Réseaux booléens : méthodes formelles et outils pour la modélisation en biologie

Loïc Paulevé

Thèse pour l'Habilitation à diriger des recherches

Réseaux booléens : méthodes formelles et outils pour la modélisation en biologie

Loïc Paulevé

2020

Soutenue le 3 septembre 2020 devant le jury composé de

— Hanna Klaudel, Université Paris-Saclay (présidente)
— François Fages, Inria Saclay (rapporteur)
— Joachim Niehren, Inria Lille (rapporteur)
— Sylvain Sené, Université Aix-Marseille (rapporteur)
— Laurence Calzone, Institut Curie
— Franck Delaplace, Université Paris-Saclay

Délivrée par l'Université Paris-Saclay

Table des matières

Avant-propos

La modélisation mathématique et informatique aide à comprendre et *in fine* à contrôler le comportement du vivant pour des applications thérapeutiques. Par exemple au niveau de la cellule, en modélisant les interactions connues entre des gènes et protéines, il est possible d'identifier, par simulations ou par raisonnement logique, des cibles potentielles pour contrôler, positivement ou négativement, l'occurrence de phénotypes d'intérêt (Collombet et al. 2017, Le Novère 2015, Saez-Rodriguez et al. 2009, Zañudo, Steinway, & Albert 2018). Ce type de modèle est souvent qualifié de "mécanique", car il cherche à reproduire les interactions physico-chimiques entre des entités clairement identifiées, toutefois toujours avec un certain degré de simplification.

Les réseaux booléens (Kauffman 1969, Thomas 1973) sont de plus en plus employés pour modéliser les réseaux de régulation génétique et les voies de signalisation en tenant compte d'un grand nombre d'entités (p.ex. Abou-Jaoudé et al. 2015, Cohen et al. 2015, Zañudo et al. 2018). L'activité (ou état) des entités sélectionnées (gènes, protéines, ARN, *etc.*) est simplifiée de manière binaire : inactif/actif, absent/présent, 0/1. L'évolution de l'activité de chaque entité est alors décrite par des règles logiques qui dépendent de l'état de ses régulateurs. Ceci reflète en partie le type de connaissance généralement disponible pour la modélisation des systèmes biologiques, centré sur l'existence et la nature des interactions entre gènes et protéines mais avec peu de détails sur leurs cinétiques. Ainsi, les modèles booléens se veulent être une simplification à gros grains du système étudié, mais robuste car ils ne dépendent que de peu de paramètres, et en particulier n'ont aucun paramètre à domaine continu.

La modélisation des réseaux biologiques est un art laborieux, demandant beaucoup de partis pris et d'hypothèses plus ou moins implicites. Contrairement à la biologie structurale, et à de nombreux domaines en physique et en chimie, la biologie des systèmes souffre d'un manque de lois générales régissant la structure et la dynamique de ces systèmes complexes. Les modèles sont construits le plus souvent manuellement, et regorgent de paramètres et d'incertitude ayant un effet considérable sur les prédictions. L'approche booléenne esquissée dans le paragraphe précédent permet déjà d'abstraire une grande partie de ces paramètres et de raisonner de manière globale sur le modèle.

Les applications des réseaux booléens en biologie emploient le plus souvent des méthodes basées sur la simulation (échantillon des comportements possibles du modèle) ou sur la vérification exhaustive des comportements (*model checking*). Il est toutefois très souvent fait état de limites importantes pour leur passage à l'échelle (Bornholdt 2008, Le Novère 2015).

D'apparence simpliste, la mise en œuvre des modélisations booléennes touche à de très

nombreux domaines des sciences fondamentales, des mathématiques discrètes, de la combinatoire, de la logique, de la complexité algorithmique, des langages formels, avec des notions de sémantique, de concurrence, d'abstraction, etc.

On observe cependant régulièrement un schisme entre les chercheurs et modélisateurs en biologie et les chercheurs en informatique. D'un côté, les théoriciens ont tendance à considérer que les problèmes rencontrés par les modélisateurs sont largement résolus ou peu intéressants et très mal formalisés. D'un autre côté, les modélisateurs réduisent très souvent l'informatique à un simple outil, où les considérations théoriques n'ont pas d'intérêt majeur. Bien entendu et fort heureusement, des discussions plus approfondies peuvent dissiper ces préjugés et aboutir à des collaborations fructueuses, d'un côté comme de l'autre.

L'emploi des réseaux booléens en biologie soulève de nombreux défis : à la fois du côté théorique, par exemple sur des problématiques de sémantiques, de complexité, de contrôle, de synthèse de modèle, et du côté modélisation, par exemple sur des problématiques de formulation informatique des propriétés recherchées, d'intégration de données expérimentales, d'apprentissage, et d'outils.

Cette thèse pour l'habilitation à diriger des recherches présente un aperçu de mes différentes contributions à ces défis. Ces contributions sont issues de collaborations à la fois avec des informaticiens théoriciens et avec des biologistes, modélisateurs et expérimentalistes.

Parcours académique

D'une formation purement informatique, j'ai effectué une thèse de doctorat sur la modélisation et l'analyse des grands réseaux de régulation génétique, dans le laboratoire d'informatique et automatique (LS2N, ex-IRCCyN) à l'école centrale de Nantes. Ma motivation première était de contribuer à montrer que l'informatique, en tant que science, peut être utile à des domaines a priori éloignés. Après un post-doctorat au laboratoire d'informatique (LIX) de l'école polytechnique et au laboratoire d'automatique (IfA) de l'école polytechnique fédérale de Zürich (ETHZ), j'ai été recruté par le CNRS en octobre 2013 en tant que chargé de recherche, affecté au laboratoire de recherche en informatique (LRI) de l'université Paris-Sud (dorénavant Paris-Saclay) dans l'équipe "Bio-info". En 2018, j'ai effectué une mobilité pour le laboratoire bordelais de recherche en informatique (LaBRI).

J'ai participé à l'encadrement de trois thèses de doctorat : Hugues Mandon (soutenue, 2016-2019), Juraj Kolčák (débutée au printemps 2017), et Stéphanie Chevalier (débutée à l'automne 2018). Je suis coordinateur du projet ANR PRCI (Luxembourg) AlgoReCell (2017-2021) ; et fut l'investigateur principal de projets CNRS PEPS INS2I (2017) et DIM RFSI (2018). Entre 2016 et 2018, j'ai initié et coordonné avec Stefan Haar (Inria Saclay) le groupe de travail Theo-BioR sur les méthodes informatiques pour la modélisation des réseaux biologiques impliquant différents laboratoires du plateau de Saclay. Je suis membre régulier du comité de programme de la conférence internationale CMSB ; et j'ai présidé le comité de programme du workshop international SASB (static analysis and systems biology) en 2014 et 2015. En 2017, j'ai été co-éditeur invité de la revue Theoretical Computer Science, avec David Safranek et Jérôme Feret pour une édition spéciale sur les méthodes formelles en biologie des systèmes.

Mes publications sont listées à https://loicpauleve.name/publications.shtml.

Cheminement personnel et choix de rédaction

Durant ma thèse de doctorat (Paulevé 2011), j'ai travaillé sur la vérification de propriétés dynamiques au sein de modèles formels discrets, les réseaux d'automates (similaires aux réseaux de Petri), en vue d'être appliquée à des réseaux biologiques de grande taille. J'ai développé des méthodes d'interprétation abstraite des réseaux d'automates, menant à des représentations compactes, mais approchées, des comportements possibles : les *graphes de causalité locale*. Ces graphes résument les dépendances entre les transitions et les états des automates. À partir de ces représentations, j'ai pu définir des conditions nécessaires et des conditions suffisantes à des propriétés d'accessibilité (existence de trajectoire entre deux points donnés) dans les réseaux d'automates purement asynchrones qui peuvent se calculer efficacement (Paulevé, Magnin, & Roux 2012).

Durant mes post-docs et premières années de recherches au LRI, j'ai pu étendre ces résultats pour le calcul de différentes propriétés :

- les *cut sets* pour l'accessibilité (Paulevé, Andrieux, & Koeppl 2013), qui identifient des combinaisons d'états d'automates qui sont impliqués dans toutes les trajectoires entre une configuration initiale et une configuration finale du réseau ;
- les transitions de bifurcations (Fitime, Roux, Guziolowski, & Paulevé 2017), qui identifient les configurations et transitions qui font perdre la capacité d'atteindre un attracteur donné ;
- la réduction de modèle préservant les trajectoires minimales (au sens de l'inclusion des transitions utilisées) (Paulevé 2018) ; réduction qui peut être effectuée à la volée et qui peut être couplée à d'autres méthodes d'exploration, notamment les dépliages de réseaux de Petri (Chatain & Paulevé 2017).

Cette dernière contribution sur la réduction de modèle permet d'effectuer des vérifications exactes de propriétés d'accessibilité au sein de très grands réseaux avec la sémantique purement asynchrone.

Tous ces résultats reposent sur des conditions nécessaires pour l'accessibilité qui se calculent à partir du graphe de causalité locale du réseau d'automate et donnent une approximation formelle supérieure des comportements possibles du réseau : si le comportement recherché n'existe pas dans cette sur-approximation, il n'existe pas dans le réseau concret. Il est à noter que les conditions nécessaires sur les graphes de causalité sont insensibles au synchronisme des transitions.

Depuis 2014, en collaboration avec Stefan Haar et Thomas Chatain (LSV/Inria Saclay), nous avons creusé les liens formels entre les réseaux booléens et les réseaux de Petri (Chatain, Haar, Jezequel, Paulevé, & Schwoon 2014, Chatain, Haar, Kolčák, Paulevé, & Thakkar 2020), que je résume dans la section 1.2. La notion de transition au sein des réseaux de Petri apporte une granularité de spécification qui permet de raisonner sur les entrelacements possibles des changements d'états des composants du réseau, la concurrence. En cherchant à transférer aux réseaux booléens des sémantiques de réseaux de Petri exploitant cette notion de concurrence, en particulier la sémantique par intervalle (Chatain, Haar, Koutny, & Schwoon 2015), nous avons pu commencer à définir de nouvelles façon d'exécuter des réseaux booléens qui produisent des trajectoires que les sémantiques classiques ne pouvait reproduire (Chatain, Haar,

& Paulevé 2018).

Après plusieurs itérations, j'ai pu définir et démontrer une sémantique des réseaux booléens qui garantit d'inclure tous les comportements réalisables par tout raffinement multivalué/discret avec mise à jour asynchrone (général), ou continu (système d'EDO). De plus, la complexité de la décision de propriété d'accessibilité et d'attracteurs est considérablement réduite, ouvrant la voie à l'analyse et à la synthèse de réseaux à l'échelle du génome. Juraj Kolčák, doctorant que je co-encadre, a pu démontrer que cette sémantique est également minimale : tout comportement qu'elle prédit peut être reproduit par au moins un raffinement du modèle booléen. Cette sémantique, qualifiée de *most permissive* (MP), offre ainsi des propriétés robustes vis-à-vis des systèmes non-booléens, comme les systèmes biologiques, et à très faible coût.

Ce résultat, que je détaille dans le chapitre 2, généralise largement l'approche que j'avais initiée sur les réseaux d'automates, pour les applications aux réseaux booléens. Bien sûr, l'histoire n'est pas terminée, puisqu'il serait intéressant de transférer cette sémantique MP aux réseaux d'automates ou d'autres formalismes connexes, comme les réseaux de réactions.

Pour toutes ces raisons, j'ai choisi de centrer l'écriture de ce manuscrit autour des réseaux booléens et de la sémantique MP, puis de résumer mes différents autres axes de recherche, liés aux ensembles de modèles, à leur synthèse, et au contrôle des réseaux booléens, et de leur application pour la modélisation de la reprogrammation cellulaire.

Organisation du manuscrit

Dans le chapitre 1, je présente les réseaux booléens et différentes sémantiques classiques, puis je détaille leurs liens avec les réseaux de Petri, ainsi que la complexité pour la décision de propriétés élémentaires de leur dynamique (accessibilité et attracteurs).

Le chapitre 2 présente en détail la sémantique MP des réseaux booléens, avec les résultats théoriques d'abstraction et de complexité, et des applications pratiques pour l'étude de réseaux biologiques.

Le chapitre 3 donne un résumé de travaux considérant des ensembles de modèles (booléens et multivalués), avec des problématiques de représentation et de sémantique, qui sont l'objet de la thèse de doctorat de Juraj Kolčák, et avec des problématiques de synthèse, qui font l'objet de la thèse de doctorat de Stéphanie Chevalier.

Le chapitre 4 donne un résumé de travaux autour de la modélisation de la reprogrammation cellulaire, avec la recherche de stratégies de contrôle des réseaux booléens, qui a fait l'objet de la thèse de doctorat de Hugues Mandon, et l'application à la trans-différenciation des adipocytes en ostéoblastes, objet du projet franco-luxembourgeois AlgoReCell que je pilote depuis 2017.

Dans le chapitre 5, je traite de la problématique de l'accessibilité et de la reproductibilité des analyses informatiques de modèles, avec l'effort collaboratif autour de la distribution du *CoLoMoTo Interactive Notebook*.

Enfin, le chapitre 6 présente quelques perspectives de recherche.

Chapitre 1

Réseaux booléens : sémantiques (a)synchrones et complexité

Ce chapitre introduit les réseaux booléens et leurs sémantiques synchrone, pleinement asynchrone, et asynchrone. Les liens entre les réseaux booléens et différents formalismes informatiques classiques sont discutés dans la section 1.2, avec un focus sur les réseaux de Petri, en montrant une traduction bidirectionnelle (Chatain, Haar, Kolčák, Paulevé, & Thakkar 2020). Enfin, les complexités pour la décision de différentes propriétés dynamiques usuelles (points fixes, accessibilité, attracteurs) sont données dans la section 1.3.

Un réseau booléen est une fonction des vecteurs binaires de dimension n vers eux-mêmes, $f : \mathbb{B}^n \to \mathbb{B}^n$ avec $\mathbb{B} = \{0, 1\}$. Il peut être considéré comme un système dynamique discret lorsqu'il est associé avec une sémantique qui spécifie les évolutions possibles d'un vecteur binaire $x \in \mathbb{B}^n$ sachant que $f(x) = x'$. Étant donné $i \in \{1, \ldots, n\}$, on note $f_i : \mathbb{B}^n \to \mathbb{B}$ la i-ème composante de f, aussi appelée fonction locale de i ; ainsi $f(x) = (f_1(x), \ldots, f_n(x))$.

Exemple 1.1. Avec la fonction f de dimension 3 où

$$f_1(x) = (\neg x_1 \vee \neg x_2) \wedge x_3$$
$$f_2(x) = x_1 \wedge x_3$$
$$f_3(x) = x_1 \vee x_2 \vee x_3\,,$$

$f(011) = (f_1(011), f_2(011), f_3(011)) = 101$, $f(100) = 001$, *etc.*

Nous appelons *configuration* du réseau les vecteurs binaires de dimension n. Une configuration modélise l'état de tous les composants du réseau. Pour les applications en biologie, ces composantes peuvent faire référence à des gènes, protéines, ARN, *etc.*, mais aussi des notions plus abstraites comme des phénotypes (apoptose, arrêt du cycle cellulaire, *etc.*) ; avec selon les cas différentes interprétations biologiques de l'état 0 et 1 de ces composantes.

Intuitivement, chaque f_i modélise les conditions nécessaires et suffisantes pour activer (état 1) le i-ème composant du réseau, en fonction de l'état de tous les composants du réseau. Nous verrons dans la section suivante différentes manières d'interpréter f (sémantiques) pour calculer la *dynamique* des configurations d'un réseau booléen.

Certains résultats présentés dans les chapitres suivants font référence à des réseaux booléens dont la fonction f est *localement monotone* : pour chaque $i \in \{1,\dots,n\}$, f_i dépend soit toujours positivement, soit toujours négativement, soit aucunement de chaque composante $j \in \{1,\dots,n\}$. Intuitivement, si l'on représente chaque f_i en logique propositionnelle, cela revient à ce qu'un littéral n'apparaisse jamais avec deux signes opposés dans une forme normale conjonctive ou disjonctive minimale. Ceci exclut par exemple les fonctions du type $f_1(x) = x_2 \text{ xor } x_3$, où la croissance de x_2 peut entraîner à la fois une croissance et une décroissance de $f_1(x)$ selon la valeur de x_3. À noter que la monotonie locale n'implique pas la monotonie de f. Cette classe de fonctions peut se caractériser formellement ainsi :

Définition 1.1. La fonction $f : \mathbb{B}^n \to \mathbb{B}^n$ est *localement monotone* si pour chaque composante $i \in \{1,\dots,n\}$, il existe un ordre par composantes $\preceq^i \in \{\leq, \geq\}^n$ tel que

$$\forall x, y \in \mathbb{B}^n, \quad (x_1 \preceq^i_1 y_1 \wedge \dots \wedge x_n \preceq^i_n y_n) \Rightarrow f_i(x) \leq f_i(y)$$

Le réseau booléen donné précédemment en exemple est localement monotone, par exemple avec $\preceq^1 = (\geq, \geq, \leq)$ et $\preceq^2 = \preceq^3 = (\leq, \leq, \leq)$.

La majorité des modèles booléens de réseaux biologiques ont une fonction localement monotones : l'action d'un facteur de transcription ou d'un inhibiteur sur l'activité d'un gène est souvent monotone, avec un effet de seuil (effet négligeable sous un seuil de concentration, et effet important au delà du seuil).

Notations

Dans la suite de ce manuscrit, f désigne la fonction d'un réseau booléen de dimension n. L'ensemble des composantes $\{1,\dots,n\}$ est abrégé en $[\![n]\!]$. Étant données deux configurations $x, y \in \mathbb{B}^n$, les composantes dont l'état diffère sont dénotées par $\Delta(x,y) := \{i \in [\![n]\!] \mid x_i \neq y_i\}$. Le symbole $\wedge$ dénote la conjonction, $\vee$ la disjonction et $\neg$ la négation. Étant donné un ensemble fini S, $|S|$ est sa cardinalité.

1.1 Sémantiques (a)synchrones des réseaux booléens

La fonction $f : \mathbb{B}^n \to \mathbb{B}^n$ d'un réseau booléen spécifie l'état vers lequel tend chaque composant dans chaque configuration. La sémantique définit alors précisément comment ces configurations évoluent avec le temps. Historiquement, deux sémantiques sont principalement usitées : la sémantique de mise à jour synchrone où tous les composants changent d'état simultanément (Kauffman 1969), et la sémantique pleinement asynchrone (Thomas 1973), où seul un composant, choisi de manière non déterministe, peut changer d'état. De nombreuses autres variantes ont été introduites et étudiées dans la littérature, comme les sémantiques séquentielles

et bloc-séquentielles, où est spécifié l'ordre des composants à mettre à jour (Aracena, Goles, Moreira, & Salinas 2009, Robert 1995). Dans ce chapitre, nous traitons de la sémantique synchrone, pleinement asynchrone, et asynchrone, cette dernière englobant les changements possibles avec les sémantiques précédemment citées.

1.1.1 Définitions

Les sémantiques de mise à jour sont définies ici comme des relations binaires non réflexives entre les configurations du réseau. Nous utilisons le symbole $\rightarrow$ décoré de f et d'un symbole représentant la sémantique employée.

Définition 1.2 (Mise à jour synchrone)**.**

$$\forall x, y \in \mathbb{B}^n \quad x \xrightarrow[\mathrm{s}]{f} y \overset{\Delta}{\Longleftrightarrow} x \neq y \wedge y = f(x) \tag{1.1}$$

Définition 1.3 (Mise à jour pleinement asynchrone[a])**.**

$$\forall x, y \in \mathbb{B}^n, \quad x \xrightarrow[\mathrm{a}]{f} y \overset{\Delta}{\Longleftrightarrow} \exists i \in [\![n]\!] : \Delta(x, y) = \{i\} \wedge y_i = f_i(x) \tag{1.2}$$

a. Dans la littérature des réseaux booléens, le *pleinement asynchrone* est souvent appelé (seulement) *asynchrone*, alors que l'*asynchrone*, relativement peu étudié, est souvent appelé *asynchrone généralisé*. Nous adoptons dans ce manuscrit la terminologie classique pour les systèmes dynamiques discrets (Fatès 2018, Manzoni 2012).

Définition 1.4 (Mise à jour asynchrone)**.**

$$\forall x, y \in \mathbb{B}^n, \quad x \xrightarrow[\mathrm{a}*]{f} y \overset{\Delta}{\Longleftrightarrow} x \neq y \wedge \forall i \in \Delta(x, y), y_i = f_i(x) \tag{1.3}$$

Étant donnée une sémantique σ, on écrit $\xrightarrow[\sigma]{f}{}^*$ la fermeture réflexive transitive de la relation binaire $\xrightarrow[\sigma]{f}$. Il en résulte que $x \xrightarrow[\sigma]{f}{}^* y$ si et seulement si $x = y$ ou s'il existe une séquence $x \xrightarrow[\sigma]{f} x' \xrightarrow[\sigma]{f} \cdots \xrightarrow[\sigma]{f} y$. L'ensemble des configurations en relation réflexive transitive avec une configuration x est donné par $\rho^f_\sigma(x)$:

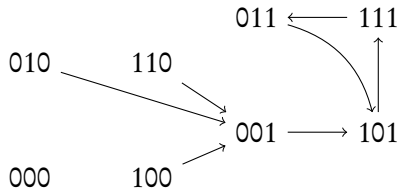
$$\rho^f_\sigma(x) := \{y \in \mathbb{B}^n \mid x \xrightarrow[\sigma]{f}{}^* y\}\,. \tag{1.4}$$

1.1.2 Exemples

Considérons la fonction f de dimension 3 donnée en exemple p.5.

Sémantique synchrone

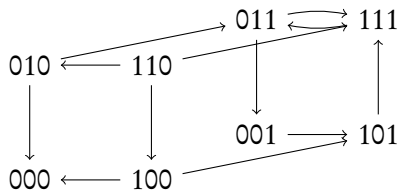
La relation binaire $\xrightarrow[s]{f}$ est la suivante, où nous omettons les décorations sur les arcs par souci de lisibilité :



Ainsi, $\rho_s^f(111) = \{111, 011, 101\}$ et $\rho_s^f(010) = \{010, 001, 101, 111, 011\}$.

Sémantique pleinement asynchrone

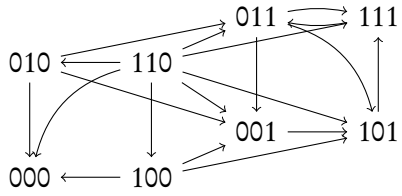
La relation binaire $\xrightarrow[a]{f}$ est la suivante :



Ici, $\rho_a^f(111) = \{111, 011, 001, 101\}$ et $\rho_a^f(010) = \{010, 000, 011, 001, 101, 111\}$.

Sémantique asynchrone

La relation binaire $\xrightarrow[a*]{f}$ est la suivante :



Ici, $\rho_{a*}^f(111) = \rho_a^f(111)$ et $\rho_{a*}^f(110) = \rho_a^f(110)$.

1.1.3 Propriétés

Le choix d'une sémantique peut avoir un effet majeur sur les évolutions des configurations d'un réseau booléen. Étant donnés un réseaux f, un sémantique σ et une configuration initiale x, l'ensemble des configurations $\rho^f_\sigma(x)$ détermine complètement les futures configurations possibles du réseau. Or, dans le cas général, cet ensemble peut être sensiblement différent selon la sémantique sélectionnée : $\rho^f_{\mathrm{s}}(x) \neq \rho^f_{\mathrm{a}}(x) \neq \rho^f_{\mathrm{a}*}(x)$. Dans l'exemple précédent, la configuration 010 ne peut pas atteindre la configuration 000 selon la sémantique synchrone, contrairement aux sémantiques asynchrones.

Par définition, la sémantique asynchrone inclut les sémantiques synchrone et pleinement asynchrone. Ainsi, $\xrightarrow[\mathrm{s}]{f}{}^* \subseteq \xrightarrow[\mathrm{a}*]{f}{}^*$ et $\xrightarrow[\mathrm{a}]{f}{}^* \subseteq \xrightarrow[\mathrm{a}*]{f}{}^*$, et donc pour toute configuration $x \in \mathbb{B}^n$, $\rho^f_{\mathrm{s}}(x) \subseteq \rho^f_{\mathrm{a}*}(x)$ et $\rho^f_{\mathrm{a}}(x) \subseteq \rho^f_{\mathrm{a}*}(x)$. Dans le cas général, la sémantique asynchrone peut inclure d'autres relations (il existe des f telles que $\xrightarrow[\mathrm{s}]{f}{}^* \cup \xrightarrow[\mathrm{a}]{f}{}^* \neq \xrightarrow[\mathrm{a}*]{f}{}^*$), dues aux mises à jour simultanées d'entre 2 et $n-1$ composants.

Noual et Sené (2018) ont étudié la classe de réseaux booléens localement monotones dits "insensibles" au synchronisme, où toute mise à jour synchrone peut se décomposer en séquences de mises à jour pleinement asynchrones. Leur résultat repose sur le *graphe d'influence* des réseaux booléens, qui résume les dépendances dirigées et signées entre les composantes et leurs fonctions. Il résulte que si le graphe d'influence de f est simple et qu'il ne contient aucun cycle de longueur paire avec un nombre pair d'arcs négatifs, et aucun cycle de longueur impaire avec un nombre impair d'arcs négatifs, alors f vérifie la propriété suivante : $\forall x \in \mathbb{B}^n$, $\rho^f_{\mathrm{s}}(x) \subseteq \rho^f_{\mathrm{a}}(x)$ et $\rho^f_{\mathrm{a}}(x) = \rho^f_{\mathrm{a}*}(x)$ [1]. Une façon d'interpréter ce résultat est de remarquer que, dans un réseau insensible au synchronisme, la mise à jour synchrone de plusieurs composants ne crée pas de nouveaux comportements par rapport à la mise à jour pleinement asynchrone.

De nombreuses autres sémantiques ont été étudiés dans la littérature, tels que les séquentiels et séquentiels par bloc, imposant un ordre (partiel) entre les composants à mettre à jour (Aracena et al. 2009). Ces sémantiques donnent des ensembles de configurations accessibles inclus dans ceux de la sémantique asynchrone.

1.2 Formalismes connexes

1.2.1 Automates cellulaires

Les réseaux booléens appartiennent à la grande famille des systèmes dynamiques discrets. De part leur spécification sous forme de fonction, les réseaux booléens peuvent s'apparenter aux automates cellulaires (Wolfram 1983). Un automate cellulaire est défini sur une grille de cellules selon un nombre de dimensions fixé. La grille est généralement infinie (classiquement $\mathbb{Z}^m$), mais peut également être finie, par exemple un anneau $\mathbb{Z}/n\mathbb{Z}$. Chaque cellule prend un état parmi un ensemble discret et fini de valeurs ; une configuration associe à chaque cellule un

1. le résultat exact de Noual et Sené (2018) donne en réalité une condition plus simple pour l'insensibilité, mais nécessite des définitions supplémentaires que nous n'abordons pas ici.

état. Enfin, une fonction dite de voisinage spécifie un ensemble de translations pour obtenir les cellules "voisines" (par exemple les cellules adjacentes) et une (unique) règle locale spécifie le calcul de l'état de chaque cellule en fonction de l'état de son voisinage. Outre la dimension finie des réseaux booléens, leur différence fondamentale par rapport aux automates cellulaires est que chaque composant (cellule) a une fonction de voisinage et une règle locale qui lui est propre. À l'inverse, les automates cellulaires à 2 valeurs en dimension finie peuvent être vus comme un cas particulier de réseaux booléens, où la règle locale est dupliquée pour chaque composant. Tout comme pour les réseaux booléens, différentes sémantiques d'application de la règle locale similaires à celles définies dans la section précédente ont été étudiées (Fatès 2018).

1.2.2 Réseaux de Petri

Les réseaux booléens, tout comme les automates cellulaires, ont une spécification basée sur les fonctions : l'état de chaque composant est calculé par une fonction, évaluée sur la configuration du réseau. Pour savoir si un composant peut changer d'état, il faut ainsi évaluer sa fonction locale et comparer le résultat avec son état actuel. Les réseaux de Petri (Murata 1989, Petri 1962) ont une spécification fondée sur des nœuds transitions qui listent explicitement les conditions dans lesquelles un ou plusieurs changements d'état ont lieu. Ce type de représentation permet entre autres de raisonner sur la causalité des changements d'états des composants, et d'éviter des explorations redondantes de configurations en exploitant la *concurrence* (indépendance) entre certaines transitions.

De nombreuses variantes de réseaux de Petri ont été considérées dans la littérature. Nous choisissons ici de montrer la relation des réseaux booléens avec les réseaux de Petri *saufs avec arcs de lecture* (*safe Petri nets with read arcs*) (Vogler, Semenov, & Yakovlev 1998), une variante classique des méthodes formelles autour de la théorie de la concurrence dans les systèmes dynamiques discrets, et dont le lien avec les réseaux booléens est le plus direct.

Un réseau de Petri est un graphe dirigé biparti, composé d'un ensemble de *places* et de *transitions*. Les places peuvent contenir des jetons : les transitions consomment un jeton de chaque place entrante et produisent un jeton dans chaque place sortante. Dans la suite de ce chapitre, nous considérons qu'une place possède soit aucun soit un seul jeton. Ainsi, le *marquage* d'un réseau de Petri est l'ensemble des places marquées. Dans les réseaux de Petri avec arcs de lecture, chaque transition est associée avec un ensemble de places de pré-condition, post-condition, et de contexte ; par souci de simplicité, nous considérons que ces ensembles sont disjoints.

Définition 1.5. Un *réseau de Petri avec arcs de lecture* (RPN ; *Read Petri Net*) est un tuple $(P, T, pre, ctx, post, M_0)$ où P et T sont respectivement des ensembles finis de *places* et de *transitions* ; *pre*, *cont* et $post : T \to 2^P$ associent chaque transition $t \in T$ à un ensemble de places, tels que $pre(t) \neq \emptyset$, $pre(t) \cap post(t) = \emptyset$, et $ctx(t) \cap (pre(t) \cup post(t)) = \emptyset$; $M_0 \subseteq P$ est le *marquage initial*.

Dans la suite, N désigne un RPN $(P, T, pre, ctx, post, M_0)$.

Une transition ne peut être déclenchée que si toutes les places de sa pré-condition et de son contexte sont marquées. L'application d'une transition enlève dans un premier temps du marquage les places de la pré-condition, puis ajoute au marquage les places de la post-condition ; les places du contexte sont préservés dans le marquage. Partant d'un marquage initial, un réseau de Petri est dit *sauf* si dans tout marquage accessible, aucune transition déclenchable n'a de place de sa post-condition dans le marquage.

Définition 1.6. La *sémantique atomique* d'un RPN N est une relation binaire entre marquages $\xrightarrow[\text{atom}]{N} \subseteq 2^P \times 2^P$ telle que $\forall M, M' \subseteq P$,

$$M \xrightarrow[\text{atom}]{N} M' \overset{\Delta}{\iff} \exists t \in T : pre(t) \cup ctx(t) \subseteq M \wedge M' = (M \setminus pre(t)) \cup post(t)$$

Exemple 1.2. La figure 1.1 donne la représentation graphique du RPN $N = (P, T, pre, ctx, post, M_0)$ où $P = \{p_1, p_2, p_3, p_4\}$, $T = \{a, b, c\}$, $pre(a) = post(b) = \{p_1\}$, $post(a) = pre(b) = \{p_3\}$, $pre(c) = \{p_2\}$, $post(c) = \{p_4\}$, $ctx(a) = \{p_2\}$, $ctx(b) = ctx(c) = \emptyset$. et $M_0 = \{p_1, p_2\}$.

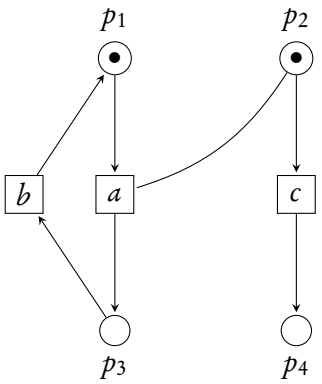


FIGURE 1.1 – Exemple de RPN sauf. Les places sont représentées par des cercles et les transitions par des carrés ; les pré- et post-conditions par des arcs dirigés, et les contextes par des arcs non-dirigés.

On obtient $\rho^N_{\text{atom}}(\{p_1, p_2\}) = \{\{p_1, p_2\}, \{p_3, p_2\}, \{p_3, p_4\}, \{p_1, p_4\}\}$. Des exemples de séquences de transitions sont les suivantes : $\{p_1, p_2\} \xrightarrow[\text{atom}]{N} \{p_1, p_4\}$, $\{p_1, p_2\} \xrightarrow[\text{atom}]{N} \{p_3, p_2\} \xrightarrow[\text{atom}]{N} \{p_1, p_2\} \xrightarrow[\text{atom}]{N} \{p_1, p_4\}$, ou encore $\{p_1, p_2\} \xrightarrow[\text{atom}]{N} \{p_3, p_2\} \xrightarrow[\text{atom}]{N} \{p_3, p_4\} \xrightarrow[\text{atom}]{N} \{p_1, p_4\}$; où $\rho^N_{\text{atom}}(\{p_1, p_4\}) = \{\{p_1, p_4\}\}$.

Nous pouvons déjà effectuer une analogie entre la sémantique atomique des RPN, où une et une seule transition est appliquée à la fois, et la sémantique pleinement asynchrone des réseaux

booléens, où un et un seul composant est mis à jour à la fois. Toutefois, il faut bien remarquer qu'une transition d'un RPN peut modifier simultanément le marquage de plusieurs places.

Tout comme pour les réseaux booléens, l'application simultanée de transitions est largement étudiée dans la littérature des RPN (Chatain et al. 2015, Janicki, Kleijn, Koutny, & Mikulski 2015) : on parle ici de sémantiques par *pas* (*step semantics*). Deux transitions peuvent être appliquées simultanément seulement si l'intersection entre leurs pre-conditions est vide, mais avec potentiellement des places partagées par leurs contextes.

Définition 1.7. Un ensemble non-vide de transitions $S \subseteq T$ est un *pas* admissible dans un marquage $M \subseteq P$ si pour tout $t \in S$, $pre(t) \cup ctx(t) \subseteq M$ et pour tout $t, t' \in S$, $pre(t) \cap pre(t') = \emptyset$.

La *sémantique par pas* est une relation binaire entre marquages de N $\xrightarrow[\text{step}]{N} \subseteq 2^P \times 2^P$ telle que $\forall M, M' \subseteq P$,

$$M \xrightarrow[\text{step}]{N} M' \overset{\Delta}{\iff} \exists S \subseteq T : S \text{ est un pas admissible dans } M \wedge M' = (M \setminus \bigcup_{t \in S} pre(t)) \cup \bigcup_{t \in S} post(t)$$

Définition 1.8. Un pas S admissible dans un marquage M est maximal s'il n'existe aucune transition $t \in T \setminus S$ telle que $S \cup \{t\}$ est un pas admissible dans M.

La *sémantique par pas maximal* est une relation binaire entre marquages de N $\xrightarrow[\text{mstep}]{N} \subseteq 2^P \times 2^P$ telle que $\forall M, M' \subseteq P$,

$$M \xrightarrow[\text{mstep}]{N} M' \overset{\Delta}{\iff} \exists S \subseteq T : S \text{ est un pas } \textit{maximal} \text{ admissible dans } M \wedge M' = (M \setminus \bigcup_{t \in S} pre(t)) \cup \bigcup_{t \in S} post(t)$$

La sémantique par pas est analogue à la sémantique asynchrone des réseaux booléens, et la sémantique par pas maximal à la sémantique synchrone.

Remarque 1.1. Il est important de remarquer que pour un RPN N sauf et sans arc de lecture (pour toute transition $t \in T$, $ctx(t) = \emptyset$), ces trois sémantiques (atomique, par pas, par pas maximal) sont similaires : toute application simultanée de transitions peut se décomposer en séquences d'applications de transitions avec la sémantique atomique. Ainsi, un RPN sauf sans arc de lecture est insensible au synchronisme : $\rho^N_{\text{mstep}}(M_0) \subseteq \rho^N_{\text{atom}}(M_0) = \rho^N_{\text{step}}(M_0)$.

Des réseaux booléens aux réseaux de Petri...

Les réseaux booléens peuvent se traduire en RPN saufs où chaque composant $i \in [\![n]\!]$ est modélisé par deux places, une pour chaque valeur possible du composant, et où les transitions reflètent l'ensemble des conditions (modélisées par des arcs de lecture) pour les changements de valeurs suivant f_i. Ces changements de valeurs sont possibles dans toute configuration $x \in \mathbb{B}^n$ où $x_i \neq f_i(x)$. Les conditions minimales pour ces changements sont obtenues symboliquement avec les impliquants premiers [2] (*prime implicants*) des formules en logique propositionnelle du premier ordre $\neg x_i \wedge f_i(x)$ et $x_i \wedge \neg f_i(x)$. Il est à noter que cela peut résulter en un nombre de transitions exponentiel par rapport au nombre de composants du réseau.

L'encodage décrit ici est similaire à celui initialement proposé par Chaouiya, Remy, Ruet, et Thieffry (2004), hormis l'emploi d'arcs de lecture qui permet d'exploiter plus précisément les notions de concurrence et de sensibilité au synchronisme. Cette traduction résulte en une classe particulière de RPN saufs où à toute place p correspond une unique place distincte $\overline{p}$ (complément) telles que tout marquage accessible contient soit p soit $\overline{p}$ et jamais les deux ; et où toute transition a exactement une seule place en pré-condition et son complément en post-condition (avec $\overline{\overline{p}} = p$).

On note $\mathrm{PI}[F]$ les impliquants premiers de la formule F, qui donnent un ensemble de clauses normales conjonctives. Un impliquant $C \in \mathrm{PI}[F]$ est représenté par un ensemble de littéraux, positifs ou négatifs ; $[x_i] \in C$ est vrai si et seulement si C contient le littéral x_i positif, et $[\neg x_i] \in C$ si C contient le littéral x_i négatif. Pour chaque composant $i \in [\![n]\!]$, deux places modélisant les états possibles de i sont créées : une place i pour l'état 0, et une place $i+n$ pour l'état 1.

Définition 1.9. (Chatain, Haar, Kolčák, Paulevé, & Thakkar 2020)

Étant donné un réseau booléen f de dimension n et une configuration $x^0 \in \mathbb{B}^n$, $(\!|f|\!)$ est le RPN $(P, T, pre, ctx, post, M_0)$ tel que

- $P = \{1, \ldots, 2n\}$ sont les places ;
- T est le plus petit ensemble tel que pour tout $i \in [\![n]\!]$, pour chaque impliquant $C \in \mathrm{PI}[\neg x_i \wedge f_i(x)]$ (resp. $C \in \mathrm{PI}[x_i \wedge \neg f_i(x)]$), il existe une transition $t \in T$ telle que $pre(t) = \{i\}$ et $post(t) = \{i + n\}$ (resp. $pre(t) = \{i + n\}$ et $post(t) = \{i\}$), et $ctx(t) = \{j \mid [\neg x_j] \in C, j \neq i\} \cup \{j + n \mid [x_j] \in C, j \neq i\}$;
- $M_0 = (\!|x^0|\!)$, où, pour toute configuration $x \in \mathbb{B}^n$, $(\!|x|\!) := \{i + nx_i \mid i \in [\![n]\!]\}$ (p.ex., $(\!|010|\!) = \{1, 5, 3\}$, $(\!|101|\!) = \{4, 2, 6\}$).

La figure 1.2 donne un exemple de RPN résultant de cet encodage.

2. conjonctions de littéraux minimales (au sens de l'inclusion de l'ensemble des littéraux) impliquant la formule logique.

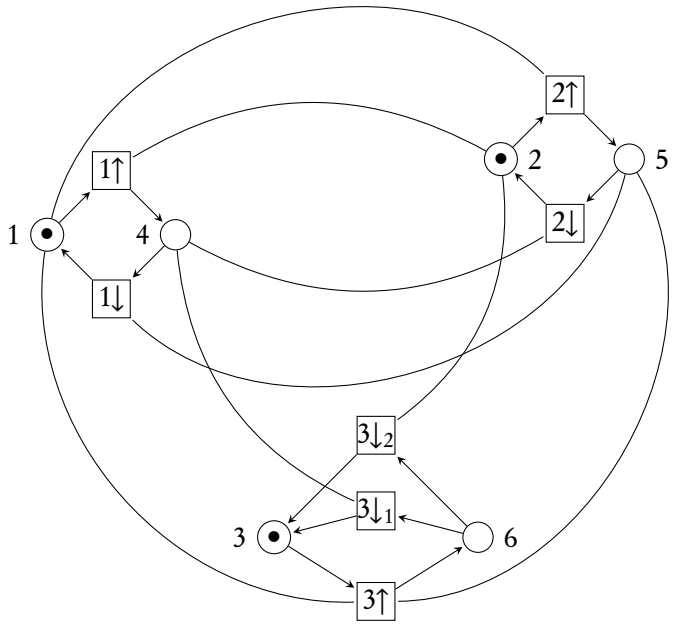


FIGURE 1.2 – RPN correspondant au réseau booléen f de dimension 3 avec $f_1(x) = \neg x_2$, $f_2(x) = \neg x_1$, $f_3(x) = \neg x_1 \wedge x_2$ et la configuration initiale 000. Les transitions sont notées $i\uparrow$ et $i\downarrow$ pour le changement d'état du composant i de 0 vers 1 et inversement, avec un indice dans les cas où plusieurs impliquants sont nécessaires : $\mathrm{PI}[x_3 \wedge \neg f_3(x)] = \{\{x_1\}, \{\neg x_2\}\}$, d'où les transitions $3\downarrow_1$ et $3\downarrow_2$ correspondant aux impliquants x_1 (place 4) et $\neg x_2$ (place 2).

On obtient alors une relation de bisimilarité entre tout réseau booléen et son encodage en réseau de Petri en considérant

— la sémantique pleinement asynchrone et atomique ;
— synchrone et par pas maximal,
— asynchrone et par pas :

Théorème 1.1. *(Chatain, Haar, Kolčák, Paulevé, & Thakkar 2020)*

Étant donné un réseau booléen f de dimension n, pour toute configuration $x, y \in \mathbb{B}^n$,

$$x \xrightarrow[\mathrm{a}]{f} y \iff (\!|x|\!) \xrightarrow[\mathrm{atom}]{(\!|f|\!)} (\!|y|\!)\,,$$

$$x \xrightarrow[\mathrm{s}]{f} y \iff (\!|x|\!) \xrightarrow[\mathrm{mstep}]{(\!|f|\!)} (\!|y|\!)\,,$$

$$x \xrightarrow[\mathrm{a*}]{f} y \iff (\!|x|\!) \xrightarrow[\mathrm{step}]{(\!|f|\!)} (\!|y|\!)$$

... et vice versa

Tout RPN $N = (P, T, pre, ctx, post, M_0)$ sauf peut également se traduire en réseau booléen, avec une complexité linéaire par rapport au nombre de places et de transitions.

La principale difficulté est qu'une transition d'un RPN peut modifier simultanément le marquage de plusieurs places, même avec la sémantique atomique. L'idée est d'associer un composant du réseau booléen à chaque place et à chaque transition du RPN et de mimer l'application d'une transition t en plusieurs étapes : quand les composants correspondant à la pré-condition et au contexte d'une transition ont tous la valeur 1, et qu'aucun composant correspondant à une transition n'est dans l'état 1, le composant correspondant à t peut être mis à 1. Alors, les composants pour les places de pré- et post-conditions sont mis à jour (dans n'importe quel ordre) pour *in fine* refléter leur présence dans le marquage final. Enfin, le composant pour t est mis à 0. La fonction modélisant le marquage d'une place p s'évalue à 1 si et seulement si une transition où p est une post-condition est en cours ; ou p est marquée et aucune transition où p est une pré-condition n'est en cours. La fonction modélisant l'application d'une transition t s'évalue à 1 si et seulement si aucune transition n'est en cours et toutes les places de la pré-condition et du contexte sont marquées ; ou la transition t est en cours et soit une place de la post-condition n'est pas encore marquée soit une place de la pré-condition est encore marquée.

Il est à noter que la traduction, et plus particulièrement ses propriétés, font l'hypothèse que le RPN est *sans boucle*, c'est-à-dire qu'aucune place n'est à la fois en pré- et post-condition d'une transition $\forall t \in T, pre(t) \cap post(t) = \emptyset$. Une telle place doit être alors inscrite dans le contexte de la transition. Enfin, pour simplifier les notations, nous faisons l'hypothèse que l'ensemble des places et des transitions sont des entiers de 1 à $|P|+|T|$, autrement dit $P \cup T \equiv \{1, \ldots, |P|+|T|\}$.

Définition 1.10. (Chatain, Haar, Kolčák, Paulevé, & Thakkar 2020)

Étant donné un RPN $N = (P, T, pre, ctx, post, M_0)$ sauf et sans boucle, $[\![N]\!]$ est le réseau booléen de dimension $n = |P| + |T|$ tel que

$$\forall p \in P, [\![N]\!]_p(x) = \left(\bigvee_{t \in T : p \in post(t)} x_t \right) \vee \left(x_p \wedge \bigwedge_{t \in T : p \in pre(t)} \neg x_t \right),$$

$$\forall t \in T, [\![N]\!]_t(x) = \left(\bigwedge_{t' \in T} \neg x_{t'} \wedge \bigwedge_{p \in pre(t) \cup ctx(t)} x_p \right) \vee \left(x_t \wedge \left(\bigvee_{p \in post(t)} \neg x_p \vee \bigvee_{p \in pre(t)} x_p \right) \right).$$

Étant donné un marquage $M \subseteq P$ de N, la configuration $[\![M]\!] \in \mathbb{B}^n$ est définie par $\forall p \in M, [\![M]\!]_p = 1$, $\forall p \in P \setminus M, [\![M]\!]_p = 0$, et $\forall t \in T, [\![M]\!]_t = 0$.

Exemple 1.3. Le réseau de Petri de la figure 1.1 (p.11) à 4 places et 3 transitions se traduit dans le réseau booléen de dimension 7 suivant :

$$f_{p_1}(x) = x_b \vee (x_{p_1} \wedge \neg x_a) \qquad f_{p_2}(x) = x_{p_2} \wedge \neg x_c$$
$$f_{p_3}(x) = x_a \vee (x_{p_3} \wedge \neg x_b) \qquad f_{p_4}(x) = x_c \vee x_{p_4}$$
$$f_a(x) = (\neg x_a \wedge \neg x_b \wedge \neg x_c \wedge x_{p_1} \wedge x_{p_2}) \vee (x_a \wedge (\neg x_{p_3} \vee x_{p_1}))$$
$$f_b(x) = (\neg x_a \wedge \neg x_b \wedge \neg x_c \wedge x_{p_3}) \vee (x_b \wedge (\neg x_{p_1} \vee x_{p_3}))$$
$$f_c(x) = (\neg x_a \wedge \neg x_b \wedge \neg x_c \wedge x_{p_2}) \vee (x_c \wedge (\neg x_{p_4} \vee x_{p_2}))$$

Son marquage initial ($\{p_1, p_2\}$) se traduit par la configuration 1100000.

Le théorème ci-après établit la correspondance entre les marquages et configurations accessibles entre les réseaux de Petri et leur encodage en réseau booléen, en montrant que la sémantique atomique des réseaux de Petri est équivalente à la sémantique pleinement asynchrone des réseaux booléens, la sémantique par pas des réseaux de Petri est équivalente à la sémantique asynchrone des réseaux booléens, et la sémantique par pas maximal des réseaux de Petri est équivalente à la sémantique synchrone des réseaux booléens. Le fait que les RPN soient sans boucle est crucial pour ces deux dernières correspondances pour assurer qu'avec tout marquage accessible M, tout ensemble de transitions $S \subseteq T$ tel que $\forall t \in S, [\![N]\!]_t([\![M]\!]) = 1$ est un pas admissible dans M.

Théorème 1.2. *(Chatain, Haar, Kolčák, Paulevé, & Thakkar 2020)*

Étant donné un RPN $N = (P, T, pre, ctx, post, M_0)$ sauf et sans boucle,

$$M \in \rho^N_{\text{atom}}(M_0) \iff [\![M]\!] \in \rho_{\text{a}}^{[\![N]\!]}([\![M_0]\!]),$$
$$M \in \rho^N_{\text{step}}(M_0) \iff [\![M]\!] \in \rho_{\text{a*}}^{[\![N]\!]}([\![M_0]\!]),$$
$$M \in \rho^N_{\text{mstep}}(M_0) \iff [\![M]\!] \in \rho_{\text{s}}^{[\![N]\!]}([\![M_0]\!])$$

Discussion

Les réseaux de Petri permettent de spécifier explicitement la causalité des transitions (les conditions pour qu'elles soient possibles). On peut alors mettre en évidence des ensembles de transitions qui peuvent être appliquées dans n'importe quel ordre pour finir toujours sur le même marquage. On parle de *concurrence* : deux transitions distinctes t et t' sont concurrentes si, depuis tout marquage dans lesquels elles sont toutes deux applicables, il est possible à la fois de faire t puis t', et t' puis t, en atteignant le même nouveau marquage. Les méthodes de *dépliage* (*unfoldings*) prennent avantage de ces transitions concurrentes pour calculer une représentation compacte de l'ensemble des exécutions possibles du réseau (Baldan et al. 2012, Esparza & Heljanko 2008). Ce type d'approche a été appliqué pour analyser les comportements de grands modèles de réseaux biologiques présentant de la concurrence (Chatain et al.

2014, Chatain & Paulevé 2017, Kolčák, Šafránek, Haar, & Paulevé 2018).

La structure des modèles en réseaux de Petri est très proche des produits synchrones de machines à état finis (automates) (Murata 1989), où plusieurs places mutuellement exclusives (au plus une peut être marquée à la fois) représentent les différents états d'un même automate. Un réseau de Petri n'indique pas explicitement ces différents automates, mais ils peuvent être déduits par un calcul d'invariants de places (par ex., Paulevé 2018).

La traduction présentée pour convertir un réseau de Pétri en réseau booléen pourrait être largement améliorée par ce calcul d'invariants de places pour chercher des ensembles de places pouvant se modéliser par une seule variable booléenne, de telle sorte que, pour tout réseau booléen f, $[\![(\!|f|\!)]\!] = f$.

Enfin, nous attirons l'attention sur l'existence de sémantiques de RPN différentes des sémantiques classiques atomiques et par pas. En particulier la sémantique par intervalle vise à reproduire des comportements existant dans des systèmes où les transitions se déroulent sur des intervalles de temps (Chatain et al. 2015). Cette sémantique peut être définie directement sur les réseaux booléens (Chatain et al. 2018) et montre que les sémantiques usuelles ne capturent pas certains comportements, pourtant potentiellement pertinents.

1.3 Propriétés dynamiques et complexité

L'analyse de la dynamique des réseaux booléens repose le plus souvent sur trois propriétés fondamentales : les points fixes, caractérisant les configurations qui ne peuvent plus évoluer ; l'accessibilité, caractérisant l'existence de trajectoires entre deux configurations données ; et les attracteurs, généralisant les points fixes, et caractérisant les comportements limites du réseau.

Basée sur des résultats de complexité sur des modèles de calculs proches des réseaux booléens, cette section établit quelques résultats de complexité en lien avec ces propriétés avec les sémantiques synchrone, pleinement asynchrone et asynchrone. Ainsi, le problème d'existence de point fixe est NP-complet, et le problème de décision des propriétés d'accessibilité et d'appartenance à un attracteur sont PSPACE-complets.

1.3.1 Points fixes

Les points fixes des réseaux booléens sont une des propriétés les plus systématiquement étudiée. Pour la modélisation en biologie, ces points fixes correspondent à des régimes particulièrement stables de la cellule, où tous les composants restent qualitativement constants.

Définition 1.11. Une configuration $x \in \mathbb{B}^n$ est un *point fixe* du réseau booléen f avec la sémantique σ si $\rho^f_\sigma(x) = \{x\}$.

Pour les sémantiques de mise à jour synchrone, pleinement asynchrone, et asynchrone, les points fixes sont identiques aux points fixes de f, c'est-à-dire aux configurations $x \in \mathbb{B}^n$ telles que $f(x) = x$.

Théorème 1.3. *Déterminer si le réseau booléen f possède au moins un point fixe est un problème NP-complet avec la sémantique synchrone, pleinement asynchrone, et asynchrone.*

La preuve peut s'effectuer par réduction du problème SAT (Alon 1985, Floréen & Orponen 1989, Formenti, Manzoni, & Porreca 2014).

Dans différents outils d'analyse de réseaux booléens, le calcul des points fixes de la fonction f est effectué à l'aide de manipulations de diagrammes de décision binaire (Naldi, Thieffry, & Chaouiya 2007) ou encore par résolution de problème de satisfaction booléenne (SAT) (Chevalier, Froidevaux, Paulevé, & Zinovyev 2019), et est applicable à des réseaux de très grande dimension ($n > 1000$).

1.3.2 Accessibilité entre configurations

L'existence d'au moins une trajectoire menant d'une configuration $x \in \mathbb{B}^n$ à une autre configuration $y \in \mathbb{B}^n$ est appelé accessibilité (*reachability*) :

Définition 1.12. Étant données deux configurations $x, y \in \mathbb{B}^n$ et une sémantique σ, y est *accessible* depuis x si $y \in \rho^f_\sigma(x)$.

Pour les applications en biologie, deux familles de propriétés liées à l'accessibilité sont couramment employées : les observations de marqueurs au sein d'une cellule au cours du temps (séries temporelles) se traduisent en propriétés d'accessibilité *positives* (existence) ; alors que les observations de phénomènes de différenciation se traduisent en propriétés d'accessibilité *négatives* (absence) : sans perturbation externe, une cellule différenciée dans un sous-type A ne peut pas atteindre un état correspondant au sous-type B, ni un état pluripotent précédent.

Avec les sémantiques (a)synchrones, il s'avère que le problème d'accessibilité est PSPACE-complet, et donc extrêmement difficile à résoudre pour des réseaux de grande dimension, même comparé à l'existence de points fixes [3].

Théorème 1.4. *Déterminer si $y \in \rho^f_\sigma(x)$ est un problème PSPACE-complet avec les sémantiques de mise à jour synchrone, pleinement asynchrone et asynchrone ($\sigma \in \{s, a, a*\}$).*

Intuitivement, la complexité peut se comprendre par le fait que deux configurations peuvent être liées seulement par des trajectoires de taille exponentielle avec n (Melliti, Regnault, Richard, & Sené 2016). Ainsi, même avec la sémantique synchrone qui est déterministe, une simulation peut prendre un temps exponentiel pour la vérification.

Le nombre de configurations accessibles étant au plus 2^n, ce problème est au plus PSPACE [4] car il suffit de simuler au plus $2^n - 1$ itérations du réseau booléen de manière non déterministe

3. Il n'est pas (encore) connu si NP=PSPACE ; mais en pratique la résolution de problèmes NP (SAT) peut s'appliquer à des problèmes avec des millions de variables, alors que la résolution de problèmes PSPACE (p. ex., model-checking par CTL) se limite à quelques centaines de variables.

4. PSPACE=NPSPACE=coNPSPACE (Papadimitriou 1995).

avec un compteur sur n bits.

Plusieurs preuves sur des modèles similaires aux réseaux booléens permettent de conclure que le problème d'accessibilité est PSPACE-complet. Dans le cas synchrone, cela découle directement du problème d'accessibilité dans les systèmes de réactions, une sous-classe des réseaux booléens synchrones, démontré PSPACE-complet par Dennunzio, Formenti, Manzoni, et Porreca (2019).

Pour le cas asynchrone et pleinement asynchrone, outre mentionner l'analogie avec le problème d'accessibilité dans les automates cellulaires finis asynchrones et pleinement asynchrones, démontré PSPACE-complet par Dennunzio, Formenti, Manzoni, Mauri, et Porreca (2015), l'aspect PSPACE-difficile peut se démontrer, entre autres, de plusieurs façons :

— Par réduction du problème d'accessibilité dans les réseaux de Petri saufs sans arcs de lecture, démontré PSPACE-complet par Cheng, Esparza, et Palsberg (1995) avec la sémantique atomique (et donc également par pas, d'après la remarque en fin d'introduction de la section 1.2.2), avec l'encodage donné dans la section 1.2.2 (Chatain, Haar, Kolčák, Paulevé, & Thakkar 2020).

— Par réduction du problème d'accessibilité avec la sémantique synchrone. Tout comme pour les automates cellulaires (Nakamura 1981), il est possible de construire un réseau booléen f' dont les sémantiques asynchrone et pleinement asynchrone préservent les propriétés d'accessibilité d'un réseau booléen f avec la sémantique synchrone, avec un nombre de dimensions polynomial avec n.
Le principe général d'une construction possible est de décomposer une itération synchrone en plusieurs étapes qui peuvent être effectuées de manière asynchrone. Trois étapes sont à discerner : (a) le calcul de la valeur suivante pour chaque composante $i \in [\![n]\!]$; (b) l'application du nouvel état pour chaque composant ; (c) la remise à zéro des composantes supplémentaires employées pour la décomposition. Cela peut s'effectuer avec un réseau f' avec $3n + 2$ dimensions : une composante z dont l'état 1 signale la remise à zéro des composantes supplémentaires (hormis z) ; une composante w dont l'état 0, sous condition que z est dans l'état 0, signale l'étape de calcul, et l'état 1 signale l'étape d'application. Pour chaque composante $i \in [\![n]\!]$ du réseau initial, deux composantes ci et $\bar{\text{c}}i$ sont ajoutées, dont l'état 1 indique respectivement si f_i est vrai ou faux. La fin de l'étape de calcul est déterminée par le fait que pour toute composante $i \in [\![n]\!]$, soit ci soit $\bar{\text{c}}i$ est dans l'état 1 ; la composante w passe alors dans l'état 1 ; les composantes i passent alors à l'état 0 si et seulement si $\bar{\text{c}}i$ vaut 1 et à l'état 1 si et seulement si ci vaut 1. La fin de l'étape d'application est déterminée par le fait que toutes les composantes i ont la valeur calculée ; la composante z passe alors dans l'état 1 qui va déclencher la transition vers l'état 0 des composants w et ci et $\bar{\text{c}}i$. Enfin, la composante z repasse dans l'état 0, ce qui permet aux composantes ci et $\bar{\text{c}}i$ de calculer la prochaine valeur de chaque composante $i \in [\![n]\!]$. Un tel réseau $f' : \mathbb{B}^{3n+2} \to \mathbb{B}^{3n+2}$ peut se définir de la façon

suivante, où $x_{1..n}$ dénote la configuration x tronquée aux n premières composantes :

$$
\begin{aligned}
f'_i(x') &= ((\neg x'_{\mathrm{w}} \vee x'_{\mathrm{z}}) \wedge x'_i) \vee (x'_{\mathrm{w}} \wedge \neg x'_{\mathrm{z}} \wedge x'_{\mathrm{c}i}) \\
f'_{\mathrm{c}i}(x') &= \neg x'_{\mathrm{z}} \wedge \left((\neg x'_{\mathrm{w}} \wedge f_i(x'_{1..n})) \vee (x'_{\mathrm{w}} \wedge x'_{\mathrm{c}i})\right) \\
f'_{\bar{\mathrm{c}}i}(x') &= \neg x'_{\mathrm{z}} \wedge \left((\neg x'_{\mathrm{w}} \wedge \neg f_i(x'_{1..n})) \vee (x'_{\mathrm{w}} \wedge x'_{\bar{\mathrm{c}}i})\right) \\
f'_{\mathrm{w}}(x') &= \neg x'_{\mathrm{z}} \wedge \left(\left(x'_{\mathrm{w}} \vee \textstyle\bigwedge_{i\in[\![n]\!]}(x'_{\mathrm{c}i} \vee x'_{\bar{\mathrm{c}}i})\right)\right) \\
f'_{\mathrm{z}}(x') &= \left(x'_{\mathrm{w}} \wedge \textstyle\bigwedge_{i\in[\![n]\!]}(x'_{\mathrm{c}i} \Leftrightarrow x'_i \wedge x'_{\bar{\mathrm{c}}i} \Leftrightarrow \neg x'_i)\right) \\
&\qquad \vee \left(x'_{\mathrm{z}} \wedge \left(x'_{\mathrm{w}} \vee \textstyle\bigvee_{i\in[\![n]\!]}(x'_{\mathrm{c}i} \vee x'_{\bar{\mathrm{c}}i})\right)\right)
\end{aligned}
$$

Les sémantiques asynchrone et pleinement asynchrone du réseau f' préservent exactement les propriétés d'accessibilité de la sémantique synchrone de f :

$$y \in \rho^f_{\mathrm{s}}(x) \iff y0^{2n+2} \in \rho^{f'}_{\mathrm{a}*}(x0^{2n+2}) \iff y0^{2n+2} \in \rho^{f'}_{\mathrm{a}}(x0^{2n+2}) \tag{1.5}$$

pour toutes configurations $x, y \in \mathbb{B}^n$, où 0^{2n+2} est le vecteur 0 de dimension $2n+2$ et $y0^{2n+2}$ dénote sa concaténation à y.

En pratique, la vérification de propriétés d'accessibilité peut s'effectuer par des représentations symboliques de l'ensemble des configurations accessibles, notamment avec des structures de données inspirées des diagrammes de décision binaires (Hamez, Thierry-Mieg, & Kordon 2009). Une approche complémentaire est l'exploration en ordre partiel des changements d'états possibles, notamment avec les dépliages des réseaux de Petri (Esparza & Heljanko 2008, Haar, Kern, & Schwoon 2013, Rodríguez & Schwoon 2013). Ces méthodes permettent d'analyser des réseaux booléens avec quelques centaines de dimensions, selon la structure du réseau et l'emploi d'analyses statiques pour réduire le modèle (Chatain & Paulevé 2017, Paulevé 2018).

1.3.3 Attracteurs

Les attracteurs d'un réseau booléen caractérisent ses comportements limites. Dans ce manuscrit, nous les définissons comme des ensembles de configurations, où toutes les paires ont une trajectoire entres elles, et où aucune configuration en dehors de l'attracteur n'est accessible depuis l'attracteur. Ainsi un attracteur ne spécifie pas de structure entre ses configurations (par exemple indiquant des cycles particuliers entre ses configurations), comme cela peut être parfois le cas dans la littérature.

Définition 1.13. Un ensemble non-vide de configurations $A \subseteq \mathbb{B}^n$ est un *attracteur* du réseau booléen f avec une sémantique σ si $\forall x \in A, \rho^f_\sigma(x) = A$.

Les attracteurs généralisent les points fixes (un point fixe est un attracteur composé d'une seule configuration) et reposent sur la notion d'accessibilité abordée dans la section précédente.

Les attracteurs modélisent typiquement des comportements stables de la cellule, potentiellement oscillatoires dans le cas des attracteurs qui ne sont pas des points fixes, c'est-à-dire quand ils sont composés d'au moins deux configurations.

Nous nous intéressons à la complexité du problème de décision suivant : étant donnée une configuration $x \in \mathbb{B}^n$, x appartient à un attracteur.

Avec la sémantique synchrone, pleinement asynchrone, et asynchrone, décider si une configuration x appartient à un attracteur est PSPACE-complet. Une façon simple de le démontrer est d'étudier le problème complémentaire : x n'appartient pas à un attracteur si et seulement si il existe une configuration y telle que y est accessible depuis x, mais x n'est pas accessible depuis y. D'après la complexité de l'accessibilité, ce problème est dans NPSPACE, et donc l'appartenance à un attracteur dans coNPSPACE = PSPACE.

Théorème 1.5. *Déterminer si une configuration $x \in \mathbb{B}^n$ appartient à un attracteur de f est PSPACE-complet avec la sémantique synchrone, asynchrone, et pleinement asynchrone.*

Pour le cas synchrone, ce résultat découle de la complexité de ce problème dans le cas des systèmes de réactions synchrones, cas particuliers des réseaux booléens, démontré PSPACE-complet par Dennunzio et al. (2019). Pour les cas asynchrones et pleinement asynchrones, cela peut alors se démontrer par réduction du cas synchrone, avec la construction esquissée dans la section précédente sur la complexité de l'accessibilité.

En pratique, le calcul exact des attracteurs des réseaux booléens passe difficilement à l'échelle. Des méthodes reposant sur des représentations symboliques des états accessibles tels que les dépliages de réseaux de Petri (Chatain et al. 2014) ou les diagrammes de décision binaires (Mizera, Pang, Qu, & Yuan 2017, 2019) sont limitées à des dimensions de l'ordre de 50 à 100 selon les spécificités structurelles du réseau et la sémantique choisie.

Chapitre 2

La sémantique MP (Most Permissive)

Les réseaux booléens sont communément employés pour modéliser des systèmes dont l'état des composants n'est pas binaire, en particulier en biologie. Ce chapitre commence par montrer que les sémantiques usuelles des réseaux booléens masquent des comportements pourtant possibles dans un système respectant la même logique de régulation.
Nous proposons alors une nouvelle interprétation des réseaux booléens, avec la sémantique *MP* (*Most Permissive Boolean Networks*), qui apporte des garanties formelles très fortes sur les comportements prédits vis-à-vis de modèles multivalués et continus. De plus, la décision des propriétés dynamiques étudiées dans le chapitre précédent a une complexité considérablement réduite avec la sémantique MP, ouvrant la voie à l'étude de modèles à l'échelle du génome.
Enfin, ce chapitre conclut sur l'application de la sémantique MP à des cas d'études de la littérature qu'elle arrive à reproduire exactement avec un coût sensiblement réduit, montrant ainsi qu'elle conserve un caractère prédictif important pour les processus de différenciation et de décision cellulaire.

2.1 Introduction

Lorsqu'ils sont appliqués pour la modélisation des systèmes biologiques, les réseaux booléens imposent une forte simplification de l'activité des composants. De nombreux autres formalismes permettent d'intégrer des précisions supplémentaires sur les quantités et vitesses des interactions (réseaux multivalués, réseaux de réactions chimiques, équations différentielles ordinaires, modèles stochastiques, *etc.*). Cependant, leurs spécifications demandent de nombreux paramètres, souvent continus, qui sont généralement inconnus.

L'activité des gènes et la concentration des ARN et protéines n'étant pas binaires, nous pouvons nous demander à quel point une simulation ou une prédiction effectuée à partir d'un réseau booléen peut être transposable au système modélisé, ou à des modèles plus précis.

Idéalement, un modèle abstrait se dérive mathématiquement depuis un modèle initial dé-

taillé (approche *bottom-up*). Des garanties peuvent être dérivées formellement à partir des propriétés des abstractions utilisées (p. ex., Abou-Jaoudé, Thieffry, & Feret 2016, Fages & Soliman 2008, Feret, Danos, Krivine, Harmer, & Fontana 2009, Glass & Kauffman 1973, Snoussi 1989). Cependant, ce schéma ne correspond pas toujours à la façon dont sont employés les réseaux booléens. En pratique, les réseaux booléens sont souvent construits directement, soit complètement manuellement (à partir de la littérature), soit automatiquement ou semi-automatiquement à partir de bases de données référençant des influences connues entre gènes et protéines, et des données expérimentales. De plus, les réseaux intègrent parfois des processus biologiques abstraits (p. ex., phénotypes) dont l'interaction avec les autres gènes/protéines serait très difficile à expliciter. Il n'y a alors, *a priori*, aucune garantie que les résultats de leur analyse puissent se transposer à un modèle plus précis, et donc au système biologique étudié.

La figure 2.1 illustre ce problème avec une boucle de réaction anticipée incohérente de type 3, *I3-FFL*, d'après la classification de Mangan et Alon (2003). Un nœud d'entrée 1 inhibe directement la sortie 3, mais l'active indirectement par le nœud 2. Des études théoriques (Ishihara, Fujimoto, & Shibata 2005, Rodrigo & Elena 2011) et des données expérimentales provenant de circuits synthétiques (Schaerli et al. 2014) montrent qu'une activation progressive de l'entrée peut mener à une activation transitoire de la sortie. Il est cependant impossible de reproduire ce comportement avec les interprétations classiques des réseaux booléens, dont la sémantique synchrone et asynchrone : si 1 n'est pas actif, ni 2 ni 3 ne peuvent être activés ; si 1 est actif, 2 peut bien s'activer, mais il est impossible d'activer 3, même de façon transitoire (Fig. 2.1(d)).

Des modèles plus avancés, utilisant par exemple des états intermédiaires pour les composants, ou des délais dans les interactions, permettraient de prédire cette activation transitoire de la sortie de I3-FFL. Mais ces détails supplémentaires nécessitent des paramètres de modélisation supplémentaires, et souvent un coût de calcul plus élevé, ce qui limite leur application systématique à des réseaux impliquant beaucoup d'acteurs.

Cet exemple semble montrer que la caricature binaire de l'activité des nœuds peut générer à la fois des comportements artificiels (mais ceci est généralement attendu et admis avec les modèles qualitatifs), mais, et de manière moins intuitive, peut également masquer des comportements existants. Ainsi, la validité d'un modèle ne peut pas être formellement garantie avec les interprétations classiques des réseaux booléens. Ceci limite grandement les méthodes d'inférence de réseaux et l'identification de motifs nécessaires d'interactions, puisque l'analyse booléenne peut fortement biaiser les modèles à considérer.

Cependant, nous avons trouvé que ce problème est dû à la sémantique choisie pour interpréter les réseaux booléens, et non à leur nature binaire.

La sémantique *MP* (*Most Permissive semantics* ; (Paulevé et al. 2020)) est une nouvelle interprétation des réseaux booléens qui permet de capturer ces trajectoires, sans ajouter d'information ni de paramètres. En particulier cette sémantique garantit de capturer tout comportement (changement d'états) réalisable par tout raffinement quantitatif du réseau booléen (au sens de la section 2.3.1) avec toute sémantique de mise à jour.

Ainsi, un raffinement du réseau booléen ne peut que supprimer des trajectoires observables dans cette abstraction booléenne, mais ne peut pas créer de nouvelles trajectoires. Les réseaux booléens avec la sémantique MP apportent ainsi une approximation supérieure des trajectoires. En conséquence, si l'analyse avec la sémantique MP conclut qu'aucune trajectoire n'existe entre

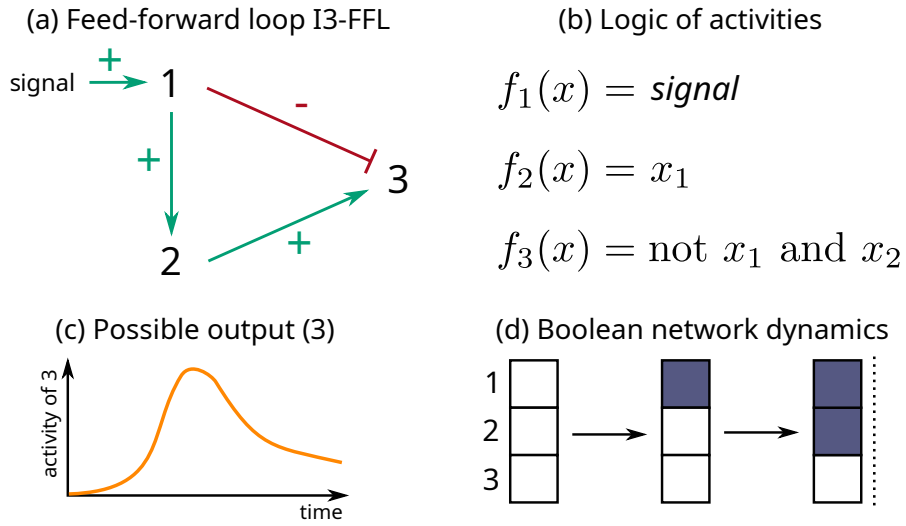


FIGURE 2.1 – (extraite de Paulevé et al. 2020) Boucle de réaction anticipée incohérente de type 3 (a) et sa logique booléenne associée (b). Alors que des études théoriques et expérimentales montrent une possible activation transitoire de la sortie quand le signal est actif (c), l'analyse booléenne classique ne peut pas prédire ce comportement ; (d) montre la dynamique complète de f depuis la configuration initiale : 3 n'est jamais prédit actif. Les configurations sont représentées par des piles de carrés, où le carré du haut représente l'état du premier composant, *etc.* Un carré blanc représente l'état inactif (0), un carré bleu l'état actif (1) ; la ligne en tirets indique qu'aucune évolution est possible. Les flèches indiquent les transitions possibles.

deux configurations données, alors aucun raffinement ne peut rendre possible cette trajectoire.

Alors que la sémantique MP prédit plus de trajectoires que les sémantiques classiques, nous montrons, que non seulement cette abstraction est minimale, mais qu'elle continue toujours d'être suffisamment restrictive pour prédire des comportements biologiques essentiels.

De plus, nous démontrons que la complexité de l'analyse avec la sémantique MP est considérablement réduite : la décision de l'accessibilité est dans P dans le cas où le réseau est localement monotone ; P^{NP} dans le cas contraire ; et la décision d'appartenance à un attracteur est dans coNP pour les réseaux localement monotones ; dans $\text{coNP}^{\text{coNP}}$ dans le cas contraire. En pratique, cela rend possible l'analyse formelle de réseaux de dimension de plusieurs ordres de grandeurs supérieurs aux sémantiques classiques ; comme nous l'illustrons sur des exemples de réseaux booléens de dimension 100 000.

2.2 Définitions

Avec les sémantiques de mise à jour, un composant booléen va basculer directement depuis son état minimal à son état maximal ou inversement. Comme illustré dans la section précédente, ceci masque certains comportements possibles dans des états intermédiaires, qui peuvent être capturés dans des raffinements multivalués.

Nous présentons ici deux formulations équivalentes de la sémantique MP.

2.2.1 Avec états dynamiques

Étant donné un réseau booléen f, la sémantique MP définit un ensemble de configurations binaires accessibles depuis une configuration binaire initiale. La formulation donnée ici est une façon de calculer cet ensemble à l'aide de pseudo-états pour les composants, appelés dans la suite *états dynamiques*. Ces états dynamiques peuvent être vus comme un moyen intermédiaire pour implémenter la sémantique MP et ils n'introduisent aucun paramètre.

Cette définition de la sémantique MP repose sur deux notions clés : le changement d'état des composants est décomposé en deux temps, avec un (pseudo) *état dynamique* indiquant une augmentation $\nearrow$ ou une diminution $\searrow$. L'ensemble des états possibles d'un composant est noté $\mathbb{P} := \{0, \nearrow, \searrow, 1\}$. Lorsqu'un composant est dans un état dynamique, les autres composants du réseau peuvent lire arbitrairement 0 ou 1. Par exemple, si un composant i est dans l'état $\nearrow$, un composant j peut l'évaluer à 1 alors qu'un composant k peut l'évaluer à 0. Ceci permet de considérer implicitement le fait que i peut avoir différents seuils d'influence selon les composants du réseau.

La fonction $\gamma : \mathbb{P}^n \to 2^{\mathbb{B}^n}$ associe à toute configuration $x \in \mathbb{P}^n$ l'ensemble des configurations booléennes qui lui correspondent :

$$\gamma(x) := \{x' \in \mathbb{B}^n \mid \forall i \in [\![n]\!], x_i \in \mathbb{B} \Rightarrow x'_i = x_i\}\,. \tag{2.1}$$

Par exemple, $\gamma(1 \nearrow 0) = \{100, 110\}$ et $\gamma(\nearrow\searrow\nearrow) = \mathbb{B}^3$.

Nous définissons d'abord cette sémantique par une relation binaire entre configurations dans $\mathbb{P}^n$; puis nous donnerons dans la section suivante une définition alternative à l'aide d'hypercubes clos par f (*trap spaces*).

En résumé, un composant $i \in [\![n]\!]$ dans l'état booléen 0 peut passer dans l'état $\nearrow$ s'il existe une façon x' d'évaluer la configuration (parmi $\gamma(x)$) telle que $f_i(x') = 1$; depuis $\nearrow$, il peut soit passer dans l'état booléen 1 sans condition, soit dans l'état $\searrow$ s'il existe une façon x' d'évaluer la configuration telle que $f_i(x') = 0$; depuis l'état booléen 1, il peut passer dans l'état $\searrow$ selon cette même dernière condition ; enfin depuis l'état $\searrow$, il peut soit passer dans l'état booléen 0 sans condition, soit dans l'état $\nearrow$ avec les mêmes conditions que le passage de 0 vers $\nearrow$. La figure 2.2 résume l'automate correspondant pour chaque composant.

Définition 2.1 (Sémantique MP de f)**.**

$$\forall x, y \in \mathbb{P}^n, \quad x \xrightarrow[\text{mp}]{f} y \overset{\Delta}{\Longleftrightarrow} \exists i \in [\![n]\!] : \Delta(x, y) = \{i\}$$

$$\wedge y_i = \begin{cases} \nearrow & \text{si } x_i \neq 1 \wedge \exists z \in \gamma(x) : f_i(z) \\ 1 & \text{si } x_i = \nearrow \\ \searrow & \text{si } x_i \neq 0 \wedge \exists z \in \gamma(x) : \neg f_i(z) \\ 0 & \text{si } x_i = \searrow \end{cases} \tag{2.2}$$

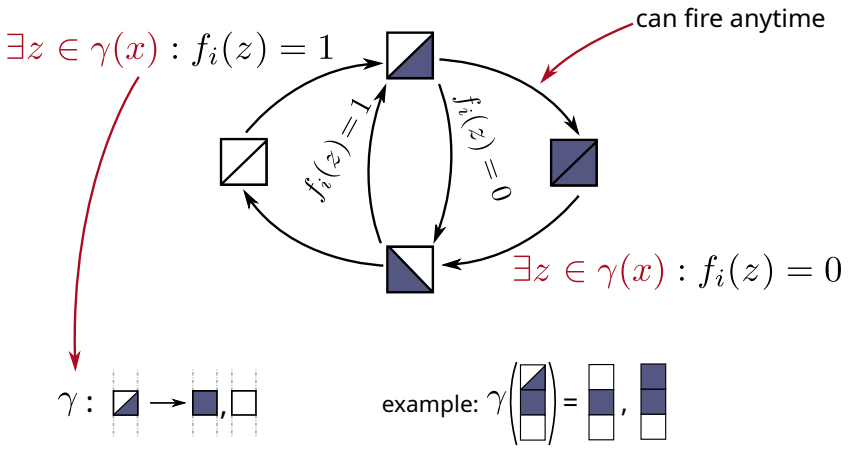


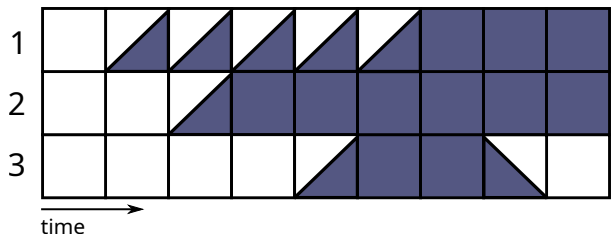
FIGURE 2.2 – (extraite de Paulevé et al. 2020) Automate de mise à jour des états du composant $i \in [\![n]\!]$ avec la sémantique MP depuis une configuration $x \in \mathbb{P}^n$. L'état dynamique ↗ est représenté par un carré avec la diagonale supérieure gauche en blanc et inférieure droite en bleu, et l'état ↘ avec la diagonale inférieure gauche en bleu et supérieure droite en blanc.

Remarque 2.1. La sémantique telle que définie ci-dessus effectue des changements d'état de façon pleinement asynchrone, mais c'est équivalent de la définir en asynchrone : on remplace la condition $\exists i \in [\![n]\!] : \Delta(x,y) = \{i\} \wedge y_i = \cdots$ par $x \neq y \wedge \forall i \in [\![n]\!] : y_i = \cdots$.

Appliquée au réseau booléen f de la Fig. 2.1, la sémantique MP prédit, entre autres, la séquence de configurations suivantes :

$$000 \xrightarrow[\text{mp}]{f} \nearrow 00 \xrightarrow[\text{mp}]{f} \nearrow\nearrow 0 \xrightarrow[\text{mp}]{f} \nearrow 10 \xrightarrow[\text{mp}]{f} \nearrow 1 \nearrow \xrightarrow[\text{mp}]{f} \nearrow 11 \xrightarrow[\text{mp}]{f} 111 \xrightarrow[\text{mp}]{f} 11 \searrow \xrightarrow[\text{mp}]{f} 110$$

que l'on peut représenter plus graphiquement comme suit :



La sémantique MP prédit bien l'activation transitoire du composant 3. Dans la configuration ↗ 10, le composant 3 peut lire la valeur 0 pour le composant 1, ce qui lui permet de passer dans l'état ↗ et d'atteindre, transitoirement, la configuration 111.

2.2.2 Avec hypercubes

L'ajout des états dynamiques ↗ et ↘ pourrait suggérer que la sémantique MP est proche d'un réseau multivalué (au sens de Thomas et d'Ari (1990)) avec 4 valeurs. Mais ce n'est pas le

cas : les états $\mathbb{P}$ ne sont pas totalement ordonnés par les transitions autorisées par la sémantique, comme c'est le cas des réseaux multivalués où les changements d'états sont unitaires.

La sémantique MP peut également se définir en termes d'hypercubes clos par f. Un hypercube est une partie de $\mathbb{B}^n$ où des composantes ont une valeur constante, et les autres sont libres (notées $*$) :

Définition 2.2. Un *hypercube* h de dimension n est un vecteur dans $(\mathbb{B} \cup \{*\})^n$. L'ensemble des configurations associées est dénoté par $c(h) := \{x \in \mathbb{B}^n \mid \forall i \in [\![n]\!], h_i \neq * \Rightarrow x_i = h_i\}$.

Étant donnés deux hypercubes $h, h' \in (\mathbb{B} \cup \{*\})^n$, h est *plus petit* que h' si et seulement si $c(h) \subseteq c(h')$, ou de manière équivalente, $\forall i \in [\![n]\!], h'_i \neq * \Rightarrow h_i = h'_i$. Un hypercube est *minimal* s'il n'existe pas d'hypercube différent plus petit que lui.

Un hypercube $h \in (\mathbb{B} \cup \{*\})^n$ est *clos* par $f : \mathbb{B}^n \to \mathbb{B}^n$ si pour toute configuration $x \in c(h)$, $f(x) \in c(h)$. Nous généralisons cette dernière définition pour n'imposer la clôture que pour un sous-ensemble de composantes $K \subseteq [\![n]\!]$:

Définition 2.3. Étant donné $K \subseteq [\![n]\!]$, un hypercube $h \in (\mathbb{B} \cup \{*\})^n$ est K-clos par f si pour toute configuration $x \in c(h)$, pour toute composante $i \in K$, $h_i \in \{*, f_i(x)\}$.

Remarque 2.2. Un hypercube h est clos par f si et seulement si h est $[\![n]\!]$-clos par f. Un hypercube h clos par f est minimal si et seulement si c'est le plus petit hypercube clos par f contenant x, pour tout $x \in c(h)$.

Exemple 2.1. Soit $f : \mathbb{B}^3 \to \mathbb{B}^3$ telle que $f_1(x) := \neg x_2$, $f_2(x) := \neg x_1$, et $f_3(x) := \neg x_1 \wedge x_2$. L'hypercube $01*$ est clos par f, avec $c(01*) = \{010, 011\}$. L'hypercube $1*0$ est le plus petit hypercube $\{2,3\}$-clos par f contenant 110 ; il n'est pas clos par f, ni le plus petit $\{2,3\}$-clos par f contenant 100.

Partant d'une configuration x, la sémantique MP peut s'exprimer avec le calcul des plus petits hypercubes contenant x et K-clos par f, pour tout ensemble K :

- x est l'unique hypercube $\emptyset$-clos par f contenant x ;
- le changement d'état d'un composant $i \in [\![n]\!]$ vers $\nearrow$ ou $\searrow$ produit une configuration x' où $\gamma(x')$ correspond à l'hypercube $h \in (\mathbb{B} \cup \{*\})^n$ avec $h_i = *$ et pour tout autre composante $j \in [\![n]\!], j \neq i$, $h_j = x_j$. Ainsi, h est le plus petit hypercube $\{i\}$-clos par f et contenant x.
- en considérant seulement les changements d'états vers $\nearrow$ ou $\searrow$, la sémantique va progressivement agrandir l'hypercube selon les composantes modifiées, et ainsi construire les plus petits hypercubes K-clos par f et contenant x, pour tout $K \subseteq [\![n]\!]$.

Avec la sémantique MP, le changement d'état d'un composant depuis $\nearrow$ ou $\searrow$ vers un état booléen est sans condition et est complètement déterminé par l'état dynamique 1 depuis $\nearrow$ et 0 depuis $\searrow$. Or, partant d'une configuration initiale booléenne, un composant peut être dans

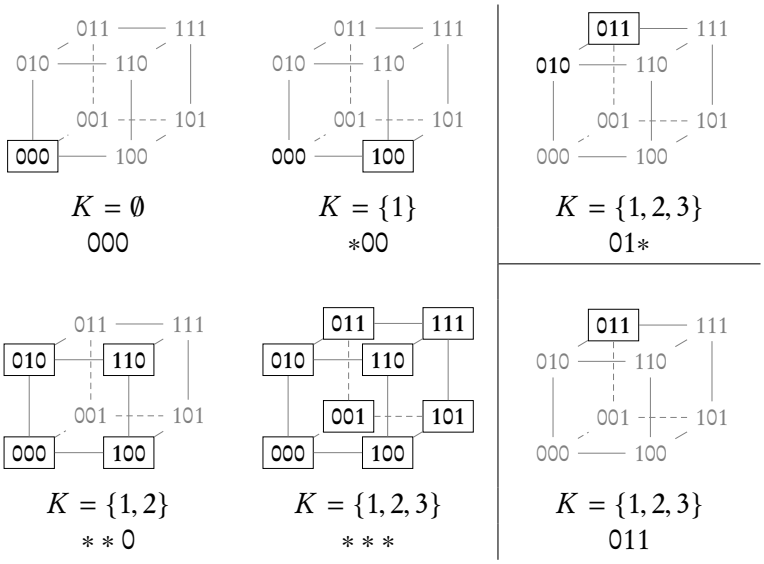


FIGURE 2.3 – Exemples de plus petits hypercubes K-clos contenant la configuration 000 (gauche), 010 (haut droite) et 011 (bas droite) pour le réseau booléen f de dimension 3 avec $f_1(x) := \neg x_2$, $f_2(x) := \neg x_1$, $f_3(x) := \neg x_1 \wedge x_2$. Les configurations appartenant à l'hypercube sont indiquées en gras ; celles vérifiant la propriété d'accessibilité sont encadrées. Seul l'hypercube 011 est clos par f et minimal.

l'état $\nearrow$ seulement si une configuration précédente $x' \in \mathbb{P}^n$ était telle qu'un $z \in \gamma(x')$ vérifiait $f_i(z) = 1$ (resp. $\searrow$ si $f_i(z) = 0$).

Ainsi, une configuration $y \in \mathbb{B}^n$ est atteinte depuis $x \in \mathbb{B}^n$ avec la sémantique MP si et seulement s'il existe un ensemble de composantes $K \subseteq [\![n]\!]$ tel que le plus petit hypercube $h \in (\mathbb{B} \cup \{*\})^n$ contenant x et K-clos par f vérifie les deux conditions suivantes : (1) il contient y, et (2) pour toute composante $i \in K$, il existe une configuration z dans h ($z \in c(h)$) telle que $f_i(z) = y_i$. La figure 2.3 illustre ce calcul d'hypercubes.

2.3 Garanties vis-à-vis de modèles quantitatifs

Nous étudions ici les garanties que peuvent offrir les réseaux booléens vis-à-vis de modèles rajoutant des informations sur les vitesses et durées des réactions ou sur les quantités des molécules. Nous considérons par la suite deux cadres de modélisation permettant une spécification plus fine de la dynamique d'un ensemble de composants en interaction : les *réseaux multivalués* de Thomas et d'Ari (1990), qui restent purement discrets et qui peuvent être vus comme une généralisation des réseaux booléens, où les composants peuvent prendre une valeur parmi un ensemble borné $\{0, \ldots, m\}$; et les *équations différentielles ordinaires* (EDO ; *Ordinary Differential Equations, ODEs*), qui spécifient précisément les évolutions des concentrations des composants au cours du temps (pour les réseaux génétiques, de Jong 2002).

2.3.1 Raffinements de réseaux booléens

Nous caractérisons ici la notion de raffinement d'un réseau booléen f par un réseau multivalué ou par un système d'EDO. Le principe adopté est le suivant : la valeur d'un composant ne peut décroître (resp. croître) dans un réseau multivalué (ou EDO) seulement si le composant peut être mis à 0 (resp. à 1) dans le réseau booléen, depuis une configuration binaire en lien avec la configuration du modèle non booléen.

Introduisons tout d'abord les réseaux multivalués.

Définition 2.4. Un *réseau multivalué* F de dimension n est une fonction discrète qui associe à toute configuration du réseau l'évolution (unitaire) de la valeur de chaque composant [a] :

$$F : \mathbb{M}^n \to \{-1, 0, 1\}^n \qquad \text{où } \mathbb{M} := \{0, 1, \ldots, m\}\,.$$

De manière similaire aux réseaux booléens, les sémantiques synchrone, pleinement asynchrone et asynchrone se définissent comme des relations binaires entre les configurations multivaluées : pour toutes les paires de configurations $x, y \in \mathbb{M}^n$

$$x \xrightarrow[\text{s}]{F} y \overset{\Delta}{\iff} x \neq y \wedge \forall i \in [\![n]\!], y_i = \max(0, \min(m, x_i + F_i(x)))\,, \tag{2.3}$$

$$x \xrightarrow[\text{a}]{F} y \overset{\Delta}{\iff} \exists i \in [\![n]\!] : \Delta(x, y) = \{i\} \wedge y_i = x_i + F_i(x)\,, \tag{2.4}$$

$$x \xrightarrow[\text{a*}]{F} y \overset{\Delta}{\iff} \forall i \in \Delta(x, y), y_i = x_i + F_i(x) \tag{2.5}$$

a. Pour simplifier les notations, mais sans perte de généralité, la valeur maximale m est fixe pour tous les composants ; en pratique elle peut être différente.

Remarque 2.3. Un réseau booléen f peut être considéré comme un réseau multivalué avec $m = 1$ et F définie telle que pour tout $i \in [\![n]\!]$, $F_i : x \mapsto \begin{cases} -1 & \text{si } \neg f_i(x) \\ 1 & \text{si } f_i(x) \end{cases}$.

Le critère de raffinement repose sur une *binarisation* des configurations multivaluées. La binarisation associe nécessairement la valeur 0 au booléen 0 et m au booléen 1, et est libre pour les valeurs intermédiaires. On note $\beta : \mathbb{M}^n \to 2^{\mathbb{B}^n}$ l'ensemble des configurations binaires associées à une configuration multivaluée $x \in \mathbb{M}^n$:

$$\beta(x) := \{x' \in \mathbb{B}^n \mid \forall i \in [\![n]\!], x_i = 0 \Rightarrow x'_i = 0 \wedge x_i = m \Rightarrow x'_i = 1\}\,. \tag{2.6}$$

Définition 2.5. Un réseau multivalué F est un *raffinement* d'un réseau booléen f de même dimension n si et seulement si, pour chaque configuration $x \in \mathbb{M}^n$, et chaque composant $i \in [\![n]\!]$:

$$F_i(x) < 0 \Longrightarrow \exists x' \in \beta(x) : \neg f_i(x')$$

$$\wedge F_i(x) > 0 \Longrightarrow \exists x' \in \beta(x) : f_i(x')$$

Comme elle repose sur le signe de la dérivée, cette caractérisation de raffinement des réseaux booléens peut être directement étendue aux systèmes d'EDO : de manière analogue aux réseaux multivalués, les EDO spécifient la dérivée de la valeur réelle de chaque composant au cours du temps continu t :

$$\frac{d\mathbb{F}(t,x)}{dt} = \mathcal{F}(x) \qquad \text{avec } \mathcal{F} : \mathbb{R}^n_{\geq 0} \to \mathbb{R}^n \,. \tag{2.7}$$

Ici, $\mathcal{F}(x)$ est la dérivée de $\mathbb{F}(t,x)$ selon le temps t en fonction des configurations continues x ; $\mathbb{F}$ étant généralement inconnue.

Seule la binarisation β doit être légèrement adaptée pour refléter l'absence *a priori* de borne maximale ($\beta(x) := \{x' \in \mathbb{B}^n \mid \forall i \in [\![n]\!], x_i = 0 \Rightarrow x'_i = 0\}$). La définition de raffinement est alors identique à celle ci-dessus. On peut enfin noter que la sémantique des EDO peut être vue comme analogue à la sémantique *synchrone* : $\mathcal{F}$ décrit l'évolution simultanée de tous les composants.

2.3.2 Contre-exemple pour les sémantiques classiques

La figure 2.4 montre un exemple de réseau booléen dont un des raffinements multivalués apporte un nouveau comportement, invisible avec la sémantique asynchrone (qui englobe les comportements synchrones et pleinement asynchrones). Ainsi toute conclusion sur l'existence ou l'absence de trajectoires asynchrones au niveau du réseau booléen ne peut pas toujours se transposer aux modèles raffinés : la sémantique asynchrone ne constitue ni une approximation supérieure (*over-approximation*), ni une approximation inférieure (*under-approximation*) des comportements des modèles raffinés.

Le réseau booléen f de l'exemple est le suivant

$$f_1(x) := \neg x_2 \qquad f_2(x) := \neg x_1 \qquad f_3(x) := \neg x_1 \wedge x_2$$

On obtient $\rho^f_{a*}(000) = \{000, 110, 010, 011, 100\}$.

Le réseau multivalué F à 3 valeurs ($m = 2$) ci-après est un raffinement de f :

$F_1(x) := 1$ si $x_2 < 2$ sinon -1

$F_2(x) := 1$ si $x_1 < 2$ sinon -1

$F_3(x) := 1$ si $x_1 < 2 \wedge x_2 \geq 1$ sinon -1

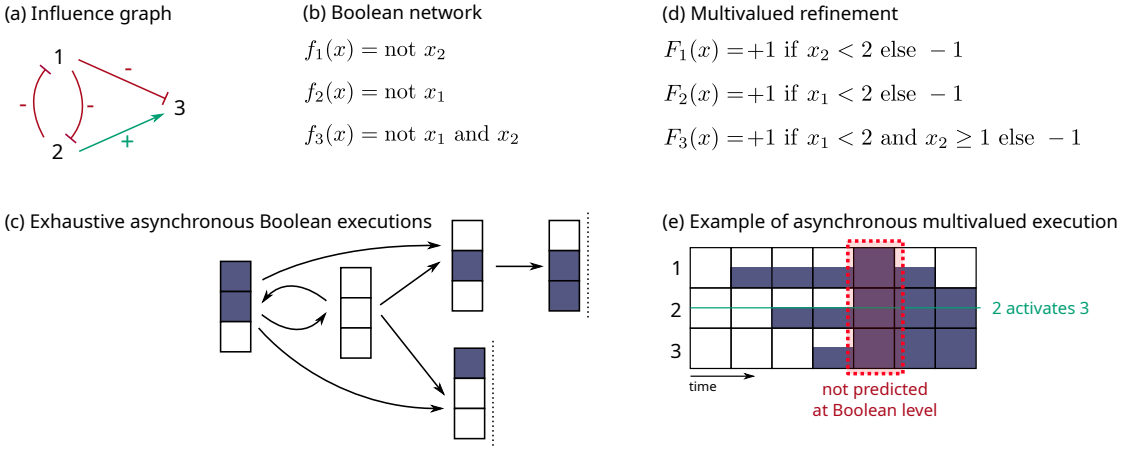


FIGURE 2.4 – (extraite de Paulevé et al. 2020) (a) Graphe d'influence listant les relations d'activation et d'inhibition entre les composants. (b) Exemple de réseau booléen compatible. (c) Liste exhaustive des transitions obtenues avec la sémantique asynchrone depuis la configuration initiale où tous les composants sont inactifs inactive. (d) Exemple de réseau multivalué raffinant (b) avec des composants à 3 états (0, 1, 2). (e) Exemple d'exécution asynchrone du réseau multivalué depuis la configuration 000. Les carrés mi- et complètement bleus représentent les états 1 et 2.

Avec la sémantique pleinement asynchrone, l'exécution suivante est possible depuis la configuration 000 :

$$000 \xrightarrow[a]{F} 010 \xrightarrow[a]{F} 110 \xrightarrow[a]{F} 111 \xrightarrow[a]{F} 112 \dots$$

Il est ainsi possible d'atteindre une configuration où les 3 composants ne sont pas inactifs, ce qui n'a pas été prédit au niveau booléen. Dans le cas où la validation de f dépendait de cette possibilité, f aurait été considéré comme invalide, à tort.

Remarque 2.4. Avec la sémantique asynchrone, il est même possible ici d'atteindre la configuration 222 :

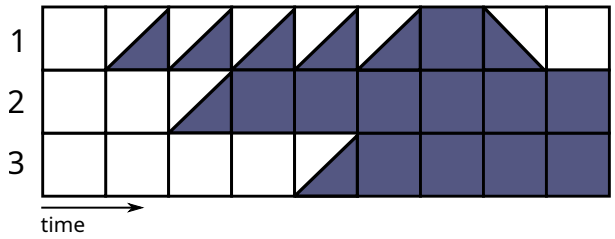
$$000 \xrightarrow[a*]{F} 010 \xrightarrow[a*]{F} 110 \xrightarrow[a*]{F} 111 \xrightarrow[a*]{F} 222 \dots$$

Imaginons maintenant un quatrième composant avec $f_4(x) := x_1 \wedge x_2 \wedge x_3$ et $F_4(x) := 1$ si $x_1 \geq 1 \wedge x_2 \geq 1 \wedge x_3 \geq 1$ sinon -1. Dans le réseau multivalué F la sémantique pleinement asynchrone prédit qu'il est possible d'activer ce composant ; contrairement à l'analyse booléenne qui prédit qu'il est impossible de l'activer. Ainsi, sans connaissances suffisantes pour définir directement un réseau multivalué, le modélisateur aurait rejeté le réseau booléen f si les données expérimentales montraient une activation de ce quatrième composant, alors qu'il constitue un modèle potentiellement pertinent du système biologique.

Ce contre-exemple montre une limitation des réseaux booléens avec les sémantiques classiques : elles ne permettent pas d'abstraire correctement la notion de seuils d'influences ni

d'ordre entre seuils. Dans l'exemple, le modèle multivalué spécifie que le composant 2 agit à des seuils différents sur 1 et 3 : il suffit que 2 soit dans l'état 1 (ou 2) pour contribuer positivement à 3, mais il doit être nécessairement dans l'état 2 pour contribuer négativement à 1 ($F_1(x) = 1$ si $\neg(x_2 \geq 2)$ sinon -1). Ainsi, lorsque le composant 2 est dans l'état 1, les composants 1 et 3 peuvent augmenter : il est suffisamment actif pour 3 mais pas encore pour 1. Il faut ainsi pouvoir différencier trois cas de figure pour l'état de 2. Avec l'interprétation booléenne classique, soit 2 est complètement inactif et 1 augmente et 3 diminue, soit complètement actif et 1 diminue et 3 augmente.

Avec la sémantique MP, le comportement peut bien être reproduit :



2.3.3 Une abstraction correcte des raffinements multivalués

Comme annoncé en début de chapitre, la sémantique MP garantit de capturer tout comportement possible avec n'importe quel raffinement de f. Pour un raffinement multivalué $F : \mathbb{M}^n \to \{-1, 0, 1\}^n$ fixé, la fonction $\alpha : \mathbb{M}^n \to 2^{\mathbb{P}^n}$ associe à toute configuration $x \in \mathbb{M}^n$ l'ensemble des configurations de la sémantique MP telles que : si un composant a une valeur extrême (0 ou m), il a la valeur booléenne correspondante, sinon les deux états dynamiques $\nearrow$ et $\searrow$ sont considérés :

$$\alpha(x) := \{\hat{x} \in \mathbb{P}^n \mid \forall i \in [\![n]\!], x_i = 0 \Leftrightarrow \hat{x}_i = 0 \wedge x_i = m \Leftrightarrow \hat{x}_i = 1\}$$

Le théorème suivant établit que pour toutes configurations multivaluées $x, y \in \mathbb{M}^n$ telles que $x \xrightarrow[\text{a*}]{F} y$, quelle que soit la configuration $\hat{x}$ choisie parmi $\alpha(x)$, il existe une configuration correspondante $\hat{y}$ dans $\alpha(y)$ telle que $\hat{y} \in \rho^f_{\text{mp}}(\hat{x})$.

Théorème 2.1. *(Paulevé et al. 2020) Soit f un réseau booléen de dimension n et $F : \mathbb{M}^n \to \{-1, 0, 1\}^n$ un raffinement multivalué. Alors,*

$$\forall x, y \in \mathbb{M}^n, \quad x \xrightarrow[\text{a*}]{F} y \implies \forall \hat{x} \in \alpha(x), \exists \hat{y} \in \alpha(y) : \hat{x} \xrightarrow[\text{mp}]{f}{}^* \hat{y}$$

$$\text{où } \forall i \in [\![n]\!], \quad \hat{y}_i = \begin{cases} \hat{x}_i & si\ y_i = x_i \\ \nearrow & si\ y_i > x_i \wedge y_i < m \\ 1 & si\ y_i > x_i \wedge y_i = m \\ \searrow & si\ y_i < x_i \wedge y_i > 0 \\ 0 & si\ y_i < x_i \wedge y_i = 0. \end{cases}$$

La démonstration découle de la définition de raffinement : pour chaque composante $i \in [\![n]\!]$, si $y_i > x_i$ (resp. $y_i < x_i$), forcément $F_i(x) > 0$ (resp. $F_i(x) < 0$). Donc, par la propriété de raffinement, il existe une binarisation $x' \in \beta(x)$ telle que $f_i(x) = 1$ (resp. $f_i(x) = 0$). Il reste à remarquer que pour tout $\hat{x} \in \alpha(x)$, $x' \in \beta(\hat{x})$. Alors, par définition de la sémantique MP, $\hat{x} \xrightarrow[\text{mp}]{f}{}^{*} \hat{y}$.

Il est important de remarquer que le théorème considère que le modèle raffiné peut être mis à jour de manière asynchrone, et donc englobe toute restriction (synchrone, pleinement asynchrone, séquentiel, *etc.*). Sous l'hypothèse que le modèle raffiné s'interprète uniquement selon la mise à jour pleinement asynchrone, une restriction de la sémantique MP est suffisante pour capturer leurs comportements (Chatain, Haar, Kolčák, Paulevé, & Thakkar 2020).

La preuve reposant uniquement sur le signe de la dérivée donnée par le raffinement de f, elle s'étend naturellement aux raffinements par EDO, qui peuvent être vus ici comme un cas limite des réseaux multivalués, appliqués avec la sémantique synchrone.

Corollaire 2.1. *Soit f un réseau booléen de dimension n et $\mathcal{F} : \mathbb{R}^n_{\geq 0} \to \mathbb{R}^n$ un système d'EDO qui raffine f. Alors,*

$$\forall x \in \mathbb{R}^n_{\geq 0}, \quad \forall \hat{x} \in \alpha(x), \hat{x} \xrightarrow[\text{mp}]{f}{}^{*} \hat{y}$$

où $\alpha(x) := \{\hat{x} \in \mathbb{P}^n \mid \forall i \in [\![n]\!], x_i = 0 \Leftrightarrow \hat{x}_i = 0 \wedge \hat{x}_i \neq 1\}$,

et $\forall i \in [\![n]\!], \hat{y}_i = \begin{cases} \nearrow & \text{si } F_i(x) > 0 \\ \searrow & \text{si } F_i(x) < 0 \wedge x_i > 0 \\ \hat{x}_i & \text{sinon} \end{cases}$

2.3.4 Minimalité de l'abstraction

La complétude de la sémantique MP vis-à-vis des raffinements possibles pourrait très bien être acquise avec une sémantique qui autorise tous les changements d'états. On peut alors se demander si la sémantique MP est minimale, c'est-à-dire si certains comportements prédits n'ont pas de correspondance avec un au moins un raffinement possible.

Dans (Chatain, Haar, Kolčák, & Paulevé 2020), nous avons pu démontrer que la sémantique MP est minimale pour les propriétés d'accessibilité pour les raffinements multivalués introduits en début de section. Ainsi, une sémantique offrant la même garantie de complétude doit prédire au moins les configurations accessibles avec la sémantique MP.

La propriété de minimalité repose sur la notion de compatibilité entre trajectoires MP et multivaluées asynchrones, qui assure une cohérence dans la succession des configurations, à la fois en termes d'abstraction des configurations, et en termes de dérivées : quand un composant i va dans l'état dynamique $\nearrow$ (resp. $\searrow$), F_i est positive (resp. négative) dans la configuration multivaluée correspondante. Alors, pour toute trajectoire MP, il existe un raffinement multivalué avec $m = 3$ qui admet une trajectoire compatible. De plus, s'il existe une trajectoire MP

entre deux configurations x et $y \in \mathbb{B}^n$, alors il existe un raffinement multivalué de f qui présente une trajectoire asynchrone entre les configurations $m.x$ et $m.y$ avec $m = 2$. Les preuves de ces deux propriétés ont été obtenues par Juraj Kolčák ; la formalisation complète est donnée dans (Chatain, Haar, Kolčák, & Paulevé 2020).

2.4 Complexité

Étudions maintenant la complexité des problèmes de décision définis dans la section 1.3 avec la sémantique MP.

Tout d'abord, il faut remarquer que les points fixes de la sémantique MP sont exactement les points fixes de f, ainsi la complexité de la décision d'existence de point fixe est identique aux sémantiques (a)synchrones (NP-complet, Théorème 1.3).

2.4.1 Accessibilité entre configurations

Déterminer si $y \in \rho^f_{\text{mp}}(x)$ est un problème bien plus simple en sémantique MP qu'avec les sémantiques asynchrones (PSPACE-complet). Nous allons l'expliquer avec la formulation de la sémantique avec les hypercubes : $y \in \rho^f_{\text{mp}}(x)$ si et seulement s'il existe un ensemble de composantes $K \subseteq [\![n]\!]$ tel que le plus petit hypercube K-clos $h \in (\mathbb{B} \cup \{*\})^n$ contenant x vérifie (1) $y \in c(h)$ et (2) pour tout $i \in K$, $\exists z \in c(h) : f_i(z) = y_i$.

Pour un K fixé, le calcul du plus petit hypercube K-clos contenant x et la vérification des deux conditions peut s'effectuer selon l'algorithme suivant :

```
h := x
répéter |K| fois: (* construction *)
    pour chaque i ∈ K tel que h_i ≠ *:
        si ∃z ∈ c(h) tel que f_i(z) ≠ x_i:
            h_i := *
pour chaque i ∈ [[n]]: (* validation *)
    si h_i ≠ * et h_i ≠ y_i:
        répondre non
    si h_i = * et si non (∃z ∈ c(h), f_i(z) = y_i):
        répondre non
répondre oui
```

Il faut d'abord remarquer qu'il n'est pas nécessaire d'explorer tout ensemble $K \subseteq [\![n]\!]$ possible. En effet, admettons qu'il existe deux composantes distinctes $j, j' \in K$ vérifiant la condition (1) mais ne vérifiant pas la condition (2) : $j \neq j'$, $h_j = h_{j'} = *$ et $\forall z \in c(h)$, $f_j(z) \neq y_j$ et $f_{j'}(z) \neq y_{j'}$. Il apparaît que j ne vérifiera pas la condition (2) avec $K \setminus \{j'\}$ ni j' avec $K \setminus \{j\}$.

Ainsi, l'exploration débute avec $K = [\![n]\!]$. Si l'algorithme ci-dessus répond non, on soustrait de K les composantes ne vérifiant pas la condition (2). À tout moment, si la condition (1) n'est pas vérifiée, on peut directement conclure à l'absence d'accessibilité, car elle ne le sera pour

aucun sous-ensemble de K. La condition (2) sera toujours vérifiée à un moment, au plus tard avec $K = \emptyset$. L'algorithme de décision est exécuté au plus n fois.

Maintenant, remarquons que l'algorithme de décision effectue plusieurs tests de la forme "$\exists z \in c(h) : f_i(z) = b$" avec $b \in \mathbb{B}$. Dans le cas général, c'est exactement le problème SAT. Le problème de décision de l'accessibilité est alors dans la classe de complexité P^{NP}. Dans le cas où f est localement monotone (voir introduction), tester si $\exists z \in c(h) : f_i(z) = b$ avec $b \in \mathbb{B}$ donné est linéaire avec n : il suffit de choisir le z maximal pour f_i si $b = 1$ et minimal si $b = 0$ (en prenant les notations de la définition de localement monotone, si $b = 1$, pour chaque $j \in [\![n]\!]$ où $h_j = *$, on choisit $z_j = 1$ si $\leq_j^i = \leq$ et $z_j = 0$ sinon, et inversement dans le cas où $b = 0$). Le problème de décision de l'accessibilité est alors dans la classe de complexité P.

Théorème 2.2. *(Paulevé et al. 2020) Déterminer si $y \in \rho^f_{\mathrm{mp}}(x)$ est dans P si f est localement monotone et dans P^{NP} sinon.*

2.4.2 Attracteurs

Les attracteurs avec la sémantique MP sont des objets bien particuliers : ils correspondent exactement aux hypercubes minimaux clos par f, aussi appelés *minimal trap spaces*.

Lemme 2.1. *$A \subseteq \mathbb{B}^n$ est un attracteur du réseau booléen f avec la sémantique MP si et seulement s'il existe un hypercube $h \in (\mathbb{B} \cup \{*\})^n$ minimal clos par f tel que $c(h) = A$.*

Ces hypercubes minimaux ont déjà été appliqués comme approximations des attracteurs de sémantiques de mises à jour (Klarner, Bockmayr, & Siebert 2015, Naldi 2018). En effet, tout hypercube minimal clos par f contient au moins un attracteur de chaque sémantique de mise à jour ; mais la réciproque n'est pas vraie.

Avec la sémantique MP, les attracteurs deviennent exactement ces hypercubes clos minimaux, ce qui rend leur identification applicable à des très grands réseaux, comme nous allons en discuter ci-après. De plus, on peut remarquer que le nombre d'attracteurs de la sémantique MP est une borne inférieure au nombre d'attracteurs de tout raffinement possible de f.

Une façon de comprendre pourquoi un attracteur de la sémantique MP est forcément un hypercube est de remarquer que si deux configurations sur une diagonale d'un hypercube sont dans un même attracteur (donc accessibles de l'un vers l'autre), alors toutes les configurations sur les sommets adjacents à la diagonale sont dans l'attracteur (figure 2.5).

Plus précisément, prenons une configuration $x \in \mathbb{B}^n$ dans un attracteur, et soit $h \in (\mathbb{B} \cup \{*\})^n$ le plus petit hypercube clos par f et contenant x. Soit la configuration $y \in c(h)$ la plus distante de x appartenant à l'hypercube : $\forall i \in [\![n]\!]$, $y_i = \neg x_i$ si $h_i = *$, sinon $y_i = x_i$. D'après la section précédente, y est accessible depuis x ($y \in \rho^f_{\mathrm{mp}}(x)$), et par hypothèse, x est également accessible depuis y. Il reste à remarquer que le plus petit hypercube clos par f et contenant y est alors h. Ainsi, pour toute composante $i \in [\![n]\!]$ libre dans h ($h_i = *$), il existe une configuration de l'hypercube $z \in c(h)$ telle que $f_i(z) = 0$ et une configuration $z' \in c(h)$ telle que $f_i(z') = 1$. Il en résulte que toute configuration de l'hypercube h est accessible depuis x.

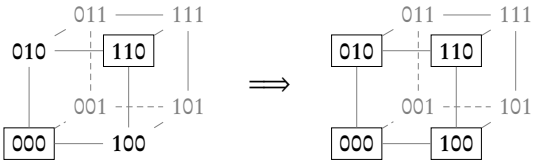


FIGURE 2.5 – Illustration de la propriété des attracteurs de la sémantique MP, avec la convention graphique de la figure 2.3 : si deux configurations sur une diagonale d'un hypercube sont accessibles deux à deux et appartiennent à un attracteur, alors les configurations adjacentes appartiennent également à l'attracteur.

Enfin, remarquons que si le plus petit hypercube h clos par f et contenant x n'est pas minimal, alors il existe une configuration $y \in c(h)$ accessible depuis x, mais x n'appartient pas au plus petit hypercube clos par f et contenant y : x n'est donc pas accessible depuis y et n'appartient donc à aucun attracteur.

Le problème de décision d'appartenance d'une configuration x donnée à un attracteur revient alors à décider si le plus petit hypercube clos par f et contenant x est minimal.

Soit N'EST-PAS-CLOS(h) le problème de décider si l'hypercube h n'est pas clos par f, autrement dit si il existe $i \in [\![n]\!]$ avec $h_i \neq *$ et $z \in c(h)$ tel que $f_i(z) \neq h_i$. Comme nous l'avons vu dans la section précédente, ce problème est dans NP dans le cas général et dans P dans le cas où f est localement monotone. Ainsi, le problème complémentaire EST-CLOS(h) est dans coNP dans le cas général et dans P dans le cas localement monotone.

Le problème N'EST-PAS-MINIMAL(h) de décider si un hypercube h clos par f est un hypercube minimal peut se résumer à trouver un hypercube h' strictement inclus dans h et clos par f, ce qui est de complexité au plus $\mathrm{NP}^{\text{EST-CLOS}}$. Ainsi, le problème complémentaire EST-MINIMAL(h) est au plus dans $\mathrm{coNP}^{\text{EST-CLOS}}$: dans $\mathrm{coNP}^{\mathrm{coNP}} = \Pi_2^{\mathrm{P}}$ dans le cas général et dans coNP dans le cas localement monotone.

Théorème 2.3. *(Paulevé et al. 2020) Déterminer si une configuration $x \in \mathbb{B}^n$ appartient à un attracteur de f avec la sémantique MP est au plus dans coNP si f est localement monotone et au plus dans Π_2^P sinon.*

Le calcul des hypercubes minimaux d'un réseau booléen localement monotone (et donc des attracteurs de la sémantique MP) peut s'effectuer efficacement en *answer set programming* (ASP) qui intègre la notion de solution minimale par inclusion, et est applicable à des réseaux de très grande échelle (Paulevé et al. 2020).

2.5 Validation sur des modèles de réseaux biologiques

Une caractéristique essentielle des modèles dits "logiques" est leur capacité à conclure sur l'*absence* de certains comportements. Par exemple, les processus de différenciation cellulaire sont modélisés par des attracteurs distincts représentant les phénotypes, avec des configurations

ne pouvant atteindre qu'un sous-ensemble des attracteurs (branche de différenciation). Une sémantique autorisant trop de comportements possible serait alors inutile pour de tels modèles.

Nous montrons ici sur des cas concrets de modèles de réseaux biologiques, que la sémantique MP reste restrictive et permet de prédire les caractéristiques majeures des processus de différenciation, notamment lié à l'absence d'accessibilité d'attracteurs particuliers. De plus, nous illustrons le gain pratique de la faible complexité de la sémantique MP.

Le détail des analyses peut être visualisé à doi:10.5281/zenodo.3936123. Les temps de calcul reportés ont été mesurés sur un processeur Intel(R) Xeon(R) E-2124 cadencé à 3.30GHz et effectués avec l'outil `mpbn`[1], basé sur Python et reposant sur une implémentation de la sémantique MP en ASP avec le solveur `clingo` (Gebser, Kaminski, Kaufmann, & Schaub 2014).

Modèle de l'invasion tumorale par Cohen et al. (2015) Réseau booléen de 32 composants qui modélise les processus de décision cellulaire impliqués dans l'invasion tumorale, avec des attracteurs liés à l'apoptose, arrêt du cycle cellulaire, et différentes étapes de la formation des métastases. Nous avons pu reproduire l'analyse des attracteurs accessibles à la fois dans le modèle initial, et dans le modèle présentant les mutations de p53 et Notch, dont la combinaison entraîne la perte d'accessibilité des attracteurs apoptotiques. L'analyse avec la sémantique MP prédit exactement les mêmes attracteurs accessibles et leur s'effectue en moins de 10ms.

Différentiation des cellules T par Abou-Jaoudé et al. (2015) Ce réseau multivalué de 102 composants modélise les capacités de reprogrammation entre les différentes sous-types cellulaires des lymphocytes T. Avec une "booleanization" du modèle (Didier, Remy, & Chaouiya 2011), nous avons notamment pu reproduire le calcul du *graphe de reprogrammation* entre les différents types étudiés avec les conditions d'entrées identifiées par Abou-Jaoudé et al. (2015). En utilisant la sémantique MP, nous retrouvons exactement le même graphe qu'avec la sémantique pleinement asynchrone. De plus, les calculs ont été effectués sur le réseau de 102 composants, alors que l'étude originale a eu recours à des approximations sur un modèle réduit pour pouvoir conduire les analyses. Le calcul des attracteurs accessibles s'effectue en moins de 100ms.

Tumorigenèse de la vessie par Remy et al. (2015) Nous avons reproduit les calculs d'attracteurs accessibles dans ce modèle de 35 composants en suivant l'étude de Mendes et al. (2018) sur la comparaison de différentes méthodes de simulation. Avec la sémantique MP, les mêmes ensembles d'attracteurs accessibles sont prédits. De plus, leur calcul a été effectué en moins de 10ms, alors que les méthodes de simulations de la sémantique pleinement asynchrone nécessitent plusieurs secondes voire dizaines de minutes, et ne sont pas toujours concluantes. Il est à noter cependant, que contrairement aux méthodes de simulations, nous ne sommes pas encore capables de quantifier la prépondérance des différents attracteurs accessibles. Néanmoins, leur énumération est complète.

1. https://github.com/pauleve/mpbn

Très grands réseaux aléatoires Nous avons généré aléatoirement des graphes d'influence avec une structure *scale-free* (Albert & Barabási 2002), comprenant entre 1 000 et 100 000 composants avec des degrés entrant jusqu'à 1 400 (Paulevé 2020). Nous avons appliqué la règle de l'"inhibiteur dominant" pour obtenir une fonction booléenne pour chaque composant : l'activation nécessite l'absence de tout inhibiteur actif, et au moins un activateur actif, s'il en possède au moins un. Les calculs suivants ont été mesurés : calcul d'un attracteur ; énumération d'au plus 1 000 attracteurs ; et énumération d'au plus 1 000 attracteurs accessibles depuis une configuration initiale aléatoire. La figure. 2.6 résume les résultats et montre que le calcul des attracteurs accessibles prend une faction de seconde avec 1 000 composants, moins de 2 secondes avec 10 000, et moins de 50 secondes avec 100 000 composants. Les temps de calcul excluent le temps de chargement du modèle (jusqu'à 20 secondes pour les réseaux étudiés).

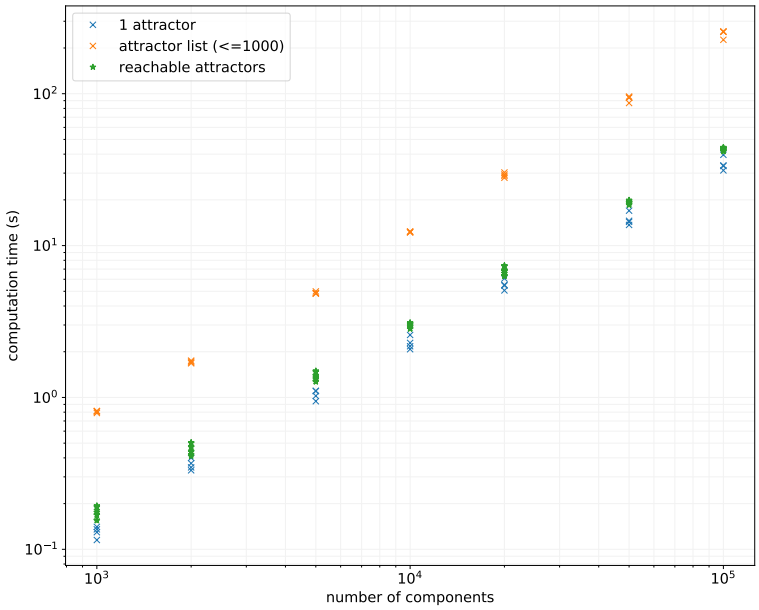


FIGURE 2.6 – Temps de calcul avec un processeur 3.3GHz en utilisant l'outil `mpbn` pour le calcul des attracteurs accessibles en sémantique MP sur des réseaux booléens aléatoires.

2.6 Discussion

Le choix de la sémantique peut avoir un effet drastique sur les prédictions engendrées. La sémantique MP permet une analyse avec le moins d'hypothèses possibles sur le système sous-jacent, hypothèses souvent utilisées pour des raisons techniques et sans lien avec les connaissances sur le système biologique modélisé. Il est commun de lire que la sémantique pleine-

ment asynchrone est la plus réaliste biologiquement souvent avec un simple argument d'autorité et l'intuition que cela permet de capturer différentes échelles temporelles dans les influences. Les exemples abordés dans ce chapitre soulignent que la mise à jour pleinement asynchrone (tout comme l'asynchrone) peut, de manière contre-intuitive, masquer certains comportements pourtant pertinents biologiquement. La sémantique MP permet de retrouver ces comportements sans avoir à "agrandir" l'espace des états, ce qui nécessiterait en général de déterminer de nombreux paramètres supplémentaires, au prix de potentiels comportements artificiels. De plus, nous avons pu montrer que la sémantique MP est minimale pour la classe de raffinement étudiée.

L'explosion du nombre de configurations à explorer avec les sémantiques classiques des réseaux booléens est une limite importante pour leur application en biologie des systèmes (Bornholdt 2008, Le Novère 2015). La sémantique MP offre ici un gain considérable de complexité pour l'analyse des propriétés d'accessibilité et d'attracteurs, ouvrant la possibilité à l'analyse de modèles à l'échelle du génome.

Je détaillerai dans le chapitre 6 différentes pistes de recherche en lien avec cette sémantique MP, notamment autour de différentes classes de raffinement qui pourraient être considérées (et pour laquelle la MP n'est peut-être pas minimale) et de la quantification des attracteurs accessibles.

Chapitre 3

Ensembles de réseaux booléens : synthèse et sémantiques

La construction de réseaux booléens à partir de connaissances et données expérimentales pose la question de l'unicité des modèles. Ce chapitre aborde la modélisation par *ensembles* de modèles, partageant certaines propriétés demandées, mais montrant une potentielle variabilité dans leur comportement. La section 3.1 résume des travaux liés à la synthèse d'ensembles de réseaux booléens avec sémantique MP à partir de contraintes sur leur architecture et sur leur dynamique ; travaux de thèse de doctorat de Stéphanie Chevalier. La section 3.2 résume des travaux liés à la sémantique d'ensembles de modèles, permettant de raisonner conjointement sur la dynamique de plusieurs modèles, travaux de thèses de doctorat de Juraj Kolčák.

Pour la majorité des applications en biologie, la connaissance actuelle ne permet pas de spécifier complètement un réseau booléen. En pratique, les modélisations partent souvent du graphe d'influence. Ce dernier rassemble toutes les interactions binaires et dirigées découvertes expérimentalement entre les composants choisis du système. Mais les fonctions booléennes sous-jacentes sont rarement connues. Par exemple, dans le cas où il serait connu que deux composants influencent directement et positivement l'état d'un troisième, il est rarement connu si l'influence est additive (disjonction) ou multiplicative (conjonction). Pour parfaire l'incertitude du modélisateur, le graphe d'influence peut être incomplet (interactions inconnues), et mélange souvent des interactions découvertes dans des types cellulaires et des conditions expérimentales très différents (souris, humain…).

En pratique, on observe que le problème de conception de réseaux booléens est souvent largement sous-spécifié, et peut aboutir à un nombre astronomique de solutions, toutes équivalentes vis-à-vis de leur adéquation aux contraintes données. De plus, on peut aisément supposer une variabilité naturelle de la logique des régulations au sein d'une population de cellule.

Ainsi, l'étude qualitative de la dynamique de systèmes biologiques nécessiterait d'opérer

non pas sur un modèle unique, mais sur une famille, un ensemble de modèles, partageant certaines propriétés. Dans ce chapitre, je résume quelques contributions allant dans cette direction, autour de la synthèse (section 3.1) et de sémantiques abstraites (section 3.2) d'ensemble de modèles.

3.1 Synthèse de réseaux booléens MP

La *synthèse* de réseaux booléens consiste à construire automatiquement des réseaux dont l'architecture et la dynamique vérifient des propriétés fixées *a priori*. Les contraintes sur l'architecture proviennent généralement de bases de données référençant les interactions potentielles entre les gènes et les protéines sélectionnés, ou de méthodes statistiques pour suggérer des interactions possibles à partir d'observations du système. Les contraintes sur la dynamique proviennent le plus souvent de l'interprétation de données biologiques montrant des comportements temporels (séries temporelles, donnant des propriétés d'accessibilité) et de différenciation (propriétés d'attracteurs et d'absence de trajectoires). Ainsi, à la complexité d'analyse d'un réseau booléen fixé s'ajoute la combinatoire des réseaux booléens potentiels à explorer.

Grâce à la faible complexité de l'analyse de la sémantique MP, nous avons récemment démontré que la synthèse de réseaux booléens peut s'effectuer à très grande échelle pour certaines familles de propriétés dynamiques, tout en ayant des garanties importantes vis-à-vis de modèles issus de raffinements quantitatifs (Chevalier et al. 2019). Sur des réseaux où chaque composant est régulé par moins de quinze composants, notre approche, basée sur la programmation logique en *answer set programming* (ASP), permet d'énumérer *toutes les solutions possibles, sans redondance*, grâce à un encodage spécifique des réseaux booléens *localement monotones*. Pour des réseaux d'échelle supérieure, notre approche permet de faire de l'échantillonnage de l'espace des solutions, dont la complexité peut être paramétrée. Notre approche permet d'aborder des applications avec plus de 1 000 composants, dont l'architecture correspond à des exportations complètes de bases de données d'interactions.

Dans la littérature, la synthèse de réseaux booléens à partir de propriétés d'architecture et de dynamique est résolue soit par des méthodes d'optimisation, soit par des méthodes de satisfaction de contraintes. Les méthodes de la première catégorie (p.ex., Dorier et al. 2016, Terfve et al. 2012), couplent des algorithmes génétiques pour explorer l'espace des modèles avec des simulations pour évaluer (par approximation) les propriétés dynamiques. En pratique, ces méthodes fonctionnent sur des réseaux entre 20 et 40 composants. Elles ne garantissent pas de terminer, ni de trouver un modèle globalement optimal ; le critère d'optimalité étant par ailleurs composé de nombreux paramètres. L'énumération de solutions optimales est également peu pratiquable, rendant difficile l'accès à la diversité des solutions. Les méthodes de satisfaction reposent sur la programmation logique (Ostrowski, Paulevé, Schaub, Siegel, & Guziolowski 2016) et SMT (Yordanov et al. 2016) pour exprimer l'ensemble des contraintes sur les modèles attendus. Ces approchent rendent possible une énumération des solutions, avec la possibilité d'ajouter des critères d'optimisation. Les travaux de Yordanov et al. (2016) considèrent des propriétés d'accessibilité positive et des propriétés de points fixes, mais seul un sous-espace des modèles candidats est exploré, et permettent d'aborder des réseaux entre 20 et 40 composants avec la sémantique synchrone.

Dans le cadre d'une collaboration avec Anne Siegel (IRISA, Rennes), Carito Guziolowsky (LS2N, Nantes) et Max Ostrowski (Univ Potsdam, Allemagne), nous avons mis au point une méthode pour la synthèse à partir de propriétés positives d'accessibilité, basée sur ASP et sur un filtrage des solutions par model-checking avec la sémantique asynchrone ou pleinement asynchrone.

Partant de ce travail, et dans le cadre de la thèse de doctorat de Stéphanie Chevalier, nous avons revu l'encodage ASP pour permettre d'aborder des réseaux de plus grande taille, et implémenter la vérification de propriétés dynamiques supplémentaires, en particulier sur l'absence d'accessibilité et les attracteurs, avec la sémantique MP.

3.1.1 Contraintes traitées

Les propriétés dynamiques traitées sont inspirées par la modélisation des processus de différenciation cellulaire. Partant d'un état pluripotent (souche), et en fonction de différents stimulus ou changements dans leur environnement, les cellules se spécialisent progressivement, en passant par différents états stables intermédiaires. Les techniques expérimentales permettent de mesurer l'activité partielle ou complète des gènes à différents instants. De ces observations peuvent alors être déduites des propriétés d'accessibilité pour reproduire les changements d'états observés, mais également des propriétés d'attracteur quand les observations sont effectuées sur des cellules stabilisées. De plus, les *branches* de différenciation impliquent aussi des propriétés négatives d'accessibilité : une cellule observée dans une certaine branche ne peut pas atteindre un état stable des autres branches. Ces contraintes peuvent provenir de connaissances expertes sur le système, mais également d'analyses statistiques, en particulier avec les données de séquençage à l'échelle de la cellule (sc-RNASeq) (Chen et al. 2019).

L'ensemble des réseaux booléens candidats est quant à lui délimité par un graphe d'influence donné en entrée. Ce graphe décrit les influences directes, dirigées et signées, entre des paires de composants. Ces graphes sont généralement extraits de bases de données, et peuvent être complétés par des analyses statistiques à partir des données. La fonction booléenne d'un composant ne peut alors dépendre (au plus) que des composants ayant une influence sur lui, avec le signe adéquat. Dans ce travail, nous ne considérons que la synthèse des réseaux booléens localement monotones : si le graphe d'influence spécifie plusieurs signes possibles pour une interaction, une seule au maximum sera effectivement utilisée dans une solution.

Ci-après, nous décrivons formellement le problème de synthèse abordé. La figure 3.1 illustre le type de propriété dynamique exprimable avec notre approche.

Une observation (partielle) o d'une configuration de dimension n est spécifiée par un ensemble de couples associant un composant à une valeur booléenne : $o \subseteq \{1,\dots,n\} \times \mathbb{B}$, et tel qu'il n'y a aucun composant $i \in \{1,\dots,n\}$ avec $\{(i,0),(i,1)\} \subseteq o$.
Étant donnés

— un graphe dirigé et signé $\mathcal{G} = (\{1,\dots,n\}, E_+ \cup E_-)$,
— p observations partielles $o^1,\dots,o^p$,
— des ensembles PR et NR de couples d'indices d'observations : $\mathsf{PR},\mathsf{NR} \subseteq \{1,\dots,p\}^2$,

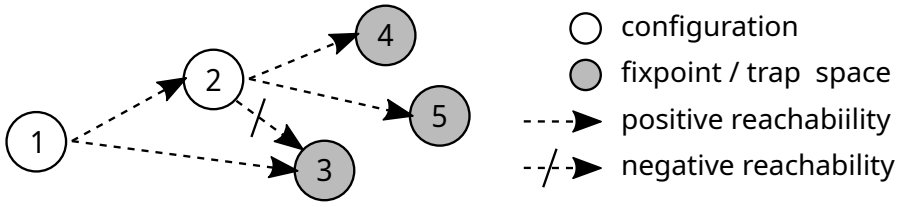


FIGURE 3.1 – (extraite de Chevalier et al. 2019) Illustration des contraintes dynamiques pour la synthèse de réseaux booléens MP

— un ensemble FP d'indices d'observations :
  $\mathsf{FP} \subseteq \{1, \ldots, p\}$,
— un ensemble TP associant des indices d'observations avec des composants :
  $\mathsf{TP} \subseteq \{1, \ldots, p\} \times \{1, \ldots, n\}$,

le problème de synthèse consiste à trouver un réseau booléen f de dimension n tel que

— $G(f) \subseteq \mathcal{G}$, et
— il existe p configurations $x^1, \ldots, x^p$ telles que :
  — $\forall m \in \{1, \ldots, p\}, \forall (i, v) \in o^m, x_i^m = v$ (compatibilité des observations),
  — $\forall (m, m') \in \mathsf{PR}, x^{m'} \in \rho^f_{\mathrm{mp}}(x^m)$ (accessibilité positive),
  — $\forall (m, m') \in \mathsf{NR}, x^{m'} \notin \rho^f_{\mathrm{mp}}(x^m)$ (accessibilité négative),
  — $\forall m \in \mathsf{FP}, f(x^m) = x^m$ (points fixes),
  — $\forall (m, i) \in \mathsf{TP}$, soit $t \in (\mathbb{B} \cup \{*\})^n$ le plus petit hypercube clos par f contenant x^m, alors $t_i = x_i^m$ (contrainte de *trap space*).

où $G(f) = (\{1, \ldots, n\}, E^f_+ \cup E^f_-)$ avec $(j, i) \in E^f_+$ (resp. $(j, i) \in E^f_-$) si et seulement si $\exists x, y \in \mathbb{B}^n$ s.t. $\Delta(x, y) = \{j\}$, $x_j < y_j$, et $f_i(x) < f_i(y)$ (resp. $f_i(x) > f_i(y)$).

La contrainte de *trap space* revient à imposer l'existence d'un attracteur (potentiellement cyclique) où un ensemble des composantes spécifiées restent fixées à une valeur donnée.

Ce problème peut être insatisfiable selon le graphe d'influence et les propriétés dynamiques. Au-delà de l'existence d'une solution, nous nous intéressons également à la caractérisation complète et non-redondante de l'ensemble des solutions. La complétude est possible car il n'existe qu'un nombre fini de réseaux booléens f tels que $G(f) \subseteq \mathcal{G}$. La non-redondance des solutions implique d'énumérer seulement des réseaux booléen non-équivalents, c'est-à-dire dont la valeur diffère au moins en une configuration.

3.1.2 Modélisation en answer set programming (ASP)

La programmation par ensemble-réponse, *answer set programming* (ASP ; Baral 2003, Gebser, Kaminski, Kaufmann, & Schaub 2012, Lee, Lifschitz, & Palla 2008)) est une approche déclarative pour la résolution de problèmes combinatoires de satisfaction. Proche de la résolution SAT (Lin & Zhao 2004), les solveurs ASP sont connus pour être très efficaces pour l'énumération de solutions de problèmes NP contenant plusieurs dizaines de millions de variables. Nous donnons ici un bref aperçu d'un programme ASP pour en montrer ses particularités.

Un programme ASP est un ensemble de règles logiques exprimées en logique du premier ordre avec la forme suivante :

a_0 `:-` a_1`,` `...,` a_n`,` `not` a_{n+1}`,` `...,` `not` a_{n+k}`.`

où a_i sont des atomes (non paramétrés).

Une telle règle établit que si tous les $a_1, \ldots, a_n$ sont vrais, et aucun des $a_{n+1}, \ldots, a_{n+k}$ ne peut être dérivé comme vrai, alors a_0 est vrai. Si a_0 est $\bot$ (faux), la règle, appelée contrainte d'intégrité, s'écrit :

`:-` a_1`,` `...,` a_n`,` `not` a_{n+1}`,` `...,` `not` a_{n+k}`.`

Cette règle est satisfaite seulement si la partie droite est fausse (au moins un $a_1, \ldots, a_n$ est faux, ou au moins un $a_{n+1}, \ldots, a_{n+k}$ est vrai). D'un autre côté a_0 `:-` $\top$ (a_0 est toujours vrai) est abrégé en a_0, appelé un *fait*. Une solution (answer set) est un modèle de Herbrand stable, c'est-à-dire, un ensemble où tous les atomes sont soit des faits, soit ont été dérivés vrais par l'application d'une règle.

Par exemple, le programme suivant a pour unique solution `{a, b}` :

```
a.  % fact
b :- a.  % a implies b
d :- a, c. % (a&c) implies d
```

Le langage permet d'utiliser des variables, ou plus exactement des *templates*, qui seront développés en atomes simples avant la résolution (étape de *grounding*). Le langage permet également de spécifier des contraintes de cardinalité (par exemple n `{a(X): b(X)}` m est vrai si au moins n et au plus m atomes `a(X)` sont vrais, où `X` varie parmi les `b(X)` vrais) et autres contraintes portant sur des ensembles d'atomes, dont la spécification d'objectifs d'optimisation.

La notion de modèle stable permet de faciliter grandement l'écriture de contraintes comparé à SAT. Il est à noter qu'il est possible de traduire un large fragment d'ASP en SAT, via l'ajout de contraintes pour éliminer les modèles non-stables, c'est-à-dire comportant des atomes vrais qui ne sont ni des faits, ni dérivés d'une règle (Lin & Zhao 2004).

Encodage du problème de synthèse

L'encodage en ASP des réseaux booléens localement monotones compatibles avec un graphe d'influence pose deux difficultés : deux solutions distinctes doivent décrire deux réseaux non-équivalents f et f', c'est-à-dire telles qu'il existe $x \in \mathbb{B}^n$ avec $f(x) \neq f'(x)$. Ceci nécessite d'employer une forme canonique pour leur modélisation en ASP. Ensuite, la taille de la spécification d'une fonction booléenne peut être exponentielle par rapport au nombre de ses variables. Ainsi, pour pouvoir passer à l'échelle, l'encodage doit permettre de borner la taille d'une fonction, si possible sans borner son nombre de variables.

Nous représentons les fonctions booléennes par leur forme normale disjonctive (DNF), c'est-à-dire par des ensembles de clauses conjonctives, où les clauses sont des ensembles de littéraux, et deux clauses distinctes n'ont pas de relation de sous-ensemble (anti-chaîne). En ASP, nous devons encoder les ensembles par des listes, et donc donner un identifiant à chaque

clause. La canonicité est alors assurée en imposant un ordre total entre les clauses. Le nombre maximum de clauses d'une DNF avec d variables est $\binom{d}{\lfloor d/2 \rfloor}$, et une clause contient au plus d littéraux (monotonie). Ainsi, toute DNF avec d variables peut s'écrire à l'aide de $d\binom{d}{\lfloor d/2 \rfloor}$ atomes ASP. Au total, en tenant compte des contraintes pour assurer la canonicité, notre encodage génère $O(ndk^2)$ atomes et $O(nd^2k^2)$ règles où k est la borne sur le nombre de clauses (par défaut $\binom{d}{\lfloor d/2 \rfloor}$). Avec la valeur maximum, le nombre de solutions correspond au nombre de fonctions booléennes monotones, le *nombre de Dedekind* (Kleitman 1969), actuellement connu jusqu'à $d = 8$ (Wiedemann 1991) [1]. Quand k est plus petit, certaines fonctions booléennes ne sont pas capturées par notre encodage. Les contraintes pour obtenir la canonicité sont nécessaires pour une énumération efficace des solutions. Toutefois, cette contrainte peut être relâchée, en particulier lorsque l'on cherche simplement l'existence d'une solution (possiblement avec un critère d'optimisation). Alors, le nombre d'atomes et de règles se réduit en $O(ndk)$.

Concernant les propriétés dynamiques, l'encodage repose sur la définition de la sémantique MP avec les hypercubes (section 2.2.2). Les contraintes sur l'existence de trajectoires, de points fixes, et de trap spaces génèrent un nombre d'atomes et de règles en $O(ndk)$; d étant le degré entrant des composants dans le graphe d'influence. La contrainte d'absence de trajectoire est quadratique avec n ; toutefois elle redevient linéaire lorsque la configuration cible est dans un trap space (Paulevé et al. 2020).

L'encodage de l'ensemble des contraintes pour la synthèse de réseaux booléens MP est détaillé dans (Chevalier et al. 2019).

ASP permet également d'exprimer des problèmes de classe supérieure à SAT avec la programmation disjonctive (Eiter & Gottlob 1995), pour la résolution de problèmes 2QBF ($\mathrm{NP}^{\mathrm{NP}}$ aussi dénommée Σ_2^{P}), c'est-à-dire de la forme "$\exists x, \forall y : P(x, y)$" (SAT se cantonnant à "$\exists x : P(x)$"). L'emploi de telles contraintes permettrait de spécifier des contraintes universelles sur les points fixes ou attracteurs du réseau, et fait l'objet de travaux en cours (Chevalier, Noël, Calzone, Zinovyev, & Paulevé 2020).

3.1.3 Mise en pratique

Passage à l'échelle

Le facteur limitant principal est le degré entrant des composants du réseau. Sur une machine avec 64Go de RAM, nous avons pu résoudre des instances aléatoires avec l'encodage canonique et sans borne sur le nombre de clauses avec un degré entrant de 15 (Chevalier et al. 2019).

En bornant le nombre de clauses ou en relâchant la contrainte de canonicité, nous pouvons aller jusqu'à la synthèse de réseaux de plus de 1 000 composants avec des degrés entrants de l'ordre de 200 (en particulier pour des applications réelles, sur des extractions complètes de bases de données d'interaction, voir la section 4.2).

1. pour $0 \leq d \leq 8$: 2, 3, 6, 20, 168, 7581, 7828354, 2414682040998, 5613043722868755790788

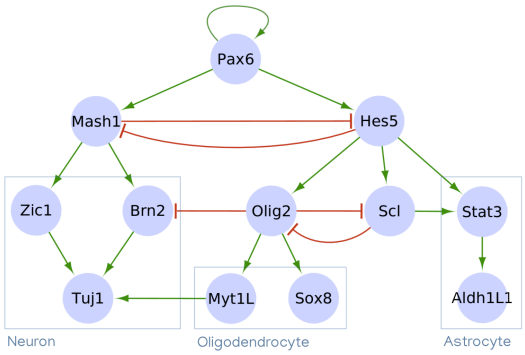


FIGURE 3.2 – Graphe d'influence pour le modèle de différenciation neuronale

obs. ID	gènes actifs	gènes inactifs
0	*none*	*all*
iPax6	Pax6	*the 11 others*
tM	Pax6	Aldh1L1, Olig2, Scl, Sox8, Tuj1
fT	Brn2, Tuj1, Zic1	Aldh1L1, Sox8
tO	Olig2, Pax6	Aldh1L1, Scl, Sox8, Tuj1
fMS	Sox8	Aldh1L1, Brn2, Tuj1, Zic1
tS	Pax6, Scl	Aldh1L1, Olig2, Sox8, Tuj1
fA	Aldh1L1	Brn2, Sox8, Tuj1, Zic1

TABLE 3.1 – Observations à différentes étapes de la différenciation cellulaire

Comptage de modèles

Une question récurrente pour la synthèse de modèle est l'obtention de données *contraignantes*, qui vont réduire considérablement le nombre de modèles compatibles. Notre encodage permettant une identification complète des solutions, nous pouvons employer notre approche pour quantifier la capacité contraignante d'une propriété dynamique.

Nous l'illustrons ici sur un modèle de différenciation cellulaire lors du développement du système nerveux central par Qiu et al. (2017). Le modèle décrit la différenciation des cellules souches neuronales en neurones, astrocytes et oligodendrocytes, avec le graphe d'influence reproduit dans la figure 3.2. Plus de 226 millions de réseaux booléens localement monotones sont compatibles avec ce graphe (produit des nombres de Dedekind pour chaque nœud).

À l'aide de notre implémentation, nous avons alors compté le nombre de modèles vérifiant tout ou une partie des contraintes dynamiques du processus de différenciation. Les observations du système sont reportées dans la table 3.1. La stabilité des phénotypes peut se spécifier soit par des contraintes de points fixes ($\mathsf{FP} = \{fMS, fA\}$) ; soit par des contraintes de trap spaces en fixant les 4 marqueurs phénotypiques Aldh1L1, Myt1L, Sox8 et Tuj1. Les contraintes d'accessibilité positives et négatives sont $\mathsf{PR} = \{(iPax6, tM), (tM, fT), (iPax6, tO), (tO, fMS), (iPax6, tS), (tS, fA)\}$, $\mathsf{NR} = \{(0, fT), (0, fMS), (0, fA)\}$.

Le nombre de modèles obtenus avec différentes combinaisons de contraintes est donné

contraintes	**nb. solutions**
type unique de contraintes	
3 negative reachability *(NR)*	224 025 280
6 positive reachability *(PR)*	24 076 416
12 trap spaces *(TP)*	17 220
3 fixpoints *(FP)*	4970
combinaisons de contraintes	
PR et *NR*	16 050 944
PR et *TP*	8964
NR et *TP*	5667
PR et *NR* et *TP*	3735
PR et *FP*	3360
PR et *NR* et *FP*	1120
PR et *NR* et *TP* et *FP*	1120

TABLE 3.2 – Nombre de solutions pour la synthèse de réseaux booléens MP en fonction des propriétés dynamiques sélectionnées

par la table 3.2 Nous observons ainsi une forte réduction du nombre de modèles vérifiant l'ensemble des contraintes dynamiques, et nous pouvons quantifier l'apport de chaque type de contrainte.

Échantillonnage

Lors d'applications sur des réseaux de grande taille, il est impossible d'énumérer toutes les solutions. Deux pistes peuvent alors être suivies : projeter les solutions composant par composant, et échantillonner l'espace des solutions.

Lorsque l'on demande à un solveur de n'énumérer qu'une fraction des solutions, il est très probable que les solutions renvoyées soient toutes très similaires, avec par exemple un seul composant qui change de fonction. Cependant, il est possible de forcer le solveur à chercher des solutions diverses en truquant ses heuristiques pour l'assignation de variables (Razzaq, Kaminski, et al. 2018). Cependant nous ne disposons pas actuellement de critère objectif pour quantifier la diversité d'un ensemble de solutions obtenu. J'aborderai dans le chapitre 6 différentes perspectives relatives à un tel échantillonnage.

3.2 Abstractions et sémantiques d'ensembles de modèles

L'approche présentée dans la section précédente traite chaque modèle individuellement. Une autre piste de recherche est de définir des sémantiques qui opèrent directement sur des ensembles de modèles, sans les énumérer. Ces travaux sont au cœur de la thèse de doctorat de Juraj Kolčák et font l'objet d'une collaboration avec David Safranek (Univ Masaryk, Tchéquie) ; je résume ici les principales lignes directrices des travaux publiés dans (Haar, Kolčák, & Paulevé 2019, Kolčák et al. 2018).

Le cadre de modélisation est ici un peu plus général que les réseaux booléens, puisque nous considérons également les réseaux multivalués. De même, le formalisme pour leur spé-

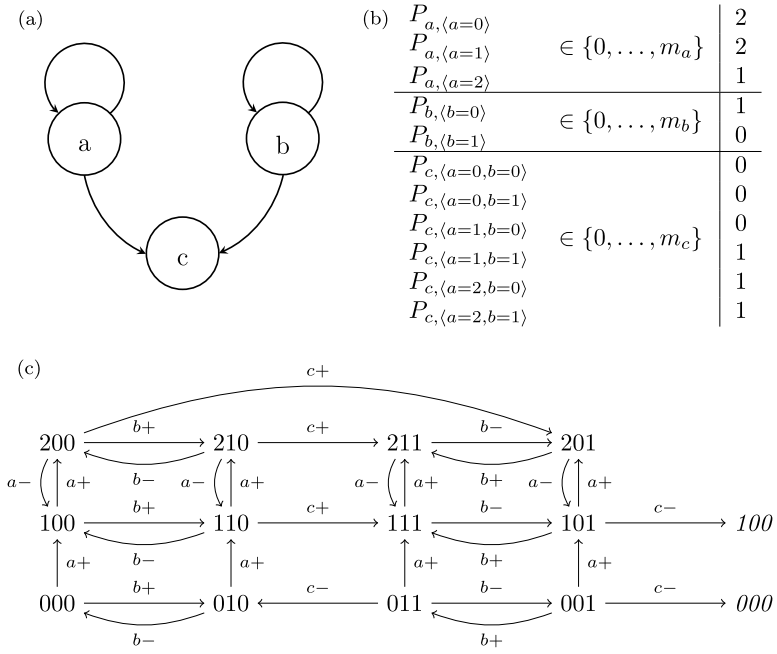


FIGURE 3.3 – Exemple de réseaux de régulation paramétré. (a) graphe d'influence non-signé ; (b) un paramétrage possible avec $m_a = 2, m_b = 1, m_c = 1$; (c) transitions entre les configurations du réseaux avec la sémantique pleinement asynchrone ; l'ordre des composantes est a, b, c.

cification diffère de celui utilisé dans le chapitre précédent (section 2.3.1, pour se rapprocher du formalisme de Thieffry et Thomas (1995) basé sur les *paramètres* délimités par un graphe d'influence.

Le principe est illustré par la figure 3.3. Un modèle est ici la donnée d'un graphe d'influence (non signé) qui liste les influences dirigées entre les composants du réseau. Les paramètres spécifient alors l'état vers lequel tend chaque composant en fonction de l'état de ses régulateurs. Ces paramètres peuvent être vus comme la table de vérité des fonctions locales des composants. Ensuite, de manière similaire aux réseaux booléens tels qu'introduits dans le chapitre 1, plusieurs sémantiques peuvent être définies pour calculer les successeurs d'une configuration du réseau ; avec généralement l'hypothèse que le changement de valeur d'un composant est unitaire : si le composant est à 0 et que sa valeur cible est 2, il doit d'abord passer par l'état 1.

3.2.1 Réseaux de régulation paramétriques

Dans la suite de cette section, nous considérons un graphe d'influence comme un simple graphe dirigé $G = (V, I)$ avec V un ensemble fini de sommets et $I \subseteq V^2$ ses arcs (influences). Étant donné un sommet $v \in V$, on note $n^-(v) := \{u \in V | (u, v) \in I\}$ ses prédécesseurs.

En sus du graphe, des contraintes sur les influences peuvent être posées :

— la monotonie positive (resp. négative) d'une influence u, v, notée + (resp. −) : l'augmentation seule de u ne peut jamais causer une décroissance (resp. croissance) de v ;

— l'observabilité d'une influence u, v, notée o : il existe deux configurations qui diffèrent seulement par u et dont la valeur cible de v diffère également.

On note $R \subseteq I \times \{+, -, o\}$ l'ensemble de ces contraintes.

Fixons maintenant pour chaque sommet v une valeur maximale $m_v \in \mathbb{N}$. Les configurations du réseau sont alors l'ensemble $\prod_{v \in V} \{0, \ldots, m_v\}$ [2].

La valeur cible d'un sommet $v \in V$ ne dépend que de ses régulateurs $n^-(v)$. On note $\Omega_v := \prod_{u \in n^-(v)} \{0, \ldots, m_u\}$ l'ensemble des états des régulateurs. Un *paramètre* $\langle v, \omega \rangle$ représente alors la valeur cible de v quand ses régulateurs sont dans l'état $\omega \in \Omega_v$. L'ensemble des paramètres d'un réseau de régulation est donné par $\Omega := \bigcup_{v \in V} \{v\} \times \Omega_v$. Une *paramétrisation* P assigne à chaque paramètre une valeur.

Définition 3.1. (Kolčák et al. 2018) On appelle *réseau de régulation paramétrique* (PRN ; *Parametric Regulatory Network*) le triplet composé d'un graphe d'influence entre n sommets $G = (V, I)$ avec les valeurs maximales $m \in \mathbb{N}^n$ et les contraintes $R \subseteq (I \times \{+, -, o\})$, que l'on note G_m^R. L'ensemble de toutes ses paramétrisations possibles peut se définir comme

$$\begin{aligned}\mathbb{P}(G_m^R) := \{P \in \textstyle\prod_{\langle v,\omega\rangle \in \Omega} \{0, \ldots, m_v\} \mid \forall u, v \in V,\\ (u, v, +) \in R \Rightarrow \forall \omega \in \Omega_v, \forall k \in \{1, \ldots, m_u\} : P_{v,\omega_{[u \to k]}} \geq P_{v,\omega_{[u \to k-1]}}\\ (u, v, -) \in R \Rightarrow \forall \omega \in \Omega_v, \forall k \in \{1, \ldots, m_u\} : P_{v,\omega_{[u \to k]}} \leq P_{v,\omega_{[u \to k-1]}}\\ (u, v, o) \in R \Rightarrow \exists \omega \in \Omega_v, \forall k \in \{1, \ldots, m_u\} : P_{v,\omega_{[u \to k]}} \neq P_{v,\omega_{[u \to k-1]}}\}\end{aligned}$$

où $\omega_{[u \to k]}$ est le vecteur égal à ω sauf pour la composante u égale à k.

3.2.2 Sémantiques d'ensembles de paramétrisations

L'ensemble de tous les changements de valeurs (unitaires) des composants pour chacun des états possibles de leurs régulateurs est noté $\Delta(G_m^R) := \{(v_i \to v_j, \omega) \mid v \in V, \omega \in \Omega_v, i, j \in \{0, \ldots, m_v\}, |i - j| = 1\}$. Un changement $t = (v_i \to v_j, \omega) \in \Delta(G_m^R)$ est *possible* dans une configuration x du réseau si $x_v = i$ et $\omega_v(x) = \omega$, où $\omega_v(x)$ est la projection de la configuration x sur les composantes $n^-(v)$. Le changement t est *permis* par une paramétrisation $P \in \mathbb{P}(G_m^R)$ si $P_{v,\omega} \geq j$ si $j > i$ et $P_{v,\omega} \leq j$ si $j < i$. On note alors $x \cdot t := x_{[v \to j]}$ la configuration obtenue par l'application de ce changement.

Nous pouvons définir la notion de trace avec la sémantique purement asynchrone d'un PRN couplé avec un ensemble de paramétrisations comme suit :

2. Nous faisons l'hypothèse que les produits cartésiens $\prod$ sont effectués en suivant un ordre total fixé.

Définition 3.2. Étant donné un PRN G_m^R et un ensemble de paramétrisations $\mathscr{P} \subseteq \mathbb{P}(G_m^R)$, une *trace* de G_m^R depuis la configuration $x \in \prod_{v \in V}\{0, \ldots, m_v\}$ est une séquence finie $\pi = (\pi_1, \ldots, \pi_{|\pi|})$ de changements dans $\Delta(G_m^R)$ telle que pour tout $i \in \{1, \ldots, |\pi|\}$, le changement π est possible dans la configuration $x \cdot \ldots \cdot \pi_{i-1}$ et est permis par au moins une paramétrisation $P \in \mathscr{P}$.

Il est important de remarquer que cette définition autorise des traces qui sont impossibles à réaliser avec une seule paramétrisation.

L'idée est alors de définir une sémantique qui, partant de l'ensemble complet des paramétrisations d'un PRN G_m^R, va progressivement raffiner cet ensemble au fur et à mesure des changements de valeurs effectués. En effet, si un changement est permis par une paramétrisation, de nombreuses paramétrisations peuvent ne pas être compatibles avec ce changement (elles ne le permettent pas).

La définition suivante délimite de telles sémantiques, en garantissant leur complétude et leur cohérence, et autorise de potentielles abstractions.

Définition 3.3. (Haar et al. 2019) Étant donné un PRN G_m^R, une *sémantique d'ensemble de paramétrisations* est une fonction $\Psi : 2^{\Delta(G_m^R)} \to 2^{\mathbb{P}(G_m^R)}$ qui vérifie les conditions suivantes :

1. $\forall T \subseteq \Delta(G_m^R), \{P \in \mathbb{P}(G_m^R) \mid \forall t \in T : t \text{ est permis par } P\} \subseteq \Psi(T)$
2. $\forall T, T' \subseteq \Delta(G_m^R), T \subseteq T' \Rightarrow \Psi(T') \subseteq \Psi(T)$.

Une trace π est réalisable avec la sémantique Ψ si et seulement si $\Psi(\tilde{\pi}) \neq \emptyset$, avec $\tilde{\pi}$ l'ensemble des changements composant la séquence π.

Une sémantique concrète vérifiant ces propriétés peut se définir comme suit :

Définition 3.4 (Sémantique concrète)**.** (Kolčák et al. 2018) $\Psi^C : 2^{\Delta(G_m^R)} \to 2^{\mathbb{P}(G_m^R)}$ est telle que $\Psi^C(\emptyset) := \mathbb{P}(G_m^R)$, et pour tout $T \subseteq \Delta(G_m^R)$ non-vide,

$$\Psi^C(T) := \bigcap_{(v_i \to v_j, \omega) \in T} \{P \in \mathbb{P}(G_m^R) \mid j > i \Rightarrow P_{v,\omega} \geq i+1, j < i \Rightarrow P_{v,\omega} \leq i-1\}$$

Cette sémantique raffine ainsi l'ensemble des paramétrisations complet, en gardant exactement le sous-ensemble qui permet les changements voulus. En particulier, étant donné une trace π, on obtient que pour toute paramétrisation $P \in \Psi^C(\tilde{\pi})$, π est également une trace avec $\mathscr{P} = \{P\}$, donc du modèle complètement paramétré par P.

Il faut bien remarquer que les ensembles de paramétrisations manipulés peuvent être représentés symboliquement, par exemple en logique propositionnelle (Bernot, Comet, Khalis,

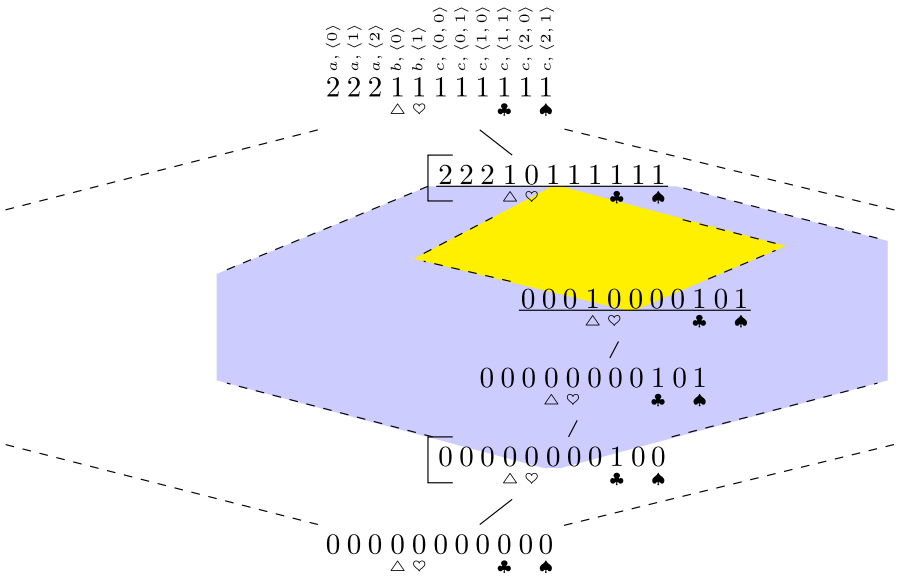


FIGURE 3.4 – Illustration de l'abstraction d'un ensemble de paramétrisations pour le PRN de la figure 3.3 par des paramétrisations spécifiant les bornes de chaque paramètre. Les paramétrisations $L = \langle 00000000000 \rangle$ et $U = \langle 22211111111 \rangle$ abstraient l'ensemble complet $\mathbb{P}(G_m^R)$. Les deux paramétrisations avec un crochet gauche correspondent au raffinement avec les changements $T = \{(c_0 \to c_1, \langle 1, 1 \rangle), (b_1 \to b_0, \langle 1 \rangle)\}$ (espace bleu) ; les contraintes d'influences $R = \{((a, c), +), ((b, b), o)\}$ amènent alors à l'ensemble délimité par les paramétrisations soulignées (jaune). Le paramètre correspondant à $b, \langle b = 0 \rangle$ est marqué avec △ ; $b, \langle b = 1 \rangle$ avec ♡ ; $c, \langle a = 1, b = 1 \rangle$ avec ♣ ; et $c, \langle a = 2, b = 1 \rangle$ avec ♠.

Richard, & Roux 2019, Gallet, Manceny, Le Gall, & Ballarini 2014). Toutefois, ces représentations symboliques peuvent très vite devenir trop longues, puisque les raffinements successifs vont les faire grandir, et que leur simplification est très ardue.

3.2.3 Abstraction des ensembles de paramétrisations

Afin de maîtriser la taille requise pour spécifier un ensemble de paramétrisations, nous avons proposé une abstraction qui repose sur l'encadrement d'un ensemble par deux paramétrisations, chacune donnant pour chaque paramètre la valeur minimale et maximale dans l'ensemble. Algébriquement, la structure obtenue est un sous-treillis convexe borné de $\mathbb{P}(G_m^R)$. Cette abstraction requiert ainsi 2 vecteurs de $|\Omega| = \prod_{v \in V} m_v^{|n^-(v))|}$ dimensions, et reste de taille constante.

Le défi (relevé par Juraj Kolčák) fut alors de définir une sémantique abstraite d'ensemble de paramétrisations, permettant notamment d'appliquer efficacement le raffinement d'un ensemble par un changement donné. Ce procédé est résumé dans la figure 3.4.

La sémantique abstraite $\Psi^\#(T)$ peut alors se définir comme un ajustement progressif des bornes de l'ensemble de paramétrisations, à la fois en tenant compte des contraintes relatives

aux changements de valeurs, et en propageant celles liées aux contraintes d'influence R.

Un des résultats majeurs est que cette abstraction reste très précise :

Théorème 3.1. *(Kolčák et al. 2018) Pour tout ensemble de changements $T \subseteq \mathbb{P}(G^r_m)$, $\Psi^\#(T)$ est le plus petit sous-treillis convexe contenant $\Psi^C(T)$.*

Il en découle notamment que toute trace π est réalisable avec la sémantique abstraite si *et seulement si* elle est réalisable avec la sémantique concrète :

$$\Psi^\#(\tilde{\pi}) = \emptyset \iff \Psi^C(\tilde{\pi}) = \emptyset \ .$$

Ainsi, alors que l'abstraction peut englober des paramétrisations faussement positives, elle n'entraîne pas de sur-approximation de l'ensemble des traces réalisables.

3.2.4 Dépliages

La sémantique d'ensembles de paramétrisations permet d'explorer toutes les traces réalisables depuis une configuration initiale donnée, et ainsi analyser la diversité des dynamiques possibles selon différents sous-ensembles de paramétrisations.

La notion de dépliage, principalement approfondie dans le cadre des réseaux de Petri (Esparza & Heljanko 2008), fournit une représentation compacte de toutes les configurations accessibles en évitant une redondance due aux entrelacements possibles de transitions sans conflit. L'ingrédient important est la condition de *cut off* qui indique quand l'exploration peut s'arrêter tout en conservant la complétude des configurations accessibles.

Nous avons pu définir une notion de dépliage pour les réseaux de régulation paramétriques, en particulier avec la sémantique abstraite définie précédemment. L'application à des exemples biologiques jusqu'à 15 composants a montré que l'approche par dépliage apporte un gain substantiel pour le passage à l'échelle (Kolčák et al. 2018).

3.2.5 Couplage avec la réduction de modèle dirigée

Une des applications des sémantiques d'ensembles de paramétrisations est l'identification de modèles pouvant reproduire un comportement donné. Dans le cas où le comportement revient à pouvoir atteindre un ensemble de configurations données, une exploration naïve des traces possibles s'avère très coûteuse, même avec les dépliages. En effet, de nombreux changements de valeurs n'auront aucune influence sur l'atteinte des configurations recherchées.

J'avais abordé cette problématique avec les réseaux d'automates non-déterministes, en introduisant la réduction de modèle *dirigée* (*goal-oriented reduction*) par des méthodes d'interprétation abstraite (Paulevé 2018). Le principe est d'identifier, sans exécuter réellement le modèle, les transitions qui ne contribuent pas à l'atteinte des configurations données. La réduction garantit de préserver toutes les traces minimales (au sens de l'inclusion des transitions) entre la configuration initiale et l'objectif (ensemble de configurations).

La réduction dirigée peut se voir comme agissant au niveau d'un modèle complètement paramétré. Dans (Haar et al. 2019), nous avons montré comment cette réduction peut être élevée au niveau des sémantiques d'ensembles, amenant à des réseaux de régulation paramétriques

dirigés. Cette méthode combine ainsi le raffinement de paramétrisations tout en réduisant les comportements à explorer, et en préservant l'ensemble complet des traces minimales pour l'accessibilité recherchée.

3.3 Discussion

Dans ce chapitre, j'ai brièvement présenté deux approches très différentes pour le raisonnement sur des ensembles de modèles qualitatifs.

D'un côté, la méthode de synthèse présentée dans la section 3.1 permet de construire des ensembles de réseaux booléens qui partagent des propriétés structurelles et dynamiques, sur leurs trajectoires et leurs attracteurs. Cette approche permet d'obtenir automatiquement des modèles formels compatibles avec des données biologiques, et qui peuvent servir à produire des prédictions sur le comportement du système. Basée sur la sémantique MP, la synthèse est applicable à des grands réseaux, avec plusieurs centaines voire milliers de composants. Toutefois, les ensembles de modèles sont obtenus par une énumération des réseaux booléens compatibles, ce qui rend difficile un raisonnement exhaustif. Il faut alors requérir à des approximations, soit par l'ajout de critères d'optimisation, soit par échantillonnage, sujet à différents paramètres.

D'un autre côté, la méthode d'abstraction présentée dans la section 3.2 offre la possibilité de raisonner exhaustivement sur des grands ensembles de modèles qualitatifs, sans avoir à les énumérer individuellement, avec une représentation dont la taille reste maîtrisée. Cependant, cette approche repose sur une exécution pas à pas de ces ensembles de modèles qui peut devenir très gourmande en calcul lorsque l'ensemble présente une grande diversité de comportements possibles. Ceci rend difficile l'analyse de propriétés dynamiques liées à l'absence de trajectoires ou aux attracteurs au sein de grands réseaux.

Chapitre 4

Applications pour la reprogrammation cellulaire

Les réseaux booléens sont largement employés pour la modélisation des processus de différenciation et de détermination cellulaire. La section 4.1 résume des résultats obtenus pour la reprogrammation des réseaux booléens, c'est-à-dire le contrôle de leurs attracteurs, en particulier au travers de la thèse de doctorat de Hugues Mandon. La section 4.2 présente de manière informelle le projet AlgoReCell portant à la fois sur le développement méthodologique et la mise en application de la reprogrammation des réseaux booléens, et qui a motivé la plupart des travaux de recherche relatés dans ce manuscrit.

Au fil de ses divisions, et en fonction de l'environnement, une cellule pluripotente (par exemple une cellule souche) peut se spécialiser en différents types de cellules (peau, graisse, os…). Ces différenciations s'observent aussi au niveau de l'expression de certains gènes, qui ne vont être exprimés que dans certains sous-types cellulaires. Pendant longtemps, il était supposé qu'une fois différenciée, une cellule ne peut pas revenir en arrière et se transformer en un autre type de cellule. Mais des expériences récentes ont montré qu'il était possible de changer le type de certaines cellules, en déstabilisant son réseau de gènes, pour transformer par exemple de la peau (cellules fibroblastes) en neurone, ou de la graisse (adipocytes) en os (ostéoblastes) (Takahashi et al. 2007, Takahashi & Yamanaka 2016, Tokuzawa et al. 2010). On parle alors de dé-différenciation (retour à un état pluripotent) et de trans-différenciation (changement transversal de type cellulaire). La figure 4.1 illustre ces processus.

De nombreux travaux portent actuellement sur l'étude de la *reprogrammation cellulaire* à l'aide des réseaux booléens (Abou-Jaoudé et al. 2015, Cohen et al. 2015, Collombet et al. 2017, Zañudo & Albert 2015). L'objectif est de prédire automatiquement des perturbations à appliquer à la cellule pour provoquer un changement de type. Ces perturbations sont typiquement des mutations de certains gènes, afin de bloquer ou de forcer leur expression.

Une des hypothèses utilisées est que les états différenciés stables de la cellule correspondent

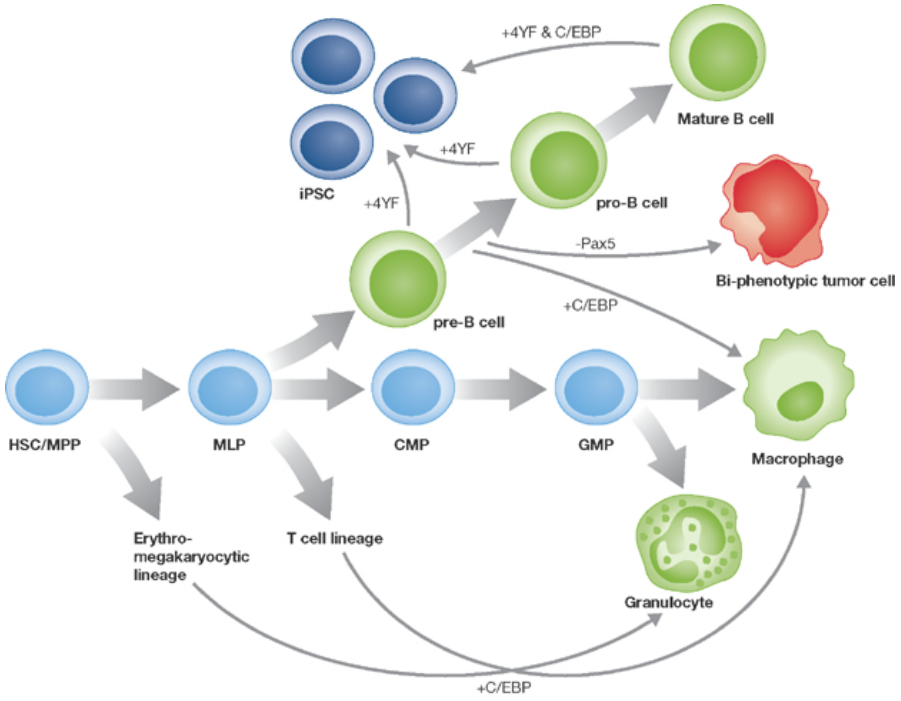


FIGURE 4.1 – (Regalo & Leutz 2013) Représentation schématique des différenciations cellulaires au cours d'une partie de l'hématopoïèse (processus de production des cellules sanguine), et quelques mécanismes de reprogrammation connus via l'action de facteurs de transcription indiqués sur les arcs ; iPSC : cellules souches induites (dé-différenciation).

à des attracteurs du réseau booléen associé. La reprogrammation des réseaux booléens revient alors à trouver les perturbations à appliquer à un point fixe pour rendre possible l'existence d'une trajectoire vers un point fixe différent.

Les réseaux booléens utilisés pour les problématiques de reprogrammation sont généralement de très grande taille, allant de plusieurs dizaines à plusieurs centaines, voire milliers de variables. La prédiction de perturbations pour la reprogrammation pose ici un problème combinatoire du nombre de perturbations candidates à tester.

Les analyses statiques permettent alors de restreindre l'espace de recherche des perturbations candidates, que ce soit d'un point de vue formel pour éviter d'étudier des candidats qui ne peuvent pas provoquer une reprogrammation (Mandon, Haar, & Paulevé 2016) ; ou d'un point de vue pratique pour guider la recherche d'une solution possible. Par exemple, Crespo, Perumal, Jurkowski, et del Sol (2013) exploitent les cycles positifs du graphe d'influence pour prédire certaines perturbations pour provoquer un changement de point fixe.

4.1 Reprogrammation séquentielle des réseaux booléens

La reprogrammation des réseaux booléens est un *contrôle* qui consiste à appliquer des perturbations de configuration ou de fonction pour garantir l'accessibilité d'un attracteur donné. La prédiction de ces perturbations donne alors des cibles potentielles pour la reprogrammation cellulaire. Le problème du contrôle a de nombreuses variantes, selon sa restriction à partir d'une configuration initiale donnée, ou le type de perturbations (permanentes, temporaires, ou instantanées) et leur application (simultanée, ou en séquence) ; mais repose toujours fondamentalement sur le calcul d'attracteurs et de propriétés d'accessibilité (Biane & Delaplace 2019, Mandon, Su, Pang, et al. 2019, Su, Paul, & Pang 2019, Zañudo & Albert 2015).

Dans la littérature, la grande majorité des approches cherchent des perturbations à appliquer en une seule fois, on parle de *one-step* ou *single-step reprogramming*. Les types de perturbations généralement considérées peuvent se décliner en trois types :

— Les perturbations *instantanées*, qui modifient l'état d'un ou plusieurs composants : cela revient à modifier la configuration actuelle du système, sans modifier sa fonction d'évolution. Expérimentalement, elles peuvent correspondre à l'injection de facteurs de transcription ou d'inhibiteurs.

— Les perturbations *temporaires*, qui forcent l'état d'un ou plusieurs composants pendant un certain temps, généralement jusqu'à une stabilisation de la cellule : cela revient à modifier la configuration, mais aussi la fonction d'évolution. Expérimentalement, elles reviennent à maintenir une perturbation, par exemple par l'activation de voies de signalisation ou encore l'injection de plasmides qui seront finalement dégradés.

— Les perturbations *permanentes*, qui forcent définitivement l'état d'un ou plusieurs composants du système. Expérimentalement, elles correspondent aux mutations de gènes (*knock-out* ou édition CRISPR) et activations constitutives.

Avec la thèse de doctorat de Mandon (2019), nous avons exploré le potentiel de la *reprogrammation séquentielle* (*sequential reprogramming*) des réseaux. Le principe est de laisser le système évoluer naturellement entre les perturbations, ce qui permet dans certains cas, d'économiser des perturbations à effectuer, comme illustré par la figure 4.2. La figure 4.3 illustre le bénéfice de la reprogrammation séquentielle sur un exemple de réseau booléen, avec la sémantique pleinement asynchrone. En one-step, la reprogrammation recherchée nécessite 3 perturbations, alors qu'en séquentiel, seulement 2 sont requises. Dans cet exemple, on peut remarquer que les perturbations fonctionneraient aussi en temporaire ou permanent.

Ce principe a été introduit dans l'article (Mandon, Haar, & Paulevé 2017) avec une formalisation générale du problème à l'aide de réseaux de Petri : partant d'un réseau booléen, nous construisons un réseau de Petri qui modélise conjointement la dynamique du réseau et les perturbations possibles, avec le type désiré (permanente, temporaire, instantanée). Le problème de reprogrammation revient alors à chercher des trajectoires avec les bonnes propriétés, desquelles peuvent être extraites les perturbations à effectuer, et dans quelles conditions.

En collaboration avec Jun Pang et Cui Su (Univ Luxembourg), ces travaux ont été à l'origine de différentes avancées, en particulier pour obtenir des algorithmes efficaces et applicables sur des réseaux de grande taille, mais également pour délimiter certaines classes de stratégies séquentielles adaptées à être transposées en protocole expérimental. En effet, l'application de

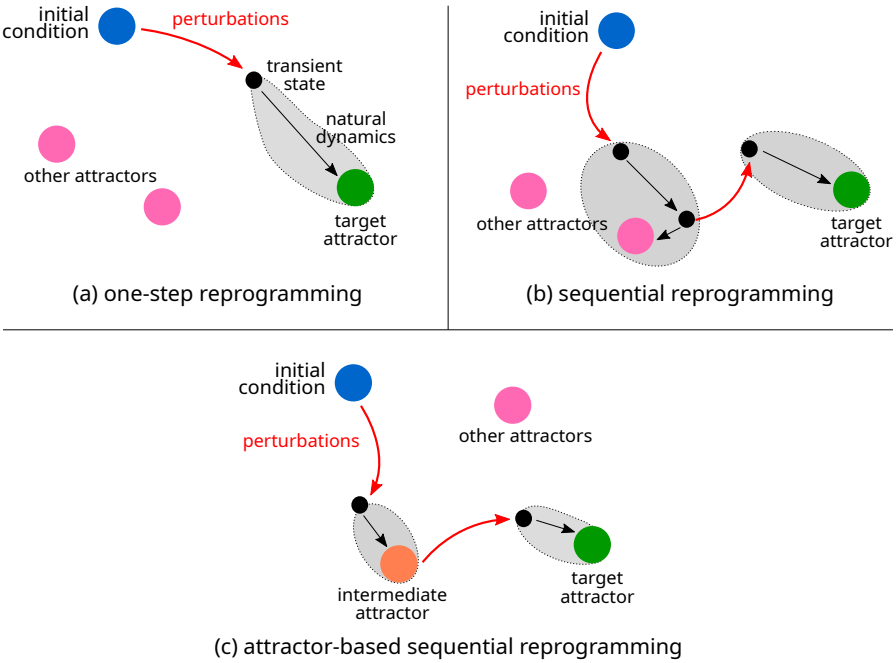


FIGURE 4.2 – Variantes de reprogrammation des réseaux booléens.

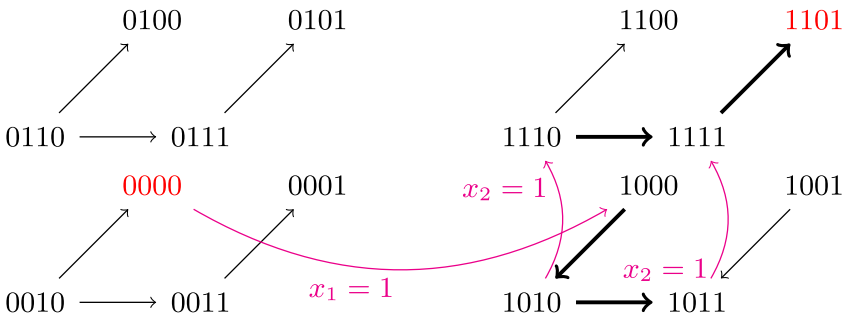


FIGURE 4.3 – Exemple de reprogrammation séquentielle avec le réseau booléen f de dimension 4 où $f_1(x) = x_1$, $f_2(x) = x_2$, $f_3(x) = x_1 \wedge \neg x_2$, $f_4(x) = x_3 \vee x_4$ pour aller de la configuration 0000 à la configuration 1101. Les flèches noires indiquent les transitions possibles avec la sémantique pleinement asynchrone ; les flèches magenta sont les perturbations (instantanées) avec la sémantique pleinement asynchrone.

stratégies de reprogrammation séquentielle pose le problème de l'observabilité de l'état du système pour pouvoir agir au bon moment.

Dans (Mandon, Su, Haar, Pang, & Paulevé 2019), nous avons montré que se restreindre aux stratégies séquentielles dont les perturbations sont à effectuer uniquement dans des attracteurs (cellules stabilisées) peut permettre d'économiser des perturbations (comparé à l'approche one-step) tout en étant plus simple à implémenter expérimentalement : il "suffit" d'attendre que les cellules se stabilisent pour appliquer les perturbations suivantes, sans avoir besoin d'observer précisément l'activité de leurs composants au cours du temps. Initialement étudié avec des perturbations instantanées, ce principe a depuis été implémenté pour les perturbations temporaires et permanentes (Su & Pang 2020).

4.2 Le projet "AlgoReCell"

En 2017, j'ai eu la chance d'obtenir le financement d'un projet bilatéral France-Luxembourg co-financé par l'ANR et le FNR sur les "Modèles informatiques et algorithmes pour la prédiction de cibles de reprogrammation cellulaire avec haute fidélité et haute efficacité" – AlgoReCell. Ce projet regroupe 6 équipes d'informatique (LRI, LSV/Inria Saclay et FSTC Computer Science/Univ Luxembourg), de biologistes théoriques (U900/Institut Curie ; LCSB/Univ Luxembourg) et expérimentaux (FSTC Life/Univ Luxembourg).

Ce projet, qui se terminera en janvier 2021, a créé l'occasion de faire collaborer des chercheurs d'horizons différents autour d'une application pratique de reprogrammation cellulaire, et avec comme but de développer et consolider les aspects théoriques, et mettre à l'épreuve les méthodes de modélisation et d'analyse qualitatives. La présence d'un groupe faisant à la fois de la modélisation et de l'expérimental permet de travailler sur la chaîne complète, de la cellule de souris à la prédiction de cible de reprogrammations de réseaux booléens, avec une validation expérimentale actuellement à l'étude. Ce projet a fortement contribué à la motivation des approches présentées dans ce manuscrit, dont les sujets des trois thèses que j'ai co-encadrées jusqu'à maintenant, avec des nouvelles collaborations qui s'inscrivent dans le long terme.

Le défi biologique qui nous rassemble porte sur la trans-différenciation *in vitro* de cellules de souris adipocytes (graisse) en ostéoblastes (précurseur de l'OS). Les données expérimentales portent sur le processus de différenciation des cellules pluripotentes de la mœlle épinière de la souris : selon le traitement appliqué, les cellules se différencient en adipocytes ou en ostéoblastes. Le processus *in vitro* prend une quinzaine de jours. À différents stades des deux différenciations, l'équipe expérimentale a prélevé un ensemble de cellules et effectué un séquençage et comptage des ARN (RNASeq bulk).

Cette application soulève des défis importants au niveau informatique, en particulier pour la caractérisation de différentes stratégies de reprogrammation des réseaux booléens (voir section précédente), mais également pour la construction réseau à partir des données expérimentales, et la modélisation des notions de fidélité (ressemblance entre la cellule reprogrammée et la cellule cible) et d'efficacité (proportion des cellules se reprogrammant vers le type cible).

Nous avons ainsi créé un pipeline illustré par la figure 4.4 qui, à partir de mesures expérimentales et de données sur l'architecture du réseau, aboutit à la prédiction et au classement de cibles potentielles de reprogrammation (article en cours d'écriture).

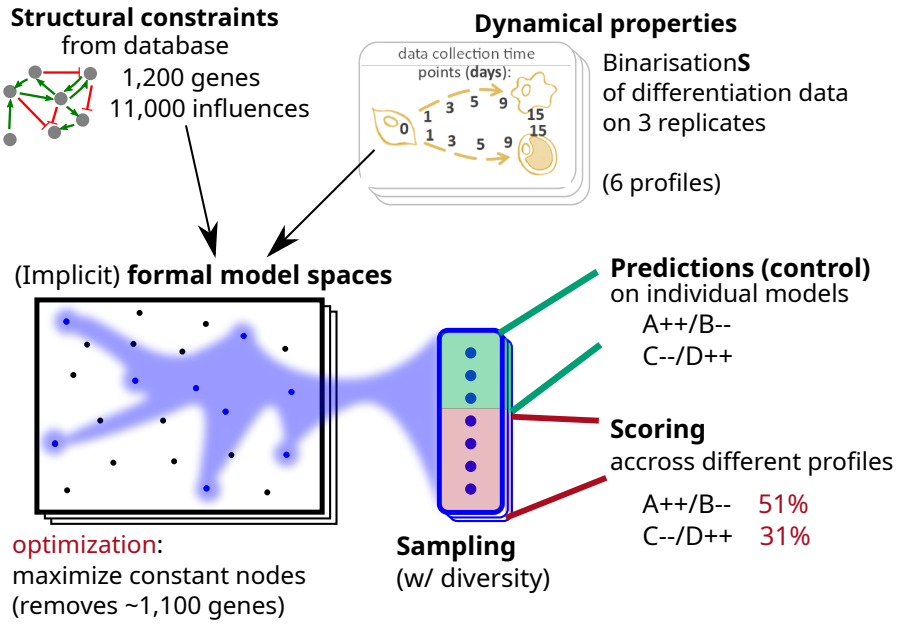


FIGURE 4.4 – Illustration du pipeline créé pour le projet ANR-FNR AlgoReCell pour la prédiction de déterminants de reprogrammation pour la trans-différenciation de cellules adipocytes en ostéoblastes à partir de données RNAseq en bulk à différents stades de la différenciation de cellules pluri-potentes ST2 chez la souris

Nous nous sommes focalisés sur les facteurs de transcription et avons utilisé une base de données (MetaCore™) référençant toutes les régulations connues entre ces composants (autour de 1 200). Le pipeline débute avec la binarisation des données expérimentales pour associer aux différents points de temps une activité (oui, non, inconnue) à chacun des facteurs de transcription. Nous avons sélectionné deux méthodes de binarisation, aboutissant à deux jeux de contraintes sur la dynamique booléenne du système. Nous employons alors la synthèse d'ensemble de réseaux booléens (section 3.1) pour extraire un ensemble de 1 000 réseaux pour chacune des méthodes de binarisation : chacun de ces réseaux reproduit le comportement de différenciation observé. Ensuite, sur une fraction de ces réseaux, nous effectuons la prédiction de reprogrammation des attracteurs correspondant au type adipocyte vers les attracteurs de type ostéoblaste sur les modèles individuels. Nous obtenons ainsi une liste de candidats, dont l'efficacité et la fidélité sont de 100% sur au moins un modèle.

La liste de candidats est alors évaluée sur l'ensemble de 2 000 réseaux booléens à l'aide de simulations stochastiques uniformes (Stoll et al. 2017) : nous pouvons alors mesurer la proportion de simulations ayant atteint un attracteur ostéoblaste (efficacité), et nous pouvons mesurer la distance aux mesures expérimentales (fidélité).

Il apparait que les candidats identifiés sont bien des gènes connus ou suspectés d'avoir une influence sur la différenciation vers les ostéoblastes, et suggère de nouvelles combinaisons pour provoquer une trans-différenciation. Ces combinaisons sont actuellement en cours d'étude par nos collaborateurs afin de décider de celles à évaluer expérimentalement.

Chapitre 5

Accessibilité et reproductibilité des analyses informatiques

Je présente dans ce chapitre mes contributions à l'accessibilité et à la reproductibilité des analyses informatiques : à la fois par des logiciels implémentant mes résultats de recherche, mais surtout par la mise au point d'un environnement facilitant l'utilisation de différents outils pour la modélisation et l'analyse de réseaux booléens et multivalués, et l'édition et le partage d'analyses informatiques.

Le développement et la mise à disposition d'outils sont des tâches cruciales dans un contexte trans-disciplinaire : ce sont les outils qui permettent de mettre en pratique des avancées théoriques, et de les rendre accessibles aux chercheurs et ingénieurs d'autres disciplines.

Outre l'implémentation en elle-même d'un algorithme, une dimension importante est la facilité d'installation, d'utilisation, et d'interopérabilité des outils, cette dernière étant nécessaire pour pouvoir enchaîner différentes analyses à partir d'un même modèle d'entrée.

La reproductibilité des analyses informatiques est un enjeu majeur qui est loin d'être trivial à satisfaire. Les logiciels évoluent, ils sont plus ou moins faciles à installer, avec beaucoup de dépendances, certaines URL disparaissent, mais surtout, la description des analyses effectuées dans les articles scientifiques manque souvent de précision. Pouvoir exécuter à nouveau une analyse décrite dans un article scientifique datant de quelques années est un vrai défi (et nécessite quasiment systématiquement le contact des auteurs...).

5.1 Logiciels développés

Pint (`https://loicpauleve.name/pint` ; depuis 2010) Logiciel initié durant ma thèse de doctorat, et qui implémente la plupart de mes résultats autour de l'analyse de la dynamique des réseaux d'automates (proches des réseaux de Petri) avec la sémantique pleinement asynchrone

(voir l'Avant-propos). En particulier, Pint implémente le calcul des points fixes (par résolution SAT), l'analyse statique de propriétés d'accessibilité successives, l'analyse statique d'ensembles de coupe, de mutation, et de transitions de bifurcations pour rendre impossible des propriétés d'accessibilité, et la réduction de modèle "orienté objectif". Pint permet également l'import et l'export depuis différents format et formalismes, en particulier des réseaux booléens. Il a été présenté à part entière pour la conférence CMSB en 2017 (Paulevé 2017), et a permis l'analyse de réseaux de tailles jusqu'alors non traitables.

J'emploie Pint régulièrement dans le cadre de collaborations autour de l'analyse de réseaux biologiques, mais il devrait petit à petit être remplacé par les outils implémentant la sémantique MP quand il s'agit de réseaux booléens. Il est également utilisé par différentes équipes de recherche, principalement au LS2N (Nantes) et NII (Japon).

Caspo-TS (`https://github.com/bioasp/caspots`; depuis 2015) Outil pour l'identification de réseaux booléens à partir de contraintes d'architecture et de contraintes d'accessibilités positives (séries temporelles) dans différentes conditions (Ostrowski, Paulevé, Schaub, Siegel, & Guziolowski 2015, Ostrowski et al. 2016). Basé sur `caspo` (Videla, Saez-Rodriguez, Guziolowski, & Siegel 2017) et `clingo` (Gebser et al. 2014), je l'ai développé avec Max Ostrowski (alors doctorant à Postdam, Allemage), dans le cadre d'une collaboration avec Anne Siegel (IRISA, Rennes) et Carito Guziolowski (LS2N, Nantes). Il est au cœur de l'étude de Razzaq, Paulevé, et al. (2018) publiée dans PLOS Computational Biology, et fait l'objet d'extensions pour différents projets de recherche en cours.

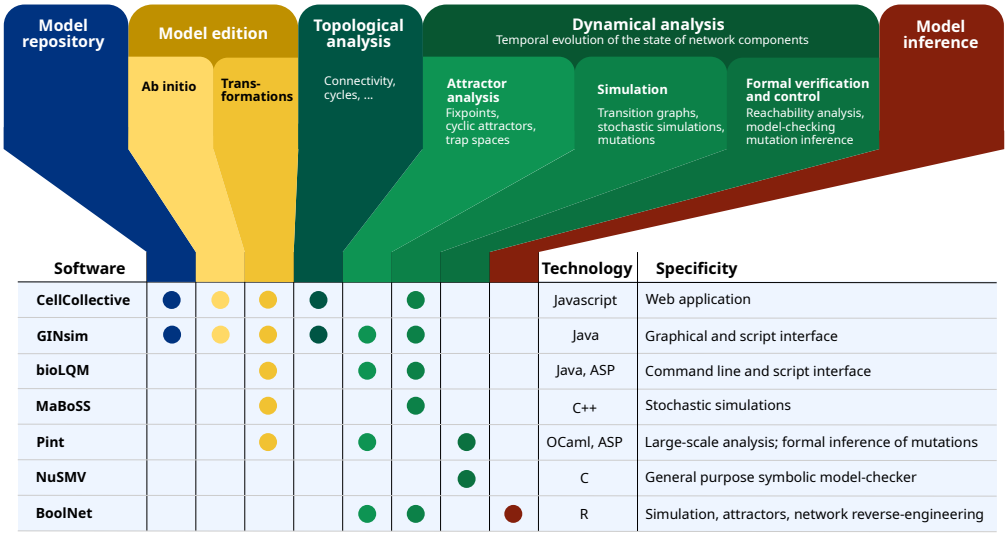
mpbn (`https://github.com/pauleve/mpbn`; depuis 2020) Implémentation légère de la sémantique MP pour l'analyse de propriétés d'accessibilité et d'attracteurs dans les réseaux booléens (voir section 2.5).

5.2 The CoLoMoTo Interactive Notebook and Docker images

En 2017, dans le cadre du groupe CoLoMoTo (Consortium for Logical Models and Tools), j'ai initié le projet *CoLoMoTo Interactive Notebook* (`http://colomoto.org/notebook`, Levy et al. 2018, Naldi et al. 2018) principalement avec Aurélien Naldi (alors à IBENS, Paris).

L'idée du projet est de faciliter l'accès à de nombreux outils développés par la communauté en biologie des systèmes autour de la modélisation qualitative des réseaux biologiques, et de faciliter l'écriture et le partage d'analyses informatiques. Le principe est de proposer un environnement logiciel pré-installé et figé, avec une interface unifiée pour appeler les différents outils et exporter les analyses et leurs résultats. Il est alors possible de ré-exécuter l'analyse plus tard, avec exactement le même environnement logiciel.

L'environnement CoLoMoTo repose sur la technologie de "conteneurs" Docker et l'interface web interactive Jupyter. Docker permet la création d'images à partir de recettes d'installation pour les différents logiciels à inclure. Ces images peuvent alors être exécutées directement sur Linux (avec un très faible surcoût pour les opérations réseau et d'accès au disque dur), mais également sur macOS et Windows, avec le surcoût d'une machine virtuelle.



Software									Technology	Specificity
CellCollective	●	●	●	●		●			Javascript	Web application
GINsim	●	●	●	●	●	●			Java	Graphical and script interface
bioLQM			●		●	●			Java, ASP	Command line and script interface
MaBoSS			●			●			C++	Stochastic simulations
Pint			●		●		●		OCaml, ASP	Large-scale analysis; formal inference of mutations
NuSMV							●		C	General purpose symbolic model-checker
BoolNet					●	●		●	R	Simulation, attractors, network reverse-engineering

FIGURE 5.1 – (extraite de Naldi et al. 2018) Exemples de logiciels intégrés dans l'environnement CoLoMoTo.

L'interface Jupyter permet l'édition de *notebooks* : ce sont des fichiers pouvant contenir du texte, des images, et des instructions dans un langage de programmation fixé. Nous avons opté pour le langage Python, et pour cela nous avons développé des bibliothèques Python pour invoquer les différents outils sélectionnés. Cela revient en quelque sorte à une interface en ligne de commande améliorée. Le notebook spécifie explicitement tous les paramètres utilisés pour appeler les différents outils, et les sorties obtenues. La couche Python rend très facile l'enchaînement d'analyses avec des outils différents, masquant la diversité des interfaces et des formats de fichiers utilisés par les logiciels. Un fichier notebook peut ensuite être publié avec les données supplémentaires d'un article, offrant la possibilité de le visualiser simplement dans un navigateur, mais également de l'importer pour pouvoir l'exécuter et le modifier.

La figure 5.1 donne un aperçu de la complémentarité d'une partie des logiciels inclus actuellement dans la distribution. La figure 5.2 montre une capture d'écran d'un notebook en cours d'édition.

Le terme reproductibilité se décline en réalité en plusieurs nuances : la "ré-exécutabilité", le fait de pouvoir relancer une analyse complète dans le même environnent logiciel ; la "répétabilité", le fait de pouvoir relancer une analyse dans un environnement un peu différent ; enfin la "reproductibilité", le fait de pouvoir aboutir à la même conclusion mais avec des outils différents.

L'environnement CoLoMoTo apporte une solution technique à ces trois aspects. Les images sont en effets "taguées" avec leur date de création et stockées publiquement sur un répertoire de Docker (`https://hub.docker.com/r/colomoto/colomoto-docker/tags`), avec toutes les informations pour pouvoir les reconstruire à l'identique. Lorsqu'un notebook est édité, le tag de l'image apparaît : il suffit alors de récupérer l'image correspondante pour pouvoir réexécuter le notebook dans le même environnement logiciel. De même, un notebook peut être

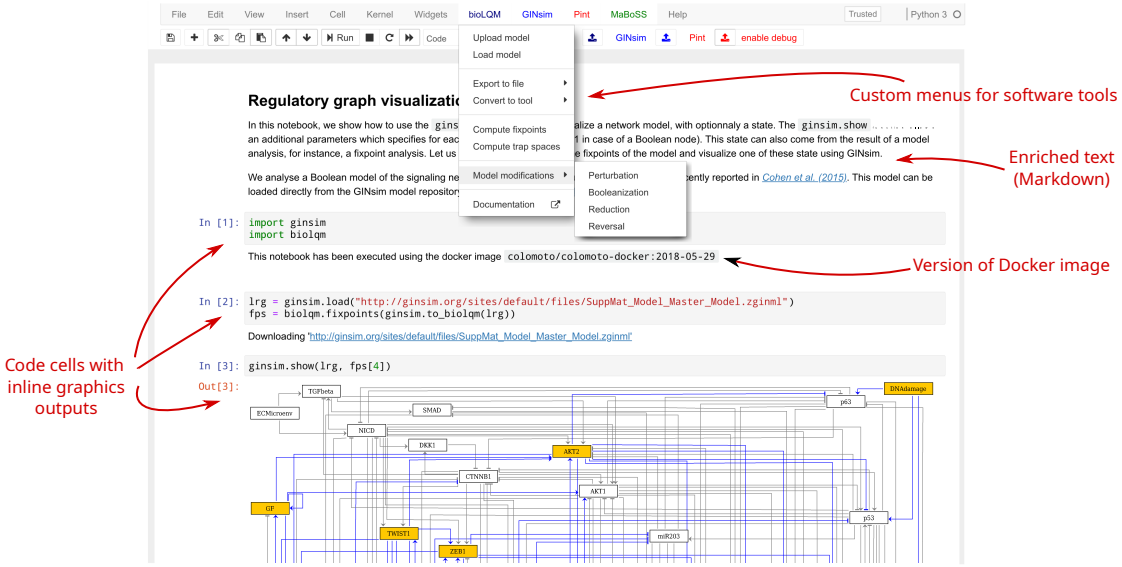


FIGURE 5.2 – Capture d'écran d'un notebook édité dans l'environnement CoLoMoTo.

exécuté dans une image plus récente, avec des logiciels mis à jour (répétabilité). Enfin, nous avons inclus différents outils offrant des fonctions similaires mais avec des méthodes différentes (p. ex., pour le calcul de points fixes ou le model-checking), et ainsi permettre de s'assurer d'un certain niveau de reproductibilité.

Si l'écriture de notebooks reproductibles est grandement facilitée par l'environnement CoLoMoTo, elle reste sujette à l'emploi de bonnes pratiques, notamment en ce qui concerne l'utilisation de ressources externes : il faut être vigilant à utiliser des URL pérennes (par exemple avec le service Zenodo [1]) ou toujours accompagner le notebook des fichiers externes requis.

La mise en place et la maintenance de cet environnement a nécessité la mise au point de workflows permettant une mise à jour et une validation automatique des images dans le temps. Actuellement, nous utilisons la plateforme GitHub, qui propose gratuitement les ressources nécessaires à l'automatisation de la maintenance du projet.

Il est possible d'utiliser l'environnement CoLoMoTo sans avoir à installer Docker, en se rendant à `https://tinyurl.com/colomoto-latest` [2] ; des exemples de notebooks peuvent être visualisés directement en ligne à partir de `http://colomoto.org/notebook` ; dont l'article (Levy et al. 2018) qui a été entièrement réalisé avec un notebook, et est donc ré-exécutable.

Pour l'instant, ce projet a reçu des contributions de 6 laboratoires (France, États-Unis, Portugal), et fait l'objet de nombreux tutoriels et enseignements. De nombreux autres logiciels sont en cours d'intégration, en particulier liés à l'apprentissage et à la reprogrammation de réseaux. Des premières publications s'accompagnent de notebooks édités avec CoLoMoTo pour permettre la reproduction de leurs résultats.

1. `https://zenodo.org`
2. `https://mybinder.org/v2/gh/colomoto/colomoto-docker/mybinder/latest`

Chapitre 6

Perspectives

Mes principales perspectives de recherche se déclinent en deux axes, autour de la sémantique MP, et de la synthèse et de l'apprentissage d'ensembles de réseaux booléens prédictifs.

Autour de la sémantique MP

La sémantique MP des réseaux booléens (chapitre 2) se différencie nettement des variantes de l'asynchrone par la complexité réduite de son analyse, en particulier avec les réseaux localement monotones, et des garanties formelles obtenues. De plus, nous avons montré sur différents cas pratiques que la sémantique MP reste prédictive et peut reproduire avec précision les processus de différenciation et de reprogrammation cellulaire.

La prédiction des attracteurs accessibles depuis des conditions initiales, et sous différentes conditions de mutations, est au cœur de très nombreuses études utilisant les modèles qualitatifs. La quantification des propensions de chaque attracteur, par exemple calculé par échantillonnage des trajectoires avec la sémantique purement asynchrone (Mendes et al. 2018, Stoll, Viara, Barillot, & Calzone 2012), est une piste importante à explorer avec la sémantique MP. Il serait également intéressant de caractériser la structure des bassins forts avec la sémantique MP, liée à la notion de transitions de bifurcations (Fitime et al. 2017) et de reprogrammation.

Plus fondamentalement, la notion de raffinement de modèle est centrale aux résultats de complétude et de minimalité apportés par la sémantique MP. Ceci motive l'étude de différentes familles de raffinement plus restrictives, par exemple en lien avec la monotonie, les modes de mises à jour, *etc.*, et invite à trouver des sémantiques minimales au niveau booléen.

Enfin, l'adaptation de la sémantique MP à d'autres formalismes, et en particulier aux réseaux de réactions mérite d'être explorée, puisqu'elle apporterait une abstraction à la fois des sémantiques discrètes et des sémantiques continues (Fages & Soliman 2008). Il faudrait toutefois alors s'assurer que la sémantique MP reste restrictive et donc prédictive.

Vers la synthèse d'ensembles de réseaux booléens *prédictifs*

Un des défis majeurs pour les applications en biologie reste la conception de modèles informatiques à partir des connaissances et données expérimentales.

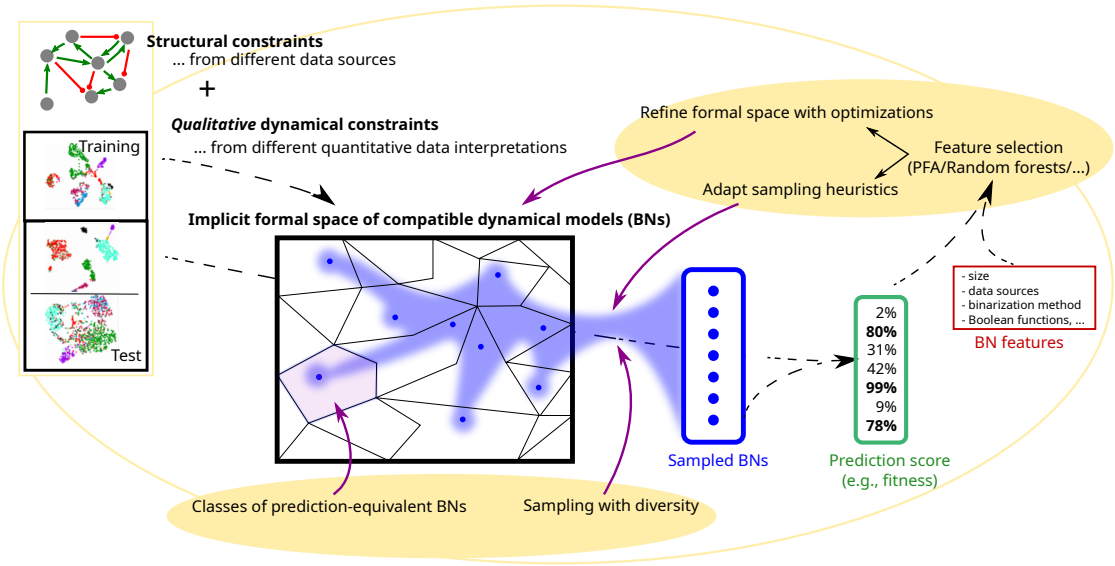


FIGURE 6.1 – Extension du pipeline AlgoReCell (section 4.2) pour la synthèse d'ensembles de réseaux *prédictifs*.

L'approche de synthèse d'ensembles de réseaux avec la sémantique MP présentée dans le chapitre 3 apporte des perspectives excitantes dans cette direction, avec encore de nombreuses questions à élucider.

En pratique, la synthèse aboutit très souvent sur un nombre astronomique de réseaux booléens vérifiant les contraintes imposées (Razzaq, Paulevé, et al. 2018). Il devient alors crucial de savoir échantillonner efficacement l'espace des solutions pour capturer la diversité des réseaux à considérer. Ceci implique des critères objectifs pour quantifier cette diversité, mais également des notions d'équivalence de modèles, pour éviter d'énumérer des réseaux différents mais présentant les mêmes comportements observables.

La notion de *modèle prédictif*, centrale dans les méthodes d'apprentissage automatique, est beaucoup plus floue avec l'approche formelle de la synthèse de modèle. Une piste de recherche pourrait appliquer les techniques classiques en apprentissage pour évaluer des modèles appris depuis un sous-ensemble des données d'entrée. Alors, on peut imaginer apprendre les caractéristiques des réseaux montrant les meilleurs scores de prédiction, et adapter en fonction les heuristiques de recherche de solutions, optimisant ainsi la capacité prédicitive d'un ensemble de réseaux, comme illustré par la figure 6.1.

Enfin, l'approche ensembliste de la modélisation apporte un nouvel angle pour la prédiction de stratégies de contrôles robustes, en particulier pour la reprogrammation cellulaire.

Ces perspectives de recherches reposent majoritairement sur des avancées théoriques, algorithmiques, et méthodologiques, mais ont également besoin d'un lien soutenu avec les applications en biologie pour guider et éprouver leur pertinence.

Références

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*. doi:10.3389/fbioe.2014.00086

Abou-Jaoudé, W., Thieffry, D., & Feret, J. (2016). Formal derivation of qualitative dynamical models from biochemical networks. *Biosystems*, *149*, 70–112. doi:10.1016/j.biosystems.2016.09.001

Albert, R., & Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, *74*(1), 47–97. doi:10.1103/revmodphys.74.47

Alon, N. (1985). Asynchronous threshold networks. *Graphs and Combinatorics*, *1*(1), 305–310. doi:10.1007/bf02582959

Aracena, J., Goles, E., Moreira, A., & Salinas, L. (2009). On the robustness of update schedules in Boolean networks. *Biosystems*, *97*(1), 1 - 8. doi:10.1016/j.biosystems.2009.03.006

Baldan, P., Bruni, A., Corradini, A., König, B., Rodríguez, C., & Schwoon, S. (2012). Efficient unfolding of contextual Petri nets. *Theoretical Computer Science*, *449*, 2–22. doi:10.1016/j.tcs.2012.04.046

Baral, C. (2003). *Knowledge representation, reasoning and declarative problem solving*. New York, NY, USA : Cambridge University Press. Hardcover.

Bernot, G., Comet, J.-P., Khalis, Z., Richard, A., & Roux, O. (2019). A genetically modified Hoare logic. *Theoretical Computer Science*, *765*, 145–157. doi:10.1016/j.tcs.2018.02.003

Biane, C., & Delaplace, F. (2019). Causal reasoning on Boolean control networks based on abduction : theory and application to cancer drug discovery. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *16*(5), 1574-1585. doi:10.1109/tcbb.2018.2889102

Bornholdt, S. (2008). Boolean network models of cellular regulation : prospects and limitations. *Journal of The Royal Society Interface*, *5*(suppl_1), S85-S94. doi:10.1098/rsif.2008.0132.focus

Chaouiya, C., Remy, E., Ruet, P., & Thieffry, D. (2004). Qualitative modelling of genetic networks : From logical regulatory graphs to standard Petri nets. In *Lecture notes in computer science* (pp. 137–156). Springer Berlin Heidelberg. doi:10.1007/978-3-540

-27793-4_9

Chatain, T., Haar, S., Jezequel, L., Paulevé, L., & Schwoon, S. (2014). Characterization of reachable attractors using Petri net unfoldings. In *Computational methods in systems biology* (Vol. 8859, p. 129-142). Cham : Springer Berlin Heidelberg. doi:10.1007/978-3-319-12982-2_10

Chatain, T., Haar, S., Kolčák, J., & Paulevé, L. (2020). *Most Permissive Semantics of Boolean Networks* (Technical Report). arXiv. Consulté sur https://hal.archives-ouvertes.fr/hal-01864693

Chatain, T., Haar, S., Kolčák, J., Paulevé, L., & Thakkar, A. (2020). Concurrency in Boolean networks. *Natural Computing*, *19*(1), 91–109. doi:10.1007/s11047-019-09748-4

Chatain, T., Haar, S., Koutny, M., & Schwoon, S. (2015). Non-atomic transition firing in contextual nets. In *Applications and Theory of Petri Nets* (Vol. 9115, p. 117-136). Springer. doi:10.1007/978-3-319-19488-2_6

Chatain, T., Haar, S., & Paulevé, L. (2018). Boolean Networks : Beyond Generalized Asynchronicity. In *Cellular Automata and Discrete Complex Systems (AUTOMATA 2018)* (Vol. 10875, pp. 29–42). Ghent, Belgium : Springer. doi:10.1007/978-3-319-92675-9_3

Chatain, T., & Paulevé, L. (2017). Goal-Driven Unfolding of Petri Nets. In *28th international conference on concurrency theory (CONCUR 2017)* (Vol. 85, pp. 18 :1–18 :16). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CONCUR.2017.18

Chen, H., Albergante, L., Hsu, J. Y., Lareau, C. A., Bosco, G. L., Guan, J., … Pinello, L. (2019). Single-cell trajectories reconstruction, exploration and mapping of omics data with STREAM. *Nature Communications*, *10*(1). doi:10.1038/s41467-019-09670-4

Cheng, A., Esparza, J., & Palsberg, J. (1995). Complexity results for 1-safe nets. *Theoretical Computer Science*, *147*(1&2), 117–136. doi:10.1016/0304-3975(94)00231-7

Chevalier, S., Froidevaux, C., Paulevé, L., & Zinovyev, A. (2019). Synthesis of Boolean Networks from Biological Dynamical Constraints using Answer-Set Programming. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)* (p. 34-41). Portland, Oregon, United States : IEEE. doi:10.1109/ICTAI.2019.00014

Chevalier, S., Noël, V., Calzone, L., Zinovyev, A., & Paulevé, L. (2020). Synthesis and Simulation of Ensembles of Boolean Networks for Cell Fate Decision. In *CMSB 2020 - 18th International Conference on Computational Methods in Systems Biology* (Vol. 12314, pp. 193–209). Cham : Springer. doi:10.1007/978-3-030-60327-4_11

Cohen, D. P. A., Martignetti, L., Robine, S., Barillot, E., Zinovyev, A., & Calzone, L. (2015). Mathematical modelling of molecular pathways enabling tumour cell invasion and migration. *PLOS Computational Biology*, *11*(11), e1004571. doi:10.1371/journal.pcbi.1004571

Collombet, S., van Oevelen, C., Sardina Ortega, J. L., Abou-Jaoudé, W., Di Stefano, B., Thomas-Chollier, M., … Thieffry, D. (2017). Logical modeling of lymphoid and myeloid cell specification and transdifferentiation. *Proceedings of the National Academy of Sciences*, *114*(23), 5792-5799. doi:10.1073/pnas.1610622114

Crespo, I., Perumal, T. M., Jurkowski, W., & del Sol, A. (2013). Detecting cellular reprogramming determinants by differential stability analysis of gene regulatory networks. *BMC Systems Biology*, *7*(1), 140. doi:10.1186/1752-0509-7-140

de Jong, H. (2002). Modeling and simulation of genetic regulatory systems : A literature review. *Journal of Computational Biology*, *9*, 67-103. doi:10.1089/10665270252833208

Dennunzio, A., Formenti, E., Manzoni, L., Mauri, G., & Porreca, A. E. (2015). Computational complexity of finite asynchronous cellular automata. *Theoretical Computer Science*. doi:10.1016/j.tcs.2015.12.003

Dennunzio, A., Formenti, E., Manzoni, L., & Porreca, A. E. (2019). Complexity of the dynamics of reaction systems. *Information and Computation*, *267*, 96–109. doi:10.1016/j.ic.2019.03.006

Didier, G., Remy, E., & Chaouiya, C. (2011). Mapping multivalued onto Boolean dynamics. *Journal of Theoretical Biology*, *270*(1), 177 - 184. doi:10.1016/j.jtbi.2010.09.017

Dorier, J., Crespo, I., Niknejad, A., Liechti, R., Ebeling, M., & Xenarios, I. (2016). Boolean regulatory network reconstruction using literature based knowledge with a genetic algorithm optimization method. *BMC Bioinformatics*, *17*(1), 410. doi:10.1186/s12859-016-1287-z

Eiter, T., & Gottlob, G. (1995). On the computational cost of disjunctive logic programming : Propositional case. *Annals of Mathematics and Artificial Intelligence*, *15*(3-4), 289–323. doi:10.1007/bf01536399

Esparza, J., & Heljanko, K. (2008). *Unfoldings : A partial-order approach to model checking (monographs in theoretical computer science. an eatcs series)* (1re éd.). Berlin, Heidelberg : Springer Publishing Company, Incorporated.

Fages, F., & Soliman, S. (2008). Abstract interpretation and types for systems biology. *Theoretical Computer Science*, *403*(1), 52 - 70. doi:10.1016/j.tcs.2008.04.024

Fatès, N. A. (2018). Asynchronous cellular automata. In R. Meyers (Ed.), *Encyclopedia of Complexity and Systems Science* (p. 21). Springer. doi:10.1007/978-3-642-27737-5_671-1

Feret, J., Danos, V., Krivine, J., Harmer, R., & Fontana, W. (2009). Internal coarse-graining of molecular systems. *Proceedings of the National Acedemy of Sciences*, *106*(16). doi:10.1073/pnas.0809908106

Fitime, L. F., Roux, O., Guziolowski, C., & Paulevé, L. (2017). Identification of bifurcation transitions in biological regulatory networks using Answer-Set Programming. *Algorithms for Molecular Biology*, *12*(1), 19. doi:10.1186/s13015-017-0110-3

Floréen, P., & Orponen, P. (1989). On the computational complexity of analyzing Hopfield nets. *Complex Systems*, *1989*(3), 577–587.

Formenti, E., Manzoni, L., & Porreca, A. E. (2014). Fixed points and attractors of reaction systems. In *Language, life, limits* (pp. 194–203). Springer International Publishing. doi:10.1007/978-3-319-08019-2_20

Gallet, E., Manceny, M., Le Gall, P., & Ballarini, P. (2014). Formal methods and software

engineering (ICFEM 2014). In (pp. 155–170). Cham : Springer International Publishing. doi:10.1007/978-3-319-11737-9_11

Gebser, M., Kaminski, R., Kaufmann, B., & Schaub, T. (2012). *Answer set solving in practice*. Morgan and Claypool Publishers.

Gebser, M., Kaminski, R., Kaufmann, B., & Schaub, T. (2014). Clingo = ASP + control : Preliminary report. *CoRR*, *abs/1405.3694*.

Glass, L., & Kauffman, S. (1973). Logical analysis of continuous, non-linear biochemical control networks. *Journal of Theoretical Biology*, *39*(1), 103-129. doi:10.1016/0022-5193(73)90208-7

Haar, S., Kern, C., & Schwoon, S. (2013). Computing the reveals relation in occurrence nets. *Theoretical Computer Science*, *493*, 66–79. doi:10.1016/j.tcs.2013.04.028

Haar, S., Kolčák, J., & Paulevé, L. (2019). Combining Refinement of Parametric Models with Goal-Oriented Reduction of Dynamics. In *VMCAI 2019 - 20th International Conference on Verification, Model Checking, and Abstract Interpretation* (Vol. 11388, pp. 555–576). Lisbon, Portugal : Springer. doi:10.1007/978-3-030-11245-5_26

Hamez, A., Thierry-Mieg, Y., & Kordon, F. (2009). Building efficient model checkers using hierarchical set decision diagrams and automatic saturation. *Fundam. Inf.*, *94*(3-4), 413–437. doi:10.3233/FI-2009-137

Ishihara, S., Fujimoto, K., & Shibata, T. (2005). Cross talking of network motifs in gene regulation that generates temporal pulses and spatial stripes. *Genes to Cells*, *10*(11), 1025–1038. doi:10.1111/j.1365-2443.2005.00897.x

Janicki, R., Kleijn, J., Koutny, M., & Mikulski, Ł. (2015). Step traces. *Acta Informatica*. doi:10.1007/s00236-015-0244-z

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly connected nets. *Journal of Theoretical Biology*, *22*, 437-467. doi:10.1016/0022-5193(69)90015-0

Klarner, H., Bockmayr, A., & Siebert, H. (2015). Computing maximal and minimal trap spaces of boolean networks. *Natural Computing*, *14*(4), 535–544. doi:10.1007/s11047-015-9520-7

Kleitman, D. (1969). On Dedekind's problem : The number of monotone Boolean functions. *Proceedings of the American Mathematical Society*, *21*(3), 677. doi:10.2307/2036446

Kolčák, J., Šafránek, D., Haar, S., & Paulevé, L. (2018). Parameter Space Abstraction and Unfolding Semantics of Discrete Regulatory Networks. *Theoretical Computer Science*, *765*, 120 - 144. doi:10.1016/j.tcs.2018.03.009

Lee, J., Lifschitz, V., & Palla, R. (2008). A reductive semantics for counting and choice in answer set programming. In *Proceedings of the 23rd national conference on artificial intelligence - volume 1* (p. 472–479). AAAI Press.

Le Novère, N. (2015). Quantitative and logic modelling of molecular and gene networks. *Nature reviews. Genetics*, *16*, 146–158. doi:10.1038/nrg3885

Levy, N., Naldi, A., Hernandez, C., Stoll, G., Thieffry, D., Zinovyev, A., ... Paulevé, L. (2018). Prediction of Mutations to Control Pathways Enabling Tumour Cell Invasion with the

CoLoMoTo Interactive Notebook (Tutorial) . *Frontiers in Physiology*, *9*, 787. doi:10.3389/fphys.2018.00787

Lin, F., & Zhao, Y. (2004). ASSAT : Computing answer sets of a logic program by SAT solvers. *Artificial Intelligence*, *157*(1), 115–137. doi:10.1016/j.artint.2004.04.004

Mandon, H. (2019). *Algorithms For Cell Reprogramming Strategies in Boolean Networks* (Theses, Université Paris-Saclay). Consulté sur https://tel.archives-ouvertes.fr/tel-02513383

Mandon, H., Haar, S., & Paulevé, L. (2016). Relationship between the Reprogramming Determinants of Boolean Networks and their Interaction Graph. In *Hybrid systems biology (HSB 2016) proceedings* (Vol. 9957, pp. 113–127). Springer International Publishing. doi:10.1007/978-3-319-47151-8_8

Mandon, H., Haar, S., & Paulevé, L. (2017). Temporal Reprogramming of Boolean Networks. In *CMSB 2017 - 15th conference on Computational Methods for Systems Biology* (Vol. 10545, pp. 179–195). Springer International Publishing. doi:10.1007/978-3-319-67471-1_11

Mandon, H., Su, C., Haar, S., Pang, J., & Paulevé, L. (2019). Sequential Reprogramming of Boolean Networks Made Practical. In *CMSB 2019 - 17th International Conference on Computational Methods in Systems Biology* (Vol. 11773, pp. 3–19). Cham : Springer. (Best paper award) doi:10.1007/978-3-030-31304-3_1

Mandon, H., Su, C., Pang, J., Paul, S., Haar, S., & Paulevé, L. (2019). Algorithms for the Sequential Reprogramming of Boolean Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *16*(5), 1610-1619. doi:10.1109/TCBB.2019.2914383

Mangan, S., & Alon, U. (2003, 14). Structure and function of the feed-forward loop network motif. *Proceedings of the National Academy of Sciences*, *100*(21), 11980–11985. doi:10.1073/pnas.2133841100

Manzoni, L. (2012). Asynchronous cellular automata and dynamical properties. *Natural Computing*, *11*(2), 269–276. doi:10.1007/s11047-012-9308-y

Melliti, T., Regnault, D., Richard, A., & Sené, S. (2016). Asynchronous simulation of Boolean networks by monotone Boolean networks. In *Lecture notes in computer science* (pp. 182–191). Springer International Publishing. doi:10.1007/978-3-319-44365-2_18

Mendes, N. D., Henriques, R., Remy, E., Carneiro, J., Monteiro, P. T., & Chaouiya, C. (2018). Estimating attractor reachability in asynchronous logical models. *Frontiers in Physiology*, *9*. doi:10.3389/fphys.2018.01161

Mizera, A., Pang, J., Qu, H., & Yuan, Q. (2017). A new decomposition method for attractor detection in large synchronous Boolean networks. In *Dependable software engineering. theories, tools, and applications* (pp. 232–249). Springer International Publishing. doi:10.1007/978-3-319-69483-2_14

Mizera, A., Pang, J., Qu, H., & Yuan, Q. (2019). Taming asynchrony for attractor detection in large Boolean networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *16*(1), 31–42. doi:10.1109/tcbb.2018.2850901

Murata, T. (1989). Petri nets : properties, analysis and applications. *Proc. of the IEEE*, *77*(4), 541–580.

Nakamura, K. (1981). Synchronous to asynchronous transformation of polyautomata. *Journal of Computer and System Sciences*, *23*(1), 22–37. doi:10.1016/0022-0000(81)90003-9

Naldi, A. (2018). BioLQM : A java toolkit for the manipulation and conversion of logical qualitative models of biological networks. *Frontiers in Physiology*, *9*. doi:10.3389/fphys.2018.01605

Naldi, A., Hernandez, C., Levy, N., Stoll, G., Monteiro, P. T., Chaouiya, C., … Paulevé, L. (2018). The CoLoMoTo Interactive Notebook : Accessible and Reproducible Computational Analyses for Qualitative Biological Networks. *Frontiers in Physiology*, *9*, 680. doi:10.3389/fphys.2018.00680

Naldi, A., Thieffry, D., & Chaouiya, C. (2007). Decision diagrams for the representation and analysis of logical models of genetic networks. In *Proceedings of the 2007 international conference on computational methods in systems biology* (pp. 233–247). Berlin, Heidelberg : Springer-Verlag. doi:10.1007/978-3-540-75140-3_16

Noual, M., & Sené, S. (2018). Synchronism versus asynchronism in monotonic Boolean automata networks. *Natural Computing*, *17*(2), 393–402. doi:10.1007/s11047-016-9608-8

Ostrowski, M., Paulevé, L., Schaub, T., Siegel, A., & Guziolowski, C. (2015). Boolean Network Identification from Multiplex Time Series Data. In *CMSB 2015 - 13th conference on Computational Methods for Systems Biology* (Vol. 9308, p. 170-181). Springer International Publishing. doi:10.1007/978-3-319-23401-4_15

Ostrowski, M., Paulevé, L., Schaub, T., Siegel, A., & Guziolowski, C. (2016). Boolean network identification from perturbation time series data combining dynamics abstraction and logic programming. *Biosystems*, *149*, 139 - 153. doi:10.1016/j.biosystems.2016.07.009

Papadimitriou, C. H. (1995). *Computational complexity*. Addison-Wesley.

Paulevé, L. (2011). *Modélisation, simulation et vérification des grands réseaux de régulation biologique* (Thèse de doctorat, École Centrale de Nantes). Consulté sur http://tel.archives-ouvertes.fr/tel-00635750

Paulevé, L. (2017). Pint : a static analyzer for transient dynamics of qualitative networks with IPython interface. In *CMSB 2017 - 15th conference on Computational Methods for Systems Biology* (Vol. 10545, pp. 309–316). Springer International Publishing. doi:10.1007/978-3-319-67471-1_20

Paulevé, L. (2018). Reduction of Qualitative Models of Biological Networks for Transient Dynamics Analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *15*(4), 1167-1179. doi:10.1109/TCBB.2017.2749225

Paulevé, L., Andrieux, G., & Koeppl, H. (2013). Under-approximating cut sets for reachability in large scale automata networks. In *Computer Aided Verification (CAV)* (Vol. 8044, p. 69-84). Berlin, Heidelberg : Springer Berlin Heidelberg. doi:10.1007/978-3-642-39799-8_4

Paulevé, L., Kolčák, J., Chatain, T., & Haar, S. (2020). Reconciling qualitative, abstract, and scalable modeling of biological networks. *Nature Communications*, *11*(1). doi:10.1038/s41467-020-18112-5

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. doi:10.1017/S0960129511000739

Paulevé, L. (2020). *VLBNs - Very Large Boolean Networks (Version 1) [dataset].* Zenodo. (Zenodo. https://doi.org/10.5281/zenodo.3714876)

Petri, C. A. (1962). *Kommunikation mit automaten* (Thèse de doctorat non publiée). University of Bonn.

Qiu, X., Mao, Q., Tang, Y., Wang, L., Chawla, R., Pliner, H. A., & Trapnell, C. (2017). Reversed graph embedding resolves complex single-cell trajectories. *Nature Methods*, *14*(10), 979–982. doi:10.1038/nmeth.4402

Razzaq, M., Kaminski, R., Romero, J., Schaub, T., Bourdon, J., & Guziolowski, C. (2018). Computing diverse boolean networks from phosphoproteomic time series data. In *Computational methods in systems biology* (pp. 59–74). Springer International Publishing. doi:10.1007/978-3-319-99429-1_4

Razzaq, M., Paulevé, L., Siegel, A., Saez-Rodriguez, J., Bourdon, J., & Guziolowski, C. (2018). Computational discovery of dynamic cell line specific boolean networks from multiplex time-course data. *PLOS Computational Biology*, *14*(10), 1-23. doi:10.1371/journal.pcbi.1006538

Regalo, G., & Leutz, A. (2013). Hacking cell differentiation : transcriptional rerouting in reprogramming, lineage infidelity and metaplasia. *EMBO Molecular Medicine*, *5*(8), 1154–1164. doi:10.1002/emmm.201302834

Remy, E., Rebouissou, S., Chaouiya, C., Zinovyev, A., Radvanyi, F., & Calzone, L. (2015). A modeling approach to explain mutually exclusive and co-occurring genetic alterations in bladder tumorigenesis. *Cancer Research*, *75*(19), 4042–4052. doi:10.1158/0008-5472.can-15-0602

Robert, F. (1995). *Les systemes dynamiques discrets* (Vol. 19). Springer Science & Business Media.

Rodrigo, G., & Elena, S. F. (2011). Structural discrimination of robustness in transcriptional feedforward loops for pattern formation. *PLOS ONE*, *6*(2), e16904. doi:10.1371/journal.pone.0016904

Rodríguez, C., & Schwoon, S. (2013). Cunf : A tool for unfolding and verifying Petri nets with read arcs. In *Automated technology for verification and analysis* (pp. 492–495). Springer. doi:10.1007/978-3-319-02444-8_42

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. doi:10.1038/msb.2009.87

Schaerli, Y., Munteanu, A., Gili, M., Cotterell, J., Sharpe, J., & Isalan, M. (2014). A unified design space of synthetic stripe-forming networks. *Nature Communications*, *5*(1). doi:10.1038/ncomms5905

Snoussi, E. (1989). Qualitative dynamics of a piecewise-linear differential equations : a discrete mapping approach. *Dynamics and stability of Systems*, *4*, 189–207.

Stoll, G., Caron, B., Viara, E., Dugourd, A., Zinovyev, A., Naldi, A., … Calzone, L. (2017). MaBoSS 2.0 : an environment for stochastic Boolean modeling. *Bioinformatics*, *33*(14), 2226–2228. doi:10.1093/bioinformatics/btx123

Stoll, G., Viara, E., Barillot, E., & Calzone, L. (2012). Continuous time Boolean modeling for biological signaling : application of Gillespie algorithm. *BMC Systems Biology*, *6*(1), 116. doi:10.1186/1752-0509-6-116

Su, C., & Pang, J. (2020). Sequential control of Boolean networks with temporary and permanent perturbations. *arXiv*(2004.07184). Consulté sur http://arxiv.org/abs/2004.07184v1 (non-peer reviewed)

Su, C., Paul, S., & Pang, J. (2019). Controlling large Boolean networks with temporary and permanent perturbations. In *Lecture notes in computer science* (pp. 707–724). Springer International Publishing. doi:10.1007/978-3-030-30942-8_41

Takahashi, K., Tanabe, K., Ohnuki, M., Narita, M., Ichisaka, T., Tomoda, K., & Yamanaka, S. (2007). Induction of pluripotent stem cells from adult human fibroblasts by defined factors. *Cell*, *131*, 861–872. doi:10.1016/j.cell.2007.11.019

Takahashi, K., & Yamanaka, S. (2016). A decade of transcription factor-mediated reprogramming to pluripotency. *Nat Rev Mol Cell Biol*, *17*(3), 183–193. doi:10.1038/nrm.2016.8

Terfve, C., Cokelaer, T., Henriques, D., MacNamara, A., Goncalves, E., Morris, M. K., … Saez-Rodriguez, J. (2012). CellNOptR : a flexible toolkit to train protein signaling networks to data using multiple logic formalisms. *BMC Systems Biology*, *6*(1), 133. doi:10.1186/1752-0509-6-133

Thieffry, D., & Thomas, R. (1995). Dynamical behaviour of biological regulatory networks—ii. immunity control in bacteriophage lambda. *Bulletin of Mathematical Biology*, *57*, 277-297. doi:10.1007/BF02460619

Thomas, R. (1973). Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, *42*(3), 563 - 585. doi:10.1016/0022-5193(73)90247-6

Thomas, R., & d'Ari, R. (1990). *Biological feedback*. Boca Raton , Florida, USA : CRC Press.

Tokuzawa, Y., Yagi, K., Yamashita, Y., Nakachi, Y., Nikaido, I., Bono, H., … et al. (2010). Id4, a new candidate gene for senile osteoporosis, acts as a molecular switch promoting osteoblast differentiation. *PLOS Genetics*, *6*(7), e1001019. doi:10.1371/journal.pgen.1001019

Videla, S., Saez-Rodriguez, J., Guziolowski, C., & Siegel, A. (2017). caspo : a toolbox for automated reasoning on the response of logical signaling networks families. *Bioinformatics*, btw738. doi:10.1093/bioinformatics/btw738

Vogler, W., Semenov, A., & Yakovlev, A. (1998). Unfolding and finite prefix for nets with read arcs. In *CONCUR '98 concurrency theory* (pp. 501–516). Springer Berlin Heidelberg. doi:10.1007/bfb0055644

Wiedemann, D. (1991). A computation of the eighth Dedekind number. *Order*, *8*(1), 5–6. doi:10.1007/bf00385808

Wolfram, S. (1983). Statistical mechanics of cellular automata. *Rev. Mod. Phys.*, *55*, 601–644. doi:10.1103/RevModPhys.55.601

Yordanov, B., Dunn, S.-J., Kugler, H., Smith, A., Martello, G., & Emmott, S. (2016). A method to identify and analyze biological programs through automated reasoning. *Systems Biology and Applications*, *2*.

Zañudo, J. G., Steinway, S. N., & Albert, R. (2018). Discrete dynamic network modeling of oncogenic signaling : Mechanistic insights for personalized treatment of cancer. *Current Opinion in Systems Biology*, *9*, 1–10. doi:10.1016/j.coisb.2018.02.002

Zañudo, J. G. T., & Albert, R. (2015). Cell fate reprogramming by control of intracellular network dynamics. *PLOS Computational Biology*, *11*(4), 1-24. doi:10.1371/journal.pcbi.1004193