{\text{LS}} : \Sigma \times \mathbf{LS} \mapsto \wp(\mathbf{LS}) \quad (4.11)$$

$$\text{minCont}_\varsigma^{\text{LS}}(a, b_i) = \begin{cases} \{b_i\} & si\ a = b \\ \{p \in \mathbf{LS} \mid \forall b_i \in \varsigma[b],\ p \in \text{minCont}_\varsigma^{\text{Obj}}(a, b_j \rightsquigarrow b_i)\} & sinon \end{cases} \quad (4.12)$$

**Lemme 4.1.** $a_k \rightsquigarrow a_j \in \text{minCont}_\varsigma(\star \rightsquigarrow a_j) \Longrightarrow \gamma_\varsigma(\star \rightsquigarrow a_j) = \gamma_\varsigma(\star \rightsquigarrow a_k :: a_k \rightsquigarrow a_j)$

**Proposition 4.7.** $\gamma_\varsigma(O) \neq \emptyset \Longrightarrow \exists ps \in \mathbf{TS}_\varsigma^\wedge(O), \forall p \in ps, \gamma_\varsigma(\star \rightsquigarrow p) \neq \emptyset, et \forall Q \in \text{minCont}_\varsigma(O), \gamma_\varsigma(Q) \neq \emptyset$

**Théorème 4.1** (Approximation sup. désordonnée)**.** $\gamma_\varsigma(\omega) \neq \emptyset \Longrightarrow \forall n \in \mathbb{I}^\omega$*, il existe une application récursive finie de la proposition 4.7 pour* $\gamma_\varsigma(\omega) \neq \emptyset$ *telle que pour tous les objectifs testés* $O, \mathbf{TS}_\varsigma^\wedge(O) \neq \emptyset$*.*

*Démonstration.* Par induction, si $\gamma_\varsigma(O) \neq \emptyset$ requiert $\gamma_\varsigma(O') \neq \emptyset$, $O \neq O'$, et si $\gamma_\varsigma(O') \neq \emptyset$ requiert à son tour $\gamma_\varsigma(O) \neq \emptyset$, alors $\gamma_\varsigma(O) = \emptyset$ ; ainsi, par la proposition 4.4, il existe une application récursive finie de la proposition 4.7 pour $\gamma_\varsigma(\omega) \neq \emptyset$, $\forall n \in \mathbb{I}^\omega$. Enfin la proposition 4.7 implique la non vacuité de $\mathbf{TS}_\varsigma^\wedge(O) \neq \emptyset$ pour chaque objectif $O$ testé. □

#### 4.5.1.1 Structure abstraite $\mathcal{A}_\varsigma^\omega$ : Graphe de Causalité Locale

Pour une séquence d'objectifs $\omega$ et un contexte $\varsigma$ donnés, nous donnons la définition de la structure abstraite $\mathcal{A}_\varsigma^\omega$ (définition 4.21) qui représente les relations entre les objectifs lors de l'exécution de la proposition 4.7. Cette structure prend la forme d'un graphe que nous appelons *Graphe de Causalité Locale* $\mathcal{A}_\varsigma^\omega$ (définition 4.21). Ce graphe est construit par un raisonnement récursif conformément au théorème 4.1.

Un objectif $O = a_i \rightsquigarrow a_j \in \mathbf{Obj}$ est résolvable si tous les états locaux contenus dans au moins une de ses abstractions de séquences de transitions $\mathbf{TS}^{\wedge}(a_i \rightsquigarrow a_j) \in \mathbf{Sol}$ (cf. définition 4.16) peuvent être activés (voir l'équivalence (3) de la définition 4.21). Une telle solution représente donc un ensemble d'états locaux qui doivent être activés pour la résolution de $O = a_i \rightsquigarrow a_j$ (4). Nous définissons donc $\mathsf{sol}(O) \subset \wp(\mathbf{LS})$ la *causalité locale* de l'objectif O (définition 4.19) : chaque $ls \in \mathsf{sol}(O)$ est un ensemble d'états locaux qui peuvent être impliqués dans les transitions permettant l'accessibilité de $a_j$ depuis $a_i$. Nous appelons $ls$ une solution pour l'objectif O.

**Définition 4.19** (Causalité locale)**.** $sol : \mathbf{Obj} \to \wp(\mathbf{Sol})$ associe à un objectif un ensemble d'états locaux (solutions) tel que $\forall O \in \mathbf{Obj}, \forall ls \in \mathsf{sol}(O), \nexists\, ls' \in \mathsf{sol}(O), ls \neq ls'$ avec $ls' \subset ls$. l'ensemble de ces associations est noté $\mathbf{Sol} \triangleq \{\langle O, ls\rangle \mid ls \in \mathsf{sol}(O)\}$.

Les ensembles de solutions $\mathsf{sol}(O = a_i \rightsquigarrow a_j)$ sont dits nécessaires si la désactivation d'au moins un état local de chaque solution rend l'accessibilité de $a_j$ impossible depuis n'importe quel état contenant $a_i$ (définition 4.20). Ainsi, si $\mathsf{sol}(O) = \{\{b_k\} \cup ls^1, \cdots, ls^m\}$ est nécessaire alors $\mathsf{sol}(O)' = \{ls^1, \cdots, ls^m\}$ est également nécessaire ; $\mathsf{sol}(a_i \rightsquigarrow a_j) = \emptyset$ implique que $a_j$ n'est jamais accessible depuis $a_i$.

**Définition 4.20** (sol nécessaire)**.** $\mathsf{sol}(a_i \rightsquigarrow a_j) = \{ls^1, \cdots, ls^n\}$ est un ensemble de solutions nécessaires pour le réseau d'automates $\mathcal{SAN} = (\Sigma, S, T)$ si et seulement si $\forall k ls$ appartenant au produit des sous-ensembles des solutions $ls^i$ pour $i \in [1\,;n]$, $a_j$ n'est pas accessible dans le réseau $\mathcal{SAN} \ominus k ls$ depuis tout état $s \in S$ où $s(a) = i$.

Le Graphe de Causalité Locale relie les objectifs à des solutions à l'aide des séquences de transitions abstraites (principe de la causalité locale) de la définition 4.16, ce qui produit de nouveaux objectifs résolus récursivement. Il s'agit donc d'un graphe dont les nœuds sont des éléments de $\mathbf{LS} \cup \mathbf{Obj} \cup \mathbf{Sol}$, c'est-à-dire des états locaux, des objectifs et des *solutions* (c'est-à-dire des ensembles d'états locaux) :

- Un nœud dans **Obj** représente un objectif requis pour la résolution de $\omega$, soit faisant directement partie de la séquence d'objectifs $\omega$, soit indirectement nécessaire à sa résolution ;
- Un nœud dans **Sol** représente un ensemble d'états locaux nécessaires pour résoudre un objectif, c'est-à-dire un élément parmi ses séquences de transitions abstraites ;
- Un nœud dans **LS** représente un état local requis pour la résolution, c'est-à-dire mentionné dans un nœud solution.

Ce graphe peut bien sûr contenir des cycles. La définition 4.21 donne une définition formelle du graphe de causalité locale.

**Définition 4.21.** Le graphe de causalité locale $\mathcal{A}^{\omega}_{\varsigma} \triangleq (V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$ est le plus petit graphe respectant $V^{\omega}_{\varsigma} \subseteq \mathbf{LS} \cup \mathbf{Obj} \cup \mathbf{Sol}$ et $E^{\omega}_{\varsigma} \subseteq V^{\omega}_{\varsigma} \times V^{\omega}_{\varsigma}$ tel que :

1. $\omega \in V^{\omega}_{\varsigma}$
2. $a_i \in V^{\omega}_{\varsigma} \cap \mathbf{LS} \Leftrightarrow \{(a_i, a_j \rightsquigarrow a_i) \mid a_j \neq \omega \wedge (a_j \in s \vee a_j \in V^{\omega}_{\varsigma} \cap \mathbf{LS})\} \subseteq E^{\omega}_{\varsigma}$
3. $O \in V^{\omega}_{\varsigma} \cap \mathbf{Obj} \Leftrightarrow \{(O, \langle O, \zeta^{\wedge}\rangle) \mid \zeta \in \mathbf{TS}(O)\} \subseteq E^{\omega}_{\varsigma}$
4. $\langle O, pps\rangle \in V^{\omega}_{\varsigma} \cap \mathbf{Sol} \Leftrightarrow \{(\langle O, pps\rangle, ps) \mid ps \in pps\} \subseteq E^{\omega}_{\varsigma}$

Dans le pire des cas, $|V^{\omega}_{\varsigma}| = |\mathbf{LS}| + |\mathbf{Obj}| + |\mathbf{Sol}|$ et $|E^{\omega}_{\varsigma}| = |\mathbf{Obj}| + |\mathbf{Sol}| + \sum_{\langle P, \zeta^{\wedge}\rangle \in \mathrm{Sol}} |\zeta^{\wedge}|$

avec $|\mathbf{LS}| = \sum_{a\in\Sigma} |S(a)|$ et $|\mathbf{Obj}| = \sum_{a\in\Sigma} |S(a)|^2$. Le nombre de solutions par objectif correspond au nombre de séquences acycliques de transitions locales au sein d'un automate qui sont donc bornées par le nombre d'états locaux de l'automate : $|\mathbf{Sol}| \leq \sum_{a\in\Sigma} \binom{|T(a)|}{|S(a)|}$. Ainsi la complexité totale est exponentielle selon le nombre d'états locaux au sein d'un seul automate, et polynomiale selon le nombre d'automates et le nombre de transitions locales.

La complexité de la construction du graphe de causalité locale $\mathcal{A}^{\omega}_{\varsigma}$, permet de détecter de façon efficace des motifs nécessaires pour l'accessibilité dans un réseau d'automate.

**Motif du GLC nécessaire pour l'accessibilité.** Étant donné un réseau d'automates $\mathcal{SAN} = (\Sigma, S, T)$, un contexte $\varsigma$ et un état local $a_j \in \mathbf{LS}$, nous pouvons déduire du théorème 4.1, de la définition 4.19 et de la définition 4.20 que l'accessibilité de $a_j$ depuis un état dans $\varsigma$ implique l'existence d'un parcours du GLC partant du nœud $a_j$ et tel que : si un nœud de type état local est traversé, alors au moins un de ses enfants (objectifs) est traversé ; si un nœud de type objectif est traversé, alors au moins un de ses enfants (solutions) est traversé ; si un nœud de type solution est traversé, alors tous ses enfants (états locaux) sont traversés ; le parcours est sans cycle. L'ensemble des solutions et états locaux à partir desquels un tel parcours existe dans le GLC $(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$ où $\omega = a_j$, est donné par la définition 4.22 ((Paulevé, Folschette, Magnin & Roux, 2015)).

**Définition 4.22** (Motif valide pour la condition nécessaire de l'accessibilité)**.** Étant donné un GLC $\mathcal{A}^{\omega}_{\varsigma} \triangleq (V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$, $\mathsf{validMotif}(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$ est le plus petit ensemble satisfaisant :

1. $\{\langle (O, \emptyset)\rangle \in V^{\omega}_{\varsigma} \cap \mathbf{Sol}\} \subset \mathsf{validMotif}(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$ ;
2. $\langle a_i \rightsquigarrow a_j, Is\rangle \in \mathsf{validMotif}(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma}) \Longrightarrow a_j \in \mathsf{validMotif}(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$ ;
3. $\forall O \in V^{\omega}_{\varsigma} \cap \mathbf{Obj}$ où $sol(O) \neq \emptyset$, $\forall Is \in sol(O)$, $Is \subset \mathsf{validMotif}(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma}) \Longrightarrow \langle O, Is\rangle \in \mathsf{validMotif}(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$.

Le théorème 4.2 donne les conditions nécessaires pour l'accessibilité d'un état local à partir d'un contexte donné. Il établit qu'un état local $\omega = a_j \in \mathbf{LS}$ est accessible depuis un contexte initial $\varsigma$ dans un réseau d'automates $\mathcal{SAN} = (\Sigma, S, T)$ si et seulement si il est possible de construire un graphe de causalité locale et d'en extraire un motif valide contenant $a_j$.

**Théorème 4.2** (Condition nécessaire pour l'accessibilité (OA))**.** *Soit $\mathcal{SAN} = (\Sigma, S, T)$ un réseau d'automates, $\omega = a_j \in \mathbf{LS}$ un de ses états locaux, et un contexte $\varsigma$ pour le réseau. $a_j$ est accessible depuis un état délimité par $\varsigma$ seulement si $a_j \in \mathsf{validMotif}(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$, où $(V^{\omega}_{\varsigma}, E^{\omega}_{\varsigma})$ est le Graphe de Causalité Locale, et $\forall O \in V^{\omega}_{\varsigma} \cap \mathbf{Obj}$, sol(O) est nécessaire (cf. définition 4.20).*

*Démonstration.* Pour toute séquence de transitions $s^1 \to \cdots \to s^n$ avec $s^n(a) = j$ et $\forall b \in S$, $s^1(b) \in \varsigma(b)$ d'après la définition 4.20, il existe une solution $Is \in sol(s^1(a) \to^* s^n(a))$ telle que soit $Is = \emptyset$, soit $\forall c_k \in Is$, il existe $m$, $1 \leq m < n$, où $s^m(c) = k$. □

Les détails sur ces conditions nécessaires pour l'accessibilité sont explicités dans (Paulevé, 2011; Paulevé et al., 2015).

*Exemple.* La figure 4.6 à la page suivante représente le graphe de causalité locale pour la sur-approximation associée au réseau d'automates stochastiques de la figure 4.4, pour

l'accessibilité de $d_2$ depuis l'état $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$.

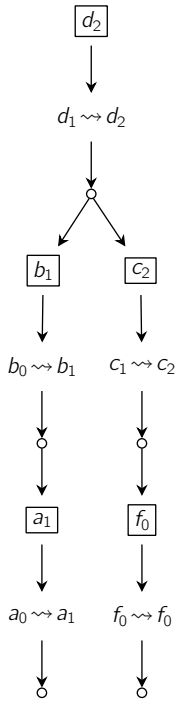



Figure 4.6 : Exemple de graphe de causalité locale pour la sur-approximation de l'accessibilité de $d_2$ partant de l'état $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$ pour le réseau d'automate de la figure 4.4. Les carrés représentent les états locaux, les objectifs ont la forme $a_i \rightsquigarrow a_j$. Les petits cercles représentent les solutions.

#### 4.5.1.2 Structure abstraite $\lceil \mathcal{B}^{\omega}_{\varsigma} \rceil$ : Graphe de Causalité Locale Simultané Saturé

Nous présentons dans cette section une structure qui permet d'établir les conditions suffisantes pour l'accessibilité dans les réseaux d'automates asynchrones. Comme pour la section précédente, la notion d'état est généralisée à la notion de contexte. Les résultats reposent sur une extension du Graphe de Causalité Locale présentée dans la section précédente pour faire apparaître les états partiels imposés par les pré-conditions de transitions impliquant un nombre arbitraire d'automates.

La structure $\lceil \mathcal{B}^{\omega}_{\varsigma} \rceil$ dont nous donnons une définition formelle à la définition 4.27 est une variante du *Graphe de Causalité Locale* (définition 4.21) sur laquelle il est maintenant possible d'établir sous certaines conditions (sans cycle, tous objectifs possèdent une solution), la preuve d'une accessibilité qualitative (Paulevé et al., 2012; Folschette et al., 2015). L'idée est de pouvoir proposer une construction d'un scénario qui concrétise la séquence d'objectifs. Cette construction est dite de haut en bas. Étant donné une séquence d'objectifs $\omega$, l'idée est de construire dans un premier temps le scénario concrétisant $\omega_1$ en ignorant la résolution de la séquence d'objectifs $\omega_{2..|\omega|}$. La concrétisation de $\omega_1$ induit la concrétisation d'un raffinement de $\omega_1$, ce qui résulte en une procédure récursive.

Nous commençons par donner une définition alternative de l'ensemble des scénarios concrétisant une séquence d'objectifs $\omega$ dans un contexte $\varsigma$.Nous définissons $\ell_\varsigma(\omega)$ comme étant égal à $\gamma_\varsigma(\omega)$ si et seulement si, pour chaque état $s \subseteq \varsigma$, $\gamma_\varsigma(\omega) \cap \mathbf{Sce}(s) \neq \emptyset$ (définition 4.23) et vide sinon.

**Définition 4.23** ($\ell_\varsigma : \mathbf{OS} \to \wp(\mathbf{Sce})$)**.**

$$\ell_\varsigma(\omega) \triangleq \begin{cases} \gamma_\varsigma(\omega) & \text{si } \forall s \in S, s \subseteq \varsigma, \exists \delta \in \gamma_\varsigma(\omega), \delta \in \mathbf{Sce}(s) \\ \emptyset & \text{sinon.} \end{cases}$$

**Lemme 4.2.** $\varsigma \subseteq \varsigma' \wedge \ell_{\varsigma'}(\omega) \neq \emptyset \Rightarrow \ell_\varsigma(\omega) \neq \emptyset$.

Pour tout objectif $O \in \mathbf{Obj}$ et tout contexte $\varsigma \in \mathbf{Ctx}$, la définition 4.24 permet d'obtenir $\mathrm{maxCont}_\varsigma(\mathsf{automate}(O), O)$ qui est l'ensemble des états locaux de $\mathsf{automate}(O)$ requis pour résoudre $O$ dans $\varsigma$. Cette définition sera utile pour correctement résoudre les accessibilités locales qui nécessitent indirectement un état local de leur propre automate, c'est-à-dire autrement que par une transition non labélisée.

**Définition 4.24** ($\mathrm{maxCont}_\varsigma : \Sigma \times \mathbf{Obj} \to \wp(\mathbf{LS})$)**.**

$$\mathrm{maxCont}_\varsigma(a, O) \triangleq \{p \in \mathbf{LS} \mid \exists ps \in \mathbf{TS}^{\wedge}(O), \exists b_i \in ps, b = a \wedge p = b_i \\ \vee\, b \neq a \wedge p \in \mathrm{maxCont}_\varsigma(a, b_j \rightsquigarrow b_i) \wedge b_j \in \varsigma[b]\} \ .$$

Étant donné que l'état local actif de chaque automate peut évoluer au cours de la résolution, le graphe de causalité locale $\lceil \mathcal{B}^{\omega}_{\varsigma} \rceil$ est obtenu par *saturation* avec tous les états locaux qu'il contient, c'est-à-dire en recouvrant le contexte initial $\varsigma$ par $\mathsf{ls}(V, E)$, défini par :

$$\mathsf{ls}(V, E) = (V \cap \mathbf{LS}) \cup \{\mathsf{cible}(O), \mathsf{bond}(O) \mid O \in V \cap \mathbf{Obj}\} \ .$$

Ce recouvrement est effectué autant de fois que nécessaire ; le graphe de causalité locale est donc re-calculé avec cette saturation jusqu'à ce qu'il n'évolue plus — autrement dit, jusqu'à atteindre un point fixe.

Pour un objectif $O = a_i \rightsquigarrow a_j \in \mathbf{Obj}$ donné, cela se traduit par la notion de *causalité locale simultanée* de O $nsol(O) \subset \wp(\wp(\mathbf{LS}))$ que la définition 4.25 détaille. En effet,

chaque $nls \in sol(P)$, appelé solution n-aire, correspond à un ensemble de pré-conditions, qui sont des ensembles d'états locaux dont la présence simultanée est requise pour pouvoir appliquer la transition à l'état global du réseau.

**Définition 4.25** (causalité locale simultanée nsol)**.** $nsol$ : **Obj** $\rightarrow \wp(\wp(\wp(\mathbf{LS})))$ associe à un objectif un ensemble de solutions n-aires, une solution n-aire étant un ensemble d'ensembles d'états locaux. L'ensemble de ces associations est noté **Nsol** $\triangleq \{\langle O, nls\rangle \mid nls \in nsol(O)\}$.

Si toute solution n-aire composant un ensemble $nsol(a_i \rightsquigarrow a_j)$ a une séquence acyclique complète de transitions locales au sein de l'automate $a$ entre ses états locaux $i$ et $j$, alors l'ensemble $nsol(a_i \rightsquigarrow a_j)$ est dit suffisant (définition 4.26).

**Définition 4.26** (nsol suffisant)**.** Étant donné un réseau d'automates asynchrone $\mathcal{SAN} = (\Sigma, S, T)$ où $a_i \rightsquigarrow a_j \in$ **Obj**, $nsol(a_i \rightsquigarrow a_j)$ est suffisant si et seulement si, pour tout $nls \in nsol(a_i \rightsquigarrow a_j)$, il existe une séquence acyclique $i \xrightarrow{l_1} \cdots \xrightarrow{l_m} j$ de transitions locales dans $T(a)$ telle que, pour tout $l \in \{l_1, \cdots, l_m\}$, $\{b_j \in {}^\bullet l \mid b \neq a\} \in nls$.

La définition 4.27 propose une version étendue du graphe de causalité locale de la définition 4.21 avec la notion de causalité locale simultanée, et en considérant tous les objectifs possibles impliquant les états locaux référencés dans les solutions n-aires. Ce graphe rend également compte de potentielles décompositions d'objectifs. Ces décompositions d'objectifs se produisent dans les cas où lors de l'accessibilité de $a_j$ depuis $a_i$, si l'état local $a_k$ ($k \notin \{i, j\}$) peut être requis avant l'atteinte de $a_j$ ($a_k$ appartient à la descendance de $a_i \rightsquigarrow a_j$ dans le graphe), $a_k \rightsquigarrow a_j$ est ajouté en tant que fils de l'objectif $a_i \rightsquigarrow a_j$.

**Définition 4.27.** Le graphe de causalité locale simultanée saturé $\lceil \mathcal{B}^\omega_\varsigma \rceil \triangleq (\lceil V^\omega_\varsigma \rceil, \lceil E^\omega_\varsigma \rceil)$ est défini par : $\lceil \mathcal{B}^\omega_\varsigma \rceil \triangleq$ **pppf**$\{\mathcal{B}^\omega_\varsigma\}\left(\mathcal{B}^\omega_\varsigma \mapsto \mathcal{B}^\omega_{\varsigma \uplus \mathbf{ls}(\mathcal{B}^\omega_\varsigma)}\right)$, où $\mathcal{B}^\omega_\varsigma \triangleq (V^\omega_\varsigma, E^\omega_\varsigma)$ est le plus petit graphe respectant $V^\omega_\varsigma \subseteq \mathbf{LS} \cup \mathbf{Obj} \cup \mathbf{Sol} \cup \mathbf{Syn}$ et $E^\omega_\varsigma \subseteq V^\omega_\varsigma \times V^\omega_\varsigma$ tel que :

1. $\omega \in V^\omega_\varsigma$
2. $a_i \in \lceil \mathcal{B}^\omega_\varsigma \rceil \cap \mathbf{LS} \Leftrightarrow \{(a_i, a_j \rightsquigarrow a_i) \mid a_j \neq \omega \wedge (a_j \in s \vee a_j \in V^\omega_\varsigma \cap \mathbf{LS})\} \subseteq E^\omega_\varsigma$
3. $O \in V^\omega_\varsigma \cap \mathbf{Obj} \Leftrightarrow \{(O, \langle O, \zeta^\wedge\rangle) \mid \zeta \in \mathbf{TS}(O)\} \subseteq E^\omega_\varsigma$
4. $\langle O, pps\rangle \in V^\omega_\varsigma \cap \mathbf{Sol} \Leftrightarrow \{(\langle O, pps\rangle, ps) \mid ps \in pps\} \subseteq E^\omega_\varsigma$
5. $ps \in V^\omega_\varsigma \cap \mathbf{Syn} \Leftrightarrow \{(ps, a_i) \mid a_i \in ps\} \subseteq E^\omega_\varsigma$
6. $a_i \rightsquigarrow a_j \in V^\omega_\varsigma \cap \mathbf{Obj} \Rightarrow \{(a_i \rightsquigarrow a_j, a_k \rightsquigarrow a_j) \mid a_k \neq a_j,$
   $a_k \in \mathrm{conn}^1_{(V^\omega_\varsigma, E^\omega_\varsigma)}(\langle a_i \rightsquigarrow a_j, \zeta^\wedge\rangle),$
   $\zeta \in \mathbf{TS}(a_i \rightsquigarrow a_j)\} \subseteq E^\omega_\varsigma$

La complexité du calcul du graphe $\lceil \mathcal{B}^\omega_\varsigma \rceil \triangleq (\lceil V^\omega_\varsigma \rceil, \lceil E^\omega_\varsigma \rceil)$ et la même que celle du calcul du GLC de la définition 4.21. Elle reste polynomiale selon le nombre d'automates et exponentielle selon le nombre d'états locaux par automate.

**Motif du GLC suffisant pour l'accessibilité.** Nous présentons dans cette section les conditions suffisantes à l'existence d'une suite de transitions au sein d'un réseau d'automates asynchrones $\mathcal{SAN} = (\Sigma, S, T)$ partant d'un état initial global $s \in S$ et atteignant

un état $s' \in S$ où $s(a) = j$ pour un automate $a \in S$ et un état local $j \in S(a)$ donnés. Comme pour les autres sections, nous généralisons la notion d'état à la notion de contexte. Nous nous intéressons ici à identifier dans un Graphe de Causalité Locale Simultané Saturé $\lceil \mathcal{B}^{\omega}_{\varsigma} \rceil \triangleq (\lceil V^{\omega}_{\varsigma} \rceil, \lceil E^{\omega}_{\varsigma} \rceil)$, les conditions suffisantes pour l'accessibilité de $a_j$ depuis tout état global délimité par le contexte.

Le théorème 4.3 établit une condition suffisante pour l'accessibilité d'un état local $a_j$ depuis n'importe quel état délimité par un contexte donné. La preuve de ce théorème est faite dans (Folschette et al., 2015). Cette condition repose sur l'absence de cycle dans le Graphe de Causalité Locale Simultané Saturé, et sur l'indépendance de l'accessibilité des états locaux de chaque pré-conditions des transitions considérées.

Cette condition n'est valable que pour les réseaux d'automates asynchrones. En effet, dans sa construction, le Graphe de Causalité Locale ne capture que la cause du changement de l'état local d'un automate, et ne prend pas en compte des effets de bords potentiels, comme le changement simultané d'états d'autres automates.

**Théorème 4.3** (Condition suffisante pour l'accessibilité (UA))**.** *un réseau d'automates asynchrone $\mathcal{SAN} = (\Sigma, S, T)$, $w = a_j \in$ **LS** un de ses états locaux, et $\varsigma$ un contexte pour le réseau. $a_j$ est accessible depuis tout état délimité par $\varsigma$ (c'est-à-dire $\ell_{\varsigma}(\omega)$) si le Graphe de Causalité Locale Simultanée Saturé $(\lceil V^{\omega}_{\varsigma} \rceil, \lceil E^{\omega}_{\varsigma} \rceil)$ satisfait les conditions suivantes :*

1. *$(\lceil V^{\omega}_{\varsigma} \rceil, \lceil E^{\omega}_{\varsigma} \rceil)$ est sans cycle ;*
2. *$\forall O \in \lceil V^{\omega}_{\varsigma} \rceil \cap$ **Obj**, nsol(O) est suffisant et $nsol(O) \neq \emptyset$ ;*
3. *$\forall ls \in \lceil V^{\omega}_{\varsigma} \rceil \cap \wp($**LS**$)$ où $|ls| \geq 2$, l'ensemble d'états locaux $cls = ls \cup conn_{(\lceil V^{\omega}_{\varsigma} \rceil, \lceil E^{\omega}_{\varsigma} \rceil)}(ls)$ contient au plus un état local par automate : $\forall a \in S, S(a) \cap ls \neq \emptyset \Rightarrow |S(a) \cap cls| \leq 1$.*

*Exemple.* La figure 4.7 à la page suivante représente le Graphe de Causalité Locale Simultané Saturé pour la sous-approximation associé au réseaux d'automates stochastiques de la figure 4.4, pour l'accessibilité de $d_2$ depuis l'état $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$.

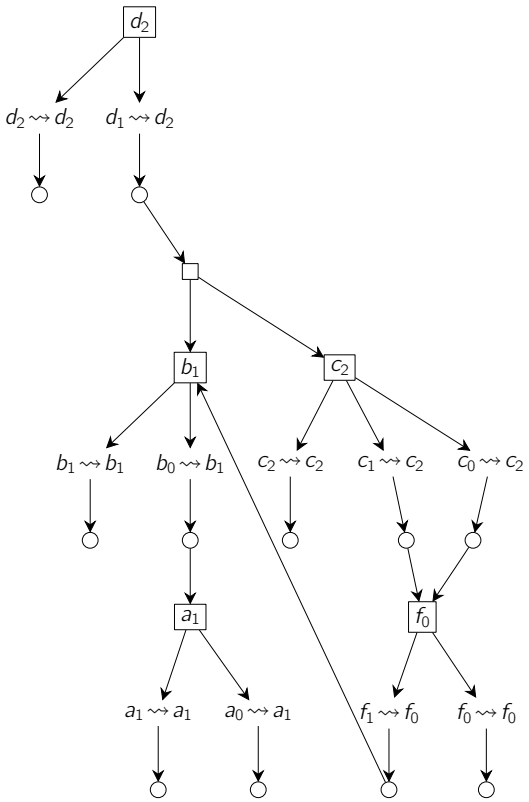



Figure 4.7 : Exemple de Graphe de Causalité Locale Simultané Saturé pour l'accessibilité de $d_2$ depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$ pour le réseau d'automates de la figure 4.4. Les carrés représentent les états locaux, les objectifs ont la forme $a_i \rightsquigarrow a_j$. Les petits cercles représentent les solutions.

#### 4.5.1.3 Structure abstraite $\lceil \mathcal{G}_\varsigma^\omega \rceil$ : Graphe de Causalité Locale Quantifié

Les constructions des structures $\mathcal{A}_\varsigma^\omega$ et $\lceil \mathcal{B}_\varsigma^\omega \rceil$ oublient délibérément une partie de l'information sur l'ordre ou l'arité des transitions locales. C'est la raison pour laquelle ces structures correspondent à des approximations supérieures et inférieures des comportements du modèle concret. Avec l'approximation supérieure, il est possible de générer plus de comportements que le système réel tandis qu'avec l'approximation inférieure, il est possible de générer potentiellement moins de comportements que le système réel.

Nous proposons ici de prendre en compte les taux sur les transitions qui peuvent être intégrés dans la construction d'une nouvelle structure que nous appelons *Graphe de Causalité Locale Quantifié* et notons $\lceil \mathcal{G}_\varsigma^\omega \rceil$. Cette structure est donc un graphe proche des graphes $\mathcal{A}_\varsigma^\omega$ et $\lceil \mathcal{B}_\varsigma^\omega \rceil$ et sa particularité est qu'elle possède à chaque nœud des quantités qui représentent soit des probabilités soit des délais. L'intérêt de cette nouvelle structure est qu'elle fournit un support de décision. En effet, dans certains cas où il n'est pas possible d'établir les conditions suffisantes (section 4.5.1.2) pour l'accessibilité dans le graphe de causalité locale $\lceil \mathcal{B}_\varsigma^\omega \rceil$, le Graphe de Causalité Locale Quantifié $\lceil \mathcal{G}_\varsigma^\omega \rceil$ permet le calcul d'une limite inférieure de la probabilité d'observer une propriété d'accessibilité. Le *Graphe de Causalité Locale Quantifié* $\lceil \mathcal{G}_\varsigma^\omega \rceil$ peut potentiellement contenir des cycles. Si ces cycles sont fermés (définition 4.29 à la page suivante), il n'est pas possible de conclure sur l'existence d'une borne inférieure pour la probabilité d'accessibilité ou pour les délais d'accessibilité. Par contre, si ces cycles sont ouverts (il existe au moins un objectif appartenant au cycle qui possède au moins une solution ne faisant pas partie du cycle), il est possible de conclure de l'existence d'une limite inférieure pour la probabilité et les délais d'accessibilité. Nous présentons dans la suite une variante des structures $\mathcal{A}_\varsigma^\omega$ et $\lceil \mathcal{B}_\varsigma^\omega \rceil$ sur laquelle il est possible d'évaluer $p \in ]0, 1]$ telle que $\mathcal{P}_{\geq p}(\gamma_\varsigma(\omega))$ proposition 4.6 en page 91.

La structure $\lceil \mathcal{G}_\varsigma^\omega \rceil$ diffère de la structure $\lceil \mathcal{B}_\varsigma^\omega \rceil$ parce que nous ajoutons une fonction de valuation $\upsilon$ qui permet d'associer une valeur (probabilité ou un délai) à chaque nœud du graphe suivant les formules du théorème 4.4 et du théorème 4.5 respectivement.

**Définition 4.28.** Le graphe de causalité locale quantifié $\lceil \mathcal{G}_\varsigma^\omega \rceil \triangleq (\lceil V_\varsigma^\omega \rceil, \lceil E_\varsigma^\omega \rceil, \upsilon)$ est défini par : $\lceil \mathcal{G}_\varsigma^\omega \rceil \triangleq \mathbf{pppf}\{\mathcal{G}_\varsigma^\omega\}\left(\mathcal{G}_\varsigma^\omega \mapsto \mathcal{G}^\omega_{\varsigma \circledcirc \mathbf{ls}(\mathcal{G}_\varsigma^\omega)}\right)$, où $\mathcal{G}_\varsigma^\omega \triangleq (V_\varsigma^\omega, E_\varsigma^\omega)$ est le plus petit graphe respectant $V_\varsigma^\omega \subseteq \mathbf{LS} \cup \mathbf{Obj} \cup \mathbf{Sol} \cup \mathbf{Syn}$ et $E_\varsigma^\omega \subseteq V_\varsigma^\omega \times V_\varsigma^\omega$ tel que :

1. $\omega \in V_\varsigma^\omega$
2. $a_i \in \lceil \mathcal{B}_\varsigma^\omega \rceil \cap \mathbf{LS} \Leftrightarrow \{(a_i, a_j \rightsquigarrow a_i) \mid a_j \neq \omega \wedge (a_j \in s \vee a_j \in V_\varsigma^\omega \cap \mathbf{LS})\} \subseteq E_\varsigma^\omega$
3. $O \in V_\varsigma^\omega \cap \mathbf{Obj} \Leftrightarrow \{(O, \langle O, \zeta^\wedge \rangle) \mid \zeta \in \mathbf{TS}(O)\} \subseteq E_\varsigma^\omega$
4. $\langle O, pps \rangle \in V_\varsigma^\omega \cap \mathbf{Sol} \Leftrightarrow \{(\langle O, pps \rangle, ps) \mid ps \in pps\} \subseteq E_\varsigma^\omega$
5. $ps \in V_\varsigma^\omega \cap \mathbf{Syn} \Leftrightarrow \{(ps, a_i) \mid a_i \in ps\} \subseteq E_\varsigma^\omega$
6. $a_i \rightsquigarrow a_j \in V_\varsigma^\omega \cap \mathbf{Obj} \Rightarrow \{(a_i \rightsquigarrow a_j, a_k \rightsquigarrow a_j) \mid a_k \neq a_j, a_k \in \mathrm{conn}^1_{(V_\varsigma^\omega, E_\varsigma^\omega)}(\langle a_i \rightsquigarrow a_j, \zeta^\wedge \rangle), \zeta \in \mathbf{TS}(a_i \rightsquigarrow a_j)\} \subseteq E_\varsigma^\omega$

Comme tout graphe, un Graphe de Causalité Locale peut contenir des cycles. Les cycles correspondent à une succession de nœuds tel que le premier nœud soit égal au dernier. Le cycle est fermé si aucun nœud objectif $\eta$ (c'est-à-dire $\eta \in V_\varsigma^\omega \cap \mathbf{Obj}$) appartenant au

cycle n'est connecté à un nœud solution n'appartenant pas au cycle. Les cycles fermés correspondent à des situations où la concrétisabilité des objectifs appartenant au cycle dépend essentiellement des nœuds du cycle. On note **Cyc** l'ensemble des cycles d'un graphe de causalité locale et $\mathbf{Cyc}_f$ l'ensemble des cycles fermés.

**Définition 4.29** (Cycle ($\mathcal{C}$) fermé dans un graphe de causalité locale)**.** Un cycle ($\mathcal{C}$) dans un graphe de causalité locale est une suite de nœuds $\eta_i \in V_\varsigma^\omega$, $i \in [1..p]$, $(\eta_1 \to \eta_2 \to \cdots \to \eta_p)$ telle que $\eta_1 = \eta_p$.
Le cycle est fermé si $\forall \eta_i \in \mathcal{C} \cap \mathbf{Obj}$, $\eta_i \not\to^* \eta_j$, et $\eta_j \notin \mathbf{Cyc}_f$.

Nous nous servons de la structure $\lceil \mathcal{G}_\varsigma^\omega \rceil$ pour proposer des estimations des quantités (probabilités, délais) pour la concrétisation des scénarios. La section 4.5.2 présente une estimation de la probabilité d'observer l'ensemble des scénarios et de l'estimation des délais. Puis dans la section 4.5.3 en page 106 nous présenterons comment extraire de la structure $\lceil \mathcal{G}_\varsigma^\omega \rceil$ des chemins vérifiant certaines propriétés extrêmes spécifiques (probabilité du scénario le plus probable, le moins probable, temps au plus tôt, au plus tard).

## 4.5.2 Approximation Inf de la probabilité et borne Inf du délai d'accessibilité

Si le *Graphe de Causalité Locale Quantifié* a certaines propriétés (sans cycle fermé voir la définition 4.29) alors le lemme 4.3 et le théorème 4.4 ci-dessous permettent de calculer une borne inférieure de la probabilité des nœuds. De même, le lemme 4.4 et le théorème 4.5 permettent de calculer une borne inférieure du délai d'accessibilité à un nœud.

La procédure de calcul est récursive et s'applique aisément sur la structure $\lceil \mathcal{G}_\varsigma^\omega \rceil$ (définition 4.28). La probabilité ou le délai d'un nœud interne dépend de ses nœuds fils. La probabilité ou le délai au niveau des feuilles est obtenue suivant le lemme 4.3. Sachant que les feuilles correspondent à un point fixe, il est donc possible de remonter l'arbre jusqu'à sa racine et d'obtenir les probabilités en se servant du théorème 4.4.

### 4.5.2.1 Borne inf de la probabilité

Ètant donnés un objectif $\omega$, un contexte $\varsigma$ et un graphe de causalité locale quantifié $\lceil \mathcal{G}_\varsigma^\omega \rceil$, nous nous servons du lemme 4.3 pour calculer les probabilités aux feuilles du graphe $\lceil \mathcal{G}_\varsigma^\omega \rceil$. En effet, la probabilité à un nœud qui est une feuille du graphe de causalité locale quantifié, est égale à la probabilité de rester actif de l'état local associé à ce nœud dans le réseau d'automates.

**Lemme 4.3.** *La probabilité à un nœud feuille $\eta$ du graphe de causalité locale quantifié est égale à la probabilité de rester actif de l'état local associé à ce nœud dans le réseau d'automates stochastiques $\mathcal{P}_{(\eta,\eta)}(t) = e^{-\lambda t}$ si ce nœud est dans le contexte $\varsigma$, et 0 sinon. Où $\lambda = \sum_{r \in R_s} r$ et $R_s = \{r : \exists (a_i, \ell, r, a_j) \in T(a) \wedge a_i \in s\}$.*

*Démonstration.* Une feuille $\eta$ dans $\lceil \mathcal{G}_\varsigma^\omega \rceil$ correspond à la situation où un point fixe a été atteint pendant la construction de $\lceil \mathcal{G}_\varsigma^\omega \rceil$. La probabilité de rester dans l'état local associé à cette feuille est $1 - \mathcal{P}_{(\eta,*)}(t)$ par la définition 4.5. Aussi, la probabilité de rester au nœud $\eta$ est : $e^{-\lambda t}$ d'où nous avons le lemme 4.3. □

*Exemple.* Calcul de la probabilité aux nœuds feuilles

Comme exemple, nous calculons la probabilité de $\mathsf{sol}(f_0 \rightsquigarrow f_0)$ qui est égale à la probabilité d'être dans l'état local $f_0$ et de rester dans l'état local $f_0$. D'où nous avons : $\mathcal{P}(\mathsf{sol}(f_0 \rightsquigarrow f_0)) = e^{-2*t} = 1$ (si on fait tendre $t$ vers 0).

Nous introduisons la fonction indicatrice $\mathbb{1}_{\varsigma}$ qui est égale à 1 si l'état local en argument de la fonction indicatrice est dans le contexte et 0 sinon. La fonction $\mathbb{1}_{\varsigma}$ est formellement définie comme suit :

**Définition 4.30** (Fonction indicatrice de contexte $\mathbb{1}_{\varsigma} : \mathbf{LS} \rightarrow \{0,1\}$)**.** La fonction indicatrice de contexte pour un état local $a_i \in \mathbf{LS}$ pour un réseau d'automates stochastiques est définie comme suit :

$$\mathbb{1}_{\varsigma}(a_i) = \begin{cases} 1 & si\ a_i \in \varsigma \\ 0 & sinon \end{cases}$$

**Théorème 4.4** (Borne inférieure (binf) de la probabilité d'un nœud $\eta \in \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil$ au rang $k$)**.** *Si le nœud $\eta$ est une feuille alors se servir du lemme 4.3. Sinon*

1. *La probabilité d'un nœud $\eta \in V_{\varsigma}^{\omega} \cap$ **Syn** au rang $k$ est donnée par :*
   $\mathcal{P}(\eta_k, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil) \triangleq \prod_{\eta_{k-1,i} \in LS_k} \mathcal{P}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil)$

2. *La probabilité d'un nœud $\eta \in V_{\varsigma}^{\omega} \cap$ **Sol** au rang $k$ est donnée par :*
   $\mathcal{P}(\eta_k, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil) \triangleq \prod_{\eta_{k-1,i} \in LS_k} \mathcal{P}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil)$

3. *La probabilité d'un nœud $\eta \in V_{\varsigma}^{\omega} \cap$ **Obj** au rang $k$ est donnée par :*
   $\mathcal{P}(\eta_k, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil) \triangleq \mathcal{P}(obj_k)[\frac{\sum_{\eta_{k-1,i} \in Sol_k} \mathcal{P}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil)}{\sum_{\eta_{k-1,i} \in Sol_k \cup Solc_k} \mathcal{P}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil)}]$

4. *La probabilité d'un nœud $\eta \in V_{\varsigma}^{\omega} \cap$ **LS** au rang $k$ est donnée par :*
   $\mathcal{P}(\eta_k, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil) \triangleq \sum_{\eta_{k-1,i} \in Obj_k} \mathbb{1}_{\varsigma}(\mathsf{cible}(\eta_{k-1})) \cdot \mathcal{P}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_{\varsigma}^{\omega} \rceil)$

*Démonstration.* La preuve s'obtient facilement par induction sur le Graphe de Causalité Locale Quantifié.

Par construction, si un Graphe de Causalité Locale Quantifié ne contient qu'un seul nœud (une feuille), alors ce nœud est un objectif sans solution, et la probabilité d'être concrétisé est 0 ; sinon, le Graphe de Causalité Locale Quantifié possède au moins un objectif associé à une solution. Si cet objectif est un objectif trivial, il a la même probabilité que le nœud solution qui lui est associé. Sinon, il faut se servir de l'équation (3) du théorème 4.4 pour le calcul de la probabilité. Or, cette équation fait intervenir le proposition 4.1 pour le calcul de $\mathcal{P}(obj_k)$. Ceci suppose qu'on connaît toujours tous les sous-ensembles d'états nécessaires pour le calcul de $\mathcal{P}(obj_k)$. Malheureusement, dans la construction du Graphe de Causalité Locale Quantifié, une partie des états (ceux connectés aux transitions concurrentes) est ignorée ; par contre, le poids des taux des transitions associées à ces états ignorés est pris en compte dans le calcul de la probabilité de réaliser l'objectif. Ceci implique que le résultat obtenu est une borne inférieure à la probabilité recherchée. C'est le cas pour tous les objectifs du Graphe de Causalité Locale Quantifié. Cette sous-approximation est propagée de façon mécanique aux autres nœuds du graphe. D'où : le résultat obtenu est une borne inférieure de la probabilité recherchée. □

Le théorème 4.4 donne la formule pour calculer la probabilité des nœuds internes d'un Graphe de Causalité Locale Quantifié. La probabilité d'un nœud dans $V_\varsigma^\omega \cap \mathbf{Syn}$ (nœud de synchronisation) est le produit des probabilités de ses fils (états locaux) (voir (1) du théorème 4.4). En effet, un nœud de synchronisation exprime le fait que l'objectif a besoin à la fois de la présence de tous les états locaux fils du nœud de synchronisation. La probabilité d'un nœud dans $V_\varsigma^\omega \cap \mathbf{LS}$ (nœud état local) est la probabilité de voir un de ses objectifs se réaliser. C'est donc une somme sur les probabilités des différents objectifs qui concrétisent l'état local comme formalisé en (4) du théorème 4.4. Il faut noter qu'un seul de ses objectifs peut se réaliser. En effet, dans un état donné du système, chaque automate a un seul état local actif. L'objectif qui se réalisera sera celui qui partira de l'état actif vers l'état local requis. La probabilité d'un nœud dans $V_\varsigma^\omega \cap \mathbf{Sol}$ (nœud solution) est la probabilité de l'état local associé à cette solution (voir (2) du théorème 4.4). La probabilité d'un nœud dans $V_\varsigma^\omega \cap \mathbf{Obj}$ (nœud objectif) est la probabilité de réaliser cet objectif, multiplié par la probabilité de réaliser un sous-scénario (donné par les probabilités des solutions) de l'objectif et divisé par le nombre de solutions sachant, qu'au final, un seul scénario contribue à la réalisation de l'objectif comme énoncé en (3) du théorème 4.4.

*Exemple.* Exemple illustratif du calcul des probabilités.

Nous illustrons ici la mise en œuvre de l'estimation des probabilités et des délais d'observer un ensemble de scénarios dans un réseau d'automates asynchrones ($\mathcal{SAN}$). Nous nous appuyons sur l'exemple de la figure 4.4 en page 86, où nous avons modifié les transitions comme indiqué ci-dessous, pour construire des Graphes de Causalité Locale Quantifiés.

En effet, dans la figure 4.4, nous remplaçons la transition $t_9 = d_1 \xrightarrow{b_1,c_2,2} d_2$ par deux nouvelles transitions $t_9 = d_1 \xrightarrow{b_1,2} d_2$ et $t_{9'} = d_1 \xrightarrow{c_2,2} d_2$ pour obtenir 7 transitions locales définies comme suit :

$$
\begin{aligned}
T(a) &= \{t_1 = a_0 \xrightarrow{\emptyset} a_1, t_2 = a_1 \xrightarrow{c_2,1} a_0\} \\
T(b) &= \{t_3 = b_0 \xrightarrow{a_1,2} b_1, t_4 = b_1 \xrightarrow{a_0,1} b_0\} \\
T(c) &= \{t_5 = c_0 \xrightarrow{f_0,2} c_1, t_6 = c_1 \xrightarrow{f_0,2} c_2, t_7 = c_1 \xrightarrow{f_1,1} c_0\} \\
T(d) &= \{t_8 = d_1 \xrightarrow{e_1,1} d_0, t_9 = d_1 \xrightarrow{b_1,2} d_2, t_{9'} = d_1 \xrightarrow{c_2,2} d_2\} \\
T(e) &= \{t_{10} = e_0 \xrightarrow{d_0,2} e_1, t_{11} = e_1 \xrightarrow{d_2,1} e_0\} \\
T(f) &= \{t_{12} = f_0 \xrightarrow{2} f_1, t_{13} = f_1 \xrightarrow{b_1,1} f_0\}
\end{aligned}
$$

Ce changement a pour but de faciliter l'illustration des notions de chemin le plus probable, chemin le plus rapide, chemin le moins probable et chemin le moins rapide. Aussi, nous conservons cet exemple pour le calcul des délais dans la section 4.5.2.2.

La figure 4.8 donne un exemple d'un graphe de causalité locale quantifié (section 4.5.2) pour l'approximation d'une borne inférieure de la probabilité de l'ensemble des scénarios pour l'accessibilité de $d_2$ dans le $\mathcal{SAN}$ de la figure 4.4 partant de l'état initial $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$.

Les probabilités des différents nœuds de la figure 4.8 sont obtenues comme indiqué

dans les calculs suivants :

$$\mathcal{P}(\mathsf{sol}(f_0 \rightsquigarrow f_0)) = e^{-2t} = 1 \ (si\ t \to 0), \qquad \mathcal{P}(\mathsf{sol}(f_1 \rightsquigarrow f_0)) = e^{-t}(1 - \mathcal{P}(b_1))$$
$$\mathcal{P}(\mathsf{sol}(a_1 \rightsquigarrow a_1)) = e^{-t}(1 - \mathcal{P}(c_2)), \qquad \mathcal{P}(\mathsf{sol}(a_0 \rightsquigarrow a_1)) = e^{-2t} = 1$$
$$\mathcal{P}(f_0 \rightsquigarrow f_0) = 1, \qquad \mathcal{P}(f_1 \rightsquigarrow f_0) = 1 - e^{-t}(\mathcal{P}(b_1))$$
$$\mathcal{P}(a_1 \rightsquigarrow a_1) = 1, \qquad \mathcal{P}(a_0 \rightsquigarrow a_1) = 1 - e^{-2*t}$$
$$\mathcal{P}(f_0) = \mathbb{1}_\varsigma(f_1)\mathcal{P}(f_1 \rightsquigarrow f_0) + \mathbb{1}_\varsigma(f_0)\mathcal{P}(f_0 \rightsquigarrow f_0)$$
$$\mathcal{P}(a_1) = \mathbb{1}_\varsigma(a_1)\mathcal{P}(a_1 \rightsquigarrow a_1) + \mathbb{1}_\varsigma(a_0)\mathcal{P}(a_0 \rightsquigarrow a_1)$$

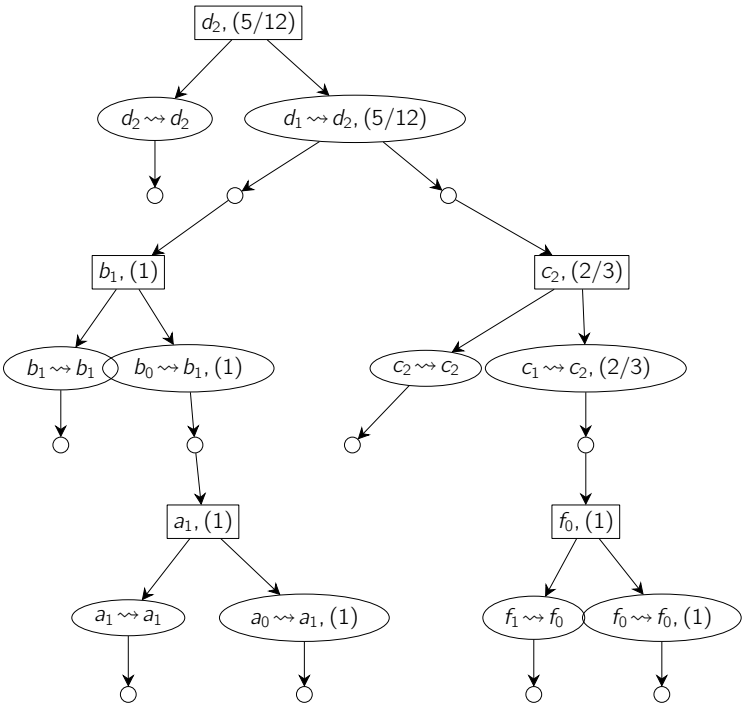



Figure 4.8 : Exemple de Graphe de Causalité Locale Quantifié pour l'estimation d'une borne inf de la probabilité de l'ensemble des scénarios pour l'accessibilité de $d_2$ depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$. Les carrés représentent les états locaux, les cercles représentent les objectifs. Les petits cercles représentent les solutions. Les chiffres entre parenthèses de chaque nœud représentent les probabilités.

#### 4.5.2.2 Borne Inf des délais

Ètant donnés un objectif $\omega$, un contexte $\varsigma$ et un graphe de causalité locale quantifié $\lceil \mathcal{G}_\varsigma^\omega \rceil$, nous nous servons du lemme 4.4 pour calculer les délais aux feuilles du graphe $\lceil \mathcal{G}_\varsigma^\omega \rceil$. En effet, le délai à un nœud qui est une feuille du graphe de causalité locale quantifié, est égale au délai nécessaire pour partir de l'état local associé à ce nœud dans le réseau d'automates. Ce qui correspond à la durée de vie de l'état local associé à ce nœud.

**Lemme 4.4.** *Le délai (moyen) passé dans un nœud feuille $\eta$ du Graphe de Causalité Locale Quantifié est égal à la durée de vie de l'état local associé à ce nœud dans le réseau d'automates stochastiques $\rho(ls(\eta))$ si ce nœud est dans le contexte $\varsigma$, et 0 sinon.*

*Démonstration.* La preuve est directe par la définition 4.9 en page 85. □

**Théorème 4.5** (Borne inférieure du délai de premier accès à un nœud $\eta \in \lceil \mathcal{G}_\varsigma^\omega \rceil$ au rang $k$)**.** *Si le nœud est une feuille, alors se servir du lemme 4.4. Sinon*

1. *Le délai de premier accès d'un nœud $\eta \in V_\varsigma^\omega \cap$ **Syn** au rang $k$ est donnée par :* $\mathcal{T}(\eta_k, \varsigma, \lceil \mathcal{G}_\varsigma^\omega \rceil) \triangleq moy_{\eta_{k-1,i} \in LS_k} \mathcal{T}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_\varsigma^\omega \rceil)$

2. *Le délais de premier accès d'un nœud $\eta \in V_\varsigma^\omega \cap$ **Sol** au rang $k$ est donnée par :* $\mathcal{T}(\eta_k, \varsigma, \lceil \mathcal{G}_\varsigma^\omega \rceil) \triangleq moy_{\eta_{k-1,i} \in LS_k} \mathcal{T}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_\varsigma^\omega \rceil)$

3. *Le délais de premier accès d'un nœud $\eta \in V_\varsigma^\omega \cap$ **Obj** au rang $k$ est donnée par :* $\mathcal{T}(\eta_k, \varsigma, \lceil \mathcal{G}_\varsigma^\omega \rceil) \triangleq \mathcal{T}(obj_k) + [moy_{\eta_{k-1,i} \in Sol_k} \mathcal{T}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_\varsigma^\omega \rceil)]$

4. *Le délais de premier accès d'un nœud $\eta \in V_\varsigma^\omega \cap$ **LS** au rang $k$ est donnée par :* $\mathcal{T}(\eta_k, \varsigma, \lceil \mathcal{G}_\varsigma^\omega \rceil) \triangleq \sum_{\eta_{k-1,i} \in Obj_k} \mathbb{1}_\varsigma(\text{cible}(\eta_{k-1})) \cdot \mathcal{T}(\eta_{k-1,i}, \varsigma, \lceil \mathcal{G}_\varsigma^\omega \rceil)$

*Démonstration.* Le principe de la preuve est le même que pour les probabilités. En effet, pendant le calcul du délai de réaliser chaque objectif, nous ne prenons pas en compte les effets des objectifs concurrents. Ce qui a pour effet d'accélérer la réalisation des objectifs d'intérêt. Le résultat que nous obtenons à la fin est une résultante des approximations effectuées pendant le calcul du délai à chaque objectif du graphe. □

Comme dans le cas du calcul des probabilités, le calcul des délais moyens se fait de bas en haut partant d'un graphe $\lceil \mathcal{G}_\varsigma^\omega \rceil$. Nous partons toujours du principe qu'aux feuilles d'un Graphe de Causalité Locale Quantifié $\lceil \mathcal{G}_\varsigma^\omega \rceil$, le système a atteint un point fixe. À ce point fixe, il est donc possible de calculer les délais moyens nécessaires pour franchir des transitions et les propager en remontant le graphe jusqu'à atteindre la racine. Ainsi, le délai moyen nécessaire pour observer un nœud dans $V_\varsigma^\omega \cap$ **Syn** (nœud de synchronisation) est la moyenne des délais pour observer ses fils (états locaux). Le délai moyen pour observer un nœud dans $V_\varsigma^\omega \cap$ **Sol** (nœud solution) qui n'est pas une feuille est le délai moyen de l'état local associé à la solution. Le délai moyen pour observer un nœud dans $V_\varsigma^\omega \cap$ **Obj** (nœud objectif) est le délai moyen de ses solutions associées, plus le délai nécessaire la réalisation de l'objectif. Enfin, le délai d'un nœud état local est le délai moyen de ses objectifs associés.

Comme pour le cas des probabilités, l'hypothèse que nous faisons de ne pas prendre en compte l'influence des objectifs concurrents a pour conséquence d'accélérer la réalisation des objectifs d'intérêt. Ce qui permet d'avoir une délai optimiste, c'est-à-dire une borne inférieure du délai d'observer un comportement.

*Exemple.* illustration du calcul des délais.

Nous illustrons ici la mise en œuvre de l'estimation des délais d'observer un ensemble de scénarios dans un réseau d'automates asynchrones ($\mathcal{SAN}$). Nous nous appuyons sur l'exemple de la figure 4.4 en page 86 pour construire des Graphes de Causalité Locale Quantifiés. Nous gardons les mêmes modifications que pour l'illustration du calcul des probabilités.

La figure 4.9 donne un exemple d'un Graphe de Causalité Locale Quantifié (section 4.5.2) pour l'approximation d'une borne inférieure des délais de l'ensemble des scénarios pour l'accessibilité de $d_2$ dans le $\mathcal{SAN}$ de la figure 4.4 partant de l'état initial $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$.

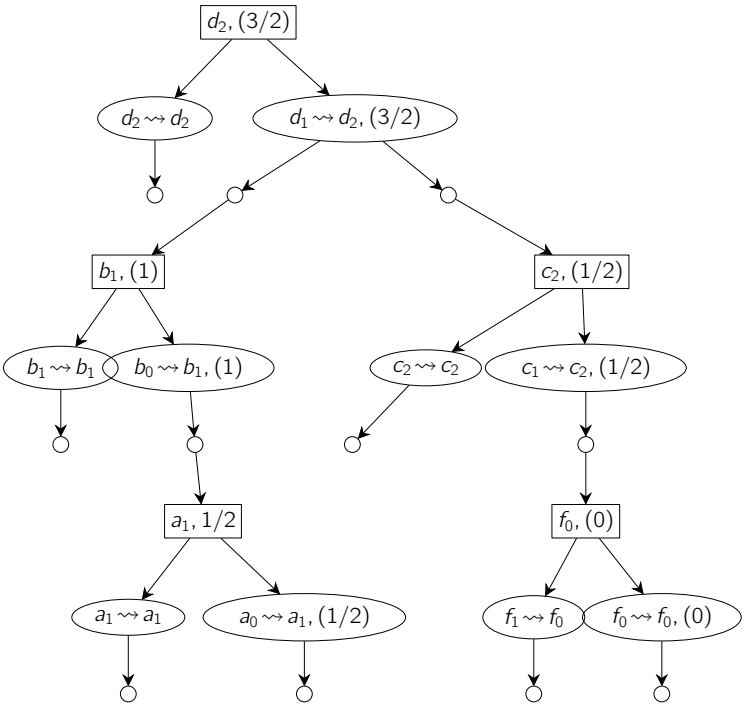



Figure 4.9 : Exemple de Graphe de Causalité Locale Quantifié pour l'estimation d'une borne inf des délais de l'ensemble des scénarios pour l'accessibilité de $d_2$ depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$. Les carrés représentent les états locaux, les cercles représentent les objectifs. Les petits cercles représentent les solutions. Les chiffres entre parenthèses de chaque nœud représentent les probabilités.

### 4.5.3 Approximation limites des probabilités et des délais d'accessibilité

Dans cette section, nous allons nous intéresser à une évaluation de la probabilité et du délai d'accessibilité des scénarios extrêmes. Ces scénarios peuvent être le plus ou le moins probable, le plus ou le moins rapide au sens de *délais de premier passage*.

Ces cas de figures, correspondent à des situations que nous pouvons qualifier de *pire des cas*. L'idée pour retrouver ces scénarios est de parcourir le *graphe de causalité quantifié* et de ne retenir que les chemins qui correspondent à la fonction de sélection de chemins utilisée pendant le parcours. La fonction de parcours est définie selon le cas. Si nous nous intéressons par exemple à la probabilité du scénario le moins probable, à chaque étape du parcours, nous utilisons la fonction *min* pour choisir les nœuds à garder dans le chemin. Si par contre nous nous intéressons au scénario qui met le plus de temps à se réaliser, nous utilisons la fonction *max* pour choisir les nœuds à garder dans le chemin.

#### Exemples de graphes de causalité locale quantifié pour des fonctions extrêmes

La figure 4.10 illustre des exemples de chemins extraits des Graphes de Causalité Locale Quantifiés (section 4.5.3) pour l'estimation de la probabilité du scénario le plus probable (à gauche) et l'estimation du délai du chemin le plus rapide (à droite) pour l'accessibilité de $d_2$ dans $\mathcal{SAN}$ figure 4.4 depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$.

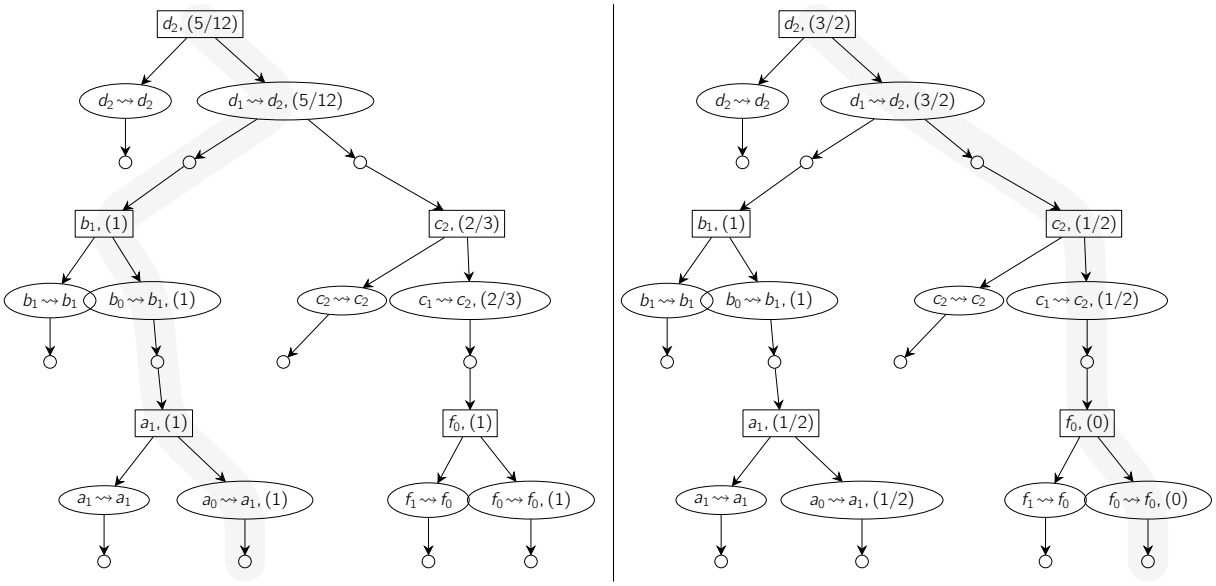



Figure 4.10 : Exemples de chemins extraits des graphes de causalités locaux quantifiés (haut) pour la probabilité du scénario le plus probable pour l'accessibilité de $d_2$ depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$ (bas) pour le délais du scénario le plus rapide pour l'accessibilité $d_2$ depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$.

La figure 4.11 illustre des exemples de chemins extraits des Graphes de Causalité Locale Quantifiés (section 4.5.3) pour l'estimation de la probabilité du scénario le plus probable (à gauche) et l'estimation du délai du chemin le plus rapide (à droite) pour l'accessibilité de $d_2$ dans $\mathcal{SAN}$ figure 4.4 depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$.

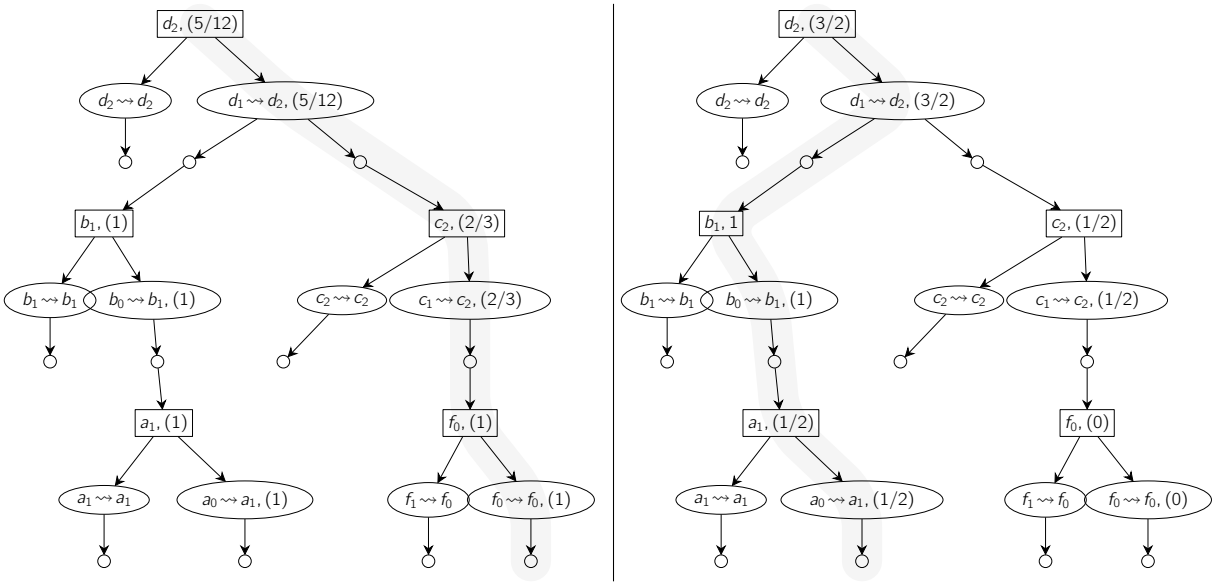



Figure 4.11 : Exemples de chemins extraits des Graphes de Causalité Locale quantifiés (haut) pour le chemin du scénario le moins probable pour l'accessibilité de $d_2$ depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$ (bas) pour le chemin du scénario le moins rapide pour l'accessibilité $d_2$ depuis $\langle a_0, b_0, c_1, d_1, e_1, f_0 \rangle$.

## 4.6 Discussion

Nous avons proposé dans ce chapitre une analyse statique efficace des propriétés quantitatives (probabilités et délais) dans les réseaux d'automates stochastiques. Cette analyse est basée sur une interprétation abstraite de la dynamique stochastique qui a pu être développée dans ce chapitre. Cette interprétation permet de dériver des structures sur lesquelles nous pouvons faire la preuve de l'existence d'une valeur non nulle de la probabilité d'observer une séquence d'objectifs et de l'existence d'une valeur finie du délai d'observer un comportement. De même, par un parcours orienté de la structure abstraite (quand elle ne contient pas de boucle fermée), nous réussissons à extraire des chemins vérifiant une certaine propriété. Nous avons par exemple montré qu'il était possible de connaître le chemin le plus/moins rapide partant d'un état initial donné à un but spécifique. De même, nous avons aussi montré que nous pouvons avoir le chemin le plus/moins probable. De plus, les valeurs obtenues à chaque nœud de la structure peuvent indiquer si le nœud en question est un *nœud critique*. Un nœud ayant par exemple une trop faible probabilité peut être considéré comme critique pour la réalisation d'un scénario.

La mise en œuvre de nos estimations quantitatives est basée sur des structures abstraites dont la taille reste polynomiale selon le nombre d'états locaux par automates. Le calcul des propriétés quantitatives est, quant à lui, polynomial selon le nombre d'automates et exponentiel selon le nombre d'états locaux par automates (ce qui est généralement le cas). De ce fait, nos estimations quantitatives restent bien efficaces quand le nombre d'états locaux est limité par automate.

Cependant, l'efficacité est obtenue au prix d'une ignorance volontaire d'une partie du réseau qui n'est pas explorée. Mais le calcul des probabilités suppose que les pré-conditions associées aux transitions qui « connectent » la partie du réseau explorée à celle qui n'est pas explorée, sont toujours vérifiées. Ce qui n'est pas vrai dans la réalité. Cette supposition conduit à une estimation d'une borne inf de la probabilité.

Des travaux futurs pourraient par exemple étendre l'analyse statique développée dans le cadre de ce chapitre pour l'analyse des modèles paramétriques des systèmes stochastiques et concurrents. En effet, les modèles paramétrisés sont d'un intérêt évident même comme les analyses associées à ces modèles restent des problèmes difficiles. Par exemple, si nous considérons un réseau d'automates stochastiques paramétriques, déterminer s'il existe une plage de valeur des paramètres du réseau, telle que le système peut atteindre un état donné partant d'un état initial est un problème indécidable. L'idée serait de se servir des méthodes par approximation pour pouvoir donner des outils d'aide à la décision sur les valeurs des paramètres du modèle.

Il serait intéressant d'appliquer cette démarche sur des systèmes de grande taille en particulier les systèmes biologiques, pour lesquels nous avons une connaissance des taux des transitions des transitions. Cela pourrait permettre l'identification des chemins et des nœuds vraisemblablement critiques.

# Chapitre 5

# Identification des bifurcations dans les réseaux d'automates

Dans le but d'analyser les processus de différentiation dans les systèmes dynamiques complexes, ce chapitre présente l'identification des états et des transitions qui jouent un rôle cruciale pour préserver ou empêcher la survenu d'un évènement. Dans le contexte des réseaux d'automates non déterministes, nous proposons une identification des *bifurcations* par analyse statique. Ici les *bifurcations* correspondent aux transitions après les quelles un but n'est plus accessible. De telles transitions sont de fait de bonnes candidates pour contrôler l'accessibilité du but, par exemple en modifiant leur fréquence. Notre méthode combine à la fois la programmation par ensemble de réponses (Answer Set Programming) et l'analyse statique des propriétés d'accessibilité pour fournir à la fois une sous- et sur-approximation de toutes les bifurcations existantes.
Nous avons illustré notre approche discrète de l'analyse des bifurcations sur plusieurs modèles des systèmes biologiques pour lesquels nous avons identifié les transitions qui impactent l'accessibilité des états locaux associés. Nous avons illustré le passage à l'échelle de notre méthode sur un exemple de 101 composants et 221 interactions.

L'identification des bifurcations dans les réseaux de régulation biologique a été publié dans (Fitime, Roux, Guziolowski & Paulevé, 2016).

## 5.1 Préliminaires

Une des questions clés de l'analyse, de la vérification et du contrôle des systèmes est de pouvoir garantir qu'un système dont on étudie le fonctionnement à partir de conditions initiales données, ne va pas évoluer vers des comportements non souhaités de sa dynamique. Ces comportements non souhaités concernant la survenue de situations critiques peuvent entraîner de graves problèmes. Par exemple, dans le cas des systèmes biologiques humains

en particulier, les comportements non souhaités peuvent être des maladies graves comme le cancer notamment.

Afin de proposer des solutions pour garantir le comportement des systèmes, plusieurs travaux ont été effectués sur différents types de systèmes avec différentes approches de modélisation (Moon, Powers, Burch & Clarke, 1992; Moon & Macchietto, 1994; Moon, 1992). Dans le domaine particulier des systèmes biologiques, (Klamt & Gilles, 2004; Samaga, Kamp & Klamt, 2010) ont introduit la notion d'ensemble minimal d'intervention. Ce concept permet de choisir une stratégie pour provoquer un comportement souhaité. Plus loin, (Paulevé et al., 2013; Acuna, Chierichetti, Lacroix, Marchetti-Spaccamela, Sagot & Stougie, 2009) ont proposé des approches basées sur les *Cuts sets* pour identifier les nœuds/réactions dont la suppression permet d'empêcher d'observer certains comportements.

Nous souhaitons partir d'une des propriétés essentielles de la dynamique des systèmes discrets qui stipule que l'ensemble d'états est organisé en bassins d'attractions. Prévenir certains comportements peut donc revenir à isoler certains états ou transitions du graphe des états qui représentent les comportements non souhaités. De par la complexité inhérente à l'explosion combinatoire de l'espace d'états, l'analyse des dynamiques des Réseaux de Régulation Biologique (RRB) requiert donc des méthodes innovantes pour contourner cette explosion combinatoire.

Le but de ce chapitre est de proposer une méthode efficace qui permette de prévenir des comportements non souhaités dans les systèmes. Cette méthode est basée sur l'analyse statique (Cousot & Cousot, 1977) et des résultats efficaces développés pour l'accessibilité dans les réseaux d'automates (Paulevé et al., 2012; Folschette et al., 2015).

Nous sommes donc confrontés au problème de l'accessibilité d'un état local donné à partir d'un état initial global dans un réseau d'automates. Nous cherchons à identifier les transitions qui peuvent compromettre cette accessibilité. De fait, si ces transitions sont effectuées dans la dynamique du réseau, l'état local à atteindre ne peut plus être atteint. De telle transitions correspondent à des transitions que nous appelons des *transitions de bifurcation* ou tout simplement ce que nous appelons des *bifurcations*. Ces transitions ne peuvent être effectuées qu'à des états bien précis (état juste avant l'application de la transition de bifurcation). L'idée est donc de proposer une spécification formelle qui permette de caractériser et d'identifier de telles transitions et les états dans lesquels cela est possible.

Pour identifier ces transitions de bifurcation, nous commençons dans une première étape par construire un ensemble d'états à partir desquels il n'est pas possible d'atteindre le but recherché. Sachant que cette construction est un problème qui est PSPACE-complet (Cheng et al., 1995), nous nous appuyons sur les résultats efficaces développés pour l'accessibilité dans les réseaux d'automates. Puis, des états à partir desquels il n'a plus possible d'atteindre le but, nous identifions dans une deuxième étape les transitions (transitions candidates) qui mènent à ces états (ceux à partir desquels il n'est plus possible d'atteindre le but) en un pas. La dernière étape consiste à vérifier que des états où les transitions candidates peuvent être jouées, qu'il existe un chemin partant de l'état initial vers ces états et qu'il existe un chemin partant de ces états vers l'état local à atteindre. La grande originalité de notre approche réside dans la capacité à utiliser des approximations développées par Paulevé (2011). Ce sont des méthodes d'analyse statique qui garantissent certaines propriétés d'accessibilité avec une complexité raisonnable (polynomiale selon le nombre d'automates et exponentielle selon le nombre d'états locaux par automate).

Parce que notre méthode est basée sur des approximations supérieure et inférieure de

la propriété d'accessibilité, nous avons proposé deux approches : une première qui consiste à proposer une approximation inférieure de l'ensemble des transitions de bifurcation. Cette démarche n'autorise pas à avoir de faux positifs. Nous pouvons cependant en manquer quelques transitions de bifurcation. La deuxième approche, par contre, s'autorise des faux positifs. Ce qui oblige à effectuer une vérification exacte de chaque transition trouvée pour éliminer celles qui ne correspondent pas à des transitions de bifurcation.

Nous avons implémenté notre démarche en nous servant de L'Answer Set Programming (ASP) (Baral, 2003) entre autre pour tirer avantage des capacités de ce paradigme de programmation à proposer de très bonnes heuristiques pour la résolution des problèmes difficiles.

Ce chapitre est structuré comme suit. La section 5.2 présente les principaux outils nécessaires à la lecture des autres sections de ce chapitre. La section 5.3 introduit quelques définitions nécessaires à la construction de notre raisonnement et principalement la définition de la notion de bifurcation. Toujours dans la section 5.3, nous présentons l'idée générale et les principales contributions de ce chapitre. La section 5.4 et la section 5.5 présentent respectivement les approximations inférieures et supérieures de l'ensemble des transitions de bifurcation dans les réseaux d'automates. Les applications effectuées sur des systèmes biologiques et la capacité de notre approche à s'appliquer sur de grands modèle sont reportées dans la section 6.3 en page 141 au chapitre 6. Nous montrons à la section 6.3.4 en page 147 une utilité de l'identification des bifurcations de transition. Enfin, la section 5.8 en page 129 résume et discute les contributions de ce chapitre.

## 5.2 Outils pour les sections suivantes

### 5.2.1 Rappels de quelques définitions

Nous faisons ici un petit rappel de quelques définitions sur lesquelles nous construisons notre raisonnement. Nous présentons donc une définition des réseaux d'automates, des notions de transition locale et de chemin local.

Un réseau d'automate regroupe un ensemble fini d'automates. Chaque automate possède des états locaux et des transitions locales à chaque automate conditionnées par la présence d'états locaux ou non des autres automates.

**Définition 5.1** (Réseau d'Automates $\mathcal{N} = (\Sigma, S, T)$)**.** Un *Réseau d'Automates* est défini par le tuple $\mathcal{N} = (\Sigma, S, T)$ où

- $\Sigma$ est l'ensemble fini d'automates ;
- Pour chaque $a \in \Sigma$, $S(a) = \{a_0, \ldots, a_{k_a}\}$ est l'ensemble fini des états locaux de l'automate $a$ ; $S \triangleq \prod_{a\in\Sigma} S(a)$ est l'ensemble fini des états globaux ;
  $\mathbf{LS} \triangleq \bigcup_{a\in\Sigma} S(a)$ est l'ensemble de tous les états locaux.
- $T = \{a \mapsto T_a \mid a \in \Sigma\}$, où $\forall a \in \Sigma, T_a \subseteq S(a) \times 2^{\mathrm{LS}\setminus S(a)} \times \mathbb{R} \times S(a)$ avec $(a_i, \ell, r, a_j) \in T_a \Rightarrow a_i \neq a_j$ et $\forall b \in \Sigma, |\ell \cap S(b)| \leq 1$, est une correspondance des automates vers l'ensemble des transitions locales. $r$ est le taux de la transition.

Nous écrivons $a_i \xrightarrow{\ell,r} a_j \in T \triangleq (a_i, \ell, r, a_j) \in T(a)$.
et $a_i \to a_j \in T \triangleq \exists \ell \in 2^{\mathrm{LS}\setminus S(a)}, a_i \xrightarrow{\ell} a_j \in T$. Étant donné une transition $t = a_i \xrightarrow{\ell} a_j \in T$, nous notons $\mathrm{orig}(t) \triangleq a_i$, et $\mathrm{dest}(t) \triangleq a_j$, $\mathrm{enab}(t) \triangleq \ell$, ${}^\bullet t \triangleq \{a_i\} \cup \ell$, et $t^\bullet \triangleq \{a_j\} \cup \ell$. La relation globale de transition $\to \subseteq S \times S$ est définie par :

$$s \to s' \triangleq \exists \ell \in \ : \forall a_i \in {}^\bullet\ell, s(a) = a_i \wedge \forall a_j \in \ell^\bullet, s'(a) = a_j \\ \wedge \forall b \in \Sigma, S(b) \cap {}^\bullet\ell = \emptyset \Rightarrow s(b) = s'(b).$$

La figure 5.1 représente un réseau d'automates $(\Sigma, S, T)$ de 3 automates ($\Sigma = \{a, b, c\}$), avec $S(a) = \{a_0, a_1, a_2\}$, $S(b) = \{b_0, b_1\}$, $S(c) = \{c_0, c_1, c_2\}$, et 8 transitions locales définies comme suit :

$$T(a) = \{t_1 = a_1 \xrightarrow{\emptyset} a_0, t_2 = a_0 \xrightarrow{b_0} a_1, t_3 = a_0 \xrightarrow{b_0, c_0} a_2\}$$
$$T(b) = \{t_4 = b_0 \xrightarrow{\emptyset} b_1, t_5 = b_1 \xrightarrow{a_0} b_0\}$$
$$T(c) = \{t_6 = c_0 \xrightarrow{a_1} c_1, t_7 = c_1 \xrightarrow{b_1} c_0, t_8 = c_1 \xrightarrow{b_0} c_2\}$$

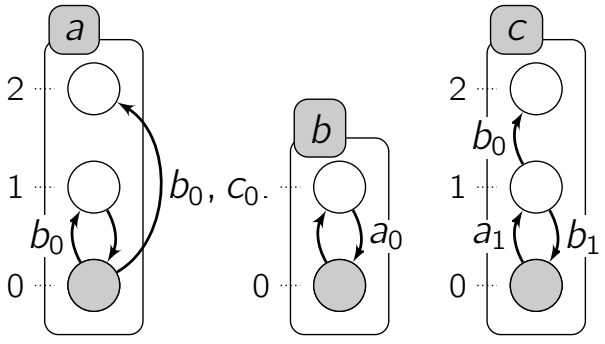



Figure 5.1 : Un exemple de réseaux d'automates. Les automates sont représentés par les boîtes labélisées, les états locaux sont représentés par des cercles. Les états locaux sont associés à leur identifiant grâce aux pointillés. Par exemple, l'état local $a_0$ est le cercle identifié 0 dans la boîte $a$. Une transition est un arc orienté entre deux états locaux du même automate. Les transitions peuvent être labélisées par un ensemble d'états locaux des autres automates. Enfin, les états locaux en gris représentent l'état global du système. $\langle a_0, b_0, c_0 \rangle$.

Une transition locale $t = a_i \xrightarrow{\ell} a_j \in T$ est applicable ou jouable dans l'état global $s \in S$ quand $a_i$ et tous les états locaux dans $\ell$ sont dans $s$. Jouer une transition locale noté $s \cdot t$,

remplace l'état local de $a$ par $a_j$ (définition 5.2). Ce qui permet d'obtenir la transition $s \xrightarrow{t} s'$ où $s' = s \cdot t$.

Dans ce chapitre nous faisons l'hypothèse que les transitions sont effectuées avec une sémantique non déterministe et *asynchrone* des réseaux d'automates. Cette sémantique suppose qu'à un instant donné, une seule transition locale peut être effectuée. Ce qui implique que seul un automate peut changer d'état local à travers une transition entre deux états globaux. On note aussi que plusieurs transitions locales différentes peuvent être appliquées au même état ce qui peut entraîner des dynamiques nombreuses. Le choix des transitions est *non-déterministe*. C'est une sémantique centrée transitions par opposition aux sémantiques centrées fonctions (modèle de Thomas)

Un état global $s'$ est accessible depuis un état $s$, ce que nous notons $s \to^* s'$, si et seulement si il existe une séquence (potentiellement vide) de transitions conduisant de $s$ à $s'$.

**Définition 5.2** (Transition, accessibilité)**.** Soit un état $s \in S$ et une transition locale $t = a_i \xrightarrow{\ell} a_j \in T$ telle que $s(a) = a_i$ et $\forall b_k \in \ell$, $s(b) = b_j$, $s \cdot t$ est l'état $s$ où $a_i$ a été remplacé par $a_j$ :

$$\forall b \in \Sigma, (s \cdot t)(b) = \begin{cases} a_j & \text{si } b = a \\ s(b) & \text{si non} \end{cases}$$

Nous écrivons donc $s \xrightarrow{t} s'$ où $s' = s \cdot t$. La relation binaire d'accessibilité $\to^* \subseteq S \times S$ satisfait

$$s \to^* s' \overset{\Delta}{\Leftrightarrow} s = s' \vee \exists t \in T : s \xrightarrow{t} s'' \wedge s'' \to^* s'$$

Nous définissons aussi la notion de *chemin local* entre deux états locaux $a_i$,$a_j$ d'un même automate $a$ qui est une notion clé de notre raisonnement.

Nous nous appuyons aussi sur la notion d'*objectif* que nous notons $a_i \rightsquigarrow a_j$.

Nous notons local-paths($a_i \rightsquigarrow a_j$) l'ensemble des chemins acycliques des transitions locales entre $a_i$ et $a_j$. La définition 5.3 donne une formalisation de local-paths dans laquelle nous utilisons les notations suivantes : pour une transition locale $t = a_i \xrightarrow{\ell} a_j \in T$, $\text{orig}(t) \overset{\Delta}{=} a_i$, $\text{dest}(t) \overset{\Delta}{=} a_j$, $\text{enab}(t) \overset{\Delta}{=} \ell$ ; $\varepsilon$ correspond à une séquence vide, et $|\eta|$ est la longueur de la séquence $\eta$.

**Définition 5.3** (local-paths)**.** Etant donné un objectif $a_i \rightsquigarrow a_j$,

- si $i = j$, local-paths($a_i \rightsquigarrow a_i$) $\overset{\Delta}{=} \{\varepsilon\}$ ;
- si $i \neq j$, une séquence $\eta$ de transitions dans $T(a)$ est dans local-paths($a_i \rightsquigarrow a_j$) si et seulement si $\text{orig}(\eta^1) = a_i$, $\text{dest}(\eta^{|\eta|}) = a_j$, $\forall n, 1 \leq n < |\eta|$, $\text{dest}(\eta^n) = \text{orig}(\eta^{n+1})$, et $\forall n, m, |\eta| \geq n > m \geq 1, \text{dest}(\eta^n) \neq \text{orig}(\eta^m)$.

Nous écrivons $t \in \eta \overset{\Delta}{\Leftrightarrow} \exists n, 1 \leq n \leq |\eta| : \eta_n = t$. Étant donné un chemin local $\eta$, $\widetilde{\eta}$ désigne l'union des conditions de toutes les transitions locales qui le compose :

$$\widetilde{\eta} \overset{\Delta}{=} \bigcup_{n=1}^{|\eta|} \text{enab}(\eta_n)$$

## 5.3 Définition de la bifurcation

Une *bifurcation* est une transition telle que étant donné le problème d'accessibilité d'un état local $g$ (tel que défini au chapitre 4, si cette transition est franchie, le système ne peut plus atteindre cet état local $g$. Nous nous appuyons sur la figure figure 5.1 en page 112 et sur son graphe d'états associé figure 5.2 pour illustrer la notion de bifurcation dans les réseaux d'automates.

La figure 5.2 représente le graphe de transition associé au réseau d'automates de la figure 5.1 en page 112. Il décrit les transitions possibles entre les différents états du modèle. L'état initial est $\langle a_0, b_0, c_0\rangle$ et le but à atteindre est l'état local $a_2$. Ce but est représenté en bleu et appartient à deux états du système $\langle a_2, b_0, c_0\rangle$, $\langle a_2, b_1, c_0\rangle$. Nous pouvons donc regrouper l'ensemble des états du système en deux groupes : ceux connectés (en gris dans la figure 5.2) au but que l'on souhaite atteindre et ceux qui ne le sont pas. À partir de certains états coloriés en gris, il existe des transitions (voir la transition $t_8$ en rouge dans la figure 5.2) qui permettent d'atteindre le sous ensemble d'états à partir duquel il n'est plus possible d'accéder à l'état $a_2$. C'est typiquement ce type de transition que nous appelons des *transitions de bifurcation* et que nous nous proposons d'identifier dans la suite de ce travail.

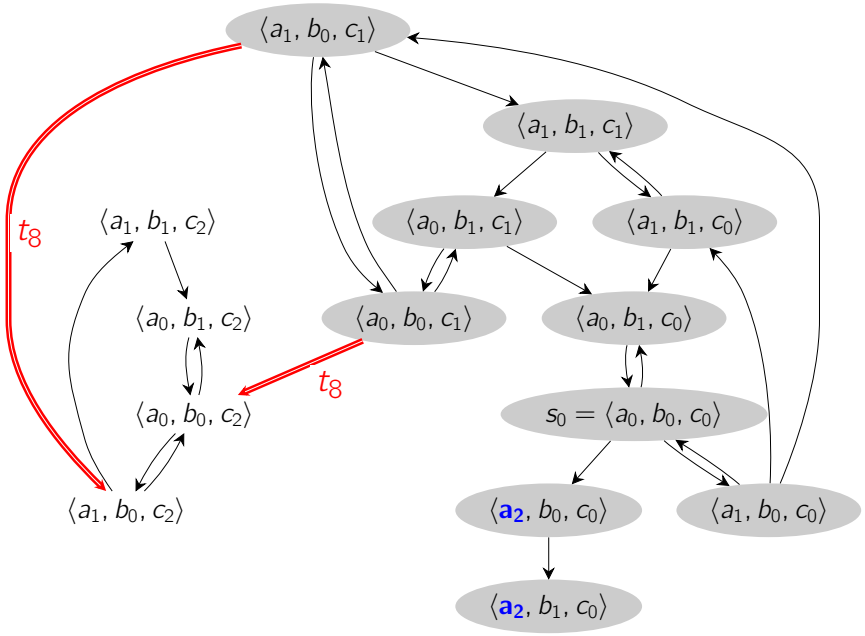



Figure 5.2 : Le graphe de transition associé au réseau de la figure 5.1 avec pour état initial $s_0 = \langle a_0, b_0, c_0\rangle$. Le but à atteindre est l'état local $a_2$ en gras qui est colorié en bleu ; les états connectés au but qu'on veut atteindre sont en gris ; les transitions ayant des traits doubles coloriées en rouge et labélisées correspondent à des transitions de bifurcations pour le but à atteindre $a_2$.

### 5.3.1 Définition formelle de la notion de bifurcation

Dans cette section, nous présentons de façon formelle la notion de bifurcation introduite dans la section 5.3. Pour cela nous nous appuyons sur une représentation à base d'ensembles d'états pour représenter les états qui vérifient les mêmes propriétés. De cette

façon, nous pouvons les construire en s'appuyant sur les propriétés qui peuvent être vérifiées efficacement.

La définition 5.4 formalise la notion de bifurcation dans les réseaux d'automates et nous proposons une illustration à la figure 5.3. Le but à atteindre est spécifié par un état local $g_1$ ($S_g = \{s \in S \mid s(g) = 1\}$).

**Définition 5.4** (Bifurcation)**.** Soit un réseau d'automates $(\Sigma, S, T)$, un état global $s_0 \in S$ et un état local $g_1$ à atteindre avec $g \in \Sigma$ et $g_1 \in S(g)$, une *bifurcation* est une transition $s_b \xrightarrow{t_b} s_u$ de l'automate avec $s_b, s_u \in S$ et $t_b \in T$, telle que $s_0 \rightarrow^* s_b$ et $\forall s' \in S$ où $s_u \rightarrow^* s'$, $s'(g) \neq g_1$.

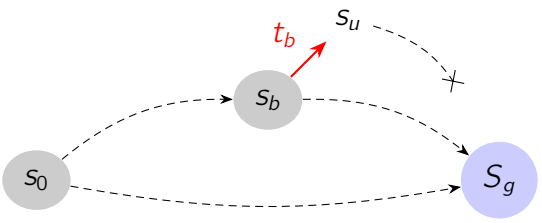



Figure 5.3 : Illustration générale d'une bifurcation. $s_0$ est l'état initial, $S_g$ est l'ensemble d'états qui contiennent le but état local à atteindre. Les flèches en interrompues représentent une séquence (potentiellement vide) de transitions. La flèche rouge en trait fort est une bifurcation d'un état global $s_b$ à $s_u$, et $t_b$ est la transition locale associée.

Après avoir proposé la définition 5.4 pour formaliser la notion de bifurcation, nous proposons maintenant un ensemble de définitions qui nous permettent d'identifier l'ensemble des transitions de bifurcations pour un problème d'accessibilité donné.

Nous commençons par proposer la définition d'un ensemble d'états accessibles depuis un état global initial donné $s_0$. Il s'agit de l'ensemble des états avec lesquels $s_0$ communique.

**Définition 5.5** ($R_{s_0}$ Ensemble d'états accessibles depuis $s_0$)**.** Soit un réseau d'automates $(\Sigma, S, T)$, un état global $s_0 \in S$, l'ensemble des états accessibles depuis $s_0$ est donné par :
$R_{s_0} = \{s \in S : s_0 \rightarrow^* s\}$.

Puis nous définissons de façon analogue l'ensemble d'états à partir desquels il est possible d'accéder à un état local donné $g$. Il s'agit de l'ensemble des états qui communique avec $g$.

**Définition 5.6** ($R^g$)**.** Soit un réseau d'automates $(\Sigma, S, T)$, un état global $s_0 \in S$ et un état local $g$ à atteindre avec $s_g \in \Sigma$ et $g \in S(s_g)$, $R^g$ est l'ensemble d'états à partir desquels il est possible d'atteindre l'état local $g$.
$R^g = \{s \in S : s \rightarrow^* g\}$.

À partir de la définition 5.5 et de la définition 5.6, le lemme suivant met en exergue une propriété simple des ensembles $R_{s_0}$ et $R^g$.

**Propriété 5.1.** *Soient $s_i, s_j \in S$. Si $s_j \in R_{s_i}$, alors $R_{s_j} \subseteq R_{s_i}$. De façon analogue, si $s_j \in R^{s_i}$, alors $R^{s_j} \subseteq R^{s_i}$.*

*Démonstration.* La preuve du lemme 5.1 découle directement de la propriété de transitivité de la relation de transition $\rightarrow^*$ (voir définition 5.2). Supposons que $s_j \in R_{s_i}$, ce qui veut dire

que nous avons la relation $s_i \rightarrow^* s_j$ (d'après la définition 5.5), alors $\forall s_k \in R_{s_j}$, c'est-à-dire $s_j \rightarrow^* s_k$, nous avons $s_i \rightarrow^* s_k$, ce qui signifie que $s_k \in R_{s_i}$ d'où $R_{s_j} \subseteq R_{s_i}$.

De la même façon, si $s_j \in R^{s_i}$, alors nous avons la relation $s_j \rightarrow^* s_i$ (d'après la définition 5.6), alors $\forall s_k \in R^{s_j}$, c'est-à-dire $s_k \rightarrow^* s_j$, nous avons $s_k \rightarrow^* s_i$, ce qui implique $R^{s_j} \subseteq R^{s_i}$. □

Le troisième ensemble d'états que nous construisons est un ensemble d'états $U^g$ à partir desquels il n'est pas possible d'atteindre un but état local $g$, nous présentons cet ensemble d'états à la définition 5.6 à la page précédente. Nous pouvons constater que si l'ensemble $U^g$ est non vide, cela suppose qu'il existe au moins un sous ensemble d'états $A$ dans l'ensemble des états $S$, tel que $S(g) \notin A$ et que pour tout $s \in A$, $s \not\rightarrow^* g$. Ces sous ensembles $A$ sont des attracteurs (voir la définition 5.7) du système. Aussi, avant de donner la définition formelle de $U^g$, nous donnons tout d'abord la définition d'un attracteur. Un attracteur est un sous ensemble d'états dans lequel toutes les trajectoires partant des états de l'attracteur finissent dans un état qui appartient à l'attracteur et pour tout état appartenant à l'attracteur, si cet état est visité une fois, alors la probabilité qu'il soit atteint une nouvelle fois est 1.

**Définition 5.7** (Attracteur)**.** Un attracteur est un ensemble d'états $A \subseteq S$ qui vérifient les deux propriétés suivantes :

- $\forall s \in A$, $R_s = A$.
- Pour tout état dans $A$, si cet état est atteint une fois, alors la probabilité qu'il soit atteint une nouvelle fois est 1 d'après (Hachtel, Macii, Pardo & Somenzi, 1996).

Un état $s \in A$, avec $A$ attracteur est un *état absorbant*. Si un état n'est pas absorbant, alors il est *transitoire*.

**Définition 5.8** ($U^g$)**.** $U^g$ est l'ensemble d'états à partir desquels il n'est pas possible d'accéder à l'état local $g$.
$U^g = \{s \in S : s \not\rightarrow^* g\}$.

Un constat immédiat est que, $U^g$ contient tous les états du système qui appartiennent à un attracteur ne contenant pas $g$. En effet, si un état $S(g) \in A$ avec $A$ étant ici un attracteur alors $S(g) \in U^g$. De plus, si $S(g)$ n'appartient pas à un attracteur, alors $U^g$ est égal à la réunion de tous les attracteurs du système.

La réunion de tous les attracteurs ne contenant pas un état $S(g)$ est contenue dans $U^g$.

**Proposition 5.1.** *$\forall A_i$ tel que $S(g) \notin A_i$, $\bigcup_{i=0}^{n} A_i \subseteq U^g$.*

*Démonstration.* Pour prouver $\bigcup_{i=0}^{n} A_i \subseteq U^g$, tel que $S(g) \notin A_i$, nous avons les deux cas de figure suivants :

- Soit $S(g)$ appartient à un état transitoire, $\forall s \in \bigcup_{i=0}^{n} A_i$, $s \not\rightarrow^* g$. D'où $\bigcup_{i=0}^{n} A_i \subseteq U^g$.
- soit $S(g)$ n'appartient pas à un état transitoire, alors il existe un unique attracteur $A_i$ tel que $S(g) \in A_i$. D'où nous pouvons en déduire que $\bigcup_{j=0, i \neq j}^{n} A_j \backslash A_i \subseteq U^g$.

□

**Définition 5.9** (état de bifurcation)**.** Soit le réseau d'automates $(\Sigma, S, T)$, un état initial $s_0$, et un état local $g$ qu'on souhaite atteindre. Un état de bifurcation pour l'accessibilité de l'état local $g$ depuis l'état initial $s_0$, est un état $s_b$ tel que $s_0 \rightarrow^* s_b \quad \wedge \quad \exists t_b \in T \quad s_b \xrightarrow{t_b} s_u \wedge s_u \in U^g \quad \wedge s_b \rightarrow^* g$.

**Proposition 5.2.** *Pour tout état $s_b$ tel que $\exists t_b \in T \quad s_b \xrightarrow{t_b} s_u \wedge s_u \in U^g \quad \wedge s_b \rightarrow^* g$, $s_b$ est un état transitoire.*

*Démonstration.* Puisque $s_u \in U^g$, alors : soit $s_u$ appartient à un attracteur et comme par définition (définition 5.9) $s_b \rightarrow^* g$ alors $s_b$ est un état transitoire ; soit $s_u$ n'appartient pas à un attracteur et comme $\exists t_b \in T \quad s_b \xrightarrow{t_b} s_u$ alors $s_b$ est bien un état transitoire. □

De plus, puisque l'état $s_0$ communique avec l'état $s_b$, alors il est facile de conclure que puisque $s_b$ est transitoire alors $s_0$ l'est aussi. D'où nous avons le lemme suivant :

**Proposition 5.3.** *$s_0$ est un état transitoire.*

Conformément à la définition 5.9 à la page ci-contre, nous proposons la définition 5.10 pour un ensemble d'états de bifurcations. Cette définition formalise la notion d'ensemble d'états de bifurcation dans un réseau d'automates $(\Sigma, S, T)$, pour l'accessibilité d'un état local $g$ depuis un état initial $s_0$. Cet ensemble est égal à l'ensemble des états $s_b$ tels qu'il existe un chemin de $s_b$ vers $g$ ($s_b \rightarrow^* g$), tels qu'il existe également un chemin de $s_0$ vers $s_b$ ($s_0 \rightarrow^* s_b$) et à partir de chaque état $s_b$, il existe une transition ($t_b$) vers un état $s_u$ à partir duquel il n'existe pas de chemin vers $g$ ($s_u \not\rightarrow^* g$).

**Définition 5.10** (L'ensemble des états de bifurcation pour l'accessibilité de $g$ depuis l'état initial $s_0$ : $B^g_{s_0}$)**.** Soit un réseau d'automates $(\Sigma, S, T)$, un état global $s_0 \in S$ et un état local $g$ à atteindre avec $s_g \in S$ and $g \in S(s_g)$, on définit l'ensemble des états de bifurcation pour l'accessibilité de $g$ depuis l'état initial $s_0$ comme suit :
$B^g_{s_0} = \{s_b \in R^g \cap R_{s_0} \mid \exists s_u \in U^g \wedge \exists t_b \in T : s_b \xrightarrow{t_b} s_u\}$

L'existence des bifurcations pour une accessibilité donnée dans un réseau d'automates $(\Sigma, S, T)$ implique nécessairement qu'il existe des états qui ne communiquent pas avec d'autres états. Ceci suggère donc que le graphe des états de transitions ne forme pas une seule composante fortement connexe et peut donc être réduit en sous-ensembles d'états. Cette intuition amène à proposer le théorème 5.1 qui stipule que, si pour une accessibilité donnée dans un réseau d'automate, le nombre d'états de bifurcation est différent de 0, alors il est possible de diviser l'ensemble des états en deux sous-ensembles d'états dont un sous-ensemble d'états transitoires et un sous-ensemble d'états absorbants qui possède au moins un attracteur.

**Théorème 5.1.** *Si $|B^g_{s_0}| \neq 0$ alors l'ensemble des états du système peut être structuré en au moins deux sous-ensembles dont un sous ensemble des états transitoires et un sous ensemble des états absorbants qui possède au moins un attracteur.*

*Démonstration.* $|B^g_{s_0}| \neq 0 \implies \exists s_b, s_u \mid \exists t_b \in T : s_b \xrightarrow{t_b} s_u$. D'après la définition 5.10, $s_b \in R^g \implies s_b \rightarrow^* g \wedge s_u \in u^g \implies s_u \not\rightarrow^* g$. Nous avons donc une situation où $s_b$ communique avec $g$ et $s_b$ communique avec $s_u$ mais $s_u$ ne communique pas avec $g$. Ce qui implique que $s_u$ ne communique pas avec $s_b$ sinon il y aurait une façon de communiquer avec $g$. Ce qui permet de conclure que les états $s_b$ et $s_u$ n'appartiennent pas à la même composante fortement connexe. D'où l'espace des états peut être réduit en deux sous-ensembles d'états dont un sous-ensemble d'états transitoires et un sous-ensemble d'états absorbants contenant au moins un attracteur. □

De façon analogue à la définition des états de bifurcation, nous définissons l'ensemble des transitions de bifurcation comme l'ensemble des transitions $t_b$ telle qu'il existe un état

$s_b$ accessible depuis $s_0$ et à partir duquel il existe un chemin vers $g$ ($s_b \in B^g_{s_0}$) et un état $s_u \in U^g$ à partir duquel il n'est pas possible d'atteindre $g$ et $s_b \xrightarrow{t_b} s_u$. Nous formalisons cet ensemble avec la définition 5.11.

**Définition 5.11** (Ensemble de transitions de bifurcation ($\mathcal{T}^g_{s_0}$))**.** Soit un réseau d'automates $(\Sigma, S, T)$, un état global $s_0 \in S$ et un état local $g$ à atteindre avec $s_g \in \Sigma$ et $g \in S(s_g)$, l'ensemble des transitions de bifurcation pour l'accessibilité de $g$ depuis l'état initial $s_0$ est défini comme suit :
$\mathcal{T}^g_{s_0} = \{t_b : \exists s_b \in B^g_{s_0}, \exists s_u \in U^g \wedge s_b \xrightarrow{t_b} s_u\}$.

*Exemple.* Partant de l'exemple de la figure 5.2 en page 114, les transitions en rouge étiquetées par $t_8$ correspondent aux transitions de bifurcation pour l'accessibilté de $a_2$ depuis l'état initial $s_0 = \langle a_0, b_0, c_0\rangle$. Aussi, elles constituent donc les éléments de $\mathcal{T}^g_{s_0}$. De même les états sources de ces transitions de bifurcation sont les états de bifurcation. Il s'agit dans le cas de cet exemple des états $\langle a_1, b_0, c_1\rangle, \langle a_0, b_0, c_1\rangle$. Les états en gris sont les états qui appartiennent à $R^{a_2}$ et enfin nous avons l'ensemble $U^{a_2} = \{\langle a_1, b_1, c_2\rangle, \langle a_0, b_1, c_2\rangle\langle a_0, b_0, c_2\rangle, \langle a_1, b_0, c_2\rangle\}$.

Avec ces définitions, nous pouvons exprimer l'identification des bifurcations comme un ensemble de propriétés d'accessibilité qui peuvent être vérifiées. Cependant, quand le réseau d'automates devient grand, l'ensemble des états accessibles peut s'avérer très grand pour les méthodes exactes de vérification. De plus, la complexité du problème d'accessibilité peut être grande (PSPACE-complet) surtout s'il faut effectuer un grand nombre de vérifications par exemple lors de l'énumération de l'ensemble des états initiaux candidats pour la vérification de l'accessibilité d'un but depuis ces états initiaux.

Dans ce chapitre nous nous appuyons sur des approximations développées pour l'accessibilité dans les réseaux d'automates par (Paulevé et al., 2012; Folschette et al., 2015).

## 5.3.2 Idée générale pour l'identification des bifurcations et principales contributions

Dans cette section nous présentons l'idée de notre démarche et résumons les principales contributions apportées dans ce chapitre.

Pour commencer, nous présentons les propriétés que nous tirons à la fois de la *sur-approximation* (OA) (cf. section 4.5.1.1 en page 93) et de la *sous-approximation* (UA) (cf. section 4.5.1.2 en page 96) du problème d'accessibilité :

**Propriété 5.2** (Les propriétés de OA et de UA)**.** *$s \to^* s'$ est vrai seulement si* OA($s \to^* s'$) *est vrai et $s \to^* s'$ est vrai si* UA($s \to^* s'$) *est vrai ; mais la contraposée n'est pas vraie en général :*

$$\text{UA}(s \to^* s') \Rightarrow s \to^* s' \Rightarrow \text{OA}(s \to^* s')$$

### 5.3.2.1 Approche générale

Telle que nous l'avons précédemment définie (voir la définition 5.4 en page 115), une transition locale $t_b \in T$ cause une bifurcation pour l'accessibilité du but $g_1$ depuis l'état $s_0$ si les conditions suivantes sont vérifiées :

$t_b$ est une bifurcation de transition si et seulement si il existe un état $s_b \in S$ tel que

$$\text{(C1)}\ s_u \not\rightarrow^* g_1 \qquad \text{(C2)}\ s_b \rightarrow^* g_1 \qquad \text{(C3)}\ s_0 \rightarrow^* s_b$$

où $s_u = s_b \cdot t_b$, $s \not\rightarrow^* g_1 \overset{\Delta}{\Leftrightarrow} \forall s' \in S, s \rightarrow^* s' \Rightarrow s'(g) \neq g_1$ et $s \rightarrow^* g_1 \overset{\Delta}{\Leftrightarrow} \exists s_g \in S : s_g(g) = g_1 \wedge s \rightarrow^* s_g$.

#### 5.3.2.2 Principales contributions

Dans une construction totale des éléments $s_b \in B^g_{s_0}$, la vérification de l'accessibilité ou non du but depuis chaque $s_b$ candidat ((C1) et (C2)) n'est pas envisageable pour les grands modèles.

Plutôt que d'effectuer une vérification exacte, nous nous servons des approximations de ces propriétés comme suit :

**Contribution 1 : approximation inf**

$$\text{(I1}^\#\text{)}\ \neg \text{OA}(s_u \rightarrow^* g_1) \qquad \text{(I2}^\#\text{)}\ \text{UA}(s_b \rightarrow^* g_1) \qquad \begin{matrix}\text{(I3)}\ s_b \in \text{unf-prefix}(s_0) \\ \text{(I3}^\#\text{)}\ \text{UA}(s_0 \rightarrow^* s_b)\end{matrix}$$

où unf-prefix($s_0$) est l'ensemble de tous les états accessibles depuis $s_0$ représenté comme le préfixe du dépliage du réseau d'automates qui doit être pré-calculé (voir section 5.7.6).

Nous pouvons utiliser soit (I3) ou (I3$^\#$) selon que le dépliage est possible ou non. À partir des propriétés de la sur-approximation OA et de la sous-approximation UA (proposition 5.2) nous obtenons :

$$\text{(I1}^\#\text{)} \Rightarrow \text{(C1)} \qquad \text{(I2}^\#\text{)} \Rightarrow \text{(C2)} \qquad \begin{matrix}\text{(I3)} \Leftrightarrow \text{(C3)} \\ \text{(I3}^\#\text{)} \Rightarrow \text{(C3)}\end{matrix}$$

Notre méthode est sûre en ce sens qu'elle n'admet pas de faux positifs. Mais elle ignore potentiellement quelques bifurcations. L'utilisation de (I3) plutôt que de (I3$^\#$) peut potentiellement réduire le nombre de faux négatifs. Toutefois, ce n'est possible que si le dépliage de l'ensemble d'états accessibles est possible. Dans le cas où ce n'est pas possible, on peut toujours se référer à (I3$^\#$).

**Contribution 2 : approximation sup**

$$\text{(I4}^\#\text{)}\ \neg \text{UA}(s_u \rightarrow^* g_1) \qquad \text{(I5}^\#\text{)}\ \text{OA}(s_b \rightarrow^* g_1) \qquad \text{(I6}^\#\text{)}\ \text{OA}(s_0 \rightarrow^* s_b)$$

Comme pour l'approximation inf, nous nous servons des propriétés de la sur-approximation OA et de la sous-approximation UA pour obtenir les propriétés suivantes :

$$\text{(C1)} \Rightarrow \text{(I4}^\#\text{)} \qquad \text{(C2)} \Rightarrow \text{(I5}^\#\text{)} \qquad \text{(C3)} \Rightarrow \text{(I6}^\#\text{)}$$

Contrairement à celle de l'approximation inf, ces propriétés admettent des faux positifs. Mais l'ensemble des transitions de bifurcation contient bien toutes les bifurcations recherchées.

Il se peut qu'on obtienne une zone où notre approche est non concluante. En effet, la sur-approximation contient l'ensemble des transitions de bifurcation recherché. Mais l'ensemble construit par la sur-approximation peut potentiellement contenir plus de bifurcations et donc des faux positifs. En enlevant toutes les transitions obtenues par la sous-approximation aux transitions obtenues par la sur-approximation, il reste un ensemble de transitions sur lequel il est difficile de se prononcer à moins d'effectuer une vérification exacte pour chacune d'entre elle.

## 5.4 Approximations Inf des états/transitions de bifurcations

Dans cette section, nous définissons deux approximations inférieures des états de bifurcations (définition 5.12, définition 5.13) et nous établissons la preuve qu'elles sont bien des sous-approximations. La grande différence entre les définitions est que la première (définition 5.12) effectue un test exact pour pour vérifier $s_0 \rightarrow^* s_b$, tandis que la seconde (définition 5.13) calcule une sous-approximation pour effectuer la vérification $s_0 \rightarrow^* s_b$.

### 5.4.1 Définition

**Définition 5.12** (Approximation Inf de $B^g_{s_0}$ : $\mathcal{B}^{g\#}_{s_0}$)**.** $\mathcal{B}^{g\#}_{s_0}$ est une sous-approximation de $B^g_{s_0}$ construite en générant l'ensemble des états qui vérifient les propriétés suivantes : $\neg\,\mathrm{OA}(s_u \rightarrow^* g)$, $\mathrm{E}(s_0 \rightarrow^* s_b)$, $\mathrm{UA}(s_b \rightarrow^* g)$.
$\mathcal{B}^{g\#}_{s_0} = \{s_b : \exists s \xrightarrow{t_b} s_u, \neg\,\mathrm{OA}(s_u \rightarrow^* g) \wedge \mathrm{E}(s_0 \rightarrow^* s_b) \wedge \mathrm{UA}(s_b \rightarrow^* g)\}$.

De la définition 5.12, nous établissons le théorème 5.2.

**Théorème 5.2.** $\mathcal{B}^{g\#}_{s_0} \subseteq B^g_{s_0}$.

Le théorème 5.2 stipule que $\mathcal{B}^{g\#}_{s_0}$ est un sous ensemble de l'ensemble de bifurcations que nous cherchons à construire. Nous construisons ici la preuve de ce théorème.

*Démonstration.* Soit $x \in \mathcal{B}^{g\#}_{s_0}$.

Par définition de $\mathcal{B}^{g\#}_{s_0}$ (voir la définition 5.12) $x \in \{s_b : \exists s \xrightarrow{t_b} s_u, \neg\,\mathrm{OA}(s_u \rightarrow^* g) \wedge \mathrm{E}(s_0 \rightarrow^* s_b) \wedge \mathrm{UA}(s_b \rightarrow^* g)\}$.

Nous avons donc $\mathrm{E}(s_0 \rightarrow^* s_b) \Rightarrow \exists s_0 \rightarrow^* s_b \wedge \mathrm{UA}(s_b \rightarrow^* g) \Rightarrow s_b \rightarrow^* g \wedge s_b \notin U_{s_0}$ par conséquent $x \notin U_{s_0}$.

Par conséquent $x \in R^g \cap R_{s_0}$.

Par ailleurs, nous avons aussi $\exists s_b \xrightarrow{t_b} s_u \neg\,\mathrm{OA}(s_u \rightarrow^* g) \Rightarrow \exists s_b \xrightarrow{t_b} s_u$ telle que $\nexists s_u \rightarrow^* g$ aussi $s_u \in U^g$.

Aussi, nous montrons que chaque fois que $x$ vérifie les propriétés de $\mathcal{B}^{g\#}_{s_0}$, $x$ vérifie également les propriétés de $B^g_{s_0}$.

Par conséquent, $x \in \mathcal{B}^{g\#}_{s_0} \Rightarrow x \in B^g_{s_0}$.

Ce qui prouve bien que $\mathcal{B}^{g\#}_{s_0} \subseteq B^g_{s_0}$. □

Cette deuxième définition est utile s'il n'est pas possible d'utiliser la propriété (I3). Ce qui peut arriver quand l'espace d'états est trop grand pour ne pas permettre l'utilisation des approches par dépliage pour une vérification exacte.

**Définition 5.13** (Approximation Inf de $B^g_{s_0}$ : $\mathbb{B}^{g\#}_{s_0}$)**.** $\mathbb{B}^{g\#}_{s_0}$ est une sous-approximation de $B^g_{s_0}$ construite en générant l'ensemble des états qui vérifient les trois propriétés suivantes : $\neg\,\mathrm{OA}(s_u \rightarrow^* g)$, $\mathrm{UA}(s_0 \rightarrow^* s)$, $\mathrm{UA}(s \rightarrow^* g)$.
$\mathbb{B}^{g\#}_{s_0} = \{s_b : \exists s_b \xrightarrow{l} s_u, \neg\,\mathrm{OA}(s_u \rightarrow^* g) \wedge \mathrm{UA}(s_0 \rightarrow^* s_b) \wedge \mathrm{UA}(s_b \rightarrow^* g)\}$.

Nous écrivons aussi le théorème 5.3 qui stipule que le construction faite avec la définition 5.13 est une sous-approximation de $B^g_{s_0}$.

**Théorème 5.3.** $\mathbb{B}^{g\#}_{s_0} \subseteq B^g_{s_0}$.

Nous établissons la preuve de ce théorème comme suit :

*Démonstration.* Soit $x \in \mathbb{B}^{g\#}_{s_0}$.

Par définition de $\mathbb{B}^{g\#}_{s_0}$ (voir la définition 5.13), $x \in \{s_b : \exists s_b \xrightarrow{t_b} s_u, \neg\,\mathrm{OA}(s_u \rightarrow^* g) \wedge \mathrm{UA}(s_0 \rightarrow^* s_b) \wedge \mathrm{UA}(s_b \rightarrow^* g)\}$.

De cette définition, nous avons les propriétés suivantes $\mathrm{UA}(s_0 \rightarrow^* s_b) \Rightarrow \exists s_0 \rightarrow^* s_b \wedge \mathrm{UA}(s_b \rightarrow^* g) \Rightarrow s_b \rightarrow^* g$ et $s_b \notin U^g$ d'où $x \notin U^g$.

Par conséquent : $x \in R^g \cap R_{s_0}$.

Nous avons également $\exists s \xrightarrow{t_b} s_u \neg\,\mathrm{OA}(s_u \rightarrow^* g) \Rightarrow \exists s_b \xrightarrow{t_b} s_u$ telle que $\not\exists s_u \rightarrow^* g$ par conséquent $s_u \in U^g$.

Nous avons montré que chaque fois que $x$ vérifie les propriétés de $\mathcal{B}^{g\#}_{s_0}$, $x$ vérifie également les propriétés de $B^g_{s_0}$.

Par conséquent : $x \in \mathbb{B}^{g\#}_{s_0} \Rightarrow x \in B^g_{s_0}$.

Ce qui permet de conclure que : $\mathbb{B}^{g\#}_{s_0} \subseteq B^g_{s_0}$.

□

Après les deux définitions de la sous-approximation que nous avons données (voir la définition 5.13 et la définition 5.12), il est légitime de se poser la question de savoir quelle relation lie les deux ensembles construits par ces définissions. Aussi, le théorème 5.4 établit la relation qui existe entre les deux ensembles.

**Théorème 5.4.** $\mathbb{B}^{g\#}_{s_0} \subseteq \mathcal{B}^{g\#}_{s_0}$

*Démonstration.* En remplaçant la vérification de la propriété (I3) ($\mathrm{E}(s_0 \rightarrow^* s_b)$) par la propriété ($\mathrm{I3}^{\#}$) ($\mathrm{UA}(s_0 \rightarrow^* s_b)$) on réduit potentiellement le nombre d'états $s_b$ qu'on peut trouver de par la propriété de la sous-approximation (proposition 5.2). □

## 5.5 Approximation Sup des états/transitions de bifurcations

Cette section présente la construction de la sur-approximation des transitions de bifurcation. On y donne une définition de cette sur-approximation et une preuve qu'elle est bien une sur-approximation.

### 5.5.1 Définition

**Définition 5.14** (Approximation Sup de $B^g_{s_0}$ : $\widetilde{\mathcal{B}^g_{s_0}}$)**.** $\widetilde{\mathcal{B}^g_{s_0}}$ est une approximation Sup de $B^g_{s_0}$ construit en générant l'ensemble des états tels que $\neg\,\mathrm{UA}(s_u \rightarrow^* g) \wedge \mathrm{OA}(s_0 \rightarrow^* s) \wedge \mathrm{OA}(s \rightarrow^* g)$.
$\widetilde{\mathcal{B}^g_{s_0}} = \{s_b : \exists s \xrightarrow{t_b} s_u, \neg\,\mathrm{UA}(s_u \rightarrow^* g) \wedge \mathrm{OA}(s_0 \rightarrow^* s_b) \wedge \mathrm{OA}(s_b \rightarrow^* g)\}$.

De cette définition, nous pouvons écrire le théorème 5.5.

**Théorème 5.5.** $B^g_{s_0} \subseteq \widetilde{\mathcal{B}^g_{s_0}}$.

*Démonstration.* Soit $x \in B^g_{s_0}$

Par définition de $B^g_{s_0}$ (voir la définition 5.10 en page 117) $x \in \{s_b : s_0 \rightarrow^* s_b \quad \wedge \quad s_b \notin U^g \wedge \quad \exists t_b \in T, \quad s_b \xrightarrow{t_b} s_u \quad \wedge \quad s_u \in U^g \wedge s_b \rightarrow^* g\}$.

Par conséquent, $s_0 \rightarrow^* s_b \Rightarrow \mathrm{OA}(s_0 \rightarrow^* s_b)$ est vrai et $s_b \rightarrow^* g \Rightarrow \mathrm{OA}(s_b \rightarrow^* g)$ est également vrai.

Par ailleurs, nous avons $(s_b \notin U^g \quad \wedge \quad \exists t_b \in T, \quad s_b \xrightarrow{t} s_u \quad \wedge \quad s_u \in U^g) \Rightarrow \neg\,\mathrm{UA}(s_u \rightarrow^* g)$ est vrai.

Par conséquent pour tout $x \in B^g_{s_0}$, $x$ vérifie les propriétés qui garantissent également les propriétés de $\widetilde{\mathcal{B}^g_{s_0}}$.

Ce qui prouve bien que $B^g_{s_0} \subseteq \widetilde{\mathcal{B}^g_{s_0}}$. □

Pour l'implémentation de ces concepts, nous avons eu recours à la programmation par ensemble de réponse (ASP) que nous introduisons dans la section 5.6.

## 5.6 Présentation de la programmation par ensemble de réponses (ASP)

L'answer set programming (Baral, 2003; Gebser & Schaub, 2013; Gebser, Kaminski, Kaufmann & Schaub, 2012) est une forme de programmation déclarative particulièrement bien adaptée pour les problèmes des recherches difficiles. Nous faisons dans cette section un bref rappel des éléments de syntaxe et de sémantique qui permettent de comprendre la suite des implémentations effectuées dans ce chapitre.

### 5.6.1 Le paradigme

La programmation déclarative est un paradigme de programmation qui vise à appréhender le problème comme un programme. Elle se distingue des autres paradigmes comme la programmation impérative, la programmation objet ou la programmation fonctionnelle. Bien que les paradigmes impératifs, objets et fonctionnelles soient très différents d'un point de vue de la conception des programmes, ils ont en commun la démarche pour trouver la solution à un problème. Cette démarche consiste à partir du problème, puis à proposer une spécification algorithmique de la solution du problème à résoudre. Après la phase de spécification algorithmique, un programme informatique est implémenté selon le paradigme (impératif, objet ou fonctionnelle) pour la résolution du problème.

La démarche déclarative est complètement différente de celle que nous venons de brièvement présenter. En programmation déclarative, le problème est un programme. L'idée est donc de proposer une modélisation du problème à résoudre sous forme d'axiomes et de contraintes exprimées dans un langage logique. Les résultats du programme sont des modèles logiques qui représentent les solutions de l'ensemble des formules. Ainsi un modèle est donc un *ensemble réponse*. Aussi, le résultat à un problème est donc un ensemble de modèles. Ce résultat est obtenu via des solveurs qui effectuent la recherche d'une, de plusieurs, ou enfin de l'ensemble des solutions.

ASP combine donc à la fois un riche mais simple langage de modélisation avec des capacités de résolution très performantes. Il tire ses racines des bases de données déductives, de la programmation logique, de la représentation des connaissances par une approche logique et des solveurs de contraintes.

L'ASP est particulièrement bien adapté pour la résolution des problèmes de recherche combinatoire des classes de complexité $P$, $NP$, $NP^{NP}$. Nous pouvons citer par exemple, les problèmes de planification des robots, d'optimisation de code, d'intégration des bases de données, le model-checking, la composition musicale, les systèmes biologiques et bien d'autres encore.

### 5.6.2 Éléments de syntaxe et de sémantique

L'ASP est très proche des autres langages de programmation du point de vue de la syntaxe. Il admet donc des constantes, des variables, des prédicats. Il comporte aussi des faits, des règles et des contraintes. Notons déjà que les commentaires en ASP se mettent avec le symbole % ou $\%*\ldots*\%$.

- **Les atomes ou constantes** sont soit des entiers, soit des mots constitués des lettres de a-z, A-Z, 0-9, _ * commençant par une minuscule.

- **Les variables** sont formées comme les constantes à la seule différence qu'elles commencent par une majuscule contrairement aux termes ou aux prédicats. En programmation logique, une variable n'est pas un contenant auquel on affecte une valeur, mais représente l'ensemble des valeurs admissibles pour la variable dans le cadre des contraintes.
- **Les prédicats** sont de la forme `constante(`$terme_1$`, ..., `$terme_n$`)` et sont définis comme vrais. Un prédicat consiste en une tête et un nombre fini d'arguments.
- **Les règles** sont de la forme `x←y`. Le symbole ← signifie « si » ; par exemple on pourrait avoir la règle suivante :

```
1 lumiere(on) ← interrupteur(on).
```

  qui indique que `lumiere(on)` est vraie si `interrupteur(on)` est vrai. Il est également possible d'utiliser les variables dans les règles. Par exemple, on a la règle :

```
2 pere(X,Y) ← parent(X,Y), male(X).
```

  pour signifier qu'un $X$ est le père d'un $Y$ si $X$ est parent de $Y$ et $X$ est mâle. Dans cette règle « , » indique une conjonction. De même on peut avoir la règle

```
3 parent(X, Y) ← pere(X, Y) ; mere(X, Y).
```

  pour signifier qu'un $X$ est parent d'un $Y$ si $X$ est père de $Y$ ou mère de $Y$ ; ici « ; » indique l'alternative.
- **L'extension conditionnelle** `r(X):q(X)`.
- **Littéraux de choix ensemblistes** `min { ... } max`. où *min* et *max* représentent respectivement les cardinalités minimales et maximales pour le choix d'un sous-ensemble des atomes contenant de *min* à *max* atomes.

Un programme ASP est donc un ensemble de règles de la forme :

```
4 a0 ← a1, ..., an, not an+1, ..., not an+k.
```

où les $a_i$ sont des atomes, des termes ou des prédicats de la logique du premier ordre.

Brièvement, la sémantique de ces règles est la suivante : quand tous les $a_1, \ldots, a_n$ sont vrais, et tous les $a_{n+1}, \ldots, a_{n+k}$ sont faux, alors $a_0$ est considéré comme vrai. $a_0$ peut être $\bot$ ou simplement être absent et dans ce cas, la règle est satisfaite seulement si le corps de la règle (côté droit) est faux (c'est-à-dire au moins un des $a_1, \ldots, a_n$ est faux ou alors au moins $a_{n+1}, \ldots, a_{n+k}$ est vrai). En ASP, l'ordre des clauses n'a aucune importance.

Une solution consiste en des ensembles vrais d'atomes/termes/prédicats avec lesquels toutes les règles du programme sont satisfaites.

### 5.6.3 Exemple basique d'utilisation de l'ASP

Par exemple, le programme ASP suivant a une unique solution minimale `b(1) b(2) c(1) c(2)`.

```
5 c(X) ← b(X).
6 b(1).
7 b(2).
```

Dans la suite nous utiliserons la notation $n$ `{ a(X) : b(X) }` $m$ qui est vrai quand au moins $n$ et au plus $m$ `a(X)` sont vrais où `X` est compris entre les valeurs vraies du prédicat `b(X)`.

## 5.7 Implémentation en ASP de notre approche pour l'identification des bifurcations

Nous présentons ici les principales règles nécessaires pour l'implémentation de l'identification des transitions de bifurcation avec l'ASP.

Une grande partie des prédicats utilisés par (I1$^{\#}$), (I2$^{\#}$), (I3), et (I3$^{\#}$) a été générée par un calcul préalable de local-paths et, dans le cas de (I3), par le calcul du préfixe du dépliage qui est l'ensemble des états accessibles depuis un état initial donné.

### 5.7.1 Déclaration des états locaux, des transitions et des états

Chaque état local $a_i \in S(a)$ de chaque automate $a \in \Sigma$ est déclaré avec le prédicat `ls(a,i)`. Nous déclarons les transitions locales d'un réseau d'automates et les conditions associées à ces transitions avec les prédicats `tr(id,a,i,j)` et `trcond(id,b,k)`, qui correspondent à la transition locale $a_i \xrightarrow{\{b_k\}\cup\ell} a_j \in T$. Les états sont déclarés avec le prédicat `s(ID,A,I)` où `ID` est l'identificateur de l'état, et `A` est l'automate et `I` l'état local présent dans l'état. Enfin, le but $g_1$ est déclaré avec `goal(g,1)`.

Par exemple, les instructions suivantes permettent de déclarer l'automate $a$ de la figure 5.1 avec ses transitions locales, l'état $s_0 = \langle a_0, b_0, c_0\rangle$, et le but étant $a_2$ :

```
ls(a,0). ls(a,1). ls(a,2).
tr(1,a,1,0).
tr(2,a,0,1). trcond(2,b,0).
tr(3,a,0,2). trcond(3,b,0). trcond(3,c,0).
s(0,a,0). s(0,b,0). s(0,c,0).        goal(a,2).
```

### 5.7.2 Implémentation des approximations Sup et Inf de l'accessibilité en ASP

#### 5.7.2.1 OA($s \to^* s'$) : implémentation de la sur-approximation en ASP

Nous présentons un possible encodage de la sur-approximation de l'accessibilité dans les réseaux d'automates telle que présentée et introduit par (Paulevé et al., 2012).

Partant de $s_u(g) = g_0$, l'analyse commence avec un *chemin local* de l'*objectif* $g_0 \rightsquigarrow g_1$ : $g_1$ est atteignable si et seulement si toutes les conditions des transitions d'au moins un chemin local $\eta \in$ local-paths$(g_0 \rightsquigarrow g_1)$ est atteignable.

Ce raisonnement récursif peut être modélisé avec un graphe établissant les dépendances entre les objectifs, les chemins locaux, et les états locaux.

Les chemins locaux générés *a priori* sont utilisés pour générer un modèle de déclaration des arcs d'un GCL

`oa_lcg(G,Parent,Child)` partant de chaque objectif possible $a_i \rightsquigarrow a_j$. Si local-paths$(a_i \rightsquigarrow a_j) = \emptyset$, l'objectif $a_i \rightsquigarrow a_j$ est lié au nœud `bottom` :

```
oa_lcg(G,obj(a,i,j),bottom) ← oa_lcg(G,_,obj(a,i,j)).
```

sinon, pour local-paths$(a_i \rightsquigarrow a_j) = \{\eta^1, \ldots, \eta^n\}$, nous déclarons un nœud `lpath` pour chaque chemin local différent $m \in \{1, \ldots, n\}$ comme un fils de $a_i \rightsquigarrow a_j$ :

```
oa_lcg(G,obj(a,i,j),lpath(obj(a,i,j),m)) ← oa_lcg(G,_,obj(a,i,j)).
```

puis, pour chaque état local différent $b_k \in \widetilde{\eta^m}$ dans les conditions d'une transition locale de $\eta^m$, nous ajoutons un arc depuis le nœud `lpath` à `ls(`$b$`,`$k$`)` :

```
oa_lcg(G,lpath(obj(a,i,j),m),ls(b,k)) ← oa_lcg(G,_,obj(a,i,j)).
```

Dans le cas où le chemin local ne nécessite aucune condition ($\widetilde{\eta^m} = \emptyset$, ce qui arrive quand l'objectif est trivial, i.e., $a_i \rightsquigarrow a_i$, ou quand la transition locale ne dépend pas d'un autre automate), nous connectons `lpath` à un nœud `top` :

```
oa_lcg(G,lpath(obj(a,i,j),m),top) ← oa_lcg(G,_,obj(a,i,j)).
```

Un GLC $G$ pour la sur-approximation est paramétré avec un état $s_G$ : si un chemin local a un état local $a_j$ dans ses conditions de transitions, le nœud `ls(`$a$`,`$j$`)` est lié, dans $G$, avec le nœud pour l'objectif $a_i \rightsquigarrow a_j$ (5), avec $a_i = s_G(a)$. Il est donc requis que l'état $s_G$ définit un (unique) état local pour chaque automate référencé dans $G$ (6).

```
oa_lcg(G,ls(A,I),obj(A,J,I)) ← oa_lcg(G,_,ls(A,I)), s(G,A,J).
1 { s(G,A,J) : ls(A,J) } 1 ← oa_lcg(G, _, ls(A, _)).
```

Les conditions nécessaires pour l'accessibilité sont donc déclarées à travers le prédicat `oa_valid(G,N)` qui est vrai si le nœud `N` satisfait les conditions suivantes : il n'est pas `bottom` (7) ; et dans le cas d'un état local ou d'un nœud objectif, un de ses fils est `oa_valid` (8,9) ; ou dans le cas d'un chemin local, soit `top` est un de ses fils, ou tous ses fils (états locaux) sont `oa_valid` (10,11).

```
← oa_valid(G,bottom).
oa_valid(G,ls(A,I)) ← oa_lcg(G,ls(A,I),X),oa_valid(G,X).
oa_valid(G,obj(A,I,J)) ← oa_lcg(G,obj(A,I,J),X),oa_valid(G,X).
oa_valid(G,N) ← oa_lcg(G,N,top).
oa_valid(G,lpath(obj(a,i,j),m)) ← ⋀_{b_k∈η̃^m} oa_valid(G,ls(b,k)).
```

#### 5.7.2.2 UA($s \rightarrow^* s'$) : implémentation de la sous-approximation de l'accessibilité en ASP

Nous présentons dans cette section une implémentation de la sous-approximation de l'accessibilité dans les réseaux d'automates tels que décrits dans la proposition 5.2 et introduits dans (Folschette et al., 2015).

La sous-approximation consiste en la construction d'un graphe reliant les objectifs, les chemins locaux, et les états locaux qui satisfont un certain nombre de contraintes. Si un tel graphe existe, alors la propriété d'accessibilité associée est vraie.

De façon analogue à (I1#), nous donnons un modèle de déclaration pour les arcs avec le prédicat `ua_lcg(G,Parent,Child)`.

Nous supposons que la propriété d'accessibilité est spécifiée en ajoutant un arc de `root` à `ls(`$a$`,`$i$`)` pour chaque état local à atteindre.

Le graphe `ua_lcg` est paramétré avec un *contexte* qui est un ensemble d'états locaux déclarés avec le prédicat `ctx(G,A,J)`. Chaque état local $a_i$ du graphe qui ne fait pas partie des spécifications d'accessibilité associées au contexte (12) ; et aux objectifs $a_j \rightsquigarrow a_i$ pour chaque $a_j$ dans le contexte (13).

```
ctx(G,A,I) ← ua_lcg(G,N,ls(A,I)), N != root.
ua_lcg(G,ls(A,I),obj(A,J,I)) ← ua_lcg(G,_,ls(A,I)), ctx(G,A,J).
```

La première contrainte est que chaque objectif dans le graphe est à un et un seul de ses chemins locaux. Une conséquence immédiate est qu'un objectif sans chemins locaux

(local-paths($a_i \rightsquigarrow a_j$) = $\emptyset$) ne peut pas être inclu (14), pour les autres, un choix doit être fait parmi local-paths($a_i \rightsquigarrow a_j$) = $\{\eta^1, \ldots, \eta^n\}$ (15).

```
14 ← ua_lcg(G,_,obj(a,i,j)).
15 1 { ua_lcg(G,obj(a,i,j),lpath(obj(a,i,j),1..n)) } 1 ← ua_lcg(G,_,obj(a,i,j)).
```

Comme pour la sur-approximation `oa_lcg`, chaque état local est associé à tous les états locaux faisant partie de ses conditions de transitions : pour chaque $m \in \{1, \ldots, n\}$, pour chaque $b_k \in \widetilde{\eta^m}$,

```
16 ua_lcg(G,lpath(obj(a,i,j),m),ls(b,k)) ← ua_lcg(G,_,obj(a,i,j)).
```

Le graphe doit être sans cycle. Ce qui se code en utilisant le prédicat `conn(G,X,Y)` qui est vrai si le nœud `X` est connecté (s'il existe un chemin direct) à `Y` (17).

Un graphe est acyclique quand il n'est pas possible de vérifier que partant d'un nœud du graphe on repasse toujours par ce nœud `conn(G,X,X)` (18).

```
17 conn(G,X,Y) ← ua_lcg(G,X,Y).  conn(G,X,Y) ← ua_lcg(G,X,Z), conn(G,Z,Y).
18 ← conn(G,X,X).
```

Par la suite, si un nœud pour un objectif $a_i \rightsquigarrow a_j$ est connecté à un état local $a_k$, la sous-approximation requiert que $a_i \rightsquigarrow a_j$ soit connecté avec $a_k \rightsquigarrow a_j$ (avec l'hypothèse que $a$ possède au moins trois états locaux, la définition n'est pas montrée ici) :

```
19 ua_lcg(G,obj(A,I,J),obj(A,K,J)) ← not boolean(A), conn(G,obj(A,I,J),ls(A,K)).
```

Quand une transition locale est conditionnée par au moins deux autres automates (par exemple $c_\circ \xrightarrow{a_i,b_j} c_\bullet$), la sous-approximation requiert que l'accessibilité de $b_j$ ne doit pas être possible par un état local de $a$ autre que $a_i$.

Ce fait est garanti par la ligne `indep(G,Y,a,i,ls(b,j))` qui ne peut pas être vraie si $b_j$ est connecté à un état local $a_k$ avec $k \neq i$ (20).

Par la suite, la sous-approximation requière qu'au plus un prédicat `indep` soit faux, pour un GCL $G$ et un chemin local donné $Y$ (21). Une telle indépendance devrait aussi être prise en compte entre les états locaux des spécifications d'accessibilité (22).

```
20 indepfailure(Y,ls(A,I)) ← indep(G,Y,A,I,N), conn(G,N,ls(A,K)), K!=I.
21 ← indepfailure(Y,N),indepfailure(Y,M),M!=N.
22 indep(G,root,A,I,ls(B,J)) ← ua_lcg(G,root,ls(A,I)),ua_lcg(G,root,ls(B,J)),B != A.
```

Pour $\eta^m \in$ local-paths($a_i \rightsquigarrow a_j$), pour chaque transition locale $a_\circ \xrightarrow{\ell} a_\bullet \in \eta^m$, pour chaque couple d'état locaux différent dans les conditions $b_k, c_l \in \ell$, $b_k \neq c_l$ :

```
23 indep(G,lpath(obj(a,i,j),m),b,k,ls(c,l)) ← ua_lcg(G,_,lpath(obj(a,i,j),m)).
```

### 5.7.3 Déclaration de $s_b$, $t_b$, et $s_u$

La bifurcation de transition $t_b$, déclarée comme `btr(b)`, est sélectionnée parmi les identificateurs de transitions (24). Si $a_i \xrightarrow{\ell} a_j$ est la transition sélectionnée, l'état global $s_u$ (pour rappel $s_u = s_b \cdot t_b$) devrait satisfaire $s_u(a) = a_j$ (25) et, $\forall b_k \in \ell$, $s_u(b) = b_k$ (26). L'état $s_b$ devrait donc correspondre à l'état $s_u$, excepté pour l'automate $a$, comme $s_b(a) = a_i$ (27,28).

```
24 1 { btr(ID) : tr(ID,_,_,_) } 1.
25 s(u,A,J) ← btr(ID),tr(ID,A,_,J).
26 s(u,B,K) ← btr(ID),trcond(ID,B,K).
```

```
27 s(b,A,I) ← btr(ID),tr(ID,A,I,_).
28 s(b,B,K) ← s(u,B,K),btr(ID),tr(ID,A,_,_),B!=A.
```

### 5.7.4 (I1#) déclaration de $\neg \mathrm{OA}(s_u \rightarrow^* g_1)$

Dans cette section, nous cherchons l'état $s_u$ à partir duquel $g_1$ n'est pas accessible. Pour ce faire, nous concevons une implémentation en ASP de sur-approximation de l'accessibilité (voir proposition 5.2).Elle consiste à construire un graphe de causalité locale à partir des chemins locaux préalablement calculés local-paths. Le prédicat `oa_valid(G,ls(A,I))` est par la suite défini sur $G$ comme vrai si l'état local $a_i$ est accessible depuis l'état initial $s_G$.

La mise en œuvre complète de la sur-approximation est donnée à la section 5.7.2.1 en page 125.

Nous créons une instance du LCG $u$ avec l'état initial $s_u$ en spécifiant que le but ne soit pas accessible (`oa_valid`) (29).

```
29 ← oa_valid(u,ls(G,I)),goal(G,I).
```

### 5.7.5 (I2#) déclaration de $\mathrm{UA}(s_b \rightarrow^* g_1)$

Cette section a pour but de trouver l'état $s_b$ depuis lequel $g_1$ est accessible. Notre conception de la sous-approximation de l'accessibilité en ASP (proposition 5.2) consiste à chercher un sous graphe de causalité locale $G$ qui satisfait les propriétés prouvant les conditions suffisantes de l'accessibilité.

Les arcs de ce sous-graphe sont déclarés avec le prédicat `ua_lcg(G,Parent,Child)`. Le graphe est paramétré avec (1) le *contexte* qui spécifie un ensemble d'états initiaux possibles et par (2) un arc depuis un nœud racine `root` jusqu'à l'état local ou aux états locaux pour lesquels l'accessibilité simultanée doit être décidée depuis le contexte fourni. L'implémentation complète est fourni à la section 5.7.2.2 en page 126.

Nous créons une instance de la sous-approximation pour construire un état $s_b$ à partir duquel si on applique une transition de bifurcation $t_b$ le but $g_1$ n'est pas accessible : $g_1$ est un fils du nœud `root` du graphe $b$ (30). Le contexte reste assujetti aux même contraintes comme $s_b$ depuis $s_u$ (31,32 reflète 27,28). Ensuite, $s_b$ défini un état local par automate à travers le contexte à partir duquel l'accessibilité de $g_1$ est garantie (33), et selon 27,28.

```
30 ua_lcg(b,root,ls(G,I)) ← goal(G,I).
31 ctx(b,A,I) ← btr(ID),tr(ID,A,I,_).
32 ctx(b,B,K) ← s(u,B,K),btr(ID),tr(ID,A,_,_),B!=A.
33 1 { s(b,A,I) : ctx(b,A,I) } 1 ← ctx(b,A,_).
```

### 5.7.6 Implémentation en ASP de l'accessibilité avec le dépliage

Un préfixe d'un réseau d'automates ou dépliage est un digraphe biparti où les nœuds sont soit des *évènements* (application des transitions) ou des *conditions* (changements d'états locaux) (Esparza & Heljanko, 2008). Nous utilisons le prédicat `post(X,Y)` pour représenter un arc de $X$ à $Y$ ; et `h(C,ls(A,I))` pour représenter le fait que la condition $C$ correspond à l'état local $a_i$. La figure 5.4 en page 131 montre un exemple de dépliage.

Un état $s$ appartient au préfixe s'il est possible de construire une configuration telle que tous les états locaux dans $s$ ont une unique condition correspondante sur la coupure de la configuration *cut* (34) dont nous donnons la définition ci-dessous.

Une configuration est un ensemble d'évènements et nous utilisons `e(E)` pour représenter le fait que l'évènement $E$ appartient à la configuration. Par définition, si $E$ est dans une configuration, tous ses évènements parents sont dans la configuration (35). Il ne devrait pas y avoir de *conflit* entre deux évènements de la configuration : deux évènements sont en conflit s'ils partagent une condition parente commune (36).

Une condition est dans la coupure si ses évènements parents sont dans la configuration (37), et aucun de ses évènements enfants n'est dans la configuration (38).

```
1 { cut(C)  : h(C,ls(A,I)) } 1 ← reach(A,I).
e(F) ← post(F,C),post(C,E),e(E).
← post(C,E),post(C,F),e(E),e(F),E != F.
e(E) ← cut(C),post(E,C).
← cut(C),post(C,E),e(E).
```

### 5.7.7 (I3) déclaration de $s_b \in$ unf-prefix($s_0$)

Étant donné un préfixe d'un dépliage depuis $s_0$ (section 5.7.6), vérifier si $s_b$ est accessible depuis $s_0$ est un problème NP-complet (Esparza & Schröter, 2001) qui peut être programmé de façon efficace en SAT (Heljanko, 1999) (et de fait en ASP). Une description synthétique de l'implémentation de l'accessibilité dans le dépliage est donnée par la section 5.7.6 à la page précédente. Pour plus de détails se référer (Esparza & Heljanko, 2008).

Notre encodage fourni le prédicat `reach(a,i)` qui est vrai si un état accessible contient $a_i$. Déclarer que $s_b$ est accessible depuis $s_0$ se fait de la façon suivante :

```
reach(A,I) ← s(b,A,I).
```

### 5.7.8 (I3#) déclaration de UA($s_0 \rightarrow^* s_b$)

Une alternative à (I3) qui ne nécessite pas de calculer le préfixe complet du dépliage est de se servir de la sous-approximation (I2#). Elle est instanciée pour l'accessibilité de $s_b$ depuis $s_0$ avec les prédicats suivants :

```
ua_lcg(0,root,ls(A,I)) ← s(b,A,I).
ctx(0,A,I) ← s(0,A,I).
```

En appliquant notre implémentation sur l'exemple de la figure 5.1 en page 112, nous trouvons bien $t_8$ comme bifurcation pour $a_2$ depuis $s_0 = \langle a_0, b_0, c_0 \rangle$. Nous présentons d'autres applications sur des exemples biologiques réels à la section 6.3 en page 141 du chapitre 6.

## 5.8 Discussion

Dans ce chapitre, nous avons proposé une méthode originale d'identification des transitions de *bifurcation*. Cette approche combine les méthodes par analyse statique et l'efficacité de la programmation par contrainte pour proposer des résultats prometteurs.

Ainsi, nous avons proposé deux approximations (une inférieure et une supérieure) des transitions de bifurcation. L'approximation inférieure est sûre et n'autorise pas de faux positifs. Cependant il y a le risque de ne pas identifier toutes les bifurcations. Nous avons vu que quand il est possible de remplacer une sous-approximation de l'accessibilité par une

méthode exacte, notre approche peut potentiellement augmenter le nombre de bonnes solutions. Nous avons effectué ce constat au chapitre 6 à la section 6.3.

La sur-approximation, par contre, autorise des faux positifs. Ce qui implique que l'ensemble des transitions de bifurcations peut contenir des transitions qui ne sont pas des transitions de bifurcation. Ce résultat fait apparaître l'existence d'une zone où notre méthode est non concluante.

Nous avons appliqué dans la section 6.3 en page 141 cette méthode à des systèmes biologiques bien connus et présentant des propriétés adéquates pour l'identification des transitions de bifurcation, à savoir que le nombre d'attracteurs est supérieur ou égal à deux. Ces modèles peuvent contenir des centaines de composants (c'est notamment le cas de th_différentiation avec 101 composants). Notre approche donne des résultats en quelques dizaines de secondes ($\sim 30s$) pour le modèle avec 101 composants sur un ordinateur de bureau classique.

Une perspective immédiate et intéressante de ce travail est de pouvoir évaluer étant donné les transitions de bifurcation, la probabilité de ne pas atteindre un objectif partant d'un état initial donné. De fait, si $g$ est le but à atteindre et $\Pr(g)$ est la probabilité de l'atteindre, à partir des transitions de bifurcation, on pourrait proposer une estimation de $1 - \Pr(g)$.

D'autres travaux futurs pourraient s'intéresser à des optimisations à apporter dans la vérification des chemins locaux afin de savoir s'ils garantissent une certaine propriété. Nous pensons notamment à évaluer l'impact de la structure du réseau pour proposer les règles plus optimales pour la génération des ensembles de réponses.

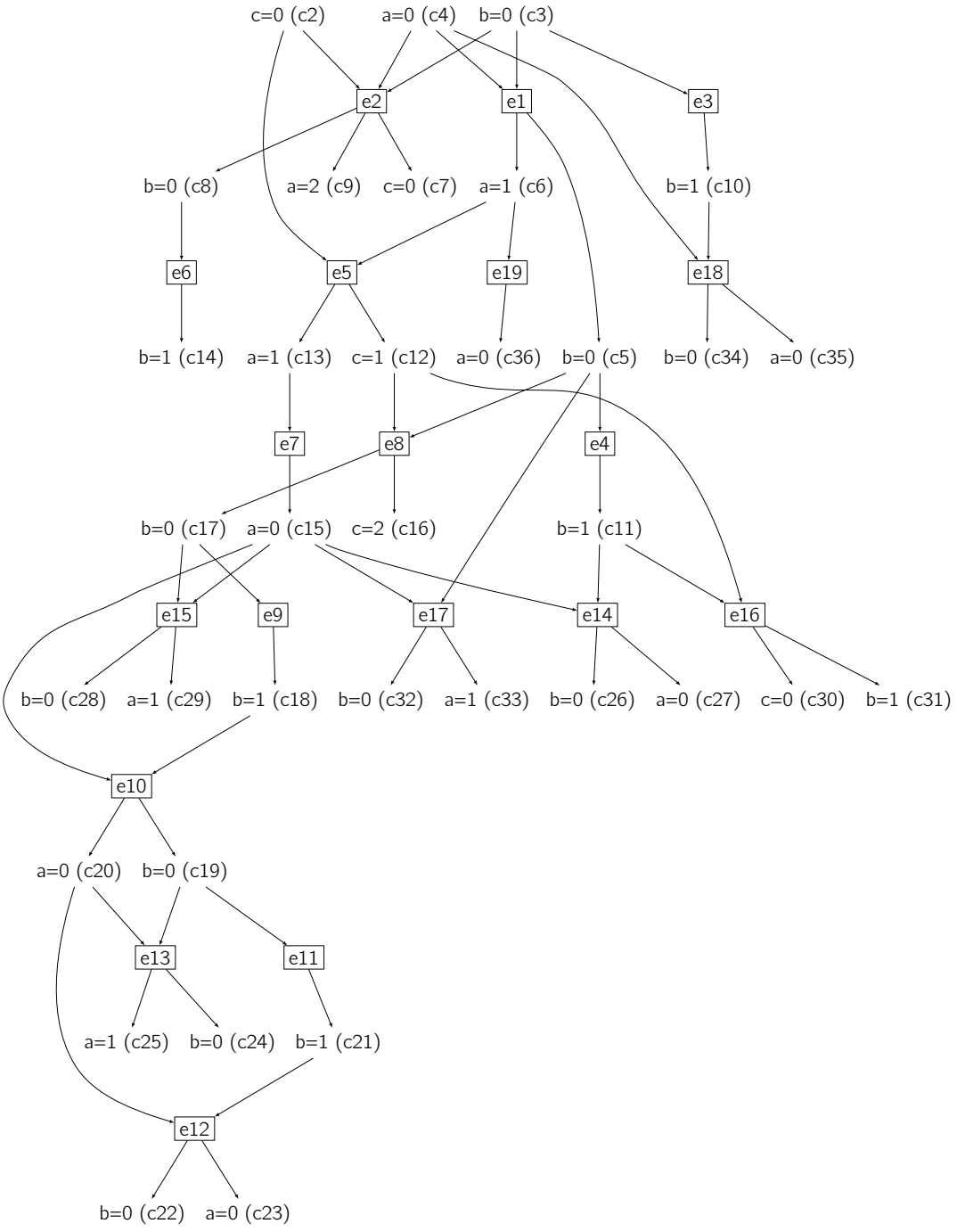



Figure 5.4 : Dépliage du réseau d'automates de la figure 5.1. Les évènements sont les nœuds carrés, les conditions n'ont pas de bornes et indique à la fois les états locaux des automates et les identificateurs de condition.

# Chapitre 6

# Applications sur des exemples biologiques

Ce chapitre présente l'application des méthodes développées pendant cette thèse sur des systèmes biologiques réels. Nous présentons dans un premier temps la mise en application de la modélisation et l'intégration des données de séries temporelles au processus de différenciation de la peau chez les humains. Le modèle utilisé pour la mise en œuvre de notre méthode est un modèle qui comporte près de 300 composants et 400 interactions. Puis, dans un second temps, nous appliquons l'analyse des bifurcations dans les réseaux d'automates asynchrones à trois systèmes biologiques qui sont : le modèle du *phage lambda*, le modèle croisé du facteur de nécrose tumorale alpha et du facteur de croissance épidermique ($EGF/TNF\_\alpha$) et le modèle de différenciation des lymphocytes T (T helper différenciation). Ce dernier modèle comporte plus de 100 composants et plus de 200 interactions, ce qui montre le passage à l'échelle de notre méthode. Puis nous montrons l'impact d'un point de vue quantitatif des transitions de bifurcation sur le modèle du *phage lambda* qui est un peu plus petit (4 composants) par une analyse des probabilités avec le model checker PRISM.

## 6.1 Préliminaires

Ce chapitre est consacré à la mise en œuvre des méthodes développées pendant cette thèse. Cette mise en œuvre est faite par l'application des méthodes à des systèmes biologiques connus et déjà utilisés dans la littérature.

Dans la section 6.2, nous présentons la mise en œuvre de la modélisation des grands RRB par les Frappes de Processus, en intégrant les données quantitatives (temporelles et stochastiques) estimées à partir des données de séries temporelles. Nous illustrons les bénéfices apportés par l'introduction à la fois du temps et de l'aléatoire dans les modèles au

chapitre 3 afin de modéliser finement et quantitativement les comportements des systèmes biologiques. Le système biologique modélisé est celui de la différenciation des cellules de la peau. La modélisation faite vise à montrer l'applicabilité de la méthode décrite dans le chapitre 3 sur les grands systèmes biologiques. En effet, le modèle de la différenciation cellulaire que nous utilisons comporte près de 300 composants et près de 400 interactions. Il est donc être question de :

- Donner des précisions sur le passage du réseau de type RSTC (multi Layer Receptor Transcription Celle State) qui modélise la différenciation cellulaire vers les Frappes de Processus. Dans cette formalisation, plusieurs choix de modélisations seront effectués et seront discutés dans la section 6.2.2. Nous aborderons notamment la discussion sur le choix des niveaux discrets pour les différents types de composants, le choix sur les formes de coopérations que nous avons choisies pour les composants agissant en coopération et plus généralement la logique des interactions entre les composants.
- Détailler l'estimation que nous avons faite des paramètres temporels et stochastiques à partir des données de séries temporelles. Ces détails seront donnés à la section 6.2.3 où nous présenterons également les principales hypothèses de simulations additionnelles que nous avons effectuées.
- Présenter les résultats de l'analyse statistique des traces des composants obtenues grâces aux simulations stochastiques.

Nous discuterons des résultats mis en évidence par cette modélisation en s'appuyant sur les résultats connus dans la littérature sur les modèles de différenciation des cellules de la peau.

Dans le cadre de la vérification des propriétés dynamiques, nous montrerons l'application des bifurcations à la section 6.3 sur trois systèmes biologiques de différentes tailles avec un système contenant notamment une centaine de composants. Les trois modèles représentant ces systèmes sont le modèle du *phage lambda*, le modèle croisé du facteur de nécrose tumorale alpha et du facteur de croissance épidermique ($EGF/TNF\_\alpha$) et le modèle de la différenciation des lymphocytes T chez les humains ($Th\_différentiation$). Ces trois systèmes seront présentés à la section 6.3.1 en page 142. Dans ce chapitre des applications, nous présenterons :

- La mise en œuvre des bifurcations sur les modèles présentés à la section 6.3.1 en page 142. Nous discuterons des bifurcations obtenues sur les différents modèles.
- L'impact d'un point de vue quantitatif des transitions de bifurcations. Il s'agira de montrer que plus la vitesse des transitions de bifurcation par rapport à un objectif augmente, plus la probabilité d'atteindre cet objectif diminue. Cette illustration s'effectuera sur le modèle du *phage lambda* en utilisant le model checker probabiliste PRISM (Kwiatkowska, Norman & Parker, 2011) qui sera présenté à la section 6.3.4.1.

## 6.2 Applications de l'intégration des données : simulations et analyses

### 6.2.1 La différenciation cellulaire : cas des cellules de la peau

*La différenciation cellulaire* est un concept de la biologie du développement décrivant le processus par lequel les cellules se spécialisent en un « type » cellulaire. La morphologie

d'une cellule peut changer radicalement durant la différenciation, mais le matériel génétique reste le même, à quelques exceptions près.

Une cellule capable de se différencier en plusieurs types de cellules est appelée *pluripotente*. Ces cellules sont appelées cellules souches chez les animaux et cellules méristématiques chez les plantes supérieures. Une cellule capable de se différencier en tous les types cellulaires d'un organisme est dite *totipotente*. Chez les mammifères, seuls le zygote et les jeunes cellules embryonnaires sont totipotentes, tandis que chez les plantes, beaucoup de cellules différenciées peuvent devenir totipotentes.

Chez la plupart des organismes multicellulaires, toutes les cellules ne sont pas identiques. Cependant, tous les différents types cellulaires sont dérivés d'une seule cellule-œuf fécondée et ce, grâce à la différenciation. Les exemples de types cellulaires différents qui peuvent être observés sont les myocytes (cellules musculaires), les cellules hépatiques (du foie), les neurones (cellules du système nerveux) ou encore les keratynocites (cellules de la peau).

Pendant la différenciation, certains gènes sont exprimés alors que d'autres sont réprimés. Le processus de la différenciation est intrinsèquement régulé grâce notamment au matériel épigénétique des cellules et notamment des facteurs de transcription spécifiques à un lignage cellulaire donné qui vont engager une cellule encore naïve dans une voie de différenciation. Ainsi la cellule différenciée va-t-elle exprimer une partie spécifique de son génome et développer des structures précises et acquérir certaines fonctions.

La différenciation peut entraîner des changements dans nombre d'aspects de la physiologie de la cellule : sa taille, sa forme, sa polarité, son activité métabolique, sa sensibilité à certains signaux et son expression des gènes peuvent toutes être modifiées durant la différenciation. En cytopathologie, le niveau de différenciation cellulaire est utilisé comme mesure de la progression d'un cancer.

La différenciation peut s'observer à deux stades différents, au cours du développement ou au cours de la vie.

Au cours de développement elle se caractérise par le fait que la cellule, formée de la fécondation d'un ovule par un spermatozoïde, peut évoluer et potentiellement former un organisme entier. En effet, cette cellule-œuf se divise en plusieurs cellules identiques dans les premiers jours tout juste après la fécondation. Dans le cas particulier de l'humain, environ quatre jours après la fécondation et après plusieurs cycles cellulaires, ces cellules commencent à se spécialiser et forment une sphère creuse appelée blastocyste. Celui-ci possède une couche de cellules externes (les cellules périphériques ou trophectoderme) et un groupe de cellules internes, appelées cellules de la masse interne. Ce sont ces cellules qui formeront tous les tissus du corps humain. Malgré cela, elles ne peuvent plus individuellement former un organisme entier : elles sont qualifiées de pluripotentes. Ces cellules continuent ensuite à être progressivement déterminées jusqu'à donner des cellules souches qui donneront des cellules de types bien définis. Par exemple, les cellules souches du sang situées dans la moelle osseuse produisent des hématies, des leucocytes et des plaquettes.

Au cours de la vie, la différenciation des cellules souches est un mécanisme qui permet à l'être humain de renouveler ses cellules.

La partie basale de la peau est constituée de cellules souches, qui se différencient de façon asymétrique : une cellule souche donne une cellule de la peau (kératinocyte) et une cellule souche. La cellule de la peau formée migre progressivement jusqu'à la surface de la peau. Ainsi, notre épiderme se renouvelle en permanence.

Dans cette section nous nous intéressons à la modélisation, la simulation et l'analyse du processus de différenciation des cellules de la peau. Le graphe des interactions modélisant

la différenciation cellulaire est représentée par un réseaux du type RSTC (section 3.2.1 en page 53). La construction de ce modèle se fait par sélection des nœuds « seed »associés à la différenciation des cellules de la peau. Le premier d'entre eux est le nœud d'entrée du réseau $E - cadherin$ qui possède un site de fixation du calcium. Puis, il y a 12 gènes (que nous présenterons plus tard) différentiellement exprimées au cours du processus de différenciation cellulaire, les nœuds représentant les états cellulaires de la différenciation et les nœuds représentant l'arrêt du cycle cellulaire. Partant de ces nœuds « seed », le réseau est automatiquement construit à partir de la base de données PID (Schaefer, Anthony, Krupa, Buchoff, Day, Hannay & Buetow, 2009*b*) en utilisant la démarche décrite par (Guziolowski et al., 2012). La figure 6.1 présente un extrait du modèle obtenu pour la différenciation cellulaire des cellules de la peau. Le modèle complet comporte près de 300 composants et 400 interactions.

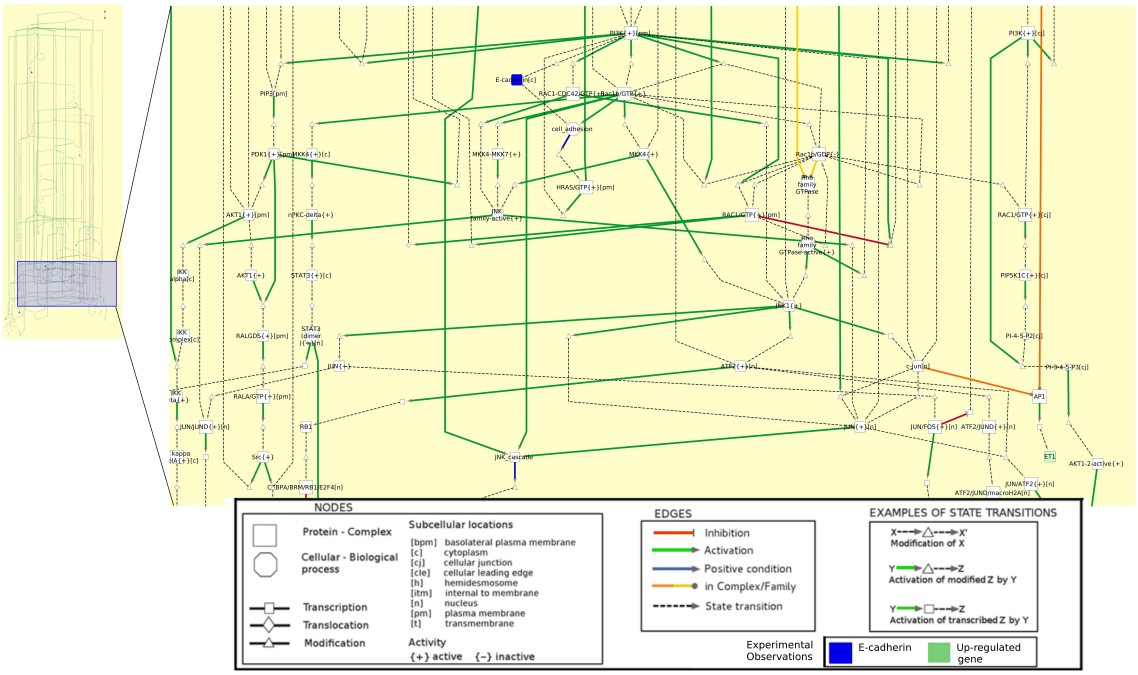



Figure 6.1 : **Extrait du modèle type RSTC du processus de différenciation des kératynocites.** La figure présentée ici est un zoom sur le modèle complet dont nous avons une minuscule image à gauche. Cet extrait permet de distinguer quelques nœuds du modèle en fonction des types associés à chaque nœud. À côté des nœuds, nous distinguons aussi les interactions. La petite légende au pied de la figure permet de comprendre le modèle.

Les nœuds différentiellement exprimés le sont à la suite d'une expérience grâce à des puces à ADN où le nœud d'entrée E-cadherin est stimulé au calcium et le signal se propage à travers la couche de signalisation (qui comporte les protéines de signalisation et les complexes) pour atteindre la couche transcription (qui, elle, comporte les facteurs de transcription et les gènes) par la suite et la couche des états cellulaires. Il est à noter que ce modèle comporte des boucles de rétro contrôle. Nous rappelons ici la figure 6.2 déjà présentée dans le chapitre 3 qui représente la dynamique des gènes différentiellement exprimés.

Afin de procéder à une modélisation hybride et une analyse du processus de différenciation des kératynocites, nous avons appliqué la démarche décrite dans le chapitre 3. Nous allons brièvement décrire quelques choix de modélisations spécifiques à ce système dans la

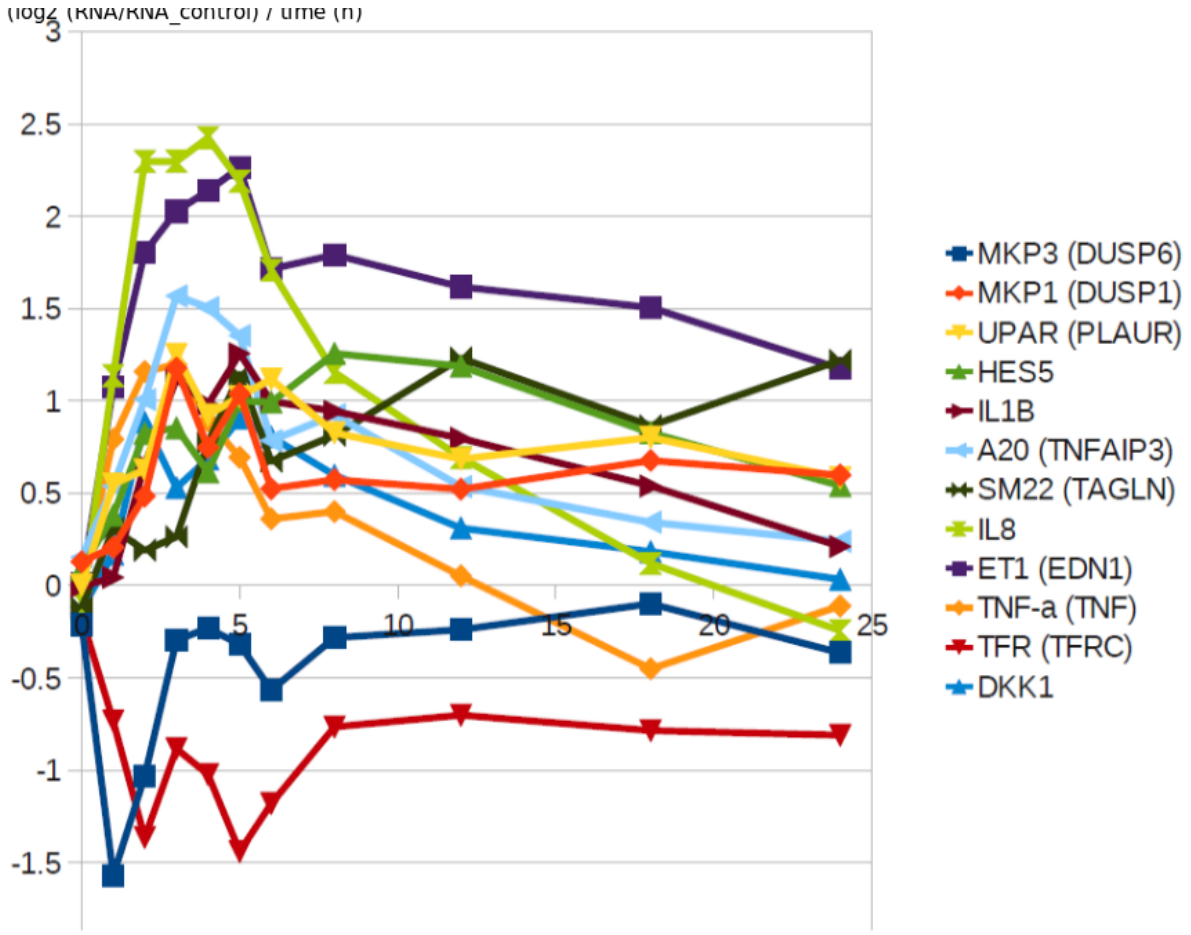



Figure 6.2 : **Données de séries temporelles.** Ces données représentent les mRNA différentiellement exprimés au cour d'une expérience où le nœud d'entrée E-cadherin du modèle de la différenciation des kératynocites a été stimulé au calcium. Ces données sont observées sur $24h$ (axe des X) et elles représentent le $\log_2$ du niveau d'expression par rapport au contrôle (axe de Y).

section 6.2.2, puis nous présenterons les résultats des simulations à la section 6.2.3 et les résultats des analyses statistiques des traces des simulations.

### 6.2.2 Choix de modélisation et hypothèses de simulations

Des choix de modélisation spécifiques ont été effectués pour mieux prendre en compte la spécificité de la différenciation des kératynocites. Ces choix concernent la représentation des différents composants en Frappes de Processus. Il s'agit de déterminer pour chaque type de nœud la sorte pour représenter ce nœud, le nombre de processus au sein de cette sorte. De plus, il faut déterminer le type de coopération pour les composants agissant en coopération sur un autre composant. Enfin, il faut dire comment nous traduisons en frappes les différents types d'interactions que nous observons dans les réseaux type RSTC. Nous présentons ici les principaux choix que nous avons effectués.

– Le nœud d'entrée E-cadherin est modélisé par une sorte avec deux processus. Dans son processus 0 il n'est pas stimulé et dans son processus 1 il est stimulé.

- Les protéines de signalisation sont modélisées par des sortes ayant chacune deux processus, à 0 elles sont inactives et à 1 elles sont actives.
- Les facteurs de transcription sont modélisés par des sortes ayant trois niveaux discrets chacune.
- Les gènes sont modélisés par des sortes ayant trois niveaux discrets chacun.
- Les états cellulaires sont modélisés par des sortes ayant deux niveaux discrets chacun. Au niveau 0, l'état est considéré comme non observé, et au niveau 1, il est considéré comme observé.

Après les hypothèses de modélisation, nous présentons les choix que nous avons effectués pour la simulation stochastique avec Pint[1].

- Pour E-cadherin (nœud d'entrée), nous avons choisi le processus 1 comme son processus initial. Puis nous avons introduit une auto frappe qui le fait passer de son processus 1 à son processus 0 pour représenter la fin de la stimulation au calcium.
- Pour les protéines de signalisation, nous avons choisi une activation et une inhibition rapide, dans le but de prendre en compte le fait que le signal transite rapidement dans cette couche. Pour cela, nous avons affecté des valeurs par défaut pour l'activation et l'inhibition des composants de cette couche à savoir 10 pour le taux (ce qui est plutôt rapide) et 50 pour l'absorption de stochasticité (ce qui permet de réduire au maximum la variance autour du taux).
- Pour les gènes, les valeurs des paramètres de la simulation sont estimées à partir des valeurs de séries temporelles comme nous l'avons décrit à la section 3.3 en page 65.

Nous tenons à préciser ici que plusieurs hypothèses ont été testées et celles que nous avons retenu et que nous présentons ici sont celles qui ont donné les meilleurs représentations des données de séries temporelles d'entrée. La prochaine section est consacrée à la présentation des résultats et l'analyse de la simulation stochastique.

### 6.2.3 La simulation stochastique et résultats

Les simulations ont été effectuées avec le logiciel Pint sur deux modèles en Frappes de Processus du processus de la différenciation cellulaire. Le premier modèle est obtenu sans introduction des sortes de synchronisation et le deuxième modèle comporte les sortes de synchronisation (section 3.2.4.2 en page 64). Nous avons choisi ces deux modèles pour illustrer la différence entre un modèle qui génère des oscillations artificielles et un modèle qui n'en génère pas.

La figure 6.3 illustre quelques traces des simulations obtenues sur le modèle sans sortes de synchronisation. Il apparaît clairement sur cette figure, que l'absence des sortes de synchronisation fait que nous observons l'apparition des oscillations artificielles. C'est par exemple le cas pour les gènes $MKP3$, $Hes5$, $IL1_{beta}$, $A20$, $SM22$, $IL8$. Ces oscillations artificielles dans le modèle perturbent la propagation du signal et créent une confusion entre le signal que nous cherchons à reproduire et celui qui est réellement observé. C'est pour palier ce problème que nous avons décidé d'introduire les sortes de synchronisation (voir section 3.2.4.2 en page 64).

Avec les sortes de synchronisation, le constat immédiat est qu'elle permettent effectivement d'éliminer les oscillations artificielles. De plus, nous réussissons à observer pour

[1] `http://process.hitting.free.fr`

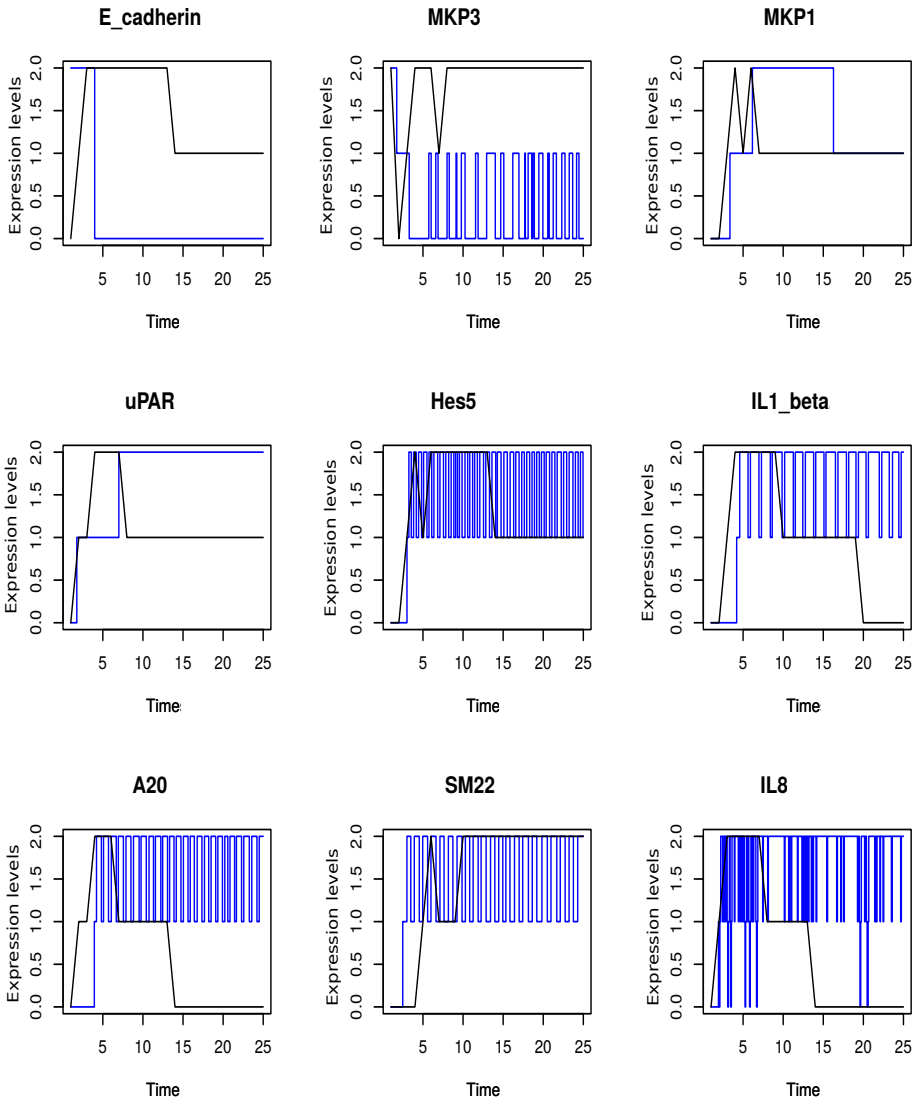



Figure 6.3 : **Résultats des simulations de** 9 **gènes sans l'introduction des sortes de synchronisation.** Les traces représentant les séries temporelles discrétisées sont en traits noir. Les traces représentant les simulations sont en bleu.

certains composants ($IL8$, $uPar$, $IL1_{beta}$, $ET1$, $A20$) une prédiction satisfaisante des courbes expérimentales. Cette prédiction permet d'obtenir à peu près les traces attendues. Cependant, il peut arriver que ces traces soient légèrement décalées d'un point de vue des délais nécessaires pour les observer. À côté des composants qui reproduisent une trace acceptable, il y a des composants qui reproduisent une trace avec des manquements dans la dynamique c'est-à-dire que la succession de processus traversés ne respecte pas l'ordre attendu. C'est par exemple le cas pour les composants $SM22$, $TNF\_alpha$, $TfR$.

Enfin, nous avons également observé la dynamique des phénomènes biologiques tels que l'adhésion cellulaire, la différenciation et l'arrêt du cycle cellulaire. La dynamique de ces phénomènes est généralement bien reproduite comme on peut le constater sur la figure 6.5. Cependant, nous observons aussi de temps en temps un décalage dans les délais qui n'est pas de très grande ampleur.

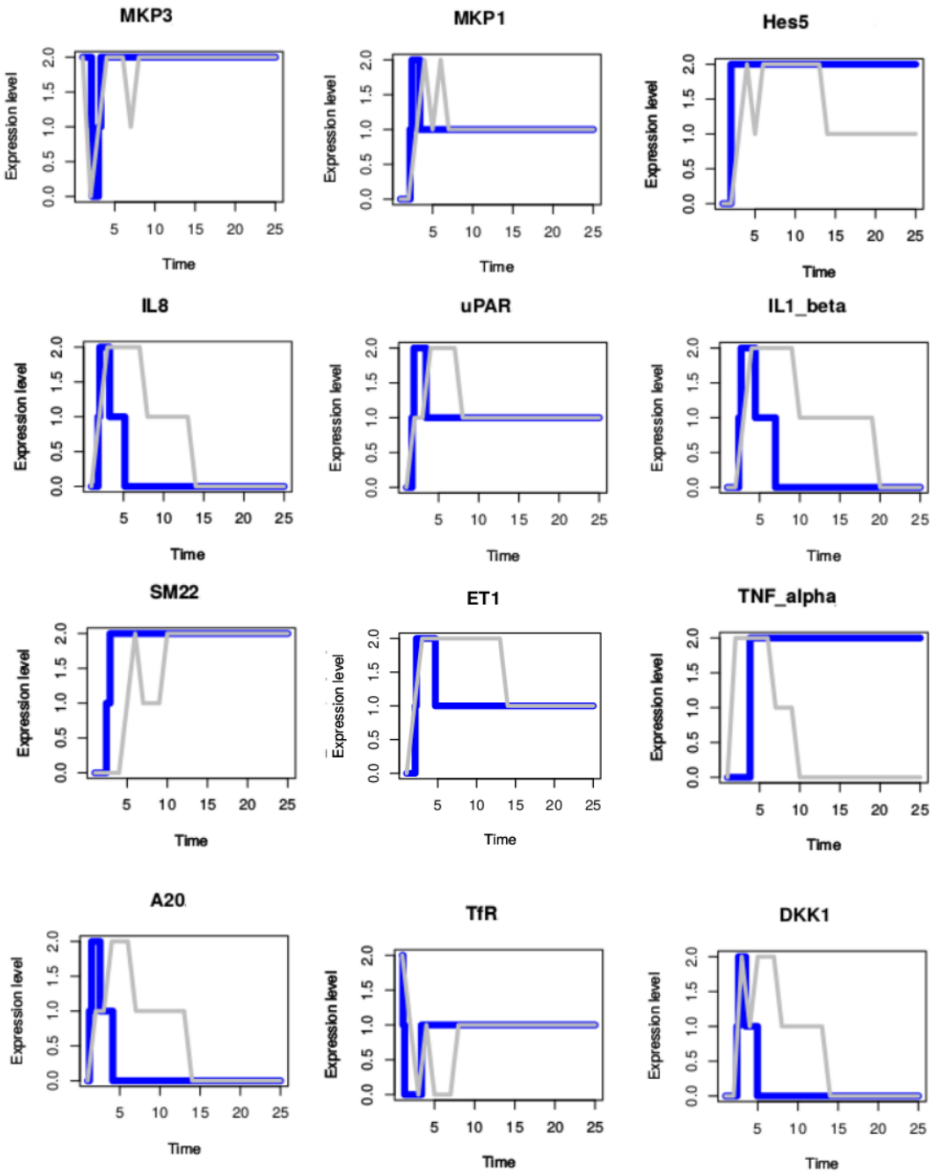



Figure 6.4 : **Résultats des simulations après introduction des sortes de synchronisation.** Les traces en gris représentent les données expérimentales après discrétisation des données de séries temporelles. Les traces en bleu représentent les résultats des simulations.

### 6.2.4 Analyse statistique de la simulation

La dynamique du système étant asynchrone et stochastique, sur une simulation, il est donc possible d'observer un comportement du système qui est susceptible de varier ou d'être différent à la prochaine simulation. C'est la raison pour laquelle nous utilisons l'analyse statistique des traces des composants obtenues après un grand nombre de simulations. Dans le présent cas d'étude, nous avons effectué 100 simulations et les résultats sont consignés dans le tableau 6.1 en page 142. Ce dernier contient pour chaque composant observé (deuxième colonne), le pourcentage de traces acceptantes (section 3.4.1) observées pour

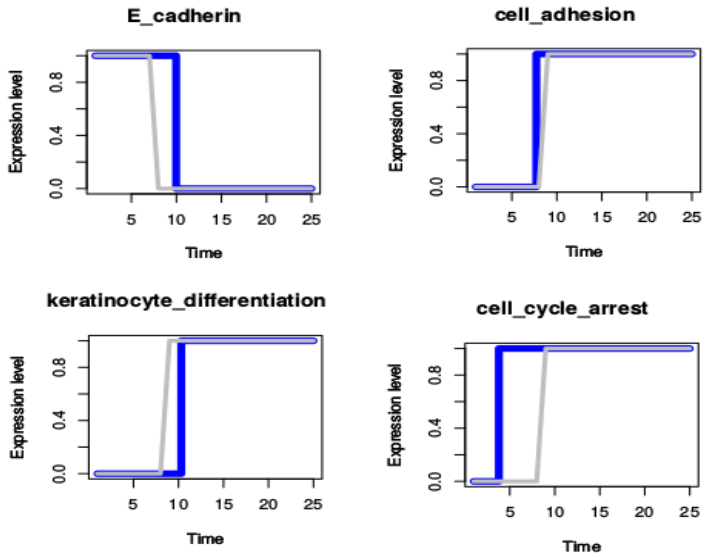



Figure 6.5 : **Résultats des prédictions des processus biologiques.** Les traces en gris représentent les expérimentations et les hypothèses tirées de la littérature. Les traces en bleu représentent les simulations d'E-cadherin et de trois processus biologiques.

l'ensemble des simulations. La première colonne contient le nom de l'automate qui reconnaît la trace acceptante du composant, la trace étant indiquée entre parenthèse après le nom l'automate. La troisième colonne représente le pourcentage d'acceptation sans aucune tolérance. La quatrième colonne représente le pourcentage d'acceptation avec une tolérance du type $T_1$. Nous rappelons que la tolérance dans la vérification d'une trace afin de déterminer si elle est acceptante ou non est la marge d'erreur autorisée (section 3.4.1 en page 70).

Le tableau montre qu'il y a des gènes pour lesquels le modèle reproduit le comportement attendu par rapport aux observations qui sont faites des séries temporelles indépendamment du fait qu'on utilise la tolérance de type $T_1$. Ces gènes sont $A20$, $IL1_{beta}$, $IL8$, $uPar$ et ils ont la particularité d'avoir une trace acceptante moins compliquée à reproduire. Cependant, ces observations ne sont pas confirmées par le gène $TNF_{alpha}$, qui bien qu'ayant une trace simple à reproduire reste non activé par le modèle, même avec la tolérance de type $T_1$. Ce qui peut indiquer des erreurs dans la modélisation ou l'incomplétude du modèle d'entrée en RSTC. Par ailleurs, il y a les composants qui ont un faible taux d'acceptation de leurs traces acceptantes, mais qui voient ce taux augmenter quand nous utilisons la tolérance de type $T_1$. C'est le cas pour les gènes $ET1$, $DKK1$, $Hes5$, $MKP1$, $SM22$, $MKP3$, $Tfr$.

## 6.3 Applications de l'identification des bifurcations sur des exemples biologiques

Nous présentons dans cette section une mise en œuvre de l'identification des transitions de bifurcation dans les systèmes biologiques. Dans la section 6.3.1 à la page suivante nous donnons une brève description des modèles choisis pour cette mise en application. Puis dans la section 6.3.2 en page 146 nous décrivons la méthode que nous avons utilisée et enfin nous présentons et commentons les résultats à la section 6.3.3 en page 147.

Table 6.1 : Résultats expérimentaux de l'analyse statistique des traces des composants après simulation et qui donne le pourcentage de traces acceptantes. La première colonne représente les automates ($\mathcal{A}_i(w)$, où $\mathcal{A}_i$ est l'automate et $w$ est le mot reconnu par l'automate $\mathcal{A}_i$) ceci est utilisé pour vérifier si une trace donnée est acceptée par le composant dans la deuxième colonne. Plusieurs composants peuvent avoir le même automate qui reconnaît leurs traces. Dans la troisième colonne figure le pourcentage de traces acceptantes ; dans la quatrième, le pourcentage de traces acceptantes avec une tolérance de type ($T_1$).

| Automate | Composant | % d'acceptation | % d'acceptation $T_1$ |
|---|---|---|---|
| $A_2(01210)$ | A20 | 91 | 100 |
| $A_2(01210)$ | $IL1_{beta}$ | 81 | 100 |
| $A_2(01210)$ | IL8 | 93 | 100 |
| $A_2(01210)$ | $TNF_{alpha}$ | 0 | 0 |
| $A_3(01211)$ | uPar | 76 | 99 |
| $A_3(01211)$ | ET1 | 8 | 19 |
| $A_4(0121210)$ | DKK1 | 13 | 43 |
| $A_5(0121211)$ | Hes5 | 0 | 17 |
| $A_5(0121211)$ | MKP1 | 9 | 97 |
| $A_6(0212)$ | SM22 | 11 | 100 |
| $A_7(02010)$ | MKP3 | 11 | 98 |
| $A_8(02121)$ | Tfr | 0 | 94 |

### 6.3.1 Présentation des modèles biologiques étudiés

**Phage Lambda**

Le phage lambda (Enterobacteria phage λ) est un virus bactériophage qui infecte la bactérie Escherichia coli. Ce bactériophage est un virus à ADN double brin, empaqueté dans une capside icosaédrique prolongée d'une queue et de fibrilles permettant l'ancrage sur la bactérie. Le phage lambda est sujet à deux cycles d'évolution possibles : le cycle **lytique** qui conduit à la réplication rapide du virus et à la mort de la cellule hôte, en l'occurrence Escherichia coli, et le cycle **lysogénique** durant lequel il insère son génome de manière dormante dans celui de la bactérie et subit des réplications en même temps que le reste du génome bactérien. Il peut ainsi se propager à la descendance de la cellule infectée. Ce cycle aboutit généralement à un cycle lytique et à l'excision du génome phagique hors du génome bactérien. L'existence de la voie lysogénique fait de lambda un phage « tempéré ». L'intégration du génome du virus dans le chromosome bactérien se fait via un mécanisme de recombinaison spécifique, sur un site bien défini entre les opérons gal (utilisation du galactose) et bio (synthèse de la biotine).

La biologie moléculaire et la biochimie de ce virus sont particulièrement bien connues, ce qui fait qu'il a été largement utilisé comme vecteur de clonage de l'ADN, ainsi que pour permettre l'insertion de séquences d'ADN spécifiques dans le génome d'E. coli, via le mécanisme d'intégration lysogénique du virus.

Après l'injection de l'ADN phagique dans le cytoplasme, il y a « prise de décision »

entre l'entrée en cycle lytique ou la croissance lysogénique, l'intégration dans le génome bactérien sous forme de prophage. Les facteurs influençant cette prise de décision sont mal connus, mais il semble que les conditions physiologiques de l'hôte participent à cette étape.

Toutefois, plusieurs gènes ont été identifiés comme prenant part à ce processus, il s'agit des gènes $cI$,$cII$,$cro$,$N$,$cIII$. Des modélisations ont été proposées, notamment (Thieffry & Thomas, 1995), pour l'analyse et la compréhension de la dynamique des interactions de ces gènes dans le processus qui conduit au choix du cycle à adopter par le phage une fois dans la bactérie. Le modèle illustré par la figure 6.6 présente une modélisation des interactions entre les quatre gènes identifiés comme prenant part à la différenciation chez le phage lambda.

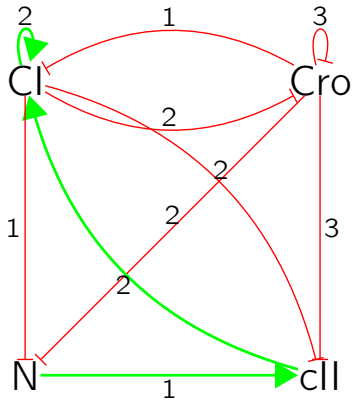



Figure 6.6 : **Modèle discret représentant les interactions entre les quatre gènes prenant par au processus qui consiste à déterminer dans quel cycle (lytique ou lysogénique).** Nous avons utilisé le formalisme de Thomas (où nous avons remplacé les signes par les couleurs (+) par le vert en trait en gras et (−) par le rouge en trait en fin) pour modéliser ces interactions. Les quatre nœuds représentent les quatre principaux gènes, les flèches vertes représentent les activations et les flèches rouges les inhibitions. Les étiquettes sur les transitions représentent les seuils.

**EGF_TNF**

Le modèle *EGF_TNF* est un modèle qui combine deux voies de signalisation. La voie de signalisation EGF (epidermal growth factor) facteur de croissance épidermique et la voie de signalisation TNF_$\alpha$ (tumor necrosis factor) facteur de nécrose tumorale.

La voie EGF conduit principalement à une augmentation de l'activité de transcription de la cellule et à une augmentation de la réplication de l'ADN, ce qui se traduit par une stimulation de la croissance cellulaire et de la division (mitose).

EGF est une molécule qui, avec des protéines EGF-like, se lie sur un récepteur de type EGF qui lui est un proto-oncogène, c'est-à-dire qu'il peut s'intégrer dans un ARN viral et faire en sorte que non seulement la cellule hôte participe à la réplication du virus comme c'est le cas pour tous les virus mais qu'en plus la cellule hôte soit elle aussi transformée et devienne une cellule cancéreuse, dans laquelle le récepteur à l'EGF n'aura plus son fonctionnement normal et participera à la multiplication cellulaire. Dans le cas

du récepteur à l'EGF celui-ci dans une cellule anormale va être sur-exprimé ou alors il y aura des mutations activatrices, conduisant dans tous les cas à une prolifération cellulaire à cause de la cascade de réaction décrite plus haut.

La voie TNF_$\alpha$, quant à elle, peut être responsable d'au moins trois évènements :

- activation de la voie NF$\kappa$B qui permet une inhibition de l'apoptose ;
- activation de l'apoptose ;
- nécrose programmée.

Le modèle que nous présentons ici est tiré de (MacNamara, Terfve, Henriques, Bernabé & Saez-Rodriguez, 2012; Chaouiya, Bérenguier, Keating, Naldi, van Iersel, Rodriguez, Dräger, Büchel, Cokelaer, Kowal, Wicks, Gonçalves, Dorier, Page, Monteiro, von Kamp, Xenarios, de Jong, Hucka, Klamt, Thieffry, Le Novère, Saez-Rodriguez & Helikar, 2013) qui ont donc proposé un modèle combinant ces deux importantes voies de signalisation chez les mammifères que sont EGF et TNF_$\alpha$. Ces deux voies stimulent les cascades ERK, JNK et p38 MAPK, les voies de signalisation PI3K/AKT et les voies de signalisation NF$\kappa$B. De plus, le modèle comporte des croisements entre ces deux voies de signalisation ainsi que des rétro-contrôles. Un rétro-contrôle se trouve dans la cascade NF$\kappa$B et un autre se trouve dans la cascade MAPK. La figure 6.7 donne une illustration du modèle.

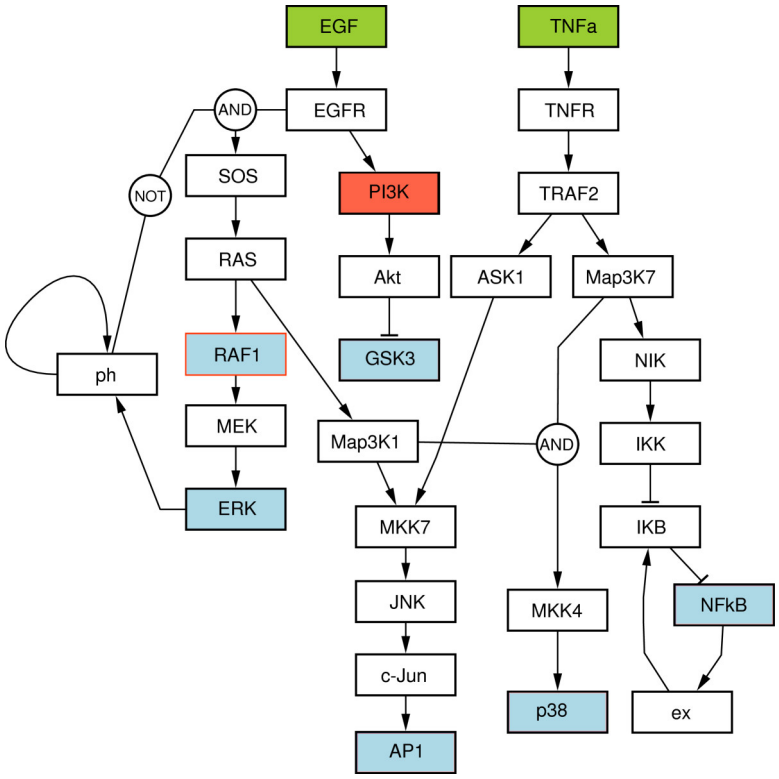



Figure 6.7 : **Modèle booléen obtenu de CellNOpt. Ce modèle combine deux voies de signalisation EGF et TNF alpha.** Les différentes couleurs définissent les modèles expérimentaux des données utilisées pour tester le modèle. Les carrés verts représentent une stimulation externe, Les carrés rouges correspondent aux composants bloqués par un inhibiteur à la kinase et les carrés bleus correspondent aux composants pour lesquels l'expression a été mesurée.

### T Helper différentiation

Nous avons présenté à la section 6.2.1 la différenciation cellulaire. Nous nous sommes particulièrement intéressés à la différenciation des cellules de la peau. Dans cette section, nous présentons la différenciation des lymphocytes T.

La différenciation des lymphocytes T est le processus par lequel un lymphocyte T naïf, ayant une spécificité donnée pour un antigène, acquiert également des capacités effectrices particulières. Ce processus est induit dans le ganglion lymphatique au cours de l'interaction entre une cellule présentatrice d'antigène présentant l'antigène spécifique et le lymphocyte T naïf.

Selon l'environnement dans lequel elles ont été activées avant de rejoindre les organes lymphoïdes, les cellules présentatrices d'antigènes produisent différentes cytokines pouvant induire différents programmes de différenciation des lymphocytes T. Il existe ainsi différentes voies de différenciation possibles pour un lymphocyte T, plus ou moins exclusives les unes des autres. Aussi, les différentes populations de lymphocytes T se distinguent notamment par la nature des cytokines qui sont produites. On distingue ainsi :

- Les lymphocytes Th1 qui produisent les interférons IFN$\gamma$ qui sont des protéines de la famille des cytokines.
- Les lymphocytes Th2 qui produisent surtout de l'interleukine 4 (IL-4) (mais aussi de l'interleukine 10 (IL-10))
- Les lymphocytes T régulateurs qui sont inhibiteurs et produisent du TGF$\beta$
- Les lymphocytes Th17 qui produisent de l'IL-17 et de l'IL-22
- Les lymphocytes Tr1 qui produisent de l'IL-10

La différenciation des lymphocytes T naïfs dépend de plusieurs conditions notamment des cytokines présentes lors de leur activation et des facteurs de transcription impliqués. En ce qui concerne les cytokines présentes, certaines empêchent l'orientation vers un type particulier, d'autres favorisent la différenciation dans une direction précise. On peut résumer cela ainsi :

- IL-12 $\Longrightarrow$ Th1
- IL-4 $\Longrightarrow$ Th2
- TGF$\beta$ $\Longrightarrow$ Treg
- TGF$\beta$ + IL6 $\Longrightarrow$ Th17

Pour ce qui est des facteurs de transcription, on considère que chaque lignée de lymphocytes T dépend de l'expression et de la régulation d'un facteur de transcription spécifique. On recense :

- Le facteur de transcription Tbet pour les Th1
- Le facteur de transcription GATA-3 pour les Th2
- Le facteur de transcription Foxp3 pour les Tregs
- Le facteur de transcription ROR$\gamma$T pour les lymphocytes Th17

Le modèle que nous utilisons est tiré de (Abou-Jaoudé, Monteiro, Naldi, Grandclaudon, Soumelis, Chaouiya & Thieffry, 2015). La figure 6.8 présente ce modèle. D'après la description que nous venons de faire de la différenciation des lymphocytes T, nous pouvons raisonnablement supposer que ce modèle contient plusieurs attracteurs qui seraient associés à différents sous-types cellulaires.

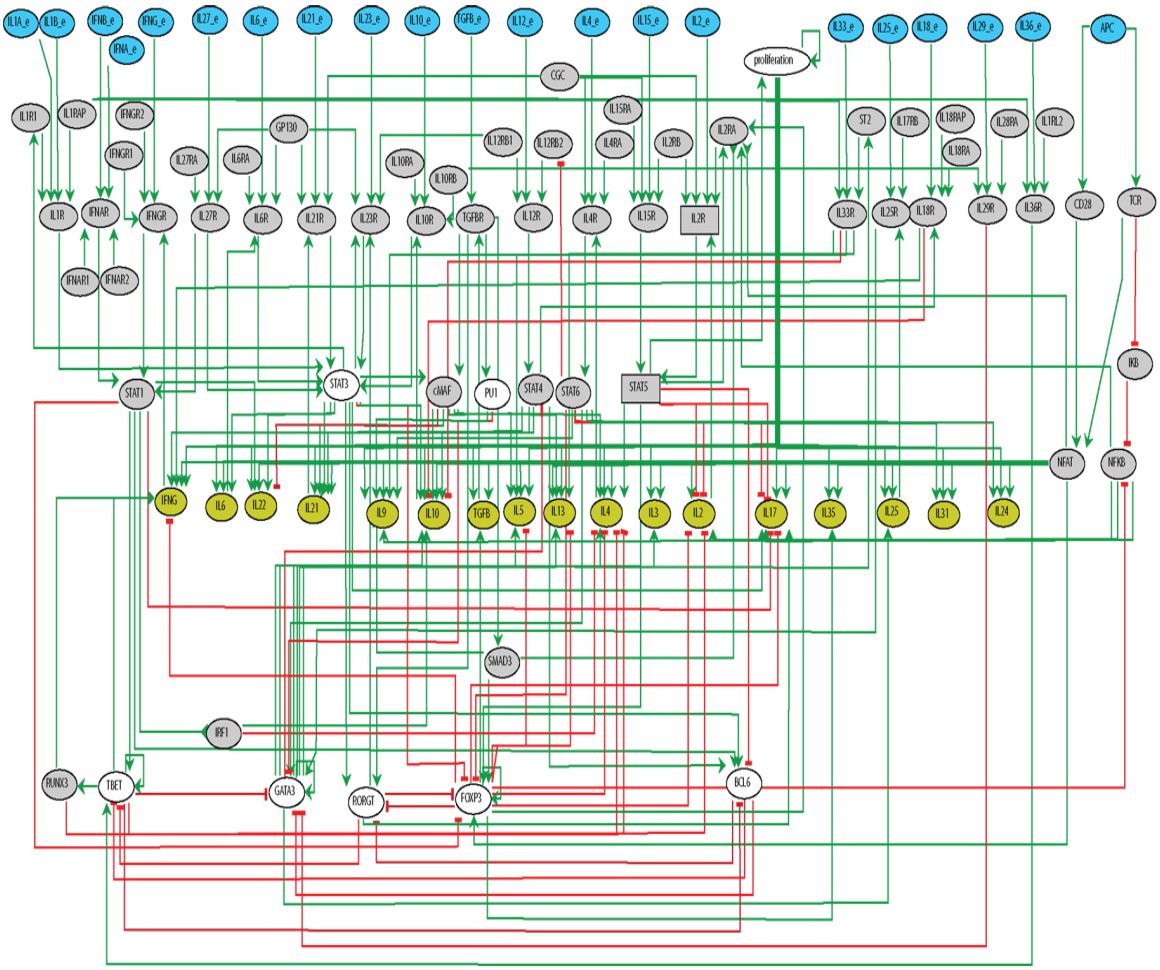



Figure 6.8 : **Modèle booléen tiré de (Abou-Jaoudé et al., 2015) pour la différenciation des lymphocytes T. Ce modèle comporte** 101 **composants (parmi lesquels** 21 **nœuds d'entrée) et** 221 **interactions.** Les nœuds d'entrés sont en bleu, les nœuds qui sécrètent les cytokines sont en olive. Les arcs verts représentent les activations, tandis que les arcs rouges représentent les inhibitions. Les ellipses représentent les composants booléens et les rectangles les composants ternaires. Les composants gris représentent les nœuds qui peuvent être supprimés pour une éventuelle réduction du modèle.

### 6.3.2 Description de la méthode

La méthode que nous avons utilisée pour tester la démarche repose sur le théorème 5.1 (section 5.3.1 en page 114) qui stipule que si l'ensemble des transitions de bifurcation est non vide, alors le graphe des états possède aux moins deux attracteurs. De la même façon, si un graphe des états possède au moins deux attracteurs, alors l'ensemble des transitions de bifurcation n'est pas vide. Aussi, notre démarche pour sous-approximer l'ensemble des transitions de bifurcation peut donc potentiellement identifier les transitions de bifurcations.

Ainsi, sur le modèle du **Phage Lambda**, qui a l'avantage d'être petit et d'avoir deux attracteurs bien connus, nous avons proposé un modèle en réseau d'automates équivalent au fonctionnement du modèle du phage lambda. Nous avons pu vérifier que quand l'état initial et le but appartiennent au même attracteur, notre méthode retourne un ensemble vide des transitions de bifurcation. Si par contre, l'état initial n'appartient pas au même attracteur que le but, notre méthode peut renvoyer un ensemble des transitions de bifur-

cation non vide. Nous avons donc pu confirmer sur cette exemple, que notre méthode est sûre au sens où elle ne renvoie pas de faux positifs. Mais elle peut cependant laisser échapper quelques transitions de bifurcations.

Nous avons généralisé l'application de notre méthode sur les modèles de **EGF/TNF_$\alpha$** et **Th_différenciation**. Pour ces modèles, nous avons défini des jeux de test en essayant de choisir le but et l'état initial de façon à ce qu'ils ne soient pas dans le même attracteur.

### 6.3.3 Résultats

Le tableau 6.2 résume les résultats obtenus lors de l'expérimentation par notre approche pour la construction de la sous-approximation des transitions de bifurcation sur chacun des modèles testés. Les résultats pour tous les modèles sont comparés avec une méthode exacte qui utilise le model-checker symbolique NuSMV [2] (voir colonne M-C (NuSMV) du tableau 6.2). Deux constats principaux peuvent être faits :

- Les approximations des ensembles des transitions de bifurcation sont des sous-ensembles de l'ensemble des transitions de bifurcation. Ce qui valide bien le théorème 5.2 et le théorème 5.3 de la section 5.4 en page 120.
- Quand la vérification s'avère impossible à cause de la taille du graphe des états, la méthode proposée pour sous-approximer les transitions de bifurcation donne des résultats très rapidement.

Le premier constat énoncé plus haut indique clairement que la méthode proposée sous-approxime l'ensemble des transitions de bifurcation. Elle montre également comme nous l'avons déjà signalé que la méthode peut laisser échapper quelques unes. Toutefois, toutes les transitions identifiées par la sous-approximation sont bien des transitions de bifurcation, ce que nous avons pu vérifier avec le model-checker symbolique NuSMV. Cependant, la deuxième constatation rappelle que, quand le graphe des états est trop grand pour une vérification exacte, la méthode développée réussit à fournir des résultats (incomplets) très rapidement. Enfin, cette démarche démontre bien qu'il y a un compromis à faire entre la qualité de la vérification (exacte/approximée) et la taille des modèles qui peuvent être traités. Ce travail est d'un intérêt certain pour le contrôle des systèmes biologiques parce qu'il ouvre la voie à l'identification des cibles pour la reprogrammation des mécanismes cellulaire.

### 6.3.4 Évaluation quantitative (probabiliste)

Comme indiqué dans la section 5.1 en page 109, identifier les transitions de bifurcation peut permettre de prévenir certains comportements. Dans cette section, nous montrons sur un exemple biologique (phage lambda) pour lequel nous avons identifié un sous-ensemble des transitions de bifurcation qu'en augmentant la vitesse associée à ces transitions on réduit le probabilité d'observer un comportement souhaité.

Dans la section 6.3.4.1 à la page suivante nous présentons brièvement le framework PRISM[3] sur lequel nous nous sommes appuyés pour effectuer nos expériences. Puis, nous décrivons le protocole et la méthode d'expérimentation utilisée dans la section 6.3.4.2 en page 149. Nos commentaires sur les résultats seront présentés dans la section 6.3.4.3 en page 149.

[2] `http://nusmv.fbk.eu/`

[3] `http://www.prismmodelchecker.org/manual/Main/Welcome`

| Automata Network | Goal | M-C (NuSMV) | | (I3) | | | (I3#) | |
|---|---|---|---|---|---|---|---|---|
| | | $\|t_b\|$ | Time | $\|\text{pfx}\|$ | $\|t_b\|$ | Time | $\|t_b\|$ | Time |
| Lambda phage | $Cl_2$ | 10 | 0.1$s$ | 45 | 6 | 0.1$s$ | 0 | 0.2$s$ |
| $\|\Sigma\| = 4 \quad \|T\| = 11$ | $Cro_2$ | 3 | 0.1$s$ | | 3 | 0.1$s$ | 2 | 0.3$s$ |
| EGF/TNF | $NFkB_0$ | 5 | 0.2$s$ | 52 | 4 | 0.1$s$ | 2 | 0.1$s$ |
| $\|\Sigma\| = 28 \quad \|T\| = 55$ | $IKB_1$ | 5 | 0.2$s$ | | 3 | 0.1$s$ | 2 | 0.1$s$ |
| Th_th1 | $BCL6_1$ | 8 | 13$s$ | 444 | 6 | 16$s$ | 5 | 23$s$ |
| $\|\Sigma\| = 101 \quad \|T\| = 381$ | $TBET_1$ | 11 | 14$s$ | | 5 | 10$s$ | 4 | 24$s$ |
| Th_th2 | $GATA3_0$ | 8 | 60$s$ | 3264 | 7 | 24$s$ | 4 | 24$s$ |
| $\|\Sigma\| = 101 \quad \|T\| = 381$ | $BCL6_1$ | 7 | 600$s$ | | 5 | 25$s$ | 4 | 25$s$ |
| Th_th17 | $RORGT_1$ | 18 | 48$s$ | 2860 | 9 | 23$s$ | 8 | 26$s$ |
| $\|\Sigma\| = 101 \quad \|T\| = 381$ | $BCL6_1$ | 7 | 26$s$ | | 5 | 23$s$ | 4 | 24$s$ |
| Th_HTG | $BCL6_1$ | OT | | OT | | | 6 | 61$s$ |
| $\|\Sigma\| = 101 \quad \|T\| = 381$ | $GATA3_1$ | | | | | | 7 | 34$s$ |

Table 6.2 : Résultats expérimentaux de l'identification des transitions de bifurcation selon que la méthode exacte (M-C (NuSMV)) ou (I3) ou (I3#) (condition suffisante) est utilisée. Les modèles Th_th1, Th_th2, Th_th17, Th_HTG correspondent au même réseau d'automates mais ont différents états initiaux. Pour chaque modèle, deux buts à atteindre différents ont été choisis pour être testés. $|\Sigma|$ est le nombre d'automates et $|T|$ le nombre de transitions ; $|\text{unf-prefix}(s_0)|$ est la taille du préfixe de dépliage depuis l'état initial du modèle ; $|t_b|$ est le nombre de bifurcations identifiées. Le temps d'exécution à été obtenu sur une machine Intel® Core™ i7-4600M 2.90GHz CPU avec 7.7GiB de RAM. OT signifie que le temps d'exécution a dépassé une heure.

#### 6.3.4.1 Présentation du framework PRISM

PRISM (Kwiatkowska et al., 2011) est un vérificateur de modèles probabilistes. Il apporte une vérification formelle efficace de CMTC (Continuous-time Markov chain). En PRISM, la composante essentielle est *le module*. Chaque module possède un ensemble fini de variables locales et permet de spécifier les transitions. L'état global du modèle dénoté par $V$ est l'union des variables locales de tous les modules. Une action dans PRISM est spécifiée comme suit :

$$[act]guard \rightarrow r : (x_1' = u_1)\&(x_2' = u_2) \tag{6.1}$$

où $act$ est le libellé optionnel de l'action, $guard$ est un prédicat optionnel sur $V$, $x_i$ est une variable locale et $u_i$ une fonction sur $V$. $r \in \mathbb{R}$ est le taux d'utilisation de l'action. Il est supposé égal à 1 quand il n'est pas spécifié. Pour être applicable, une action libellée doit être synchronisée avec une autre action d'un autre module possédant le même libellé. Le taux d'une action synchronisée est le produit du taux des deux actions. $x_i'$ représente la valeur de $x_i$ une fois que l'action a été appliquée.

#### 6.3.4.2 Méthode d'évaluation

Pour effectuer notre évaluation afin de montrer quel peut être l'impact des transitions de bifurcations, nous avons exporté le modèle du lambda phage en réseau d'automates vers le formalisme de PRISM.Puis, nous avons affecté la même valeur $r_n$ du rate à toutes les actions exceptées celles qui ont été identifiées comme étant des bifurcations de transition. Pour les bifurcations de transition, nous avons donné une valeur $r_b \in [r_{min}, r_{max}]$où $r_{min} = r_n$.

Nous avons alors demandé le calcul de la probabilité d'atteindre le but $CI = 2$ quand $r_b$ varie de $r_{min}$ à $r_{max}$.

#### 6.3.4.3 Résultats

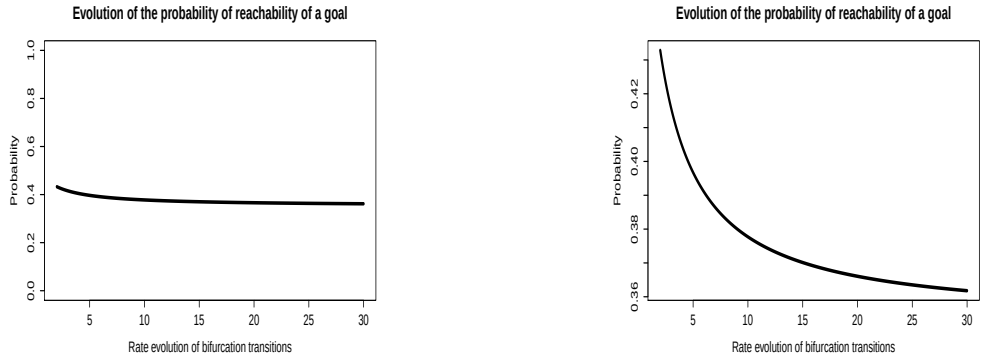



Figure 6.9 : **Évolution de la probabilité d'atteindre le but $CI = 2$ en fonction de la vitesse de franchissement des transitions de bifurcation.** (à gauche) évolution sur une échelle de $[0, 1]$. (à droite) zoom sur l'intervalle de probabilité $[0.36, 0.5]$.

Les résultats que nous pouvons observer à la figure 6.9 montrent clairement une réduction de la probabilité d'accéder au but quand la vitesse des transitions identifiées comme étant des transitions de bifurcation augmente. Cette probabilité qui est initialement de 0.43 quand toutes les transitions ont le même taux diminue progressivement pour atteindre moins de 0.37. Ce résultat témoigne de l'intérêt de s'intéresser à des transitions de bifurcation. En effet, elles constituent des cibles thérapeutiques potentielles parce qu'elles peuvent être identifiées et inhibées en vue d'empêcher la survenue des comportements non souhaités. Elles peuvent aussi, sans être inhibées, modifiées (en favorisant ou défavorisant selon le comportement souhaité) privilégier un comportement du système sur un autre.

## 6.4 Discussion

Dans ce chapitre, nous avons présenté des applications des méthodes développées dans cette thèse pour la modélisation, l'analyse et la vérification des grands réseaux de régulation biologique. Des travaux récents (Paulevé, 2011) (Folschette, 2014) avaient déjà montré la capacité du formalisme de Frappes de Processus et de sa paramétrisation temporelle et stochastique à reproduire avec beaucoup de précision les phénomènes quantitatifs, c'est par exemple le cas dans le processus de segmentation chez les métazoaires.

Avec la modélisation et l'intégration des données de séries temporelles dans le processus de différenciation des kératynocites, nous avons montré que ce formalisme pouvait

s'appliquer également aux modèles comportant des centaines de composants et d'interactions. Nous avons montré qu'il est possible de reproduire les dynamiques avec une grande précision. Nous avons également montré que ces comportements pouvaient être vérifiés par une analyse statistique, par l'analyse des traces acceptantes après un grand nombre de simulations. Toutefois, nous n'avons pas effectué les expérimentations qui valident les traces en prenant en compte les délais. Cette partie fera l'objet des travaux futurs.

Concernant l'analyse statique des bifurcations, nous avons appliqué notre démarche sur trois systèmes biologiques de tailles différentes. Sur les modèles de petites et moyenne tailles comme le modèle du Lambda phage et EGF/TNF, nous réussissons à construire des sous-ensembles des transitions de bifurcation en quelques dixièmes de secondes. Pour le modèle de la différenciation des lymphocytes T qui comporte 101 composants et 381 transitions, nous avons testé quatre configurations différentes selon le choix d'états initiaux différents, pour faire varier la taille du préfixe (du dépliage de l'ensemble des états atteignables depuis un état initial). Même dans les cas des grands réseaux, notre approche a pu produire les résultats en une trentaine de secondes dans la plupart des cas. Cependant, nous n'avons pas expérimenté le calcul des ensembles des transitions de bifurcations en sur-approximation pour pouvoir évaluer combien de transitions, manquaient dans notre sous-approximation.

Nous avons également pu vérifier en utilisant le logiciel PRISM, dans le cas du modèle du Lambda phage que les transitions que nous avons identifiées comme étant des transitions de bifurcation, contribuaient à réduire la probabilité d'atteindre le but quand on augmentait la vitesse de franchissement de ces transitions. Ces applications nous confortent dans l'idée que nous pouvons estimer les probabilités de ne pas observer un comportement donné dans la dynamique des grands systèmes biologiques en passant par le calcul des transitions de bifurcations.

# Chapitre 7

# Conclusion et perspectives

Ce chapitre récapitule les contributions apportées dans cette thèse. Les principales problématiques de cette thèse étaient de permettre : (1) l'introduction d'une dimension aussi bien stochastique que temporelle dans les modèles à large échelle (des centaines de composants). Cette introduction permet ainsi de prendre en compte la notion de **chronométrie**. (2) L'**analyse des propriétés quantitatives** (temps, délai d'observer ou de ne pas observer un comportement) sur dans les modèles à large échelle. (3) L'identification précise des comportements satisfaisant les propriétés et les comportements ne les satisfaisant pas (grâce à la détermination de **Bifurcations**).
Les contributions de cette thèse ont permis de mettre en évidence quelques pistes de recherche dont nous présenterons quelques une dans ce chapitre. Parmi les plus importantes nous évoquerons : (1) l'utilisation des méthodes de fouille de données notamment **les règles d'association** pour analyser les traces des simulations stochastiques afin de pouvoir découvrir les relations non encore connues entre les composants. (2) L'analyse **statique des propriétés quantitatives** sur des **modèles paramétriques**. Cette perspective, permettra de prendre en compte les modèles, où les paramètres ne sont pas fixes, mais varient dans une plage de valeurs et sont donc représentés par des paramètres.

Les bases de données associées aux systèmes biologiques offrent une quantité énorme de modèles. Ces modèles sont de tailles de plus en plus importantes pouvant aller à des centaines voire des milliers de composants. Une approche communément utilisée en biologie des systèmes afin de modéliser et de comprendre les dynamiques au sein de ces systèmes est la modélisation discrète. Cette modélisation permet une abstraction de la complexité inhérente à ces systèmes tout en conservant des propriétés intéressantes. Malgré cette puissante abstraction, la modélisation discrète souffre encore d'une explosion combinatoire des comportements. Ceci rend les méthodes classiques de vérification formelle inefficaces et limitées à l'étude des systèmes de taille restreinte.

Par ailleurs, la disponibilité des données expérimentales notamment des séries temporelles offrent la possibilité de prendre en compte la notion de **chronométrie** dans le

processus de modélisation. Cet ajout permet de proposer des modèles hybrides sur lesquels il est possible d'étudier des propriétés dynamiques beaucoup plus précises notamment des dynamiques quantitatives.

Comme nous l'avons mentionné au chapitre 2, de nombreux travaux ont permis l'introduction à la fois des formalismes et des méthodes d'analyse pour permettre de prendre en compte la dynamique souvent très complexe des systèmes biologiques. De plus, nous avons vu qu'il est très difficile de prendre en compte à très grande échelle les paramètres temporels et stochastiques, et de faire ainsi ressortir la notion de délai dans la modélisation.

Comme pour le problème de l'analyse des propriétés dynamiques qualitatives, l'analyse fine des modèles hybrides (espace des états discrets et dynamique continue et stochastique) à large échelle reste difficile. L'explosion combinatoire de l'espace des états reste très problématique. En effet, une étude des propriétés dynamiques quantitatives exactes nécessite souvent d'abstraire le modèle comme une chaîne de Markov à temps continu et de procéder à son étude. Comme pour l'analyse des propriétés essentielles des dynamiques qualitatives, il est plus que nécessaire de faire recours à des méthodes d'analyse efficaces.

Dans cette thèse, nous avons apporté une contribution à la modélisation hybride, l'analyse et la vérification formelle des propriétés quantitatives des grands réseaux de régulation biologique. En effet, la modélisation hybride a été obtenue par l'estimation et l'intégration des données temporelles et stochastiques dans les réseaux d'automates asynchrones. Les modèles hybrides ainsi obtenus sont des réseaux d'automates asynchrones stochastiques ou une de leurs restrictions les *Frappes de Processus* précédemment introduites par (Paulevé, 2011). L'analyse et la vérification efficace de tels modèles nécessite des méthodes originales. Ainsi, nous avons tiré avantage des analyses statiques par interprétation abstraite introduites pour la vérification des propriétés qualitatives dans les Frappes de processus pour proposer une analyse statique par interprétation abstraite des propriétés quantitatives dans les réseaux d'automates asynchrones stochastiques. Toujours partant des analyses statiques très efficaces par interprétation abstraite développées pour la vérification des propriétés d'accessibilité au sein des Frappes de Processus d'une part et du raisonnement et de la programmation logique d'autre part, nous avons apporté une contribution qui va dans le sens du contrôle des réseaux d'automates par l'identification des **bifurcations**.

## 7.1 Contributions

Nous rappelons ici les contributions majeures de cette thèse :

**Formalisation en Frappes de Processus et réseaux d'automates stochastiques des réseaux de régulation biologique type RSTC (multi Layer Receptor Transcription Celle State).**

Le formalisme des réseaux d'automates asynchrones et sa restriction en Frappes de Processus permet une modélisation formelle des réseaux de régulation biologique. Dans les réseaux d'automates, chaque composant est représenté par un automate. Chaque automate possède à son tour un ensemble fini d'états locaux (niveaux discrets). À chaque instant un automate est dans un état local donné. le changement d'état local au sein d'un même automate est conditionné par la présence d'un ensemble (potentiellement vide) d'états locaux qui représentent une pré-condition pour la transition. C'est donc un forma-

lisme bien adapté pour la modélisation formelle des RRB. Aussi nous avons proposé une approche pour modéliser en réseaux d'automates les RRB type RSTC. Cette approche est basée sur une **détection automatique des motifs minimaux** (section 3.2) qui sont en suite transformés en réseaux d'automates (section 3.2.4). Étant tout aussi expressifs que d'autres formalismes, les réseaux d'automates permettent de construire des vérifications formelles par des analyses statiques très efficaces. De plus, un intérêt majeur de la détection automatique des motifs dans les RRB de type RSTC est que cela permet de formaliser les RRB disponibles dans les bases de données publiques comme Pathway Interaction Database (PID) (Schaefer et al., 2009*a*). Ainsi, les analyses développées pour les réseaux d'automates peuvent désormais être accessibles pour l'analyse ces RBB.

**Intégration des données de séries temporelles.**

Dans le but de raffiner la dynamique des transitions dans les modèles formels, notamment les réseaux d'automates asynchrones et les Frappes de Processus en particulier, nous avons procédé à **l'estimation des paramètres temporels et stochastiques des données de séries temporelles** (section 3.3). Ces paramètres sont le taux avec lequel les transitions sont appliquées et l'absorption de stochasticité qui est le paramètre qui permet une réduction de la variance autour du taux d'utilisation d'une transition ou d'une action. Nous avons montré comment, partant d'une courbe d'expression d'un composant, nous estimons les délais nécessaires pour effectuer les transitions. Les données d'expression utilisées dans le cadre de l'application de notre méthode ont été obtenues du Centre Allemand de Lutte contre le Cancer (DKFZ) et avec la collaboration du Pr. Dr. Peter Angel.

**Simulation stochastique et validation des modèles par analyse statistique des traces.**

Dans le but de vérifier statistiquement si le modèle construit pouvait reproduire les traces équivalentes aux données expérimentales, nous avons procédé à des simulations stochastiques. Pour cela, nous avons utilisé le simulateur Pint qui implémente une machine abstraite générique pour la simulation markovienne et non-markovienne. La puissance de ce simulateur nous a permis d'effectuer des centaines de simulations sur des modèles contenant des centaines de composants et d'interactions. Les résultats de ces simulations ont été analysées statistiquement par des programmes que nous avons développés pour l' **analyse des traces** (section 3.4). Dans le cadre de ce travail, nous avons proposé un sujet de stage pour un étudiant de niveau Licence 3. Ce qui nous a permis d'accueillir et d'encadrer Mr. Guillaume Taupiac, élève en première année de l'école centrale de Nantes.

**Analyse statique des propriétés quantitatives (probabilité et délai) d'accessibilité dans les réseaux d'automates stochastiques.**

Nous avons tiré avantage des analyses statiques efficaces, déjà introduites pour l'analyse des propriétés qualitatives d'accessibilité dans les Frappes de Processus et les réseaux d'automates asynchrones, pour proposer une nouvelle analyse statique pour la dérivation des propriétés quantitatives. Cette analyse statique est basée sur les techniques d'interprétation abstraite. Cette analyse statique possède une complexité théorique polynomiale selon le nombre d'automates et exponentielle selon le nombre d'états locaux dans chaque automate. Avec ce résultat, il est désormais possible de donner une estimation de la probabilité

d'observer un comportement donné, d'estimer le délai moyen pour observer un comportement sur les grands RRB. De plus, ces méthodes sont très facilement transposables pour la vérification des systèmes autres que les systèmes biologiques.

**Analyse statique des bifurcations dans les réseaux d'automates asynchrones.**

Toujours partant des analyses statiques développées pour la propriété d'accessibilité et en particulier les approximations supérieures et inférieures de l'accessibilité, nous avons proposé une méthode qui tire aussi avantage de la programmation et de la modélisation logique pour **identifier les transitions de bifurcation** (chapitre 5). Il s'agit des transitions telle que, une fois qu'elles sont franchies, il n'est possible d'atteindre un objectif donné. L'identification de telles transitions ouvre la voie au contrôle des systèmes en ce sens qu'elles peuvent permettre de prévenir ou d'empêcher des comportements indésirables de survenir. Partant de ces bifurcations, il est possible d'envisager de calculer la probabilité de ne pas atteindre un objectif et donc avoir une estimation de la probabilité de l'atteindre. Ce travail a été réalisé avec la collaboration de Loïc Paulevé (CNRS & LRI, Paris, France).

**Applications aux réseaux de régulation biologique.**

Le chapitre 6 montre les applications des différentes contributions de cette thèse à l'étude des RRB.

Nous avons illustré la modélisation des grands RRB par les réseaux d'automates stochastiques avec l'estimation et l'intégration des paramètres temporels et stochastiques. L'estimation des paramètres a été effectuée à partir des données expérimentales obtenues à la suite d'une expérience sur la différenciation des kératinocytes. Ainsi, la démarche méthodologique décrite au chapitre 3 a pu être mise en œuvre sur un modèle de la différenciation des kératinocytes comportant près de 300 composants et 400 interactions.

Les analyses statiques développées pour l'identification des bifurcations ont été appliquées à plusieurs modèles des RRB de tailles variables avec des RRB contenant une centaine de composants. Nous avons pu identifier les bifurcations en quelques secondes même pour les très grands réseaux pour lesquels une vérification exacte avec les model-checkers classiques s'avère impossible. Ceci démontre l'efficacité et l'intérêt de la démarche proposée.

## 7.2 Perspectives

Nous présentons dans cette section les ouvertures potentielles vers lesquelles amènent les résultats présentés dans cette thèse.

**Calculer la probabilité de faire une bifurcation**

L'identification des bifurcations ouvre la voie à une estimation de la probabilité de ne pas atteindre un état donné. En effet étant donné le problème d'accessibilité tel que défini au chapitre 5, si $g$ est le but à atteindre et si $\Pr(g)$ est la probabilité de l'atteindre, à partir des transitions de bifurcation on pourrait proposer une estimation de $1 - \Pr(g)$ qui représenterai donc la probabilité de ne pas atteindre le but $g$.

### Une borne supérieure de la probabilité avec l'analyse statique

Avec l'analyse statique que nous avons proposée pour l'estimation de la probabilité et des délais d'atteindre un état, une des hypothèses est de considérer que les transitions concurrentes ont une influence maximale. Ceci n'est pas vrai dans la pratique. Cette hypothèse de travail impose que la probabilité que nous estimons est une borne inférieure de la probabilité réelle. Cette hypothèse peut être considérée dans l'autre sens, en choisissant de donner une influence minimale aux transitions concurrentes. Un tel choix conduirait à obtenir **une estimation supérieure de la probabilité** que nous recherchons.

### Vérifier les trois approches pour l'évaluation quantitative

Étant donné que nous avons introduit deux approches (analyse statistique des traces et analyse statique des propriétés quantitatives (probabilité)) pour la vérification des propriétés quantitatives, et que des transitions de bifurcation il est possible d'estimer la probabilité de ne pas atteindre un but donné, nous prévoyons de vérifier que les trois approches convergent vers la valeur recherchée. En effet, si $g$ est le but à atteindre :

- l'analyse statistique des traces permet d'obtenir une fréquence $f(g)$ avec laquelle $g$ est atteint sur un ensemble de simulations.
- l'analyse statique des propriétés quantitatives fournie un encadrement de $\Pr(g)$ en fournissant une limite inférieure $\mathsf{linf}_{\Pr(g)}$ et une limite supérieure $\mathsf{lsup}_{\Pr(g)}$.
- à partir des transitions de bifurcations, si $b$ est l'évènement « effectuer une bifurcation », on note $\Pr(b)$ la probabilité d'effectuer une bifurcation. Ainsi, la probabilité d'atteindre le but est donnée par $\Pr(g) = 1 - \Pr(b)$.

L'objectif principal de cette perspective est de vérifier si nous avons la relation

$$\mathsf{linf}_{\Pr(g)} \leq f(g) \simeq \Pr(g) \leq \mathsf{lsup}_{\Pr(g)}$$

### Analyse statique des propriétés quantitatives des modèles paramétriques

L'analyse statique des propriétés quantitatives développée au chapitre 4 est faite pour les modèles où les taux des transitions sont connus et fixés. Elle ne ne prend pas en compte des modèles où les taux des transitions pourraient être des paramètres inconnus mais dont les valeurs appartiennent à un intervalle donné. Des travaux futurs pourraient par exemple étendre l'analyse statique développée pour l'analyse et la vérification des propriétés quantitatives, pour l'analyse des modèles paramétriques des systèmes stochastiques et concurrents. En effet, les modèles paramétrisés sont d'un intérêt évident même comme les analyses associées à ces modèles restent des problèmes difficiles. Par exemple, si nous considérons un réseau d'automates stochastiques paramétriques, déterminer s'il existe une plage de valeur des paramètres du réseau, telle que le système peut atteindre un état donné partant d'un état initial est un problème indécidable. L'idée serait de se servir des méthodes par approximation pour pouvoir donner des outils d'aide à la décision sur les valeurs des paramètres du modèle.

### Règles d'association pour la découverte d'influences non encore connues

Les règles d'association est une méthode en fouille de données dont le but est de découvrir les règles ayant un intérêt entre deux ou plusieurs variables stockées dans de très

importantes bases de données. Dans le cas de la dynamique des réseaux d'automates, une des principales questions est de savoir quels sont les composants qui ont une influence sur d'autres. Les influences directes sont assez évidentes à connaître ; il suffit de regarder le graphe des interactions. Cependant, les influences indirectes ne sont pas évidentes à observer, même en construisant des graphes de dépendances. Nous souhaitons explorer une nouvelle approche qui consiste à profiter de la puissance des règles d'association (au vu des résultats obtenus dans d'autres domaines) pour mettre en évidence les relations non encore connues dans les réseaux de régulation biologique. La démarche consisterait à considérer l'ensemble des états locaux du modèle comme un ensemble d'items. Puis, à partir des simulations, construire un ensemble de faits couramment appelés transactions dans le domaine des règles d'association. Enfin, il conviendrait de construire les règles d'association à partir des items et des transactions. L'idée de cette démarche est de mettre en évidence les relations insoupçonnées entre les états des composants du système.

# Bibliographie


Abou-Jaoudé, W., Monteiro, P. T., Naldi, A., Grandclaudon, M., Soumelis, V., Chaouiya, C. & Thieffry, D. (2015), 'Model checking to assess t-helper cell plasticity', *Frontiers in Bioengineering and Biotechnology* **2**(86).

Acuna, V., Chierichetti, F., Lacroix, V., Marchetti-Spaccamela, A., Sagot, M.-F. & Stougie, L. (2009), 'Modes and cuts in metabolic networks : Complexity and algorithms', *Biosystems* **95**(1), 51–60.

Ahmad, J., Roux, O., Bernot, G., Comet, J.-P. & Richard, A. (2008), 'Analysing formal models of genetic regulatory networks with delays', *International Journal of Bioinformatics Research and Applications (IJBRA)* **4**(2).

Alur, R., Belta, C., Kumar, V., Mintz, M., Pappas, G. J., Rubin, H. & Schug, J. (2002), 'Modeling and analyzing biomolecular networks', *Computing in Science & Engineering* **4**(1), 20–31.

Alur, R., Courcoubetis, C. & Dill, D. (1990), Model-checking for real-time systems, *in* 'Logic in Computer Science, 1990. LICS '90, Proceedings., Fifth Annual IEEE Symposium on e', pp. 414–425.

Aziz, A., Sanwal, K., Singhal, V. & Brayton, R. (2000), 'Model-checking continuous-time markov chains', *ACM Transactions on Computational Logic (TOCL)* **1**(1), 162–170.

Baier, C., Bertrand, N., Bouyer, P., Brihaye, T. & GröBer, M. (2008), Almost-sure model checking of infinite paths in one-clock timed automata, *in* 'Logic in Computer Science, 2008. LICS'08. 23rd Annual IEEE Symposium on', IEEE, pp. 217–226.

Baier, C., Bertrand, N., Bouyer, P., Brihaye, T. & Größer, M. (2007), Probabilistic and topological semantics for timed automata, *in* 'International Conference on Foundations of Software Technology and Theoretical Computer Science', Springer, pp. 179–191.

Baral, C. (2003), *Knowledge Representation, Reasoning and Declarative Problem Solving*, Cambridge University Press.

Ben Abdallah, E., Ribeiro, T., Magnin, M., Roux, O. & Katsumi, I. (2016), Inference of delayed biological regulatory networks from time series data, *in* 'CMSB 2016 - 14th conference on Computational Methods for Systems Biology', Vol. 9859 of *Lecture Notes in Bioinformatics*, Springer.
`https://www.cl.cam.ac.uk/events/cmsb2016/`

Bernot, G., Comet, J.-P. & Khalis, Z. (2008), Gene regulatory networks with multiplexes, *in* 'European Simulation and Modelling Conference Proceedings', pp. 423–432.

Bertrand, N., Bouyer, P., Brihaye, T. & Markey, N. (2008), Quantitative model-checking of one-clock timed automata under probabilistic semantics, *in* 'Quantitative Evaluation of Systems, 2008. QEST'08. Fifth International Conference on', IEEE, pp. 55–64.

Bouissou, O. (2008), Analyse statique par interprétation abstraite de systèmes hybrides., Theses, Ecole Polytechnique X.
`https://pastel.archives-ouvertes.fr/pastel-00004412`

Cerami, E. G., Gross, B. E., Demir, E., Rodchenkov, I., Babur, Ö., Anwar, N., Schultz, N., Bader, G. D. & Sander, C. (2011), 'Pathway commons, a web resource for biological pathway data', *Nucleic acids research* **39**(suppl 1), D685–D690.

Chaouiya, C., Bérenguier, D., Keating, S. M., Naldi, A., van Iersel, M. P., Rodriguez, N., Dräger, A., Büchel, F., Cokelaer, T., Kowal, B., Wicks, B., Gonçalves, E., Dorier, J., Page, M., Monteiro, P. T., von Kamp, A., Xenarios, I., de Jong, H., Hucka, M., Klamt, S., Thieffry, D., Le Novère, N., Saez-Rodriguez, J. & Helikar, T. (2013), 'SBML qualitative models : a model representation format and infrastructure to foster interactions between qualitative modelling formalisms and tools', *BMC Systems Biology* **7**(1), 1–15.

Cheng, A., Esparza, J. & Palsberg, J. (1995), 'Complexity results for 1-safe nets', *Theor. Comput. Sci.* **147**(1&2), 117–136.

Courcoubetis, C. & Yannakakis, M. (1988), Verifying temporal properties of finite-state probabilistic programs, *in* 'Foundations of Computer Science, 1988., 29th Annual Symposium on', IEEE, pp. 338–345.

Cousot, P. & Cousot, R. (1977), Abstract interpretation : A unified lattice model for static analysis of programs by construction or approximation of fixpoints, *in* 'Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL'77)', ACM, New York, NY, USA, pp. 238–252.

Danos, V., Feret, J., Fontana, W. & Krivine, J. (2007), Scalable simulation of cellular signaling networks, *in* 'Asian Symposium on Programming Languages and Systems', Springer Berlin Heidelberg, pp. 139–157.

Danos, V., Feret, J., Fontana, W. & Krivine, J. (2008), Abstract interpretation of cellular signalling networks, *in* 'Verification, Model Checking, and Abstract Interpretation', Springer, pp. 83–97.

Emerson, E. A. (1990), 'Temporal and modal logic.', *Handbook of Theoretical Computer Science, Volume B : Formal Models and Sematics (B)* **995**(1072), 5.

Esparza, J. & Heljanko, K. (2008), *Unfoldings – A Partial-Order Approach to Model Checking*, Springer.

Esparza, J. & Schröter, C. (2001), 'Unfolding based algorithms for the reachability problem', *Fund. Inf.* **47**(3-4), 231–245.

Evans, M., Hastings, N. & Peacock, B. (2000), 'Statistical distributions'.

Fages, F., Soliman, S. & Chabrier-Rivier, N. (2004), 'Modelling and querying interaction networks in the biochemical abstract machine biocham', *Journal of Biological Physics and Chemistry* **4**, 64–73.

Feret, J. (2005), 'Analysis of mobile systems by abstract interpretation', *These de PhD, École Polytechnique* .

Feret, J., Henzinger, T., Koeppl, H. & Petrov, T. (2012), 'Lumpability abstractions of rule-based systems', *Theoretical Computer Science* **431**, 137–164.

Feret, J., Koeppl, H. & Petrov, T. (n.d.), 'Stochastic fragments : A framework for the exact reduction of the stochastic semantics of rule-based models', *International Journal of Software and Informatics* .

Fitime, L. F., Roux, O., Guziolowski, C. & Paulevé, L. (2016), Identification of bifurcations in biological regulatory networks using answer-set programming, *in* 'Constraint-Based Methods for Bioinformatics - 12th International Workshop, WCB 2016'.

Fitime, L. F., Schuster, C., Angel, P., Roux, O. & Guziolowski, C. (2015), Integrating time-series data in large-scale discrete cell-based models, *in* 'Hybrid Systems Biology - Fourth International Workshop, HSB 2015, Madrid, Spain, September 4-5, 2015. Revised Selected Papers', pp. 75–95.
`http://dx.doi.org/10.1007/978-3-319-26916-0_5`

Folschette, M. (2014), Algebraic Modeling of the Dynamics of Multi-scale Biological Regulatory Networks, Theses, École Centrale de Nantes.
`https://tel.archives-ouvertes.fr/tel-01105203`

Folschette, M., Paulevé, L., Magnin, M. & Roux, O. (2015), 'Sufficient conditions for reachability in automata networks with priorities', *Theoretical Computer Science* **608, Part 1**, 66 – 83. From Computer Science to Biology and Back.

François, P., Hakim, V. & Siggia, E. D. (2007), 'Deriving structure from evolution : metazoan segmentation', *Molecular Systems Biology* **3**(1).

Gao, Y., Xu, M., Zhan, N. & Zhang, L. (2013), 'Model checking conditional csl for continuous-time markov chains', *Information Processing Letters* **113**(1), 44–50.

Gardner, T. S., Di Bernardo, D., Lorenz, D. & Collins, J. J. (2003), 'Inferring genetic networks and identifying compound mode of action via expression profiling', *Science* **301**(5629), 102–105.

Gay, S., Soliman, S. & Fages, F. (2010), 'A graphical method for reducing and relating models in systems biology', *Bioinformatics* **26**(18), i575–i581.

Gebser, M., Kaminski, R., Kaufmann, B. & Schaub, T. (2012), *Answer Set Solving in Practice*, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan and Claypool Publishers.

Gebser, M. & Schaub, T. (2013), 'Answer set solving in practice'. Tutorial at IJCAI'13.

Guziolowski, C., Bourdé, A., Moreews, F. & Siegel, A. (2009), 'Bioquali cytoscape plugin : analysing the global consistency of regulatory networks', *BMC Genomics* **10**(1), 1–11. http://dx.doi.org/10.1186/1471-2164-10-244

Guziolowski, C., Kittas, A., Dittmann, F. & Grabe, N. (2012), 'Automatic generation of causal networks linking growth factor stimuli to functional cell state changes', *FEBS Journal* **279**(18), 3462–3474.

Guziolowski, C., Videla, S., Eduati, F., Thiele, S., Cokelaer, T., Siegel, A. & Saez-Rodriguez, J. (2013), 'Exhaustively characterizing feasible logic models of a signaling network using answer set programming', *Bioinformatics* .

Hachtel, G. D., Macii, E., Pardo, A. & Somenzi, F. (1996), 'Markovian analysis of large finite state machines', *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **15**(12), 1479–1493.

Hamez, A., Thierry-Mieg, Y. & Kordon, F. (2009), 'Building efficient model checkers using hierarchical set decision diagrams and automatic saturation', *Fundamenta Informaticae* **94**(3-4), 413–437.

Hanahan, D. & Weinberg, R. A. (2000), 'The hallmarks of cancer', *Cell* **100**(1), 57 – 70. http://www.sciencedirect.com/science/article/pii/S0092867400816839

Hansson, H. & Jonsson, B. (1994), 'A logic for reasoning about time and reliability', *Formal aspects of computing* **6**(5), 512–535.

Hecker, M., Lambeck, S., Toepfer, S., Van Someren, E. & Guthke, R. (2009), 'Gene regulatory network inference : data integration in dynamic models—a review', *Biosystems* **96**(1), 86–103.

Heckerman, D. (1998), A tutorial on learning with bayesian networks, *in* 'Learning in graphical models', Springer, pp. 301–354.

Heiner, M., Gilbert, D. & Donaldson, R. (2008), Petri nets for systems and synthetic biology, *in* 'International School on Formal Methods for the Design of Computer, Communication and Software Systems', Springer Berlin Heidelberg, pp. 215–264.

Heiner, M., Rohr, C., Schwarick, M. & Streif, S. (2010), A comparative study of stochastic analysis techniques, *in* 'Proceedings of the 8th International Conference on Computational Methods in Systems Biology', ACM, pp. 96–106.

Heljanko, K. (1999), 'Using logic programs with stable model semantics to solve deadlock and reachability problems for 1-Safe petri nets', *Fundamenta Informaticae* **37**(3), 247–268.

Ideker, T. & Sharan, R. (2008), 'Protein networks in disease', *Genome research* **18**(4), 644–652.

Joshi-Tope, G., Gillespie, M., Vastrik, I., D'Eustachio, P., Schmidt, E., de Bono, B., Jassal, B., Gopinath, G., Wu, G., Matthews, L. et al. (2005), 'Reactome : a knowledgebase of biological pathways', *Nucleic acids research* **33**(suppl 1), D428–D432.

Kanehisa, M., Sato, Y., Kawashima, M., Furumichi, M. & Tanabe, M. (2015), 'Kegg as a reference resource for gene and protein annotation', *Nucleic acids research* p. gkv1070.

Kauffman, S. A. (1969), 'Metabolic stability and epigenesis in randomly constructed genetic nets', *Journal of Theoretical Biology* **22**(3), 437–467.

Kemeny, J. G., Snell, J. L. et al. (1960), *Finite markov chains*, Vol. 356, van Nostrand Princeton, NJ.

Klamt, S. & Gilles, E. D. (2004), 'Minimal cut sets in biochemical reaction networks', *Bioinformatics* **20**(2), 226–234.

Klamt, S., Saez-Rodriguez, J. & Gilles, E. D. (2007), 'Structural and functional analysis of cellular networks with cellnetanalyzer', *BMC Systems Biology* **1**(1).

Kwiatkowska, M., Norman, G. & Parker, D. (2011), PRISM 4.0 : Verification of probabilistic real-time systems, *in* G. Gopalakrishnan & S. Qadeer, eds, 'Proc. 23rd International Conference on Computer Aided Verification (CAV'11)', Vol. 6806 of *LNCS*, Springer, pp. 585–591.

Liang, S., Fuhrman, S. & Somogyi, R. (1998), 'Reveal, a general reverse engineering algorithm for inference of genetic network architectures'.

Ma'ayan, A., Jenkins, S. L., Neves, S., Hasseldine, A., Grace, E., Dubin-Thaler, B., Eungdamrong, N. J., Weng, G., Ram, P. T., Rice, J. J. et al. (2005), 'Formation of regulatory patterns during signal propagation in a mammalian cellular network', *Science* **309**(5737), 1078–1083.

MacNamara, A., Terfve, C., Henriques, D., Bernabé, B. P. & Saez-Rodriguez, J. (2012), 'State–time spectrum of signal transduction logic models', *Physical Biology* **9**(4), 045003.

Maurin, M., Magnin, M. & Roux, O. (2009), Modeling of genetic regulatory network in stochastic $\pi$-calculus, *in* 'Bioinformatics and Computational Biology', Springer Berlin Heidelberg, pp. 282–294.

Moon, I. (1992), 'Automatic verification of discrete chemical process control systems'.

Moon, I. & Macchietto, S. (1994), Formal verification of batch processing control procedures, *in* 'Proc. PSE'94', p. 469.

Moon, I., Powers, G. J., Burch, J. R. & Clarke, E. M. (1992), 'Automatic verification of sequential control systems using temporal logic', *AIChE Journal* **38**(1), 67–75.

Naldi, A., Remy, E., Thieffry, D. & Chaouiya, C. (2009), 'A reduction of logical regulatory graphs preserving essential dynamical properties', *Computational Methods in Systems Biology* pp. 266–280.

Naldi, A., Thieffry, D. & Chaouiya, C. (2007), Decision diagrams for the representation and analysis of logical models of genetic networks, *in* 'International Conference on Computational Methods in Systems Biology', Springer, pp. 233–247.

Needham, C. J., Bradford, J. R., Bulpitt, A. J. & Westhead, D. R. (2007), 'A primer on learning in bayesian networks for computational biology', *PLoS Comput Biol* **3**(8), e129.

Ostrowski, M., Paulevé, L., Schaub, T., Siegel, A. & Guziolowski, C. (2015), Boolean network identification from multiplex time series data, *in* 'International Conference on Computational Methods in Systems Biology', Springer, pp. 170–181.

Paulevé, L. (2011), Modélisation, Simulation et Vérification des Grands Réseaux de Régulation Biologique, PhD thesis, École Centrale de Nantes.

Paulevé, L. (2016), Goal-oriented reduction of automata networks, *in* 'CMSB 2016 - 14th conference on Computational Methods for Systems Biology'. accepted. `https://www.cl.cam.ac.uk/events/cmsb2016/`

Paulevé, L., Andrieux, G. & Koeppl, H. (2013), Under-approximating cut sets for reachability in large scale automata networks, *in* N. Sharygina & H. Veith, eds, 'Computer Aided Verification', Vol. 8044 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 69–84.

Paulevé, L., Folschette, M., Magnin, M. & Roux, O. (2015), 'Analyses statiques de la dynamique des réseaux d'automates indéterministes', *Technique et Science Informatiques (TSI)* **34**(4), 463–484.

Paulevé, L., Magnin, M. & Roux, O. (2011*a*), Refining dynamics of gene regulatory networks in a stochastic $\pi$-calculus framework, *in* 'Transactions on Computational Systems Biology XIII', Springer, pp. 171–191.

Paulevé, L., Magnin, M. & Roux, O. (2011*b*), 'Tuning temporal features within the stochastic $\pi$-calculus', *IEEE Transactions on Software Engineering* **37**(6), 858–871.

Paulevé, L., Magnin, M. & Roux, O. (2012), 'Static analysis of biological regulatory networks dynamics using abstract interpretation', *Mathematical Structures in Computer Science* **22**(04), 651–685.

Pico, A. R., Kelder, T., Van Iersel, M. P., Hanspers, K., Conklin, B. R. & Evelo, C. (2008), 'Wikipathways : pathway editing for the people', *PLoS Biol* **6**(7), e184.

Pinna, A., Soranzo, N. & de la Fuente, A. (2010), 'From knockouts to networks : Establishing direct cause-effect relationships through graph analysis', *PLoS ONE* **5**(10), e12912.

Plateau, B. & Atif, K. (1991), 'Stochastic automata network of modeling parallel systems', *IEEE transactions on software engineering* **17**(10), 1093–1108.

Popova-Zeugmann, L., Heiner, M. & Koch, I. (2005), 'Time petri nets for modelling and analysis of biochemical networks', *Fundamenta Informaticae* **67**(1-3), 149–162.

Priami, C. (1995), 'Stochastic $\pi$-calculus', *The Computer Journal* **38**(7), 578–589.

Radulescu, O., Gorban, A. N., Zinovyev, A. & Lilienbaum, A. (2008), 'Robust simplifications of multiscale biochemical networks', *BMC systems biology* **2**(1), 86.

Richard, A. (2010), 'Negative circuits and sustained oscillations in asynchronous automata networks', *Advances in Applied Mathematics* **44**(4), 378–392.

Rizk, A., Batt, G., Fages, F. & Soliman, S. (2008), On a continuous degree of satisfaction of temporal logic formulae with applications to systems biology, *in* 'International Conference on Computational Methods in Systems Biology', Springer Berlin Heidelberg, pp. 251–268.

Saez-Rodriguez, J., Alexopoulos, L. G., Epperlein, J., Samaga, R., Lauffenburger, D. A., Klamt, S. & Sorger, P. K. (2009), 'Discrete logic modelling as a means to link protein signalling networks with functional analysis of mammalian signal transduction', *Molecular systems biology* **5**(1), 331.

Samaga, R., Kamp, A. V. & Klamt, S. (2010), 'Computing combinatorial intervention strategies and failure modes in signaling networks', *Journal of Computational Biology* **17**(1), 39–53.

Schaefer, C. F., Anthony, K., Krupa, S., Buchoff, J., Day, M., Hannay, T. & Buetow, K. H. (2009*a*), 'PID : the Pathway Interaction Database', *Nucleic Acids Research* **37**(suppl 1), D674–D679.

Schaefer, C. F., Anthony, K., Krupa, S., Buchoff, J., Day, M., Hannay, T. & Buetow, K. H. (2009*b*), 'Pid : the pathway interaction database', *Nucleic acids research* **37**(suppl 1), D674–D679.

Siebert, H. & Bockmayr, A. (2006), Incorporating time delays into the logical analysis of gene regulatory networks, *in* 'International Conference on Computational Methods in Systems Biology', Springer, pp. 169–183.

Steuer, R., Kurths, J., Daub, C. O., Weise, J. & Selbig, J. (2002), 'The mutual information : detecting and evaluating dependencies between variables', *Bioinformatics* **18**(suppl 2), S231–S240.

Stewart, W. J. (1994), *Introduction to the numerical solutions of Markov chains*, Princeton Univ. Press.

Stuart, J. M., Segal, E., Koller, D. & Kim, S. K. (2003), 'A gene-coexpression network for global discovery of conserved genetic modules', *Science* **302**(5643), 249–255. `http://science.sciencemag.org/content/302/5643/249`

Terfve, C. & Saez-Rodriguez, J. (2012), Modeling signaling networks using high-throughput phospho-proteomics, *in* 'Advances in Systems Biology', Springer, pp. 19–57.

Thieffry, D. & Thomas, R. (1995), 'Dynamical behaviour of biological regulatory networks–ii. immunity control in bacteriophage lambda.', *Bulletin of mathematical biology* **57**, 277–97.

Thomas, R. (1973), 'Boolean formalization of genetic control circuits', *Journal of Theoretical Biology* **42**(3), 563 – 585.

Tyson, J. J. & Othmer, H. G. (1978), 'The dynamics of feedback control circuits in biochemical pathways', *Progress in Theorrtical Biology* **5**.

Videla, S., Guziolowski, C., Eduati, F., Thiele, S., Grabe, N., Saez-Rodriguez, J. & Siegel, A. (2012), Revisiting the training of logic models of protein signaling networks with asp, *in* 'Computational Methods in Systems Biology', Springer, pp. 342–361.

Zhang, L., Jansen, D. N., Nielson, F. & Hermanns, H. (2011), Automata-based csl model checking, *in* 'Automata, Languages and Programming', Springer, pp. 271–282.

UNIVERSITE BRETAGNE LOIRE

Centrale Nantes

# Thèse de Doctorat

Louis FIPPO FITIME

## Modélisation Hybride, Analyse et Vérification Quantitative des Grands Réseaux de Régulation Biologique

## Hybride Modelling, Analysis and Quantitative Verification of Large Biological Regulatory Networks


### Résumé

Les Réseaux de Régulation biologique (RRB) sont couramment utilisés en biologie des systèmes pour modéliser, comprendre et contrôler les dynamiques des différentes fonctions biologiques (différenciation, synthèse protéique, apoptose) au sein des cellules. Ces réseaux sont enrichis des données expérimentales de plus en plus nombreuses et disponibles qui renseignent sur les dynamiques des composants des RRB. L'analyse formelle de tels modèles se heurte rapidement à l'explosion combinatoire des comportements engendrés malgré le fait que les RRB offrent une représentation abstraite des systèmes biologiques.

Cette thèse traite de la modélisation hybride, de la simulation, de la vérification formelle et du contrôle des grands Réseaux de Régulation Biologique. Cette modélisation est effectuée grâce aux réseaux d'automates stochastiques, puis aux Frappes de Processus qui en sont une restriction, en intégrant des données de séries temporelles.

En premier lieu, cette thèse propose un raffinement des dynamiques par l'estimation des paramètres stochastiques et temporels (délais) à partir des données de séries temporelles et l'intégration de ces paramètres dans les modèles en réseaux d'automates. Cette intégration permet de paramétrer les transitions entre les états du système. Puis, une analyse statistique des traces des simulations stochastiques est proposée afin de comparer les dynamiques des simulations à celles des données expérimentales.

En deuxième lieu, cette thèse développe une analyse statique par interprétation abstraite dans les réseaux d'automates permettant des approximations inférieures et supérieures très efficaces des propriétés d'accessibilité d'un point de vue quantitatif (probabilité et délai). Cette analyse permet de faire apparaître des composants critiques pour la satisfaction de ces propriétés.

Enfin, tirant avantage des analyses statiques développées pour l'accessibilité dans les réseaux d'automates, et de la puissance de la programmation logique (ASP), cette thèse aborde le domaine du contrôle des systèmes en proposant l'identification des *transitions de bifurcation*. Les bifurcations sont des transitions après lesquelles le système ne peut plus atteindre un état précédemment accessible.



### Mots clés

systèmes complexes ; réseaux de régulation biologique ; modélisation hybride ; intégration des données ; simulation stochastique ; analyse statique ; vérification formelle ; interprétation abstraite ; bifurcation ; contrôle.



### Abstract

Biological Regulatory Networks (BRNs) are usually used in systems biology for modelling, understanding and controlling the dynamics of different biological functions (differentiation, proliferation, proteins synthesis, apoptose) inside cells. Those networks are enhanced with experimental data that are nowadays more available which give an idea on the dynamics of BRNs components. Formal analysis of such models fails in front of the combinatorial explosion of generated behaviours despite the fact that BRNs provide abstract representation of biological systems.

This thesis handles hybrid modelling, the simulation, the formal verification and control of Large Biological Regulatory Networks. This modelling is done thanks to stochastic automata networks, thereafter to Process Hitting by integrating time-series data.

Firstly, this thesis proposes a refining of the dynamics by estimation of stochastic and temporal (delay) parameters from time-series data and integration of those parameters in automata networks models. This integration allows the parametrisation of the transitions between the states of the system. Then, a statistical analysis of the traces of the stochastic simulation is proposed to compare the dynamics of simulations with the experimental data.

Secondly, this thesis develops static analysis by abstract interpretation in the automata networks allowing efficient under- and over-approximation of quantitative (probability and delay) reachability properties. This analysis enables to highlight the critical components to satisfy these properties.

Finally, taking advantage from the previous developed static analyses for the reachability properties in the qualitative point of view, and from the power of logic programming (Answer Set Programming), this thesis addresses the domain of control of system by proposing the identification of *bifurcation transitions*. Bifurcations are transitions after which the system can no longer reach a state that was previously reachable.



### Key Words

complex systems; biological regulatory networks; hybrid modelling; data integration; stochastic simulation; static analysis; formal verification; abstract interpretation; bifurcation; control.