Rapport de Stage de Master : Algorithmes pour la reprogrammation cellulaire

Hugues Mandon[1], Loïc Paulevé[1], and Stefan Haar[2]

[1]LRI UMR 8623, Univ. Paris-Sud – CNRS, Université Paris-Saclay, France
[2]LSV, ENS Cachan, INRIA, CNRS, Université Paris-Saclay, France

26 août 2016

Table des matières

1 Introduction

Le stage a eu lieu au LRI, dans le département bio-informatique, et était encadré par Stefan Haar (du LSV, à Cachan) et Loïc Paulevé (LRI).

Dans le contexte de la médecine régénérative, une des méthodes, en plein essor, pour traiter les patients, est la reprogrammation cellulaire, permettant de régénérer, par exemple, muscles ou neurones. De telles opérations sont devenues réalistes après que des expériences aient prouvées qu'un partie de la différentiation cellulaire peut être inversée[14]. Pendant leur développement, les cellules passent par plusieurs étapes où elles sont encore multipotentes, puis atteignent un état différentié. Ce processus peut être inversé, créant ainsi des cellules souches pluripotentes induites (iPSCs) depuis une cellule déjà différentiée. En forçant la cellule à se différentier autrement, cela permet de "transformer" le phénotype d'une cellule. Alternativement, il est également possible de directement "trans-différencier" une cellule sans passer par un état pluripotent intermédiaire[8, 6].

Dans la suite de ce rapport, la de-différentiation comme la trans-différentiation sont effectuées en ciblant certains gènes, appelés *Déterminants de Reprogrammation* (RDs), via l'usage de leurs facteurs de transcription [14, 5].

La prédiction automatisée des RDs nécessite de prendre en compte de nombreuses facettes de la dynamique cellulaire et de la stratégie de reprogrammation, telles que le type de perturbations (permanentes ou temporaires) et leur impact ; leur ordre ; la nature du phénotype cellulaire voulu (pluripotent ou différentié) ; le niveau d'inévitabilité voulue dans l'accessibilité de la cible (fidélité) ; la nature et durée de la cascade de régulations engendrée (efficacité) ; et également la robustesse des RDs vis à vis de l'hétérogénéité initiale au sein de la culture cellulaire, et en considérant les possibles incertitudes du modèle computationnel.

Actuellement, aucune méthode ne permet de satisfaire tous ces aspects afin de prédire systématiquement la meilleure combinaison de RD pour des reprogrammations cellulaires différentes.

Dans ce rapport, nous résolvons l'identification des RDs des *Réseaux Booléens*, qui modélisent les dynamiques des régulations géniques et des réseaux de signalisation. Les composants (les nœuds) du réseau sont représentés par des variables booléennes, et le changement d'état est décrit par des fonctions booléennes qui donnent le futur état des nœuds suivant l'état de leurs régulateurs[15, 1]. Les réseaux booléens sont adaptés pour une recherche algorithmique sur de vastes réseaux biologiques, où les connaissances sont majoritairement sur l'activation et l'inhibition entre les divers composants. De telles activations et inhibitions entre les composants forment un graphe orienté signé, appelé *Graphe d'Interaction* (IG).

Dans ces travaux, nous avons supposé que les états cellulaires différentiés correspondent aux *attracteurs* de la dynamique du modèle, i.e. les comportements sur le long terme. Dans le cadre des réseaux booléens, ces attracteurs peuvent être soit un unique état (un point fixe) ou un comportement de cycle terminal.

Les relations entre le graphe d' interaction d'un réseau booléen et le nombre d'attracteurs ont déjà été bien étudiées[1, 12, 13]. Par contre, il n'existe actuellement que peu de travaux sur les caractérisations des perturbations qui permettraient de déclencher le passage à un attracteur, ou de changer d'attracteur. Actuellement, la majorité des prédictions de RDs sont faites soit en utilisant une analyse statistique sur les données d'expression, pour classer les facteurs de transcription par ordre d'importance[2, 11, 9]. Bien que basées sur des modèles de réseaux, ces approches ne permettent pas d'obtenir un ensemble complet de solutions pour le problème de la reprogrammation cellulaire. Il existe également une heuristique pour trouver des RDs candidats depuis une analyse purement topologique du graphe d'interaction[5] : les RDs sont choisis uniquement dans les cycles positifs qui prennent une valeur différente entre le point fixe d'origine et celui d'arrivée. Cependant, il n'y a aucune garantie que les RDs trouvés permettent effectivement un changement d'attracteur dans les réseaux booléens, ni que le point fixe voulu est le seul atteignable. Finalement, on peut trouver une caractérisation formelle des RDs permettant d'entraîner un changement d'attracteur quand ils sont modifiés temporairement, dans le cas de la dynamique synchrone des réseaux booléens conjonctifs[7].

Contribution

. À partir d'un réseau booléen dont tous les attracteurs sont des points fixes, et étant donnés un point fixe initial et un point fixe cible, il est possible de donner une caractérisation des RDs potentiels (un ensemble de nœuds) en utilisant le graphe d'interaction et pour deux possibilités de reprogrammation cellulaire :

— soit avec une perturbation permanente des RDs, pour que le point fixe cible devienne atteignable dans la dynamique asynchrone du réseau booléen

— soit avec une perturbation permanente, pour que le point fixe cible devienne le seul attracteur atteignable dans la dynamique asynchrone du réseau booléen

Dans le premier cas, j'ai prouvé que tous les RDs sont uniquement dans certaines composantes fortement connexes du graphe d'interaction, et j'ai trouvé des algorithmes pour trouver ces RDs dans les deux cas. Dans le second cas, j'ai prouvé que seulement certains d'entre eux sont dans les composantes fortement connexes mentionnées précédemment. J'ai développé un algorithme pour obtenir les combinaisons possibles de perturbations permanentes permettant l'atteinte inévitable (i.e. le point fixe cible étant le seul point fixe atteignable). Cet algorithme peut rater certaines solutions, mais toutes les solutions données sont correctes.

Notations

Soit un ensemble fini I, 2^I est l'ensemble des parties de I, $|I|$ est le cardinal de I. Soit n un entier naturel, on définit $[n] = \{1, \ldots, n\}$.

Soit un état booléen$x \in \{0,1\}^n$ et un ensemble d'indices $I \subset [n]$, $\bar{x}^I$ est l'état où ${\bar{x}_i}^I = x_i$ si $i \notin I$ et ${\bar{x}_i}^I = 1 - x_i$ si $i \in I$. De la même manière, si on

a $x, y \in \{0,1\}^n$, $x_{[x_I=y_I]}$ est l'état où, pour tout $i \in I$, $(x_{[x_I=y_I]})_i = y_i$ et pour tout $i \notin I$, $(x_{[x_I=y_I]})_i = x_i$

2 Contexte

Cette section permet de définir formellement les réseaux booléens, leur graphe d'interaction, et leur graphe de transition dans la dynamique asynchrone. Elle permet aussi de rappeler des propriétés indispensables à la compréhension de l'algorithme.

2.1 Définitions

Réseau booléen : Un réseau booléen est un ensemble fini de variables booléennes, chacune ayant une fonction booléenne. Cette fonction booléenne est une fonction logique qui dépend des variables du réseau et permet de déterminer l'état futur de la variable.

Définition 1 (Réseau booléen)**.** Un réseau booléen de taille n est un ensemble de variables $\{x_1, \ldots, x_n\}$ et une fonction f telle que :

$$\begin{aligned} f : \quad & \{0,1\}^n \to \{0,1\}^n \\ & x = (x_1, ..., x_n) \mapsto f(x) = (f_1(x), ..., f_n(x)) \end{aligned}$$

Exemple 1. Un exemple de réseau booléen de dimension 3 ($n = 3$) :

$$\begin{aligned} f_1(x) &= x_3 \vee (\neg x_1 \wedge x_2) \\ f_2(x) &= \neg x_1 \vee x_2 \\ f_3(x) &= x_3 \vee (x_1 \wedge \neg x_2) \end{aligned}$$

Graphe d'interaction : Pour déterminer les RDs, il faut simplifier les interactions entre les gènes et les concentrations requises. Un gène est considéré soit comme actif, soit comme inhibé. Les interactions sont simplifiées de la même manière : un gène inhibe ou active un autre gène, et les échelles de temps sont ignorées. Un *graphe d'interaction* (Déf.2) peut être construit à partir de ces simplifications, les gènes sont les sommets, les interactions sont les arcs orientés, marqués + ou − suivant si l'interaction est une activation ou une inhibition.

Définition 2 (Graphe d'interaction)**.** Un graphe d'interaction $G = (V, E)$ consiste en V l'ensemble des sommets, et $E \subset (V \times V \times \{-, +\})$ l'ensemble des arêtes.

Un circuit dans un ensemble de nœuds $C \subset V$ est dit positif (resp. négatif) si et seulement si il contient un nombre pair (resp. impair) d'arêtes négatives entre ces nœuds.

Un graphe d'interaction peut également être définit comme une abstraction d'un réseau booléen : les fonctions ne sont pas données et pas forcément connues, mais si un sommet u est utilisé dans la fonction f_v alors il y a une arête de u vers v, signée négativement si f_v contient $\neg x_u$ et positivement si f_v contient x_u.

Définition 3 (Graphe d'interaction d'un réseau booléen $(G(f))$)**.** Un graphe d'interaction peut être obtenu à partir d'un réseau booléen f : l'ensemble des sommets est $[n]$, et pour tout $u, v \in [n]$, il y a une arête positive (resp. négative) de u vers v si $f_{vu}(x)$ est positive (resp. négative) pour au moins un $x \in \{0,1\}^n$ (pour tout $u, v \in [n]$, la fonction f_{vu} est la dérivée discrète de f_v par rapport à u, définie sur $\{0,1\}^n$ par $f_{vu}(x) := f_v(x_1, .., x_{u-1}, 1, x_{u+1}, .., x_n) - f_v(x_1, .., x_{u-1}, 0, x_{u+1}, .., x_n)$).

Étant donné un graphe d'interaction $G = (V, E)$ et un de ses sommets $u \in V$, on note P_u l'ensemble des ancêtres de u, i.e. les sommets v tels qu'il existe un chemin dans E de uv à u. De la même manière, p_u est l'ensemble des parents de u, i.e. $v \in p_u \Rightarrow (v, u, s) \in E$. De plus, $G[P_u]$ est le sous-graphe induit de G avec P_u comme ensemble de sommets.

Fig. 1 donne un exemple de graphe d'interaction, qui est égal à $G(f)$, où f est le réseau booléen de l'exemple 1.

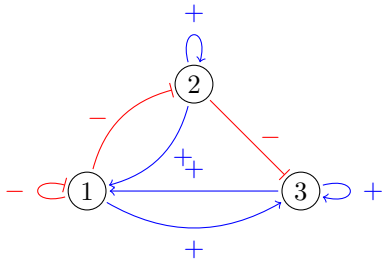


FIGURE 1 – Graphe d'interaction d'Ex.1.
Une flèche bleue indique une activation, une flèche aplatie rouge indique une inhibition.

Graphe de Transition : Il est possible de modéliser la dynamique d'un réseau booléen f à l'aide de *transitions* entre ses états $x \in \{0,1\}^n$. Dans l'ensemble de ce papier, nous considérons la *sémantique asynchrone* des réseaux booléens : une transition met à jour seulement la valeur d'un unique sommet $u \in [n]$. Depuis un état $x \in \{0,1\}^n$, il existe plusieurs transitions pour chaque nœud u tel que $f_u(x) \neq x_u$. Cela permet de définir un *graphe de transition* (Déf.4) où les sommets sont les états possibles $\{0,1\}^n$ et les arêtes correspondent aux transitions asynchrones.

Définition 4 (Graphe de Transition)**.** Le graphe de transition est le graphe ayant $\{0,1\}^n$ comme ensemble de sommets et l'ensemble $\{x \to \bar{x}^{\{u\}} \mid x \in \{0,1\}^n, u \in [n], x_u \neq (f(x))_u\}$ comme l'ensemble des arêtes. Un chemin existant de x vers y est noté $x \to^* y$.

Fig.2 donne le graphe de transition en dynamique asynchrone du réseau booléen de l'exemple 1.

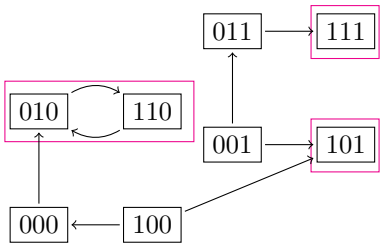


FIGURE 2 – Graphe de transition du réseau booléen défini dans Ex.1. J'utiliserai des notations simplifiées, 010 étant la notation pour quand le sommet 1 a pour valeur 0, le sommet 2 vaut 1, et le sommet 3 vaut 0. Les attracteurs sont encadrés en magenta.

Attracteurs, Points Fixes Les attracteurs d'un réseau booléen sont les composantes fortement connexes terminales du graphe de transition, et donc les dynamiques à long terme du système. La terminologie est différente si l'ensemble des nœuds de la composante contient plusieurs éléments, alors le système oscille entre plusieurs états (c'est un *attracteur cyclique*) ou un unique sommet (*point fixe*).

Définition 5 (Attracteur)**.**

$$S \subseteq \{0,1\}^n \text{ est un attracteur} \Leftrightarrow S \neq \emptyset \quad (1)$$
$$\text{et } \forall x \in S, \forall y \in \{0,1\}^n \setminus S, x \not\to y \quad (2)$$
$$\text{et } \forall x \in S, S \setminus x \text{ ne vérifie pas (2)} \quad (3)$$

Si $|S| = 1$ alors S est un point fixe. Sinon, S est un attracteur cyclique.

Soit f un réseau booléen, $\mathrm{FP}(f) \subseteq \{0,1\}^n$ est l'ensemble de ses points fixes ($\forall x \in \mathrm{FP}(f), f(x) = x$).

Exemple 2. Le réseau booléen de l'exemple 1 a 3 attracteurs, qui correspondent aux trois composantes fortement connexes terminales, encadrées en magenta, dans Fig.2 : $\{010, 110\}$ (attracteur cyclique), $\{101\}$ et $\{111\}$ (points fixes).

2.2 Lien entre attracteurs et graphe d'interaction

Le théorème 1 est une conjecture de René Thomas [15] qui a depuis été prouvé pour les réseaux booléens et discrets [1, 10] : si un réseau booléen a plusieurs attracteurs, alors son graphe d'interaction contient forcément un circuit positif. Dans le cas où plusieurs attracteurs sont des points fixes, toute paire de points fixes prennent des valeurs différentes sur au moins un ensemble de nœuds formant un circuit positif.

Théorème 1. *Si $G = (V, E)$ n'a aucun circuit positif, alors f a au plus un attracteur. De plus, si f a deux points fixes distincts x et y, alors G a un circuit positif $C \subset V$ tel que, pour tout sommet v de C, on a $x_v \neq y_v$.*

Il est également possible de remarquer que pour qu'un sommet v conserve sa valeur y_v, où y est un point fixe, il est suffisant que ses ancêtres aient la même valeur que dans y.

Remarque. $\forall y \in \mathrm{FP}(f), \forall u \in [n], \forall z \in \{0,1\}^n$, *$z$ verifiant* $\forall j \in P_u$, $z_j = y_j$, *on a* $f_u(y) = y_u = f_u(z)$.

Démonstration. Soit u un sommet de $[n]$. $f(u)$ dépend uniquement des arêtes entrantes, donc uniquement de p_u. L'ensemble des sommets de p_u dépendent également uniquement de leurs parents, donc, par récurrence, $f_u(y)$ dépend uniquement de P_u. On en déduit que si $f_u(y) = y_u$ dans G, alors $f_u(y) = y_u$ dans $G[P_u]$ □

3 Formalisation de la reprogrammation de réseaux booléens avec des perturbations permanentes

Soient x et y deux points fixes du réseau booléen f, je veux trouver un ensemble de nœuds, les déterminants de reprogrammation (RDs), qui, quand modifiés dans x, permettent de passer à y. Dans ce rapport, "modifier" veut dire imposer de manière permanente une valeur au nœud. Si l'on modifie u à la valeur 1 (resp. 0) alors on a $f_u(x) = 1$ (resp. 0) pour tout x. Quand atteindre y (en changeant un ensemble I) est possible, deux cas de figure sont envisageables : soit y est atteignable depuis $x_{[x_I=y_I]}$ (atteinte existentielle, Def.6), soit y est l'unique point fixe atteignable depuis $x_{[x_I=y_I]}$ (atteinte inévitable, Def.7). J'ai étudié les deux approches.

Définition 6 (Atteinte existentielle)**.** A partir du réseau booléen F ; une fonction ER_F peut être définie dans $2^{2^{[n]}}$, avec $ER_F(x, y) \mapsto v$ où v est l'ensemble de tous les ensemble minimaux de sommets I tels que $x_{[x_I=y_I]} \rightarrow^* y$.

Définition 7 (Atteinte inévitable)**.** De la même manière, une fonction IR_F : $2^{2^{[n]}}$ peut être définie par $IR_F(x, y) \mapsto v$ où v est l'ensemble de tous les ensemble minimaux de sommets I tels que $\forall z \in \{0, 1\}^n$, $x_{[x_I=y_I]} \rightarrow^* z \Rightarrow z \rightarrow^* y$.

C'est deux fonctions donneront des résultats différents, comme montré dans l'exemple ci-dessous.

Exemple 3. Soit f le réseau booléen de la Fig.3 et dont le graphe de transition est visualisé dans la Fig.4. f a quatre points fixes (en rouge) : 0000, 0001, 1100 et 1101. On pose $x = 0000$ et $y = 1100$. Fixer la valeur du noeud $\{1\}$ a 1 dans x rend y atteignable : 1100 (=y) est atteignable depuis $x_{[x_1=1]} = 1000$ dans le réseau booléen f' défini par $f'_1(x) = 1$ et $f'_2 = f_2$, $f'_3 = f_3$, $f'_4 = f_4$. Le graphe de transition de f' correspond à la partie gauche du graphe de transition de la Fig.4. On peut cependant remarquer que y n'est pas l'unique point fixe atteignable : depuis 1000, 1101 est également atteignable. En fixant le nœud $\{4\}$ à 0, y devient l'unique point fixe atteignable depuis $x_{[x_1=1,x_4=0]}$ dans le réseau booléen f'' tel que $f''_1(x) = 1$, $f''_2 = f_2$, $f''_3 = f_3$, et $f''_4(x) = 0$.

Donc, avec les définitions précédentes, on a $\{1\} \in ER_F(0000, 1100)$ mais $\{1\} \notin IR_F(0000, 1100)$; et $\{1, 4\} \in IR_F(0000, 1100)$ mais $\{1, 4\} \notin ER_F(0000, 1100)$. De plus, on a également $\{1, 2\}$ et $\{1, 3\} \in IR_F(0000, 1100)$.

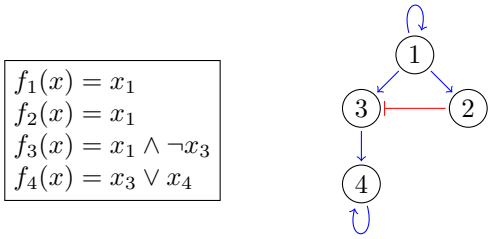


FIGURE 3 – Un réseau booléen de dimension 4

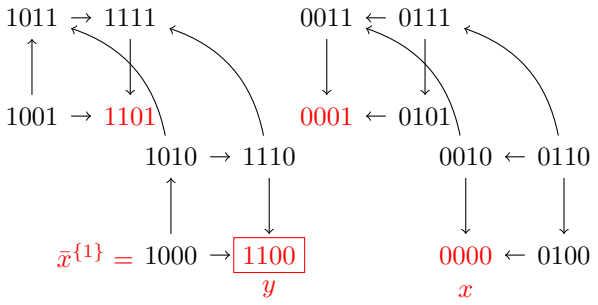


FIGURE 4 – Graphe de transition du réseau booléen de la Fig.3

4 Déterminants de reprogrammation et Composantes fortement connexes (SCC) du graphe d'interaction

Dans cette section, je vais parler du lien entre les RDs et les SCCs du graphe d'interaction du réseau booléen f. Les résultats sont basés sur l'hypothèse que tous les attracteurs de f sont des points fixes (pas d'attracteurs cycliques).

4.1 Trier les SCCs

Pour passer de x à y, on veut changer la valeur de chaque sommet u qui a des valeurs différentes en x et y ($x_u \neq y_u$) et empêcher chaque sommet v qui vérifie $x_v = y_v$ de changer de valeur. On sait que changer la valeur d'un sommet va avoir un effet sur les valeurs des autres sommets (en prenant en compte la dynamique) mais qu'un sommet ne va également influer que ses descendants (par construction du graphe d'interaction, Def.3).

J'en ai donc déduit que, si un sommet u a une valeur différente entre x et y mais qu'aucun de ses ancêtres ne change, alors il est nécessaire de modifier ce sommet. Donc, la meilleure manière de savoir quels sommets doivent être changés en premier est de les trier, avec un ordre topologique par exemple. Évidemment, s'il y a des boucles ou des circuits, un ordre est impossible à déterminer, c'est pour ça qu'il est nécessaire de réduire tous les SCCs à de simples "super-sommets" afin de réussir ce tri. Dans la suite de ce rapport, j'ai uniquement inclus les SCCs qui contiennent des circuits positifs, car ce sont ceux qui changent entre les points fixes (Theor.1), et on appelle $\mathcal{O}$ l'ensemble de tous ces SCCs. Réduire le graphe à ses SCCs permet de les ordonner de 1 à k à l'aide de n'importe quel ordre topologique, noté $\prec$: pour tout $i, j \in [k], j > i \Rightarrow \mathcal{O}_j \not\prec \mathcal{O}_i$.

Soit $\mathcal{O}_0$ l'ensemble $\{\mathcal{O}_i \in \mathcal{O} \mid \nexists \mathcal{O}_j, \mathcal{O}_j \prec \mathcal{O}_i\}$, qui définit la première "tranche" de SCCs (ceux qui n'ont aucun parents),on peut définir récursivement les tranches suivantes $C_K = \{\mathcal{O}_i \in (\mathcal{O} \setminus \bigcup_{l\in\{1,..,K-1\}} C_l) \mid \nexists \mathcal{O}_j, \mathcal{O}_j \prec \mathcal{O}_i\}$. Avec cette définition de tranche, quelque soit l'ordre topologique, les tranches et leur ordre seront les mêmes. On indexe les tranches de 1 à c.

À partir de cet ordre, on sait quels SCCs doivent être modifiés en premier, mais ce n'est pas suffisant pour ne pas changer les SCCs placés plus bas dans la hiérarchie (cf ex.4). La relation $\prec$ ne donne qu'un ordre dans lequel modifier les nœuds, duquel il est possible de déterminer si plus de modifications sont nécessaires.

Exemple 4. Preuve que l'ordre topologique uniquement n'est pas suffisant.

Tout algorithme qui n'utiliserait que l'ordre topologique sans aller plus loin dans la recherche de points fixes atteignables ne suffirait pas, comme l'exemple de la Fig.5 le montre : la modification nécessaire à passer de 01100 à 10101 serait calculée comme étant uniquement le nœud {1} à modifier, mais {4} sera toujours à 0, car {4} est toujours inhibé par {3}, donc {5} doit également être changé.

$$
\begin{aligned}
f_1(x) &= \neg x_2 \\
f_2(x) &= \neg x_1 \\
f_3(x) &= x_1 \vee x_2 \\
f_4(x) &= x_2 \wedge \neg x_3 \\
f_5(x) &= x_4 \vee x_5
\end{aligned}
$$

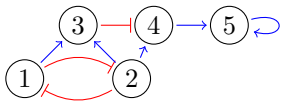


FIGURE 5 – Réseau booléen empêchant le changement dans le SCC le plus bas

4.2 Filtrer les SCCs

Pour que y soit l'unique point fixe ou pour qu'il soit seulement un des attracteurs atteignables, l'ordre de la partie précédente sera le même, mais filtrer les SCCs sera différent.

Théorème 2. *Si un sommet u tel que $x_u \neq y_u$ et que u n'est pas dans un cycle positif, alors modifier les ancêtres de u est suffisant pour modifier u.*
Plus généralement, pour passer de x à y, uniquement modifier les SCCs qui contiennent au moins un circuit positif est suffisant.

Démonstration. Soit u un sommet tel que $x_u \neq y_u$ et u n'est pas dans un circuit positif. Si u est dans un circuit négatif, l'arc entrant provenant du circuit négatif n'influe pas sur u, car x et y sont des points fixes et que u a une valeur distincte dans chacun. Comme u n'est pas dans un circuit positif, u n'est pas dans un SCC de $\mathcal{O}$ (ou, si il est dans un circuit négatif dans un SCC de $\mathcal{O}$, enlever l'arc "inactif" entrant permet de l'en séparer). Donc aucun des ancêtres de u n'est également un de ses descendants. Soit z l'état où tous P_u (tous les ancêtre de u) ont la même valeur qu'en y. Par la remarque de la Sect.2.2, pour tout $v \in G[P_u]$, on a $f_v(z) = z_v = y_v$. Donc, soit $f_u(z) = y_u$, et le théorème est prouvé, soit $f_u(z) \neq y_z$ et alors, par Theor.1, u est dans un circuit positif, contradiction. □

Par récurrence sur la première partie, modifier tous les SCCs qui contiennent des circuits positifs afin que leurs sommets prennent les mêmes valeurs qu'en y modifie tous leurs enfants, et leurs descendants, jusqu'à ce que tout le graphe ait les mêmes valeurs qu'en y. □

La sélection des SCCs est différente suivant le résultat voulu. Je m'appuie sur les mêmes bases, chercher le SCC le plus "haut" (avec le moins d'ancêtres) qui devrait avoir ses valeurs modifiées et qui n'est pas déjà sélectionné.

Pour diviser le problème, les algorithmes présentés vont sélectionner des SCCs à modifier, et non des nœuds. C'est un autre algorithme dont je discuterai plus tard. L'ensemble sélectionné de SCCs est noté $\mathcal{S}$.

4.3 Filtrer les SCCs pour l'Atteinte existentielle

Cette partie explique un algorithme qui permet de trouver différents ensembles de SCCs qui permettent d'atteindre le point fixe cible, et donne aussi certaines propriétés de cet algorithme (complétude et minimalité).

Le principe de l'algorithme est de regarder de manière linéaire les tranches C_i définies par $\prec$, et d'ajouter les combinaisons minimales de SCCs à $\mathcal{S}$ sont différentes entre y et les points fixes atteignables depuis $x_{[x_S=y_S]}$:

1. $\mathcal{S} := \emptyset$
2. Pour i de 1 à c :
   — $T := \emptyset$
   — $\forall s \in P(C_i)$ tel que s minimal
   $\exists z \in \{0,1\}^n$, $z_{C_i \setminus s} = y_{C_i \setminus s}$, $x_{[x_I=y_I | I \in \mathcal{S}]} \rightarrow^* z$, $T := T \cup s$.
   — $\mathcal{S} := \mathcal{S} \bar{\times} T$.

Avec $\bar{\times}$ défini comme un mélange de produit et d'union : soit un ensemble I d'ensembles $I_1, .., I_k$ et un ensemble J d'ensembles $J_1, .., J_l$, ce produit $\bar{\times}$ est défini par : $I \bar{\times} J = \{I_1 \cup J_1, .., I_1 \cup J_l, I_2 \cup J_1, ..., I_k \cup J_l\}$

Complexité Dans le pire des cas, l'algorithme ci-dessus effectue $c \times 2^l$ tests d'atteinte (PSPACE-complets, [4]) avec l la taille de la plus grande tranche.

Existence d'une solution et preuve de correction En imposant à tous les SCCs qui sont différents entre x et y, d'avoir la valeur qu'ils ont en y, on a une solution (par Theor.2. Dans le pire des cas, c'est la solution trouvée par l'algorithme. Comme l'algorithme test l'atteinte possible, et qu'une solution existe, il en trouvera une.

Exemple 5. On applique l'algorithme au réseau booléen de Fig.6 avec $x = 00000$ et $y = 11011$.

$f_1(x) = x_1$
$f_2(x) = x_1$
$f_3(x) = x_1 \wedge \neg x_2$
$f_4(x) = x_3 \vee x_4$
$f_5(x) = x_2 \vee x_5$

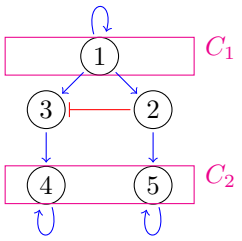


FIGURE 6 – Réseau booléen de dimension 5 (à gauche) et son graphe d'interaction (à droite). Les tranches sont encadrées. $C_1 = \{\{1\}\}$, $C_2 = \{\{4\}, \{5\}\}$.

1. $\mathcal{S} := \emptyset$
2. $C_1 : s$ minimal $\Leftrightarrow\ s = \{1\}$
3. $\mathcal{S} := \mathcal{S} \bar{\times} \{1\} = \{\{1\}\}$
4. $C_2 : s$ minimal $\Leftrightarrow\ s = \emptyset$ [1]

1. en prenant le chemin 10000 → 10100 → 10110 → 11110 → 11111 (et le point fixe est l'état suivant, → 11011 mais l'algorithme n'a pas besoin d'aller plus loin que 11111.)

5. $\mathcal{S} := \mathcal{S} \bar{\times} \emptyset = \{\{1\}\}$.

J'ai prouvé que cet algorithme est complet et que les ensembles de SCCs renvoyés sont minimaux (dans le Theor.3) et que tout RD dans $ER(x, y)$ est nécessairement dans un des ensembles de SCCs identifiés par l'algorithme (Theor.4).

Théorème 3. *$\mathcal{S}$ ne contient que des ensembles minimaux d'ensembles de SCCs, et $\mathcal{S}$ est complet*

Démonstration. **Minimalité :** Dans chaque tranche, les SCCs sont indépendants les uns des autres (par définition de la tranche). De plus, étant donné l'utilisation de l'ordre, on peut en déduire que la somme des minima sur chaque tranche est le minimum sur tout le graphe. □

Complétude : Soit I un ensemble de SCCs minimal, tel que $x_{[x_J = y_J | J \in I]} \rightarrow^* y$. Alors, pour toute tranche C_i, $I \cap C_i$ est minimal, car si on peut changer les SCCs de manière à ce qu'ils soient tous avec les mêmes valeurs qu'en y, on peut toujours choisir de prendre le chemin (dans le graphe de transition) qui permet ce changement. Donc, par définition de $\mathcal{S}$, $I \in \mathcal{S}$. □

Théorème 4. $\forall c \in ER(x, y), \exists I \in \mathcal{S}, \forall u \in c, \exists scc \in I, u \in scc.$

Démonstration. Soit c un ensemble de sommets de $ER(x, y)$n et u un des sommets. Si $u \notin \mathcal{O}$ alors c n'est pas minimal, par Théor.2. Si pour tout $I \in \mathcal{S}$, u est dans $o \in (\mathcal{O} \setminus I)$ alors il existe un chemin tel que changer les ancêtres de o permet de changer o, et donc les ancêtres doivent être changés également, par construction de I. Donc $c \setminus u$ aurait le même effet, et c ne serait pas minimal. Si $u \notin o$ alors il existe $I \in \mathcal{S}$ et $scc \in I$ tel que $u \in scc$. □

4.4 Filtrer les SCCs pour l'atteinte inévitable

Je donne ici l'algorithme qui permet de trouver les ensembles de SCCs qui permettent l'atteinte inévitable du point fixe cible.

L'algorithme recherche tous les points fixes atteignables depuis $x_{x_I = y_I | I \in \mathcal{S}}$ et trouve z, celui qui a le SCC le plus petit au sens de $\prec$ tel que z possède un sommet u ayant des valeurs différentes en z et y ($z_u \neq y_u$). Comme l'on considère l'ensemble des points fixes atteignables, le z retourné sera toujours le même, permettant ainsi à l'algorithme d'être déterministe. On ajoute z à $\mathcal{S}$ jusqu'à ce que y soit le seul point fixe atteignable.

1. $\mathcal{S} := \emptyset$
2. Tant que $\exists z \in \mathrm{FP}(f)$, $z \neq y$, $x_{[x_I = y_I | I \in \mathcal{S}]} \rightarrow^* z$
   — $\mathcal{S} := \mathcal{S} \cup \{\mathcal{O}_i\}$, avec
   $i = min_{a \in \{1,..,k\}}(a \mid \exists z \in \mathrm{FP}(f),\ z_{\mathcal{O}_a} \neq y_{\mathcal{O}_a},\ x_{[x_I = y_I | I \in \mathcal{S}]} \rightarrow^* z)$

Si deux (ou plus) SCCs A et B ne sont pas dans le même ordre si on prend deux ordres topologiques différents, alors ils sont dans la même tranche, et donc n'ont pas d'influence l'un sur l'autre. Donc si ils doivent tous deux être modifiés, l'ordre de sélection de l'algorithme ne changera rien, et ils seront tous les deux ajoutés à $\mathcal{S}$.

Existence d'une solution et preuve de correction Une solution est d'imposer à tous les SCCs du graphe d'interaction leur valeur en y. Comme il existe une solution, que l'algorithme cherche si y est le seul point fixe, et que l'algorithme suit l'ordre topologique, l'algorithme finit et renvoie une solution.

Complexité Chercher tous les points fixes atteignables est PSPACE-complet [3]. On l'utilise k fois dans le pire des cas.

Exemple 6. On applique le théorème au réseau booléen de Fig.7. avec $x = 00000$ et $y = 11011$. À partir de $\mathcal{S} := \emptyset$, le seul point fixe atteignable est (0)00(0)(0) (les SCCs de $\mathcal{O}$ sont entre parenthèses). Le plus petit SCC o vérifiant $x_{[x_\mathcal{S}=y_\mathcal{S}],o} \neq y_o$ est $\mathcal{O}_1$, donc $\mathcal{S} := \emptyset \cup \{\mathcal{O}_1\} = \{\mathcal{O}_1\}$. Les points fixes atteignables depuis $x_{[x_I=y_I|I\in\mathcal{S}]} = 10000$ sont maintenant : (1)10(1)(1) et (1)10(0)(1). Le plus petit SCC o vérifiant $x_{[x_\mathcal{S}=y_\mathcal{S}],o} \neq y_o$ est $\mathcal{O}_3$. On pose $\mathcal{S} := \mathcal{S} \cup \mathcal{O}_3 = \{\mathcal{O}_1, \mathcal{O}_3\}$ et on a alors que le seul point fixe atteignable depuis $x_{[x_I=y_I|I\in\mathcal{S}]} = 10010$ est (1)10(1)(1) $= y$. L'algorithme s'arrête donc ici.

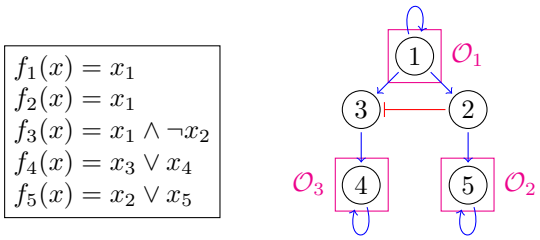


FIGURE 7 – Réseau booléen de dimension 5 (à gauche) et son graphe d'interaction (à droite) sur lequel les SCCs contenant des cycles positifs ($\mathcal{O}$) sont encadrés.

Théorème 5. *$\mathcal{S}$ est minimal.*

Démonstration. Si un ensemble S_1 existe tel que S_1 a un cardinal plus faible que $\mathcal{S}$ et que modifier S_1 permet que y soit l'unique point fixe atteignable, alors il est possible de réduire S_1 à un sous-ensemble de $\mathcal{O}$. Soit s un SCC dans $\mathcal{S} \setminus S_1$, donc il existe un point fixe z tel que $z_s \neq y_s$ et donc, par construction de $\mathcal{S}$, z est atteignable depuis x modifié par S_1. □

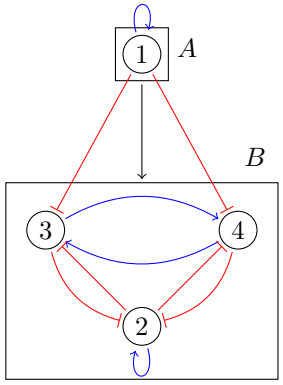
Il est possible de remarquer que, contrairement au cas de l'atteinte existentielle, les RDs pour l'atteinte inévitable ne sont pas forcément dans des SCCs contenant des circuits positifs. En effet, dans l'exemple 3, j'ai montré que $IR_F(x, y)$ peut utiliser des nœuds qui ne sont pas dans $\mathcal{O}$ (comme le nœud $\{2\}$ de la Fig.3). On peut également remarquer que si un RD v n'est pas dans un SCC contenant un cycle positif, alors $x_v = y_v$.

4.5 L'impossibilité d'enlever les tests d'accessibilité

Je me suis demandé si enlever les tests d'accessibilité est possible. Le principe d'un algorithme serait sensiblement le même, mais en comparant les points modifiés à la liste de points fixes du réseau booléen. En simplifiant les SCCs en super-nœuds avec de multiples états possibles (tous les états possibles dans les points fixes), on aurait un automate ordonné assez simple et comparer les états de cet automate aux points fixes simplifiés est trivial.

Cependant, cette méthode ne donne pas des résultats minimaux, comme le montre l'exemple 7 ci-dessous.

$$
\begin{aligned}
f_1(x) &= x_1 \\
f_2(x) &= x_2 \wedge \neg x_3 \wedge \neg x_4 \\
f_3(x) &= x_4 \wedge \neg x_2 \\
f_4(x) &= x_3 \wedge \neg x_2
\end{aligned}
$$



Exemple 7.

FIGURE 8 – Réseau booléen de dimension 4 (à gauche) et son graphe d'interaction (à droite). Une simplification des SCCs et des arêtes est également effectuée, en noir, donnant deux SCCs A et B et leur ordre $A \prec B$.

Le réseau booléen de Fig.8 donne les points fixes suivants :

fp	a	b	c	d	e
1 :	0	0	0	1	1
2 :	0	0	1	0	1
3 :	1	0	0	0	0
4 :	1	0	0	0	0

Simplifiable en :

fp	a	b	c	d	e
A :	0	0	0	1	1
B :	0	1	2	1	2

On choisit $x = a$ et $y = d$. On doit modifier $\{1\}$ pour changer A. Avec l'algorithme qui ne fait pas les tests d'accessibilités, on a d comme e qui sont des points fixes existants, mais e n'est pas accessible à partir de $\bar{x}^1$. Donc, changer uniquement $\{1\}$ est suffisant, mais l'algorithme sans tests d'accessibilités, on trouverait qu'il faut changer A et B.

Conjecture :

Tout ensemble de SCCs renvoyé par cette méthode est un ensemble de $\mathcal{S}$ auquel des éléments ont possiblement été ajoutés.

5 Identifier les déterminants au sein des SCCs

J'ai prouvé que modifier tous les SCCs sélectionnés dans $\mathcal{S}$ permet de passer de x à y, mais, pour réduire le nombre de gènes sélectionnés (surtout dans le cas où les SCCs sont grandes), il est possible d'essayer de modifier uniquement certains des sommets pour atteindre le même résultat. Cependant, à cause de l'aspect dynamique, il est plutôt facile d'avoir des changements non voulus (ou des changements voulus non prédits, dans le cas de l'atteinte existentielle).

Une idée est de sélectionner le Feedback Vertex Set du SCC, l'ensemble minimal de sommets tel que, privé de ces sommets, le SCC ne contient plus aucun circuit positif. En modifiant les sommets de cet ensemble, tout cycle serait effectivement détruit (car comme on change la fonction en fixant un sommet, on supprime toutes les arrêtes entrantes), et donc le seul point fixe atteignable serait y (par Theor.1). Cependant, le problème de la dynamique se pose encore, comme le démontre l'exemple 8. De plus, cette méthode ne renvoie pas toute les solutions (modifier $\{2\}$ ou $\{3\}$ dans l'ex.8 permet de modifier tout le SCC) et même peut ne pas renvoyer la meilleure solution (dans l'ex.8, modifier uniquement $\{3\}$ permet à y d'être le seul point fixe atteignable et de résoudre le problème).

Exemple 8. Illustration du problème au sein d'un SCC en prenant en compte la dynamique.

$$
\begin{aligned}
f_1(x) &= \neg x_3 \wedge \neg x_2 \\
f_2(x) &= \neg x_1 \\
f_3(x) &= \neg x_1 \\
f_4(x) &= x_2 \wedge \neg x_1 \wedge \neg x_3 \\
f_5(x) &= x_4 \vee x_5
\end{aligned}
$$

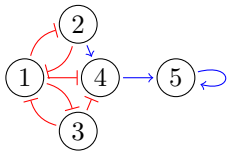


FIGURE 9 – Réseau booléen de dimension 5 et son graphe d'interaction

Soient $x = 10000$ et $y = 01100$, et $01101 = z$, qui sont tous trois des points fixes. Supposons que l'on veut qu' y soit l'unique point fixe atteignable. L'algorithme trouve que si le premier SCC $\{1, 2, 3\}$ est modifié, alors y est l'unique point fixe atteignable. En modifiant le Feedback Vertex Set, $\{1\}$, mais plutôt que d'avoir le chemin $00000 \to 00100 \to 01100$, on peut avoir

$$00000 \to 01000 \to 01010 \to 01011 \to 01111 \to 01101$$

Ce qui permet à z d'être atteignable en modifiant uniquement $\{1\}$, et qui donc rend l'algorithme faux.

Pour résoudre le problème, il est donc nécessaire, soit de trouver une méthode de sélection des sommets qui empêche d'avoir des répercussion sur les descendants en dehors du SCC (en s'intéressant particulièrement aux sommets ayant des arrêtes sortantes du SCC par exemple), soit de transformer $\mathcal{S}$ en liste de sommets et de chercher les sommets à modifier dans le SCC en tant qu'étape intermédiaire dans les algorithmes précédents.

En utilisant les résultats de la section précédente, j'ai un algorithme basique qui permet de trouver un ensemble de RD qui garantit l'accessibilité inévitable du point fixe cible. L'algorithme choisit récursivement un sommet u dans le SCC le plus bas relativement à l'ordre $\prec$ et modifie sa fonction pour qu'elle vaille la constante y_u. Le graphe d'interaction du réseau booléen résultant est le sous-graphe du graphe d'interaction initial, mais où toutes les arêtes entrantes vers u ont été supprimées. Donc le SCC $\mathcal{O}_1$ est divisé dans le nouveau graphe. Si nécessaire, un nouveau sommet peut être choisi dans le SCC le plus faible :

RecursiveAlgorithm(f, rd) :
- Si $\exists z \in \mathrm{FP}(f)$, $x_{[x_{rd}=y_{rd}]} \to^* z$ alors :
  - $res = \emptyset$
  - $i = min_{a \in \{1,..,k\}}(a \mid \exists z \in \mathrm{FP}(f),\, z_{\mathcal{O}_a} \neq y_{\mathcal{O}_a},\, x_{[x_I=y_I | I \in \mathcal{S}]} \to^* z)$
  - Pour tout $u \in \mathcal{O}_i$:
    - $g := f$ avec $g_u := y_u$
    - $res := res \cup$ RecursiveAlgorithm(g, $rd \bar{\times} \{u\}$)
  - renvoyer res
- sinon :
  - renvoyer rd

On peut remarquer que l'algorithme trouve toujours une solution : si le point fixe cible n'est pas l'unique point fixe atteignable, alors il y a au moins un circuit positif (et donc un SCC) qui a des valeurs différentes entre y et un des points fixes atteignables (et sera donc choisi par l'algorithme).

Exemple 9. Appliqué au réseau booléen de la Fig.9 avec $x = 10000$ et $y = 01100$, l'algorithme ci-dessus renvoie, par exemple, l'ensemble $\{2, 5\}$. En effet, $\{2\}$ est dans $\mathcal{O}_1$. En fixant $f_2 = 1$, le nouveau graphe d'interaction a deux SCCs contenant des cycles positifs : $\{1, 3\}$ et $\{5\}$. Depuis l'état 11000, deux points fixes sont atteignables dans ce nouveau réseau booléen, 01100 et 01101. Comme le SCC $\{1, 3\}$ a les mêmes valeurs qu'en y dans ces deux points fixes, le SCC sélectionné par l'algorithme est $\{5\}$. Après, depuis 11001 le seul point fixe accessible est y.

6 Méthodes

Afin d'implémenter un algorithme, quel que soit le langage de programmation utilisé, il est nécessaire de pouvoir calculer les composantes fortement connexes et de pouvoir faire des tests d'atteinte. J'ai décidé d'écrire une version moins aboutie de l'algorithme de recherche des RDs pour l'atteinte inévitable en OCaml. Le code source est donné dans les fichiers joints.

Le calcul des composantes fortement connexes est fait en temps linéaire par rapport au nombre de sommets du graphe, par la méthode de Tarjan, qui est implémentée dans la bibliothèque de graphes en OCaml.

Pour ce qui est du calcul des points fixes atteignables à partir d'un état (les tests d'atteinte), deux méthodes existent : calculer tous les points fixes (problème NP-complet), puis vérifier lesquels de ces points sont atteignables en utilisant des méthodes PSPACE-complètes de model-checking ; ou alors calculer directement la dynamique et en extraire les attracteurs ce qui est également PSPACE-complet[3]. J'ai utilisé l'outil "pint" de Loïc Paulevé pour calculer les points fixes atteignables.

Le code source en OCaml en annexe nécessite pint d'installé, ainsi qu'un compilateur OCaml avec les bibliothèques Graph, Unix, Array, et List.

7 Discussion

Ce rapport donne la première caractérisation formelle des déterminants de reprogrammation pour passer d'un point fixe à un autre dans la dynamique asynchrone d'un réseau booléen.

Dans le cas de la reprogrammation pour l'atteinte existentielle, j'ai prouvé que tous les ensembles minimaux possibles de RDs sont dans certains SCCs particuliers du graphe d'interaction, et j'ai donné un algorithme permettant de déterminer cette exacte combinaison de nœuds. Cette caractérisation s'appuie sur les vérifications de l'atteinte possible du point fixe cible.

Dans le cas de la reprogrammation pour l'atteinte inévitable, j'ai montré que les RDs ne sont pas forcément dans ces SCCs particuliers. Cependant, l'algorithme donné garantit l'atteinte inévitable du point fixe en trouvant des RDs dans ces SCCs. Cet algorithme se base sur l'énumération des points fixes.

J'ai également prouvé que les tests d'accessibilité sont indispensables pour avoir un ensemble de SCCs (et donc par conséquent de RDs) minimal.

Une des limites majeures de ces algorithmes est donc les nombreux tests d'atteinte qu'ils ont besoin d'effectuer. La poursuite de ce stage en doctorat me permettra de me pencher sur la factorisation possible de l'exploration de la dynamique des réseaux booléens.

Une autre variable à ne pas oublier est la limitation de ce travail aux changements permanents : quand un nœud est changé, il ne peut pas revenir à sa valeur précédente et gardera donc sa nouvelle valeur de manière permanente. Étudier les mutations temporaires (où le noeud reviendrait à sa fonction initiale après un certain "temps") serait donc intéressant, mais également bien plus difficile, car il faut alors prendre en compte l'ordre des mutations, leur durée, et les RDs ne seraient plus *a priori* uniquement dans les composantes fortement connexes contenant des circuits positifs, comme c'était le cas pour les mutations permanentes.

Une autre problématique est l'ordonnancement des mutations, qui a un impact à la fois pour les mutations permanentes et temporaires. L'exemple 10 donne une idée de l'importance de l'ordre des mutations et de la notion de durée.

Exemple 10. Sur le réseau booléen de la fig.10, en choisissant $x = 0000$ et $y = 1101$, un algorithme qui ne prend pas en compte le temps est forcé de modifier $\{1, 2, 4\}$ si l'on cherche à avoir une atteinte inévitable. Cependant, si l'on modifie $\{1\}$ et que l'on attend "suffisamment longtemps", il suffit ensuite de modifier uniquement $\{2\}$. "Attendre suffisamment longtemps" dans ce contexte est attendre que la succession d'états $1000 \to 1010 \to 1011$ soit effectuée.

$$f_1(x) = x_1$$
$$f_2(x) = x_2$$
$$f_3(x) = x_1 \wedge \neg x_2$$
$$f_4(x) = x_3 \vee x_4$$

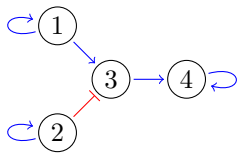


FIGURE 10 – Réseau booléen de dimension 4 et son graphe d'interaction

8 Acquis du stage

Cette section rapporte mes acquis pendant le stage.

8.1 Méthodes de recherche

Tout au long du stage, l'objectif a été de trouver des théorèmes qui donneraient un lien entre les RDs et le graphe d'interaction. J'ai donc appris les bases des techniques de recherche d'article, que ce soit trouver les articles constructifs sur le sujet ou la lecture de ces articles. J'ai également appris les techniques sur les résultats : la discussion de ceux qui sont possibles, de ceux intéressants, de la preuve de ces résultats mais aussi comment trouver des contre-exemples.

8.2 Algorithmique et programmation

Comme je cherche à avoir une méthode algorithmique pour trouver les RDs, j'ai eu à me pencher sur l'écriture de programmes et le calcul de complexité. Cela m'a permis de me perfectionner en OCaml, et également à chercher l'optimisation des temps de calcul maximaux.

8.3 Réseau

Des conférences et ateliers ont eu lieu pendant mon stage, auxquels j'ai été invité. Cela m'a permis de connaitre un peu plus les personnes qui travaillent dans le cadre des réseaux booléens et des graphes d'interaction, et d'élargir ma culture sur leurs usages.

8.4 Rédaction scientifique et présentation de résultats

En plus du rapport de stage, qui se présente sous une forme assez proche de celle de l'article scientifique, j'ai également eu à rédiger un article pour la conférence de Hybrid Systems Biology qui aura lieu en octobre. J'ai donc appris à formater et à rédiger sous un style scientifique, ainsi que comment rédiger en LaTeX.

De plus, j'ai parfois eu à présenter mes résultats, ce qui m'a permis d'améliorer mes performances de présentation orale, mais également de savoir introduire mon sujet de manière précise et concise et de développer des exemples pour faciliter la compréhension.

8.5 Inscription dans le parcours professionnel

Je souhaite devenir enseignant-chercheur, donc ce stage m'a permis de commencer la recherche.De plus, je prolonge ces recherchs en doctorat, ce qui me permettra de pousser plus loin autant dans mes activités de recherche que dans ma volonté d'enseigner.

Références

[1] Julio Aracena. Maximum number of fixed points in regulatory boolean networks. *Bulletin of Mathematical Biology*, 70(5) :1398–1409, feb 2008.

[2] Rui Chang, Robert Shoemaker, and Wei Wang. Systematic search for recipes to generate induced pluripotent stem cells. *PLoS Computational Biology*, 7(12) :e1002300, Dec 2011.

[3] Thomas Chatain, Stefan Haar, Loïg Jezequel, Loïc Paulevé, and Stefan Schwoon. Characterization of reachable attractors using Petri net unfoldings. In Pedro Mendes, Joseph Dada, and Kieran Smallbone, editors, *Computational Methods in Systems Biology*, volume 8859 of *Lecture Notes in Computer Science*, pages 129–142. Springer Berlin Heidelberg, 2014.

[4] Allan Cheng, Javier Esparza, and Jens Palsberg. Complexity results for 1-safe nets. *Theor. Comput. Sci.*, 147(1&2) :117–136, 1995.

[5] Isaac Crespo, Thanneer M Perumal, Wiktor Jurkowski, and Antonio del Sol. Detecting cellular reprogramming determinants by differential stability analysis of gene regulatory networks. *BMC Systems Biology*, 7(1) :140, 2013.

[6] Antonio del Sol and Noel J. Buckley. Concise review : A population shift view of cellular reprogramming. *STEM CELLS*, 32(6) :1367–1372, 2014.

[7] Zuguang Gao, Xudong Chen, and Tamer Başar. On the stability of conjunctive boolean networks. March 2016.

[8] Thomas Graf and Tariq Enver. Forcing cells to change lineages. *Nature*, 462(7273) :587–594, dec 2009.

[9] Junghyun Jo, Sohyun Hwang, Hyung Joon Kim, Soomin Hong, Jeoung Eun Lee, Sung-Geum Lee, Ahmi Baek, Heonjong Han, Jin Il Lee, Insuk Lee, and et al. An integrated systems biology approach identifies positive cofactor 4 as a factor that increases reprogramming efficiency. *Nucleic Acids Research*, 44(3) :1203–1215, Jan 2016.

[10] Élisabeth Remy, Paul Ruet, and Denis Thieffry. Graphic requirements for multistability and attractive cycles in a boolean dynamical framework. *Advances in Applied Mathematics*, 41(3) :335 – 350, 2008.

[11] Owen J L Rackham, Jaber Firas, Hai Fang, Matt E Oates, Melissa L Holmes, Anja S Knaupp, Harukazu Suzuki, Christian M Nefzger, Carsten O Daub, Jay W Shin, Enrico Petretto, Alistair R R Forrest, Yoshihide Hayashizaki, Jose M Polo, and Julian Gough. A predictive computational framework for direct reprogramming between human cell types. *Nature Genetics*, 48(3) :331–335, jan 2016.

[12] Adrien Richard. Positive circuits and maximal number of fixed points in discrete dynamical systems. *Discrete Applied Mathematics*, 157(15) :3281 – 3288, 2009.

[13] Adrien Richard. Negative circuits and sustained oscillations in asynchronous automata networks. *Advances in Applied Mathematics*, 44(4) :378 – 392, 2010.

[14] Kazutoshi Takahashi and Shinya Yamanaka. A decade of transcription factor-mediated reprogramming to pluripotency. *Nat Rev Mol Cell Biol*, 17(3) :183–193, Feb 2016.

[15] René Thomas. Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, 42(3) :563 – 585, 1973.