



HAL
open science

Modélisation grande échelle de réseaux biologiques : vérification par contraintes booléennes de la cohérence des données

Philippe Veber

► **To cite this version:**

Philippe Veber. Modélisation grande échelle de réseaux biologiques : vérification par contraintes booléennes de la cohérence des données. Génie logiciel [cs.SE]. Université Rennes 1, 2007. Français. NNT: . tel-00185895v1

HAL Id: tel-00185895

<https://theses.hal.science/tel-00185895v1>

Submitted on 7 Nov 2007 (v1), last revised 29 Feb 2008 (v3)

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

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

N° d'ordre: 3684

THÈSE

présentée

devant l'Université de Rennes 1

pour obtenir

le grade de : DOCTEUR DE L'UNIVERSITÉ DE RENNES 1
Mention INFORMATIQUE

par

Philippe VEBER

Équipe d'accueil : SYMBIOSE - IRISA

École Doctorale : Matisse

Composante universitaire : IFSIC

Titre de la thèse :

*Modélisation grande échelle de réseaux biologiques :
vérification par contraintes booléennes de la cohérence des données*

À soutenir le 17 décembre 2007 devant la commission d'examen

M. :		Président
MM. :	Alexander BOCKMAYR	Rapporteurs
	Jean-Paul COMET	
MM. :	Laurent TRILLING	Examineurs
	François KÉPÈS	
	Michel LE BORGNE	
	Rumen ANDONOV	
Mme. :	Anne SIEGEL	Membre invité

Table des matières

Table des matières	i
1 Introduction	1
1.1 Motivations	1
1.2 Approche suivie	2
1.3 Sur les données	4
1.4 Travaux connexes	8
1.4.1 Panorama	8
1.4.2 Modèles physiques de réseaux biologiques	10
1.5 Nos contributions	11
2 Présentation générale de l’approche : modélisation de l’opéron lactose	13
2.1 Vérification	17
2.2 Prédiction	18
3 Équations qualitatives pour la consistance des données	21
3.1 Formalisation	21
3.1.1 Le graphe d’interaction et ses étiquetages	21
3.1.2 Algèbre des signes	23
3.1.3 Contrainte de consistance	25
3.1.4 Propriétés des contraintes qualitatives	27
3.2 Justification différentielle	29
3.2.1 Graphe d’interaction	30
3.2.2 Réponse statique à une perturbation	31
3.2.3 Hypothèses de modélisation	32
3.2.4 Déplacement d’équilibre et variations	33
3.2.5 Discussion	35
3.2.6 Cinétiques usuelles en modélisation	37
3.2.7 Graphes de réactions	38
3.3 Justification booléenne	41
3.3.1 Graphe d’interaction	41
3.3.2 Déplacement d’équilibre	42

4	Résolution par diagrammes de décision	45
4.1	Diagrammes de décision	46
4.1.1	Définition	46
4.1.2	Opérations sur les diagrammes	47
4.1.3	Fonctions à variables dans un ensemble fini	53
4.2	Problème de vérification	54
4.2.1	Diagramme associé à une contrainte qualitative	54
4.2.2	Algorithme pour la vérification	57
4.3	Problème de prédiction	57
4.3.1	Invariant de l'ensemble des modèles	57
4.3.2	Marginales	59
4.4	Diagnostic des contraintes non satisfiables	63
4.4.1	Données bruitées	64
4.4.2	Reconstruction de réseau	64
4.4.3	Recherche des sous-systèmes incompatibles	65
4.4.4	Calcul des diagnostics	65
4.5	Réduction, décomposition des systèmes	65
4.5.1	Réduction préservant l'existence de solution	67
4.5.2	Décomposition	68
4.5.3	Calcul de la consistance selon une décomposition	69
4.5.4	Calcul des invariants selon une décomposition	70
4.5.5	Choix de la décomposition	72
5	Résolution par Answer Set Programming	75
5.1	Une introduction à la programmation par ensemble réponse	75
5.1.1	Syntaxe	76
5.1.2	Sémantique des modèles stables	77
5.1.3	Variables	80
5.1.4	Contraintes d'intégrité	81
5.1.5	Contraintes de cardinalité	82
5.1.6	Optimisation	83
5.1.7	Complexité et résolution	83
5.2	Consistance aux sommets	84
5.2.1	Codage des données	85
5.2.2	Génération des solutions	85
5.2.3	Test des solutions	86
5.3	Prédiction	86
5.4	Contrainte non satisfiable	87
6	Validation expérimentale	89
6.1	Prédiction de la réponse à une perturbation	89
6.1.1	Construction du graphe d'interaction	90
6.1.2	Confrontation aux données d'expression issues de la littérature, premier essai	90

6.1.3	Diagnostic par isolement des défauts	91
6.1.4	Ajout des facteurs σ dans le modèle	92
6.1.5	Prédiction de la réponse globale au stress nutritionnel	93
6.2	Inférence de graphes d'interactions	97
6.2.1	Limites théoriques de l'approche	97
6.2.2	Validation par des mesures d'expression	99
6.2.3	Application chez <i>S. cerevisiae</i>	100
7	Discussion	103
7.1	Travaux connexes	103
7.1.1	Circuits du graphe d'interaction	103
7.1.2	Régulons	104
7.1.3	Chemins métaboliques	104
7.1.4	Cascades de régulations	105
7.1.5	Bilan	105
7.2	Chemins dans le graphe d'interaction	106
7.2.1	Le modèle de Yeang-Ideker-Jaakkola (YIJ)	106
7.2.2	Relation modèle – données	107
7.2.3	Consistance de chemin	108
7.2.4	Consistance au sommet et consistance de chemin	109
7.2.5	Chemins et déplacement d'équilibre	110
7.2.6	Bilan	111
8	Conclusion	113
8.1	Bilan	113
8.2	Perspectives	116
A	Inférence de l'effet des facteurs de transcription sur leurs gènes cibles	119
	Bibliographie	145
	Table des figures	147

Notations

Conventions générales

- ensembles en majuscules, italiques et police ordinaire : $A, E, F \dots$
- éléments d'un ensemble de type non déterminé (ou non déterminant) en minuscule, italique et police ordinaire : $x, f \in F \dots$
- ensemble des variables d'une fonction comme les ensembles ordinaires : dans $f(X, Y)$, X et Y sont des ensembles de variables.
- objets structurés (arbres, graphes, automates) en majuscule, italique et police dite caligraphique : \mathcal{A}, \mathcal{G} .
- termes en majuscule et police sans serif : X, y, C .

Notations spécifiques

$ X $	cardinal de l'ensemble X
$ \pi $	longueur d'un chemin π dans un graphe
$\deg_{\mathcal{G}}^{-}(s)$	degré entrant d'un sommet s dans le graphe \mathcal{G}
$\deg_{\mathcal{G}}^{+}(s)$	degré sortant d'un sommet s dans le graphe \mathcal{G}
$\text{dom}(f)$	domaine (espace de définition) d'une fonction f
$\mathbb{T}(X)$	où \mathbb{T} est un terme, et X un ensemble de variables. Précision (optionnelle) des variables libres du terme.
$[x_1 := c_1, \dots]$	désigne la substitution σ définie par $\sigma(x_i) = c_i$.
$\exists X \mathbb{T}$	où $X = \{x_1, \dots, x_n\}$ est un ensemble de variables, et \mathbb{T} est un terme. Équivalent à $\exists x_1 \exists x_2 \dots \exists x_n \mathbb{T}$.

Chapitre 1

Introduction

1.1 Motivations

La recherche en biologie moléculaire dispose depuis quelques années déjà de techniques expérimentales qui permettent de mesurer un grand nombre de variables simultanément, avec un nombre limité d'interventions humaines. Ces progrès sont le fruit de deux efforts de recherche orthogonaux, à savoir la miniaturisation des instruments de mesure, et la robotisation des tâches. On parle souvent de mesures « haut-débit », expression qui désigne un ensemble assez hétérogène de techniques et de types de données expérimentales, allant du séquençage des génomes à la recherche systématique d'interactions moléculaires. Toutes ces techniques sont de plus en plus utilisées en routine, et produisent des masses de données très conséquentes, le plus souvent mises à disposition sur des interfaces web. L'ambition que ces données suscitent, c'est de permettre une étude *globale* d'un système biologique, intégrant des informations les plus complètes possible sur l'expression des gènes, les interactions au sein de la cellule *etc* ... On cherche ainsi à modéliser le comportement d'un système comme la résultante d'un grand nombre d'interactions entre ses éléments. Cette approche, connue sous le nom de *biologie systémique* est bien entendu complémentaire d'une approche *réductionniste* où l'on considère les systèmes les plus simples possible afin d'en déterminer les mécanismes élémentaires.

Cette apparente avalanche de données doit être quelque peu nuancée. Les mesures haut-débit restent des techniques onéreuses ; en pratique, cela signifie que pour une étude ciblée, on ne peut matériellement réaliser qu'un petit nombre d'expériences de ce type, en regard du *nombre de variables mesurées*, et du *bruit* généralement observé. Pour fixer les idées, disons que les techniques courantes affichent de l'ordre du millier de variables mesurées, et sont au plus utilisées une dizaine de fois en pratique¹. On peut bien sûr compter sur les données déjà disponibles dans des conditions voisines de l'étude, au prix d'une diminution – peut-être considérable – du rapport signal sur bruit. L'exploitation des données haut-débit est donc particulièrement difficile, non pas seulement à cause du volume d'information en jeu, mais aussi parce qu'elle doit

¹ce qui est peu, attendu qu'une même expérience doit être répétée au minimum entre 3 et 5 fois

être adaptée à la qualité des données et au déséquilibre entre le nombre de variables mesurées et le nombre de mesures disponibles.

Intéressons-nous maintenant à une deuxième difficulté : aussi surprenant que cela puisse paraître de prime abord, il n'est absolument pas trivial de préciser ce que l'on attend d'une « analyse » des données haut-débit. Il s'agit d'une problématique générale en biologie systémique : parvenir à formuler des questions ou des propriétés d'intérêt pour le système que l'on étudie est, dans ce domaine, un problème à part entière. Les raisons de cette difficulté sont diverses, à commencer par le fait qu'il est relativement ardu de relier les processus biologiques étudiés (apoptose, adaptation à un stress) à des acteurs moléculaires précis (gènes, protéines . . .). Or si l'on s'intéresse le plus souvent aux propriétés desdits processus, ce sont bien les espèces chimiques que l'on mesure. Dans le cas des données haut-débit, on peut évoquer une difficulté plus spécifique : ce type de mesure est généralement réalisé comme un travail exploratoire ; pour un phénomène donné, les techniques de mesure haut-débit donnent une image globale du système, à partir de laquelle on espère débiter un travail d'investigation plus ciblé. Pour cela, il faut pouvoir repérer dans la masse de données recueillies des éléments susceptibles d'être intéressants pour l'élucidation du phénomène étudié. Dit autrement, il faut savoir interpréter les données produites, c'est-à-dire y distinguer ce qui est surprenant de ce qui est attendu, ce qui fait sens de ce qui est contradictoire avec les connaissances sur le système ; puis dans un deuxième temps être capable d'utiliser les données comme une base pour générer des hypothèses réfutables par l'expérience.

Ces considérations nous amènent au problème étudié dans cette thèse : pour un processus biologique donné, on dispose de données haut-débit provenant de différentes sources. Comment comparer les données entre elles et tester leur cohérence ? Comment les combiner pour en déduire des informations nouvelles ?

1.2 Approche suivie

Pour répondre à ces questions, nous introduisons un formalisme permettant d'intégrer un large spectre de données haut-débit. Dans ce formalisme, les données expérimentales ou les connaissances s'interprètent soit comme des éléments d'un modèle physique des interactions cellulaires, soit comme des mesures sur l'état de ce modèle. Nous proposons dans ce cadre une notion formelle de consistance entre modèle et mesures expérimentales. Cette notion de consistance est à la base d'une démarche complète d'analyse de données, décrite en figure 1.1. Partant des données existantes (comportant des mesures expérimentales et les régulations connues d'un système), nous testons en premier lieu leur cohérence. Dans le cas où ce test échoue, nous montrons des approches de diagnostic nous permettant d'en comprendre la cause. En prenant appui sur ce diagnostic, on peut rechercher des corrections, soit sur la base de sources bibliographiques, soit en étudiant l'ensemble des corrections possibles. Une fois que l'on a obtenu un modèle cohérent avec les données, il peut être utilisé pour produire des prédictions sur le système et ses variables non observées. Cela inclut notamment la problématique de *reverse-engineering*, c'est-à-dire la découverte de tout ou partie des

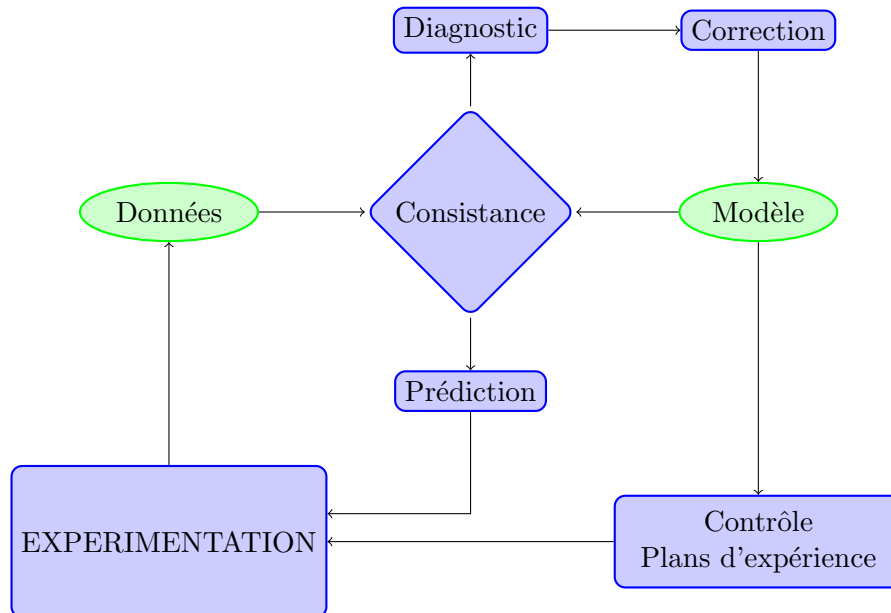


FIG. 1.1 – Cycle d'analyse des données haut-débit

mécanismes à l'œuvre dans un système donné à partir de mesures expérimentales. Les prédictions obtenues pourront enfin guider les expérimentations suivantes, soit par une vérification directe, soit éventuellement dans le cadre de plans d'expériences.

Le formalisme que nous proposons est adapté aux caractéristiques des données haut-débit, que nous résumons comme suit :

1. elles sont en général fortement bruitées et peu répliquées ;
2. elles portent sur un grand nombre de variables, mais sur peu de conditions différentes ;
3. elles représentent un volume considérable d'information ;

Le premier point nous conduira d'une part à adopter une *approche qualitative*, robuste aux valeurs numériques bruitées ; il motive surtout notre approche basée sur l'introduction d'un critère de *consistance entre les sources de données*, et son utilisation pour *détecter, voire corriger* les valeurs aberrantes. Le deuxième point implique qu'il nous faudra travailler avec des modèles sous-déterminés, c'est-à-dire en admettant qu'il y a plusieurs modèles plausibles d'après les données disponibles. Ainsi, nous devons être capables de proposer des *prédictions* malgré l'incertitude sur le modèle réel. Mieux encore, on pourra s'intéresser à la *conception d'expériences* permettant de déterminer le modèle réel le plus efficacement possible. Enfin, le troisième point nous obligera à soigner tout particulièrement les aspects algorithmiques associés à chacune de ces tâches. Nous voulons à présent mentionner quelques uns des types de données haut-débit les plus courants ; nous nous appuyons notamment pour cela sur la revue de Joyce et Palsson [46].

1.3 Sur les données

Données d'expression Il s'agit des mesures sur le *transcriptome* des cellules, c'est-à-dire l'ensemble des molécules d'ARN présentes à un instant donné dans un tissu donné. Ces ARN peuvent coder pour des protéines ou avoir une activité propre (transport des acides aminés, régulation d'autres ARN, modifications du génome ...). Les techniques les plus utilisées sont notamment les puces à ADN, la PCR quantitative ou la méthode SAGE. Plus précisément, ces techniques mesurent la quantité d'ARN présente dans un échantillon à l'aide de sondes qui sont spécifiques de chaque gène. Certaines puces à ADN par exemple sont munies de plusieurs dizaines de milliers de sondes, et servent à réaliser des mesures dites *pangénomique*, c'est-à-dire portant sur l'ensemble des séquences génomiques transcrites connues.

La valeur trouvée pour chaque sonde ne donne la quantité d'ARN présente dans l'échantillon qu'à une constante multiplicative près. Celle-ci dépend notamment de la taille de l'échantillon, et de constantes de l'appareil de mesure, le tout étant difficile à étalonner. C'est pourquoi on procède en général par comparaison avec une condition de référence : par exemple, cellules en culture contre cellules soumises à un stress, cellules tumorales contre cellules saines, cellules nerveuses contre cellules musculaires. Le résultat d'une mesure d'expression est donc un vecteur contenant pour chaque transcrit le ratio entre les valeurs trouvées dans la condition d'intérêt et la condition de référence.

La technique la plus courante aujourd'hui est la puce à ADN², qui est une plaque sur laquelle sont fixées des sondes. Ces sondes sont constituées d'une séquence d'ADN complémentaire d'une séquence recherchée. Les plaques peuvent contenir jusqu'à plusieurs dizaines de milliers de sondes, et l'on peut choisir la composition de chaque plaque. Les quantités typiques d'ARN peuvent énormément varier d'un ARN à l'autre, et les signaux forts rendent les signaux faibles peu précis. C'est pourquoi on trouve des plaques dédiées pour certains types d'ARN qui sont connus pour être peu exprimés (ARN de facteurs de transcription ou de micro-ARN par exemple). La précision que l'on obtient pour les ratios d'expression est de l'ordre de l'unité (voir figure 1.2). Pour des mesures plus précises, on a recours à la RT-PCR (pour *Real Time Polymerase Chain Reaction*). La technique de PCR permet de créer un grand nombre de copies d'un brin d'ADN dont la séquence est connue, même si l'on ne dispose initialement que d'un petit nombre d'exemplaires. Lors d'une RT-PCR, on itère des phases de copies (dites d'amplification) et la vitesse d'apparition du brin d'ADN (mesurée par fluorescence) en donne la quantité initiale. La RT-PCR offre une précision de l'ordre du dixième, mais devient relativement lourde au-dessus de la centaine de mesures.

Dans tous les cas, il faut en premier lieu obtenir l'ARN contenu dans les cellules (ARN total), par une opération d'extraction. L'ARN obtenu est « converti » en ADN, par rétrotranscription. Cette opération est nécessaire parce les molécules d'ARN sont particulièrement instables, contrairement aux chaînes d'ADN. C'est pour cela qu'on parle de puces à ADN, ou d'amplification de l'ADN.

Il faut en pratique disposer d'une quantité suffisante d'ARN total, et pour cela uti-

²La seule base de données GEO [7] comptait en septembre 2006 plus de 120000 de ces puces, réparties sur plus de 200 organismes.

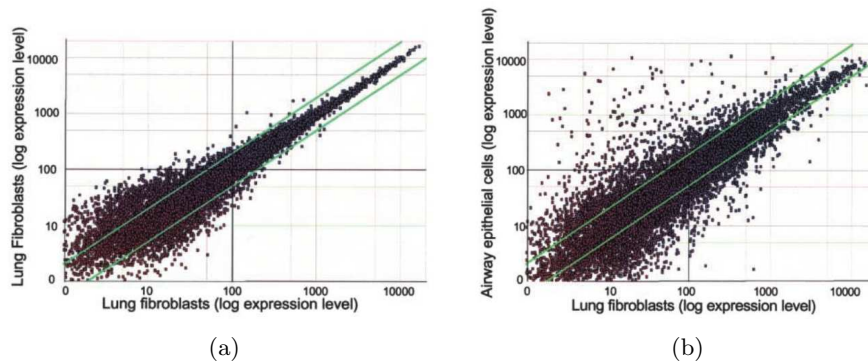


FIG. 1.2 – (a) Mesure d’expression sur un échantillon de poumon humain, extraite de [47]. Le même échantillon a été analysé sur deux puces de même modèle. Chaque point correspond à une sonde de la puce. Chaque sonde correspond spécifiquement à un ARN transcrit. Sur chaque axe est représentée la mesure d’expression normalisée obtenue sur chaque puce. Dans le cas idéal, tous les points devraient se trouver sur la droite d’équation $y = x$. Les droites vertes correspondent à une variation d’un facteur 2 (augmentation et diminution respectivement). (b) Même type de comparaison, mais cette fois entre deux échantillons issus de tissus différents.

liser un échantillon suffisamment important. Par conséquent, cela oblige à travailler sur un échantillon non homogène, parce qu’un tissu peut contenir plusieurs types cellulaires très distincts, et parce que les cellules d’un même type sont le plus souvent dans des états notablement différents. Il existe des contextes expérimentaux dans lesquels on peut « synchroniser » les cellules, mais la plupart du temps, une mesure d’expression est une mesure « en moyenne » et peut fort bien ne correspondre à l’état d’aucune cellule en particulier. Il existe des approches comme [6] qui proposent de corriger ce problème sous certaines conditions, mais elles sont en pratique rarement applicables.

Électrophorèse sur gels bidimensionnels De même que l’on peut, à l’aide d’une puce à ADN, mesurer la quantité de chaque séquence d’ARN dans un échantillon, on voudrait pouvoir mesurer la quantité des protéines présentes; en quelque sorte, disposer d’une « puce à protéines ». Malheureusement, les choses sont plus compliquées. La différence importante est que la notion de séquence complémentaire n’existe pas chez les protéines. Il est donc extrêmement difficile de fabriquer une sonde spécifique d’une protéine donnée (voir plus loin avec la technique de *chIP-chip*). Pour explorer le *protéome* d’une cellule (l’ensemble des protéines présentes), la méthode la plus efficace à l’heure actuelle semble être les gels bidimensionnels, qui sont des plaques recouvertes d’un milieu particulier sur lequel on fait migrer les protéines extraites d’un échantillon. Leur déplacement est provoqué dans une dimension, par un champ électrique et dans l’autre, par un gradient de pH. On dit que l’on obtient une bonne résolution quand chaque tache est individualisée et ne contient qu’une seule sorte de protéine. Si c’est le cas, on peut alors mesurer la quantité de protéines présentes en mesurant la surface

de la tache qui lui correspond. Reste – et c’est là la plus grande difficulté – à identifier la protéine associée à chaque tache. Il s’agit en pratique de l’étape limitante en terme de débit. La procédure la plus performante actuellement fait appel à des techniques de spectrométrie de masse. Notons bien qu’il n’y a pas une relation simple entre la quantité d’un ARN donné et celle de la protéine correspondante. Ceci est dû à l’existence de régulations dites *post-transcriptionnelles*, qui peuvent éventuellement dégrader un ARN avant qu’il ne soit traduit en protéine.

Chromatographie/Spectrométrie La technique précédente peut être adaptée pour explorer le *métabolome* d’une cellule, c’est-à-dire l’ensemble des métabolites présentes dans un tissu. Les métabolites sont des molécules impliquées dans la régulation énergétique et dans la structure (cytosquelette) des cellules. Dans ce cas ; l’électrophorèse est remplacée par des techniques de chromatographie : la migration n’est plus provoquée par des champs électriques mais par le déplacement d’un fluide (gaz ou liquide).

Séquençage C’est sans nul doute la source la plus ancienne de données haut-débit. Le séquençage de génome est l’un des protocoles les plus automatisés à l’heure actuelle. La mise en place de nombreux centres a permis d’obtenir en l’espace de quelques années plusieurs centaines de génomes complets. Selon la base de données GOLD [59], on comptait en mai 2007 près de 700 génomes complètement séquencés, et plus de 1800 projets de séquençage en cours, soit un total dépassant les 2500 espèces³. D’après les estimations courantes chacun de ces génomes contient de plusieurs centaines à quelques dizaines de milliers de gènes codant pour des protéines ou des petits ARN.

Disposer d’un génome a essentiellement deux bénéfices. Premièrement on peut y rechercher des indices sur le fonctionnement de la cellule par recherche directe de la séquence. On sait par exemple déterminer les protéines codées dans un génome et certaines de leurs variations (épissage alternatif notamment), ou encore – dans une certaine mesure – détecter les sites dans le génomes où se lient les facteurs de transcription. Le deuxième bénéfice, bien plus important en pratique, est de faciliter nombre de manipulations expérimentales touchant à la transcription, à commencer par la conception des sondes pour les mesures d’expression.

Par ailleurs, le génome séquencé correspond bien à un individu particulier dans une espèce donnée. Pour ne pas biaiser les conclusions d’une étude il est donc important de connaître les variations existant entre les individus d’une même espèces. Le type le plus simple de variation est la variation ponctuelle d’un nucléotide ou SNP (pour *Single Nucleotide Polymorphism*). Il existe également des bases de données de SNP, voir par exemple [63].

ChIP-chip Cette technique permet de détecter tous les sites de fixation d’un facteur de transcription donné sur le génome. Le protocole est le suivant : les protéines liées à l’ADN dans la cellule sont fixées à l’aide d’un produit particulier, puis l’ADN est extrait et fragmenté en petits brins par ultrasons. Il est le plus souvent possible, quoique

³dont, il faut le préciser, un grand nombre d’êtres unicellulaires

très technique, de fabriquer des anticorps qui se lient spécifiquement à une protéine donnée. Si l'on dispose d'un tel anticorps, on peut l'utiliser pour marquer les complexes protéine-ADN d'un facteur de transcription particulier, puis les séparer du reste. Le complexe est ensuite détruit, et on extrait uniquement les courts brins d'ADN. Cette phase correspond à la partie « *ChIP* » (pour *Chromatin Immuno-Precipitation*). La partie « *chip* » est une analyse de l'ensemble des brins d'ADN par puce à ADN classique. L'ensemble permet donc de détecter toutes les séquences du génome qui sont des sites de fixation d'un facteur de transcription *dans une condition donnée*. L'application de cette technique peut mettre à jour des milliers de sites dans le génome. Néanmoins elle ne renseigne pas sur l'effet des liaisons découvertes. En particulier, la fixation d'un facteur de transcription peut fort bien n'avoir aucun effet sur la transcription des gènes.

Double hybride Il s'agit d'une technique permettant de détecter à très grande échelle les couples de protéines capables de former un complexe. Le principe de la manipulation est le suivant : supposons que l'on cherche à tester la complexation de deux protéines A et B ; on construit dans un organisme simple (principalement la levure) un système rapporteur, qui est constitué d'un facteur de transcription d'une part et d'un gène cible de ce facteur d'autre part. Le gène en question produit une protéine phosphorescente, dont la présence sera donc détectable facilement. Le facteur de transcription n'est pas produit par l'organisme directement ; à la place on introduit dans le génome de l'organisme un gène codant pour la protéine A fusionnée à un morceau du facteur, et un gène B codant pour la protéine B fusionnée à l'autre morceau du facteur. Si A et B forment une interaction, les 2 parties du facteur de transcription formeront un complexe actif et on observera une fluorescence. Ce principe peut être automatisé pour tester plusieurs dizaines de milliers de couples, comme dans [79].

Extraction de la littérature À bien y réfléchir, la source d'information la plus conséquente se trouve probablement dans les centaines de milliers d'articles publiés depuis trois ou quatre décennies, dont la plupart est répertoriée (au moins en ce qui concerne les résumés) dans le serveur Pubmed du NCBI⁴. Le problème bien sûr, c'est que l'information est « cachée » dans du texte en langue naturelle, et par conséquent difficilement accessible à un traitement automatique⁵. Plusieurs groupes [100, 44, 35, 48, 38] y ont répondu de manière très pragmatique : puisqu'il est impossible d'extraire automatiquement et de manière fiable l'information dispersée dans la littérature, il suffit de l'extraire manuellement, quitte à cibler les problématiques et mettre suffisamment de personnes à la tâche. Certaines bases de données ainsi développées contiennent jusqu'à quelques dizaines de milliers d'interactions. La plupart de ces bases (à l'exception notable de Kegg et de RegulonDB) sont développées par des sociétés privées et ne sont pas dans le domaine public. Outre l'effort immense qu'une telle entreprise constitue,

⁴ Accessible à l'adresse <http://www.ncbi.nlm.nih.gov/sites/entrez>

⁵ Il nous faut modérer ce propos, puisqu'il existe un corpus de recherche très important sur l'extraction d'information depuis un texte écrit en langue naturelle, et beaucoup d'applications à la biologie. Néanmoins les résultats obtenus jusqu'à présent ne sont à notre connaissance pas assez fiables pour servir de point de départ à d'autres investigations.

il faut aussi considérer les problèmes liés à la formalisation du contenu des publications : quelles informations extraire ? Comment spécifier le contexte de l'étude ? Même partielles, ces données sont essentielles du fait de leur grande fiabilité. Notons qu'à ce jour les bases de données issues de la littérature se limitent aux interactions. Il n'existe à notre connaissance qu'une base de données portant sur des observations (évolution temporelle, réponse typique à une perturbation) [35].

1.4 Travaux connexes

1.4.1 Panorama

Grâce aux puces à ADN notamment, il est aujourd'hui relativement simple, quoiqu'encore coûteux d'obtenir une image globale de la réponse transcriptionnelle d'une cellule à une perturbation. Ce qui pose problème, c'est l'exploitation de ces données : sous sa forme brute, une mesure d'expression est un résultat peu lisible et très volumineux. Elle peut bien sûr dans un premier temps servir à vérifier ce que l'on sait déjà. Le problème est nettement plus complexe en revanche, dès que l'on souhaite s'appuyer dessus pour inférer des mécanismes ou guider l'expérimentation plus avant.

L'utilisation la plus simple (et la plus courante) des mesures d'expression à l'échelle génomique, c'est le criblage de gènes impliqués dans un phénomène biologique donné. Les transcrits exhibant une forte variation entre les deux conditions (c'est-à-dire supérieure à un certain seuil) sont alors utilisés comme candidats prioritaires pour l'investigation des mécanismes dudit phénomène. Ici, deux difficultés apparaissent en pratique. La première est liée au choix du seuil au-dessus duquel une variation est jugée significative. Il existe une littérature abondante sur le sujet (voir les revues rapides dans [85, 47]); mais comme on peut le constater sur la figure 1.2, le niveau de bruit est relativement important dans le cas des puces à ADN. Est en général jugée significative une variation d'un facteur au moins 2 ou 3. Pourtant rien n'exclut *a priori* qu'une variation faible – disons trop faible pour sortir significativement du bruit de fond – ait un rôle important dans un phénomène donné. Il semble donc difficile de sélectionner une liste de gènes à partir des seules données d'expression. Deuxièmement, l'expérience montre qu'il y a très souvent plusieurs centaines de gènes dont la variation est significative. Une bonne partie d'entre eux sont en général décrits dans la littérature, mais il ne serait guère raisonnable de se lancer dans une compilation manuelle des données disponibles sur les gènes identifiés – ne serait-ce qu'à cause du temps nécessaire à sa constitution, bien supérieur au temps nécessaire à la production des mesures d'expression. Notons bien que de toute façon, le problème n'aurait été en rien résolu : on aurait transformé une masse de données numériques en une masse de texte guère plus propice à l'exploitation.

Ces constats ont amené à un grand nombre de propositions, que nous regroupons en trois catégories. La première [71] consiste à annoter chaque gène avec des informations diverses issues de bases de données publiques (dont, typiquement plusieurs mesures d'expression), puis à utiliser des instruments d'analyse statistique pour structurer l'ensemble des candidats, par des méthodes de *clustering*. L'intérêt est qu'on diminue ainsi

le nombre d'entités à considérer, et que l'on fait apparaître des groupes pouvant – dans le meilleur des cas – avoir une pertinence biologique. C'est une approche à double tranchant : d'un côté elle est intéressante parce qu'elle permet d'intégrer (au travers de la distance utilisée lors du clustering) des informations très diverses sur les gènes candidats (cocitations dans les articles, interactions connues, description ontologique [19], coexpressions dans d'autres conditions) ; de l'autre il devient difficile d'expliquer le regroupement de deux gènes, à mesure qu'on ajoute des informations. Il apparaît en pratique que ces approches de type *data mining* sont un bon moyen de faire ressortir des candidats, ou de suggérer des liens fonctionnels entre plusieurs gènes. Leur limite est qu'elles calculent un résultat qui n'est pas réfutable par l'expérience. Il n'est donc pas facile d'évaluer objectivement la qualité du résultat. De plus elles ne fournissent pas d'explication physique des liens trouvés : une fois quelques candidats mis en avant, le travail d'élucidation des mécanismes reste entier.

Une deuxième approche consiste à utiliser les données d'expression comme entrée dans des problèmes d'apprentissage (classification ou régression). Autrement dit, il s'agit de proposer des modèles statistiques liant les données d'expression (et éventuellement d'autres sources d'information) à des propriétés vérifiables. Donnons quelques exemples. Les travaux décrits dans [8, 98] sont des tentatives de classification des tumeurs dans différents cancers à partir de profils d'expression. Ces classifications correspondent à des stades ou des conditions cliniques et peuvent être utiles au choix d'un traitement approprié. Les données d'expression ont été également utilisées pour la prédiction d'interactions protéine-protéine [101], de réseaux génétiques [62]. Comme dans les approches de data-mining, il existe des outils puissants pour combiner d'autres types de données aux mesures d'expression. Citons notamment les développements produits autour des fonctions noyaux revus dans [98]. Les prédictions obtenues sur la structure du système peuvent ensuite être testées expérimentalement.

En classification ou en régression, l'objectif est donc d'estimer une grandeur particulière à partir d'observations et de connaissances sur le système. La troisième et dernière approche que nous souhaitons mentionner, c'est l'utilisation des données d'expression dans un modèle physique des réseaux de réactions. Dans ce cadre, on définit explicitement les états du système et son évolution dans le temps ou sous l'action d'une perturbation. La modélisation des cinétiques chimiques par des équations différentielles ou des processus stochastiques en sont des exemples. Le principal avantage de ce type d'approche, c'est de permettre l'intégration de données diverses dans un langage plus lisible que les mesures de similarité utilisées en apprentissage. On évite ainsi le côté « boîte noire » des prédictions réalisées en classification notamment. Le travail rapporté dans le présent mémoire appartient à cette catégorie. Notons néanmoins qu'il n'y a pas de frontière nette entre modèles physiques d'une part, et modèles statistiques d'autre part. C'est particulièrement clair dans [54], où Kundaje *et al* montrent que découvrir la fonction de régulation d'un gène à partir de sa séquence promotrice peut se ramener à un problème de classification supervisée. Les propositions à base de réseaux bayésiens (voir [67] pour une introduction) en sont un autre exemple.

1.4.2 Modèles physiques de réseaux biologiques

L'énumération faite au paragraphe 1.3 montre que les données haut-débit peuvent généralement être distinguées en deux catégories, selon qu'elles portent sur la structure et les mécanismes élémentaires du système étudié (réactions biochimiques) ou sur son état (variation en concentration par exemple). Ces deux types d'information ont en commun – encore que pour des raisons différentes – d'être de nature qualitative. Dans le premier cas, la raison en est que s'il est techniquement possible de détecter l'existence de réactions à grande échelle, il reste toujours extrêmement délicat d'en connaître les constantes cinétiques ; dans le second cas, les mesures apparaissent le plus souvent comme une grandeur physique quantitative (ratio d'expression, de quantités mesurées), mais elles sont extrêmement bruitées. Cette variabilité correspond majoritairement à une variabilité biologique des échantillons, mais également aux limites des instruments de mesures, comme nous le voyons sur la figure 1.2.

Pour concevoir un modèle physique d'un système en présence de données bruitées et incomplètes, les modèles probabilistes sont une option attractive : les informations sur la structure sont codées par des variables aléatoires discrètes représentant généralement un graphe, et les mesures quantitatives sont supposées suivre une distribution paramétrique. Ses paramètres – notamment les caractéristiques du bruit, et les constantes cinétiques – sont estimés à partir des données selon un critère d'optimisation, de type maximum de vraisemblance. La piste la plus étudiée repose sur l'emploi de réseaux bayésiens [82, 81, 28, 14, 66]. Ces approches ont principalement deux faiblesses : premièrement, elles reposent sur des problèmes d'optimisation non convexes, c'est-à-dire pouvant comporter des optima locaux ; dès que les modèles comportent quelques dizaines de variables, il devient particulièrement difficile de trouver un optimum global. Deuxièmement, même en supposant un modèle optimal trouvé, celui-ci peut être très différent des modèles quasi-optimaux. Autrement dit, l'inférence par maximisation d'un score est potentiellement peu robuste. Ce problème a une solution⁶ élégante – mais extrêmement coûteuse d'un point de vue calculatoire – consistant à étudier la distribution postérieure, comme cela est fait dans les approches bayésiennes pour la phylogénie [42]. Enfin, l'utilisation de méthodes probabilistes nécessite des échantillons suffisamment importants pour estimer les paramètres du modèle. On peut penser d'après la littérature [5] que la limite basse d'applicabilité des méthodes probabilistes se situe autour de 100 à 300 expériences indépendantes. Or il est bien rare en pratique de disposer d'autant de données pour un seul système.

Une réponse partielle à ces problèmes peut être trouvée dans les approches décrites dans [10, 4, 96], où les modèles probabilistes sont remplacés par des équations différentielles ordinaires, le plus souvent linéaires. Les techniques d'estimation sous-jacentes sont plus abordables sur le plan complexité (résolution de systèmes linéaires, optimisation convexe) et le traitement du bruit requiert moins de paramètres (interpolation, estimation au sens des moindres carrés). L'étude réalisée dans [5] confirme l'intuition : ces méthodes se comportent mieux que les méthodes probabilistes dans le cas où peu de données sont disponibles ; et les différences s'atténuent avec l'augmentation du nombre

⁶Solution qui, à notre connaissance, n'est pas souvent mise en œuvre

d'expériences. Néanmoins dans les deux approches, il faut pouvoir fournir un nombre suffisant d'expériences indépendantes.

Le *raisonnement qualitatif* est une alternative pour traiter des problèmes où les données sont imprécises et/ou incomplètes [53, 93, 39]. L'approche en raisonnement qualitatif consiste à sur-approximer l'ensemble des comportements observables, en abstrayant des propriétés plus robustes des mesures, comme leur signe, ou leur ordre de grandeur. Les relations quantitatives sont à leur tour abstraites en contraintes qualitatives, qui constituent des conditions nécessaires (mais pas suffisantes) à vérifier. Cette démarche a déjà été appliquée en biologie systémique pour modéliser la dynamique des réseaux génétiques [78, 21, 9], ou des réseaux de signalisation [15, 36]. Dans tous ces travaux, le processus d'abstraction qualitative permet de dériver une notion de cohérence entre un modèle et des mesures expérimentales adaptée à la qualité et la précision des données disponibles. Le travail présenté dans cette thèse procède de la même démarche, appliquée à l'étude des données haut-débit.

1.5 Nos contributions

Nous présentons maintenant les différentes contributions de ce travail de thèse.

Critère de consistance et contraintes qualitatives Nous introduisons un critère de consistance entre un modèle simple des interactions cellulaires et des mesures expérimentales. Ce critère stipule essentiellement que la variation d'une espèce entre deux états d'un système donné doit toujours pouvoir être expliquée par la variation d'une espèce qui la régule. Nous exprimons cette règle intuitive comme une contrainte sur variables à domaines finis, dont la résolution est montrée NP-complète.

Par ailleurs, nous démontrons la validité de notre critère de consistance dans un cadre différentiel. Cette étude précise les limites d'applicabilité de notre formalisme, et fournit des guides précieux pour l'interprétation des données.

Algorithmes pour l'étude des contraintes qualitatives La deuxième contribution de ce travail porte sur la résolution et l'étude des contraintes qualitatives. Nous proposons deux approches offrant des possibilités complémentaires. La première utilise les diagrammes de décision pour représenter explicitement mais de manière compacte l'ensemble des solutions de la contrainte. Cette approche est associée à des techniques de décomposition et de réduction que nous décrivons, afin d'accroître significativement la taille des contraintes pouvant être traitées. La deuxième approche fait appel à des techniques récentes de programmation logique : nous montrons comment coder les solutions de nos contraintes qualitatives comme les modèles d'un programme logique sous la sémantique des modèles stables. L'utilisation de solveurs dédiés particulièrement performants nous fournit une alternative convaincante pour l'étude des contraintes qualitatives.

Validation à grande échelle de l'approche Nous décrivons enfin deux applications de notre approche sur données réelles. La première consiste à prédire la réponse transcriptionnelle globale de la bactérie *E. coli* à partir de données bibliographiques ; la deuxième aborde un cas particulier de reconstruction de réseau génétique, où l'on cherche à inférer l'effet des facteurs de transcription (activation ou inhibition) sur leurs gènes cibles. Ces deux applications démontrent d'une part la capacité de nos algorithmes à traiter un volume d'information réaliste : les réseaux considérés comportent plusieurs milliers de gènes et de régulations et sont confronté à plusieurs dizaines de mesures d'expression. D'autre part, nous montrons par ces expériences que notre critère de consistance est un guide fiable et informatif pour l'analyse de données. Les prédictions obtenues par notre approche ont pu être validées de manière significative, et les désaccords importants nous ont permis dans plusieurs cas de corriger nos modèles.

La suite de ce document est structurée comme suit : nous commençons par donner une présentation générale et intuitive de notre approche au chapitre 2 ; nous introduisons ensuite formellement notre notion de consistance et la modélisation associée au chapitre 3. Les deux chapitres suivants détaillent les méthodes algorithmiques utilisées pour la résolution et l'étude des contraintes qualitatives. Suivent enfin les applications sur données réelles au chapitre 6.

Chapitre 2

Présentation générale de l'approche : modélisation de l'opéron lactose

Dans ce chapitre, nous illustrons sur un exemple simple la démarche détaillée dans cette thèse. Il s'agit moins ici d'en faire un exposé formel que de la présenter de manière pragmatique et – nous l'espérons – intuitive.

Les données d'expression Nous avons vu en introduction que les données d'expression fournies par les puces à ADN sont caractérisées par un bruit très important relativement au nombre de variables observées et au nombre de réplicats effectués. Pour s'en apercevoir, on peut par exemple examiner les données produites dans les travaux de Maurer *et al* [64], qui portent sur la réponse génétique de la bactérie *E. coli* à différents pH dans le milieu de culture. Les bactéries ont été exposées à trois pH distincts, et pour chaque pH, cinq réplicats ont été produits. Pour chaque pH et chaque réplicat, une puce à ADN a été utilisée pour mesurer le niveau d'expression d'environ 3800 gènes. Un extrait des résultats est donné en table 2.1.

Les mesures qui y sont présentées correspondent au niveau d'expression des gènes (à l'état stable) quand les bactéries ont été cultivées sur un milieu à pH de 5 ou pH de 7. Le traitement statistique effectué sur les mesures brutes assure que ces données sont normalisées, c'est-à-dire qu'elles sont comparables d'un gène à l'autre. La première observation est que selon le gène il peut y avoir un écart relatif à la moyenne dépassant les 25%. Le bruit observé est dû d'une part à l'instrument de mesure (comme illustré à la figure 1.2), et d'autre part à la variabilité des échantillons biologiques. Il est clair qu'un échantillon de 5 mesures est insuffisant pour estimer une valeur moyenne, et ce d'autant plus que la loi du bruit n'est pas connue, et *a priori* difficile à modéliser. Il ne s'agit pas ici d'un cas au pire : la plupart des données disponibles sont moins, voire pas du tout répliquées.

Gène	Expression sous pH 5					Expression sous pH 7				
	1	2	3	4	5	1	2	3	4	5
agaA	128.7	347.1	344.1	346.6	381.4	620.5	558.5	420.0	393.7	419.2
agaB	12.7	18.8	14.5	35.6	16.3	6.5	17.6	4.6	12.6	16.0
agaC	23.5	66.5	78.1	70.8	71.3	85.6	78.4	53.9	63.5	53.5
agaI	51.7	65.1	125.3	116.4	104.0	248.4	104.9	167.7	198.2	175.4
agp	657.3	1019.4	1142.2	1254.3	1060.8	1711.4	1048.8	1551.6	1040.5	1289.6
alaX	6481.9	8344.6	8435.7	7064.6	4838.6	51.9	5855.5	5646.2	6318.4	6245.0
aldA	1588.1	1689.7	1489.3	1494.4	1227.9	1126.5	631.1	553.6	526.6	918.1

TAB. 2.1 – Extrait des résultats obtenus dans [64]. Chaque ligne correspond à un gène de *E. coli*, chaque colonne correspond à un couple (condition,réplikat). Le tableau donne les mesures du niveau d'expression des gènes pour cinq réplikats et deux conditions (pH=5 et pH=7).

Gene	agaA	agaB	agaC	agaI	agp	alaX	aldA
Variation pH 5 → 7	+	+	0/?	0/?	0/?	-	-

TAB. 2.2 – Interprétation des données présentées dans le tableau 2.1. Les données quantitatives sont remplacées par le signe de variation en expression entre les deux conditions

Interprétation qualitative Quelle information peut-on tirer de ces données d'expression pour caractériser le passage d'un pH faible à un pH neutre? La dispersion observée rapportée à la taille de l'échantillon rend une interprétation numérique (sous la forme d'une moyenne, ou d'un intervalle de confiance) un peu hasardeuse. Nous proposons dans ce travail de ne considérer que le signe de la variation entre les deux conditions. C'est-à-dire qu'il nous faut décider, à partir de ces données si l'expression de chaque gène a augmenté ou diminué de manière significative. Au moins intuitivement, il semble que cette interprétation soit moins problématique : dans la table 2.1, il est à peu près clair que les gènes agaA, agaI, alaX et aldA ont une variation significative, respectivement positive, positive, négative, négative. L'algorithme – trop naïf – derrière cette interprétation consiste à calculer la soustraction des moyennes pour chaque condition, et rendre son signe. Dans le cas où l'écart relatif des moyennes est trop faible, on peut au choix, assigner une variation nulle (négligeable), soit déclarer la variation inconnue. Ces deux alternatives ne sont bien sûr pas équivalentes : il faut choisir entre exploiter toute l'information disponible ou se préserver des erreurs d'interprétation. Nous verrons plus loin une façon de trancher.

La démarche que nous suivons consiste donc à abstraire des données quantitatives bruitées en attributs moins précis mais plus robustes, en l'occurrence le signe des grandeurs. Sur notre exemple, on obtient ainsi la mesure donnée en table 2.2.

Un modèle des interactions cellulaires Pour exploiter ces données, nous proposons de les comparer à d'autres informations disponibles sur le système étudié, à

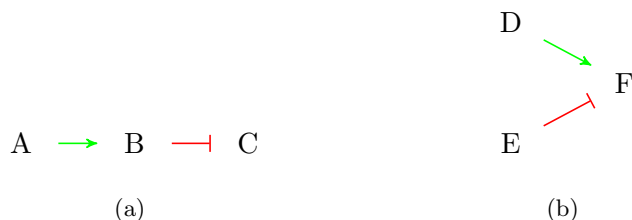


FIG. 2.1 – Exemples de graphe d’interaction. Les flèches d’extrémité triangulaire (en vert) représentent des activations, les flèches d’extrémité en T (en rouge) représentent des inhibitions.

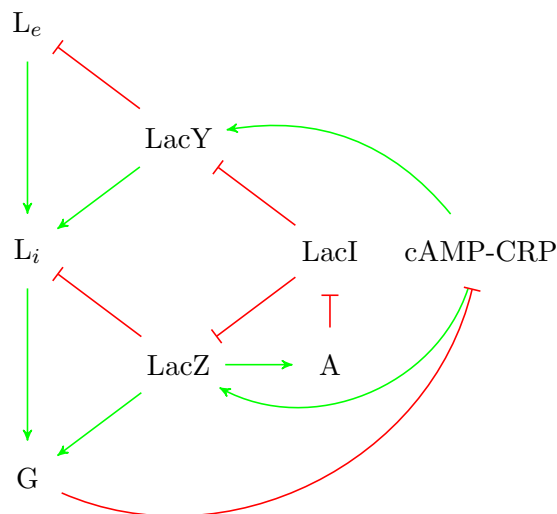
commencer par les régulations génétiques décrites dans la littérature. Classiquement, ces régulations sont représentées sous la forme d’un graphe, où chaque sommet correspond à un gène, et chaque arc représente une régulation. Les arcs sont de deux types, selon que la régulation est une activation ou une inhibition. Ces graphes – souvent appelés *graphes d’interaction* – sont généralement construits par une fouille ciblée des publications disponibles sur un sujet donné, ce qui peut demander un effort conséquent. On trouve par exemple des graphes sur les gènes contrôlant la segmentation chez la drosophile [17], sur le cycle cellulaire des mammifères [52] ou même un graphe synthétisant l’ensemble des régulations transcriptionnelles connues chez la bactérie *E. coli* [35]. Nous en donnons quelques exemples à la figure 2.

D’un point de vue expérimental, il est relativement simple¹ de tester l’existence d’une régulation génétique, voire d’en connaître l’effet (activation ou inhibition); ce type d’information peut facilement être trouvé dans la littérature. En revanche, il est beaucoup plus difficile d’obtenir des renseignements quantitatifs sur la régulation. Les graphes d’interactions sont donc adaptés à la précision des données disponibles, même si comme nous le verrons, ils ne sont pas une description univoque d’un système donné.

Critère de consistance Il nous faut maintenant donner une relation entre un graphe d’interaction et les mesures d’expression. Commençons par un cas simple, en examinant le graphe donné en figure 2.1(a). Il semble assez clair que si on fait augmenter A , qui est un activateur de B , alors B doit également augmenter. Le gène B inhibant C , on s’attend à ce que C diminue. De manière analogue, si on fait diminuer A , B et C doivent respectivement diminuer et augmenter. Que se passe-t-il lorsqu’un gène est régulé par plusieurs autres gènes, comme dans le graphe présenté en figure 2.1(b)? Supposons que D et E augmentent tous les deux, ceux-ci ayant des effets contraires sur F , il est impossible de conclure sur sa variation. Il faudrait pour cela disposer d’informations quantitatives. Dans ce cas, on admettra n’importe quelle variation pour F . En revanche si D diminue et E augmente, les deux régulateurs tendent à faire diminuer la quantité de F et seule une variation négative peut être admise.

Les raisonnements que nous venons d’effectuer peuvent être synthétisés en une for-

¹Cela nécessite néanmoins un travail importante

FIG. 2.2 – Graphe d'interaction pour l'opéron lactose chez *E. coli*.

mule simple : toute variation en expression d'un gène doit pouvoir être expliquée par la variation d'au moins un de ses régulateurs. Nous appellerons une règle de ce genre un critère de consistance entre le modèle des régulations (le graphe d'interaction) et les données d'expression. Nous allons maintenant voir plus en détail comment utiliser ce critère pour étudier un système. Nous illustrerons notre démarche sur l'exemple de l'opéron lactose, dont le graphe d'interaction est donné en figure 2.2.

L'opéron lactose Sans trop entrer dans les détails, donnons quelques indications sur le fonctionnement de ce système. Le glucose et le lactose sont des sucres, mais seul le glucose est suffisamment « simple » pour être utilisé directement par la bactérie. Si le milieu de culture contient du lactose, mais pas de glucose, la bactérie utilise un mécanisme lui permettant de réaliser la conversion. L_e représente le lactose présent dans le milieu de culture, L_i le lactose présent dans la bactérie, G le glucose. La transformation comporte deux étapes : d'abord l'entrée du lactose dans la cellule via l'action de la perméase LacY, puis transformation en glucose par l'enzyme LacZ avec production d'allolactose A. Cette chaîne de production est habituellement inhibée par le facteur LacI, mais elle peut être activée par le complexe cAMP-CRP si le niveau de glucose dans la cellule est suffisamment bas. Dans ce système, L_e et G doivent être considérés comme des entrées, c'est-à-dire des espèces dont la variation n'est pas expliquée dans le modèle, mais dépend également de facteurs extérieurs. Le lactose dans le milieu extérieur est bien entendu contrôlé par l'expérimentateur ; quant au glucose, son niveau dépend d'autres mécanismes qui ne sont pas représentés dans le graphe d'interaction.

Produit	L_e	L_i	G	LacY	LacZ	LacI	A	cAMP-CRP
μ_1	-	-	0	-	-	+	-	0
μ_2	+	+	0	+	-	0	0	-
μ_3	+	?	-	?	?	+	?	?
μ_4	?	?	?	-	+	?	?	0

TAB. 2.3 – Exemples de mesures pour l’opéron lactose décrit en figure 2.2.

2.1 Vérification

La première utilisation du critère de consistance consiste, comme nous l’avons déjà esquissé, à vérifier la compatibilité de données d’expression avec les régulations connues du système étudié. Soit par exemple les mesures données au tableau 2.3. La mesure μ_1 est compatible avec le graphe de l’opéron lactose. En effet on peut vérifier que pour chaque sommet (hormis L_e et G qui sont des entrées) toutes les variations peuvent être expliquées. Plus précisément elles respectent bien notre critère de consistance parce que pour tout sommet, on peut trouver un prédécesseur avec une influence du signe porté par le sommet. Ainsi la diminution de L_i s’explique par la diminution de LacY, qui s’explique par l’augmentation de LacI *etc* ...

En revanche la mesure μ_2 n’est pas compatible avec le graphe d’interaction : les variations de LacY, A et cAMP-CRP ne sont pas explicables par la variation de leurs régulateurs. Par exemple LacY augmente selon μ_2 , mais LacI ne varie pas (donc ne peut expliquer aucune variation) et cAMP-CRP – un activateur de LacY – diminue ; il ne peut donc pas expliquer l’augmentation de LacY. Plus globalement, on peut dénombrer 77 mesures compatibles avec le graphe d’interaction, sur un total de $3^8 = 6561$ possibilités², soit un ratio d’environ 1.2%.

On voit ici qu’il est relativement simple de vérifier le critère de consistance, lorsque tous les sommets du graphe sont observés. Que se passe-t-il lorsque les mesures sont partielles ? On dira qu’une mesure partielle satisfait au critère de consistance si l’on peut trouver des valeurs pour les sommets non observés, telles que l’ensemble vérifie le critère de consistance. Sous cette définition, la mesure μ_3 est compatible avec le graphe d’interaction, parce que la mesure :

Produit	L_e	L_i	G	LacY	LacZ	LacI	A	cAMP-CRP
μ'_3	+	0	-	-	-	+	-	+

étend μ_3 et respecte le critère de consistance. En revanche, la mesure μ_4 n’est pas compatible avec le graphe d’interaction parce que toute extension contredit le critère de consistance. La vérification est plus difficile dans le cas de données manquantes, puisqu’elle se ramène à une résolution (trouver une valeur pour les inconnues qui respecte une contrainte donnée). Dans cet exemple, les données manquantes correspondent à des

²Si, pour une question d’interprétation des données que l’on verra plus loin, on décide d’exclure la variation nulle, alors il y a 18 mesures compatibles avec le graphe d’interaction, sur un total de $2^8 = 256$ possibilités.

Produit	L_e	L_i	G	LacY	LacZ	LacI	A	cAMP-CRP
μ_3^1	+	0	-	-	-	+	-	+
μ_3^2	+	-	-	-	-	+	-	+
μ_3^3	+	+	-	-	-	+	-	+
μ_3^4	+	+	-	0	-	+	-	+
μ_3^5	+	+	-	+	-	+	-	+

TAB. 2.4 – Extensions de la mesure μ_3 qui sont consistantes avec le graphe d'interaction de l'opéron lactose.

mesures incomplètes ; plus généralement, les inconnues peuvent porter sur l'effet d'une régulation (activation ou inhibition), ou même sur son existence.

2.2 Prédiction

Lorsque l'on dispose de mesures compatibles avec un graphe d'interaction, on peut les utiliser pour prédire la valeur des variables non observées (variation d'un gène dans une condition donnée, effet ou existence d'une régulation). Par prédire, on entend essentiellement déduire par l'intermédiaire du critère de consistance. Par exemple, dans le cas du graphe décrit en fig. 2.1(a), nous avons vu que si on connaît la variation de A, alors on connaît également la variation de B et C. Les choses se compliquent pour des systèmes plus complexes : revenons à l'opéron lactose, et supposons que l'on ne dispose que de la mesure μ_3 . Celle-ci est consistante avec le graphe, et admet 5 extensions, données dans le tableau 2.4. Un examen de ces solutions au problème de vérification montre qu'outre L_e , G et LacI qui sont fixées par la mesure, LacZ, cAMP-CRP et A n'admettent qu'une seule variation ; on en déduit que si L_e et LacI augmentent, et G diminue alors nécessairement LacZ et A diminuent pendant que cAMP-CRP augmente. Récapitulons : on dispose de données incomplètes sur un système et ses réponses à des perturbations. Le critère de consistance décrit l'ensemble des valeurs admissibles pour les données manquantes. Enfin on appelle prédictions les invariants de cet ensemble.

On peut également s'intéresser à une notion de prédiction moins forte : d'après le tableau 2.4, LacY n'est pas invariant, mais prend dans 3 possibilités d'extension sur 5 la valeur -. En considérant les différentes possibilités comme équiprobables, on peut ainsi établir une loi de probabilité discrète sur les valeurs que peut prendre chaque variable non observée. Un exemple est donné au tableau 2.5, dans le cas où l'on observe une augmentation de L_e et une diminution de LacI.

Bilan

Nous avons décrit de manière informelle les premiers pas d'une analyse de données telle que proposée dans ce travail. Nous avons en particulier insisté sur la phase d'interprétation des données brutes, leur confrontation avec un modèle graphique des régulations du système (étape de vérification), et la recherche de prédictions sur les

Produit	L_e	L_i	G	LacY	LacZ	LacI	A	cAMP-CRP
+	1	$\frac{1}{3}$	0.6	0.6	1	0	1	0.2
0	0	$\frac{1}{3}$	0.2	0.2	0	0	0	0.2
-	0	$\frac{1}{3}$	0.2	0.2	0	1	0	0.6

TAB. 2.5 – Loi de probabilité des variations pour chaque espèce, en supposant une augmentation de L_e et une diminution de LacI. Les différentes possibilités d’extension sont considérées comme équiprobables.

variables non observées. Dans le reste de ce travail, nous formalisons cette démarche, et nous montrons comment réaliser efficacement les déductions sur le modèle et les données. Le prochain chapitre précise notamment la relation entre signe des régulations et signe des variations et en propose plusieurs justifications.

Chapitre 3

Équations qualitatives pour la consistance des données

Le problème posé dans cette thèse concerne le traitement de données haut-débit en biologie moléculaire. Nous l'avons vu, il s'agit de réussir à intégrer des informations de nature et d'origine hétérogènes, pour poser quatre types de questions :

- définir et tester la *consistance* entre ces données,
- prédire les variables non observées dans le cas où les données sont consistantes,
- aider au diagnostic en cas de non consistance, et aider à l'amendement de ces informations
- aider à la conception d'expériences pertinentes.

Nous abordons dans ce chapitre la question de la représentation et de l'intégration des informations disponibles. En introduction, nous avons distingué les sources d'information selon qu'elles concernent la *structure*, ou l'*état* du système étudié. La modélisation que nous proposons ici reprend cette dichotomie, en ce qu'elle comporte essentiellement deux types d'objets : des graphes pour représenter la structure, et des étiquetages de ce graphe pour représenter l'état. Le troisième et dernier ingrédient est une contrainte que ces objets doivent vérifier pour être déclarés consistants.

Nous commençons par présenter sans justification la modélisation étudiée dans cette thèse. Cette formulation servira de référence pour le reste du document. Nous évaluons ensuite la pertinence de notre formalisation en démontrant sa validité dans un cadre différentiel, puis booléen. L'intérêt de ces connexions sera de nous guider dans l'utilisation pratique de notre formalisme, par exemple pour l'interprétation des données quantitatives ou de réseaux de réactions complexes.

3.1 Formalisation

3.1.1 Le graphe d'interaction et ses étiquetages

Représentation du système Les systèmes que nous cherchons à représenter sont des réseaux de réactions biochimiques, telles que des régulations géniques ou des réseaux

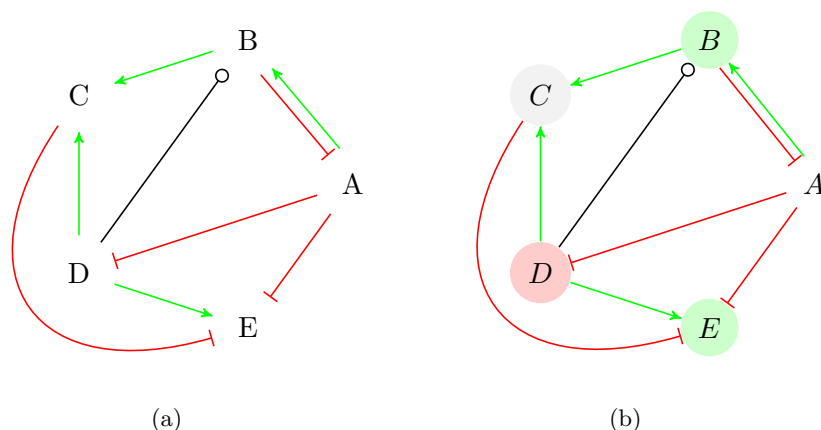


FIG. 3.1 – (a) Un exemple de graphe d’interaction. Les sommets sont des espèces chimiques, les flèches représentent des régulations entre espèces. Les flèches vertes (extrémités triangulaires) indiquent des activations, les flèches rouges (extrémités en T) des inhibitions, les flèches noires (extrémité en cercle) des régulations d’action indéterminée. (b) Le même graphe d’interaction, où l’on a figuré une mesure (partielle) des sommets. La couleur verte d’un sommet indique un accroissement du niveau de l’espèce entre deux conditions expérimentales, la couleur rouge une diminution, la couleur grise l’absence de variation. L’absence de coloration signifie que le sommet n’a pas été mesuré

métaboliques. Nous représenterons de tels systèmes par un graphe orienté, appelé *graphe d’interaction*, dont les sommets sont les espèces chimiques présentes dans le système. Il peut typiquement s’agir de gènes (ou plus précisément, de leur transcrit), de protéines ou de métabolites. Les arcs de ce graphe représentent les régulations existant entre les différents espèces. Une flèche entre un produit A et un produit B signifie que A régule B . On parle d’*activation* (resp. d’*inhibition*) quand la présence de A tend à augmenter (resp. diminuer) le niveau de B . Pour représenter cette information, les arcs d’un graphe d’interaction sont étiquetés par des signes $\{+, 0, -, ?\}$ selon que la régulation a un effet activateur, nul, inhibiteur ou indéterminé respectivement. L’ensemble $\mathbb{S} = \{+, 0, -, ?\}$ est appelé *algèbre des signes* et nous l’aborderons plus en détail plus loin. Dans la suite on appellera graphe d’interaction un triplet $\mathcal{G} = (V, E, \rho)$ où V est l’ensemble des espèces chimiques du système, $E \subset V \times V$ est l’ensemble des régulations, et $\rho : E \rightarrow \mathbb{S}$ est un étiquetage des régulations par des signes. La figure 3.1(a) montre un exemple de graphe d’interaction.

Mesures expérimentales Comme nous l’avons évoqué au chapitre précédent, les données disponibles en biologie moléculaire sont le plus souvent de nature comparative : on ne mesure pas une grandeur, mais sa variation entre deux états d’un système, ou

deux conditions expérimentales. Le résultat est en général sous forme d'un ratio, dans des assertions du type « une dose d de X provoque une diminution de Y d'un ratio r ». Selon la technique expérimentale utilisée, ledit ratio est plus ou moins fiable ; il n'est d'ailleurs que très rarement donné avec une précision plus grande que l'unité. Nous proposons dans ce travail de ne conserver que le signe de la variation. À savoir, pour une expérience donnée, où l'on a comparé deux conditions expérimentales, nous appellerons *mesure* une application généralement partielle $\mu : V \rightarrow \mathbb{S} \setminus \{?\}$. Cette mesure ne rapporte que les signes définis, c'est-à-dire connus avec certitude. Le *domaine* (ensemble de départ) d'une mesure correspond donc à l'ensemble des sommets mesurés. La figure (3.1(b)) montre la représentation graphique que nous utiliserons par la suite pour les mesures.

3.1.2 Algèbre des signes

Jusqu'ici nous avons proposé de représenter les informations disponibles sur un réseau de réactions par un graphe dont les sommets et les arcs sont étiquetés par des signes. Pour pouvoir expliciter la relation existant entre ces différents étiquetages, il nous faut préciser cette notion de signe.

Définition et propriétés L'algèbre des signes est une abstraction de l'ensemble des réels, qui permet de raisonner sur le signe d'expressions arithmétiques. On peut la définir comme l'ensemble suivant de parties de \mathbb{R} :

\mathbb{S}	\rightarrow	$\mathcal{P}(\mathbb{R})$
$\mathbf{0}$	\rightarrow	$\{0\}$
$+$	\rightarrow	\mathbb{R}_+^*
$-$	\rightarrow	\mathbb{R}_-^*
$?$	\rightarrow	\mathbb{R}

On note $\text{sgn} : \mathbb{R} \rightarrow \mathbb{S}$ l'application qui associe son signe à un réel. Par extension, sgn associe à un ensemble réel le plus petit signe contenant cet ensemble. Pour toute opération \otimes sur les réels on peut définir une opération correspondante $\overline{\otimes}$ dans l'algèbre des signes, en posant :

$$s \overline{\otimes} t = \text{sgn}(\{x \otimes y \mid x \in s, y \in t\})$$

Nous utiliserons en particulier les opérations $\overline{+}$ et $\overline{\times}$ que nous noterons plutôt $+$ et \times . Voici, pour l'exemple, leur table :

$+$	$-$	$\mathbf{0}$	$+$	$?$	\times	$+$	$\mathbf{0}$	$+$	$?$
$-$	$-$	$-$	$?$	$?$	$-$	$-$	$\mathbf{0}$	$-$	$?$
$\mathbf{0}$	$-$	$\mathbf{0}$	$+$	$?$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$
$+$	$?$	$+$	$+$	$?$	$+$	$-$	$\mathbf{0}$	$+$	$?$
$?$	$?$	$?$	$?$	$?$	$?$	$?$	$\mathbf{0}$	$?$	$?$

Remarquons que $?$ est absorbant pour $+$ et $\mathbf{0}$ pour \times . Il reste enfin à ajouter l'abstraction de l'égalité, à savoir la compatibilité des signes, notée \approx , et définie par $s \approx t$

si et seulement si $s \cap t \neq \emptyset$. Cela signifie que deux expressions sont déclarées de signe compatible quand il est possible qu'elles soient de même signe. Cela donne la relation suivante :

+	-	0	+	?
-	<i>T</i>	<i>F</i>	<i>F</i>	<i>T</i>
0	<i>F</i>	<i>T</i>	<i>F</i>	<i>T</i>
+	<i>F</i>	<i>F</i>	<i>T</i>	<i>T</i>
?	<i>T</i>	<i>T</i>	<i>T</i>	<i>T</i>

dont il faut se méfier, parce qu'elle est bien réflexive et symétrique, mais pas transitive. Par exemple, $+\approx?$ et $?\approx-$ mais $+\not\approx-$. La fonction sgn vérifie par ailleurs la relation suivante :

$$\forall x, y, z \in \mathbb{R} \quad xy = z \Rightarrow \text{sgn}(x) \text{sgn}(y) = \text{sgn}(z) \Rightarrow \text{sgn}(x) \text{sgn}(y) \approx \text{sgn}(z)$$

ainsi que :

$$\forall x, y, z \in \mathbb{R} \quad x + y = z \Rightarrow \text{sgn}(x) + \text{sgn}(y) \approx \text{sgn}(z)$$

Contraintes dans l'algèbre des signes Dans la suite, nous allons étudier des *contraintes qualitatives*, que nous définissons comme un mot du langage suivant :

$$\begin{array}{l}
\mathbf{c} ::= \bigwedge_{i \in I} \mathbf{c}_i \\
\quad | \exists \mathbf{v} \mathbf{c} \mid \forall \mathbf{v} \mathbf{c} \\
\quad | \mathbf{p} \approx \mathbf{p} \\
\mathbf{p} ::= \mathbf{p} \times \mathbf{p} \mid \mathbf{p} + \mathbf{p} \mid - \mathbf{p} \\
\quad | \mathbf{v} \\
\quad | + \mid \mathbf{0} \mid - \mid ?
\end{array} \tag{3.1}$$

en prenant le symbole \mathbf{c} comme axiome et où les symboles non terminaux \mathbf{c} , \mathbf{p} et \mathbf{v} représentent une contrainte, un polynôme et une variable respectivement. Ce que nous appelons système qualitatif est donc un terme construit par induction, pouvant comporter des variables. On aura recours à la notation $\mathbf{C}[\sigma]$ pour désigner l'effet d'une substitution σ dans un terme, c'est-à-dire le remplacement de chaque variable libre \mathbf{v} de \mathbf{C} dans le domaine de σ par $\sigma(\mathbf{v})$.

La sémantique associée à ces termes est un peu particulière : on définit une *solution* d'une contrainte qualitative comme une valuation de toutes les variables libres de cette contrainte à valeurs dans $\{+, \mathbf{0}, -\}$ qui satisfait cette contrainte. La valeur $?$ est exclue dans les solutions d'une contrainte, parce qu'elle n'apporte aucune information. Si une variable peut prendre plusieurs valeurs pour lesquelles la contrainte est toujours vérifiée, alors la contrainte admet plusieurs solutions, une pour chaque alternative. Mais chaque solution est représentée sous la forme la plus précise possible, c'est-à-dire sans utiliser $?$.

Nous allons à présent formuler la consistance entre un graphe d'interaction et un ensemble de mesures comme une contrainte qualitative.

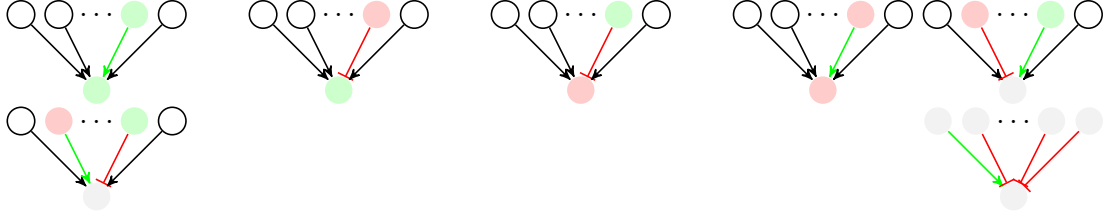


FIG. 3.2 – Résumé des cas où une variation d'une espèce est effectivement expliquée par la variation des espèces qui la régulent. Première ligne : le sommet cible admet une variation, on doit donc parmi les prédécesseurs pouvoir trouver une influence de même signe. Deuxième ligne : si la variation du sommet est nulle, alors soit on peut trouver deux contributions de signe opposé, soit tous les régulateurs ont une variation nulle

3.1.3 Contrainte de consistance

La notion de consistance que nous étudions dans ce travail exprime simplement que toute variation observée doit avoir une cause. Plus précisément, toute variation d'une espèce doit pouvoir être expliquée par la variation de l'un de ses régulateurs. Par exemple si A augmente entre deux conditions expérimentales, on doit pouvoir trouver une influence positive, par exemple l'action d'un activateur B de A qui augmente également. Les différents cas possibles sont résumés dans la figure (3.2).

Plus formellement, soit $\mathcal{G} = (V, E, \rho)$ un graphe d'interaction et $\{\mu_1, \dots, \mu_m\}$ un ensemble de mesures sur \mathcal{G} . Pour chaque sommet $i \in V$, et chaque mesure $k \in \{1, \dots, m\}$, on introduit la variable qualitative X_{ik} . Cette variable représente le signe de variation de l'espèce chimique i lors de l'expérience correspondant à la mesure k . On confondra la mesure μ_k avec la substitution σ définie pour tout $i \in \text{dom}(\mu_k)$ par $\sigma(X_{ik}) = \mu_k(i)$. Pour chaque arc $j \rightarrow i$ de \mathcal{G} , on introduit la variable S_{ji} , qui représente le signe de la régulation entre j et i . De même on confondra ρ avec la substitution σ définie pour tout $(j, i) \in E$ par $\sigma(S_{ji}) = \rho(j, i)$. On appelle contrainte de consistance au sommet i pour la mesure k , la contrainte

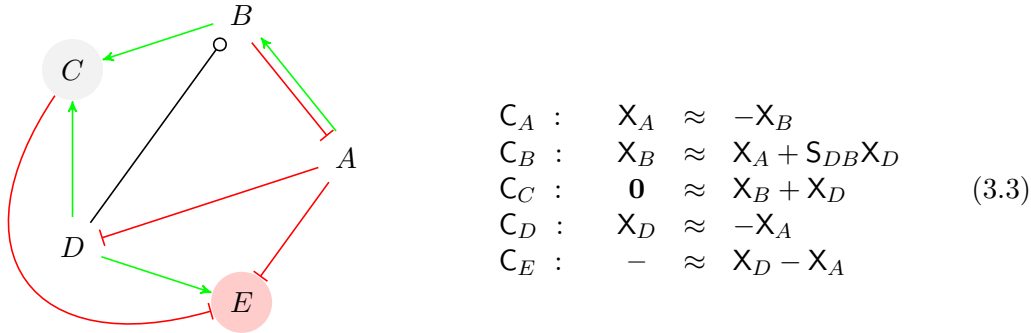
$$C_{ik} = \left(X_{ik} \approx \sum_{j \rightarrow i} S_{ji} X_{jk} \right) [\rho, \mu_k] \quad (3.2)$$

Nous dirons qu'un graphe d'interaction $\mathcal{G} = (V, E, \rho)$ est *consistant* avec les mesures $\mu = \{\mu_1, \dots, \mu_m\}$ si la contrainte

$$C_{\mathcal{G}}^{\mu} = \bigwedge_{i \in V, 1 \leq k \leq m} C_{ik}$$

admet une solution. Ce que nous appelons contrainte de consistance est donc une conjonction d'équations ; il y a une équation par sommet et par mesure. Chaque équation relie la variation du sommet associé à celle de ses prédécesseurs. Les variables libres sont les variations non observées et les signes de régulation inconnus. Les variations X_{ik} non observées sont propres à chaque mesure, mais les signes sur les régulations S_{ji} sont

partagées entre les équations associées à chaque mesure. Par convention, la contrainte C_G^\emptyset désigne la contrainte obtenue avec une seule mesure de domaine vide. Dans ce cas, on omettra généralement l'indice k . Cette contrainte sera utile notamment lorsque l'on voudra parler de l'ensemble des mesures compatibles avec un graphe d'interaction. Dans le reste du manuscrit, nous étudierons la contrainte C_G^μ et nous chercherons en particulier des moyens efficaces pour extraire de l'information de l'ensemble de ses solutions. Voici un premier exemple pour clarifier notre définition :



Le graphe d'interaction, ainsi qu'une mesure sont figurés à gauche selon les conventions que nous avons introduites plus haut. À droite figurent les contraintes de consistance à chaque sommet. Puisqu'il n'y a ici qu'une seule mesure, l'indice des mesures a été omis. La contrainte de consistance pour cet exemple est la conjonction $C_G^\mu = C_A \wedge C_B \wedge C_C \wedge C_D \wedge C_E$. En voici les solutions :

X_A	X_B	X_D	S_{DB}	
+	-	-	+	(3.4)
-	+	+	+	

En pratique il peut s'avérer utile de pouvoir spécifier que certains sommets du graphe d'interaction sont des *entrées* du modèle, c'est-à-dire que leur variation n'est pas expliquée dans le modèle. Pour ces sommets, on se dispense donc d'ajouter une contrainte. En particulier, si un sommet n'a pas de prédécesseur dans le graphe d'interaction, alors son unique variation admissible est $\mathbf{0}$, ce qui fait disparaître sa contribution dans toutes les autres contraintes. Pour cette raison, les sommets sans prédécesseur dans le graphe d'interaction seront systématiquement considérés comme des entrées du système.

Définition 1. (*Consistance aux sommets, \mathcal{N} -consistance*) Soit un graphe d'interaction $\mathcal{G} = (V, E, \rho)$, muni d'un ensemble d'entrées U , tel que $\{v \in V \mid \deg_{\mathcal{G}}^+(v) = 0\} \subset U$. Sauf mention contraire U sera par défaut exactement l'ensemble des sommets sans prédécesseurs. \mathcal{G} est dit \mathcal{N} -consistant avec un ensemble μ de mesures si la contrainte qualitative

$$C_G^\mu = \bigwedge_{i \in V \setminus U, 1 \leq k \leq m} C_{ik}$$

admet au moins une solution.

Passons à présent aux propriétés générales des contraintes qualitatives.

3.1.4 Propriétés des contraintes qualitatives

Les contraintes qualitatives peuvent ne pas avoir de solution, et quand elles en ont, il n'y a pas nécessairement unicité (comme nous venons de le voir dans l'exemple précédent). La propriété suivante donne des indications sur quelques cas particuliers.

Proposition 1. *Soient $\mathcal{G} = (V, E, \rho)$ un graphe d'interaction et $\mu = \{\mu_1, \dots, \mu_m\}$ un ensemble de mesures.*

1. $\mathcal{C}_{\mathcal{G}}^{\emptyset}$ admet une solution.
2. si s est une solution de $\mathcal{C}_{\mathcal{G}}^{\emptyset}$ alors $-s$ est encore une solution
3. si $\text{dom}(\rho) = E$ alors toute variable de $\mathcal{C}_{\mathcal{G}}^{\mu}$ peut prendre soit une valeur, soit les valeurs $+$ et $-$ mais pas $\mathbf{0}$, soit n'importe quelle valeur.

Démonstration. 1. C'est la solution nulle.

2. On vérifie facilement que si pour $s, t \in \mathbb{S}$, $(-s)+(-t) = -(s+t)$, $s \times (-t) = -(s \times t)$ et $s \approx t \Rightarrow -s \approx -t$. L'assertion est ainsi prouvée par induction sur les termes constituant une contrainte de consistance.
3. si $\text{dom}(\rho) = E$ alors $\mathcal{C}_{\mathcal{G}}^{\mu}$ est une conjonction de contraintes linéaires, et le résultat est alors donné par la proposition 1.12 à la page 27 de [94].

□

La résolution des contraintes qualitatives a été particulièrement étudiée dans le cas de contraintes linéaires [93]. Ces développements proposent de transposer les outils connus en algèbre linéaire dans le cas qualitatif, comme le pivot de Gauss ou le déterminant. Sur le plan algorithmique, cette tentative n'aboutit pas à des outils particulièrement performants, et qui sont de plus limités au cas linéaire.

Le problème de résolution de contraintes qualitatives est prouvé NP-complet dans le cas des systèmes linéaires comme nous l'illustrons maintenant.

Théorème 1. *La construction d'une solution à une contrainte qualitative linéaire est un problème NP-complet.*

Démonstration. On procède par réduction polynomiale de SAT. Soit à résoudre un ensemble de clauses $C = \{C_1, \dots, C_r\}$ sur un ensemble de variable V . Chaque clause C_i est une disjonction de littéraux de la forme v ou $\neg v$. Pour arriver au résultat, il suffit de trouver un codage de C en un système qualitatif calculable en temps polynomial. L'idée est d'associer la valeur \mathbf{T} à $+$ et \mathbf{F} à $-$. Une variable propositionnelle v de V est envoyée sur une variable qualitative \bar{v} . Voici un tableau synthétisant la procédure de

codage d'une clause :

clause		contrainte qualitative
T	→	+
F	→	-
$a \in V$	→	\bar{a}
$\neg a, a \in V$	→	$-\bar{a}$
$l_1 \vee l_2 \cdots \vee l_k$	→	$\bar{l}_1 + \bar{l}_2 \cdots + \bar{l}_k$

La contrainte résultante de C est donnée par :

$$\bigwedge_{i=1, \dots, n} \bar{C}_i \approx +$$

Voici un exemple de la transformation : soit l'ensemble de clauses $C = \{x_1 \vee x_2, \neg x_4 \vee x_2 \vee x_3, \neg x_1 \vee x_4\}$. La contrainte résultant du codage est :

$$C = (\bar{x}_1 + \bar{x}_2 \approx +) \wedge (\bar{x}_3 + \bar{x}_2 - \bar{x}_4 \approx +) \wedge (\bar{x}_4 - \bar{x}_1 \approx +)$$

Il nous faut montrer comment une solution à ce système fournit une solution qui satisfait les clauses de C . Si dans la valuation obtenue, on a $\bar{v} = +$ (resp. $\bar{v} = -$) pour $v \in V$, alors on pose $v = \mathbf{T}$ (resp. $v = \mathbf{F}$). Si on a $\bar{v} = \mathbf{0}$, alors on choisit une valeur quelconque pour v . Soit une clause $C_i \in C$, on note l'ensemble de ses littéraux $\{l_1, \dots, l_k\}$. La valuation obtenue est une solution du système qualitatif donc au moins un des littéraux, disons l_j , est tel que $\bar{l}_j = +$. Soit par construction tel que $l_j = \mathbf{T}$. Réciproquement, toute valuation des variables propositionnelles satisfaisant C fournit une solution à la contrainte qualitative correspondante. \square

Toutefois il n'est pas immédiat de savoir si le problème reste difficile dans le cas des contraintes issues de graphes d'interaction. Par exemple, dans le cas où le graphe d'interaction est acyclique et les seuls sommets observés sont les sommets sans prédécesseurs, la résolution s'avère particulièrement simple :

Proposition 2. *Soit $\mathcal{G} = (V, E, \rho)$ un graphe d'interaction acyclique et $\mu = \{\mu_1, \dots, \mu_m\}$ un ensemble de mesures. Si pour tous i et k , $X_{ik} \in \text{dom}(\mu_k) \Rightarrow \text{deg}_{\mathcal{G}}^+(i) = 0$ alors $\mathcal{C}_{\mathcal{G}}^{\mu}$ admet une solution, calculable en temps polynomial.*

Démonstration. Ce résultat est un cas particulier de la proposition 7 qui sera donnée plus loin. Contentons-nous ici d'esquisser informellement la preuve.

Tout d'abord, si certains signes sur les arcs de \mathcal{G} sont inconnus, on les fixe à des valeurs arbitraires. Une fois fait, on partitionne les contraintes de consistance aux sommets selon les mesures, c'est-à-dire les sous-systèmes $\mathcal{C}_{\mathcal{G},k}^{\mu_k}$ pour $k = 1, \dots, m$ où

$$\mathcal{C}_{\mathcal{G},k}^{\mu_k} = \bigwedge_{i \in V} C_{ik}$$

Puisque tous les signes sur les arcs sont maintenant instanciés, les sous-systèmes que nous avons construits ne partagent aucune variable. En effet, comme mentionné plus

haut, si l'on partitionne les équations de la contrainte de consistance selon les mesures, alors les seules variables partagées sont les signes sur les arcs. Par conséquent, les sous-systèmes que nous avons isolés peuvent être résolus séparément.

Pour chaque sous-système, on décide de l'affectation de chaque variable suivant un tri topologique de \mathcal{G} (qui existe, puisque \mathcal{G} est acyclique). On commence donc par les sommets sans prédécesseurs, qui sont nécessairement des entrées. Soit ils sont dans le domaine de la mesure, auquel cas leur valeur est fixée, soit on choisit leur valeur arbitrairement. Ensuite, pour chaque sommet, on décide de sa valeur *après* avoir décidé de la valeur de ses prédécesseurs. Il suffit donc de prendre une valeur compatible avec la somme qualitative sur les prédécesseurs, ce qui est toujours possible. \square

En revanche dans le cas général, le problème est encore NP-complet pour les systèmes issus d'un graphe d'interaction :

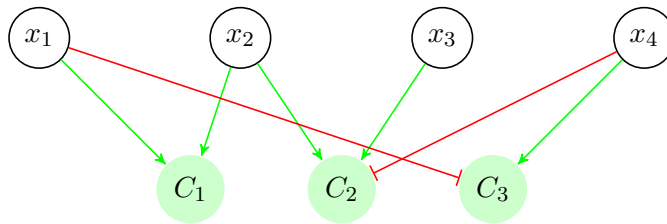
Théorème 2. *La construction d'une solution pour la contrainte $C_{\mathcal{G}}^{\mu}$ associé à un graphe d'interaction \mathcal{G} et à un ensemble de mesures μ est un problème NP-complet.*

Démonstration. On procède encore par réduction polynomiale de SAT. Soit à résoudre un ensemble de clauses $\{C_1, \dots, C_r\}$ sur un ensemble de variables $\{x_1, \dots, x_s\}$. Chaque clause C_i est un ensemble de littéraux de la forme x_k ou $\neg x_k$.

Soit le graphe (bipartite) \mathcal{G} :

- dont l'ensemble des sommets est $\{C_1, \dots, C_r\} \cup \{x_1, \dots, x_s\}$,
- et où l'arc $x_j \rightarrow C_i$ existe si x_j apparaît dans C_i . Cet arc est étiqueté par le signe $+$ si $x_j \in C_i$ et par $-$ si $\neg x_j \in C_i$.

Enfin, on construit une mesure μ telle que $\mu(C_i) = +$ et n'est pas définie ailleurs. Voici un exemple de cette construction, sur l'ensemble de clauses $C = \{x_1 \vee x_2, \neg x_4 \vee x_2 \vee x_3, \neg x_1 \vee x_4\}$:



La contrainte qualitative de consistance aux sommets entre \mathcal{G} et μ est alors exactement celle utilisée dans la preuve du théorème 1, ce qui prouve le résultat. \square

3.2 Justification différentielle

Le modèle de régulation que nous avons introduit au paragraphe précédent est relativement simple et par certains égards assez intuitif. Néanmoins, comment déterminer s'il est une représentation adéquate des mécanismes à l'œuvre dans une cellule? Le meilleur moyen de trancher est bien entendu de vérifier par l'expérience les prédictions

que l'on peut en tirer, et c'est l'objet des chapitres 4 et 5. Nous proposons dans ce paragraphe de montrer que notre formalisme peut être vu comme une conséquence d'un modèle différentiel général de cinétique chimique.

Les équations différentielles ordinaires sont omniprésentes en biologie des systèmes, et servent fréquemment à modéliser des réseaux biologiques de nature très différentes (métabolisme [27, 37], transduction de signal [41, 50], régulation génétique [104, 18, 68]). Les utiliser comme hypothèse pour dériver notre formalisme n'est donc pas absurde ; l'intérêt principal de cette démarche sera de préciser les conditions d'application des contraintes de consistance, et de guider l'interprétation des données expérimentales. Nous commençons par introduire le formalisme différentiel, puis par énoncer un ensemble d'hypothèses suffisant pour démontrer les contraintes de consistances dans le cadre différentiel.

3.2.1 Graphe d'interaction

Définition On considère un ensemble de réactions chimiques impliquant n espèces. On note X le vecteur de concentration. Le système est supposé suivre une dynamique différentielle :

$$\frac{dX}{dt} = F(X, U) \quad (3.5)$$

où U est le vecteur de taille p représentant des entrées du système, c'est-à-dire des variables contrôlées par l'environnement. La fonction F est une description quantitative du système de réactions ; elle est généralement inconnue.

Si F est dérivable, la *matrice jacobienne* de F en (X, U) est la matrice $(J(X, U) \ K(X, U))$ où $J(X, U) = (\frac{\partial F_i}{\partial X_j}(X, U))_{ij}$ et $K(X, U) = (\frac{\partial F_i}{\partial U_j}(X, U))_{ij}$. On appelle *graphe d'interaction* en (X, U) de F le graphe $\mathcal{G}(X, U)$:

- dont les nœuds sont les entiers $1, \dots, n + p$
- où l'arc $j \rightarrow i$ existe si $\frac{\partial F_i}{\partial X_j}(X, U) \neq 0$ ou $\frac{\partial F_i}{\partial U_j}(X, U) \neq 0$. Les arcs sont étiquetés par le signe de la dérivée partielle, qu'on notera $s(j, i)$.

Notons que $\mathcal{G}(X, U)$ est bien un graphe d'interaction au sens défini dans le paragraphe précédent : il y a un arc entre j et i si le niveau (ou concentration) de j influe sur la vitesse de production de i . La différence majeure ici, c'est que le graphe d'interaction peut dépendre de l'état et des entrées si F est non linéaire. Pour un état et une entrée données, $\mathcal{G}(X, U)$ est appelé graphe d'interaction *local* ; le graphe d'interaction *global* est obtenu en calculant l'union des graphes d'interaction locaux, et en prenant pour étiquette d'un arc la somme qualitative des signes apparaissant sur cet arc pour tous les graphes locaux.

Topologie et propriétés dynamiques La topologie des graphes d'interaction d'un système donné peut renseigner sur ses propriétés dynamiques. Un premier résultat, appelé *première conjecture de Thomas* dit informellement que la présence d'un circuit positif dans le graphe d'interaction est une condition nécessaire à l'existence de plusieurs états stables. Ce résultat a été énoncé dans différents cadres, plus ou moins généraux,

aussi bien différentiels que discrets. Soulé en propose une excellente revue dans [89] et y démontre le théorème suivant :

Théorème 3 (Conjecture de Thomas, modèles différentiels avec dégradation). *Soient $\Omega \subset \mathbb{R}^n$ un produit d'intervalles ouverts, $F = (F_i) : \Omega \rightarrow \mathbb{R}^n$ un vecteur de fonctions différentiables. On considère le système d'équations différentielles :*

$$\frac{dX_i}{dt} = F_i(X) - \gamma_i X_i$$

où $\gamma_1, \dots, \gamma_n$ sont des réels strictement positifs.

Si l'on peut trouver deux vecteurs distincts X et Y dans Ω tels que

$$|F_i(X) - \gamma_i X_i - F_i(Y) + \gamma_i Y_i| < \gamma_i |X_i - Y_i|$$

alors il existe $Z \in \Omega$ tel que $\mathcal{G}(Z)$ contient un circuit positif.

On pourra également trouver des résultats plus précis dans [49] ou même d'autres résultats concernant la stabilité des états d'équilibres [92].

La *deuxième conjecture de Thomas* dit quant à elle que la présence d'un circuit négatif dans le graphe d'interaction est une condition nécessaire à l'existence d'oscillations stables ou amorties [87].

Enfin, une classe importante de systèmes dynamiques, appelés *systèmes monotones* [1] est caractérisable en termes de graphe d'interaction. Plus précisément si un système dynamique est de graphe d'interaction constant, et que celui-ci ne contient aucun cycle (non-orienté) négatif, alors il est monotone [20]. Entre autres propriétés, les systèmes monotones (bornés) convergent presque sûrement pour toute condition initiale vers un état d'équilibre stable ; ils n'admettent pas d'attracteurs chaotique, ni même d'orbite périodique.

3.2.2 Réponse statique à une perturbation

Un vecteur X est un état d'équilibre à entrée constante u si $F(X, u) = 0$; il est dit stable si toutes les valeurs propres de $J(X, u)$ ont leur partie réelle négative ; enfin il est dit non dégénéré quand $\det J(X, u) \neq 0$.

Il est classique d'étudier un système différentiel non-linéaire autour d'un point d'équilibre stable (x, u) , en remplaçant l'équation (3.5), par sa linéarisation :

$$\frac{d\Delta X}{dt} = J(x, u)\Delta X \quad (3.6)$$

Ce système admet un voisinage dans lequel ses trajectoires sont confinées. Parce que linéaire, il est simple à étudier et l'allure de ses trajectoires est caractérisée par les valeurs propres de $J(x, u)$. L'intérêt de la manœuvre vient de ce que les trajectoires du système linéaire sont qualitativement similaires à celle du système d'origine. Plus précisément le théorème de Hartman-Grobman dit que les trajectoires dans l'un et l'autre cas sont homéomorphes (identiques à une fonction inversible et continue près). On peut ainsi étudier le comportement des trajectoires autour du point d'équilibre.

Dans la suite, nous ne nous préoccupons jamais de la trajectoire du système dynamique, mais seulement de la forme de ses *nullclines*. Une nullcline est une variété $\{F_i(X, U) = 0\}$, et l'ensemble des états d'équilibre est l'intersection des nullclines. En différentiant l'équation $F_i(X, U) = 0$, on calcule l'espace tangent en un point (x, u) de la i^{e} nullcline :

$$\sum_j \frac{\partial F_i}{\partial X_j}(x, u) dX_j + \sum_k \frac{\partial F_i}{\partial U_k}(x, u) dU_k = 0$$

ou encore sous forme matricielle :

$$J(x, u)dX + K(x, u)dU = 0 \quad (3.7)$$

Cette relation détermine l'effet (au premier ordre) dX d'une perturbation dU des entrées du système. Elle montre le déplacement du point d'équilibre initial sous l'action d'une perturbation, quand celle-ci est suffisamment faible. Ce que nous verrons dans la suite, c'est ce qu'il reste de cette relation pour des perturbations d'intensité quelconque.

3.2.3 Hypothèses de modélisation

Comme nous venons de le voir, le graphe d'interaction d'un système différentiel dépend en toute généralité de l'état dudit système. Dans la mesure où notre formalisme ne représente pas l'état du système (mais seulement les variations d'états), il ne nous est *a priori* pas possible de traiter des systèmes où le signe d'un arc du graphe d'interaction peut varier en fonction de l'état ou des entrées du système. Il nous faut donc explicitement rajouter l'hypothèse suivante :

Hypothèse 1 (H1). *On considère une dynamique différentielle $\frac{dX}{dt} = F(X, U)$ de graphe d'interaction constant, définie sur \mathbb{R}^{n+p} .*

On appelle *mesure* un quatre-uplet $(x^{(1)}, u^{(1)}, x^{(2)}, u^{(2)})$. Cette définition est cohérente avec celle que nous avons donnée précédemment ; pour construire une mesure qualitative, il suffit de calculer le vecteur des $\text{sgn}(x_i^{(2)} - x_i^{(1)})$.

Hypothèse 2 (H2). *Toute mesure $(x^{(1)}, u^{(1)}, x^{(2)}, u^{(2)})$ est telle que $x^{(1)}$ (resp. $x^{(2)}$) est un état d'équilibre sous $u^{(1)}$ (resp. $u^{(2)}$) stable et non dégénéré.*

Cette hypothèse signifie que nous restreignons notre analyse aux expériences de *déplacement d'équilibre* : un système initialement au repos subit une perturbation, puis revient à l'équilibre au bout d'un certain temps. Les données disponibles concernent alors la variation entre état final et état initial.

L'hypothèse suivante dit essentiellement que les espèces du système sont soumises à un phénomène de dégradation, d'intensité au moins linéaire en fonction de la concentration. C'est une hypothèse analogue à celle utilisée par Soulé dans [88].

Hypothèse 3 (H3). *Pour toute espèce i du système, on a*

$$\exists \kappa_i > 0 \forall (X, U) \in \mathbb{R}_+^{n+p} \frac{\partial F_i}{\partial X_i}(X, U) < -\kappa_i \quad (3.8)$$

Enfin, notre dernière hypothèse découle de ce que les concentrations sont des grandeurs positives.

Hypothèse 4 (H4). *Pour toute espèce i du système et pour tout u , on a*

$$F_i(X_1, \dots, X_i = 0, \dots, u) \geq 0 \quad (3.9)$$

3.2.4 Déplacement d'équilibre et variations

Dans le cadre des hypothèses précédentes, nous proposons de déterminer des relations entre la variation d'un sommet du graphe d'interaction et celle de ses prédécesseurs. Nous commençons par une version globale du théorème des fonctions implicites. On notera \hat{X}^i le vecteur X dont on a ôté la i^e coordonnée et \hat{Y}^i le vecteur (\hat{X}^i, U) .

Théorème 4. *Sous les hypothèses H1, H3 et H4, pour tout \hat{Y}^i , l'équation $F_i(\hat{Y}^i, X_i) = 0$ admet une unique solution en X_i . Soit la fonction Φ_i définie par*

$$\forall Y \quad F_i(Y) = 0 \Leftrightarrow X_i = \Phi_i(\hat{Y}^i)$$

Alors

1. Φ_i est dérivable
2. Φ_i vérifie pour tout Z variable de \hat{Y}^i

$$\frac{\partial \Phi_i}{\partial Z}(\hat{Y}^i) = \left(\frac{\partial F_i}{\partial X_i}(\hat{Y}^i, \Phi_i(\hat{Y}^i)) \right)^{-1} \frac{\partial F_i}{\partial Z}(\hat{Y}^i, \Phi_i(\hat{Y}^i))$$

Démonstration. Soit l'application $\phi_{i, \hat{Y}^i} : X_i \mapsto F_i(\hat{Y}^i, X_i)$. Par l'hypothèse H4, on sait que $\phi_{i, \hat{Y}^i}(0) \geq 0$. La fonction ϕ_{i, \hat{Y}^i} est dérivable parce que F_i l'est et l'hypothèse H3 implique que ϕ'_{i, \hat{Y}^i} est partout négative. Plus précisément,

$$\phi_{i, \hat{Y}^i}(x) = \int_0^x \phi'_{i, \hat{Y}^i}(s) ds + \phi_{i, \hat{Y}^i}(0) \quad (3.10)$$

$$= \int_0^x \frac{\partial F_i}{\partial X_i}(\hat{Y}^i, s) ds + \phi_{i, \hat{Y}^i}(0) \quad (3.11)$$

$$\leq \int_0^x -\kappa_i ds + \phi_{i, \hat{Y}^i}(0) \quad (3.12)$$

$$\leq -\kappa_i x + \phi_{i, \hat{Y}^i}(0) \quad (3.13)$$

On déduit de cette borne que $\phi_{i, \hat{Y}^i}(\frac{F(\hat{Y}^i, 0)}{\kappa_i}) \leq 0$, puis par continuité et monotonie de ϕ_{i, \hat{Y}^i} qu'il existe un unique réel, noté $\Phi_i(\hat{Y}^i)$, tel que $\phi_{i, \hat{Y}^i}(\Phi_i(\hat{Y}^i)) = 0$.

On a ainsi démontré l'existence et l'unicité de Φ_i . La dérivabilité et la formule des dérivées partielles découlent de l'application du théorème des fonctions implicites, version locale. \square

Ce résultat permet d'établir une formule quantitative exprimant la variation entre deux états stationnaires.

Théorème 5. *Sous les hypothèses H1-H4, si $Y^{(1)}$ et $Y^{(2)}$ sont deux états stationnaires du système alors la variation en concentration d'une espèce i entre les 2 états stationnaires $Y^{(1)}$ et $Y^{(2)}$ est donnée par :*

$$X_i^{(2)} - X_i^{(1)} = \int_S - \left(\frac{\partial F_i}{\partial X_i}(\hat{Y}^i, \Phi_i(\hat{Y}^i)) \right)^{-1} \sum_{k \rightarrow i} \frac{\partial F_i}{\partial Z_k}(\hat{Y}^i, \Phi_i(\hat{Y}^i)) dZ_k \quad (3.14)$$

où S est un chemin régulier quelconque entre $Y^{(1)}$ et $Y^{(2)}$, et Z_k représente la variable U_k ou X_k selon que k est une entrée ou non.

Démonstration. Soit S une courbe régulière quelconque entre $Y^{(1)}$ et $Y^{(2)}$. On a :

$$X_i^{(2)} - X_i^{(1)} = \Phi_i(Y^{(2)}) - \Phi_i(Y^{(1)}) \quad (3.15)$$

$$= \int_S d\Phi_i \quad (3.16)$$

$$= \int_S \sum_{Z \in \hat{Y}^i} \frac{\partial \Phi_i}{\partial Z} dZ \quad (3.17)$$

$$= \int_S \sum_{Z \in \hat{Y}^i} - \left(\frac{\partial F_i}{\partial X_i}(\hat{Y}^i, \Phi_i(\hat{Y}^i)) \right)^{-1} \frac{\partial F_i}{\partial Z}(\hat{Y}^i, \Phi_i(\hat{Y}^i)) dZ \quad (3.18)$$

On obtient la bonne formule en notant que si $k \not\rightarrow i$ dans le graphe d'interaction, alors $\frac{\partial F_i}{\partial Z_k} = 0$ \square

À ce stade, il nous faut faire deux remarques. La première c'est que si l'hypothèse H3 ou H4 n'est pas vérifiée pour un sommet, rien n'empêche de dériver cette relation quantitative pour les autres sommets. Deuxièmement, nous n'avons à aucun moment parlé du chemin *réel* suivi par le système entre les deux états stationnaires. En particulier, ce chemin n'est pas celui utilisé pour l'intégration qui donne la formule.

Nous pouvons à présent donner une preuve de la contrainte de consistance à un sommet i (3.2). Il s'agit comme nous allons le voir d'une version qualitative du théorème (5).

Théorème 6. *Soit $\mathcal{G} = (V, E, \rho)$ un graphe d'interaction. Sous les hypothèses H1-H4, la variation $x_i^{(2)} - x_i^{(1)}$ au sommet i vérifie :*

$$\text{sgn}(x_i^{(2)} - x_i^{(1)}) \approx \sum_{k \rightarrow i} \rho(k, i) \text{sgn}(z_k^{(2)} - z_k^{(1)}) \quad (3.19)$$

où $z_k^{(l)}$ représente la variable $u_k^{(l)}$ ou $x_k^{(l)}$ selon que k est une entrée ou non.

Démonstration. Réordonner les termes dans l'équation (3.14) grâce à la commutativité somme/intégrale (la somme est finie) :

$$x_i^{(2)} - x_i^{(1)} = \sum_{k \rightarrow i} \int_S \left(-\frac{\partial F_i}{\partial X_i} \right)^{-1} \frac{\partial F_i}{\partial X_k} dZ_k \quad (3.20)$$

Le terme $\left(-\frac{\partial F_i}{\partial X_i} \right)^{-1} \frac{\partial F_i}{\partial Z_k}$ est de signe constant, et on a :

$$\text{sgn} \left(\left(-\frac{\partial F_i}{\partial X_i} \right)^{-1} \frac{\partial F_i}{\partial Z_k} \right) = \text{sgn} \left(-\frac{\partial F_i}{\partial X_i} \right) \text{sgn} \left(\frac{\partial F_i}{\partial Z_k} \right) \quad (3.21)$$

$$= r(k, i) \quad (3.22)$$

En utilisant la formule de la moyenne, on sait qu'il existe $\bar{y} \in S$ tel que :

$$\int_S \left(-\frac{\partial F_i}{\partial X_i} \right)^{-1} \frac{\partial F_i}{\partial Z_k} dX_k = A_{ki} (x_k^{(2)} - x_k^{(1)}) \quad (3.23)$$

avec $A_{ki} = \left(-\frac{\partial F_i}{\partial X_i}(\bar{y}, \Phi_i(\bar{y})) \right)^{-1} \frac{\partial F_i}{\partial Z_k}(\bar{y}, \Phi_i(\bar{y}))$. On obtient ainsi :

$$z^{(2)} - z^{(1)} = \sum_{k \in \text{pred}(i)} A_{ki} (z_k^{(2)} - z_k^{(1)}) \quad (3.24)$$

avec $\text{sgn}(A_{ki}) = r(k, i)$. D'où le résultat. \square

3.2.5 Discussion

Dynamique inconnue Dans le développement qui précède, nous avons supposé que la seule information disponible sur le système était synthétisée dans le graphe d'interaction. Notre motivation est de produire une méthode applicable sur de grands systèmes où, en l'état actuel, la plupart des réactions ont une cinétique inconnue. Notons que toute conclusion obtenue dans ce cadre aura l'avantage d'être très générale car ne dépendant pas d'une forme particulière pour la cinétique. L'obtention de résultats indépendants du type de cinétique choisi ou de ses paramètres est un objectif important dans les modèles par équations différentielles [16, 90, 73]. Il est en effet techniquement très difficile d'estimer précisément la cinétique réelle des équations : il faut pour cela isoler une réaction dans une condition expérimentale où les mesures sont possibles, ce qui demande énormément de travail, quand cela est possible. De plus cette condition est souvent fort différente du milieu intra-cellulaire, fort complexe et très encombré. Pour cette raison, les constantes cinétiques durement acquises peuvent donc être bien loin de la réalité. Les prédictions valides pour un large spectre de cinétiques sont pour cette raison plus fiables. On peut également invoquer une deuxième raison : les cellules sont connues pour être des systèmes particulièrement robustes à des changements de condition (température, pression, concentration ...). On préfère donc d'une manière générale que les prédictions ne dépendent pas d'une valeur précise d'un paramètre.

Il est vrai néanmoins (et nous y reviendrons plus loin), que l'on connaît des grands types de cinétiques pour les réactions biochimiques, comme les fonctions de Hill pour les régulations génétiques, les lois d'action de masse pour la signalisation ou encore

les réactions de type Michaelis-Mentens pour le métabolisme [95]. Chaque type est caractérisé par un certain nombre de paramètres qui décrivent complètement la cinétique d'une réaction. Il existe *grosso modo*, deux façons d'utiliser cette information :

- une approche numérique, qui consiste à estimer ces paramètres à partir d'un nombre fini de mesures, soit à l'aide de séries temporelles ([11]), soit à l'aide de données de perturbations [12], telles que celles que nous utilisons. Pour un système comportant plusieurs dizaines de réactions, l'estimation des paramètres requiert un nombre de mesures et une précision déraisonnables, compte tenu des techniques actuelles.
- une approche qualitative, qui consiste à étudier des abstractions discrètes des modèles différentiels [9]. Les techniques existantes sont très adaptées à la qualité des mesures disponibles, mais d'une complexité prohibitive pour le traitement de grands systèmes. De plus, ces approches sont limitées aux réseaux génétiques.

Dans les deux cas, il s'agit donc d'outils réservés à l'étude fine de « petits » systèmes.

Graphe d'interaction constant Nous l'avons vu à plusieurs reprises, le graphe d'interaction dépend en toute généralité de l'état et des entrées du système. Or l'hypothèse H1 stipule que nous ne travaillons qu'avec des graphes d'interaction constants. Cette limitation n'a que peu de conséquences en pratique. La principale raison en est que pour dériver la contrainte de consistance à un sommet, nous n'avons utilisé que des conditions locales à ce sommet (dégradation, positivité des concentrations, influences de signe constant). Par conséquent, si l'une de ces conditions n'est pas remplie pour un sommet, cela n'affecte en rien les autres sommets du graphe d'interaction. Si une interaction n'est pas de signe constant, elle invalide la contrainte du sommet à son extrémité, mais rien de plus.

Dans nos expériences sur données réelles, nous avons trouvé des cas où le signe d'une interaction peut changer en fonction de l'état. Un tel exemple est décrit chez *E. coli* dans [40] : le promoteur du gène *cdd* contient trois sites de fixation le facteur Crp, d'affinités différentes. L'étude met en évidence que Crp se lie à des sites différents selon que la protéine CytR est présente ou non. La conformation globale empêche la transcription dans le premier cas, et la permet dans le deuxième.

En pratique, il faut donc disposer de moyens permettant de détecter les régulations de signe non constant, ce que nous verrons au chapitre suivant.

Séries temporelles L'hypothèse H2 restreint le champ d'applicabilité de notre méthode aux données de perturbation. Néanmoins il est assez courant de disposer de séries temporelles, c'est-à-dire de mesures établies en régime transitoire. Le principal intérêt de ce type de mesure est de faire apparaître une notion de causalité : si deux gènes g et g' ont des variations systématiquement identiques sous diverses conditions, il n'est pas possible à partir de données de perturbation de décider si g régule g' , ou si g' régule g , ou bien encore si g et g' sont régulés par un gène tiers. Des mesures temporelles peuvent permettre de lever cette ambiguïté si l'on observe systématiquement un gène varier *avant* l'autre, ou si les deux gènes varient de façon synchrone. Il peut donc être – selon le contexte – particulièrement dommageable de ne pas utiliser de telles données.

Nuançons quelque peu ce problème : il existe une littérature relativement riche (voir [30, 96, 4] par exemple) sur la reconstruction de réseaux génétiques à partir de séries temporelles, et se basant sur la modélisation suivante. En notant X le vecteur des concentrations, et U le vecteur des entrées, ces travaux proposent de modéliser un réseau génétique par la relation :

$$\dot{X}(t_k) = AX(t_k) + BU(t_k) \quad (3.25)$$

où \dot{X} représente le vecteur dérivé de X , et où A et B sont des matrices. Autrement dit, ces travaux proposent de modéliser un réseau génétique par un système différentiel linéaire. Il est clair, et particulièrement dans le cas des réseaux génétiques, que cette hypothèse est fautive. Sur le plan quantitatif, il s'agit donc de modèles trop approximatifs. En revanche, si le but se limite à reconstruire le graphe d'interaction, ces algorithmes ont un comportement tout à fait respectable, comme le démontre la comparaison effectuée dans [5].

Exploiter la relation 3.25 en qualitatif (c'est-à-dire l'interpréter dans l'algèbre des signes) est particulièrement simple. D'un point de vue pratique, cela apporterait quelques avantages : 1. exploiter des séries temporelles fortement bruitées, 2. lever l'hypothèse de linéarité (linéaire dans l'algèbre des signes n'implique absolument pas linéarité en quantitatif). On pourrait notamment comparer les graphes obtenus dans les deux approches : traitement des données quantitatives puis abstraction en graphe d'interaction d'une part ; abstraction des données en signes puis raisonnements qualitatifs d'autre part.

Bilan Dans cette section, nous avons montré que si l'on peut parler *du* graphe d'interaction d'un système, on peut sous certaines hypothèses dériver les contraintes de consistance aux sommets. En particulier, ce cadre s'applique aux expériences où un système initialement au repos converge vers un nouvel état d'équilibre après une perturbation de ses entrées. Rappelons qu'aucune hypothèse n'a été faite sur l'intensité de cette perturbation.

L'objectif des paragraphes qui suivent est de revenir sur l'hypothèse du graphe d'interaction constant. Il s'agit de mesurer les limites de cette hypothèse – tout autant que de constater qu'elle est relativement générale, à condition de décrire le système de manière suffisamment précise. Pour cela, on procédera en deux temps : d'abord on étudiera le cas des cinétiques usuelles en modélisation, avec un système réduit à une réaction ; puis on verra sous quelles conditions on peut conserver l'hypothèse dans les systèmes à plusieurs réactions. Comme sous-produit important de cette étude, on montrera sous quelles hypothèses on peut déduire le graphe d'interaction d'un ensemble de réactions, sans en demander une description explicite ou quantitative.

3.2.6 Cinétiques usuelles en modélisation

La première étape consiste à montrer que les cinétiques habituellement utilisées dans les modèles différentiels de réseaux biologiques donnent lieu à des graphes d'interaction

où les signes sont constants. Nous passons en revue quelques unes des cinétiques les plus courantes [95].

Cinétiques linéaires Elles sont de la forme : $\Phi(X) = \lambda'X$, où λ est un vecteur de réel et λ' désigne la transposée de λ . Les cinétiques linéaires sont utilisées pour modéliser des phénomènes de transport passif d'un compartiment à un autre, la dégradation des espèces chimiques ou la dilution lors de la croissance de bactéries. Leur dérivée partielle est donnée par :

$$\frac{\partial \phi}{\partial X_j} = \lambda_j$$

qui est signe constant.

Lois d'action de masse Cette cinétique constitue une bonne approximation lorsque les réactions sont des processus élémentaires (transformations chimiques simples). Elles sont de la forme $\Phi(X) = \kappa_i \prod_i X_i^{\alpha_i}$. La dérivée partielle est donnée par :

$$\frac{\partial \Phi}{\partial X_i} = \kappa_i \alpha_i X_i^{\alpha_i-1} \prod_{j \neq i} X_j^{\alpha_j}$$

qui est de signe constant.

Cinétique de Michaelis-Menten Elle décrit la transformation d'un substrat S en un produit P lorsque celle-ci est catalysée par une enzyme E . Elle est de la forme $\Phi(S, E) = \frac{E \cdot S}{k + S}$, où k est une constante positive. Les dérivées partielles sont :

$$\frac{\partial \Phi}{\partial E} = \frac{S}{k + S} \qquad \frac{\partial \Phi}{\partial S} = \frac{kE}{(k + S)^2}$$

qui sont de signe positif.

Dans la suite, on appellera *réaction monotone* une réaction dont la cinétique admet des dérivées partielles de signe constant.

3.2.7 Graphes de réactions

Nous venons de montrer que pour les cinétiques les plus courantes en modélisation, on peut associer à une réaction un graphe d'interaction dont les signes sont constants. Nous étudions maintenant le cas des systèmes comportant plusieurs réactions. Nous précisons les conditions permettant d'appliquer notre approche à l'étude des réseaux métaboliques et de signalisation, tels qu'on peut les trouver dans Kegg [48] ou BIOBASE [100]. De plus nous montrons qu'une transformation purement graphique permet de déduire le graphe d'interaction des descriptions fournies dans ces bases de données.

Les réseaux décrits dans les bases de données Kegg ou BIOBASE sont des réseaux de réactions biochimiques, c'est-à-dire des graphes bipartis spécifiant les substrats et les produits de chaque réaction. Nous en donnons la définition suivante :

Définition 2. (*Grappe de réactions*) Un graphe de réactions est un graphe orienté biparti $\mathcal{R} = (P, R, I, O)$ où P représente l'ensemble des produits, R l'ensemble des réactions, $I \subset P \times R$ les arcs d'entrée des réactions, $O \subset R \times P$, et tel que $\forall (i, r) \in P \times R \neg((i, r) \in I \wedge (r, i) \in O)$.

Un graphe de réactions est une description qualitative d'un système. S'agissant d'un ensemble de réactions biochimiques, on peut lui donner une description quantitative, à base d'équations différentielles. La définition qui suit explicite la compatibilité entre ces deux descriptions :

Définition 3. (*Système différentiel compatible avec un graphe de réactions*) On dit d'un système d'équations différentielles $\frac{dX}{dt} = F(X)$ qu'il est compatible avec un graphe de réactions $\mathcal{R} = (P, R, I, O)$ si il existe un ensemble de cinétiques monotones $(M_r)_{r \in R}$ telles que :

- $(i, r) \in I \Leftrightarrow \frac{\partial M_r}{\partial X_i} > 0$
- $(r, i) \in O \Leftrightarrow \frac{\partial M_r}{\partial X_i} = 0$
- pour tout $j \in P$,

$$F_i(X) = \sum_{(r,i) \in O} M_r(X) - \sum_{(i,r) \in I} M_r(X) - \gamma_i X_i$$

où γ_i est une constante de dégradation.

Dans cette définition, toutes les réactions sont considérées comme non réversibles. On représente donc les réactions réversibles par deux réactions distinctes.

Nous pouvons à présent formuler l'objectif de ce paragraphe : pour un graphe de réactions, nous avons défini l'ensemble des dynamiques qui lui sont compatibles. Chacune de ces dynamiques admet un graphe d'interaction. Nous formulons une condition purement graphique sur le graphe de réactions pour que toutes les dynamiques admettent le même graphe d'interaction. D'autre part nous montrons qu'on peut déduire ce graphe d'interaction à partir du graphe de réactions.

Cette transformation graphe de réactions/graphes d'interaction est donnée dans la définition suivante :

Définition 4. (*Grappe d'interaction associé à un graphe de réactions*) On appelle graphe d'interaction associé à un graphe de réactions $\mathcal{R} = (P, R, I, O)$ le graphe $\mathcal{G}_{\mathcal{R}}$ de sommets contenus dans P obtenu :

1. en plaçant un arc $i \xrightarrow{-} j$ chaque fois qu'on a $(i, r) \in I$, $(j, r) \in I$ et $i \neq j$,
2. en plaçant un arc $j \xrightarrow{+} i$ chaque fois qu'on a $(j, r) \in I$, $(r, i) \in O$ et $i \neq j$,
3. en remplaçant tout couple d'arcs $j \xrightarrow{+} i$, $j \xrightarrow{-} i$ par un unique arc $j \xrightarrow{?} i$.

Nous pouvons à présent formuler le critère qui garantit que cette construction aboutit à un graphe d'interaction où tous les signes sont définis.

Théorème 7. Soient un graphe de réactions $\mathcal{R} = (P, R, I, O)$ et :

$$\mathcal{S} = (\{p_1, p_2\}, \quad (3.26)$$

$$\{r_1, r_2\}, \quad (3.27)$$

$$\{(p_1, r_1), (p_1, r_2), (p_2, r_2)\}, \quad (3.28)$$

$$\{(r_1, p_2)\} \quad (3.29)$$

$\mathcal{G}_{\mathcal{R}}$ est sans signe ? si et seulement si \mathcal{R} ne contient aucun sous-graphe isomorphe à \mathcal{S} .

Démonstration.

Par contraposée, supposons \mathcal{R} contient un sous-graphe \mathcal{T} isomorphe à \mathcal{S} . On renomme les sommets de \mathcal{T} comme ceux de \mathcal{S} . Lors de la construction de $\mathcal{G}_{\mathcal{R}}$, on ajoute l'arc $p_1 \xrightarrow{-} p_2$ parce que $(p_1, r_2) \in I$ et $(p_2, r_2) \in I$; puis l'arc $p_1 \xrightarrow{+} p_2$ parce que $(p_1, r_1) \in I$ et $(r_1, p_2) \in O$; finalement les arcs $p_1 \xrightarrow{-} p_2$ et $p_1 \xrightarrow{+} p_2$ sont remplacés par $p_1 \xrightarrow{?} p_2$. Pour la réciproque, on procède encore par contraposée. Supposons que $\mathcal{G}_{\mathcal{R}}$ contient un arc $p \xrightarrow{?} q$. Alors pendant la construction de $\mathcal{G}_{\mathcal{R}}$, on a introduit successivement un arc $p \xrightarrow{+} q$ et un arc $p \xrightarrow{-} q$. De la première règle, on déduit qu'il existe une réaction r telle que $(p, r) \in I$ et $(r, q) \in O$; on en déduit également que $p \neq q$ (par définition des graphes de réaction). De la deuxième, on a qu'il existe une réaction s telle que $(p, s) \in I$ et $(q, s) \in I$. On a nécessairement $r \neq s$ puisque sinon on aurait $(q, s) \in I$ et $(s, q) \in O$. Par conséquent, le sous graphe de \mathcal{R} engendré par p, q, r, s est isomorphe à \mathcal{S} . \square

Le résultat suivant montre que le graphe d'interaction déduit du graphe de réactions est identique au graphe d'interaction des dynamiques compatibles avec le graphe de réactions.

Théorème 8. Soient \mathcal{R} un graphe de réactions, F une dynamique différentielle compatible, et \mathcal{G} son graphe d'interaction global. Alors

$$\mathcal{G}_{\mathcal{R}} \text{ sans signe ?} \Rightarrow \mathcal{G}_{\mathcal{R}} = \mathcal{G}$$

Démonstration. Supposons $\mathcal{G}_{\mathcal{R}}$ sans signe ?. Soit un sommet i , montrons qu'il a les mêmes prédécesseurs dans $\mathcal{G}_{\mathcal{R}}$ et dans \mathcal{G} . Soit j un autre sommet, on a :

$$\frac{\partial F_i}{\partial X_j} = \sum_{(r,i) \in O} \frac{\partial M_r}{\partial X_j} - \sum_{(i,r) \in I} \frac{\partial M_r}{\partial X_j} - \gamma_i \mathbf{1}_{i=j} \quad (3.30)$$

1^{er} cas : $i \neq j$ Supposons que les deux sommes de la formule ci-dessus sont à support non vide : on peut trouver deux réactions r et s telles que $(r, i) \in O$, $(i, s) \in I$, $\frac{\partial M_r}{\partial X_j} \neq 0$ et $\frac{\partial M_s}{\partial X_j} \neq 0$. On a alors $(r, i) \in O$, $(j, r) \in I$, $(j, s) \in I$ et $(i, s) \in I$. Nécessairement les réactions r et s sont distinctes sinon $(i, s) \in I$ et $(s, i) \in O$, ce qui contredit la définition des graphes de réactions. Le sous-graphe engendré par p, q, r, s est isomorphe à \mathcal{S} ce qui contredit l'hypothèse $\mathcal{G}_{\mathcal{R}}$ sans signe ?. Par conséquent, l'une des sommes au moins

est à support vide. Si les deux le sont, alors j n'est pas un prédécesseur de i dans \mathcal{G} ; de plus j n'est le substrat d'aucune réaction consommant ou produisant i . Le sommet j n'est pas non plus un prédécesseur de i dans $\mathcal{G}_{\mathcal{R}}$. Si l'une des sommes est à support non vide, alors j est un prédécesseur de i dans \mathcal{G} et le signe de $\frac{\partial F_i}{\partial X_j}$ est déterminé. L'arc et le signe sont également trouvés dans $\mathcal{G}_{\mathcal{R}}$, en employant l'une des deux premières règles de la définition 4.

2^e cas : $i = j$ Supposons qu'il existe une réaction $r \in R$ telle que $(r, i) \in O$. Par définition des dynamiques compatibles avec le graphe de réaction, $\frac{\partial M_r}{\partial X_i}$ est nul. Donc la première somme dans l'équation 3.30 est à support vide. Par conséquent, on obtient bien $\frac{\partial F_i}{\partial X_i} < 0$. \square

Ce théorème montre que les graphes de réactions peuvent être convertis en graphe d'interaction, de manière cohérente avec toute dynamique raisonnable. Le point important ici, c'est que à une condition près, le graphe d'interaction trouvé admet des signes définis. De plus cette condition est vérifiable par un simple parcours du graphe de réactions.

3.3 Justification booléenne

Nous reproduisons dans cette section un résultat dû à Adrien Richard qui montre que la contrainte de consistance est également vérifiée dans le cadre des réseaux booléens synchrones. Soit un ensemble de gènes indexé par $\{1, \dots, n\}$. L'état d'activation des gènes est représenté par un vecteur booléen de $\{0, 1\}^n$, et l'évolution du système est donnée par une fonction $F : \{0, 1\}^n \rightarrow \{0, 1\}^n$. Une trajectoire du réseau booléen est une suite $(F^n(x))_{n \in \mathbb{N}}$ pour un état initial x donné. Un état x est dit stable si c'est un point fixe de F .

3.3.1 Graphe d'interaction

De manière analogue à ce que nous avons vu dans le cadre différentiel, on peut définir une dérivée discrète de F et un graphe d'interaction. Pour un état x , on notera \bar{x}^i le vecteur obtenu en inversant la i^e coordonnée de x . On a alors la définition suivante :

Définition 5. (*Graphe d'interaction*) Pour $x \in \{0, 1\}^n$, on appelle :

- j^e dérivée discrète partielle de F_i la fonction

$$F_{ji}(x) = \frac{F_i(x) - F(\bar{x}^j)}{x_j - (\bar{x}^j)_j}$$

- et graphe d'interaction en x de F le graphe sur l'ensemble de sommets $\{1, \dots, n\}$ où figure l'arc $j \xrightarrow{\varepsilon} i$ avec $\varepsilon \in \{+, -\}$ si $\text{sgn}(F_{ji}(x)) = \varepsilon$

Dans ce cadre, il existe aussi une formulation et des preuves des conjectures de Thomas, que l'on pourra trouver dans [74, 76]. Comme là encore, le graphe d'interaction

dépend de l'état du système, et on peut parler de graphe d'interaction local et global. Ceci nous conduira à ajouter explicitement une hypothèse, toutefois moins forte que dans le cas différentiel :

Hypothèse 5 (H1). *Le graphe d'interaction global de F est défini, c'est-à-dire que $r(j, i) = \sum_{x \in \{0,1\}^n} \text{sgn}(F_{ji}(x)) \in \{+, 0, -\}$.*

3.3.2 Déplacement d'équilibre

Le résultat suivant montre que les équations qualitatives sont encore vérifiées dans le cadre booléen synchrone, à une différence près.

Théorème 9 (Richard, 2007). *Soit F une dynamique de graphe d'interaction défini. Pour tous $x, y \in \{0, 1\}^n$ et tout $i \in \{1, \dots, n\}$,*

$$F_i(x) \neq F_i(y) \Rightarrow \text{sgn}(F_i(y) - F_i(x)) \approx \sum_{j=1}^n r(j, i) \text{sgn}(y_j - x_j)$$

Démonstration. La preuve se fait par induction sur la distance de Hamming entre x et y , définie par :

$$d(x, y) = \#\{ i \mid i \in \{1, \dots, n\}, x_i \neq y_i \}$$

Cas initial : $d(x, y) = 0$ implique $x = y$ et $F(x) = F(y)$. La propriété est donc vérifiée.

Hérédité : supposons la propriété vraie pour tout couple (x, y) tel que $d(x, y) \leq N$. Soit (x, y) tel que $d(x, y) = N + 1$ et supposons $F_i(x) \neq F_i(y)$. On pose :

$$\alpha = \sum_{j=1}^n r(j, i) \text{sgn}(y_j - x_j)$$

Il nous faut démontrer $\text{sgn}(F_i(y) - F_i(x)) \approx \alpha$. Si $\alpha = ?$, la propriété est vérifiée, sinon on peut trouver $k \in \{1, \dots, n\}$ tel que $x_k \neq y_k$, puisque $d(x, y) \geq 1$. On a alors l'égalité suivante :

$$\beta = \sum_{j=1}^n r(j, i) \text{sgn}(y_j - (\bar{x}^k)_j) = \sum_{j=1, j \neq k}^n r(j, i) \text{sgn}(y_j - x_j)$$

En posant

$$\gamma = r(k, i) \text{sgn}((\bar{x}^k)_k - x_k) = r(k, i) \text{sgn}(y_k - x_k)$$

on a $\alpha = \beta + \gamma$ (parce que ni α , ni β ni γ ne sont indéterminés. On distingue maintenant deux cas :

- $[F_i(\bar{x}^k) \neq F_i(x_k)]$ Alors $F_{ki}(x) \neq 0$, et comme F est de graphe d'interaction défini, on a $r(k, i) = F_{ki}(x)$. D'après la définition de F_{ki} on a donc :

$$F_i(\bar{x}^k) - F_i(x) = F_{ki}(x)(\bar{x}^k - x)$$

d'où

$$\text{sgn}(F_i(\bar{x}^k) - F_i(x)) = r(k, i) \text{sgn}(\bar{x}^k - x)$$

par passage aux signes, ce qui donne

$$\text{sgn}(F_i(\bar{x}^k) - F_i(x)) = \gamma$$

Comme $\alpha, \gamma \neq \mathbf{0}, ?$, on a $\alpha = \gamma$. De plus $F_i(x) \neq F_i(\bar{x}^k)$ et $F_i(x) \neq F_i(y)$ impliquent $F_i(y) = F_i(\bar{x}^k)$. On obtient ainsi $\text{sgn}(F_i(y) - F_i(x)) = \alpha$

- $[F_i(\bar{x}^k) \neq F_i(x_k)]$ Alors dans ce cas $F_i(y) \neq F_i(\bar{x}^k)$. Comme $d(y, \bar{x}^k) = d(y, x) - 1$, on a par hypothèse d'induction :

$$\text{sgn}(F_i(y) - F_i(\bar{x}^k)) \approx \beta$$

Comme $\beta, \alpha \neq ?$ et $\text{sgn}(F_i(y) - F_i(\bar{x}^k)) = \text{sgn}(F_i(y) - F_i(x^k))$, on obtient $\text{sgn}(F_i(y) - F_i(x^k)) = \alpha$

□

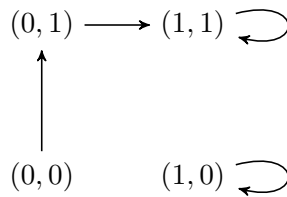
Notons bien que lorsqu'on a utilisé dans cette preuve des arguments de transitivité dans l'algèbre des signes, nous avons montré avant que les termes substitués étaient déterminés.

Théorème 10 (Déplacement d'équilibre dans le cas booléen). *Soient F une dynamique de graphe d'interaction défini, x et y deux états stables pour F . Alors pour tout $i \in \{1, \dots, n\}$,*

$$x_i \neq y_i \Rightarrow y_i - x_i \approx \sum_{j \rightarrow i} r(j, i) \text{sgn}(y_j - x_j)$$

Démonstration. Il suffit d'utiliser le théorème précédent en remarquant que $F_i(x) = x_i$, et que $j \rightarrow i$ si et seulement si $r(j, i) \neq \mathbf{0}$. □

Le résultat que nous obtenons dans le cas booléen est moins fort que dans le cas différentiel. Nous trouvons ici que si il y a variation *non nulle*, alors on doit pouvoir l'expliquer par la variation d'un au moins des prédécesseurs du sommet dans le graphe d'interaction global. Dans le cas d'une variation nulle, on ne peut pas conclure, comme le montre cet exemple :



En numérotant abscisses et ordonnées par 1 et 2 respectivement, on obtient

$$\begin{aligned}
 F_{12}(0,0) &= F_{12}(1,0) && = -1 \\
 F_{12}(0,1) &= F_{12}(1,1) && = 0 \\
 \\
 F_{21}(0,0) &= F_{21}(0,1) && = 1 \\
 F_{21}(1,0) &= F_{21}(1,1) && = 0
 \end{aligned}$$

Ce qui signifie que F admet un graphe d'interaction défini. Si l'on choisit $y = (1, 1)$ et $x = (1, 0)$, on obtient effectivement que la contrainte de consistance au sommet 1 n'est pas vérifiée : d'une part $y_1 - x_1 = 0$, et d'autre part $r(2, 1) = +$ et $y_2 - x_2 = 1$.

Le résultat obtenu dans le cas booléen asynchrone est très similaire à celui obtenu dans le cadre différentiel. Cette analogie vaut tout particulièrement pour l'hypothèse de stationnarité des états comparés. Il faut noter que le résultat obtenu est un peu moins fort dans le cas discret, puisque l'équation de consistance au sommet ne tient que pour les sommets dont la variation est non nulle. Cette restriction à la contrainte de consistance confirme ce qu'on peut intuitivement penser en pratique : il est difficile de décider quand une variation observée est négligeable ou nulle. Cette difficulté est mise en lumière dans le cadre booléen où les variables d'état sont discrétisées. La perte de précision empêche dans ce cas – contrairement au cas différentiel – de mesurer des variations trop faibles.

Bilan

Nous avons formalisé un critère de consistance entre un graphe d'interaction et des mesures expérimentales comparant deux états stables du système. Nous déduisons de ce critère des contraintes reliant le signe des régulations dans le graphe d'interaction aux signes de variation des espèces du système. Ces contraintes, appelées contraintes qualitative de consistance aux sommets, sont exprimées comme des termes interprétés dans l'algèbre des signes. En particulier, la résolution de ces contraintes est un problème NP-complet.

Afin de mieux saisir les limites d'applicabilité de ce critère de consistance, nous avons entrepris d'en démontrer la validité dans un cadre différentiel. Cette étude met en évidence des conditions sur la stationnarité des états comparés, et sur la monotonie des réactions constituant le système. Nous rapportons une démarche analogue dans le cadre des réseaux booléens synchrones, qui confirme ces résultats.

Nous passons à présent à l'étude des contraintes qualitatives. Cette étude inclut non seulement leur résolution, mais également le calcul de certaines propriétés de l'ensemble de leurs solutions. Dans tous les cas, ces problèmes sont au moins NP-complets ; les applications visées impliquant le traitement de grands volumes de données, il nous faut fournir des algorithmes particulièrement efficaces. Nous proposerons deux approches, décrites dans les deux prochains chapitres.

Chapitre 4

Résolution par diagrammes de décision

Nous avons défini au chapitre précédent une notion de consistance entre des données de perturbation et un modèle graphique des interactions cellulaires. Pour un graphe \mathcal{G} et un ensemble μ de mesures, nous avons introduit la contrainte qualitative $C_{\mathcal{G}}^{\mu}$ qui décrit la compatibilité entre le graphe d'interaction \mathcal{G} et les données de variation μ . Nous montrons à présent comment résoudre ces contraintes qualitatives. Une part importante de cette étude a été publiée dans [97]

Calculer l'ensemble des solutions d'une contrainte L'approche que nous suivons vise à calculer efficacement *toutes* les solutions d'une contrainte, plutôt qu'une seule. Ceci nous permettra notamment d'étudier des propriétés de l'*ensemble* des solutions. Cependant, le nombre de solutions aux contraintes de consistance est généralement très élevé, même pour des graphes d'interactions de dimension modeste. Notre approche repose pour cette raison sur l'utilisation d'un *diagramme de décision*, qui est une structure de données utilisée en vérification de circuit et en *model checking*. Les diagrammes de décision nous fourniront une représentation compacte de l'ensemble des solutions d'une contrainte. Ils nous permettront en outre par un parcours approprié de calculer des propriétés diverses de l'ensemble des solutions.

Définition des tâches d'analyse Une fois l'utilisation des diagrammes de décision précisée, nous introduisons et formulons précisément les problèmes liés à notre démarche d'analyse (voir figure 1.1). Nous aborderons successivement les tâches de vérification, de prédiction et de correction/diagnostic ; nous montrerons comment résoudre efficacement chaque problème par un parcours approprié du diagramme de décision.

Passage à l'échelle Les algorithmes que nous introduisons dans un premier temps supposent que l'on peut construire intégralement le diagramme de décision associé à une contrainte qualitative. Dans les applications que nous visons – notamment l'étude de réseaux transcriptionnels étendus – cela n'est pas toujours possible à cause de la

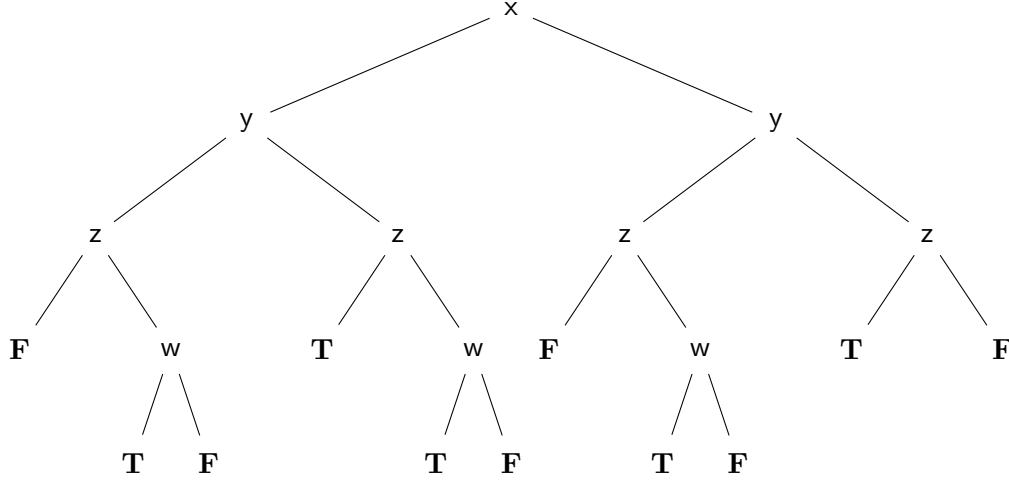


FIG. 4.1 – L'arbre de décision associé à la fonction booléenne $F(w, x, y, z) = ((y \otimes z) \vee (x \wedge w \wedge \neg z)) \wedge (w \vee \neg y)$, où \otimes désigne le ou exclusif et $x \prec y \prec z \prec w$.

taille du diagramme. Nous étudions à la fin de ce chapitre des approches de réduction et de décomposition de contraintes permettant d'analyser des données à l'échelle d'un organisme simple.

4.1 Diagrammes de décision

4.1.1 Définition

Un diagramme de décision est une structure de données permettant de représenter de manière compacte des fonctions booléennes. Nous l'introduisons ici en plusieurs étapes. On dispose d'un ensemble de variables (binaires) et d'un ensemble de constantes. Leur union est munie d'un ordre total \prec tel que pour toute constante c et toute variable V on a $V \prec c$.

Définition 6. On appelle arbre de décision un arbre binaire dont les sommets internes sont des variables, dont les feuilles sont des constantes, et tel que :

- tout chemin de la racine à une feuille est croissant pour \prec
- pour tout sous arbre $\mathcal{A} = (V, \mathcal{A}_0, \mathcal{A}_1)$ on a $\mathcal{A}_0 \neq \mathcal{A}_1$

À tout arbre de décision \mathcal{A} on peut associer une fonction booléenne $F_{\mathcal{A}}$ de la façon suivante. On note X_1, \dots, X_n les variables présentes dans \mathcal{A} , ordonnées selon \prec . Une valuation des variables X_i décrit un chemin de la racine à une feuille dans \mathcal{A} : c'est le chemin tel qu'en chaque noeud interne X_i de \mathcal{A} , on choisit la branche de gauche si $X_i = \mathbf{T}$ dans la valuation, et la branche de droite sinon. $F_{\mathcal{A}}$ est alors définie comme la fonction qui à toute valuation de X_1, \dots, X_n associe la constante au bout du chemin dans \mathcal{A} décrit par cette valuation.

La fonction $u : \mathcal{A} \mapsto F_{\mathcal{A}}$ est injective sur son image. Les fonctions qui ne sont pas dans l'image de u sont toujours de la forme $F(X, Y) = G(X)$ où G est dans l'image de u . Autrement dit, ce sont les fonctions définies avec des variables « inutiles ». Ce détail aura son importance lorsque nous voudrions définir le nombre de solutions d'une contrainte qualitative. Pour rendre u bijective on peut quotienter l'ensemble des fonctions booléennes avec la relation d'équivalence $F \sim G$ quand $F(X, Y) = G(X, Z)$ pour tous X, Y, Z . Dans la suite on confondra sauf mention contraire une fonction booléenne et sa classe d'équivalence selon \sim .

Ainsi pour une fonction F , l'arbre de décision associé est l'unique \mathcal{A} tel que $u(\mathcal{A}) \sim F$. On appelle *support* de F l'ensemble des variables apparaissant dans \mathcal{A} . Notons que le support de F ne dépend pas de \prec .

Un arbre de décision est une représentation simple d'une fonction booléenne mais de peu d'intérêt sur le plan algorithmique : sa taille, en nombre de noeuds croît en $O(2^n)$ où n est le cardinal du support de la fonction représentée. L'idée exploitée dans les *diagrammes* de décision est qu'un arbre de décision \mathcal{A} peut contenir plusieurs sous-arbres identiques. Auquel cas, on décide de ne le représenter qu'une seule fois en mémoire. Intuitivement, cela revient à fusionner les sous-arbres, et par conséquent à transformer l'arbre en graphe orienté acyclique. Donnons-en maintenant une définition plus formelle.

Définition 7. Soit une fonction booléenne F , et $\mathcal{A}_F = u^{-1}(F)$ l'arbre de décision associé. Le diagramme de décision \mathcal{D}_F de F est le graphe :

- dont les sommets sont les sous-arbres de \mathcal{A}_F ,
- et où il existe un arc $a \xrightarrow{\mathbf{T}} b$ (resp. $a \xrightarrow{\mathbf{F}} b$) si b est le fils gauche (resp. droit) de a dans \mathcal{A}_F .

\mathcal{D}_F est une représentation compacte de \mathcal{A}_F , c'est-à-dire où l'on a supprimé les redondances. Dans les cas favorables, la taille de \mathcal{D}_F est notablement plus faible que celle de \mathcal{A}_F . Le gain en taille dépend de l'ordre des variables ; cependant, déterminer l'ordre optimal est un problème NP-complet [13]. En pratique, les implémentations disposent d'heuristiques de réordonnement de variables. La transformation d'un arbre de décision en un diagramme est bijective. Il en découle qu'à une fonction booléenne correspond un unique diagramme de décision, et réciproquement.

4.1.2 Opérations sur les diagrammes

Nous présentons à présent les opérations couramment disponibles sur les diagrammes de décision. C'est l'occasion d'introduire les notations qui seront utilisées dans les algorithmes qui sont l'objet de ce chapitre, tout autant que de se familiariser avec la manipulation des diagrammes. L'ensemble des opérations sera présenté sous la forme d'un type abstrait, dans l'esprit de [29].

Constructeurs élémentaires Nous considérerons essentiellement trois types, à savoir *constant*, *variable*, et *diagram*. Le type *constant* représentera selon le contexte des

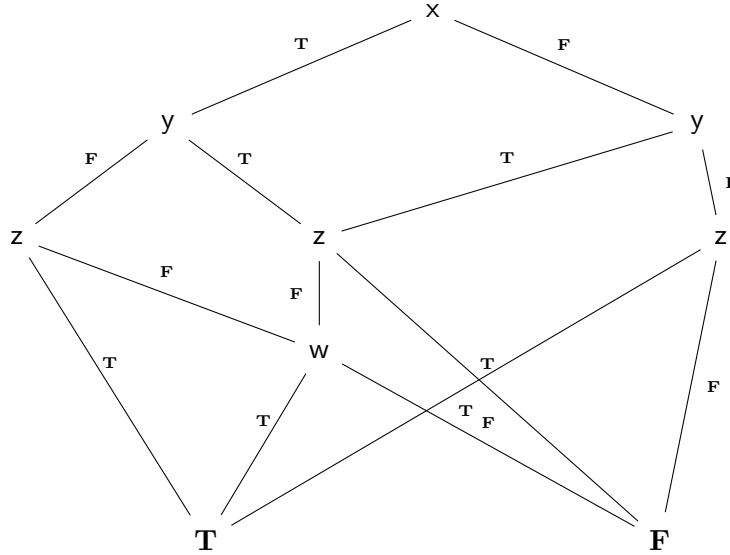


FIG. 4.2 – Diagramme de décision associé à l'arbre donné en figure 4.1. Notons le gain en nombre de sommets, passé de 21 à 9.

booléens ou des signes ; *variable* représente les symboles de variables qualitatives. L'appel $\text{Leaf}(c)$ construit le diagramme constitué d'une seule feuille qui est la constante c , qui représente la classe d'équivalence des fonctions à variables qualitatives constantes valant c . L'appel $\text{Node}(V, d_1, d_2)$ construit un diagramme qui se lit « si $V = +$, alors d_1 sinon ($V = -$) d_2 ». L'usage de la fonction Node est un peu délicat, parce qu'il faut l'appeler avec des paramètres qui respectent la définition des diagrammes de décision. Pour la première condition (monotonie des variables sur un chemin), il faut ajouter une précondition à l'appel, disant que la variable du nœud créé doit être plus petite selon \prec que les variables se trouvant à la racine des diagrammes fils. Le plus souvent, les diagrammes sont construits récursivement à partir d'autres diagrammes, ce qui aide à vérifier cet invariant. Pour la deuxième condition on opte en général pour une solution différente : l'appel $\text{Node}(V, d, d)$ retourne d .

Pour ces deux constructeurs, il est essentiel de maintenir l'invariant selon lequel un nœud donné n'a qu'une seule représentation en mémoire. Pour cela, on utilise des tables de hachage pour stocker les feuilles et les nœuds présents en mémoire centrale. Ces tables sont vérifiées avant de créer de nouveaux diagrammes. Nous utiliserons la notation $==$ pour parler d'égalité physique (c'est-à-dire le fait que deux objets ont une même représentation en mémoire). Rappelons que l'égalité physique implique l'égalité structurelle (*i.e.* l'égalité des valeurs représentées), mais que la réciproque est fautive en général. Les choses sont différentes dans le cas des diagrammes de décision, où une implémentation doit vérifier pour tous diagrammes d_1, d_2 , $d_1 = d_2 \Leftrightarrow d_1 == d_2$. Cet invariant est une conséquence de propriétés suivantes sur les constructeurs $c = c' \Rightarrow \text{Leaf}(c) == \text{Leaf}(c')$ et $d_1 == d'_1 \wedge d_2 == d'_2 \wedge x = x' \Rightarrow \text{node}(x, d_1, d_2) ==$

$\text{node}(x', d'_1, d'_2)$.

Visiteurs Nous aurons également besoin d'outils pour inspecter les diagrammes. La fonction `is_const` distingue les diagrammes qui sont des feuilles ; la fonction `root` fournit la variable racine d'un diagramme qui n'est pas une feuille ; les fonctions `dthen` et `delse` donnent les fils gauche et droit respectivement d'un diagramme qui n'est pas une feuille. On a par exemple l'invariant suivant : $\text{root}(\text{node}(x, d_1, d_2)) == x$. Ou encore $d_1 \neq d_2 \Rightarrow \text{delse}(\text{node}(x, d_1, d_2)) == d_2$

Opérations définie sur l'ensemble d'arrivée Pour définir des opérations sur les fonctions, on peut simplement se servir des opérations existant sur l'espace d'arrivée. Soit p un opérateur unaire sur les constantes, et soit F une fonction booléenne, sur les variables X_1, \dots, X_n (on les suppose ordonnées). Le calcul du diagramme de $p \circ F$ se base sur l'identité suivante :

$$p(F(X_1, \dots, X_n)) = \begin{cases} p(F(0, X_2, \dots, X_n)) & \text{si } X_1 = 0 \\ p(F(1, X_2, \dots, X_n)) & \text{sinon} \end{cases} \quad (4.1)$$

De là, on calcule $\mathcal{D}_{p \circ F}$ à partir de \mathcal{D}_F par l'algorithme suivant :

Function `op_unaire`(*op* : *constant* → *constant*, *d* : *diagram*)

```

if is_const(d) then
  | return leaf(op(to_const(d)))
else
  |  $f_0 \leftarrow \text{op\_unaire}(\text{op}, \text{dthen}(d))$ 
  |  $f_1 \leftarrow \text{op\_unaire}(\text{op}, \text{delse}(d))$ 
  | return node(root(d),  $f_0$ ,  $f_1$ )

```

Cet algorithme est remarquablement simple. Toutefois, la représentation compacte des diagrammes ne doit pas faire oublier que l'on travaille *in fine* sur un arbre binaire. Une conséquence directe est que l'algorithme présenté plus haut est en $O(2^n)$ si n est le nombre de variables dans le support de la fonction booléenne. En effet, même si chaque sous-arbre n'est *représenté* qu'une seule fois en mémoire, cet algorithme les visite autant de fois qu'ils apparaissent dans l'arbre de décision.

Pour s'en sortir, on a recours à des techniques de mise en cache des résultats. Cette amélioration est décisive en pratique, puisque dans le meilleur des cas, elle ramène le nombre de calcul en temps quasi-linéaire en la taille du diagramme (quadratique pour les opérations binaires, *cf infra*). Pour alléger les algorithmes, on ne mentionnera plus la gestion des caches dans les algorithmes. Profitons néanmoins de l'exemple des opérateurs unaires pour illustrer la démarche. Elle se transpose sans difficulté majeure dans les algorithmes à venir.

Function `op_unaire'` ($op : constant \rightarrow constant, d : diagram$)

```

if is_const( $d$ ) then
  return leaf(op(to_const( $d$ )))
else
  try return lookup(op,  $d$ )
  if échec then
     $f_0 \leftarrow op\_unaire'(op, dthen(d))$ 
     $f_1 \leftarrow op\_unaire'(op, delse(d))$ 
    store(op,  $d$ )
    return node(root( $d$ ),  $f_0$ ,  $f_1$ )

```

où `lookup` et `store` sont les opérations permettant de consulter et entrer une valeur dans le cache respectivement. Dans le même ordre d'idée, on peut construire des opérations sur les fonctions booléennes à l'aide d'opérations binaires sur les constantes. Soit une opération binaire \oplus , ainsi que deux fonctions F et G sur les variables X_1, \dots, X_n . Comme précédemment, on décompose par rapport à la plus petite variable.

- si F et G sont constantes et valent f et g respectivement, $(F \oplus G)(X_1, \dots, X_n) = f \oplus g$
- sinon, on a

$$(F \oplus G)(X_1, \dots, X_n) = \begin{cases} F(0, X_2, \dots, X_n) \oplus G(0, X_2, \dots, X_n) & \text{si } X_1 = 0 \\ F(1, X_2, \dots, X_n) \oplus G(1, X_2, \dots, X_n) & \text{sinon} \end{cases} \quad (4.2)$$

Évaluation Un diagramme représentant une fonction, on doit pouvoir évaluer cette fonction pour une valeur donnée de ses variables. Plus précisément, soit un diagramme D représentant une fonction f de variables x_1, \dots, x_n , et soit une substitution σ définie par $\sigma(x_i) = a_i$, on souhaite calculer $f(a_1, \dots, a_n)$. Il suffit pour cela de « descendre » dans le diagramme, en choisissant à chaque nœud x_i le sous-arbre correspondant à la valeur a_i .

Function `eval` ($d : diagram, \sigma : variable \rightarrow constant$)

```

Pre : support( $d$ )  $\subset$  dom( $\sigma$ )
if is_const( $d$ ) then
  return to_const( $d$ )
else
  if  $\sigma(\text{root}(d)) = \text{leaf}(+)$  then
    return eval(dthen( $d$ ))
  else
    return eval(delse( $d$ ))

```

Élimination de quantificateurs Terminons cette courte introduction par deux autres opérations qui seront utiles par la suite, à savoir l'élimination des quantifica-

teurs \exists et \forall . Formellement, on considère un prédicat $p(X, Y)$ – c’est-à-dire une fonction booléenne à valeurs dans \mathbb{B} – et il s’agit de trouver un nouveau prédicat q tel que $q(X)$ est vrai si et seulement si il existe une valeur pour Y telle que $p(X, Y)$. On notera $q(X) = \exists Y p(X, Y)$. Dans le cas fini, cette question a une réponse particulièrement simple :

$$q(X) = \bigvee_{y_1, \dots, y_k \in \mathbb{B}} p(X, y_1, \dots, y_k) \quad (4.3)$$

Néanmoins, un algorithme naïf qui s’aventurerait à calculer cette disjonction de 2^k termes serait bien peu utile. Une façon plus réaliste de procéder consisterait à éliminer successivement les variables :

$$\begin{aligned} & \exists Y_1 \dots \exists Y_k P(X, Y_1, \dots, Y_k) \\ &= \exists Y_1 \dots \exists Y_{k-1} P(X, Y_1, \dots, Y_{k-1}, 0) \vee P(X, Y_1, \dots, Y_{k-1}, 1) \\ &= \exists Y_1 \dots \exists Y_{k-1} P^{(1)}(X, Y_1, \dots, Y_{k-1}) \\ &= \exists Y_1 \dots \exists Y_{k-2} P^{(1)}(X, Y_1, \dots, Y_{k-2}, 0) \vee P^{(1)}(X, Y_1, \dots, Y_{k-2}, 1) \\ &= \exists Y_1 \dots \exists Y_{k-2} P^{(2)}(X, Y_1, \dots, Y_{k-2}) \\ &\dots \end{aligned}$$

En s’appuyant sur la représentation en diagrammes de décision, on peut proposer un algorithme qui effectue les k éliminations en un seul passage, comme dans la fonction `exists`. On obtiendrait l’élimination de \forall de la même manière en remplaçant dans l’algorithme le `ou` par un `et`. On voit sur cet exemple une façon de procéder que l’on appliquera régulièrement : grouper plusieurs opérations lors du parcours récursif du diagramme. Ici, on effectue les k éliminations en un seul parcours.

Function `exists`(Y : *variable set*, d : *diagram*)

```

if is_const( $d$ ) then
  return  $d$ 
else
   $f_0 \leftarrow$  exists( $Y$ , dthen( $d$ ))
   $f_1 \leftarrow$  exists( $Y$ , delse( $d$ ))
  if root( $d$ )  $\in Y$  then
    return op_binaire(or,  $f_0$ ,  $f_1$ )
  else
    return node(root( $d$ ),  $f_0$ ,  $f_1$ )

```

Synthèse Nous récapitulons les opérations (et les notations) introduites dans le type abstrait présenté à la page 52. La syntaxe suivie est un langage proche des signatures de modules dans *Objective Caml*.

Module BDD

```

type variable
type constant
type diagram

// Visiteurs
val is_const : diagram → bool
val to_const : diagram → constant
  [Pre : to_const (d) if is_const (d)]
val root : diagram → variable
  [Pre : root (d) if ¬ is_const (d)]
val dthen : diagram → diagram
  [Pre : dthen (d) if ¬ is_const (d)]
val delse : diagram → diagram
  [Pre : delse (d) if ¬ is_const (d)]

invariant d : diagram ⇒ ¬ (dthen (d) = delse (d))

// Constructeurs
val leaf : constant → diagram
  [Post : c = c' ⇒ leaf (c) == leaf (c')]
  [Post : to_const (leaf (c)) = c]
val node : variable → diagram → diagram → diagram
  [Pre : node (v,t,f) if is_const (t) ∨ v < root (t)]
  [Pre : node (v,t,f) if is_const (f) ∨ v < root (f)]
  [Post : root (node (v,t,f)) = v]
  [Post : node (v,t,f) == node (v,t,f)]
  [Post : dthen (node (v,t,f)) == t]
  [Post : delse (node (v,t,f)) == f]

// Opérations
val op_unaire : (constant → constant) → diagram → diagram
val op_binaire : (constante → constant → constant) → diagram → diagram →
diagram

// Élimination de quantificateurs
val exists : variable set → diagram → diagram
val forall : variable set → diagram → diagram

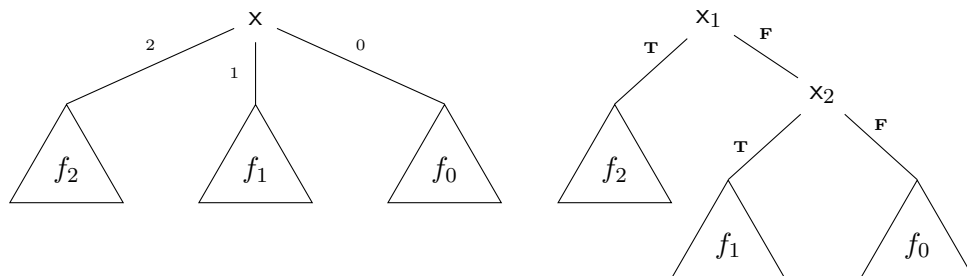
```

4.1.3 Fonctions à variables dans un ensemble fini

Il est tout à fait possible de généraliser les diagrammes de décision aux fonctions à variables dans un ensemble fini quelconque. La démarche est la même, sauf que l'on considère des arbres n -aires au lieu d'arbres binaires. Plus précisément si l'on dispose d'un ensemble V de variables prenant leurs valeurs dans un ensemble fini $E = \{e_1, \dots, e_n\}$, un arbre de décision $\mathcal{A} = (x, \mathcal{A}_1, \dots, \mathcal{A}_n)$ doit être tel que $\exists i, j \mathcal{A}_i \neq \mathcal{A}_j$.

Du point de vue implémentation, les choses sont nettement moins simples, et en pratique la plupart des bibliothèques disponibles ne proposent que les diagrammes pour fonctions booléennes. On peut donner deux raisons principales à cela :

- la condition *sine qua non* pour calculer avec les diagrammes de décisions, c'est que le diagramme tienne en mémoire principale. Une bonne implémentation doit donc veiller à optimiser la description du diagramme en mémoire. De ce point de vue, il est nettement plus simple de connaître à l'avance l'arité de l'arbre de décision.
- des variables pouvant prendre plus de deux valeurs peuvent toujours être codées en utilisant plusieurs variables binaires. Dans ce cas, on utilise plus de variables. Cependant une question (ouverte) est de savoir dans quel cas on fait apparaître le plus de redondances.
- on peut même simuler le comportement d'une implémentation d'arbre n -aire en jouant sur l'ordre d'apparition des variables. À une variable x sur un domaine fini à n éléments est associé un ensemble de $k = \lceil \log_2 n \rceil$ variables binaires x_1, \dots, x_k . De plus on impose que les variables x_i soient consécutives selon \prec , c'est-à-dire $x_i \prec y \prec x_{i+2} \Rightarrow y = x_{i+1}$. Enfin, on transforme tout arbre n -aire en un arbre binaire de façon à trouver le i^e sous-arbre du premier au bout du chemin dans le deuxième qui correspond au codage binaire de i . Voici un exemple pour $n = 3$.



Ce codage est légèrement plus volumineux qu'une implémentation directe des diagrammes de décision n -aires. Mais il présente l'avantage de reposer sur des implémentations très efficaces. Notamment on pourrait – mais cela reste à faire – répondre à la question évoquée au point précédent en comparant la taille des diagrammes quand on relâche la contrainte sur l'ordre des x_i .

Dans ce qui suit, nous représentons les contraintes qualitatives à l'aide de diagrammes de décisions. Les variables qualitatives ont un domaine à trois valeurs ($+$, $-$, et $\mathbf{0}$) et nous aurons donc besoin de diagrammes ternaires. Pour la réalisation des algorithmes nous nous sommes appuyés sur le logiciel Sigali [61], qui possède une implémentation dédiée des diagrammes ternaires. Nous avons supposé, mais pas vérifié,

que cette option était la plus appropriée pour limiter l'espace mémoire consommé par les diagrammes. Dans la suite, nous exposerons des algorithmes travaillant sur des arbres binaires, pour alléger leur présentation. À cette fin, nous limiterons le domaine des variables à $\{+, -\}$, excluant ainsi la valeur $\mathbf{0}$. Le développement qui précède montre une façon simple d'adapter nos propositions au cas de variables sur domaines finis¹.

4.2 Problème de vérification

4.2.1 Diagramme associé à une contrainte qualitative

Une contrainte qualitative peut être vue comme une fonction booléenne, à savoir la fonction indicatrice des solutions de la contrainte. Ce que nous proposons dans ce chapitre, c'est de représenter une contrainte qualitative sous la forme d'un diagramme de décision. Plus précisément, il s'agit d'une représentation en extension – mais compacte – de l'ensemble des solutions d'une contrainte qualitative ; nous verrons comment répondre à un grand nombre de questions sur l'ensemble des solutions par des parcours récursifs du diagramme. Nous commençons par préciser sa construction.

Comme déjà mentionné, la façon la plus simple de construire le diagramme de décision d'une fonction est de procéder de proche en proche, ou plus précisément en suivant la structure de l'expression définissant la fonction. Ici, il suffit de définir par induction le diagramme associé à une contrainte qualitative, comme montré dans la fonction `of_term`. Il nous reste maintenant à préciser la relation entre l'ensemble des solutions d'une contrainte C et le diagramme `of_term(C)`. Intuitivement, il suffit de suivre un chemin de la racine du diagramme à la feuille \mathbf{T} pour obtenir une solution. Sur ce chemin, il peut manquer certaines variables apparaissant dans C ; ces variables peuvent alors prendre n'importe quelle valeur.

Plus formellement, soient C une contrainte qualitative, et $\mathcal{D} = \text{of_term}(C)$. Un chemin π de \mathcal{D} est dit *chemin solution* s'il part de la racine et finit au sommet \mathbf{T} , c'est-à-dire s'il est de la forme $x_{i_1} \xrightarrow{s_{i_1}} x_{i_2} \xrightarrow{s_{i_2}} \dots x_{i_k} \xrightarrow{s_{i_k}} \mathbf{T}$, où s_{i_1}, \dots, s_{i_k} sont dans $\{+, -\}$. On associe à π une valuation v_π telle que $v_\pi(x_{i_r}) = s_{i_r}$. v_π est en général une valuation partielle de C , c'est-à-dire $\text{dom}(v_\pi) \subset \text{support}(C)$. On note $\text{val}_C(\pi)$ l'ensemble des valuations totales de C qui sont des prolongements de v_π .

Proposition 3. *Soit $S = \{\text{val}_C(\pi) \mid \pi \text{ chemin solution dans } \mathcal{D}\}$. S est une partition de l'ensemble des solutions de C .*

Démonstration.

1. les éléments de S sont non vides Soit π un chemin solution dans \mathcal{D} . $\text{val}_C(\pi)$ est au minimum de cardinal 1 si v_π est une valuation totale.

2. les éléments de S ont une intersection vide Soit π' un chemin solution distinct de π . π et π' admettent un préfixe commun de longueur maximale u contenant

¹Notre implémentation dédiée au cas des variables à trois valeurs conduit à des algorithmes similaires à ceux exposés dans cette thèse, mais un peu compliqués par la présence de trois fils à chaque nœud, au lieu de deux.

Function of_term($T : term$)

 match T with

 $\bigwedge_{i \in I} c_i$

 if $I = \emptyset$ then

 | return leaf(T)

else

 | Let $j \in I$

 | return op_binaire(and, of_term(c_j), of_term($\bigwedge_{i \in I \setminus \{j\}} c_i$))

 $\exists v c$

 return exists($\{v\}, c$)

 $\forall v c$

 return forall($\{v\}, c$)

 $p \otimes q$

 return op_binaire(\otimes , of_term(p), of_term(q))

 $\neg p$

 return op_unaire(\neg , of_term(p))

 v

 return node(v , leaf(+), leaf(-))

 $c \in \mathbb{S}$

 return leaf(c)

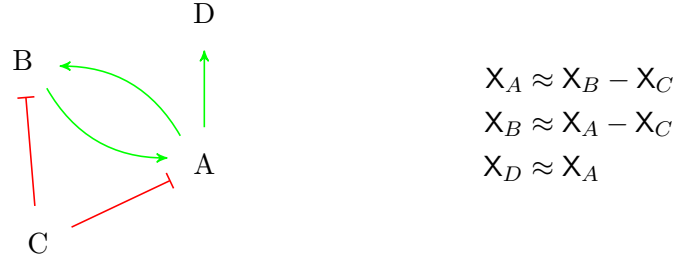


FIG. 4.3 – Exemple de graphe d'interaction et de la contrainte de consistance associée pour la construction des diagrammes.

X_A	$X_B \approx X_A - X_C$	$X_B - X_C$	$X_D \approx X_A$	$X_A \approx X_B - X_C$	$\begin{aligned} X_A &\approx X_B - X_C \\ X_B &\approx X_A - X_C \\ X_D &\approx X_A \end{aligned}$

TAB. 4.1 – Principales étapes de construction pour le graphe d'interaction donné plus haut.

au moins la racine de \mathcal{D} , et ne contenant pas \mathbf{T} (sinon π et π' sont identiques). π est de la forme $u \xrightarrow{s} v$ et π' de la forme $u \xrightarrow{t} v'$ avec $s \neq t$. Soit x le dernier sommet de u . Pour toutes valuations $(v, v') \in \text{val}_C(\pi) \times \text{val}_C(\pi')$, on a par conséquent $v(x) \neq v'(x)$.

3. L'union des éléments de S est l'ensemble des solutions de C Soit une solution de C . Alors cette valuation décrit un chemin (unique) dans \mathcal{D} qui se termine par la feuille \mathbf{T} . \square

Voyons cette construction à l'œuvre sur un exemple, avec le graphe d'interaction présenté en figure 4.3. Le sommet C est considéré par défaut comme une entrée parce qu'il n'a aucun prédécesseur. Le tableau 4.1 donne les principales phases de la construction de la contrainte. La variable X_C n'apparaît pas dans le diagramme final, qui admet deux chemins solution $\pi_1 = X_A \xrightarrow{+} X_B \xrightarrow{+} X_D \xrightarrow{+} \mathbf{T}$ et $\pi_2 = X_A \xrightarrow{-} X_B \xrightarrow{-} X_D \xrightarrow{-} \mathbf{T}$. Chaque chemin permet de construire deux solutions à la contrainte qualitative, une pour chaque valeur de X_C .

4.2.2 Algorithme pour la vérification

Les diagrammes de décision sont une représentation canonique des fonctions booléennes (et donc en particulier de nos contraintes qualitatives). Plus précisément, deux contraintes qualitatives sont équivalentes (décrivent le même ensemble de solutions) si elles ont même ensemble de variables libres et même diagramme². Nous utiliserons souvent un cas particulier important : quand un diagramme n'a aucun chemin de la racine vers **T**, alors il est nécessairement égal au diagramme contenant la seule feuille **F**. Cela nous permet de tester facilement l'existence d'une solution à une contrainte à partir de son diagramme.

Proposition 4. *Soit une contrainte qualitative C , et \mathcal{D} le diagramme de décision associé. C admet une solution si et seulement si \mathcal{D} n'est pas réduit à la feuille **F**.*

Nous avons à ce stade obtenu un algorithme pour le problème suivant :

Problème 1 (Vérification sous \mathcal{N} -consistance). *Soient un graphe d'interaction $\mathcal{G} = (V, E, \rho)$, et un ensemble μ de mesures. Déterminer si \mathcal{G} et μ sont \mathcal{N} -consistants.*

Il suffit de calculer la contrainte $C_{\mathcal{G}}^{\mu}$ introduite dans la définition 1, calculer son diagramme et vérifier qu'il est différent du diagramme **F**.

4.3 Problème de prédiction

4.3.1 Invariant de l'ensemble des modèles

Une contrainte qualitative décrit un ensemble de solutions. Il n'y a *a priori* aucune raison d'en privilégier une, mais on peut en revanche s'intéresser aux invariants de cet ensemble, c'est-à-dire aux propriétés vraies pour toute solution de la contrainte. Dit autrement, un invariant est une conséquence de la contrainte. Dans la suite, nous travaillerons uniquement sur des invariants simples, qui stipulent qu'une variable donnée prend la même valeur dans toutes les solutions de la contrainte.

Définition 8 (Invariant). *Soit une variable v et un signe s dans \mathbb{S}^* . Le couple (v, s) est un invariant d'une contrainte satisfiable C sur l'ensemble de variables X si $s' \neq s \Rightarrow \neg \exists X C[v := s']$.*

Il découle de la définition que toute solution v de C vérifie $v(v) = s$. La recherche des invariants d'une contrainte est un problème de prédiction, que nous étudions maintenant :

Problème 2 (Prédiction sous \mathcal{N} -consistance). *Déterminer tous les invariants d'une contrainte.*

²Notons que le test d'égalité des diagrammes se réduit même à un test d'égalité physique, puisque les implémentations assurent l'unicité de la représentation en mémoire.

Voici un premier algorithme, calqué sur la définition :

Function `invariant`⁽¹⁾($d : \text{diagram}$)

Pre : $d \neq \text{leaf}(\mathbf{F})$

$V \leftarrow \text{support}(d)$

for $x \in V$ **do**

| $E \leftarrow \{s \in S \mid \text{eval}(d, [x := s]) \neq \text{leaf}(\mathbf{F})\}$

| **if** $|E| > 1$ **then** $V \leftarrow V \setminus \{x\}$

return V

Pour chaque variable x , l'algorithme calcule les diagrammes de $C[x := s]$ pour s dans \mathbb{S}^* . Si le diagramme obtenu est la constante \mathbf{F} alors il n'est pas possible de trouver une solution de C telle que $x = s$. L'ensemble E contient donc les signes possibles pour x . Par conséquent :

- puisque C est satisfiable (précondition) E contient au moins un élément
- x peut former un invariant ssi E contient au plus un élément.

Cet algorithme est simple, mais un peu maladroit puisqu'il implique de calculer $|\mathbb{S}^*| \cdot |\text{support}(d)|$ diagrammes, à travers l'emploi de la fonction `eval`. Voyons à présent comment caractériser les invariants directement dans le diagramme de décision.

Proposition 5. *Soit C une contrainte, et \mathcal{D}_C son diagramme de décision. (x, s) est un invariant de C ssi :*

- x est dans le support de \mathcal{D}_C
- tout chemin de la racine à la feuille \mathbf{T} est de la forme $\dots x \xrightarrow{s} \dots$

La première condition implique que \mathcal{D}_C n'est pas une constante, et par conséquent que C admet au moins une solution. Cette propriété justifie l'algorithme récursif `values`.

Function `values`($x : \text{variable}, d : \text{diagram}$)

Pre : $d \neq \text{leaf}(\mathbf{F})$

case `is_const`(d) $\vee x \prec \text{root}(d)$

| **return** $\{\mathbf{T}, \mathbf{F}\}$

case $x = \text{root}(d)$

| $P_+ \leftarrow \text{if } d_{\text{then}}(d) = \text{leaf}(\mathbf{F}) \text{ then } \{-\} \text{ else } \emptyset$

| $P_- \leftarrow \text{if } d_{\text{false}}(d) = \text{leaf}(\mathbf{F}) \text{ then } \{+\} \text{ else } \emptyset$

| **return** $P_+ \cup P_-$

case $x \succ \text{root}(d)$

| **return** $\text{values}(d_{\text{then}}(d)) \cup \text{values}(d_{\text{false}}(d))$

À un diagramme \mathcal{D} représentant une contrainte C et une variable x , l'algorithme `values` associe l'ensemble des valeurs s telles que $C[x := s]$ admet une solution. Pour s'en convaincre, examinons les trois cas (disjoints et exhaustifs) envisagés dans l'algorithme :

1. si \mathcal{D} est une constante ou si la variable de tête de \mathcal{D} suit x pour l'ordre \prec , cela signifie que x n'apparaît pas dans \mathcal{D} , et n'est donc pas contrainte.

2. si la variable de tête de \mathcal{D} est x alors on vérifie l'existence de solutions pour $x = +$ et pour $x = -$. Par définition du diagramme, les nœuds fils de \mathcal{D} représentent les contraintes $C[x = +]$ et $C[x = -]$. On teste l'existence de solution de chacune de ces deux contraintes en les comparant au diagramme constant \mathbf{F} .
3. si la variable de tête w de \mathcal{D} n'est pas x , alors on utilise la relation suivante

$$\{s \mid C[x = s]\} = \{s \mid C[x = s, w = +]\} \cup \{s \mid C[x = s, w = -]\}$$

On peut en fait calculer en un seul parcours du diagramme *tous* les invariants, comme montré dans la fonction `invariant`⁽²⁾. Cette fonction retourne un ensemble de couples variable/valeur qui sont des invariants du diagramme considéré. Le cas terminal de la récursion dit qu'une constante n'admet aucun invariant. Si le diagramme considéré est un nœud, alors soit un seul de ses fils mène à la feuille \mathbf{T} – et alors on a trouvé un invariant, que l'on ajoute à ceux que l'on trouve récursivement ; soit les deux fils peuvent conduire à la feuille \mathbf{T} . Dans ce cas la variable correspondant au nœud n'est pas un invariant, et on cherche récursivement les invariants dans chaque branche. On utilise enfin le fait que si un couple n'est pas présent dans les deux résultats, alors il n'est pas un invariant dans l'un des fils, ou alors la valeur de la variable est différente dans les deux fils.

Function `invariant`⁽²⁾ ($d : \text{diagram}$)

```

if is_const( $d$ ) then
  return  $\emptyset$ 
case dthen( $d$ ) = leaf( $\mathbf{F}$ )
  return  $\{(\text{root}(d), -)\} \cup \text{invariant}(\text{delse}(d))$ 
case delse( $d$ ) = leaf( $\mathbf{F}$ )
  return  $\{(\text{root}(d), +)\} \cup \text{invariant}(\text{dthen}(d))$ 
otherwise
   $I_+ \leftarrow \text{invariant}(\text{dthen}(d))$ 
   $I_- \leftarrow \text{invariant}(\text{delse}(d))$ 
  return  $I_+ \cap I_-$ 

```

4.3.2 Marginales

Définition Un invariant est une conséquence d'une contrainte donnée, et constitue pour cette raison une prédiction *bona fides* d'un modèle. On peut également s'intéresser à une notion moins forte, que nous introduisons ici. La contrainte de consistance C_G^μ définie au paragraphe 3.1.3 décrit l'ensemble des modèles admissibles. En supposant tous ces modèles équiprobables, on peut s'intéresser à la probabilité qu'une variable x prenne la valeur s , soit $\mathbb{P}[x = s \mid C_G^\mu]$. Cette probabilité peut être calculée en comptant les solutions de la contrainte.

Solutions de la contrainte et chemin dans le diagramme À première vue – mais à première vue seulement – on pourrait penser que compter les solutions d’une contrainte revient à compter les chemins de la racine du diagramme au sommet **T**. Cette approche simple est malheureusement fautive, puisque comme on l’a vu, on peut associer plusieurs solutions à chaque chemin de la racine à la feuille **T**. On peut le voir sur l’exemple présenté en figure 4.3 : le système qualitatif comporte 4 variables, mais le support du diagramme représentant les solutions de ce système n’en a plus que 3 ; la variable représentant la variation du sommet *C* n’est pas contrainte, et n’apparaît donc pas dans le diagramme. En conséquence, il y a deux chemins dans le diagramme qui mènent au sommet **T**, mais bien quatre solutions à la contrainte initiale.

Comme nous l’avons vu précédemment, un diagramme de décision peut être vu comme une représentation canonique pour une classe d’équivalence de fonctions booléennes. Ces fonctions diffèrent par leur ensemble de départ, vu comme un ensemble de variables. Pour désigner une fonction – ou une contrainte – booléenne, il faut donc donner à la fois son diagramme et l’ensemble de ses variables. C’est pour cette raison qu’on ne peut compter le nombre de solutions d’une contrainte à partir de son seul diagramme.

Valuations associées à un chemin Donnons-nous une contrainte *C*, son digramme de décision \mathcal{D}_C , et un chemin π de la racine au sommet **T**. Nous avons introduit au paragraphe 4.2.1 la valuation partielle v_π décrite par π , ainsi que l’ensemble $\text{val}_C(\pi)$ des solutions de *C* qui sont des prolongements de v_π . Soit *X* l’ensemble des variables libres de *C*. Pour construire un prolongement de v_π sur *X*, il suffit de choisir une valeur parmi deux pour chaque variable de *C* n’apparaissant pas dans π . On en déduit l’égalité suivante :

$$|\text{val}_C(\pi)| = 2^{|X| - |\pi| + 1} \quad (4.4)$$

Pour obtenir le nombre total de solutions de la contrainte (noté $\#C$), on somme $|\text{val}_C(\pi)|$ pour tous les chemins π de la racine à la feuille **T** :

$$\#C = \sum_{\pi} |\text{val}_C(\pi)| \quad (4.5)$$

Cette formule n’est pas encore très opérationnelle ; nous voyons à présent une formule de récurrence qui constitue la base d’un algorithme sur les diagrammes de décision.

Formulation récursive Il suffit de décomposer le terme par rapport à une variable libre. Soit *x* une variable libre dans *C*, on a :

$$\#C = \#C[x := +] + \#C[x := -] \quad (4.6)$$

Cette relation est utile pour le calcul récursif que nous proposons avec la fonction cardinal. Cette fonction reçoit en argument une représentation canonique d’une contrainte, *via* son diagramme et l’ensemble de ses variables libres. La précondition indiquée assure que le support du diagramme est bien contenu dans les variables libres de la contrainte. Trois cas sont envisagés : tout d’abord, si le diagramme est constant, alors le résultat

est donné par la formule 4.4. Dans le cas contraire, le diagramme a une variable de tête, que l'on va comparer à la variable libre de la contrainte la plus prioritaire. Si elles sont égales, on applique la formule 4.6. Si la contrainte admet une variable libre plus prioritaire que la racine du diagramme, cela signifie que la variable en question n'apparaît pas dans le diagramme, et que par conséquent les diagrammes de $C[x := +]$ et $C[x := -]$ sont identiques, d'où la récurrence employée. Dans les deux cas, la précondition pour l'appel récursif de la fonction cardinal est bien vérifiée. Enfin le cas où la variable à la racine du diagramme est strictement plus prioritaire que les variables libres du diagramme ne peut pas arriver, parce qu'il contredit la précondition.

Function `cardinal`($d : \text{diagram}$, $S : \text{variable set}$)

```

Pre : support( $d$ )  $\subseteq S$ 
if is_const( $d$ ) then
  | return  $2^{|S|}$ 
 $x \leftarrow \min S$  // Minimum selon  $\prec$ 
if root( $d$ ) =  $x$  then
  |  $n_1 \leftarrow \text{cardinal}(\text{dthen}(d), S \setminus \{x\})$ 
  |  $n_2 \leftarrow \text{cardinal}(\text{delse}(d), S \setminus \{x\})$ 
  | return  $n_1 + n_2$ 
if root( $s$ )  $\prec x$  then
  | return  $2 \times \text{cardinal}(d, S \setminus \{x\})$ 

```

Proportion de solutions Avec cet équipement, nous pouvons calculer la probabilité qu'une variable x prenne la valeur s sous la contrainte C . Elle s'exprime par :

$$\mathbb{P}[x = s | C] = \frac{\#C[x := s]}{\#C}$$

Comme nous l'avons vu, le nombre de solutions d'une contrainte dépend du nombre de ses variables libres et ce, même si certaines de ces variables peuvent prendre n'importe quelle valeur. De telles variables n'apparaissent pas dans le diagramme de la contrainte. Cela signifie que l'on peut augmenter « artificiellement » le nombre de solutions en ajoutant des termes dans une conjonction qui sont des tautologies et qui ont des variables libres (comme $x \approx ?$ par exemple).

Nous allons voir maintenant que cette propriété désagréable disparaît quand on considère non plus le nombre de solutions d'une contrainte, mais le rapport entre le nombre de solutions et le nombre total de valuations des variables de la contrainte. Cette *proportion de solutions* est invariante quand on ajoute des « variables inutiles » à la contrainte. Soit C une contrainte et X l'ensemble de ses variables libres. La *proportion* de solutions de C est définie comme :

$$\rho_C = \frac{\#C}{2^{|X|}}$$

La proposition suivante dit essentiellement que cette grandeur est indépendante de l'ensemble des variables libres de la contrainte.

Proposition 6. *La fonction $C \mapsto \rho_C$ est constante sur une classe d'équivalence selon \sim .*

Démonstration. Soit une contrainte booléenne C , et X l'ensemble de ses variables libres. Le couple (\mathcal{D}_C, X) où \mathcal{D}_C est le diagramme de décision de C désigne C de manière univoque. Nous avons en particulier $\text{support}(\mathcal{D}_C) \subseteq X$. Nous pouvons considérer 3 cas :

1. $\mathcal{D}_C = \mathbf{T}$. Toute valuation des variables de X est une solution donc $\rho_C = 1$
2. $\mathcal{D}_C = \mathbf{F}$. Aucune valuation des variables de X n'est une solution donc $\rho_C = 0$
3. Alors soit x la variable à la racine de \mathcal{D}_C . On a :

$$\begin{aligned}
 \rho_C &= \frac{\#C}{2^{|X|}} \\
 &= \frac{\#C[x := +] + \#C[x := -]}{2^{|X|}} \\
 &= \frac{1}{2} \left(\frac{\#C[x := +]}{2^{|X|-1}} + \frac{\#C[x := -]}{2^{|X|-1}} \right) \\
 &= \frac{1}{2} \left(\frac{\#C[x := +]}{2^{|X \setminus \{x\}|}} + \frac{\#C[x := -]}{2^{|X \setminus \{x\}|}} \right) \\
 &= \frac{\rho_{C[x:=+]} + \rho_{C[x:=-]}}{2}
 \end{aligned}$$

Dans chacun des cas, la proportion ne dépend pas de X , mais seulement du diagramme \mathcal{D}_C , qui est constant sur une classe d'équivalence selon \sim . \square

Calcul de la proportion de solutions On déduit de cette preuve un algorithme pour le calcul de la proportion, décrit dans la fonction `proportion`. La fonction procède récursivement sur le diagramme de décision en reprenant les trois cas de la preuve ci-dessus.

Function `proportion(d : diagram)`

```

case  $d = \text{leaf}(\mathbf{T})$ 
   $\sqsubseteq$  return 1
case  $d = \text{leaf}(\mathbf{F})$ 
   $\sqsubseteq$  return 0
otherwise
   $\left[ \begin{array}{l} p_+ \leftarrow \text{proportion}(\text{dthen}(d)) \\ p_- \leftarrow \text{proportion}(\text{delse}(d)) \\ \mathbf{return} \frac{p_+ + p_-}{2} \end{array} \right.$ 

```

Calcul des marginales On considère à nouveau une contrainte C , X l'ensemble de ses variables libres, et x une variable quelconque. On voit maintenant par un calcul rapide que la marginale $\mathbb{P}[x := s|C]$ est elle aussi constante sur une classe d'équivalence selon \sim :

$$\begin{aligned}\mathbb{P}[x = s|C] &= \frac{\#C[x := s]}{\#C} \\ &= \frac{\#C[x := s]}{2^{|X|}} \frac{2^{|X|}}{\#C} \\ &= \frac{2\rho_{C[x:=s]}}{\rho_C}\end{aligned}$$

Les marginales peuvent être calculées en construisant le diagramme de $C[x := +]$ pour chaque x dans X , et obtenir le résultat par la formule ci-dessus.

Notons pour finir un point important sur l'implémentation : le nombre de solutions d'une contrainte croît au pire en 2^n si n est le nombre de variables dans le support du diagramme. En pratique, le nombre de modèles satisfaisant la contrainte est souvent très élevé, et pour éviter les débordements il faut avoir recours à des entiers de taille arbitraire. Les proportions sont elles calculées sous forme de rationnels.

4.4 Diagnostic des contraintes non satisfiables

Jusqu'ici, nous n'avons considéré que le cas où la contrainte étudiée est satisfiable. Lorsque ce n'est pas le cas, on aimerait pouvoir circonscrire des raisons possibles pour ce problème. Si un graphe n'est pas compatible avec des observations, ce peut être pour trois raisons :

- signe erroné
- flèche manquante
- équation non applicable (voir partie différentiel)

Nous proposons une formulation générale pour définir ce type de problème et ce que nous appellerons un *diagnostic*.

Définition 9. Soit C une contrainte et σ une substitution, telles que $C[\sigma]$ n'admet aucune solution. On appelle correction une substitution σ' de même domaine, telle que $C[\sigma']$ admette une solution.

La distance entre deux mesures σ et σ' de même domaine est le nombre de leurs différences. Plus précisément,

$$d(\sigma, \sigma') = |\{x \in \text{dom}(\sigma) | \sigma(x) \neq \sigma'(x)\}|$$

Une correction est un diagnostic si $d(\sigma, \sigma')$ est minimale.

Voyons tout de suite comment cette définition se spécialise en différents problèmes pratiques.

4.4.1 Données bruitées

Les données de puces à ADN fournissent le rapport des concentrations/niveau d'expression entre deux conditions expérimentales. Quand le rapport est significativement différent de 1, l'interprétation en un signe de variation est relativement sûre. Néanmoins pour la plupart des gènes, la variation n'est pas significative, et peut mener à une interprétation incorrecte : un ratio légèrement supérieur à 1 doit-il être entré dans le modèle comme une variation positive, ou nulle, ou carrément rejetée ? On peut envisager deux stratégies :

- soit rejeter toutes les ratios en-dessous d'un certain seuil, au risque de perdre de l'information,
- soit garder toutes les données, quitte à obtenir une contrainte qualitative sans solution.

La deuxième alternative requiert de disposer d'un outil permettant d'identifier les données peu fiables. Voici une façon de procéder :

- construire la contrainte de consistance aux sommets C_G^θ
- construire la mesure μ_θ correspondant aux données expérimentales, avec l'interprétation suivante :

ratio pour x	→	$\mu(x)$
$r < -\theta$	→	–
$-\theta < r < \theta$	→	0
$\theta < r$	→	+

- dans le cas où $C[\mu_\theta]$ n'est pas satisfiable, déterminer l'ensemble des diagnostics
- chercher *dans l'ensemble des diagnostics* les invariants, ou calculer les marginales.

4.4.2 Reconstruction de réseau

Une problématique récurrente en biologie moléculaire consiste à déterminer les interactions moléculaires dans un système biologique donné, à partir de données de perturbation. La quantité de données disponible est en général très insuffisante pour suffire à cet objectif. Plus formellement, cela signifie que le problème inverse est mal posé, c'est-à-dire qu'il admet un grand nombre de solutions. On impose donc en général un critère de parcimonie, qui limite le nombre de solutions. Ce genre de tâche entre tout à fait dans notre problématique :

- on considère un graphe d'interaction \mathcal{G} complet, dont les arcs sont étiquetés avec un signe, éventuellement nul. Ainsi, les équations qualitatives sont de la forme :

$$X_{ik} \approx \sum_{j \in \mathcal{G}} S_{ji} X_{jk}$$

- on considère la mesure μ où les données expérimentales sont intégrées comme au-dessus, et où l'on ajoute $\mu(S_{ji}) = \mathbf{0}$.
- le problème de reconstruction consiste alors à trouver l'ensemble des μ' qui sont des diagnostics de μ , et à y chercher des invariants ou calculer les marginales

4.4.3 Recherche des sous-systèmes incompatibles

Une contrainte C est une conjonction de contraintes plus simples, notées C_{ik} . Chacune de ces contraintes C_{ik} est associée à un sommet i et une mesure k . Pour faciliter la lecture et la compréhension des inconsistances détectées, une possibilité consiste à isoler les contraintes qui en sont à l'origine ; et si possible d'en isoler le plus petit nombre possible. Là encore notre formulation permet d'aborder ce problème :

- à chaque contrainte C_{ik} , on associe une variable booléenne B_{ik} , et on considère les contraintes :

$$C'_{ik} = B_{ik} \vee C_{ik}[\mu_k]$$

et leur conjonction :

$$C' = \bigwedge_{i,k} C'_{ik}$$

- on se donne la mesure $\bar{\mu}$ telle que $\bar{\mu}(B_{ik}) = \mathbf{F}$
- si $C[\bar{\mu}]$ n'est pas satisfiable alors on calcule les diagnostics $\bar{\mu}'$

4.4.4 Calcul des diagnostics

Nous montrons à présent comment calculer *tous* les diagnostics d'une contrainte non consistante avec une mesure (voir la fonction `diagnostic`). Comme dans les sections précédentes, il s'agit d'un calcul récursif sur le diagramme représentant la contrainte. La difficulté ici repose sur le stockage des diagnostics trouvés, qui peuvent en effet être très nombreux. L'astuce consiste à stocker l'ensemble des diagnostics comme un diagramme de décision. La valeur calculée par la fonction `diagnostic` est une paire, comportant la distance de la mesure à ses diagnostics et l'ensemble des diagnostics, lui-même représenté par un diagramme.

Pour alléger la présentation, nous avons introduit une fonction `lin` qui est un constructeur de diagramme analogue à `node`, et qui reçoit trois arguments : une variable x , un diagramme \mathcal{D} et un signe s . La fonction `lin` construit le diagramme avec x à la racine, avec \mathcal{D} pour le fils correspondant au signe s , et `leaf(F)` pour l'autre fils.

Une limite importante de cet algorithme, c'est qu'il requiert de pouvoir construire la contrainte (c'est-à-dire la conjonction complète). Or celle-ci peut contenir un grand nombre de variables et ne pas tenir en mémoire centrale. C'est un sérieux handicap pour la reconstruction de graphe d'interaction : le nombre de variables de la contrainte croît évidemment en $O(n^2)$ si n est le nombre de gènes.

4.5 Réduction, décomposition des systèmes

Tous les algorithmes que nous avons proposés jusqu'à présent pour étudier une contrainte supposent de construire au préalable son diagramme de décision. C'est vrai aussi bien pour trouver les solutions, les prédictions et les diagnostics d'une contrainte donnée. Il s'agit là d'une faiblesse importante de cette approche, puisque si le nombre de variables augmente trop, le diagramme peut ne plus tenir en mémoire centrale.

Function $\text{diagnostic}(d : \text{diagram}, \mu : \text{substitution})$

```

if  $d = \text{leaf}(\mathbf{F})$  then return  $(\infty, d)$ 
if  $\text{dom}(\mu) = \emptyset$  then return  $(0, \text{leaf}(\mathbf{T}))$ 
 $x \leftarrow \text{mindom}(\mu)$  // minimum selon  $\prec$ 
if  $d = \text{leaf}(\mathbf{T})$  then
   $(\delta, m) \leftarrow \text{diagnostic}(d, \mu /_{\text{dom}(\mu) \setminus \{x\}})$ 
  return  $(0, \text{lin}(x, m, \mu(x)))$ 
if  $\text{root}(d) \prec x$  then
   $(\delta_+, m_+) \leftarrow \text{diagnostic}(\text{dthen}(d), \mu)$ 
   $(\delta_-, m_-) \leftarrow \text{diagnostic}(\text{delse}(d), \mu)$ 
   $\delta \leftarrow \min \{\delta_+, \delta_-\}$ 
  return  $(\delta, \bigvee_{s \in \{+, -\}, \delta_s = \delta} \text{lin}(\text{root}(d), m_s, s))$ 
if  $\text{root}(d) \succ x$  then
   $(\delta, m) \leftarrow \text{diagnostic}(d, \text{dthen}(\mu))$ 
  return  $\text{lin}(x, m, \mu(X))$ 
if  $\text{root}(d) = x$  then
   $(\delta_+, m_+) \leftarrow \text{diagnostic}(\text{dthen}(d), \mu /_{\text{dom}(\mu) \setminus \{x\}})$ 
   $(\delta_-, m_-) \leftarrow \text{diagnostic}(\text{delse}(d), \mu /_{\text{dom}(\mu) \setminus \{x\}})$ 
  if  $\mu(x) = +$  then  $\delta_- \leftarrow \delta_- + 1$  else  $\delta_+ \leftarrow \delta_+ + 1$ 
   $\delta \leftarrow \min \{\delta_+, \delta_-\}$ 
  return  $(\delta, \bigvee_{s \in \{+, -\}, \delta_s = \delta} \text{lin}(x, m_s, s))$ 

```

Nous proposons ici de déterminer si l'on peut répondre aux mêmes questions sans jamais calculer le diagramme complet. Pour cela, nous allons étudier les contraintes qui sont des conjonctions et qui sont telles qu'une variable donnée apparaît dans un petit nombre de contraintes.

Définition 10 (Graphe d'une conjonction). Soit $C = \bigwedge_{1 \leq i \leq n} C_i(X^{(i)})$, où $X^{(i)}$ l'ensemble des variables libres apparaissant dans la contrainte C_i . Le graphe de la conjonction C noté \mathcal{G}_C est le graphe biparti défini comme suit :

- les sommets de \mathcal{G}_C sont d'une part les contraintes C_i pour $1 \leq i \leq n$, et d'autre part les variables libres de C .
- pour chaque contrainte C_i et pour chaque variable $x \in X^{(i)}$, le graphe contient l'arête $\{x, C_i\}$

Nous utiliserons ce graphe pour étudier les systèmes obtenus en sélectionnant un sous-ensemble des contraintes de la conjonction. Pour obtenir le sous-graphe correspondant à un sous-ensemble de la conjonction, il suffit de construire le sous-graphe engendré par les sommets contrainte correspondant et tous leurs voisins directs. On parlera notamment de sous-graphe induit par un sous-ensemble de contraintes, que l'on notera $\langle E \rangle$.

4.5.1 Réduction préservant l'existence de solution

Dans ce paragraphe, nous exploitons la forme particulière des contraintes de consistance aux sommet.

Proposition 7. Soit une contrainte $C = \bigwedge_{1 \leq i \leq n} C_i(X^{(i)})$. Si \mathcal{G}_C contient un sommet contrainte C_k relié à un sommet variable v de degré 1, alors la contrainte $D = \bigwedge_{1 \leq i \leq n, i \neq k} C_i(X^{(i)})$ est telle que toute solution de D peut être étendue en une solution de C .

Démonstration. Soit une solution μ de D . La variable v n'apparaît que dans la contrainte C_k , donc il suffit de trouver une valeur de v rendant C_k satisfiable. C_k est une contrainte qualitative, de la forme $X_1 \approx X_2 + \dots + X_l$. Soit i tel que $X_i = v$.

1. si $i = 1$, alors $c = (X_2 + \dots + X_l)[\mu]$ est une constante. Si $c = ?$ toute valeur pour v convient. Sinon, il suffit de poser $v = c$.
2. si $i > 1$, il suffit de poser $v = \mu(X_1)$.

□

Proposition 8. Soient une contrainte $C = \bigwedge_{1 \leq i \leq n} C_i(X^{(i)})$ et \mathcal{G}_C le graphe de contraintes correspondant.

On considère l'application $t : C \mapsto \bigwedge_{i \in E \setminus F} C_i(X^{(i)})$, où $E = \{C_i \mid 1 \leq i \leq n\}$ et $F = \{C_k \mid \exists v \{C_k, v\} \in \mathcal{G}_C \wedge \deg(v) = 1\}$.

1. La suite $(t^n(C))_n$ admet une limite, appelée contrainte réduite.
2. Toute solution de la contrainte réduite peut être étendue en une solution de la contrainte originale.

Démonstration. La fonction t est décroissante au sens de l'inclusion, en identifiant les conjonctions à l'ensemble des contraintes dans la conjonction. La suite $(t^n(C))_n$ est donc elle aussi décroissante et elle est de plus minorée par \emptyset . Par conséquent elle admet une limite. □

Nous venons de définir une opération de *réduction du système*, telle que toute solution du système réduit peut être étendue pour construire une solution du système original. De plus, la preuve donne un algorithme effectif et simple pour cette construction.

4.5.2 Décomposition

Si la réduction décrite ci-dessus n'est pas suffisante, nous recourons maintenant à une approche de type « diviser pour régner ». Notons que d'autres formes de décomposition des réseaux biologiques (et notamment des graphes d'interaction) ont déjà été abordées [20], avec toutefois des motivations différentes. Soit une contrainte de la forme :

$$C(X, Y, Z) = C_1(X, Y) \wedge C_2(Y, Z)$$

où X et Z sont des ensembles de variables disjoints. Pour un calcul donné, on effectuera un traitement sur chaque partie de la conjonction, puis on combinera les résultats intermédiaires pour obtenir le résultat final. La récurrence dépend du calcul effectué, mais dans tous les cas, on évite ainsi de construire un diagramme sur les variables X , Y et Z , susceptible d'avoir une représentation en mémoire trop volumineuse.

Ce procédé est applicable récursivement, jusqu'à ce que les sous-systèmes contiennent un nombre suffisamment petit de variables. Pour simplifier la présentation, nous introduisons une représentation explicite de cette procédure.

Définition 11 (Décomposition). *Soit $C(X) = \bigwedge_{1 \leq i \leq n} C_i(X^{(i)})$, on appelle décomposition de C de grain θ un arbre binaire dont les sommets sont des sous-ensembles de $\{1, \dots, n\}$, tel que :*

- une feuille F est telle que la contrainte $\bigwedge_{i \in F} C_i(X^{(i)})$ contient au plus θ éléments.
- pour tout nœud N , ses fils N_0 et N_1 forment une partition de N .

Les feuilles d'une décomposition sont des sous-systèmes dont le diagramme est *a priori* de taille suffisamment petite. Le « suffisamment » dépend bien évidemment de l'implémentation, d'où l'existence du paramètre θ .

Notons dès à présent qu'une condition nécessaire et suffisante pour l'existence d'une décomposition est qu'aucune des contraintes C_i n'ait un support de cardinal supérieur à θ .

Remarquons, enfin, que la condition sur les feuilles est difficile à vérifier : pour connaître le support d'une contrainte, il faut en construire le diagramme, ce qui est un calcul coûteux. En pratique, on utilisera une borne supérieure simple à obtenir, à savoir le nombre de variables libres dans la contrainte.

Nous définissons à présent trois étiquetages d'une décomposition. Ces étiquetages décrivent les variables présentes dans le sous-système associé à chaque nœud. Soit C une contrainte et \mathcal{A} une décomposition de C . Pour tout sommet N de \mathcal{A} , on note $\mathcal{V}(N) = \bigcup_{i \in N} X^{(i)}$ l'ensemble des variables libres apparaissant dans au moins une des contraintes $C_i(X^{(i)})$ pour $i \in N$. Nous aurons aussi besoin de l'étiquette notée $\mathcal{X}(N)$ qui correspond à l'ensemble des variables « privées » de N : ce sont les variables qui apparaissent uniquement dans le sous-système correspondant à N . On définit enfin le complémentaire, noté $\mathcal{Y}(N)$ qui correspond aux variables du sous-système décrit par N qui sont partagées avec d'autres équations hors de N . Ces deux derniers étiquetages sont mutuellement récursifs, à partir de la racine de \mathcal{A} :

- Soit R la racine de \mathcal{A} , on pose :

$$\mathcal{X}(R) = \mathcal{V}(R)$$

- Soit un nœud N ayant deux fils N_0 et N_1 respectivement fils gauche et droit, on pose :

$$\begin{aligned} \mathcal{X}(N_0) &= \mathcal{V}(N_0) \setminus (\mathcal{V}(N_1) \cup \mathcal{Y}(N)) \\ \mathcal{X}(N_1) &= \mathcal{V}(N_1) \setminus (\mathcal{V}(N_0) \cup \mathcal{Y}(N)) \end{aligned}$$

- Pour tout sommet N de \mathcal{A} , on pose

$$\mathcal{Y}(N) = \mathcal{V}(N) \setminus \mathcal{X}(N)$$

Si un nœud M est dans la descendance d'un nœud N , alors le système décrit par M est un sous-système (au sens de l'inclusion de l'ensemble des contraintes) de N . Les étiquetages que nous avons introduits décrivent l'ensemble des variables présentes dans chaque sous-système de la décomposition. Des sommets situés sur des branches différentes de la décomposition ont une intersection vide, mais les sous-systèmes qu'ils représentent peuvent avoir des variables en commun. L'étiquetage \mathcal{V} donne l'ensemble des variables de chaque sous-système. L'étiquetage \mathcal{X} donne les variables d'un sous-système qui apparaissent uniquement dans sa branche ; l'étiquetage \mathcal{Y} est le complémentaire : les variables du sous-système qui apparaissent au moins une fois dans une autre branche de l'arbre. On s'autorisera à appliquer ces étiquetages à des arbres, ce qui par convention correspond à appliquer sur l'unique sommet de l'arbre s'il s'agit d'une feuille, ou sur la racine de l'arbre sinon.

4.5.3 Calcul de la consistance selon une décomposition

Nous montrons à présent comment décider de l'existence d'une solution à une conjonction, sans calculer le diagramme représentant la conjonction. La proposition suivante justifie l'emploi des décompositions introduites au-dessus.

Proposition 9. Soient $C(X) = \bigwedge_{1 \leq i \leq n} C_i(X^{(i)})$, et (A, B) une partition de X . Alors

$$\exists X C(X) \equiv \exists(A \cap B) \left(\exists(A \setminus B) \bigwedge_{i \in A} C_i(X^{(i)}) \right) \wedge \left(\exists(B \setminus A) \bigwedge_{i \in B} C_i(X^{(i)}) \right)$$

L'astuce proposée consiste à diviser la conjonction en deux parties (en profitant de l'associativité de \wedge), à éliminer les variables n'apparaissant que dans une seule des deux parties, et enfin à calculer la conjonction, avec (dans les cas favorables) un nombre réduit de variables. Dans la terminologie introduite au paragraphe précédent, cela donne : si dans la décomposition on a un nœud N possédant deux fils N_0 et N_1 , alors on peut calculer les diagrammes correspondant à N_0 et N_1 , éliminer les variables $\mathcal{X}(N_0)$ et $\mathcal{X}(N_1)$ dans chacun des diagrammes, calculer la conjonction, puis éliminer les variables dans $\mathcal{Y}(N)$. Cette relation est mise à profit dans l'algorithme décrit à la fonction consistency.

Function consistency(C : *diagram set*, \mathcal{A} : *decomposition*)

```

case  $\mathcal{A}$  is a leaf  $F$ 
  | return exists( $\mathcal{X}(F)$ , conjunction( $\{c_i \in C \mid i \in F\}$ ))
case  $\mathcal{A}$  is a node  $(N, \mathcal{A}_0, \mathcal{A}_1)$ 
  |  $r_0 \leftarrow$  consistency( $C, \mathcal{A}_0$ )
  |  $r_1 \leftarrow$  consistency( $C, \mathcal{A}_1$ )
  | return exists( $\mathcal{X}(N)$ , conjunction( $\{r_0, r_1\}$ ))

```

Soit $C = \bigwedge_{1 \leq i \leq n} C_i(X^{(i)})$ une contrainte et \mathcal{A} une décomposition de C . Un appel consistency(C, \mathcal{A}) calcule, pour chaque sommet S de \mathcal{A} , le diagramme de la contrainte :

$$\exists \mathcal{X}(S) \bigwedge_{i \in S} C_i(X^{(i)})$$

En particulier l'appel de la fonction retourne le diagramme de $\exists X C(X)$ qui est la constante **T** ou **F** selon que la contrainte a, ou non, une solution.

4.5.4 Calcul des invariants selon une décomposition

Nous montrons à présent comment calculer les invariants d'une contrainte sans calculer explicitement cette contrainte. Nous restons dans les mêmes conditions que précédemment : soit une contrainte $C = \bigwedge_{1 \leq i \leq n} C_i(X^{(i)})$ et \mathcal{A} une décomposition de C . Nous aurons notamment besoin de la fonction γ qui à un sommet S de \mathcal{A} associe le diagramme de $\exists \mathcal{X}(S) \bigwedge_{i \in S} C_i(X^{(i)})$. Il s'agit précisément des diagrammes calculés par l'algorithme précédent, et nous les supposons pré-calculés, et accessibles en temps constant.

L'idée que nous exploitons est la suivante : supposons que l'on partitionne l'ensemble des contraintes formant la conjonction en deux sous-systèmes A et B ; les variables libres de A se divisent en deux catégories, selon qu'elles sont également libres dans B ou non (c'est-à-dire si elles sont dans $\mathcal{X}(A)$ ou dans $\mathcal{Y}(A)$ respectivement). Pour calculer les invariants sur les variables « privées » de A , on calcule séparément les deux sous-systèmes, puis on élimine les variables « privées » de B dans le diagramme de B . On calcule ensuite de A et B (dans laquelle n'apparaissent plus les variables privées de B), sur laquelle on recherche les invariants qui concernent des variables de A . Cette procédure est justifiée par la proposition suivante.

Proposition 10. *Soit une contrainte $C(X, Y, Z) = C_1(X, Y) \wedge C_2(Y, Z)$ telle que X et Z sont des ensembles de variables disjoints. Soient*

$$\begin{aligned} C_X(X, Y) &= C_1(X, Y) \wedge (\exists Z C_2(Y, Z)) \\ C_Z(Y, Z) &= C_2(Y, Z) \wedge (\exists X C_1(X, Y)) \end{aligned}$$

On a alors :

$$\text{inv}(C(X, Y, Z)) = \text{inv}(C_X(X, Y)) \cup \text{inv}(C_Z(Y, Z))$$

Le même principe peut être appliqué récursivement, en suivant une décomposition. Cela est illustré dans la fonction `invariants*` qui est, à un détail près, une application directe de la formule ci-dessus.

Function `invariants*` (C : *diagram set*, \mathcal{A} : *decomposition*, R : *diagram*)

```

case  $\mathcal{A}$  is a leaf  $F$ 
  | return invariant(conjunction( $\{c_i \in C \mid i \in F\} \cup \{R\}$ ))
case  $\mathcal{A}$  is a node  $(N, \mathcal{A}_0, \mathcal{A}_1)$ 
  |  $R_0 \leftarrow$  exists( $\mathcal{V}(\mathcal{A}_1) \setminus \mathcal{V}(\mathcal{A}_0)$ , conjunction( $\{R, \gamma(\mathcal{A}_1)\}$ ))
  |  $R_1 \leftarrow$  exists( $\mathcal{V}(\mathcal{A}_0) \setminus \mathcal{V}(\mathcal{A}_1)$ , conjunction( $\{R, \gamma(\mathcal{A}_0)\}$ ))
  |  $I_0 \leftarrow$  invariants*( $C, \mathcal{A}_0, R_0$ )
  |  $I_1 \leftarrow$  invariants*( $C, \mathcal{A}_1, R_1$ )
  | return  $I_0 \cup I_1$ 

```

Le principe de l'algorithme est le suivant : soit un appel `invariants*`(C, \mathcal{A}, R). Intuitivement, l'ensemble C représente la conjonction complète, l'arbre \mathcal{A} le sous-système courant, et le diagramme R la contrainte exercée sur le sous-système courant par le reste de la conjonction. On note S le sous-système courant, qui est un ensemble de contraintes inclu dans C . Le résultat de cet appel est l'ensemble des invariants concernant les variables libres de S . Si on tombe sur une feuille, alors le diagramme de S a suffisamment peu de variables libres pour être calculé ; R correspond au diagramme de $C \setminus S$ où l'on a éliminé toutes les variables qui n'apparaissent pas dans S . Son support est donc inclu dans celui du diagramme de S . On peut par conséquent calculer leur conjonction, et donc les invariants. Si \mathcal{A} est un nœud, alors on calcule récursivement les invariants dans chacun des sous-systèmes de S (c'est-à-dire ceux qui correspondent aux fils de \mathcal{A}). Pour procéder à ces appels récursifs, il faut notamment calculer la conjonction des contraintes dans le reste du système. Là encore, il faut calculer les conjonctions et les éliminations de variables dans un ordre correct, et limitant autant que possible le nombre de variables libres dans le support des intermédiaires de calcul. On parvient ainsi à limiter le support de R au cardinal de $\mathcal{V}(\mathcal{A})$.

L'efficacité pratique des procédés de décomposition décrits jusqu'ici dépend clairement de la décomposition choisie. Pour clôturer ce chapitre nous montrons comment caractériser une « bonne » décomposition, et comment la calculer.

4.5.5 Choix de la décomposition

Les deux applications précédentes suggèrent deux critères pour évaluer une décomposition. En premier lieu, le but de ces décompositions est d'éviter la construction d'un diagramme comportant trop de variables. Pour cela, les deux sous-systèmes produits à chaque étape de la décomposition doivent avoir le minimum de variables en commun. Pour s'en convaincre, il suffit de regarder le cas limite : dans le meilleur des cas, les deux sous-systèmes ne partagent pas de variables et peuvent être traités indépendamment.

Deuxièmement, une bonne décomposition doit comporter le minimum de sommets, pour limiter le nombre d'opérations à effectuer. Cela implique d'arriver par partitions successives à des sous-systèmes de support « suffisamment petit » le plus rapidement possible. Une stratégie simple consiste à imposer qu'à chaque étape la partition choisie détermine deux sous-systèmes comportant à peu près le même nombre de variables.

En suivant à la lettre ces deux critères, on arrive au problème d'optimisation suivant : pour une décomposition \mathcal{A} donnée, soit k le nombre de ses sommets, et pour tout $S \in \mathcal{A}$, on définit

$$\alpha_S = \begin{cases} -\infty & \text{si } S \text{ est une feuille} \\ |\mathcal{Y}(S_0) \cap \mathcal{Y}(S_1)| & \text{si } S_0 \text{ et } S_1 \text{ sont les fils de } S \end{cases}$$

La décomposition recherchée est celle minimisant l'objectif suivant :

$$\text{obj} = k + \lambda \max_{S \in \mathcal{A}} \alpha_S$$

où λ est un paramètre permettant de contrôler l'importance relative des 2 critères. Il faut bien admettre que ce problème est un peu compliqué, et que par ailleurs, sa résolution exacte n'est pas à proprement parler cruciale.

Dans cet esprit, nous proposons un algorithme glouton pour déterminer une décomposition. Son principe est le suivant : si un système donné comporte trop de variables dans son support, alors on choisit une partition des contraintes, telle que les deux sous-systèmes ainsi formés aient le moins de variables en commun, et approximativement le même nombre de variables au total. Cette même procédure est appliquée récursivement, jusqu'à ce que le nombre de variables dans le support des sous-systèmes aux feuilles soit assez petit.

Chaque étape de décomposition en deux parties peut être vue comme un problème de graphe : soit $C(X) = \bigwedge_{1 \leq i \leq n} C_i(X^{(i)})$ une contrainte et $\mathcal{G} = (C, V, E)$ le graphe de conjonction associé, où C est l'ensemble des contraintes C_i , V est l'ensemble des variables apparaissant dans C et E l'ensemble des arêtes du graphe. \mathcal{G} est un graphe biparti contraintes – variables. Les voisins d'une contrainte dans \mathcal{G} sont les variables apparaissant dans la contrainte. Par extension, les voisins d'un ensemble de sommets contrainte sont exactement les variables apparaissant dans la conjonction de ces contraintes, ce que nous avons noté $\mathcal{V}(E)$ pour tout ensemble E de contraintes. Avec ces notations, donnons maintenant un énoncé formel du problème de partition posé à chaque étape de la décomposition.

Problème : DECOMP-STEP

Données : graphe de conjonction $\mathcal{G} = (C, V, E)$, réel ε
partition $\{A, B\}$ de C telle que

Solution :
$$\begin{cases} |\mathcal{V}(A)| \leq \frac{1+\varepsilon}{2}|\mathcal{V}| \\ |\mathcal{V}(B)| \leq \frac{1+\varepsilon}{2}|\mathcal{V}| \\ |\mathcal{V}(A) \cap \mathcal{V}(B)| \text{ est minimal} \end{cases}$$

Ce problème de partition peut être vu comme une variante d'un problème bien connu en théorie des graphes, connu sous le nom de *Minimum Bisection Problem*. Ce dernier consiste à déterminer une partition des sommets d'un graphe non orienté en deux parties de cardinal égal et minimisant la capacité des arêtes coupées par la partition. Sa généralisation aux partitions de taille k ((k, ε) *balanced partitioning* en anglais) est également bien étudiée. Il s'agit dans tous les cas de problèmes difficiles : *Minimum Bisection Problem* est prouvé NP-complet, et la meilleure approximation en temps polynomial est en $O(\log^2(n))$ où n est le nombre de sommets dans le graphe. Nous adaptons maintenant la preuve de NP-complétude à DECOMP-STEP, qui procède par réduction de MAX-CUT, défini comme suit :

Problème : MAX-CUT

Données : graphe non orienté $\mathcal{G} = (V, E)$
une partition $\{A, B\}$ de V telle que

Solution : $\{\{a, b\} \in E \mid a \in A \wedge b \in B\}$ est de cardinal minimale

Théorème 11. *DECOMP-STEP est NP-complet.*

Démonstration. Considérons pour commencer une variante de DECOMP-STEP, nommée DECOMP-STEP', où le nombre de variables partagées doit être maximisé au lieu d'être minimisé. DECOMP-STEP' peut être réduit en DECOMP-STEP (et réciproquement) par une transformation simple du graphe : plutôt que de relier une contrainte à ses variables libres, on relie une contrainte C aux variables qui n'apparaissent pas dans C .

Soit $\mathcal{G} = (V, E)$ un graphe non orienté. On construit le graphe biparti $\mathcal{H} = (V \cup W, E, E')$ où :

- W est un ensemble de cardinal $|V|$ disjoint de V
- E' est l'ensemble construit en ajoutant deux arêtes $\{v, \{v, v'\}\}$ et $\{v', \{v, v'\}\}$ pour toute arête $\{v, v'\} \in E$.

La solution de DECOMP-STEP' sur \mathcal{H} et $\varepsilon = 0$ fournit une solution à MAX-CUT sur \mathcal{G} , en retirant de A et B les sommets de W .

□

Il nous reste enfin à préciser l'algorithme que nous avons utilisé pour résoudre le problème DECOMP-STEP. Il existe un certain nombre d'algorithmes d'approximation pour le problème *Minimum Bisection Problem*, et ceux-ci peuvent être adaptés pour DECOMP-STEP. À cette possibilité, nous avons préféré l'utilisation de techniques de résolution de contraintes booléennes décrites au prochain chapitre. Notons pour finir que de tels problèmes de partitionnement de graphes biologiques vérifiant certaines propriétés et partageant un minimum de sommets ont déjà été introduits dans d'autres

contextes, notamment dans des études de modularité des systèmes différentiels [51], ou dans le cadre des systèmes différentiels monotones [20].

Bilan

Dans ce chapitre nous avons décrit une méthode de résolution des contraintes qualitatives, basée sur les diagrammes de décision. Nous avons tout d'abord montré comment calculer efficacement l'ensemble des solutions d'une contrainte donnée à partir d'opérations de composition des diagrammes. Le résultat est une structure de données représentant de manière compacte chacune des solutions ; le gain en espace est obtenu en exploitant les redondances trouvées entre les solutions. Dans un deuxième temps, nous avons mis à profit cette structure de données pour étudier l'ensemble des solutions. Ainsi on peut, par le biais de procédures récursives parcourant le diagramme en profondeur, calculer les invariants d'une contraintes, les probabilités marginales sous une contrainte de chaque variable, ou les corrections minimales à effectuer pour rendre une contrainte compatible avec une mesure.

Le prix à payer pour ces résultats, c'est le risque permanent de tomber sur une contrainte dont la représentation en mémoire sous forme de diagramme de décision est trop volumineuse. Ce problème nous a conduit à mettre au point des procédures de réduction et de décomposition des systèmes qualitatifs. Celles-ci sont applicables quand une contrainte est une conjonction de contraintes comportant un petit nombre de variables ; elles visent à produire le résultat demandé sur une contrainte sans jamais construire complètement son diagramme.

Ces approches sont des réponses appropriées, mais il faut bien avouer qu'elles compliquent un peu les calculs, et qu'elles ne sont pas toujours applicables. Notamment, dès que l'une des contraintes dans une conjonction comporte un grand nombre de variables, on ne peut plus trouver de décomposition adéquate. Ce cas particulier survient en pratique dans les problématiques de reconstruction de réseau comme décrites au paragraphe 4.4.2.

Nous avons pour cette raison proposé une seconde approche pour l'étude des contraintes qualitatives, qui repose sur des techniques de résolution de contraintes booléennes. La stratégie est cette fois totalement différente : pour une contrainte donnée, on cherche cette fois seulement à déterminer *une* solution. Le compromis recherché est de faciliter le traitement de données plus volumineuses, quitte à perdre certaines des possibilités offertes par les diagrammes de décision.

Chapitre 5

Résolution par Answer Set Programming

Nous revisitons dans ce chapitre les problèmes formulés plus haut à l'aide de techniques de résolution de contraintes booléennes. Plus précisément, il s'agit d'étudier les possibilités offertes par la *programmation par ensemble réponse* pour aborder les problèmes de vérification, prédiction, correction/diagnostic. Il s'agit d'une technique relativement récente, qui peut être vue comme l'intersection de deux axes de recherche :

1. la définition d'un langage et d'une sémantique pour les programmes logiques facilitant la modélisation de problèmes
2. la recherche d'un moteur de résolution efficace pour déterminer le ou les modèles d'un programme logique.

Cette combinaison permet d'appliquer la programmation par ensemble réponse pour résoudre des problèmes combinatoires éventuellement difficiles. La démarche consiste à écrire un programme logique dont les modèles sont exactement les solutions du problème considéré ; on s'appuie ensuite sur un solveur dédié pour trouver les modèles du programme logique. L'efficacité actuelle des solveurs disponibles rend cette approche tout à fait réaliste, même pour le traitement de données volumineuses.

Dans la suite, nous introduisons brièvement la programmation par ensemble réponse, puis nous montrons comment l'utiliser pour résoudre les problèmes de vérification, de prédiction et de diagnostic/correction.

5.1 Une introduction à la programmation par ensemble réponse

La programmation par ensemble réponse (Answer Set Programming, ASP) désigne une famille de langages de programmation logique. Cette famille est caractérisée par la sémantique commune utilisée, dites des modèles stables, que nous introduirons plus loin. Brièvement, un programme ASP décrit un ensemble d'atomes à l'aide de règles. Un atome est un terme, au sens habituel en programmation logique. L'ensemble d'atomes

décrit est appelé ensemble réponse (*answer set*) ou modèle du programme logique. Les règles stipulent essentiellement que si certains atomes sont dans l'ensemble réponse (corps positif d'une règle), et que d'autres atomes ne s'y trouvent pas (corps négatif d'une règle), alors un ou plusieurs atomes (tête de la règle) doivent se trouver dans l'ensemble réponse.

Un atome peut être vu comme un fait, et les règles comme des déductions permises pour déterminer de nouveaux faits. La sémantique des modèles stables assure 1. que l'ensemble réponse d'un programme en vérifie toutes les règles (autrement dit, que c'est un modèle du programme), 2. que tout atome dans l'ensemble réponse admet une preuve, sous la forme d'une suite finie de règles applicables.

Dans la suite, nous introduisons en détail les programmes *normaux* (*normal logic programs*), qui sont une variante de la programmation par ensemble réponse. Nous mentionnerons ensuite plusieurs extensions.

5.1.1 Syntaxe

Nous nous limitons pour le moment aux programmes logiques où tous les atomes sont des constantes d'un ensemble \mathcal{A} . Un programme logique Π est un ensemble de *règles* de la forme suivante :

$$\underbrace{h}_{\text{tête}} \leftarrow \underbrace{a_1, \dots, a_n}_{\text{corps positif}}, \underbrace{\text{not } b_1, \dots, \text{not } b_m}_{\text{corps négatif}}$$

avec $m, n \geq 0$, où $h, a_1, \dots, a_n, b_1, \dots, b_m \in \mathcal{A}$. L'opérateur *not* est appelé négation par défaut (*negation as failure*). On note $\text{head}(r)$ La tête h d'une règle r , $\text{body}(r) = \{a_1, \dots, a_n\}$ le corps de r , $\text{body}^+(r) = \{a_1, \dots, a_k\}$ le corps positif et $\text{body}^-(r) = \{a_{k+1}, \dots, a_n\}$. Voici des exemples quelconques de règles syntaxiquement correctes :

$$\begin{aligned} a &\leftarrow \\ p &\leftarrow \text{not } q \\ q &\leftarrow \text{not } p \\ d(3) &\leftarrow \\ e(b) &\leftarrow p \\ p &\leftarrow \text{not } p, d(X), \text{not } e(X) \end{aligned}$$

Les deux propriétés d'un ensemble réponse E que nous avons énoncées plus haut s'expriment de la manière suivante :

- pour toute règle r d'un programme logique Π , si $\text{body}^+(r) \subset E$ et $\text{body}^-(r) \cap E = \emptyset$ alors $\text{head}(r) \in E$
- pour tout $a \in E$, on peut trouver r dans Π telle que $\text{head}(r) = a$, $\text{body}^+(r) \subset E$ et $\text{body}^-(r) \cap E = \emptyset$.

Ces deux propriétés ne caractérisent pas complètement les ensembles réponses. Nous donnons à présent leur définition qui est basée sur la notion de *modèle stable* introduit par [32].

5.1.2 Sémantique des modèles stables

Cas des programmes définis Un programme logique Π est *défini* (*basic program*) s'il ne contient pas de négation par défaut, c'est-à-dire si :

$$\forall r \in \Pi \text{ body}^-(r) = \emptyset$$

Dans ce cas particulier, la définition de l'ensemble réponse est naturelle : un ensemble réponse est l'ensemble des déductions possibles en utilisant les règles du programme. Cet ensemble est unique et se construit facilement, comme nous allons le voir. Pour un programme défini Π , on note α_Π l'application définie par :

$$\alpha_\Pi : \begin{cases} 2^A \rightarrow 2^A \\ X \mapsto X \cup \text{head}(\{r \mid r \in \Pi, \text{body}^+(r) \subset X\}) \end{cases}$$

L'application α_Π calcule les conséquences d'un ensemble d'atomes selon Π et les ajoute à son argument. Dit autrement, pour un ensemble d'atomes X , on applique les règles ayant tous leurs prérequis dans X , et on ajoute les têtes de ces règles à X . Cette application est donc croissante au sens de l'inclusion.

Voyons ce qui arrive si on itère cette opération. Soit H l'ensemble des atomes se trouvant en tête d'une règle de Π , et soit $X \subset H$; on a alors $\alpha_\Pi(X) \subset H$. On en déduit que la suite $(\alpha_\Pi^n(\emptyset))_n$ est bornée (par H). Elle admet par conséquent une limite, que l'on note $Cn(\Pi)$. $Cn(\Pi)$ est l'ensemble des atomes que l'on peut déduire en utilisant un nombre fini de règles de Π . Plus formellement, $Cn(\Pi)$ est l'unique plus petit point fixe de α_Π et correspond donc au plus petit ensemble clos par les règles du programme Π . On dira que c'est l'*ensemble réponse* de Π dans le cas défini.

Réduit d'un programme par rapport à un ensemble d'atomes Passons maintenant au cas général : on appelle *réduit* d'un programme Π par rapport à un ensemble d'atomes X le programme

$$\Pi^X = \{\text{head}(r) \leftarrow \text{body}^+(r) \mid r \in \Pi, \text{body}^-(r) \cap X = \emptyset\}$$

Le passage au réduit transforme un programme logique en un programme défini, en supprimant :

- les prérequis négatifs des règles,
- et les règles qui ne sont pas applicables à cause de certains atomes présents dans X

Ensembles réponse d'un programme normal Puisque Π^X est un programme défini, on peut calculer son ensemble réponse $Cn(\Pi^X)$. On appelle *ensemble réponse* (ou *modèle stable*) d'un programme logique Π tout ensemble d'atomes X tel que $Cn(\Pi^X) = X$. Intuitivement, on peut comprendre cette définition de la manière suivante. Pour qu'un ensemble d'atomes X soit un ensemble réponse, ce doit être un modèle où tout atome admet une preuve, sous la forme d'une suite d'applications de règles. Pour savoir si X est un ensemble réponse, il faut donc commencer par supprimer toutes les règles

qui ne sont pas applicables sous X , à cause des négations par défaut. Dans les règles restantes, le corps négatif n'est donc pas utile. C'est ainsi le réduct de Π par X que l'on a calculé. Maintenant, si les conséquences de Π^X sont exactement X , cela signifie que tout atome de X a une preuve valide, et que rien de plus ne peut être prouvé en utilisant les règles applicables.

Allons jusqu'à totalement nous débarrasser du formalisme. L'idée derrière les ensembles réponse consiste à se donner un ensemble de faits (les atomes) et à en apprécier la cohérence. On le jugera cohérent si sous l'hypothèse que ces faits décrivent correctement une situation, chacun d'eux admet une preuve finie, non circulaire par applications successives de règles.

Voyons cette définition à l'œuvre sur un exemple. Pour le programme

$$\begin{aligned} p &\leftarrow \text{not } q \\ q &\leftarrow \text{not } p \end{aligned} \tag{5.1}$$

les différents possibilités sont résumées dans le tableau suivant :

X	Π^X	$Cn(\Pi^X)$
\emptyset	$p \leftarrow$ $q \leftarrow$	$\{p, q\}$
$\{p\}$	$p \leftarrow$	$\{p\}$
$\{q\}$	$q \leftarrow$	$\{q\}$
$\{p, q\}$	\emptyset	\emptyset

qui montrent que seuls $\{p\}$ et $\{q\}$ sont des ensembles réponses. En effet ce sont les seuls ensembles X tels que $Cn(\Pi^X) = X$. Ce programme exprime donc l'exclusion mutuelle des atomes p et q .

Considérons maintenant le programme

$$p \leftarrow \text{not } p$$

et les différentes possibilités

X	Π^X	$Cn(\Pi)$
\emptyset	$p \leftarrow$	$\{p\}$
$\{p\}$	\emptyset	\emptyset

Ce programme n'a pas d'ensemble réponse et nous y aurons recours plus tard pour augmenter le langage des programmes logiques. Après avoir vu la définition des ensembles réponse, et quelques exemples, il nous reste à les distinguer d'autres définitions de modèles *a priori* plus intuitives. Les modèles stables sont :

- des modèles minimaux (au sens de l'inclusion) : supposons que X et X' sont deux modèles stables tels que $X' \subset X$. Alors nécessairement $\Pi^X \subset \Pi^{X'}$, et par conséquent $Cn(\Pi^X) \subset Cn(\Pi^{X'})$ puisque Π^X et $\Pi^{X'}$ sont définis. Or comme $Cn(\Pi^X) = X$ et $Cn(\Pi^{X'}) = X'$, on a $X = X'$. En revanche, les modèles minimaux ne sont pas forcément stables : le programme $\{p \leftarrow \text{not } p\}$ admet un (unique) modèle minimal $\{p\}$, mais pas d'ensemble réponse, comme nous l'avons vu précédemment.

- des modèles minimaux où tous les atomes sont supportés par l'application d'une règle. Mais la réciproque est fautive, comme le montre l'exemple suivant :

$$\begin{aligned} p &\leftarrow q \\ q &\leftarrow p \\ r &\leftarrow \text{not } q \end{aligned} \tag{5.2}$$

$\{r\}$ et $\{p, q\}$ sont des modèles minimaux où tous les atomes sont supportés par l'application d'une règle, mais seul $\{r\}$ est un ensemble réponse. En effet, $\Pi^{\{p, q\}} = \{p \leftarrow q, q \leftarrow p\}$ et $Cn(\Pi^{\{p, q\}}) = \emptyset$. L'ensemble $\{p, q\}$ n'est donc pas stable ; intuitivement la raison en est qu'il n'est pas possible de trouver des preuves non-circulaires pour la présence de p et q .

Avant de passer à la suite, et aux extensions des programmes normaux, essayons de bien comprendre ce qui fait la difficulté ici. Les programmes définis constituent un fragment de logique classique, celui des clauses de Horn. Il est très facile de définir des modèles raisonnables de ce type de programme, et ces modèles ont deux propriétés importantes : premièrement les programmes définis admettent un unique modèle ; deuxièmement, il s'agit d'une sémantique *monotone* : si j'ajoute des règles au programme, le modèle ne peut qu'augmenter, au sens de l'inclusion. L'utilisation de la négation par défaut perturbe complètement ces propriétés : l'ajout de nouveaux faits ou de nouvelles règles peut diminuer le modèle (considérer par exemple les programmes $\{p \leftarrow \text{not } q\}$ et $\{p \leftarrow \text{not } q, q \leftarrow\}$). On parle dans ce cas de logique *non monotone*. Dès qu'un programme comporte des négations par défaut, il peut avoir plusieurs modèles minimaux, et la question revient à définir le, ou les « bons » modèles.

Pour finir, insistons bien sur le fait que la négation par défaut est très différente de la négation en logique classique. Si un modèle vérifie *not p*, cela signifie qu'on ne peut pas trouver de preuve de p sous ce modèle. Illustrons cette différence sur un « classique » de logique non-monotone, avec ce programme décrivant le protocole à observer avant de traverser une voie de chemin de fer :

$$\begin{aligned} \text{check} &\leftarrow \text{not } \neg\text{check} \\ \neg\text{check} &\leftarrow \text{not } \text{check} \\ \text{train} &\leftarrow \text{not } \neg\text{train}, \text{check} \\ \neg\text{train} &\leftarrow \text{not } \text{train}, \text{check} \\ \text{cross} &\leftarrow \text{not } \text{train} \end{aligned}$$

Les deux premières lignes stipulent que l'on peut ou non vérifier avant de traverser, selon que l'ensemble réponse contient *check* ou $\neg\text{check}$ (les deux premières règles assurent l'exclusion mutuelle, comme vu plus haut. Les deux lignes suivantes signifient que si l'on vérifie avant de traverser, on peut prouver la présence ou l'absence de train (selon le même mécanisme d'exclusion mutuelle). La dernière ligne donne la condition pour traverser la voie. Le problème de ce protocole est qu'il admet, entre autres, le modèle $\{\neg\text{check}, \text{cross}\}$, c'est-à-dire « traverser sans regarder ». En effet, si $\neg\text{check}$ est dans le modèle, alors *check* ne peut pas y être ; par conséquent ni *train*, ni $\neg\text{train}$ ne peuvent être dans le modèle. Dit plus simplement, si on ne vérifie pas, on ne peut prouver ni

la présence, ni l'absence de train. La dernière règle dit en substance : « traverser si on ne peut pas prouver la présence de train », au lieu de dire « traverser si l'on peut prouver l'absence de train ». Or il vaut mieux – on en conviendra – prouver *l'absence* de train avant de traverser. Pour cela, la règle $cross \leftarrow not\ train$ doit être corrigée en $cross \leftarrow \neg train$.

5.1.3 Variables

Le langage des atomes peut être enrichi pour faciliter la modélisation de problèmes. On se donne trois ensembles dénombrables et disjoints de symboles $C = \{c_1, c_2, \dots\}$, $V = \{v_1, v_2, \dots\}$ et $P = \{p_1, p_2, \dots\}$, qui sont respectivement les constantes, les variables et les prédicats. Un *atome* est un terme sur ces ensembles de symboles :

$$a ::= \underbrace{c_i}_{\text{constante}} \mid \underbrace{v_j}_{\text{variable}} \mid \underbrace{p_k(a_1, \dots, a_n)}_{\text{prédicat}}$$

L'utilisation des variables permet, comme en Prolog, de séparer un programme logique en un ensemble de règles génériques d'une part, et une base de faits d'autre part. Les termes quant à eux, permettent de représenter les relations. Illustrons tout de suite leur utilisation, avec le programme suivant :

$$\left\{ \begin{array}{l} t(X, Y) \leftarrow r(X, Y) \\ t(X, Z) \leftarrow t(X, Y), r(Y, Z) \\ \\ r(1, 2) \\ r(1, 3) \\ r(2, 5) \\ r(3, 4) \end{array} \right.$$

Dans ce programme, le prédicat r représente une relation, dont les éléments sont donnés dans la deuxième partie du programme. La première partie définit la fermeture transitive de la relation, de manière générique. L'unique ensemble réponse de ce programme est

$$\{r(1, 2), r(1, 3), r(2, 5), r(3, 4), t(1, 2), t(1, 3), t(2, 5), t(3, 4), t(1, 5), t(1, 4)\}$$

Comme en Prolog, on adoptera d'une part la notation $pred/k$ pour spécifier l'arité des prédicats, et d'autre part la convention selon laquelle seules les variables commencent par une majuscule. Dans le programme précédent, on a par exemple utilisé les prédicats $t/2$ et $r/2$, et parlé des atomes $t(X, Y)$. Pour définir la sémantique des programmes avec variables – comme d'ailleurs pour en trouver un modèle – on passe par une étape dite de *grounding*, qui instancie les variables avec des termes ne contenant plus que des constantes.

5.1.4 Contraintes d'intégrité

Certaines pratiques apparentées à de l'ingénierie logicielle peuvent aider à formuler ce que l'on a en tête. L'une de ces « méthodologies » consiste à séparer les règles génériques en deux parties, l'une dite de génération et l'autre de test. Les premières ont pour fonction de décrire un sur-ensemble des solutions, les secondes d'y sélectionner les solutions par des sortes de filtres, appelés contraintes d'intégrité.

Pour illustrer cette démarche, revenons sur le programme (5.1). Nous avons vu que ses ensembles réponse sont $\{p\}$ et $\{q\}$ et en avons déduit que ce programme assurait l'exclusion mutuelle de p et q . Ce n'est que partiellement vrai puisque si les deux règles sont utilisées dans un programme plus grand, on peut tout à fait produire p et q par d'autres moyens. Par exemple

$$\begin{array}{l} p \leftarrow \text{not } q \\ q \leftarrow \text{not } p \\ p \leftarrow r \\ q \leftarrow r \\ r \end{array}$$

admet $\{p, q, r\}$ comme unique ensemble réponse. Pour exprimer précisément l'exclusion mutuelle de deux atomes, on utilisera le programme :

$$\begin{array}{l} p \leftarrow \text{not } q \\ q \leftarrow \text{not } p \\ \leftarrow p, q \end{array}$$

où la troisième règle est une *contrainte d'intégrité* signifiant que si p et q sont dans l'ensemble alors ce n'est pas un ensemble réponse. Plus généralement les contraintes d'intégrité sont des règles de tête vide, soit de la forme :

$$\text{ci} ::= \leftarrow a, \dots, a, \text{not } a, \dots, \text{not } a$$

Les solutions vérifiant le prérequis d'une contrainte d'intégrité sont éliminées. Il n'est pas nécessaire d'adapter la sémantique définie plus haut pour intégrer les contraintes d'intégrité : un programme comportant des contraintes d'intégrité peut être transformé en un programme normal équivalent. On introduit à cet effet un symbole \perp représentant la valeur logique faux, qui ne peut pas être utilisé dans un programme logique avec contraintes d'intégrité. Toute contrainte d'intégrité

$$\leftarrow a_1, \dots, a_n, \text{not } b_1, \dots, \text{not } b_n$$

est transformée en :

$$\perp \leftarrow \text{not } \perp, a_1, \dots, a_n, \text{not } b_1, \dots, \text{not } b_n$$

Si le prérequis d'une contrainte d'intégrité est vérifié, alors il reste une règle de type $\perp \leftarrow \text{not } \perp$. où \perp ne peut être produit par aucune autre règle. Nous avons vu qu'alors le programme n'admet aucun ensemble réponse. Nous verrons dans la suite de nombreuses utilisations de ces contraintes d'intégrité.

5.1.5 Contraintes de cardinalité

Le programme (5.1) permet de définir les ensembles réponses contenant un et un seul atome parmi deux. Les contraintes de cardinalité sont une généralisation de cette construction. On écrira une expression de la forme $k \{a_1, \dots, a_n\} l$ pour désigner un sous-ensemble des a_i de cardinal compris entre k et l . Les contraintes de cardinalités peuvent apparaître aussi bien dans la tête d'une règle que dans le corps. Dans le premier cas, la contrainte doit être respectée dans l'ensemble réponse; dans le deuxième, elle constitue un prérequis à l'applicabilité de la règle. On pourra trouver la sémantique précise des programmes logiques avec contraintes de cardinalité dans [84].

Ainsi le programme (5.1) se code plus naturellement en $\{1 \{p, q\} 2.\}$. La syntaxe inclut également une notation en intention de l'ensemble des a_i . Voyons en exemple le programme suivant :

$$d(1..4) \\ 1 \{c(X) : d(X)\} 3$$

La notation $d(1..4)$ est un raccourci pour la règle $d(1) d(2) d(3) d(4)$; les ensembles réponse de ce programme sont les sous-ensembles de cardinal 1 à 3 de $\{c(1), c(2), c(3), c(4)\}$ ajoutés à $\{d(1), d(2), d(3), d(4)\}$.

Pour illustrer l'approche « générer puis tester » décrite au paragraphe précédent, intéressons à la résolution du problème de coloration de graphe. Soit un graphe non orienté \mathcal{G} , et un ensemble (fini) de couleurs; le problème de coloration revient à attribuer à chaque sommet du graphe une couleur de telle façon que deux sommets adjacents n'ont pas la même couleur. Il faut dans un premier temps coder les données du problème : on écrit pour ça une première partie du programme qui constitue une base de faits :

```
vertex(allemande)
vertex(espagnole)
vertex(française)
...
edge(allemande, française).
edge(allemande, suisse).
edge(française, italienne).
...
col(bleu)
col(rouge)
...
```

Vient ensuite une partie générique, c'est-à-dire exprimée à l'aide de variables, qui produit les solutions au problème, et spécifie les contraintes qu'elles doivent respecter :

```
génération : 1 {label(V, C) : col(C)} 1 ← vertex(V)
test :      ← label(V, C), label(W, C), edge(V, W)
```

Le prédicat $label/2$ représente l'association sommet/couleur. La première règle force la présence dans tout ensemble réponse d'un atome $label$ précisant la couleur de chaque

sommet. Cette règle « produit » des solutions. La deuxième règle est une contrainte d'intégrité, qui élimine une solution si deux sommets voisins dans le graphe ont la même couleur.

Nous retrouvons dans ce programme le découpage typique que nous avons décrit plus haut. Tout d'abord, la séparation entre une partie générique (avec des variables), et la base de faits (suite d'atomes). La partie générique est elle-même divisée entre une partie génération (première ligne) et une partie test (la deuxième ligne).

5.1.6 Optimisation

Nous l'avons vu à plusieurs reprises, un programme logique peut avoir plusieurs modèles stables. Il peut donc se révéler utile de spécifier une fonction objectif pour sélectionner davantage les différents modèles. On l'exprimera par une commande de la forme suivante :

$$\text{minimize } \{a_1 = w_1, \dots, a_n = w_n, \text{not } b_1 = w_{n+1}, \dots, \text{not } b_m = w_{n+m}\}$$

Le poids d'un ensemble réponse est la somme des poids des atomes le constituant. Utilisons tout de suite cette construction pour spécifier dans le programme précédent les colorations *minimales* d'un graphe :

$$\begin{aligned} 1 \{label(V, C) : col(C)\} 1 &\leftarrow vertex(V). \\ &\leftarrow label(V, C), label(W, C), edge(V, W), \\ &\quad vertex(V), vertex(W). \\ used(C) &\leftarrow col(C), vertex(V), label(V, C). \\ \\ \text{minimize } \{used(C) : col(C)\}. \end{aligned}$$

Nous n'avons reproduit ici que la partie générique. Comme précédemment, la première règle dit que pour chaque sommet, il faut choisir une couleur parmi celles définies par le prédicat *col/1*. La règle d'intégrité qui suit assure que deux sommets voisins dans le graphe ne peuvent avoir la même couleur. La troisième règle introduit le prédicat *used/1* qui représente l'ensemble des couleurs utilisées dans la coloration. Enfin la directive *minimize* assure que l'ensemble réponse fourni utilise un nombre minimal de couleurs pour rendre une coloration correcte.

5.1.7 Complexité et résolution

Déterminer un modèle stable d'un programme normal est un problème NP : pour vérifier une solution il suffit de calculer le réduit, puis le modèle minimal du réduit, toutes choses que l'on peut faire en temps polynomial. Par ailleurs, il est très facile (et c'est bien l'intérêt de la programmation par ensemble réponse, vu comme un langage) de réduire un problème NP-complet à la recherche d'un modèle stable. Nous avons déjà vu l'exemple de la coloration de graphe, et l'application développée dans ce chapitre

en sera un autre. Les extensions revues dans les paragraphes qui précèdent sont toutes NP-complètes.

Il existe un certain nombre de solveurs pour la recherche de modèles stables, en particulier `smodels` [91], `dlv` [56], `cmodels` [58] et `clasp` [31]. Dans tous les cas, ces solveurs utilisent des techniques proches de celles mises en œuvre dans les solveurs SAT, tels que `miniSAT` ou `zchaff`. Ainsi ils bénéficient des progrès considérables effectués depuis deux dizaines d'années dans ce domaine.

Sans entrer dans le détail de fonctionnement des solveurs, mentionnons néanmoins une différence importante entre les solveurs ASP et les solveurs SAT. Pour une classe assez grande de programmes logiques, il existe une transformation de ces programmes en une formule de logique propositionnelle telle que les modèles stables des programmes sont exactement les modèles de la formule. Les programmes en question sont dits « *tight* » dans la littérature, la transformation est appelée complétion de Clark. Cette propriété, démontrée par F. Fages [25] assure donc les programmes *tight* peuvent être résolus à l'aide d'un solveur SAT. Les programmes non *tight* sont les programmes possédant des dépendances circulaires positives. Nous en avons donné un exemple plus haut, avec le programme (5.2) : dans un modèle stable, chaque atome a une preuve sous la forme d'une suite finie de règles. Or les solveurs SAT ne procèdent pas à ce genre de vérification. Quelle incidence en pratique ? La principale conséquence, c'est que les programmes non *tight* sont nettement plus difficiles à exprimer et à résoudre ? en logique propositionnelle qu'en programmation par ensemble réponse.

Avant de revenir aux contraintes qualitatives, rappelons les caractéristiques majeures de la programmation par ensemble réponse :

1. un langage déclaratif pour la modélisation de problèmes de recherche, à savoir les programmes logiques, augmenté de constructions telles que les contraintes de cardinalité (génération de modèles), les contraintes d'intégrité (filtrage de modèles) et directives d'optimisation (modèles minimaux),
2. l'existence de solveurs performants, similaires aux ou basés sur les solveurs SAT, permettant de résoudre des problèmes combinatoires réputés durs.

Nous allons à présent proposer une formulation de la contrainte de consistance par un programme logique : les solutions de la contraintes seront données exactement par les ensembles réponse du programme.

5.2 Consistance aux sommets

Nous proposons ici un codage des solutions d'une contrainte qualitative comme ensembles réponse d'un programme logique. Suivant la méthodologie proposée plus haut, un tel programme sera constitué de trois parties : la première pour les données, la deuxième pour la génération des solutions, et la dernière pour le test des solutions.

5.2.1 Codage des données

Le graphe d'interaction est codé à l'aide de deux prédicats : $vertex/1$ pour les sommets et $edge/2$ pour les arcs. Comme toujours, un graphe d'interaction est étiqueté par des signes, que nous définissons ici comme les atomes p , n et z pour $+$, $-$ et $\mathbf{0}$ respectivement. La variation observée d'un sommet I dans la mesure K sera donnée par le prédicat $m/3$; un atome $m(heatshock, cro, n)$ dit donc que lors d'une expérience de choc thermique, l'expression du gène cro a diminué. Le signe des arcs est donné par le prédicat $r/3$: un atome $r(cro, crp, p)$ dit que le gène cro active le gène crp . Voici par exemple le résultat de ce codage sur l'exemple donné en figure 3.1(b).

```

sign(p)  sign(n)  sign(z)
measure( $\mu_1$ )

vertex("A")  vertex("B")  vertex("C")
vertex("D")  vertex("E")

m( $\mu_1$ , "B", n)  m( $\mu_1$ , "C", z)
m( $\mu_1$ , "D", p)  m( $\mu_1$ , "E", n)

edge("A", "B")  edge("A", "D")  edge("B", "A")
edge("A", "E")  edge("B", "C")  edge("C", "E")
edge("D", "B")  edge("D", "C")  edge("D", "E")

r("A", "B", p)  r("A", "D", n)  r("A", "E", n)
r("B", "A", n)  r("B", "C", p)  r("C", "E", n)
r("D", "C", p)  r("D", "E", p)

```

5.2.2 Génération des solutions

Les solutions que nous cherchons sont composées des variations aux sommets et des régulations portées par chaque interaction. Le signe des régulations est décrit par le prédicat $r/3$ que nous avons déjà introduit. La variation d'un sommet dans une mesure est donnée par le prédicat $x/3$: par exemple, l'atome $x(\mu, "araC", z)$ signifie que dans la mesure μ , l'expression du gène $araC$ n'a pas varié. La génération des solutions est donnée par :

$$\begin{aligned}
1 \{ x(K, I, S) : sign(S) \} 1 &\leftarrow vertex(I), measure(K). \\
1 \{ r(J, I, S) : sign(S) \} 1 &\leftarrow edge(J, I).
\end{aligned}$$

La première règle spécifie que pour chaque sommet et chaque mesure, un ensemble réponse doit spécifier une variation dans $\{+, -, \mathbf{0}\}$; de manière analogue la deuxième règle indique que chaque régulation porte un signe. Il ne reste plus qu'à ajouter les contraintes de consistance.

5.2.3 Test des solutions

Il faut en premier lieu faire correspondre les variations de chaque sommets (prédicats $x/3$) avec les variations observées (prédicats $m/3$). C'est assuré par les contraintes d'intégrité suivantes :

$$\leftarrow m(K, I, S), \text{not } x(K, I, S)$$

Pour la contrainte de consistance, nous procédons en deux étapes, en commençant par identifier les contributions de chaque signe non nul.

$$\begin{aligned} \text{contrib}(K, I, p) &\leftarrow r(J, I, S), x(K, J, S), S \neq z \\ \text{contrib}(K, I, n) &\leftarrow r(J, I, S), x(K, J, T), S \neq T, S \neq z, T \neq z \end{aligned}$$

Les prédicats $\text{contrib}/3$ sont des indicateurs de la présence de termes de chaque signe non nul dans la somme 3.2. Plus précisément, un ensemble réponse contient un atome $\text{contrib}(\mu, g, s)$ si, dans la solution trouvée, le membre droit de l'équation 3.2 associée à g et μ contient un terme de signe s non nul. Puis nous indiquons l'effet des contributions :

$$\begin{aligned} x(K, I, p) &\leftarrow \text{contrib}(K, I, p), \text{not } \text{contrib}(K, I, n), \text{vertex}(I) \\ x(K, I, n) &\leftarrow \text{contrib}(K, I, n), \text{not } \text{contrib}(K, I, p), \text{vertex}(I) \\ x(K, I, z) &\leftarrow \text{not } \text{contrib}(K, I, p), \text{not } \text{contrib}(K, I, n), \text{vertex}(I). \end{aligned}$$

S'il n'y a que des termes strictement positifs (resp. négatifs) dans la partie droite de l'équation 3.2, alors la première (resp. deuxième) règle implique une variation positive (resp. négative) qui exclut – par l'effet des contraintes de cardinalité sur $x/3$ – toute autre variation. S'il n'y a aucun terme non nul, alors la variation doit être nulle. Enfin s'il y a un terme positif et un terme négatif, alors la variation n'est pas contrainte.

Un ensemble réponse de ce programme fournit exactement une solution à la contrainte de consistance aux sommets. Réciproquement, toute solution permet de construire un ensemble réponse. Nous avons donc ici un deuxième algorithme pour le problème de vérification sous consistance aux sommets.

5.3 Prédiction

Comme nous l'avons montré, on peut faire correspondre l'ensemble des solutions d'une contrainte qualitative et l'ensemble des ensembles réponse d'un programme logique. Rechercher les invariants de l'ensemble des solutions revient donc à chercher les invariants dans l'ensemble des ensembles réponses, c'est-à-dire l'intersection de tous les ensembles réponse. Une première idée consiste à utiliser la capacité de certains solveurs ASP à énumérer tous les ensembles réponses d'un programme donné. Dans notre cas cette stratégie n'est pas adaptée, à cause du nombre de solutions généralement observé. Nous recourrons donc à une stratégie par contre-exemples : 1. calculer un ensemble-réponse, 2. pour chaque atome a dans cet ensemble réponse, calculer un nouvel ensemble réponse ne contenant pas a . Cette approche est détaillée dans l'algorithme 15.

Au chapitre précédent, nous avons introduit une deuxième notion de prédiction, avec le calcul des probabilités marginales pour chaque variable. À la base de ce calcul

Algorithm 15: Algorithme de calcul des invariants d'un programme logique

Input: un programme logique Π admettant au moins un modèle

Output: un ensemble I d'atomes, tel que pour tout modèle M de Π , $I \subset M$.

$R \leftarrow \text{TrouverModele}(\Pi)$

for $a \in R$ **do**

if $\Pi \cup \{\leftarrow \text{not } a\}$ *admet un modèle* M **then**

$R \leftarrow (R \setminus \{a\}) \cap M$

return R

se trouvait la possibilité de compter efficacement le nombre de solutions d'une contrainte qualitative. Comme nous venons de l'évoquer, les solveurs ne peuvent (dans le meilleur des cas) compter les modèles d'un programme qu'en les énumérant explicitement. Cette approche n'est pas praticable parce que le nombre de solutions croît en général très vite avec le nombre de variable de la contrainte.

5.4 Contrainte non satisfiable

La recherche de diagnostic, comme proposée en définition 9, se traduit ici de la manière suivante. Soit une contrainte qualitative C , et une mesure μ . Nous avons vu comment calculer un programme logique Π dont les ensembles réponses coïncident avec les solutions de C . Par ailleurs la mesure μ peut être traduite en un ensemble d'atomes M , où chaque atome précise l'association (variable,valeur). La contrainte C est alors compatible avec la mesure μ si l'on peut trouver un ensemble réponse X contenant M . Si ce n'est pas le cas, on cherchera un ensemble réponse affichant le minimum de divergences avec M .

Plus formellement, pour un programme logique Π et un ensemble d'atomes M , on cherche un ensemble réponse X de Π tel que $M \subset X$ et tel que le cardinal de $M \setminus X$ est minimal. Pour y arriver, il suffit d'ajouter au programme Π la directive suivante :

$$\text{minimize } \{\text{not } a_1, \dots, \text{not } a_n\}$$

avec $M = \{a_1, \dots, a_n\}$.

Bilan

Nous avons présenté une solution basée sur la programmation par ensemble réponse pour les différentes tâches d'analyse des données. Rappelons que contrairement à l'approche proposée au chapitre précédent, la démarche consiste ici à ne calculer qu'une seule solution à la contrainte étudiée. Il s'agissait donc de cerner les gains et les limites que cela implique pour l'analyse de données.

Pour la vérification, nous avons proposé un programme logique dont les modèles sont exactement les solutions de la contrainte de consistance aux sommets. En anticipant

sur les résultats du chapitre suivant, il est clair que le gain escompté est bien là : la vérification de consistance entre un graphe d'interaction de plusieurs milliers de gènes et quelques dizaines de mesures ne pose aucun problème.

Le même programme logique est utilisé pour la recherche des invariants, dans le cadre d'un algorithme simple de recherche de contre-exemples. On montre au passage que le calcul des invariants ne nécessite pas vraiment de calculer l'ensemble des solutions. À l'inverse, le calcul des probabilités marginales n'est pas réalisable (au moins simplement) avec ASP, parce que l'énumération (nécessaire, dans ce cas) des solutions est beaucoup trop longue en pratique.

Nous avons enfin montré comment traduire la recherche de diagnostic en une formulation générale sur les ensembles réponse. Celle-ci fait notamment appel aux directives d'optimisation disponibles sur la plupart des solveurs ASP.

Le chapitre qui suit porte sur la validation de notre approche avec des données réelles et/ou de volume réaliste ; du point de vue algorithmique, il s'agit de vérifier *le passage à l'échelle* des méthodes que nous avons proposées. Elles devront notamment permettre – c'est notre but initial – le traitement de données haut-débit telles que des mesures d'expression, où plusieurs milliers de variables sont mesurées simultanément.

Chapitre 6

Validation expérimentale

Les deux derniers chapitres ont détaillé plusieurs approches pour la résolution et l'étude des contraintes qualitatives. Le présent chapitre présente l'application des algorithmes décrits à des données réelles, avec essentiellement deux objectifs :

1. évaluer la pertinence de notre modélisation qualitative. Permet-elle de décrire correctement un système réel et d'apporter des prédictions non triviales ?
2. déterminer si les algorithmes passent à l'échelle, c'est-à-dire s'ils sont utilisables sur des données telles que rencontrées en pratique.

Cette validation expérimentale porte majoritairement sur les problèmes de vérification et prédiction, et de correction/diagnostic dans une moindre mesure. Elle comporte trois volets : la première étude porte sur un des organismes les mieux étudiés à l'heure actuelle, à savoir la bactérie *E. coli*, dont il s'agit de déterminer la réponse globale à un stress nutritionnel ; la deuxième partie s'intéresse à la reconstruction du réseau transcriptionnel de *S. cerevisiae*, qui est un problème très étudié [66, 103, 14, 57, 62]. Nous comparons notamment nos résultats avec ceux obtenus dans [103]. Cette comparaison est étendue sur le plan théorique dans la troisième partie.

6.1 Prédiction de la réponse à une perturbation

Nous avons cherché dans un premier temps à vérifier que la règle de consistance que nous avons proposée est effectivement observée dans les données réelles. Dans ce but nous nous sommes intéressé au réseau transcriptionnel de la bactérie *E. coli* et à sa réponse sous l'effet d'un stress nutritionnel. Notre démarche est la suivante : 1. construire un graphe d'interaction à partir des informations disponibles dans les bases de données publiques, 2. vérifier sa consistance avec des données issues de la littérature décrivant la réponse typique de la bactérie à un stress nutritionnel, 3. utiliser ces données pour prédire les variations dans le reste du réseau, 4. confronter ces prédictions à des mesures obtenues par puces à ADN.

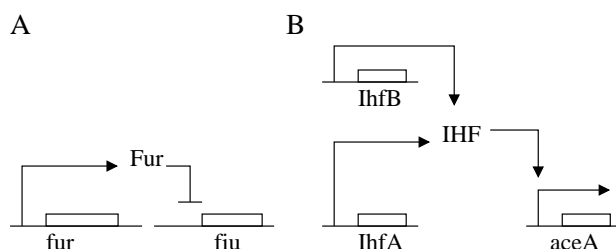


FIG. 6.1 – Exemples de régulations trouvées dans RegulonDB. (A) Inhibition du gène *fiu* par la protéine Fur (produite par le gène *fur*). On lui fait correspondre l’interaction $\text{fur} \xrightarrow{-} \text{fiu}$. (B) Production du dimère IHF par deux gènes *ihfA* et *ihfB* (un pour chaque sous-unité). Le complexe IHF active l’expression du gène *aceA*. On en déduit les interactions $\text{ihfA} \xrightarrow{+} \text{IHF}$, $\text{ihfB} \xrightarrow{+} \text{IHF}$ et $\text{IHF} \xrightarrow{+} \text{aceA}$.

6.1.1 Construction du graphe d’interaction

Nous avons utilisé la base de données RegulonDB [35] qui synthétise les informations disponibles sur les régulations transcriptionnelles de la bactérie *E. coli*. Elle propose notamment pour chaque gène, une liste des facteurs de transcription intervenant dans sa régulation, leur site de fixation et parfois quelques détails sur leur mécanisme d’action.

Pour construire notre graphe d’interaction, nous avons collecté les régulations contenues dans la table « *transcription factor to gene interactions* » de RegulonDB (version de mars 2006). Celle-ci se présente comme une liste d’interactions $A \xrightarrow{S} B$ où S est un signe (éventuellement indéterminé ou dépendant de l’état) et où A et B sont des gènes ou des protéines selon le cas :

- si une protéine A produite par un gène a est l’un des facteurs de transcription régulant l’expression du gène b , alors on crée l’interaction $a \rightarrow b$,
- si une protéine A est un complexe régulant l’expression d’un gène b , on crée l’interaction $A \rightarrow b$ (on trouve dans le réseau transcriptionnel de *E. coli* exactement 4 dimères, à savoir IHF, HU, RcsB et GatR),
- si un gène a produit un élément d’un complexe protéique B , on crée l’interaction $a \rightarrow B$.

Cette construction est illustrée à la figure 6.1. Les sommets sans prédécesseurs dans le graphe obtenu sont considérés comme des entrées du système. Nous avons de cette manière obtenu un graphe d’interaction contenant 1258 sommets et 2526 arcs, dont 160 portant un signe indéterminé. La structure du graphe est assez particulière, on notera notamment l’existence de 7 sommets (*crp*, *fnr*, IHF, *fis*, *arcA*, *narL* et *lrp*) ayant plus de 80 successeurs.

6.1.2 Confrontation aux données d’expression issues de la littérature, premier essai

Afin de valider notre réseau, nous avons voulu tester sa cohérence avec des données d’expression considérées comme particulièrement fiables. Nous avons pour cela utilisé

TAB. 6.1 – Variations pour 40 transcrits sous stress nutritionnel, comme fourni dans la base RegulonDB, version de mars 2006.

(a)		(b)		(c)		(d)		(e)	
gene	effect	gene	effect	gene	effect	gene	effect	gene	effect
acnA	+	csiE	+	gadC	+	osmB	+	recF	+
acrA	+	cspD	+	hmp	+	osmE	+	rob	+
adhE	+	dnaN	+	hns	+	osmY	+	sdaA	–
appB	+	dppA	+	hyaA	+	otsA	+	sohB	–
appC	+	fic	+	ihfA	–	otsB	+	treA	+
appY	+	gabP	+	ihfB	–	polA	+	yeiL	+
blc	+	gadA	+	lrp	+	proP	+	yfiD	+
bolA	+	gadB	+	mpl	+	proX	+	yihI	–

des informations également mises à disposition dans RegulonDB. Il s'agit de tables recensant les variations connues de certains gènes pour quelques conditions expérimentales. Chaque variation est fournie avec la ou les publications en apportant la preuve expérimentale. L'une des tables en particulier porte sur la réponse d'*E. coli* à un stress nutritionnel, et comporte 40 variations ; nous la reproduisons au tableau 6.1.

Il s'avère que ces 40 mesures ne sont pas consistantes avec le graphe d'interaction que nous avons construit. Pour en comprendre la raison, nous nous sommes appuyés sur la démarche proposée au paragraphe 4.4.3, qui consiste à isoler un sous-ensemble de contraintes dont la conjonction n'admet aucune solution. On appellera un tel sous-ensemble un *défaut* à la règle de consistance. Nous développons maintenant la recherche et l'analyse pratique de ces défauts.

6.1.3 Diagnostic par isolement des défauts

La contrainte de consistance que nous avons donnée à la définition 1 est une conjonction de contraintes locales, c'est-à-dire se rapportant à un sommet et ses prédécesseurs (et une mesure). Si une conjonction n'admet pas de solution, on peut toujours essayer de trouver des sous-ensembles de contraintes qui n'admettent pas non plus de solution. Dans le meilleur des cas, ces sous-ensembles sont suffisamment petits pour être facilement interprétables.

Pour faciliter cette interprétation, on peut utiliser le fait que chaque contrainte locale correspond à un sous-graphe du graphe d'interaction : il s'agit du sous-graphe engendré par le sommet attaché à la contrainte locale et ses prédécesseurs. Un sous-ensemble de contraintes locales correspond donc aussi à un sous-graphe du graphe d'interaction : le sous-graphe engendré par tous les sommets associés aux contraintes locales, et leurs prédécesseurs.

L'approche proposée au paragraphe 4.4.3 consiste à déterminer un sous-ensemble de contraintes locales dont la conjonction n'admet aucune solution, où le sous-ensemble en question est de *cardinal minimal*. Si le sous-ensemble est suffisamment petit, on peut donc en tirer une représentation simple, sous forme d'un graphe et des mesures

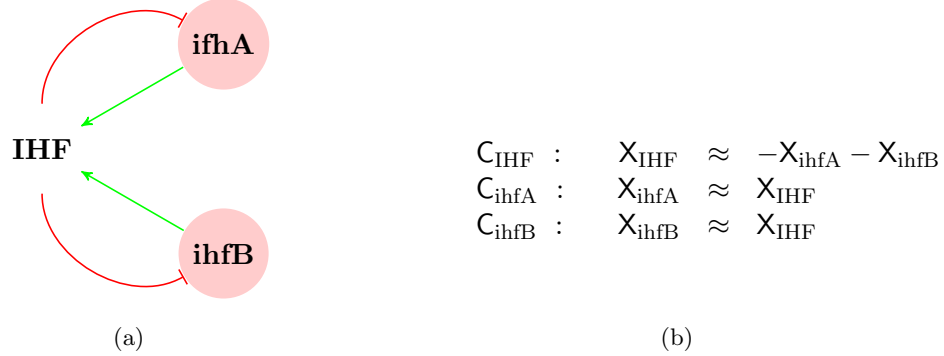


FIG. 6.2 – (a) Sous-graphe inconsistant avec les données du tableau 6.1. (b) Contrainte qualitative correspondante

qui posent problème. Une approche alternative, plus simple d'un point de vue calculatoire, consiste à chercher des sous-ensembles minimaux au sens de l'inclusion. Les sous-ensembles inconsistants de cardinal minimal sont aussi minimaux au sens de l'inclusion mais la réciproque est fautive. En pratique, nous commençons par calculer un ensemble \subset -minimal, ce qui est relativement facile d'un point de vue algorithmique. Si ce dernier est trop gros pour être facilement interprétable, nous recourons à la recherche d'ensembles de cardinal minimal.

Illustrons à présent le résultat de cette démarche sur notre problème. Comme on peut le voir sur la figure 6.2(a), on peut isoler un sous-système, composé des sommets *ihfA*, *ihfB* et *IHF*, qui n'est pas compatible avec les variations données en tableau 6.1. Cela peut se voir sur la contrainte qualitative associée à ce système, donnée en figure 6.2(b). La conjonction des contraintes C_{IHF} , C_{ihfA} et C_{ihfB} n'admet que la solution nulle partout, ce qui est contradictoire avec l'observation rapportée dans RegulonDB. Même sans regarder de près les équations, on remarque tout de suite sur le graphe d'interaction qu'il n'y a aucune boucle positive dans le graphe d'interaction. Par conséquent, le système n'admet qu'un seul état stable et la seule variation observable suite à un déplacement d'équilibre est la variation nulle.

6.1.4 Ajout des facteurs σ dans le modèle

Cette observation nous a conduit à rechercher d'autres régulateurs des gènes *ihfA* et *ihfB*, et finalement à considérer l'action des *facteurs* σ que nous avons initialement omis dans le modèle. Les facteurs σ sont des protéines intervenant dans l'initiation de la transcription et régulent de ce fait un très grand nombre de gènes. Ces régulations sont également recensées dans RegulonDB, et nous les avons ajoutées à notre modèle. Le tableau 6.2 indique le nombre de gènes cibles pour chaque facteur σ . Le graphe d'interaction obtenu comporte cette fois 1529 sommets et 3802 arcs, dont 175 de signe indéterminé ; il est présenté en figure 6.5. En particulier notre sous-graphe inconsistant est modifié comme présenté en figure 6.3(a). La contrainte correspondante 6.3(b) admet

TAB. 6.2 – Nombre de gènes cibles pour chaque facteur σ (régulations répertoriées dans RegulonDB, version de mars 2006)

Protéine	Gène	Gènes cibles
σ^{70}	rpoD	1047
σ^{38}	rpoS	114
σ^{54}	rpoN	100
σ^{24}	rpoE	48
σ^{32}	rpoH	29
σ^{19}	fecI	7

cette fois une variation négative pour ihfA et ihfB.

Néanmoins il s'avère que ce nouveau réseau n'est toujours pas compatible avec les données bibliographiques sur la réponse au stress nutritionnel. Le diagnostic produit dans ce cas est présenté en figure 6.4(a). Cette fois le problème est un peu plus complexe : si ihfA et ihfB diminuent, alors IHF doit également diminuer. Or si c'est le cas, seule une augmentation de rpoD pourrait justifier l'augmentation de dppA. Par ailleurs rpoS ne peut qu'augmenter, puisqu'il est le seul régulateur connu de fic, qui augmente. Mais alors, il n'y a pas d'explication à la diminution de ihfA. Nous nous sommes cette fois tourné vers les données, et nous avons trouvé que l'annotation fournie par RegulonDB (version de mars 2006) pour ihfA et ihfB est en contradiction avec les publications qui doivent la justifier. On trouve notamment dans [2] l'extrait suivant :

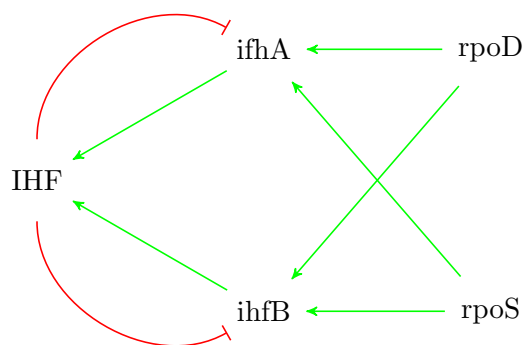
[...] transcription of himA and himD promoters increases as the *E. coli* cells enter the stationary phase of growth [...]

où himA et himD sont des synonymes de ihfA et ihfB respectivement ; l'entrée en phase stationnaire correspond à la réaction physiologique des bactéries sous l'effet d'un stress nutritionnel¹. Une fois les deux annotations corrigées, les données bibliographiques sur le stress nutritionnel sont consistantes avec le graphe d'interaction avec facteurs σ .

6.1.5 Prédiction de la réponse globale au stress nutritionnel

Disposant de données sûres mais partielles sur la réponse d'*E. coli* à un stress nutritionnel, nous avons cherché à prédire les variations dans le reste du réseau. Nous avons obtenu au total 381 variations prédites (invariants de la contrainte qualitative associée au graphe et aux données bibliographiques), soit approximativement un quart des sommets du graphe d'interaction. Afin de valider ces prédictions nous les avons comparées à des données expérimentales sur l'entrée en phase stationnaire. Plus précisément les prédictions ont été comparées à des résultats de puces à ADN réalisées dans [45] et [99]. Ces données sont compilées et mises à disposition dans la base de données *Gene Expression Omnibus* (GEO).

¹La phase stationnaire est ainsi nommée parce que les bactéries cessent de se multiplier.

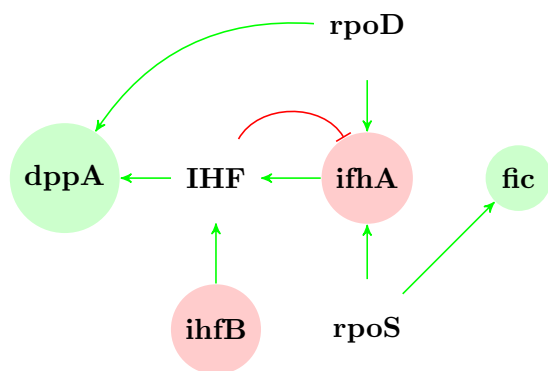


(a)

$$\begin{aligned}
 C_{\text{IHF}} : \quad X_{\text{IHF}} &\approx -X_{\text{ifhA}} - X_{\text{ifhB}} \\
 C_{\text{ifhA}} : \quad X_{\text{ifhA}} &\approx X_{\text{IHF}} + X_{\text{rpoS}} + X_{\text{rpoD}} \\
 C_{\text{ifhB}} : \quad X_{\text{ifhB}} &\approx X_{\text{IHF}} + X_{\text{rpoS}} + X_{\text{rpoD}}
 \end{aligned}$$

(b)

FIG. 6.3 – Correction du graphe d'interaction, selon les données disponibles sur les facteurs σ .



(a)

$$\begin{aligned}
 C_{\text{IHF}} : \quad X_{\text{IHF}} &\approx -X_{\text{ifhA}} - X_{\text{ifhB}} \\
 C_{\text{ifhA}} : \quad X_{\text{ifhA}} &\approx X_{\text{IHF}} + X_{\text{rpoS}} + X_{\text{rpoD}} \\
 C_{\text{fic}} : \quad X_{\text{fic}} &\approx X_{\text{rpoS}} \\
 C_{\text{dppA}} : \quad X_{\text{dppA}} &\approx X_{\text{rpoD}}
 \end{aligned}$$

(b)

FIG. 6.4 – Sous-graphe inconsistant avec les données bibliographiques, après ajout des régulations par les facteurs σ

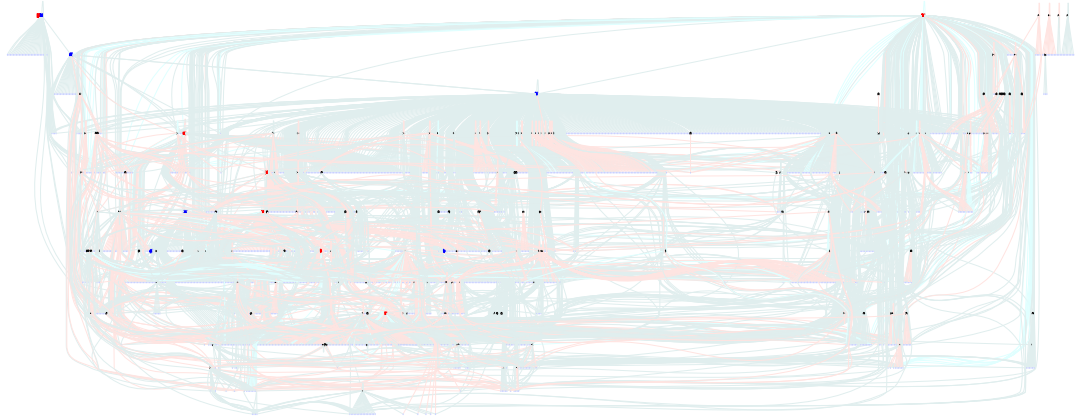


FIG. 6.5 – Graphe d’interaction pour le réseau transcriptionnel d’*E. coli*, avec facteurs σ .

Comme mentionné au chapitre 2, ainsi qu’au paragraphe 4.4.1, l’interprétation des données brutes en signes passe par la définition d’un seuil, au-dessus duquel la variation mesurée est jugée significative. Si la variation est trop faible, on peut au choix la considérer nulle ou inconnue. Nous avons opté pour la deuxième possibilité, en commençant par un seuil nul : toutes les données sont interprétées en signes $+$ et $-$ uniquement. Pour être cohérent avec ce choix, les calculs de prédictions ont également été faits en excluant les solutions contenant des variations nulles. Ainsi les prédictions sont également dans $\{+, -\}$.

TAB. 6.3 – Validation des prédictions obtenues à partir des données bibliographiques. Les prédictions sont comparées à des données de puces à ADN, provenant de [45] et [99].

Source de données	Nombre de gènes comparés	Gènes validés (%)
Phase stationnaire après 20 minutes	292	51.71%
Phase stationnaire après 60 minutes	281	51.2%

Les résultats de la comparaison sont donnés au tableau 6.3. Nous avons inclus dans cette comparaison deux temps expérimentaux (20 et 60 minutes). Dans chaque cas, il y a des variations manquantes (mesure brute absente), et la deuxième colonne du tableau montre le nombre de gènes qui sont à la fois prédits par notre approche et dont la mesure est disponible dans la source. La troisième colonne donne le pourcentage de gènes dans cette intersection qui ont effectivement un signe commun.

Les chiffres obtenus sont à première vue peu encourageants : les prédictions font à peine mieux qu’un tirage aléatoire. Néanmoins ce résultat peut être partiellement

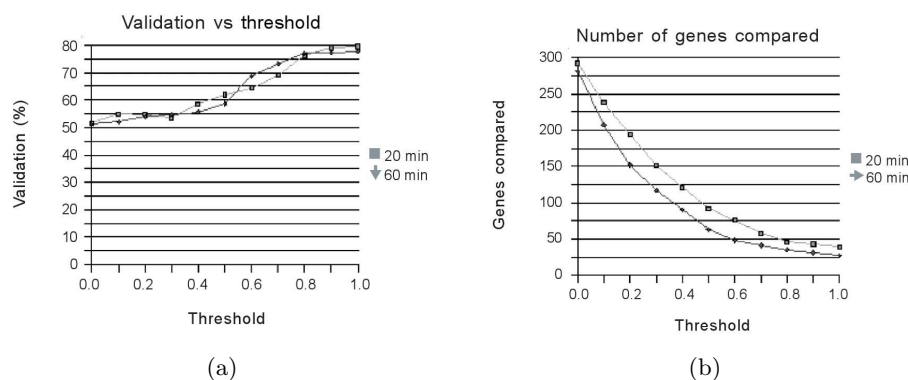


FIG. 6.6 – Prédiction de la réponse au stress nutritionnel. Les courbes montrent l'évolution des résultats en fonction du seuillage des données. On se donne un seuil en dessous duquel les variations mesurées sont jugées non significatives. Dans ce cas, la variable concernée est déclarée non observée. Plus on choisit un seuil élevé, moins on garde de données, mais plus ces données sont fiables. (a) Pourcentage de prédictions correctes en fonction du seuil. (b) Nombre de gènes conservés en fonction du seuil.

dû au bruit dans les données : comme indiqué ci-dessus, nous n'avons initialement effectué aucun filtrage sur les données. En choisissant un seuil en-dessous duquel les variations sont ignorées, on peut s'attendre à observer une meilleure concordance entre les prédictions et les données. Pour le vérifier, nous avons itéré le calcul pour diverses valeurs du seuil ; le tracé est donné en figure 6.6

La première courbe donne le pourcentage de prédictions validées en fonction du seuil choisi. Plus le seuil est élevé, plus on exclut de gènes de la comparaison. Le nombre de gènes considérés (c'est-à-dire qui ont une variation expérimentale suffisante et qui sont prédits par notre approche) est donné en fonction du seuil sur la courbe de droite. Ainsi, lorsque l'on sélectionne les gènes dont la variation est la plus importante, le taux de prédiction s'améliore sensiblement. Cet effet n'est pas très étonnant si l'on se souvient du niveau de bruit généralement observé dans les mesures d'expression (voir figure 1.2), et particulièrement pour les gènes faiblement exprimés dans les deux conditions comparées.

Il reste que le pourcentage de prédictions validées est loin d'être parfait, même après avoir choisi un seuil très restrictif. Nous nous sommes donc intéressés aux erreurs de prédiction les plus flagrantes, c'est-à-dire pour lesquelles la variation mesurée ne pose aucun problème d'interprétation. C'est notamment le cas pour le gène *ilvC* dont l'expression devrait augmenter d'après notre modèle, et qui diminue en fait fortement. Le gène *ilvC* n'admet qu'un seul activateur, à savoir le gène *ilvY*, qui n'a lui-même qu'un seul activateur, à savoir *rpoD*. Ce dernier est prédit comme augmentant à l'entrée en phase stationnaire, ce qui implique l'augmentation de *ilvY* et de *ilvC*. Une recherche bibliographique nous a conduit à des publications portant spécifiquement sur la régulation de ces gènes [75, 69]. Ces travaux montrent l'influence des phénomènes de super-hélicité de l'ADN dans la régulation transcriptionnelle de ces gènes. On peut

notamment lire dans [75] :

Evidence that this promoter coupling is DNA supercoiling-dependent is provided by the observation that a novobiocin-induced decrease in global negative superhelicity results in an increase in *ilvY* promoter activity and a decrease in *ilvC* promoter activity predicted by the *in vitro* data. We suggest that this transcriptional coupling is important for coordinating basal level expression of the *ilvYC* operon with the nutritional and environmental conditions of cell growth.

Des travaux expérimentaux [3] montrent que le phénomène de superhélicité tend à diminuer lors du passage en phase stationnaire. Cet évènement, provoqué *in vitro* dans [75] conduit à une diminution de l'activité transcriptionnelle du gène *ilvC*, et une augmentation pour le gène *ilvY*. En complétant notre modèle avec l'influence de la topologie de l'ADN, nous obtenons une correction validée de notre modèle.

6.2 Inférence de graphes d'interactions

Nous abordons maintenant un deuxième problème essentiel en biologie systémique : la reconstruction de modèles à partir de données expérimentales – *reverse-engineering* dans la littérature. Dans notre contexte, il s'agit de construire un graphe d'interaction compatible avec des données de déplacement d'équilibre. Nous nous sommes intéressés à un cas particulier de ce problème, où l'on suppose les arcs du graphe d'interaction connus, et où seuls les signes sur les arcs sont à déterminer.

L'intérêt pratique de cette question est de permettre l'intégration des données *chIP-on-chip* avec les mesures d'expression : les premières fournissent les arcs du graphe d'interaction, puisqu'elles déterminent les cibles des facteurs de transcription ; les deuxièmes vont nous permettre de déduire le signe des arcs, c'est-à-dire l'effet (activation ou inhibition) de ces facteurs de transcription sur leurs cibles.

L'intégralité de ce travail est reproduite en annexe A, et nous en synthétisons ici les principaux résultats. Nous avons procédé en trois étapes, partant de données bien validées et/ou artificielles sur la bactérie *E. coli*, pour arriver à un contexte d'utilisation réaliste chez la levure. En voici les grandes lignes.

6.2.1 Limites théoriques de l'approche

Inférence à partir d'une seule mesure On peut commencer par une remarque simple : si un gène a admet un unique régulateur b , alors il suffit d'une seule mesure voyant a et b varier pour déterminer l'effet de la régulation. Elle est positive si a et b varient dans le même sens, négative sinon. En supposant que l'on mesure tous les sommets d'un graphe d'interaction donné, ce raisonnement simple règle donc la question pour tous les sommets n'ayant qu'un seul prédécesseur. Sur le graphe d'interaction issu de RegulonDB, cela représente plus de 600 régulations (sur un total de 3802). La valeur ajoutée de notre approche est donc de savoir combiner plusieurs mesures pour déduire le signe des régulations dans le cas général.

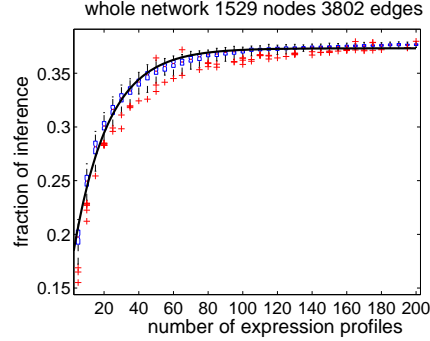


FIG. 6.7 – Statistique du nombre de signes déduits à l’aide d’un nombre donné de mesures. En abscisse, nombre de mesures disponibles ; en ordonnées le nombre de signes de régulation que l’on peut en déduire.

Inférence à partir de plusieurs mesures Si l’on dispose de plusieurs mesures, nous étudions les invariants de la contrainte de consistence, et rapportons tous les invariants qui sont des signes (les invariants peuvent également être des variations non mesurées). Plus précisément, pour un graphe d’interaction \mathcal{G} entièrement signé, on calcule un ensemble de k solutions de la contrainte de consistence $\mathcal{C}_{\mathcal{G}}^{\emptyset}$, notées $\mu = \{\mu_1, \dots, \mu_k\}$. On construit le graphe \mathcal{H} , qui a les mêmes arcs que \mathcal{G} mais où aucun signe n’est connu. Enfin, on cherche les invariants de la contrainte $\mathcal{C}_{\mathcal{H}}^{\mu}$. Pour étudier l’intérêt pratique de cette approche, nous examinons son comportement sur le graphe d’interaction basé sur RegulonDB (voir section précédente). Ce réseau étant entièrement signé, nous l’avons utilisé pour générer aléatoirement des mesures compatibles. Plus précisément, pour un entier k donné, on génère aléatoirement k mesures compatibles avec le graphe d’interaction. On construit ensuite une version *non signée* de ce graphe d’interaction, que l’on combine avec les k mesures pour obtenir les signes prédits, dont on compte le nombre. Cette opération est répétée une centaine de fois pour obtenir une statistique du nombre de signes déduits à l’aide de k mesures. Le résultat obtenu est donné en figure 6.7. On retrouve, à l’origine de la courbe, les régulations inférables par une seule mesure. La courbe monte rapidement vers un plateau supérieur à 35%. Nous voyons notamment qu’on peut en moyenne déterminer 30% des régulations du graphe à l’aide d’environ 30 mesures.

Réalisation Pour l’inférence des régulations, le calcul des invariants peut, au choix être effectué par un moteur ASP soit en utilisant les diagrammes de décision et notre méthode de décomposition. Le calcul des simulations est relativement coûteux, aussi nous avons cherché à optimiser autant que faire ce peut le calcul des invariants. À cette fin, nous l’avons décomposé en deux parties. La première consiste à calculer des parties bien choisies de la contrainte de consistence avec des diagrammes de décision. Nous obtenons ainsi la plupart des invariants dans un temps relativement court. La deuxième phase utilise le moteur ASP `clasp` pour prouver la non-invariance des autres variables. La procédure complète est détaillée dans l’article reproduit en annexe.

Signes inférables La courbe 6.7 semble indiquer l'existence d'une limite au nombre de signes pouvant être retrouvés, correspondant à un peu plus de 35% du graphe d'interaction. Nous pouvons en fait calculer cette limite exactement, en déterminant les signes que l'on peut déduire quand *toutes* les mesures compatibles sont disponibles. L'algorithme naïf consistant à générer toutes les mesures explicitement, puis à les utiliser comme au paragraphe précédent n'est pas envisageable. On peut procéder de la manière suivante : la contrainte $C_{\mathcal{G}}^{\emptyset}$ construite plus haut décrit toutes les mesures compatibles avec \mathcal{G} . De plus toute mesure compatible avec \mathcal{G} doit être compatible avec \mathcal{H} , ce qui se traduit par la contrainte :

$$\forall X \ C_{\mathcal{G}}^{\emptyset}[X] \Rightarrow C_{\mathcal{H}}^{\emptyset}[X, S]$$

qui porte sur les variables de signe uniquement, et dont on peut calculer les invariants. Ces invariants sont les seuls signes inférables à partir de mesures expérimentales. Notons bien la raison de cette limitation : il est impossible de prouver expérimentalement qu'une mesure arbitraire n'est pas compatible avec le système étudié. Dans le cas contraire, on pourrait remplacer dans la formule ci-dessus l'implication par une équivalence, et tous les signes seraient inférables, moyennant le bon ensemble d'observations.

Sur notre graphe d'interaction pour *E. coli*, nous obtenons que 40.8% des régulations sont inférables, c'est-à-dire un peu plus de 1550 signes. Ce maximum peut n'être atteint que pour un très grand nombre de mesures. Le calcul a été réalisé grâce aux procédures sur les diagrammes de décision, en utilisant les techniques de décomposition sur les contraintes qualitatives décrites au paragraphe 4.5.2.

Bilan Cette étude sur le réseau transcriptionnel d'*E. coli* nous permet d'évaluer les limites de l'inférence de régulation par contraintes de consistance, *dans le cas où les données – ou du moins leur interprétation qualitative – ne sont pas bruitées*. Nous voyons ainsi que tous les signes ne sont pas inférables, et d'autres ne le sont qu'au prix d'un très grand nombre de mesures. Néanmoins, nous montrons sur cet exemple qu'un petit nombre de mesures permet déjà d'approcher de manière significative cette limite.

6.2.2 Validation par des mesures d'expression

Prédictions sous données non consistantes Dans une deuxième étape, nous remplaçons les mesures artificielles par des données d'expressions, compilée dans [26]. Dans ce cadre, les mesures peuvent donc n'être pas consistantes avec le graphe d'interaction que nous avons construit. Les données expérimentales peuvent donc nous amener à déduire des signes erronés (c'est-à-dire non conformes à l'annotation données dans RegulonDB). Plus ennuyeux, les données peuvent ne pas être consistantes avec le graphe non signé. On peut en donner un exemple simple : soit un système composé de deux espèces A et B avec une régulation $B \rightarrow A$. Supposons qu'on dispose de deux mesures $\mu_1(A) = +$, $\mu_1(B) = +$ d'une part, et $\mu_1(A) = +$, $\mu_1(B) = -$ d'autre part. L'ensemble n'est pas consistant, car la contrainte qualitative de consistance aux sommets n'admet aucune solution. *A fortiori*, on ne peut donc pas calculer les invariants. Il nous faut donc répondre à la question suivante : comment obtenir (définir ?) des prédictions lorsque les

données expérimentales ne sont pas consistantes avec le graphe d'interaction non signé ? La réponse la plus sage consiste à détecter chaque défaut à la règle de consistance, et examiner manuellement le problème. Il n'est toutefois pas toujours possible d'investir le temps nécessaire pour arriver corriger complètement les défauts.

Un algorithme pour la prédiction en présence de bruit Nous avons ici proposé une approche intuitive proche de la notion de diagnostic. En voici les grandes lignes :

- soit un graphe non signé, et un ensemble de mesures qui ne sont pas consistantes avec ce graphe
- en utilisant l'approche de diagnostic décrite plus haut (voir paragraphe 6.1.3), on isole un sous-ensemble d'équations et de mesures inconsistantes, que l'on supprime de la contrainte de consistance.
- cette opération est répétée jusqu'à obtenir un graphe et des données consistants pour lesquels on calcule les invariants sur les signes des régulations,
- pour chaque prédiction, on appelle *indice de confiance* le nombre de mesures compatibles avec la prédiction.

Il s'agit donc essentiellement de retirer des données jusqu'à arriver à une contrainte satisfiable, sur laquelle on peut calculer des prédictions. Bien sûr cette approche n'est pas pleinement satisfaisante, puisque les données mises à l'écart ne sont pas uniquement déterminées : plusieurs sous-ensembles de données et d'équations peuvent expliquer l'inconsistance constatée. Néanmoins cette approche a le mérite de la simplicité, et nous permet d'observer l'intérêt « grandeur nature » de la notion de diagnostic (par le biais de l'indice de confiance).

Résultats expérimentaux Les résultats obtenus par l'algorithme ci-dessus sont donnés en figure 6.8. Pour l'indice de confiance le plus faible ($k = 1$), on obtient 183 signes, mais 42% de ceux-ci sont conformes à l'annotation de RegulonDB. Pour des indices de confiance plus élevés, le nombre de prédictions chute (c'est attendu), mais le taux de faux-positifs diminue également de manière sensible, arrivant à 8% pour $k = 15$.

Bilan Nous avons montré une approche simple pour la formulation de prédictions dans le cas de données non consistantes. Son principe peut être perfectionné, mais elle illustre déjà l'intérêt pratique de la notion de diagnostic, puisqu'elle privilégie les prédictions confirmées par le plus grand nombre d'observations (c'est-à-dire celles qui en contredisent le minimum).

6.2.3 Application chez *S. cerevisiae*

La dernière étape de ce travail expérimental consiste à appliquer l'algorithme de prédiction décrit plus haut dans un contexte plus difficile : le graphe d'interaction provient cette fois de données chIP-on-chip, et constitue donc un modèle beaucoup moins fiable que le réseau fourni par RegulonDB pour la bactérie *E. coli*.

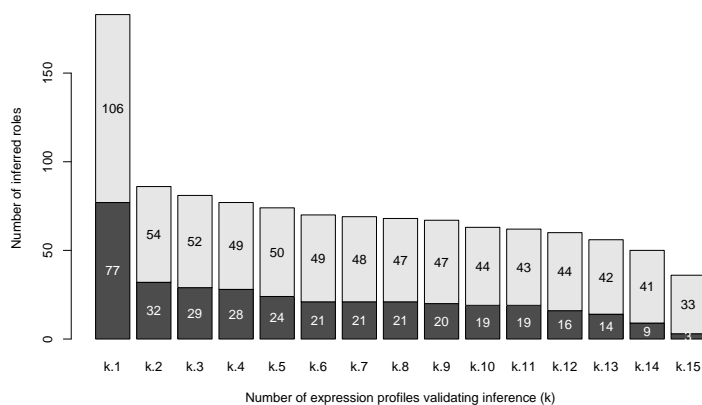


FIG. 6.8 – Résultats de l'inférence des signes de régulation sur le réseau d'*E. coli*, à partir de données d'expression.

Données Nous avons étudié quatre réseaux transcriptionnels, correspondant aux données produites par Lee *et al* [55] et Macisaac *et al* [60]. Les trois premiers sont de taille modeste (moins de 100 sommets) car limités aux facteurs de transcriptions. Le dernier regroupe toutes les cibles des facteurs de transcriptions étudiés dans [55] ; il compte plus de 2400 sommets et 4300 régulations. Les mesures d'expression utilisées sont celles qui ont été compilées dans [43].

Résultats Comme avec le réseau transcriptionnel d'*E. coli*, les réseaux que nous avons construits ne sont pas consistants avec les données d'expression. La procédure de diagnostic décrite plus haut nous a permis d'isoler les défauts à la règle de consistance ; il s'avère que les défauts typiques tombent systématiquement dans un des cas montrés en figure 6.9. Dans le cas du plus grand graphe, nous avons compté plus de 740 de ces défauts, couvrant un peu moins de 18% du graphe d'interaction total. En utilisant l'algorithme de prédiction décrit plus haut, nous obtenons 631 signes prédits avec un indice de confiance supérieur à 1, et 198 avec un indice supérieur à 3. Pour valider ces prédictions, nous utilisons comme référence le réseau construit dans [65] à partir de données bibliographiques. Sur les 631 régulations prédites avec un indice supérieur à 1, 23 sont annotées dans le réseau, et 16 concordent ; sur les 198 régulations prédites avec un indice supérieur à 3, 19 sont annotées dans le réseau et 18 concordent.

Bilan

Nous avons exposé dans ce chapitre deux applications de notre approche sur des données réelles. La première porte sur la réponse transcriptionnelle de la bactérie *E. coli* à un stress nutritionnel : il s'agissait, partant d'un graphe d'interaction complètement annoté, et d'un ensemble (restreint) d'observations issues de la littérature, de prédire

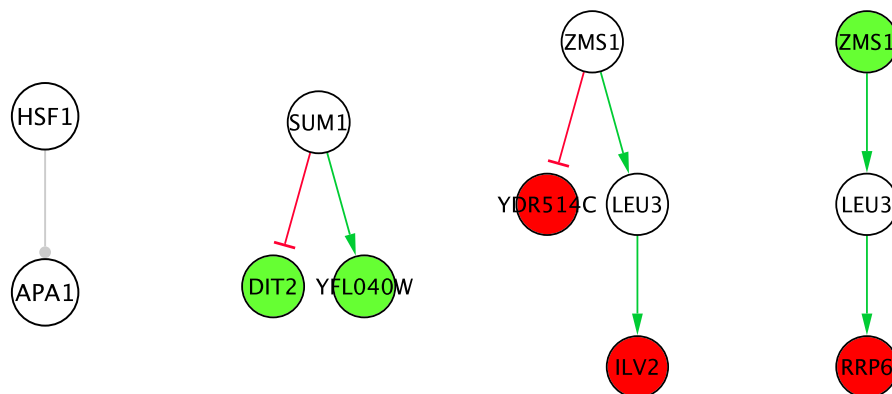


FIG. 6.9 – Cas typiques de défaut à la contrainte de consistance, trouvés dans les données sur *S. cerevisiae*.

la réponse globale de la bactérie. Dans la seconde application, le but était de prédire l'influence des facteurs de transcription sur leurs gènes cibles, en combinant des données chIP-on-chip et des données d'expression. Nous avons dans un premier temps démontré la faisabilité de la méthode en utilisant des données fiables sur la bactérie *E. coli*, puis produit des prédictions à partir de données sur la levure.

Validation algorithmique Ces expérimentations répondent positivement à la question du passage à l'échelle : les algorithmes que nous avons proposés sont à même de traiter des données transcriptomiques portant sur plusieurs milliers de transcrits, dans un temps qui n'excède pas quelques minutes. À ce titre, nos deux méthodes de résolution (par diagramme de décision, ou par programmation logique) ont un comportement similaire, même si l'utilisation des diagrammes de décision pour de si grands systèmes reste délicate – notamment à cause des passages obligés de réduction/décomposition des contraintes. L'utilisation du solveur ASP `clasp` donne en revanche des résultats tout à fait satisfaisants, pour une utilisation relativement simple.

Analyse de données Ces travaux sur données réelles amènent à une observation capitale : le critère de consistance n'est généralement pas vérifié dans les mesures expérimentales disponibles. Quoique décevant de prime abord, ce résultat est au contraire un formidable levier pour l'analyse de données, puisque nous avons mis en évidence que l'étude des défauts permet dans de nombreux cas de corriger le modèle étudié, ou les données utilisées. Dit autrement, nous avons proposé un modèle suffisamment peu précis pour s'accommoder des données disponibles, mais qui néanmoins peut guider vers des connaissances nouvelles sur le système étudié.

Chapitre 7

Discussion

Nous avons à présent décrit en détail notre approche, tant en ce qui concerne son principe que ses aptitudes au traitement de données réelles. Nous proposons dans ce chapitre de resituer notre travail parmi d'autres contributions abordant la comparaison grande échelle d'un modèle graphique et de données expérimentales. Nous approfondissons notamment la comparaison avec l'approche développée par Yeang, Ideker et Jaakkola [102].

7.1 Travaux connexes

Notre travail peut être vu comme une proposition pour relier une représentation graphique d'un système biologique au comportement dudit système. La relation que nous avons décrite est basée sur un modèle physique qui donne d'une part une sémantique à la représentation graphique, et d'autre part une interprétation des mesures expérimentales. Cette relation porte essentiellement sur une propriété topologique (prédécesseurs d'un sommet) sur un type de graphe (les graphes d'interaction) et un type de mesure (signe des variations entre deux états d'équilibre). Nous allons dans un premier temps mentionner un certain nombre de travaux abordant, dans des contextes distincts, la même question : comment expliquer ou prédire des observations expérimentales sur un système à partir de sa description sous forme d'un graphe ?

7.1.1 Circuits du graphe d'interaction

Nous avons déjà cité au paragraphe 3.2.1 quelques résultats connus sur le graphe d'interaction d'un système. Ainsi, l'absence de circuit positif implique l'unicité de l'équilibre d'un système ; l'absence de circuit négatif implique l'absence d'oscillations amorties ou entretenues. Par ces résultats, on a donc relié des propriétés dynamiques (unicité de l'équilibre par exemple) aux circuits d'un graphe d'interaction, sous diverses sémantiques (modèles différentiels avec ou sans dégradation [88], réseau booléens [74]/multi-valués [76] ...).

7.1.2 Régulons

Gutiérrez et collègues [34] ont adopté une approche très similaire au travail présenté dans cette thèse : ils proposent de comparer des connaissances issues de la littérature (en particulier le contenu de la base RegulonDB étudiée au chapitre 6) à des mesures sur le transcriptome de la bactérie *E. coli* dans différentes conditions expérimentales. Le modèle physique justifiant la comparaison intègre des connaissances générales sur la régulation transcriptionnelle chez les prokaryotes. De manière analogue à ce qui est présenté dans cette thèse, ce formalisme explicite la relation entre l'état d'un gène et celui de ses prédécesseurs. Il introduit notamment la notion de *régulon*, qui désigne les ensembles de gènes ayant exactement le même ensemble de prédécesseurs. Tout comme la comparaison données/modèle nous a permis de déterminer l'effet des facteurs de transcription sur leurs gènes cibles, elle permet dans ce cadre d'inférer les fonctions de régulation propres à chaque régulon.

7.1.3 Chemins métaboliques

On appelle métabolites les « petites » molécules¹ présentes dans le milieu cellulaire qui sont liées à la production d'énergie ou des structures cellulaires (cela inclut les acides aminés, acides nucléiques, lipides et sucres simples). Les mécanismes de production d'énergie, de synthèse ou de destruction des métabolites (l'ensemble est appelé *métabolisme*) est souvent représenté par un graphe de réactions analogue à ce que nous avons utilisé au paragraphe 3.2.7. Chaque réaction transforme ou assemble des métabolites par le biais d'une réaction enzymatique ; chaque réaction est donc associée à l'enzyme qui la catalyse, et on s'intéresse au *flux* de métabolites transformées par la réaction. Ce flux est dans certains cas mesurable, et en partie fonction des régulations génétiques : si le gène codant pour l'enzyme n'est pas exprimé, alors le flux à travers la réaction associée est nul. Pour relier ces observations au graphe de réactions, on a recours à une analyse de flux à l'équilibre (*Flux Balance Analysis*, FBA) [80] : on suppose que les réactions suivent une dynamique différentielle et on étudie l'ensemble des flux à l'équilibre compatibles avec les observations. Cet ensemble peut être infini, donc difficile à visualiser, mais s'avère être le noyau d'une application linéaire de rang fini. On peut donc en rechercher une famille génératrice finie ; chaque vecteur de flux dans cette famille peut être représenté par un sous-graphe « actif ». Notamment, si un vecteur de flux a peu de composantes non-nulles, cette représentation fait apparaître des *chemins métaboliques* (*metabolic pathways*), c'est-à-dire une suite de transformations menant typiquement de métabolites simples à la production d'énergie ou de biomasse. On obtient finalement que tout flux à l'équilibre est une superposition de chemins métaboliques simples. On a ainsi relié des observations sur les flux métaboliques à un graphe de réactions en faisant l'hypothèse d'un modèle différentiel à l'équilibre. L'analyse offre par ailleurs une visualisation intuitive des états de flux possibles comme superposition de chemins métaboliques.

¹Elles sont petites à côté des protéines, typiquement. Elles constituent en général les briques de bases de structures moléculaires plus complexes.

7.1.4 Cascades de régulations

La réponse d'une cellule aux signaux présents dans l'environnement passe par une série de réactions et de mécanismes physiques, comprenant notamment des interactions avec des récepteurs sur la membrane, des phénomènes de transport par des vésicules, des interactions protéine/protéine et la régulation de gènes dans le noyau. Là encore on s'intéresse aux chemins en tant qu'ils constituent des suites d'évènements reliant une perturbation à une réaction observable. Les chemins trouvés sont appelées cascades de régulation (*regulatory pathways*) ; ils diffèrent des chemins métaboliques par le modèle physique sous-jacent : les chemins métaboliques correspondent à des flux équilibrés minimaux (en un certain sens) ; les cascades de régulations sont des chemins explicatifs reliant une perturbation à ses effets observables². Nous mentionnerons deux approches précisant cette notion de chemins explicatifs. La première, implémentée dans les logiciels BIOCHAM [15] et Pathway Logic [24], consiste à interpréter un graphe de réaction comme un ensemble de règles de réécriture : l'état d'un système est représenté par un vecteur de booléens (absence/présence de chaque espèce), et cet état est modifié par l'application des règles. Les règles sont applicables dès que tous les substrats de la réaction sont présents. Pour relier des observations expérimentales (état d'activation des gènes suite à une perturbation) au graphe de réactions, on recherche donc une suite d'applications de règles menant du profil initial au profil final. On montre dans [24] que sous la sémantique choisie, cela correspond à un sous-graphe du graphe de réactions, qui correspond à la cascade de régulation expliquant la réponse à la perturbation. La deuxième approche [102], que nous détaillons plus loin, permet d'étudier les effets d'une délétion de gène (*knock-out*) : par manipulation génétique on produit une variété d'un organisme où l'un des gènes ne s'exprime plus. On soumet les deux variétés (sauvage et mutante) aux mêmes conditions et on compare les différences de comportement. Les observations portent typiquement sur l'expression des gènes et correspondent aux variations entre les deux souches. Dès lors, l'objectif est de chercher à expliquer toute variation observée par une suite de régulations (chemin dans le graphe) partant du gène muté.

7.1.5 Bilan

Nous avons revu quelques approches proposées pour comparer une représentation graphique à des observations expérimentales, *via* l'introduction d'un modèle physique. La comparaison avec notre approche n'est pas systématiquement possible, puisque les types de données expérimentales utilisés dans chaque approche ne sont pas nécessairement compatibles. Il n'en reste pas moins que chaque approche fournit un critère de consistance – au sens où nous l'avons défini dans cette thèse – entre une représentation du

²Formellement, rien n'empêche de rechercher des chemins explicatifs dans les réseaux métaboliques, ni d'appliquer des méthodes de *Flux Balance Analysis* aux réseaux de signalisation (comme par exemple dans [70]). Le choix du bon outil reste avant tout une question de modélisation (que cherche-t-on ?) et d'adéquation aux données disponibles. La différence que nous avons voulu souligner est que les chemins métaboliques sont des composantes simples d'un état d'équilibre, alors que les chemins explicatifs doivent être vus comme des séries d'évènements.

système et des données expérimentales. L'existence de ces différents critères de consistance pose naturellement la question de leur comparaison. Étant donnés deux critères A et B , cela demande :

1. d'expliciter un cadre formel pour leur comparaison, c'est-à-dire une interprétation commune des données ;
2. de déterminer si des modèles admis par A sont admis par B (et réciproquement) ;
3. d'étudier, dans le cas où une telle inclusion n'existe pas, l'intersection des deux critères.

Donnons-en un exemple : prenons comme critère A la satisfaction des contraintes introduites dans cette thèse, et comme critère B l'existence d'une trajectoire dans un modèle différentiel linéaire par morceaux, comme ceux définis dans [87], [22] ou [77]. Ces modèles sont connus pour être des abstractions qualitatives de systèmes différentiels continus. En particulier [86], le modèle discrétisé conserve exactement l'ensemble des points stationnaires du modèle continu. Cela implique en particulier que tout déplacement d'équilibre du modèle discrétisé est admis par nos équations qualitatives. On en déduit que le critère B est plus restrictif que le critère A .

Nous passons à présent à une comparaison plus détaillée entre notre approche et celle développée par Yeang, Ideker et Jaakkola dans [102]. Cette étude est facilitée par la relative proximité entre les modèles que nous utilisons. Elle illustre notamment l'intérêt d'explicitier et de comparer les critères de consistance données/modèle.

7.2 Chemins dans le graphe d'interaction

Les travaux de Yeang *et al* [102, 103] portent sur la modélisation des expériences dites de *knock-out* (délétion de gène), où l'on construit une souche bactérienne « mutante » à partir d'une souche « sauvage ». La mutation consiste généralement à supprimer l'activité d'un gène, voire deux. Quand la mutation n'est pas létale, on compare alors la réponse des deux souches à une même perturbation. Sur le plan modélisation, l'objectif est de relier les effets de la délétion (c'est-à-dire les variations observées) au rôle connu du gène inactivé. Notamment, on cherche à faire apparaître une notion de chemin (*pathway* dans la littérature) menant de la cause (gène inactivé) aux effets (variations observées).

7.2.1 Le modèle de Yeang-Ideker-Jaakkola (YIJ)

Le modèle décrit dans [102] intègre les interactions protéine/protéine et protéine/ADN connues ou supposées dans une structure de graphe. Les arcs du graphe sont étiquetés par un signe $+$ ou $-$ indiquant l'effet d'un gène (ou d'une protéine) sur un autre gène. On suppose disposer par ailleurs de données de *knock-out* : pour chaque délétion de gène, on connaît les effets sur le reste des sommets du graphe (augmentation, diminution, pas d'effet significatif).

Nous introduisons maintenant plus formellement ce modèle en en excluant les éléments liés aux interactions protéine/protéine. Cette hypothèse ne modifie pas fon-

damentalement les conclusions de cette étude, et en facilite grandement l'exposition : le graphe du modèle YIJ est alors tout à fait analogue à un graphe d'interaction.

Description du modèle et des données Les éléments du modèle sont les suivants :

- un ensemble de gènes $V = \{g_1, \dots, g_n\}$
- un ensemble d'arcs (dirigés) $E \subset V \times V$, représentant l'ensemble des interactions protéine-ADN potentielles, c'est-à-dire celles pour lesquelles, il existe un support expérimental minimum (par exemple, p -valeur raisonnable dans une expérience de chip-on-chip). On suppose qu'aucune autre interaction n'existe en dehors de celles-ci.
- des variables binaires $X_E = \{x_e \mid e \in E\}$ indiquant qu'une interaction est fonctionnelle (autrement dit, qu'une liaison protéine-ADN est réellement suivie d'effet).
- des variables qualitatives (i.e. à valeurs dans \mathbb{S}) $S_E = \{s_e \mid e \in E\}$ indiquant l'effet de l'interaction (activation, inhibition, sans effet)
- une collection de *knock-outs*, qui est un sous-ensemble \mathcal{K} de $G \times G \times \mathbb{S}$ dont les triplets (g_i, g_j, k_{ij}) sont tels que $g_i \neq g_j$ et k_{ij} représente la variation de g_j entre un *knock-out* de g_i et la condition de référence.

Pour un gène i muté, on s'intéresse à l'ensemble des chemins de i vers j où j est un autre gène du modèle. Cet ensemble est noté Π_{ij} . Pour un chemin $a \in \Pi_{ij}$, on notera X_a (resp. E_a) l'ensemble des sommets (resp. arcs) qu'il contient.

7.2.2 Relation modèle – données

Pour déterminer la valeur des paramètres d'un modèle (existence des arcs avec les variables x_e , signe des régulations avec les variables s_e), le modèle YIJ est étiqueté par une loi de probabilité, dont nous décrivons la construction. Soit un triplet $(g_i, g_j, k_{ij}) \in \mathcal{K}$, on introduit pour tout $a \in \Pi_{ij}$ la fonction ψ_{ija} définie par :

$$\psi_{ija}(X_a, S_a) = \prod_{e \in E_a} \mathbf{1}[x_e = 1] \cdot \mathbf{1} \left[\prod_{e \in E_a} s_e = k_{ij} \right]$$

où $\mathbf{1}[\]$ représente la fonction indicatrice. La fonction ψ_{ija} est à 1 si a est un chemin explicatif pour l'effet de la délétion de g_i sur g_j . Il faut ensuite pouvoir détecter que l'un au moins des chemins entre g_i et g_j est un chemin explicatif. La fonction ψ_{ij} calcule la disjonction des indicatrices ψ_{ija} :

$$\psi_{ij}(X_E, S_E) = 1 - \prod_{a \in \Pi_{ij}} (1 - \psi_{ija}(X_a, S_a))$$

La loi de probabilité complète pour un modèle et des données de *knock-out* est donnée par :

$$\mathbb{P}(X_E, S_E) = \prod_{(g_i, g_j, k_{ij}) \in \mathcal{K}} \psi_{ij}(X_E, S_E) \quad (7.1)$$

Cette relation exprime une hypothèse d'indépendance entre les chemins explicatifs choisis. Le modèle peut aussi intégrer, selon la même hypothèse d'indépendance (donc en ajoutant quelques termes multiplicatifs) les indices de confiance généralement fournis avec les mesures expérimentales.

7.2.3 Consistance de chemin

La forme des lois de probabilité introduites dans le modèle YIJ suggère une notion de consistance basée sur les chemins dans un graphe d'interaction. Nous la formulons à présent dans le cadre introduit au chapitre 3.

Rappelons tout d'abord que l'on cherche à définir la consistance entre d'une part un graphe d'interaction $\mathcal{G} = (V, E, \rho)$ et un ensemble de mesures $\{\mu_1, \dots, \mu_r\}$. Le graphe \mathcal{G} est en tout généralité partiellement signé, et muni d'un ensemble U d'entrées. L'ensemble U décrit les espèces dont la variation dépend aussi de l'environnement. Il contient donc les éventuels gènes mutés, ainsi que toute espèce dont le niveau est influencé par l'environnement.

Nous appellerons consistance de chemin la situation où pour tout sommet (hors entrées), on peut trouver une entrée et un chemin de l'entrée audit sommet, tel que les signes du chemin et des variations soient compatibles. Plus précisément, à tout sommet i et toute expérience k , on associe une variable X_{ik} , et à toute régulation $j \rightarrow i$ la variable S_{ji} . Pour un sommet $i \in V \setminus U$, on considère l'ensemble Π_{ui} des chemins partant de $u \in U$ et arrivant à i . Soit $\pi_1 = u, \dots, \pi_{|\pi|} = i$ un tel chemin. On appelle signe du chemin le produit (dans l'algèbre des signes) $S_\pi = \prod_{j \in \{1, \dots, |\pi|-1\}} S_{\pi_j \pi_{j+1}}$. On définit $\mu(X_{ik}) = \mu_k(i)$ pour $i \in V$ et $k \in \{1, \dots, r\}$, et $\rho(S_{ji}) = \rho(j, i)$ pour $(j, i) \in \text{dom}(\rho)$.

Définition 12 (Consistance de chemin, \mathcal{P} -consistance). *On dit que \mathcal{G} et $M = \{\mu_1, \dots, \mu_r\}$ sont \mathcal{P} -consistants si la contrainte qualitative*

$$P_{\mathcal{G}}^M = \bigwedge_{i \in G \setminus U, k \in \{1, \dots, r\}} P_{ik}$$

admet une solution, avec

$$P_{ik} = \left(X_{ik} \approx \sum_{u \in U} \sum_{\pi \in \Pi_{ui}} S_\pi X_{uk} \right)$$

Commentons un peu cette définition. Si l'on souhaite modéliser une expérience de délétion d'un gène l , il suffit de choisir $U = \{l\}$ et poser $\mu(X_l) = -$. Si un sommet i de \mathcal{G} admet une variation non nulle suite à la délétion, alors il faut trouver un chemin π de l à i , tel que $S_\pi X_l = X_i$. Si la variation est nulle, alors soit i n'est pas accessible depuis l , soit on peut trouver deux chemins, l'un de contribution positive, l'autre de contribution négative³.

Examinons pour fixer les idées, l'exemple donné dans [102], représenté en figure 7.1. Le cas de gauche est bien \mathcal{P} -consistant : il suffit par exemple de poser $S_{AB} =$

³Pour être tout à fait précis, le modèle YIJ ne traite pas le cas des variations nulles.

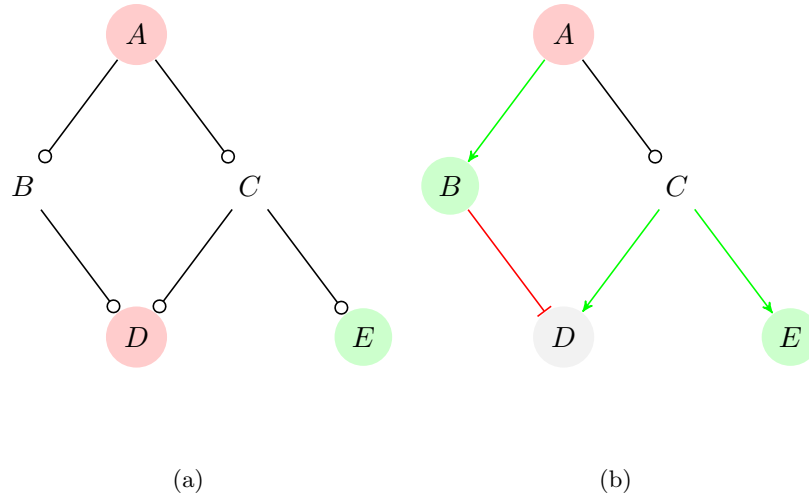


FIG. 7.1 – Illustrations pour la consistance de chemins. Le premier cas est présenté dans [102]. Les deux graphes ont une seule entrée : A .

$S_{BD} = S_{CE} = X_C = +$ et $S_{AC} = X_C = -$. En revanche, l'exemple de droite n'est pas \mathcal{P} -consistant : pour expliquer la variation nulle de D , il faudrait un chemin négatif et un chemin positif, ce qui contraint S_{AC} à $+$, mais alors il n'y a aucun chemin négatif de A à E .

7.2.4 Consistance au sommet et consistance de chemin

Nous sommes à présent en possession de deux notions de consistance, qu'il serait bon de pouvoir comparer. On peut en fait facilement prouver le résultat suivant :

Théorème 12. *Il n'y a pas d'inclusion entre \mathcal{N} -consistance et \mathcal{P} -consistance.*

Il suffit d'exhiber des contre-exemples, comme ceux donnés en figure 7.2. Le cas (a) est \mathcal{P} -consistant, mais pas \mathcal{N} -consistant ; on peut en effet trouver des chemins expliquant les variations de E et F à partir de la variation de A . Cependant, on ne peut localement expliquer les variations opposées de E et F par la seule variation de D . La compétition entre les voies $A \rightarrow B \rightarrow D$ et $A \rightarrow C \rightarrow D$ se résout en D , et ne se propage pas à sa descendance dans le graphe d'interaction. Dans le cas (b), les équations qualitatives sont effectivement vérifiées, mais il n'y a aucun chemin explicatif entre l'entrée A et les sommets B et C . Il semble en effet étrange que le système initialement à l'équilibre passe, sous l'effet d'une inhibition à un état où B et C sont augmentés. Les variations de B et C « s'autojustifient », alors qu'elles devraient découler d'une contrainte imposée sur les entrées.

Cette comparaison suggère que les deux critères de consistance sont *complémentaires* : la consistance au sommet assure la cohérence des variations et des régulations directes ; la consistance de chemins assure que les variations sont explicables par une sollicitation extérieure au système.

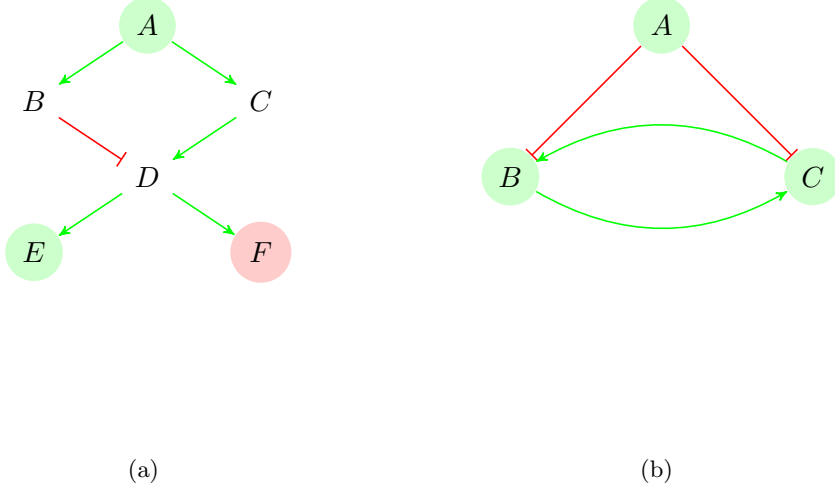


FIG. 7.2 – Contre-exemples pour la preuve du théorème 12. (a) Cas \mathcal{P} -consistant mais pas \mathcal{N} -consistant. (b) Cas \mathcal{N} -consistant mais pas \mathcal{P} -consistant. Les deux graphes ont une seule entrée : A .

7.2.5 Chemins et déplacement d'équilibre

Nous avons cherché une justification différentielle au critère de consistance par chemin, dans le cadre présenté au chapitre 3. Nous avons obtenu le résultat suivant, publié dans [83] et sous une forme raffinée dans [72].

Théorème 13. *Soit un système régi par une dynamique différentielle $\frac{dX}{dt} = F(X, U)$. Soit $\mathcal{G} = (V, E, \rho)$ le graphe d'interaction associé, et U l'ensemble de ses entrées. On note $\mathring{\mathcal{G}}$ le sous-graphe de \mathcal{G} privé de ses entrées, et \mathring{F} la restriction de F aux sommets de $\mathring{\mathcal{G}}$. Avec les hypothèses suivantes :*

- $H1$ est vérifiée (voir chapitre 3),
- \mathring{F} est non singulière,
- \mathring{F} est de la forme

$$\mathring{F}(X) = \Psi(\hat{X}, \mathring{X}) - \Lambda' \mathring{X}$$

où Λ est un vecteur de réels strictement positifs et Ψ est une fonction bornée vérifiant pour tout $i \in \mathring{\mathcal{G}}$,

$$\Psi_i(X_1, \dots, X_i = 0, \dots, \hat{X}) > 0$$

- $\mathring{\mathcal{G}}$ ne contient pas de boucle positive

Alors deux états d'équilibre stables X^1 et X^2 de F vérifient :

$$\text{sgn}(X_i^2 - X_i^1) = \sum_{u \in U} \sum_{\pi \in \Pi_{ui}^*} \prod_{k < |\pi|} \rho(\pi_k, \pi_{k+1}) \text{sgn}(X_k^2 - X_k^1) \quad (7.2)$$

où Π_{ui}^* est l'ensemble des chemins sans circuit de u à i tels que u est le seul sommet dans U .

Ce résultat diffère sur deux points avec le critère de consistance de chemin. D'une part, les contraintes sont plus fortes, à cause des conditions portant sur les chemins explicatifs. Ceux-ci doivent être sans circuit et ne jamais revenir à une entrée. D'autre part ses conditions d'application sont restreintes, puisque le résultat n'est valable que pour les graphes d'interaction *sans circuit positif*.

7.2.6 Bilan

Nous avons explicité l'utilisation dans les travaux de Yeang *et al* [103, 102] d'une notion de consistance, que nous avons appelée consistance de chemin. Nous avons montré que cette notion n'est pas comparable au critère étudié dans cette thèse. Elle lui apporte essentiellement le fait que toute variation s'explique, *via* une cascade de régulations, par une perturbation appliquée au système considéré.

Chapitre 8

Conclusion

8.1 Bilan

Problématique et approche suivie Le problème posé dans cette thèse porte sur l'analyse des données haut-débit par des modèles physiques des interactions cellulaires. Comme nous l'avons vu, cette question est compliquée par la nature des données disponibles qui portent sur un très grand nombre de variables, et sont le plus souvent fortement bruitées et/ou incomplètes.

Nous avons suivi une approche d'*abstraction qualitative* d'un modèle quantitatif, où les grandeurs mesurées sont remplacées par leur signe. De manière analogue à ce qui est proposé dans [23] et [36], nous dérivons du modèle quantitatif sous-jacent des relations qui sont abstraites en contraintes purement discrètes. Ces contraintes, appelées « critère de consistance », sont la base d'une comparaison entre la description du système (le graphe d'interaction) et les données disponibles.

Ce critère de consistance s'inscrit dans une démarche générale d'analyse de données, qui comporte quatre étapes (voir aussi figure 1.1) :

1. une phase de vérification, où le critère est utilisé pour décider de la compatibilité entre les données et le modèle ;
2. une phase de diagnostic/correction en cas d'incompatibilité, consistant à déterminer des causes possibles pour la non consistance, puis à modifier le modèle ou les données ;
3. une phase de prédiction, quand le modèle et les données sont consistantes, où le critère est utilisé pour déduire le comportement du système ;
4. enfin une phase de conception d'expériences, permettant de cibler l'acquisition de nouvelles données.

Nous avons étudié les trois premières phases de cette démarche, que nous avons formalisées comme des problèmes de contraintes sur domaines finis. Nous avons apporté plusieurs réponses algorithmiques à ces problèmes, qui ont été validées par des applications sur données réelles.

Modélisation discrète/modélisation quantitative L'objectif de ce travail est de formuler un modèle qualitatif adapté au volume et à la qualité des données existantes. Il s'agit de fournir une technique d'analyse des mesures globales (transcriptome, protéome, métabolome *etc*) dans les conditions techniques prévisibles à court et moyen terme – c'est-à-dire un contexte où les mesures haut-débit sont disponibles en nombre limité, et très peu répliquées. Il ne s'agit pas d'un bon outil pour l'analyse ciblée d'un mécanisme ; en comparaison, les modèles quantitatifs à base d'équations différentielles ou de processus stochastiques sont de bien meilleures descriptions.

Une question sous-jacente à ce travail est celle de l'application des méthodes quantitatives comme les équations différentielles ordinaires ou les réseaux bayésiens à l'étude des données haut-débit. Ces modèles requièrent l'estimation d'un nombre important de paramètres réels, et nécessitent de ce fait une masse de données très importante, qui n'est que rarement disponible en pratique. Nul doute que cette difficulté n'est que temporaire, et que les techniques expérimentales progressant, la disponibilité des données ne sera plus un problème. Nous voulons en revanche attirer l'attention sur une question plus épineuse : l'exploration de l'espace des paramètres dans les approches différentielles et probabilistes est un problème particulièrement difficile sur le plan calculatoire. Elle repose le plus souvent sur des simulations ou sur des problèmes d'optimisation non convexes, qui sont particulièrement coûteux en temps de calcul et qui n'offrent en général aucune garantie d'exactitude.

L'approche que nous avons développée repose sur une modélisation discrète ; les données bruitées ou les paramètres inconnus sont représentés par leur signe, et les relations entre eux sont abstraites en contraintes à vérifier. L'étude et la résolution de ces contraintes sont des problèmes beaucoup mieux maîtrisés, pour lesquels nous avons exhibé des algorithmes exacts. Ces algorithmes permettent une exploration exhaustive de l'espace des paramètres ou des données manquantes, et fournissent des *preuves* des propriétés trouvées.

Critère de consistance La première contribution de ce travail porte sur la modélisation d'un formalisme adapté au traitement des données haut-débit. Nous l'avons vu, le résultat est un compromis entre propriétés calculatoires, disponibilité des données et précision de la description. Notre modèle repose sur deux types d'objets : d'une part un graphe recensant les espèces chimiques présentes dans le système (gène, protéine, *etc*) ainsi que les influences (positives ou négatives) des uns sur les autres ; d'autre part, un étiquetage des sommets du graphe, indiquant la variation du niveau des espèces entre deux mesures. Le critère de consistance que nous avons introduit stipule que la variation de chaque sommet doit être expliquée par l'une au moins des influences qu'il reçoit.

Nous avons formalisé cette règle intuitive en utilisant l'algèbre des signes, et montré comment associer à un graphe d'interaction et des données une contrainte définissant leur compatibilité. Nous avons ensuite démontré formellement la validité de cette contrainte dans un cadre différentiel. Cette démarche nous a notamment permis de cerner les conditions d'applicabilité de notre critère de consistance.

Résolution de contraintes qualitatives En nous appuyant sur notre critère de consistance, nous avons formulé de manière précise des problèmes liés à la vérification de la compatibilité données/modèles, à la prédiction des variables non observées et au diagnostic des inconsistances. Ces problèmes se ramènent à la résolution et à l'étude de contraintes booléennes exprimant des relations dans l'algèbre des signes. La deuxième contribution de ce travail concerne la résolution de ces contraintes, pour laquelle nous avons proposé deux approches.

La première utilise une structure de données appelée diagramme de décision pour représenter l'ensemble des solutions des contraintes booléennes. Une fois construit, un parcours récursif du diagramme permet d'obtenir une grande variété d'informations sur l'ensemble des solutions. Nous avons détaillé la construction du diagramme, ainsi qu'une série d'algorithmes répondant efficacement aux problèmes posés. La principale limite de cette approche se situe au niveau de la construction du diagramme, dont la taille est au pire exponentielle en fonction du nombre de variables de la contrainte. Ainsi l'utilisation directe des diagrammes de décision se limite à des contraintes d'au plus quelques centaines de variables. Pour cette raison, nous avons développé des méthodes de réduction et de décomposition des contraintes qualitatives permettant d'étendre très sensiblement leur applicabilité.

La deuxième approche fait appel à des techniques de résolution issues de la programmation logique. Nous avons montré comment construire un programme logique dont les modèles sont exactement les solutions de la contrainte qualitative. En nous appuyant sur les moteurs de résolution ASP, nous avons ainsi obtenu un algorithme particulièrement efficace pour la résolution des contraintes qualitatives.

Nous avons également vu que ces deux techniques ne sont pas forcément adaptées à tous les problèmes introduits. Le tableau 8.1 récapitule ces différences. En voici les grandes lignes :

- le problème de vérification est nettement mieux résolu par l'approche programmation logique. Dans nos expériences sur la levure par exemple, on a pu travailler sur un réseau de plus de 2000 sommets et 4000 arcs non signés, et quelques dizaines de mesures. Pour le problème de vérification, cela signifie résoudre une contrainte de plusieurs dizaines de milliers de variables. Il serait particulièrement difficile (notamment pour le choix de la décomposition) d'obtenir des performances similaires à partir des diagrammes de décision.
- la recherche des invariants est du coup également plus efficace avec l'approche programmation logique, grâce à la recherche de contre-exemples.
- en revanche, les diagrammes de décision sont clairement supérieurs dès que – ce n'est pas une surprise – il est nécessaire d'énumérer ou de compter les solutions. C'est le cas notamment avec le calcul des probabilités marginales.
- les diagrammes de décision sont également incontournables quand les contraintes qualitatives contiennent plusieurs alternances de quantificateurs existentiels et universels. Nous l'avons mentionné lors de la recherche des signes inférables au paragraphe 6.2.1.

D'un point de vue modélisation, il faut souligner que l'utilisation de programmes logiques est d'une souplesse bien supérieure à celle des diagrammes de décision. Certes

Problème	BDD	ASP
Vérification sous consistance aux sommets	••	•••
Invariants d'une contrainte	••	•••
Probabilités marginales	•	
Paramètres inférables	••	
Diagnostic	•	••

TAB. 8.1 – Récapitulatif des problèmes formulés dans la thèse, et des algorithmes proposés

la sémantique des modèles stables joue parfois des tours au programmeur débutant (ou distrait !), mais il nous semble nettement plus simple et plus sûr d'expérimenter de cette façon de nouvelles idées, et ce d'autant plus que le résultat final est – au moins avec l'habitude – particulièrement lisible.

Validation sur données réelles La troisième et dernière contribution de cette thèse est un premier pas vers la validation expérimentale de notre approche. Nous avons notamment travaillé sur l'application des notions de vérification et de prédiction à des données réelles sur la bactérie *E. coli* et sur la levure. Dans une première expérience, nous avons étudié la réponse transcriptionnelle globale d'*E. coli* à un stress nutritionnel. Nous avons confirmé sur cet exemple la validité du critère de consistance, et montré comment obtenir des corrections non triviales de notre modèle du réseau transcriptionnel. Notre seconde application porte sur la reconstruction de réseaux transcriptionnels à partir de données d'expression. Nous avons étudié un cas particulier de ce problème, où les régulations sont connues, mais pas leur effet. Nous avons montré, par une étude préliminaire sur le réseau d'*E. coli* qu'un nombre limité de mesures (moins de 30) permet de déterminer une fraction raisonnable des régulations (de 15 à 40% environ). L'application de cette approche pour l'intégration de données chIP-on-chip et données d'expression chez la levure a confirmé ces estimations en fournissant des prédictions que nous avons partiellement validées.

8.2 Perspectives

Le travail présenté dans cette thèse peut être selon nous prolongé selon trois axes, que nous détaillons maintenant.

Nouvelles notions de consistance La première suite que nous suggérons concerne l'étude de nouvelles notions de consistance entre données et modèle. Il s'agit notamment de systématiser le genre de comparaison esquissée à la section 7.2. Un point de départ consisterait à éclaircir la notion de chemin explicatif et à l'incorporer dans le critère de consistance utilisé dans cette thèse. Nos études préliminaires dans ce sens montre que la notion de consistance qui en résulte contraint significativement plus les données et peut être efficacement traitée par les techniques de programmation logique utilisées au

chapitre 5. D'autres critères de consistance peuvent provenir d'une modélisation plus spécifique des réseaux étudiés. Les travaux de Gutiérrez [34] peuvent être vus comme un exemple de spécialisation des contraintes entre un sommet et ses prédécesseurs dans le cas des réseaux génétiques. De manière analogue, un rapprochement avec les techniques de flux à l'équilibre dans les réseaux métaboliques pourrait être envisagé, offrant ainsi un formalisme d'étude grande échelle des mécanismes génétiques sur le métabolisme.

Traitement des données bruitées La contribution majeure de ce travail nous semble être d'avoir exhibé une règle simple qui est généralement vérifiée dans les systèmes biologiques, mais souvent mise en défaut dans les données expérimentales. Ce double constat fait du critère de consistance que nous avons proposé un guide précis pour l'analyse de données, à condition de traiter correctement les défauts trouvés. Il nous apparaît pour cette raison essentiel d'approfondir notre travail sur la partie diagnostic/correction de la démarche présentée en figure 1.1. Il convient notamment de formaliser et de systématiser les approches utilisées au chapitre 6.

L'une des pistes, qui a été proposée dans le chapitre 4 mais non validée sur données réelles, consiste à s'appuyer sur la notion de diagnostic : si une série de mesures est incompatible avec un graphe donné, on calcule le nombre minimal de modifications (du graphe et des données) permettant de vérifier le critère de consistance. Ces modifications minimales constituent ce que nous avons appelé des diagnostics de l'inconsistance. Nous pensons que les *invariants de l'ensemble des diagnostics* peuvent constituer des corrections particulièrement fiables pour les données et/ou le modèle. Cette approche pourrait notamment être validée par l'une des applications, à savoir la reconstruction de graphe d'interaction à partir de données.

Nous avons à plusieurs reprises souligné que l'interprétation des variations faibles est difficile en pratique : soit on ne considère que les fortes variations, en perdant une partie de l'information ; soit on considère aussi les faibles variations, quitte à rajouter du bruit. Même en choisissant un compromis, nous avons observé un grand nombre de défauts lors de l'étude des données d'expression. Beaucoup trop, en tout cas, pour espérer les corriger tous, ce qui pourtant est nécessaire pour prédire les variables non observées. La solution que nous envisageons consiste à introduire une notion d'*invariant sous correction minimale*, permettant de proposer des prédictions même dans le cas où les données ne sont pas compatibles avec le modèle. Dit autrement, on s'intéresse à l'intersection des prédictions des modèles corrigés.

Ces propositions visent à formaliser l'interprétation des données bruitées. Elles nécessitent avant même une étude algorithmique poussée, une validation à petite échelle sur données réelles.

Plans d'expérience Le dernier axe que nous voudrions mentionner concerne la conception d'expériences, que nous n'avons pas abordée dans ce travail. Brièvement, on peut distinguer deux tâches. La première porte sur le *contrôle des systèmes étudiés*. Étant donné un ensemble d'entrées, voire un nombre limité de modifications permises

du graphe d'interaction, comment provoquer à coup sûr une variation donnée ? Il s'agit d'un problème difficile puisqu'il nécessite de trouver des valeurs pour certaines variables d'une contrainte, telle que d'autres variables deviennent invariantes. Les outils développés aux chapitres 4 et 5 peuvent fournir des solutions algorithmiques abordables. Notamment, il semble possible de construire à l'aide des diagrammes de décision – et pour des systèmes de taille raisonnable – la fonction qui à une valeur des entrées associe l'ensemble des variables invariantes du système. Côté programmation logique, ce type de problème doit pouvoir être abordé à l'aide de programmes disjonctifs [33]. La deuxième tâche que nous identifions concerne la *discrimination de modèles*. Nous avons vu que les graphes d'interactions peuvent être partiellement connus. L'existence de certains arcs peut être incertaine, ou le signe de certaines régulations peut manquer. Nous avons montré comment travailler malgré l'incertitude sur le modèle réel, et raisonner sur toutes les valeurs des variables non observées. Un problème intéressant consisterait à déterminer, parmi un ensemble d'expériences techniquement réalisables, celles qui discrimine le plus efficacement parmi les modèles admissibles. Idéalement, les expériences proposées devraient, quelle que soit leur issue, invalider le plus grand nombre possible de modèles.

Annexe A

Inférence de l'effet des facteurs de transcription sur leurs gènes cibles

Nous reproduisons ici un article non publié portant sur un cas particulier de reconstruction de modèle à partir de données expérimentales. Nous supposons connus les arcs du graphe d'interaction, mais pas leur signe. L'objectif est d'utiliser des données de variation pour en déduire l'effet des régulations (activation ou inhibition). Cet article complète le chapitre de validation expérimentale de notre approche et plus précisément le résumé donné à la section 6.2

Inferring the role of transcription factors in regulatory networks

P. Veber^a, C. Guziolowski^a, M. Le Borgne^b, O. Radulescu^{a,c}, A. Siegel^d

^a Centre INRIA Rennes Bretagne Atlantique, IRISA, Rennes, France ^b Université de Rennes 1, IRISA, Rennes, France ^c, Université de Rennes 1, IRMAR, Rennes, France ^d CNRS, UMR 6074, IRISA, Rennes, France

ABSTRACT

Background Expression profiles obtained from multiple perturbation experiments are increasingly used to reconstruct transcriptional regulatory networks, from well studied, simple organisms up to higher eukaryotes. Admittedly, a key ingredient in developing a reconstruction method is its ability to integrate heterogeneous sources of information, as well as to comply with practical observability issues: measurements can be scarce or noisy. The purpose of this work is (1) to build a formal model of regulations among genes; (2) to check its consistency with gene expression data on stress perturbation assays; (3) to infer the regulatory role of transcription factors as inducer or repressor if the model is consistent with expression profiles; (4) to isolate ambiguous pieces of information if it is not.

Results We validate our methods on *E. Coli* network with a compendium of expression profiles. We investigate the dependence between the number of available expression profiles and the number of inferred regulations, in the case where all genes are observed. This is done by simulating artificial observations for the transcriptional network of *E. Coli* (1529 nodes and 3802 edges). We prove that at most 40,8% of the network can be inferred and that 30 distinct expression profiles are enough to infer 30% of the network on average. We repeat this procedure in the case of missing observations, and show that our approach is robust to a significant proportion of unobserved genes. Finally, we apply our inference algorithms to *S. Cerevisiae* transcriptional network, and demonstrate that for small scale subnetworks of *S. Cerevisiae* we are able to infer more than 20% of the regulations. For more complex networks, we are able to detect and isolate inconsistencies between experimental sources and a non negligible portion of the model (15% of all interactions).

Conclusions Our approach does not require accurate expression levels, nor times series. Nevertheless, we show both on real and artificial data that a relatively small number of perturbation experiments are enough to determine a significant portion of regulatory effects. This is a key practical asset compared to statistical methods for network reconstruction. In addition, we illustrate the capability of our method to validate networks. We conjecture that inconsistencies we detected might be good candidates for further experimental investigations.

Contact philippe.veber@irisa.fr

1 INTRODUCTION

A central problem in molecular genetics is to understand the transcriptional regulation of gene expression. A transcription factor (TF) is a protein that binds to a typical domain on the DNA and influences

transcription. Depending on the type of binding site, on the distance to the coding regions and on the presence of other molecules that also bind to the DNA, the effect can either be a repression or an activation of the transcription. Finding which gene is controlled by which TF is a reverse engineering problem, usually named *network reconstruction*. This question has been approached over the past years by various groups.

A first approach to achieve this task consists in expanding information spread in the primary literature. A number of important databases that take protein and regulatory interactions from the literature and curate them have been developed [1, 2, 3, 4, 5]. For the bacteria *E. Coli*, RegulonDB is a dedicated database that contains experimentally verified regulatory interactions [6]. For the budding yeast (*S. Cerevisiae*), the Yeast Proteome Database contains amounts of regulatory information [7]. Even in this latter case, the amount of available information is not sufficient to build a reasonably accurate model of transcriptional regulation. It is nevertheless an unavoidable starting point for network reconstruction.

The alternative to the literature-curated approach is a data-driven approach. This approach is supported by the availability of high-throughput experimental data, including microarray expression analysis of deletion mutants (simple or more rarely double non-lethal knockouts), over expression of TF-encoding genes, protein-protein interactions, protein localization or chIP-chip experiments coupled with promoter sequence motifs. We may cite several classes of methods: perturbations and knock-outs, microarray analysis of promoter binding (chIP-chip), sequence analysis, various microarray expression data analysis such as correlation, mutual information or causality studies, Bayesian networks, path analysis, information-theoretic approaches and ordinary differential equations [8, 9, 10].

In short, most available approaches so far are based on a probabilistic framework, which defines a probability distribution over the set of models. Then, an optimization algorithm is applied in order to determine the most likely model given by the data. Due to the size of the inferred model, the optimal model may be a local but not a global optimal. Hence, errors can appear and no consensual model can be produced. As an illustration, a special attention has been paid to the reconstruction of *S. Cerevisiae* network from chIP-chip data and protein-protein interaction networks [11]. A first regulatory network was obtained with promoter sequence analysis methods [12, 13]. Non-parametric causality tests proposed some previously undetected transcriptional regulatory motifs [14]. Bayesian analysis also proposed transcriptional networks [15, 16]. Though, the results obtained with the different methods do not coincide and a fully data-driven search is in general subject to overfitting and to unifiability [17].

In regulatory networks, an important and nontrivial physiological information is the regulatory role of transcription factors as inducer of repressor, also called *the sign of the interaction*. This information is needed if one wants to know for instance the physiological effect of a change of external conditions or simply deduce the effect of a perturbation on the transcription factor. While this can be achieved for one gene at a time with (long and expensive) dedicated experiments, probabilistic methods such as Bayesian models [18] of path analysis [19, 20] are capable to propose models from high-throughput experimental data. However, as for the network reconstruction task, these methods are based on optimization algorithms to compute an optimal solution with respect to an interaction model.

In this paper, we propose to use formal methods to compute the sign of interactions on networks for which a topology is available. By doing so, we are also capable of validating the topology of the network. Roughly, expression profiles are used to constrain the possible regulatory roles of transcription factor, and we report those regulations which are assigned the same role in all feasible models. Thus, we over-approximate the set of *feasible* models, and then look for *invariants* in this set. A similar idea was used in [21] in order to check the consistency of gene expression assays. We use a deeper formalisation and stronger algorithmic methods in order to achieve the inference task.

We use different sources of large-scale data: gene expression arrays provide indications on signs of interactions. When not available, ChIP-chip experiments provide a sketch for the topology of the regulatory network. Indeed, microarray analysis of promoter binding (ChIP-chip) is an experimental procedure to determine the set of genes controlled by a given transcription factor in given experimental conditions [22]. A particularly interesting feature of this approach is that it provides an *in vivo* assessment of transcription factor binding. On the contrary, testing affinity of a protein for a given DNA segment *in vitro* often results in false positive binding sites.

The main tasks we address are the following:

1. Building a formal model of regulation for a set of genes, which integrates information from ChIP-chip data, sequence analysis, literature annotations;
2. Checking its consistency with expression profiles on perturbation assays;
3. Inferring the regulatory role of transcription factors as inducer or repressor if the model is consistent with expression profiles;
4. Isolating ambiguous pieces of information if it is not.

Both, probabilistic approaches and our formal approach mainly aim to deal with incomplete knowledge and experimental noise. However, statistical methods usually require a minimal number of samples (about a hundred), because they explicitly model the distribution of experimental noise. In practice it is feasible but very costly to obtain enough expression profiles to apply them. In contrast, our approach may be used even with less perturbation experiments (some tens) at hand, which makes it a suitable alternative when statistical methods cannot be applied.

Additionally, since our predictions are consensual with *all* profiles and since they are not based on heuristics, our methods are well

designed to validate networks inferred with probabilistic methods, and eventually identify the location of inconsistencies.

The paper is organized as follows. Sec. 2 briefly introduces the mathematical framework which is used to define and to test the consistency between expression profiles and gene networks. In Sec. 3 we apply our algorithms to address three main issues.

- We illustrate and validate our formal method on the transcriptional network of *E. Coli* (1529 nodes and 3802 edges), as provided in RegulonDB [6], together with a compendium of expression profiles [9]. We identified 20 inconsistent edges in the graph.
- We investigate the dependence between the number of available observations and the number of inferred regulations, in the case where all genes are observed. This is done by simulating artificial observations for the transcriptional network of *E. Coli*. We prove that at most 40,8% of the network can be inferred and that 30 perturbation experiments are enough to infer 30% of the network on average. By studying a reduced network, we also comment about the complementarity between our approach and detailed analysis of times series using dynamical modeling.
- We repeat this procedure in the case of missing observations, and estimate how the proportion of unobserved genes affects the number of inferred regulations. With these two situations we also demonstrate that our approach is able to handle networks containing thousands of genes, with several hundreds of observations.
- We apply our inference algorithms to *S. Cerevisiae* transcriptional network, in order to assess their relevance in real conditions. We demonstrate that for small scale subnetworks of *S. Cerevisiae* we are able to infer more than 20% of the roles of regulations. For more complex networks, we are able to detect and isolate inconsistencies (also called ambiguities) between expression profiles and a quite important part of the model (15% of all the interactions).

The last two sections discuss the results we obtained, and give more details on the algorithmic procedures.

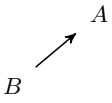
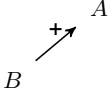
2 APPROACH

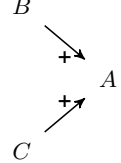
2.1 Detecting the sign of a regulation and validating a model

Our goal is to determine the regulatory action of a transcription factor on its target genes by using expression profiles. Let us illustrate our purpose with a simple example.

We suppose that we are given the topology of a network (this topology can be obtained from ChIP-chip data or any computational network inference method). In this network, let us consider a node A with a single predecessor. In other words, the model tells that the protein B acts on the production of the gene coding for A and no other protein acts on A .

Independently, we suppose that we have several gene expression arrays at our disposal. One of these arrays indicates that A and B simultaneously increase during a steady state experiment. Then, the *common sense* says that B must have been as activator of A during the experiment. More precisely, protein B cannot have inhibited

Model	Expression profiles	Prediction
	B increases C increases	The action from B to A is an activation.
	B increases C decreases	Model and data are ambiguous (also called <i>incompatible</i>).

Model	Expression profiles	Prediction
	B decreases C decreases	A decreases

gene A , since they both have increased. We say that the model predicts that the sign of the interaction from B to A is positive.

This naive rule is actually used in a large class of models, we will call it the *naive inference rule*. When several expression profiles are available, the predictions of the different profiles can be compared. If two expression profiles predict different signs for a given interaction, there is a *ambiguity* or *incompatibility* between data and model. Then, the ambiguity of the regulatory role can be attributed to three factors: (1) a complex mechanism of regulation: the role of the interaction is not constant in all contexts, (2) a missing interaction in the model, (3) an error in the experimental source.

Algorithm: Naive Inference algorithm

Input:

A network with its topology
 A set of expression profiles

Output:

a set of predicted signs
 a set of ambiguous interactions

For all Node A with exactly one predecessor B

if A and B are observed simultaneously **then return**

$prediction\ sign(B \rightarrow A) = sign(A) * sign(B)$

if $sign(B \rightarrow A)$ was predicted different by another expression profile **then return** Ambiguous arrow $B \rightarrow A$

Let us consider now the case when A is activated by two proteins B and C . No more natural deduction can be done when A and B increase during an experiment, since the influence of C must be taken into account. A *model* of interaction between A , B and C has to be proposed. Probabilistic methods estimate the most probable signs of regulations that fit with the theoretical model [18, 23].

Our point of view is different: we introduce a *basic rule* that shall be checked by every interactions. This rule tells that **any variation of A must be explained by the variation of at least one of its predecessors**. Biologically, this assumes that the nature of differential gene expression of a given gene is likely to affect the differential expression in other genes. Even if this is not universally true, this can be viewed as a crude approximation of the real event. In previous papers, we introduced a formal framework to justify this basic rule under some reasonable assumptions. We also tested the consistency between expression profiles and a graphical model of cellular interactions. This formalism will be here introduced in an informal way ;

its full justification and extensions can be found in the references [24, 25, 26, 27].

In our example, the basic rule means that if B and C activate A , and both B and C are known to decrease during a steady state experiment, A cannot be observed as increasing. Then A is *predicted* to decrease. More generally, in our approach, we use the rule as a constraint for the model. We write constraints for all the nodes of the model and we use several approaches in order to solve the system of constraints. From the study of the set of solutions, we deduce which signs are surely determined by these rules. Then we obtain *minimal obligatory conditions* on the signs, instead of *most probable signs* given by probabilistic methods. Notice that by construction, our constraints coincide with probabilistic models in the predictions of the naive inference algorithm.

2.2 A formal approach

Consider a system of n chemical species $\{1, \dots, n\}$. These species interact with each other and we model these interactions using an *interaction graph* $G = (V, E)$. The set of nodes is denoted by $V = \{1, \dots, n\}$. There is an edge $j \rightarrow i \in E$ if the level of species j influences the production rate of species i . Edges are labeled by a sign $\{+, -\}$ which indicates whether j activates or represses the production of i .

In a typical stress perturbation experiment a system leaves an initial steady state following a change in control parameters. After waiting long enough, the system may reach a new steady state. In genetic perturbation experiments, a gene of the cell is either knocked-out or overexpressed; perturbed cells are then compared to wild cells. Most high-throughput measurements provide the ratio between initial and final levels, like in expression arrays for instance. In many experimental settings, the numerical value is not accurate enough to be taken "as it is". The noise distribution may be studied if enough measurements are available. Otherwise, it is safer to rely only on a qualitative feature, such as the order of magnitude, or the *sign of the variation*. Let us denote by $sign(X_i) \in \{+, -, 0\}$ the sign of variation of species i during a given perturbation experiment, and by $sign(j \rightarrow i) \in \{+, -\}$ the sign of the edge $j \rightarrow i$ in the interaction graph.

Let us fix species i such that there is no positive self-regulating action on i . For every predecessor j of i , $sign(j \rightarrow i) * sign(X_j)$ provides the sign of the *influence* of j on the species i . Then, we can write a constraint on the variation to interpret the rule previously stated: *the variation of species i is explained by the variation of at least one of its predecessors in the graph*.

$$sign(X_i) \approx \sum_{j \rightarrow i} sign(j \rightarrow i) sign(X_j). \quad (1)$$

When the experiment is a genetic perturbation the same equation stands for every node that was not genetically perturbed during the

experiment and such that all its predecessors were not genetically perturbed. If a predecessor X_M of the node was knocked-out, the equation becomes

$$\text{sign}(X_i) \approx -\text{sign}(M \rightarrow i) + \sum_{j \rightarrow i, j \neq M} \text{sign}(j \rightarrow i) \text{sign}(X_j). \quad (2)$$

The same holds with $+\text{sign}(M \rightarrow i)$ when the predecessor X_M was overexpressed. There is no equation for the genetically perturbed node.

The *sign algebra* is the suitable framework to read these equations [26]. It is defined as the set $\{+, -, ?, \mathbf{0}\}$, provided with a sign compatibility relation \approx , and arithmetic operations $+$ and \times . The following tables describe this algebra:

$$\begin{array}{cccccc} ++ = ? & +++ = + & +- = - & +\times = - & +\times = + & -\times = + \\ ++\mathbf{0} = + & \mathbf{0} + \mathbf{0} = \mathbf{0} & - + \mathbf{0} = - & +\times \mathbf{0} = \mathbf{0} & \mathbf{0} \times \mathbf{0} = \mathbf{0} & -\times \mathbf{0} = \mathbf{0} \\ ? + = ? & ? + + = ? & ? + ? = ? & ? \times = ? & ? \times + = ? & ? \times ? = ? \\ ? + \mathbf{0} = ? & ? \times \mathbf{0} = \mathbf{0} & & & & \end{array}$$

$$+ \not\approx - \quad + \approx \mathbf{0} \quad - \approx \mathbf{0} \quad ? \approx + \quad ? \approx - \quad ? \approx \mathbf{0}$$

Even if the sign compatibility relation \approx provides a rule for the $\mathbf{0}$ value, we are not able to infer with our approach regulations of sign $\mathbf{0}$. This limitation is because the sign of an arrow in an interaction graph is only restricted to be $\{+, -\}$, thus we do not generate an equation for products which have no variation during a specific experiment.

For a given interaction graph G , we will refer to the *qualitative system* associated to G as the set made up of constraint (1) for each node in G . We say that node variations $X_i \in \{+, -, \mathbf{0}\}$ are *compatible* with the graph G when they satisfy all the constraints associated to G using the sign compatibility relation \approx .

With this material at hand, let us come back to our original problem, namely to infer the regulatory role of transcription factors from the combination of heterogeneous data. In the following we assume that :

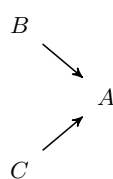
- The interaction graph is either given by a model to be validated, or built from chIP-chip data and transcription factor binding site searching in promoter sequences. Thus, as soon as a transcription factor j binds to the promoter sequence of gene i , j is assumed to regulate i . This is represented by an arrow $j \rightarrow i$ in the interaction graph.
- The regulatory role of a transcription factor j on a gene i (as inducer or repressor) is represented by the variable S_{ji} , which is constrained by Eqs. (1) or (2).
- Expression profiles provide the sign of the variation of the gene expression for a set of r steady-state perturbation or mutant experiments. In the following, x_i^k will stand for the sign of the observed variation of gene i in experiment k .

Our inference problem can now be stated as finding values in $\{+, -\}$ for S_{ji} , subject to the constraints :

$$\begin{array}{l} \text{for all } (1 \leq i \leq n), (1 \leq k \leq r), \\ i \text{ not genetically perturbed in the } k\text{-th experiment} \\ \left\{ \begin{array}{l} x_i^k \approx \sum_{j \rightarrow i} S_{ji} x_j^k \text{ if no genetic perturbations on all nodes } j \\ x_i^k \approx -S_{Mi} + \sum_{j \rightarrow i, j \neq l} S_{ji} x_j^k \text{ if knocked-out node } M \\ x_i^k \approx S_{Mi} + \sum_{j \rightarrow i, j \neq l} S_{ji} x_j^k \text{ if overexpressed node } M \end{array} \right. \quad (3) \end{array}$$

Most of the time, this inference problem has a huge number of solutions. However, some variables S_{ji} may be assigned the same value in *all* solutions of the system. Then, the recurrent value assigned to S_{ji} is a logical consequence of the constraints (3), and a prediction of the model. We will refer to these inferred interaction signs as *hard components* of the qualitative system, that is, sign variables S_{ji} that have the same value in all solutions of a qualitative system (3). When the inference problem has *no solution*, we say that the model and the data are *inconsistent* or *ambiguous*.

Let us illustrate this formulation on a very simple (yet informative) example. Suppose that we have a system of three genes A, B, C , where B and C influence A . The graph is shown in Table 1. Let us say that for this interaction graph we obtained six experiments, in each of them the variation of all products in the graph was observed (see Table 1). Using some or all of the experiments provided in Table 1 will lead us to a different qualitative system, as shown in Table 2, hence to different inference results. The process of inference for this example can be summarized as follows: starting from a set of experiments we generate the qualitative system of equations for our graph, studying its compatibility we will be able to set values for the signs of the regulations (edges of the interaction graph), but only we will infer a sign if in all solutions of the system the sign is set to the same value. Following with the example, in Table 2 we illustrate this process showing how the set of inferred signs of regulation varies with the set of experiments provided.



Stress perturbation expression profile	x_A	x_B	x_C
e_1	+	+	+
e_2	+	+	-
e_3	-	+	-
e_4	-	-	-
e_5	-	-	+
e_6	+	-	+

Table 1. Interaction graph of three genes A, B, C , where B and C influence A . Table with the variation of genes A, B , and C observed in six different stress perturbation experiments.

2.3 Algorithmic procedure

When the signs on edges are known (i.e. fixed values of S_{ji}) finding compatible node variations X_i is a NP-complete problem [26]. When the node variations are known (i.e. fixed values of X_i) finding the signs of edges S_{ji} from X_i can be proven NP-complete in a very similar way. Though, we have been able to design algorithms that perform efficiently on a wide class of regulatory networks. These algorithms predict signs of the edges when the network topology and

the expression profiles are compatible. In case of incompatibility, they identify ambiguous motifs and propose predictions on parts of the network that are not concerned with ambiguities.

The general process flow is the following (see Sec. 6 for details):

Step 1 Sign Inference

Divide the graph into motifs (each node with its predecessors). For each motif, find sign valuations (see Algorithm 1 in Sec.6) that are compatible with all expression profiles. If there are no solutions, call the motif *Multiple Behaviors Module* and remove it from the network.

Solve again the remaining equations and determine the edge signs that are fixed to the same value in all the solutions. These fixed signs are called *edge hard components* and represent our predictions.

Step 2 Global test/correction of the inferred signs

Solutions at previous step are not guaranteed to be global. Indeed, two node motifs at step 1 can be compatible separately, but not altogether (with respect to all expression profiles). This step checks global compatibility by solving the equations for each expression profile. New *Multiple Behavior Modules* can be found and removed from the system.

Step 3 Extending the original set of observations

Once all conflicts removed, we get a set of solutions in which signs are assessed to both nodes and edges. *Node hard components*, representing inferred gene variations can be found in the same way as we did for edges. We add the new variations to the set of observations and return to step 1. The algorithm is iterated until no new signs are inferred.

Step 4 Filtering predictions

In the incompatible case, the validity of the predictions depends on the accuracy of the model and on the correct identification of the MBMs. The model can be incomplete (missing interactions), and MBMs are not always identifiable in a unique way. Thus, it is useful to sort predictions according to their reliability. Our filtering parameter is a positive integer k representing the number of different experiments with which the predicted sign is compatible. For a

filtering value k , all the predictions that are consistent with less than k profiles are rejected.

The inference process then generates three results:

1. *A set of multiple behavior modules (MBM)*, containing interactions whose role was unclear and generated incompatibilities. We have identified several types of MBMS:
 - *Modules of Type I*: these modules are composed of several direct regulations of the same gene. These modules are detected in the Step 1 of the algorithm. Most of the MBMs of Type I are made of only one edge like illustrated in Fig. 1, but bigger examples exist.
 - *Modules of Type II, III, IV*: these modules are detected in Steps 2 or 3, hence, they contain either direct regulations from the same gene or indirect regulations and/or loops. Each of these regulations represent a consensus of all the experiments, but when we attempt to assess them globally, they lead to contradictions. The indices II-IV have no topological meaning, they label the most frequent situations illustrated in Fig. 1.
2. *A set of inferred signs*, meaning that all expression profiles fix the sign of an interaction in a unique way.
3. *A reliability ranking of inferred signs*. The filtering parameter k used for ranking is the number of different expression profiles that validate a given sign.

On computational ground, the division between Step 1 (which considers each small motif with all profiles together) and Step 2 (which considers the whole network with each profile separately) is necessary to overcome the memory complexity of the search of solutions. To handle large-scale systems, we combine a model-checking approach by decision diagrams and constraint solvers (see details in Sec. 6).

Since our basic rule is a crude approximation of real events, we expect it to produce very robust predictions. On the other hand, a regulatory network is only a rough description of a reaction network. For certain interaction graphs, not a single sign may be inferred even

Experiments used	Qualitative system	Replacing values from experiments	Compatible solutions (S_{BA}, S_{CA})	Inferred signs (identical in all solutions)
$\{e_1\}$	$x_A^1 \approx S_{BA}x_B^1 + S_{CA}x_C^1$	$(+) \approx S_{BA} \times (+) + S_{CA} \times (+)$	$(+, +)$ $(+, -)$ $(-, +)$	\emptyset
$\{e_1, e_2\}$	$x_A^1 \approx S_{BA}x_B^1 + S_{CA}x_C^1$ $x_A^2 \approx S_{BA}x_B^2 + S_{CA}x_C^2$	$(+) \approx S_{BA} \times (+) + S_{CA} \times (+)$ $(+) \approx S_{BA} \times (+) + S_{CA} \times (-)$	$(+, +)$ $(+, -)$	$\{S_{BA} = +\}$
$\{e_1, e_2, e_3\}$	$x_A^1 \approx S_{BA}x_B^1 + S_{CA}x_C^1$ $x_A^2 \approx S_{BA}x_B^2 + S_{CA}x_C^2$ $x_A^3 \approx S_{BA}x_B^3 + S_{CA}x_C^3$	$(+) \approx S_{BA} \times (+) + S_{CA} \times (+)$ $(+) \approx S_{BA} \times (+) + S_{CA} \times (-)$ $(-) \approx S_{BA} \times (+) + S_{CA} \times (-)$	$(+, +)$	$\{S_{BA} = +, S_{CA} = +\}$

Table 2. Sign inference process. In this example variables are only the roles of regulations (signs) in an interaction graph, the variations of the species in the graph are obtained from the set of six experiments described in Table 1. For different sets of experiments we do not infer the same roles of regulations. We observe in this example that if we take into account experiments $\{e_1, e_2, e_3\}$, our qualitative system will have three constraints and not all valuations of variables S_{BA} and S_{CA} satisfy this system according to the sign algebra rules. As we obtain unique values for these variables in the solution of the system, we consider them as inferred.

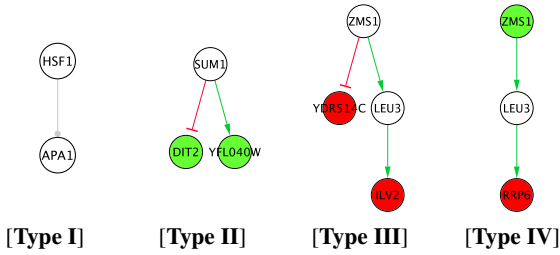


Figure 1. Classification of the multiple behavior modules (MBM). These modules are some of the MBM found in the global regulatory network of *S. Cerevisiae* extracted from [11]. Green and red interactions correspond to inferred activations and repressions respectively. Genes are colored by their expression level during certain experiment (green: more than 2-fold expression, red: less than 2-fold repression) (a) **Type I** modules are composed by direct regulations of one gene by its predecessors. Sources of the conflict in this example are: Heat shock 21°C to 37°C [28], and Cells grown to early log-phase in YPE [29]. (b) **Type II** The genes in this module have the same direct predecessor. Explanation: The interaction among *SUM1* and *YFL040W* is inferred at the beginning of the inference process, as an activation while among *SUM1* and *DIT2* is inferred as an inhibition. During the *correction* step, expression profile related to YPD Broth to Stationary Phase [28], shows that these two genes: *YFL040W* and *DIT2* overexpress under this condition. Resulting impossible to determine the state (overexpressed or underexpressed) of *SUM1*, we mark this module as a MBM. (c) **Type III** The genes in this module share a predecessor, but not the direct one. Source of the conflict: Diauxic shift [30]. (d) **Type IV** The predecessor of one gene is the successor of the other. Source of the conflict: Heat Shock 17°C to 37°C [28].

with a high number of experiments. In Sec. 3, we comment the maximum number of signs that can be inferred from a given graph.

3 RESULTS

In perturbation experiments, gene responses are observed following changes of external conditions (temperature, nutritional stress, etc.) or following gene inactivations, knock-outs or overexpression. When expression profile is available for all the genes in the network we say that we have a *complete profile*, otherwise the profile is *partial* (data is missing). The effect of gene deletions is modelled as the one of inactivations, which is imposing negative gene variations. Thus, we may say that we deal with perturbation experiments that do not change the topology of the network. An experiment in which topology is changed would be to record the effect of stresses on mutants; this possibility will be discussed elsewhere.

In order to validate our formal approach, we evaluate the percentage of the network that might be recovered from a reasonable number of perturbation experiments. We first provide theoretical limits for the percentage of recovered signs. These limits depend on the topology of the network. For the transcriptional network of *E. Coli*, these limits are estimated first by a deterministic and then by a statistical algorithm. The statistical approach uses artificial random data. Then we combine expression profiles with a publicly available structure of *E. Coli* network, and compute the percentage of recovered signs. Finally, we combine real expression profiles with chIP-chip data on *S. Cerevisiae*, and evaluate the percentage of recovered signs in a real setting.

On computational ground, we check that our algorithms are able to handle large scale data, as produced by high-throughput measurement techniques (expression arrays, chIP-chip data). This is demonstrated in the following by considering networks of more than several thousand genes.

3.1 Stress perturbation experiments: how many do you need ?

For any given network topology, even when considering all possible experimental perturbations and expression profiles, there are signs that can not be determined (see Table 2). Sign inference has thus a theoretical limit that we call *theoretical percentage of recovered signs*. This limit is unique for a given network topology. If only some perturbation experiments are available, and/or data is missing, the percentage of inferred signs will be lower. For a given number N of available expression profiles, the *average percentage of recovered signs* is defined over all sets of N different expression profiles compatible with the qualitative constraints Eqs. (1) and (2).

In this section, we calculate and comment the theoretical and the average percentages of recovered signs for the transcriptional network of *E. Coli*.

We first validate our method on the *E. Coli* network. We build the interaction graph corresponding to *E. Coli* transcriptional network, using the publicly available RegulonDB [6] as our reference. For each transcriptional regulation $A \rightarrow B$ we add the corresponding arrow between genes A and B in the interaction graph. This graph will be referred to as the *unsigned interaction graph*.

From the unsigned interaction graph of *E. Coli*, we build the *signed interaction graph*, by annotating the edges with a sign. Most of the time, the regulatory action of a transcription factor is available in RegulonDB. When it is unknown, or when it depends on the level of the transcription factor itself, we arbitrarily choose the value $+$ for this regulation. This provides a graph with 1529 nodes and 3802 edges, all edges being signed. The signed interaction graph is used to generate complete expression profiles that simulate the effect of perturbations. More precisely, a perturbation experiment is represented by a set of gene expression variations $\{X_i\}_{i=1,\dots,n}$. These variations are not entirely random: they are constrained by Eqs.(1) and (2). Then we forget the signs of network edges and we compute the qualitative system with the signs of regulations as unknowns.

The *theoretical maximum percentage of inference* is given by the number of signs that can be recovered assuming that expression profiles of *all* conceivable perturbation experiments are available. We computed this maximum percentage by using constraint solvers (the algorithm is given in Sec. 6). We found that *at most* 40.8% of the signs in the network can be inferred, corresponding to $M_{max} = 1551$ edges.

However, this maximum can be obtained only if all conceivable (much more than 2^{50}) perturbation experiments are done, which is not possible. We performed computations to understand the influence of the number of experiments N on the inference. For each value of N , where N grows from 5 to 200, we generated 100 sets of N random expression profiles. Each time our inference algorithm is used to recover signs. Then, the average percentage of inference is calculated as a function of N . The resulting statistics are shown in Fig. 2.

When the number of experiments (X-axis) equals 1, the value $M_1 = 609$ corresponds to the average number of signs inferred from a single perturbation experiment. These signs correspond to

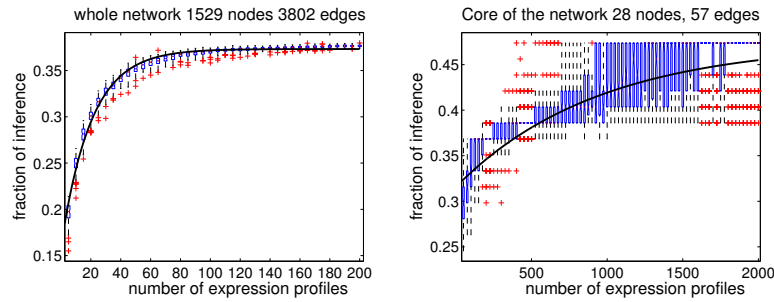


Figure 2. (Both) Statistics of inference on the regulatory network of *E. Coli* from complete expression profiles. The signed interaction graph is used to randomly generate sets of X artificial expression profiles which cover the *whole* network (*complete expression profile*). Each set of artificial profiles is then used with the unsigned interaction graph to recover regulatory roles. X-axis: number of expression profiles in the dataset. Y-axis: percentage of recovered signs in the unsigned interaction graph. This percentage may vary for a fixed number of expression profile in a set. Instead of plotting each dot corresponding to a set, we represent the distribution by boxplots. Each boxplot vertically indicates the minimum, the first quartile, the median, the third quartile and the maximum of the empiric distribution. Crosses show outliers (exceptional data points). The continuous line corresponds to the theoretical prediction $Y = M_1 + M_2(1 - (1 - p)^X)$, where M_1 stands for the number of signs that should be inferred from any expression profile (that is, inferred by the naive inference algorithm); and M_2 denotes the number of signs that could be inferred with a probability p .

(Left) Statistics of inference for the *whole* *E. Coli* transcriptional network. We estimate that at most 37,3% of the network can be inferred from a limited number of different complete expression profiles. Among the inferred regulations, we estimate to $M_1 = 609$ the number of signs that should be inferred from any complete expression profile. The remaining $M_2 = 811$ signs are inferred with a probability estimated to $p = 0.049$. Hence, 30 perturbation experiments are enough to infer 30% of the network.

(Right) Statistics of inference for the core of the former graph (see definition of a core in the text). An estimation gives $M_1 = 18$ and $M_2 = 9$ so that the maximum rate of inference is 47,3%. Since $p = 0.0011$, the number of expression profiles required to obtain a given percentage of inference is much greater than in the whole network.

single incoming regulatory interactions and are thus within the scope of the naive inference algorithm. We deduce that the naive inference algorithm allows to infer on average 18% of the signs in the network.

Surprisingly, by using our method we can significantly improve the naive inference, with little effort. For the whole *E. Coli* network it appears that a few expression profiles are enough to infer a significant percentage of the network. More precisely, 30 different expression profiles may be enough to infer one third of the network, that is about 1200 regulatory roles. Adding more expression profiles continuously increases the percentage of inferred signs. We reach a plateau close to 37,3% (this corresponds to $M = 1450$ signed regulations) for $N = 200$.

The saturation aspect of the curve in Fig. 2 is compatible with two hypotheses. According to the first hypothesis, on top of the M_1 single incoming regulations (that can be inferred with a single expression profile), there are M_2 interactions whose signs are inferred with more than one expression profile. On average, a single expression profile determines with probability $p < 1$ the sign of interactions of the latter category. According to the second hypothesis, the contributions of different experiments to the inference of this type of interactions are independent. Thus, the average number of inferred signs is $M(N) = M_1 + M_2(1 - (1 - p)^N)$. The two numbers satisfy $M_1 + M_2 < E$ (E is the total number of edges), meaning that there are edges whose signs can not be inferred.

According to this estimate the position of the plateau is $M = M_1 + M_2$ and should correspond to the theoretical maximum percentage of inferred signs M_{max} . Actually, $M < M_{max}$. The difference, although negligible in practice (to obtain M_{max} one has to perform $N > 10^{15}$ experiments) suggests that the plateau has a very weak slope. This means that contributions of different experiments to sign inference are weakly dependent.

The values of M_1, M_2, p estimate the efficiency of our method: large p, M_1, M_2 mean small number of expression profiles needed for inference. For the *E. Coli* full transcriptional network we have $p = 0.049$ per observation. This means that we need about 20 profiles to reach half of the theoretical limit of our approach.

3.2 Inferring the core of the network

Obviously, not all interactions play the same role in the network. The *core* is a subnetwork that naturally appears for computational purpose and plays an important role in the system. It consists of all oriented loops and of all oriented chains leading to loops. All oriented chains leaving the core without returning are discarded when reducing the network to its core. Acyclic graphs and in particular trees have no core. The main property of the core is that if a system of qualitative equations has no solution, then the reduced system built from its core also have no solution. Hence it corresponds to the most difficult part of the constraints to solve. It is obtained by reduction techniques that are very similar to those used in [31] (see details in Sec. 6). As an example, the core of *E. Coli* network only has 28 nodes and 57 edges. It is shown in Fig. 3.

We applied the same inference process as before to this graph. Not surprisingly, we noticed a rather different behavior when inferring signs on a core graph than on a whole graph as demonstrated in Fig. 2. In this case we need much more experiments for inference: sets of expression profiles contain from $N = 50$ to 2000 random profiles.

Two observations can be made from the corresponding statistics of inference. First as can be seen on X-axis, a much greater number of experiments is required to reach a comparable percentage of inference. Correspondingly, the value of p is much smaller than for the full network. This confirms that the core is much more difficult to infer than the rest of network. Second, Fig. 2. displays a much less continuous behavior. More precisely, it shows that for the core,

estimate $d = 0.14$, meaning that on average one loses one interaction sign for about 7 missing values. However, for the same number of expression profiles, the core of the network is more sensitive to missing data (the value of d is larger, it corresponds to lose one sign for about 4.8 missing values). For the core, increasing the number of expression profiles increases d and hence the sensitivity to missing data.

3.4 Application to *E. Coli* network with a compendium of expression profiles

We first validate our method on the *E. Coli* network. We use the compendium of expression profiles publicly available in [9].

For each experimental assay several profiles were available (including a profile for the reference initial state). We processed time series profiles, considering only the last time expression data. For each measured gene, we calculated its average variation in all the profiles of the same experiment. Then, we sorted the measured genes/regulators in four classes: 2-fold induced, 2-fold repressed, non-observed and zero variation, this last class corresponds to genes whose expression did not vary more than 2-fold under an experimental condition. Only the first two classes were used in the algorithm. Obviously this leads us to missing data: there will be edges for which neither the input, nor the output are known. Altogether, we have processed 226 sets of expression profiles corresponding to 68 different experimental assays (over-expression, gene-deletion, stress perturbation).

It appears that the signed network is consistent with only 40 complete profiles of the 68 selected. After discarding the incompatible motifs from the profiles (deleting observations that cause conflicts), 67 profiles remained that were compatible with the signed network. In these 67 expression profiles, 14,47% of the nodes of the network were observed on average as varying. When summing all the observations, we obtained that 9,8% of the edges (input and output) are observed in at least one expression profile. In order to test our algorithm we wipe out the information on edge signs and then try to recover it.

Since the profiles and network were compatible, our algorithm found no ambiguity and predicted 51 signs, i.e. 1,8% of the edges. The naive inference algorithm inferred 43 signs. Hence our algorithm inferred 8 signs, that is 15% of the total of prediction, that were not predicted by the naive algorithm.

Then we applied our algorithm, filtering our inference with different parameters, on the full set of 68 expression profiles including incompatibilities. This time 16% of the network products were observed on average. Several values of the filtering parameter k were used from $k = 1$ to $k = 15$. Without filtering we predicted 183 signs of the network (6,3%), among which 131 were inferred by the naive algorithm. We compared the predictions to the known interaction signs: 77 signs were false predictions (42% of the predictions). A source of the error in the prediction could lie on non-modelled interactions (possibly effects of sigma-factors). Filtering greatly improves our score, allowing us to retain only reliable predictions. Thus, for $k = 15$, we inferred 36 signs, of them, only 3 were incorrect predictions (8% of false prediction). We conclude that filtering is a good way to strengthen our predictions even when the model is not precise enough. We illustrate the effect of the filtering process in Fig. 5.

We notice that the inference rate is much more lower in this case than the theoretical inference rate predicted in Sec. 3.3. This shows

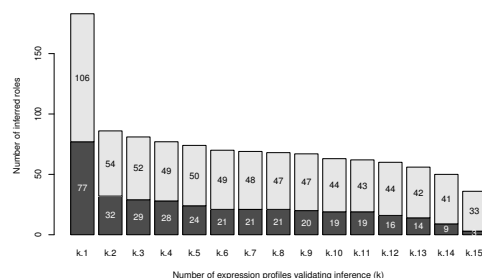


Figure 5. Results of the inference algorithm on *E. Coli* network from a compendium of 68 expression profiles. The profiles were not globally coherent. With no filtering, there are 42% of false predictions. With filtering – keeping only the sign predictions confirmed by k different sets of expression profiles – the rate of false prediction decreases to 8%.

that when the percentage of observation is very low (as it is the case here), the sign-inference process is very dependent from the type of available expression profiles. To overcome this problem, we should take into account more stress perturbation experiments and less genetic perturbation experiments.

Our algorithm also detected ambiguous modules in the network. There are 10 modules of typeI (i.e. single incoming interactions) in the network. Among these interactions, 5 are also stated as ambiguous by the naive algorithm. There are also 6 modules of typeII and III, which are not detected by the naive inference algorithm. All ambiguities are shown in Fig. 6. The list of experimental assays that yields to ambiguities on each interaction is given in the Supplementary Web site. Notice that in RegulonDB, only two of these interactions are annotated with a double-sign, i.e. they are known to have both repressor and inducer effect depending on external condition. On the other 18 interactions belonging to an ambiguous module, this analysis shows that there exist non-modelled interactions that balance the effects on the targets.

3.5 A real case: inference of signs in *S. Cerevisiae* transcriptional regulatory network

We applied our inference algorithm to the transcriptional regulatory network of the budding yeast *S. Cerevisiae*. Let us here briefly review the available sources that can be used to build the unsigned regulatory network. The experimental dataset proposed by Lee *et al.* [11] is widely used in the network reconstruction literature. It is a study conducted under nutrient rich conditions, and it consists of an extensive chIP-chip screening of 106 transcription factors. Estimations regarding the number of yeast transcription factors that are likely to regulate specific groups of genes by direct binding to the DNA vary from 141 to 209, depending on the selection criteria. In follow up papers of this work, the chIP-chip analysis was extended to 203 yeast transcription factors in rich media conditions and 84 of these regulators in at least one environmental perturbation [12]. Analysis methods were refined in 2005 by MacIsaac *et al.* [13]. From the same chIP-chip data and protein-protein interaction networks, non-parametric causality tests proposed some previously undetected transcriptional regulatory motifs [14]. Bayesian analysis also proposed transcriptional networks [15, 16, 10]. Here we selected four

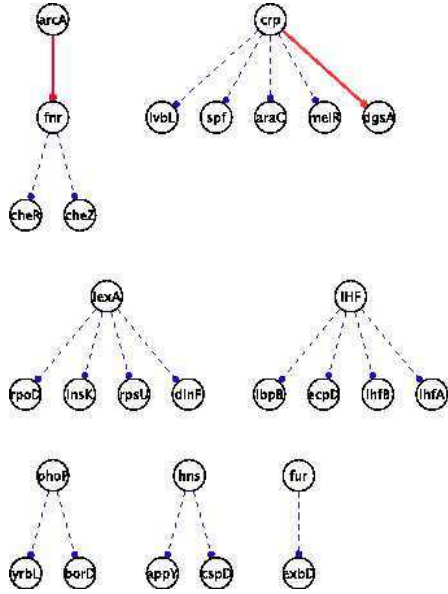


Figure 6. Interactions in the regulatory network of *E. Coli* that are ambiguous with a compendium data of expression profiles [9]. For each interaction, there exist at least two expression profiles that do not predict the same sign on the interaction. In this subnetwork, only 2 interactions (red edges) are annotated with a double-sign in RegulonDB.

of these sources. All networks are provided in the Supplementary Web site.

- (A) The first network consists in the core of the transcriptional chip-chip regulatory network produced in [11]. Starting from the full network with a p-value of 0.005, we reduced it to the set of nodes that have at least one output edge. This network was already studied in [31]. It contains 31 nodes and 52 interactions.
- (B) The second network contains all the transcriptional interactions between transcription factors shown by [11] with a p-value below 0.001. It contains 70 nodes and 96 interactions.
- (C) The third network is the set of interactions among transcription factors as inferred in [13] from sequence comparisons. We have

considered the network corresponding to a p-value of 0.001 and 2 bindings (83 nodes, 131 interactions).

- (D) The last network contains all the transcriptional interactions among genes and regulators shown by [11] with a p-value below 0.001. It contains 2419 nodes and 4344 interactions.

3.5.1 Inference process with gene-deletion expression profiles

We first applied our inference algorithm to the large-scale network (D) extracted from [11] using a panel of expression profiles for 210 gene-deletion experiments [40]. The information given by this panel is quite small, since 1,6% of all the products in the network is on average observed, and 12% of the edges (input and output) of the network are observed in at least one expression profile. Using this data, we obtain 162 regulatory roles.

We validated our prediction with a literature-curated network on Yeast [41]. We found that among the 162 sign-predictions, 12 were referenced with a known interaction in the database, and 9 with a good sign.

Gene-deletion expression profiles were used so we could compare our results to path analysis methods [23, 20] since the latter can only be applied to knock-out data (<http://chianti.ucsd.edu/idekerlab/>). Other sign-regulation inference methods need either other sources of gene-regulatory information (promoter binding information, protein-protein information), or time-series data to be performed [15, 18, 10].

Before comparing our inference results to the work of Yeang *et al.*, we tested the compatibility between their inferred network with the 210 gene-deletion experiments. We obtained that their network was incompatible with 28 of the 210 experiments. The comparison of both results showed us that the method of Yeang *et al.* infers 234 roles of widely connected paths, while our method infers 162 roles in the branches of the network. Both results intersect on 17 interactions, and no contradiction in the inferred role was reported. An illustration of these results is given in the Supplementary Web site.

This suggests that our approach is complementary to path analysis methods. Our explanation is the following: In [23, 20], network inference algorithms identify probable paths of physical interactions connecting a gene knockout to genes that are differentially expressed as a result of that knockout. This leads to search for the smallest number of interactions that carry the largest information in the network. Hence, inferred interactions are located near the core

Experiment Identifier	Description	Reference
E1	Diauxic Shift	[30]
E2	Sporulation	[33]
E3	Expression analysis of Snf2 mutant	[34]
E4	Expression analysis of Swi1 mutant	[34]
E5	Pho metabolism	[35]
E6	Nitrogen Depletion	[28]
E7	Stationary Phase	[28]
E8	Heat Shock from 21°C to 37°C	[28]
E9	Heat Shock from 17°C to 37°C	[28]

Experiment Identifier	Description	Reference
E10	Wild type response to DNA-damaging agents	[36]
E11	Mec1 mutant response to DNA-damaging agents	[36]
E12	Glycosylation defects on gene expression	[37]
E13	Cells grown to early log-phase in YPE (Rich medium with 2% of Ethanol)	[29]
E14	Cells grown to early log-phase in YPG (Rich medium with 2% of Glycerol)	[29]
E15	Titrateable promoter alleles - Ero1 mutant	[38]

Table 3. List of genome expression experiments of *S. Cerevisiae* used in the inference process. Experiments contain information on steady state shift and their curated data is available in SGD (Saccharomyces Genome Database) [39].

Interaction network	Nodes	Edges	Average number of observed nodes	In/Out observed simulnat.	Inferred signs	MBM Int. TypeI	MBM Int. Type II,III,IV	Total Inf. rate	Predictions of the naive algorithm
(A) Core of Lee transcriptional network [11, 31]	31	52	28%	46	11 (21.1%)	3 (5.7%)	0	26.8%	11%
(B) Extended Lee transcriptional network [11]	70	96	26%	70	29 (30.2%)	7 (7.2%)	0	37.4%	15,6%
(C) Inferred network [12, 13] threshold = 0.001 ; bindings=2	83	131	33%	91	21 (16%)	4 (3%)	0	19%	11%
(D) Global transcriptional network [11] p-value = 0.001	2419	4344	30%	2270	631 (14.5%) 198 (4.5%) filter k=3	281 (6.5%) no filter	463 (11%)	32%	13.9%

Table 4. Budding yeast transcriptional regulatory networks on which the sign inference algorithm was applied. For each network 14 or 15 different expression profiles were used for calculating the inference. The set of observations provided by one expression profile, was composed by at least two expressed/repressed (ratio over/under 2-fold) genes of the network. The *Input/Output observed simultaneously* column, is an indicator of the maximum possible number of sign-inferred interactions. There are three different inference results: *Inferred signs*, signs fixed in a unique way by all experiments, *MBM Interactions of TypeI*, the set of non-repeated interactions that belong to all the multiple behavior modules of TypeI detected, and *MBM Interactions of TypeII,III,IV*, the number of non-repeated interactions belonging to MBM of Type II,III,IV. For all the inference results a percentage concerning the total number of edges of the network, is calculated. The *Total inference rate* represents the percentage of the total number of edges that was inferred (inferred signs plus interactions in MBM). It is compared to the results of the naive algorithm.

of the network (even though not exactly in the core). On the contrary, as we already detailed it, the combinatorics of interaction in the core of the network is too intricate to be determined from a few hundreds of parse expression profiles with our algorithm, and we concentrate on interactions around the core.

3.5.2 Inference with stress perturbation expression profiles

In order to overcome the problem raised by the small amount of information contained in [40], we have selected stress perturbation experiments. This data corresponds to curated information available in SGD (Saccharomyces Genome Database) [39]. When time series profiles were available, we selected the last time expression array. Therefore, we collected and treated 15 sets of arrays described in Table 3. For each expression array, we sorted the measured genes/regulators in four classes: 2-fold induced, 2-fold repressed, non-observed and zero variation. We were only interested in the expression of genes that belong to any of the four networks we studied. Full datasets are available in the Supplementary Web site.

As for *E. Coli* network, it appeared that all networks (A), (B), (C) and (D) are not consistent with the whole set of expression arrays and ambiguities appeared. We performed our inference algorithm. We identified motifs that hold ambiguities, and we marked them as Multiple Behavior Modules of type I, II and III, as described in Sec. 3.1. The algorithm also generates a set of inferred signs. Then we applied the filtered algorithm (with filter $k = 3$) to the large-scale network (D).

We obtain our total inference rate adding the number of inferred signs fixed in a unique way to the number of non-repeated interactions that belong to all the detected multiple behavior modules and dividing it by the number of edges in the network. In Table 4 we show the inference rate for Networks (A), (B), (C) and (D). Depending on the network, the rate of inference goes from 19% to

37%. Hence, the rates of inference are very similar to the theoretical rates obtained for *E. Coli* network, still with a small number of perturbation experiments (14 or 15).

We validated the inferred interaction by comparing them to the literature-curated network published in [41]. We first obtained that among the 631 interactions predicted when no filtering is applied, 23 are annotated in the network, and seven annotations are contradictory to our predictions. However, among the 198 interactions predicted with a filter parameter $k = 3$, 19 are annotated in the network, and only one annotation is contradictory to our predictions. As in the case of *E. Coli*, we conclude that filtering is a good way to make strong predictions even when the model is not precise enough. We also compared the sign predictions to the predictions of the naive inference algorithm. We found that the naive algorithm usually predicts half of the signs that we obtain. In Fig. 7 we illustrate the inferred interactions for Network (B), that is, the Transcriptional network among transcription factors produced in [11].

As mentioned already, the algorithm identified a large number of ambiguities. The exhaustive list of MBM is given in the Supplementary Web site. We notice that MBM of Type I are detected in the four networks; we list the Type I modules of size 2 found for the networks (A), (B) and (C) in Table 5. In contrast, MBM of Type II, III and IV are only detected, in an important number, for Network (D) following the distribution: 85.4% of Type II, 5.3% of Type III and 9.3% of Type IV. In network (D), all the results were obtained after 3 iterations of the inference algorithm. For each MBM, a precise biological study of the species should allow to understand the origin of the ambiguity: error in expression data, missing interaction in the model or changing in the sign of the interaction during the experimentation.

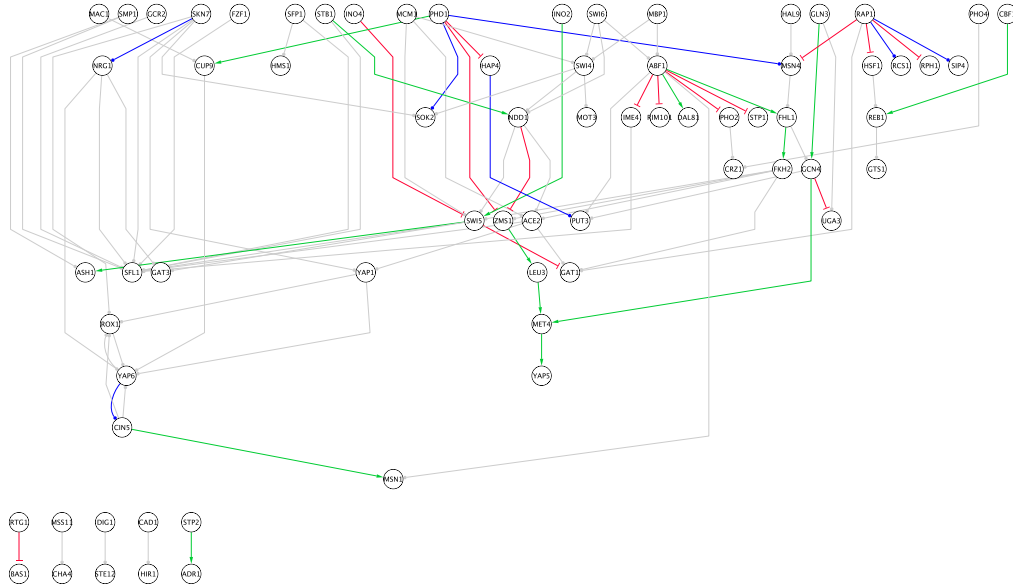


Figure 7. Transcriptional regulatory network among transcription factors (70 nodes, 96 edges) extracted from [11]. A total of 29 interactions were inferred: arrows in green, respectively in red, correspond to positive, respectively negative, interactions inferred; blue arrows correspond to the detected multiple behavior modules of TypeI. Diagram layout is performed automatically using the Cytoscape package [42].

3.6 Contribution of expression profiles to the inference

In order to evaluate the contribution of the 14 experiments used for the inference in the global network provided in [11] (2419 nodes and 4344 arcs), we addressed the following question: assuming that all inferred roles are correct, which is the experiment that causes the suppression of most of the inferred roles? For example, in Fig. 1 expression data related to YPD Broth to Stationary Phase [28], caused the suppression of the inferred interactions of the module of Type II.

We compared the 14 expression profiles according to the MBM of TypeII, III and IV that are detected by using an element of the dataset. MBM of TypeI are not included in this computation, since

they do not invalidate any interaction role, as no interaction role is inferred before their detection. The results of this comparison are shown in Fig. 8. The fourth chart illustrates that the real contribution of each expression profile does not depends on the amount of observations.

4 DISCUSSION

In this work we show how a qualitative reasoning framework can be used to infer the role of transcription factor based on expression profiles. The regulatory effect of a transcription factor on its target genes can either be an activation or a repression. Our framework

Interaction network	Actor	Target	Experiment 1	Experiment 2
Core of Lee network	YAP6 GRF10 PDH1	CIN5 MBP1 MSN4	Expression during Sporulation [33] YPD Broth to Stationary Phase [28] Nitrogen Depletion [28]	YPD Broth to Stationary Phase [28] Mec1 mutant + Heat [36] Heat shock 21 to 37 [28]
Extended Lee network	YAP6 RAP1 SKN7 PHD1 RAP1 PHD1 HAP4	CIN5 SIP4 NRG1 SOK2 RCS1 MSN4 PUT3	Expression during Sporulation [33] Expression during Sporulation [33] YPD Broth to Stationary Phase [28] Heat shock 21 to 37 [28] Wild type + Heat [36] Nitrogen Depletion [28] Expression during the diauxic shift [30]	YPD Broth to Stationary Phase [28] Expression during the diauxic shift [30] Expression during the diauxic shift [30] YPD Broth to Stationary Phase [28] Transition from fermentative to glycerol-based respiratory growth [29] Heat shock 21 to 37 [28] Snf2 mutant, YPD [34]
MacIssac inferred network	SWI5 SKN7 NRG1 NRG1	ASH1 NRG1 YAP7 GAT3	Expression regulated by the PHO pathway [35] YPD Broth to Stationary Phase [28] Expression regulated by the PHO pathway [35] Glycosylation [37]	YPD Broth to Stationary Phase [28] Nitrogen Depletion [28] Transition from fermentative to glycerol-based respiratory growth [29] Transition from fermentative to glycerol-based respiratory growth [29]

Table 5. Result of the diagnosis procedure for three networks related to budding yeast *S. Cerevisiae* (core, extended transcriptional networks of Lee, inferred network of MacIssac). We found ambiguities between single interactions and pairs of data (we call them Multiple Behavior Modules of Type I and size 2). For each ambiguous interaction found, we list two experiments that deduce a different role of interaction among these genes.

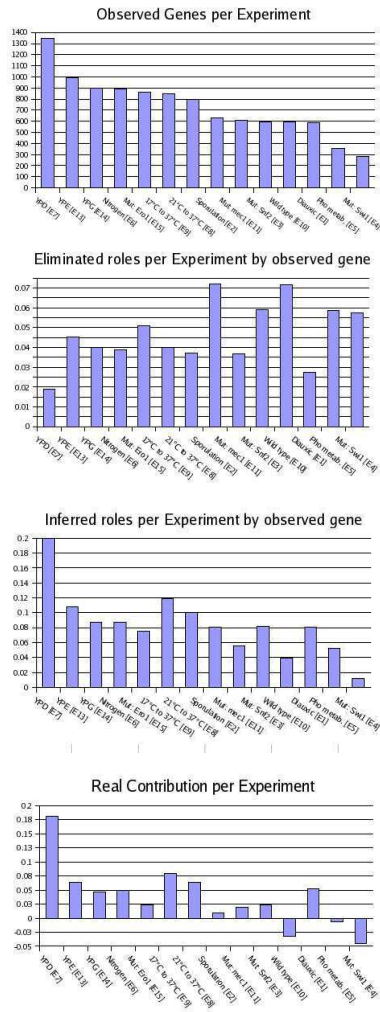


Figure 8. Comparison of 14 experiments used in the sign-inference process for the global transcriptional network in [11] (2419 genes, 4344 interactions). Each experiment has a twofold contribution: it spots inconsistent modules (MBM, that are further excluded from inference) and it predicts interaction roles. Some experiments have more predictive power, just because they include more genes. In order to normalize the predictive power, we divide the percentage of predictions by the percentage of observed nodes. For each experiment we have estimated, from top to down: (First) Number of 2-fold expressed or 2-fold repressed genes. (Second) Percentage of edges in the spotted MBMs of type II,III,IV divided by the percentage of observed nodes. (Third) Percentage of inferred interactions divided by the percentage of observed nodes. (Fourth) Real contribution of each experiment, calculated by subtracting the third quantity (inference) from the second quantity (eliminated inconsistency); negative values correspond to experiments whose main role is to spot ambiguities.

models a single qualitative rule, which basically says that the variation of expression for a gene should be explained by at least one of its regulators.

While intuitive and simple, this rule is sufficient to infer a significant number of regulatory effects from a reasonable amount of expression profiles.

On computational grounds, we designed algorithms that are able to cope with systems consisting of several thousands of genes. Our methods can thus readily be applied to networks and expression data that are produced by current high-throughput measurement techniques.

Inferring the role of transcription factor from expression profile can be seen as a particular case of network reconstruction. Let us now review some of the most relevant approaches in this domain.

Looking for high correlation or mutual information in expression profiles [16, 43] can be used to find interactions among genes. Much progress has been done over the past few years to improve the quality of statistical estimators or to detect indirect correlations, and some promising results were obtained in higher eukaryotes [43]. There remains some open problems however. First, the relation between network structure and correlation is not one to one (inference procedures rely on calculating pseudoinverses of singular matrices). Consequently, many false positive or false negatives exist among the inferred interactions. Moreover, the orientation of the inferred interactions (A acts on B) is impossible to tell if both A and B are transcription factors. Other non-parametric statistical methods are designed to test hypothetical causality relations [14].

Bayesian networks have been widely applied to gene network reconstruction [44]. Though it is limited to the class of acyclic graphs (regulation loops are excluded), the framework of Bayesian networks is attractive because it offers an intuitive, graphical representation of regulatory networks, and a simple way to deal with stochasticity in regulatory networks. This approach is however demanding, both in computational resources and experimental measurements.

Segal and coworkers [15] proposed a probabilistic model to infer transcriptional networks from promoter sequences and gene expression data. They introduce a principled framework to integrate heterogeneous sources of information. Computing the most probable model in this setting requires to solve a hard non linear optimization problem.

Network inference based on ordinary differential equation relates changes in RNA concentration to each other and to an external perturbation [45, 46]. AS ODE's are deterministic, the inferred interactions represent influences and not statistical dependencies as the other methods. It yields signed directed graphs. The main restriction is that it requires knowledge on the perturbed gene in each experiment.

More recently, some methods focused on paths of interactions [19, 20]. Global expression profiles are used to validate models of transcriptional regulation inferred from protein-protein interaction, genome-wide location analysis and expression data. A network inference algorithm identifies probable paths of physical interactions connecting a gene knockout to genes that are differentially expressed as a result of that knockout. These methods are really dependent on the topology of the networks: complex networks in which many competing or alternative paths connect a knockout to differentially expressed genes may be difficult to infer. Then, dynamical Boolean analysis is efficient to infer competing behaviors

on models containing tens of products [20, 31]. The main restriction to this method is that expression profiles have to result from a gene-deletion perturbation.

In this work, we rely on a discrete modeling framework, which consists in calculating an over-approximation of the set of possible observations, by abstracting noisy quantitative values into more robust properties. In contrast, statistical methods deal with experimental noise by explicitly modeling the noise distribution, provided enough measurements are available – which usually means hundreds of independent experiments. Moreover while most methods report the most likely model given the data, we describe the (possibly huge) set of consistent experimental behaviors with a system of qualitative constraints. Then we look for invariants in this set. In the worst case, not a single regulatory effect can be deduced from the set of constraints, whereas computing the most likely model provides with signs for all regulations. However, we expect the inferred regulations to be more robust. Another crucial difference is that the system of constraints might have no solution at all. In combination with a diagnosis procedure, we illustrated how this approach can be a relevant tool for the curation of network databases.

We compared our inference approach to a naive inference algorithm and path analysis methods introduced in [23, 20]. As detailed above, all other inference methods need additional information to infer the signs of regulations. We found that both our algorithm and path analyses infer non-trivial interactions. Both approaches are complementary: path analyses identify coupled with boolean analysis allows to infer the signs of interactions located in paths that are connected to a large number of targets; whereas our method yields information on paths connected to a quite small number of targets. Another difference is that paths analysis requires gene-deletion perturbation expression profiles, while our method give better results with stress perturbation experiments (though it can be applied to any type of experiment).

Using simulations we investigated the dependence between the number of inferred signs and the number of available observations. Not surprisingly we noticed that the topology of the regulatory graph alone had a strong influence on the estimated relationship. This was illustrated by computing statistics both on a complete regulatory network and on its core, as defined in the Methods section. The complete network is characterized by an over-representation of feedback-free regulatory cascades, which are controlled by a small number of transcription factors. In this setting, the number of inferred signs grows quasi continuously with the number of observations. In contrast, the core network does not obey the simple law “the more you observe, the better”: some expression profiles are clearly more informative than others. A challenging sequel to this work deals with experimental planification: given some control parameters, how to find the most informative experiments while keeping their number as low as possible ?

As a practical assessment of our method, we conducted sign inference experiments on *E. Coli* and *S. Cerevisiae*, using curated expression measurements, and regulatory networks either already published or based on chIP-chip data. When expression profiles mostly consisted in genetic perturbations, the inference rate was quite low, even though comparable to the results of paths analysis [20]. When expression profiles consisted in stress perturbation, our inference results corresponded to the theoretical rate of inference.

For smaller networks, of about 100 interactions, we were able to infer 20% of the regulatory roles. For bigger networks, of thousands interactions, we were only able to infer the 14%, however, a huge number of inconsistencies (that we called multiple behaviour modules) were detected. Even if we were able to state some corrections over the model or data, all our inferences and corrections proposed depend on the model we worked with. If the orientation sense of some interaction was mistaken, our inferences will be mistaken as well. In our opinion, what is even more relevant than correctly inferring signed regulations among genes is the ability to detect and isolate situations where different data sources are not consistent with each other. Moreover, if we group some of the MBM found according to the common genes they share, it is possible to assign a higher relevance to the correction of some specific interaction or data; in other words, it is possible to choose which of all the interactions is the most inappropriate.

5 CONCLUSION

In this work, we showed that our approach is suitable to infer regulatory roles of transcription factor from a *limited* amount of data. More precisely, we could infer 30-40% of the networks we studied from about 20-30 perturbation expression arrays. We believe that our approach is complementary to previous statistical methods: while qualitative modeling is a less accurate description of regulatory networks, it requires less data in order to make robust predictions. Thus, it is more adapted to situations where diverse but even limited expression profiles (some tens) are available, instead of the large panel of expression profiles usually needed for statistical methods.

We proposed a characterization of sub-networks that are more difficult to infer, called the *core* of a network. We showed on simulated data that in these core networks an unfeasible number of experiments is necessary to infer a small number of signs with high probability. For these core networks, two different strategies may be adopted. The first strategy is to build a more accurate model for these restricted subnetworks, using dynamic modeling techniques (see ([32] for a review), The alternative is to develop experiment design in our qualitative framework: find suitable values for control parameters to infer the maximum number of signs.

Finally, we illustrated another advantage of discrete modeling, namely that models can be submitted to exhaustive verification and diagnosis. As we show it in this paper it is possible to reason on systems with thousands of observations, constraints and variables, and provide intuitive diagnosis representations automatically when expression profiles happen to be ambiguous with the regulation model. As a follow-up to this work, we plan to deepen diagnosis representation, and eventually propose automatic hypothesis generation for the existence of defects.

6 METHODS

Problem statement We consider the set of equations derived from a given interaction graph G :

$$X_i^k \approx \sum_{j \rightarrow i} S_{ji} X_j^k \text{ for } 1 \leq i \leq n, 1 \leq k \leq r \quad (4)$$

where X_i^k stands for the sign of variation of species i in experiment k , and S_{ji} the sign of the influence of species j on species i . Recall that the graph G itself comes from chIP-chip experiments or sequence analysis. Using expression arrays, we obtain an experimental value for some variables X_i^k , which

will be denoted x_i^k ; more generally uppercase (resp. lowercase) letters will stand for variables of the systems (resp. constants $+$, $-$ or 0).

A single equation in the system (4) can be viewed as a predicate $P_{i,k}(X, S)$ where i denotes a node in the graph and k one of the r available experiments. If the value for some variables in the equation is known, the predicate resulting from their instantiation will be denoted $P_{i,k}(X, S)[x^k, s]$.

Our problem can now be stated as follows: given a set of expression profiles x^1, \dots, x^r , decide if the predicate:

$$P(X, S) = \bigwedge_{1 \leq i \leq n, 1 \leq k \leq r} P_{i,k}(X, S)[x^k] \quad (5)$$

can be satisfied. If so, find all variables that take the same value in all admissible valuations (so called *hard components* of the system).

Decision diagram encoding In a previous work [26], we showed how the set of solutions of a qualitative system can be computed as a decision diagram [47]. A decision diagram is a data structure meant to represent functions on finite domains; it is widely used for the verification of circuits or network protocols. Using such a compact representation of the set of solutions, we proposed efficient algorithms for computing solutions of the systems, hard components, and other properties of a qualitative system. Back to our problem, in order to predict the regulatory role of transcription factors on their target genes, it is enough to compute the decision diagram representing the predicate (5), and compute its hard components as proposed in [26]. This approach is suitable for systems of at most a couple of hundred variables. Above this limit, the decision diagram is too large in memory complexity. In our case however, we consider systems of about 4000 variables at most, which is far too large for the above mentioned algorithms.

In order to cope with the size of the problem, we propose to investigate a particular case, when all species are observed, in all experiments. In this case, $i \neq j$ implies that $P_{i,k}(X, S)[x^k]$ and $P_{j,k}(X, S)[x^k]$ share no variables. This means that P may be satisfied if and only if each predicate

$$P_{i,\cdot}(S) = \bigwedge_{1 \leq k \leq r} \exists X P_{i,k}(X, S)[x^k] \quad (6)$$

may be satisfied. As a consequence, a variable S_{ji} is a hard component of P if and only if it is a hard component of $P_{i,\cdot}$. $P_{i,\cdot}$ correspond to the constraints which relate species i to its predecessors in G for all experiments. The number of variables in $P_{i,\cdot}$ is exactly the in-degree of species i in G , which is at most 10-20 in biological networks.

As soon as some species are not observed in some experiment, the predicates $P_{i,\cdot}$ share some variables and it is not guaranteed to find all hard components by studying them separately. A brief investigation showed (data not shown) that due to the topology of the graph, most of the equations are not independent any more, even with few missing nodes. Note however, that any hard component of $P_{i,\cdot}$ still is a hard component of P . The same statement holds for

$$P_{\cdot,k}(X) = \bigwedge_{1 \leq i \leq n} \exists S P_{i,k}(X, S)[x^k] \quad (7)$$

where $P_{\cdot,k}$ corresponds to the constraints that relate all species in G for a *single* experiment. Relying on this result, we implemented the following algorithm

In practice, this algorithm is very effective in terms of computation time and number of hard components found. However, as already stated, it is not guaranteed to find all hard components of P . This is what motivates the technique described in the next paragraph.

Solving with Answer Set Programming In order to solve large qualitative systems, we also tried to encode the problem as a logic program, in the setting of answer set programming (ASP). While decision diagrams represent the set of *all* solutions, finding a model for a logic program provides *one* solution. In order to find hard components, it is enough to check for each variable V , if there exists a solution such that $V = +$ and another solution

Input:

the predicates $P_{i,\cdot}$ and $P_{\cdot,k}$ for all i and k
observed variations x

Output:

a set s of hard components of P

$s \leftarrow \emptyset$

while True do

$s' \leftarrow \bigcup_i \text{hard_components}(P_{i,\cdot}[x^k, s])$

if $s' = \emptyset$ **then return** s

$s \leftarrow s \cup s'$

$x' \leftarrow \bigcup_k \text{hard_components}(P_{\cdot,k}[x^k, s])$

if $x' = \emptyset$ **then return** s

$x \leftarrow x \cup x'$

end

Algorithm 1: Heuristic for finding hard components in large interaction networks with many expression profiles.

such that $V = -$. The ASP program we used in order to solve the qualitative system is given in supplementary materials. In the following we will denote by $\text{asp_solve}(P)$ the call to the ASP solver on the predicate P . The returned value is an admissible valuation if there is one, or \perp otherwise. The complete algorithm is reported below

Algorithm: Hard components using ASP

Input:

the predicates P
observed variations x

Output:

a set h of hard components of P

$h \leftarrow \emptyset$

$C \leftarrow \{S_{ji} | j \rightarrow i\}$

$s^* \leftarrow \text{asp_solve}(P)$

if $s^* = \perp$ **then return** \perp

while $C \neq \emptyset$ **do**

choose V in C

$s \leftarrow \text{asp_solve}(P[V = -s_V^*])$

if $s = \perp$ **then**

$h \leftarrow \{(V, s_V)\} \cup h$

else

delete from C all W in C s.t. any $s_W^* \neq s_W$

end

end

Algorithm 2: Exact algorithm for finding the set of hard components of P , based on logic programming.

We use `clasp` for solving ASP programs [48], which performs astonishingly well on our data. The procedure described in Algorithm 2 is particularly efficient to find *non* hard components: generating one solution may be enough to prove non hardness of many variables at a time.

To sum up, in order to solve a system of qualitative equations (4) with only partial observations, we use Algorithm 1 first and thus determine most (if not all) hard components. Then, Algorithm 2 is used for the remaining components, which are nearly all non hard.

Reduction technique As mentioned in the Result section, interaction graphs may be reduced in a way that preserves the satisfiability of the associated qualitative system. Consider a graph G with defined signs on its edges. If

some node n has no successor, then delete it from G . Note then, that any solution of the qualitative system associated to the new graph can be extended in a solution to the system associated to G . The same statement holds if one iteratively delete all nodes in the graph with no successor. The result of this procedure is the subgraph of G such that any node is either on a cycle, or has a cycle downstream. We refer to it as the core of the interaction graph.

The core of an interaction graph corresponds to the most difficult part to solve, because extending a solution for the core to the entire graph can be done in polynomial time, using a breadth-first traverse.

Diagnosis for noisy data When working with real-life data, it may happen that the predicate P defined in Eq. (5) cannot be satisfied. This may be due to three (non exclusive) reasons:

- a reported expression data is wrong
- an arrow (or more generally a subgraph) is missing
- the sign on an edge depends on the state of the system

In the third case, the conditions for deriving Eq. (1) are not fulfilled for one node and its qualitative equation should be discarded. This, however, does not affect the validity of the remaining equation.

In all cases, isolating the cause of the problem is a hard task. We propose the following diagnosis approach: as P is a conjunction of smaller predicates, it might happen that some subsets of the predicates are not satisfiable yet. Our strategy is then to find a “small” subsets of predicates which cannot be satisfied. A particularly interesting feature of this approach is that by selecting subsets of P_i . predicates, the result might directly be interpreted and visualized as a subgraph of the original model.

How to determine if a sign can be inferred In section 2, we have seen some examples showing that even when all feasible observations are available, it might not be possible to infer all signs in the interaction graph. Whether or not a sign can be inferred depends on the topology of the graph, but also on the actual signs on interactions. In practice, it is thus impossible to tell from the unsigned graph only if a sign can be recovered. However, it is still interesting to evaluate on fully signed interaction networks which part can be inferred. A trivial algorithm for this consists in explicitly generating all feasible observations and use the algorithms described above. This is unfeasible due to the number of observations.

With the notations introduced above, consider an observation X and sign variables S for an interaction graph. $P_i(X, S)$ denotes the constraint that link the variation of a node i to that of its predecessors given the signs of the interactions. Moreover, the real signs in the graph are denoted by s . For each node i , we build the predicate giving the feasible observations on node i and its predecessors, given the rest of the graph and the real signs s

$$O_i(X) = \exists X_{j \in \{i\} \cup \text{pred}(i)} \bigwedge_{1 \leq i \leq n} P_i(X, s)$$

Then, the constraint that we can derive on S variables is: for any observation X that is feasible $P_i(X, S)$ should hold. This constraint is more formally defined by

$$C_i(S) = \forall X O_i(X) \Rightarrow P_i(X, S)$$

Finally, the hard components of C_i are exactly the signs that can be inferred using all feasible observations. Let us sum up the procedure:

1. compute $P(X, S) = \bigwedge_{1 \leq i \leq n} P_i(X, s)$
2. compute O_i from P and the actual signs s
3. compute C_i , the constraints of signs given all feasible observations
4. compute the hard components of C_i , which are exactly the signs that can be inferred.

If it is not possible to compute $P(X, S)$ (mainly because the interaction graph is too large), we use a more sophisticated approach based on a modular decomposition of the interaction graph. The resulting algorithm can be found in the Supplementary materials.

SUPPLEMENTARY MATERIAL

Inference algorithms and all the results obtained for the *S. cerevisiae* regulatory network can be found at: www.irisa.fr/symbiose/interactionNetworks/supplementaryInference.html

ACKNOWLEDGMENT

The authors are particularly grateful to B. Kauffman, M. Gebser and T. Schaub from the University of Potsdam for their help on CLASP software. They also wish to thank the referee for their interesting and constructive remarks.

REFERENCES

- [1]Zanzoni A, Montecchi-Palazzi L, Quondam M, Ausiello G, Helmer-Citterich M, et al. (2002) MINT: a Molecular INteraction database. FEBS Lett 513:135–40.
- [2]Hermjakob H, Montecchi-Palazzi L, Lewington C, Mudali S, Kerrien S, et al. (2004) IntAct: an open source molecular interaction database. Nucleic Acids Res 32:D452–5.
- [3]Peri S, Navarro JD, Kristiansen TZ, Amanchy R, Surendranath V, et al. (2004) Human protein reference database as a discovery resource for proteomics. Nucleic Acids Res 32:D497–501.
- [4]Kanehisa M, Goto S, Kawashima S, Okuno Y, Hattori M (2004) The KEGG resource for deciphering the genome. Nucleic Acids Res 32:D277–80.
- [5]Ingenuity-Systems (1998). Ingenuity pathways knowledge base. Available: <http://www.ingenuity.com>.
- [6]Salgado H, Gama-Castro S, Peralta-Gil M, Diaz-Peredo E, Sanchez-Solano F, et al. (2006) RegulonDB (version 5.0): Escherichia coli K-12 transcriptional regulatory network, operon organization, and growth conditions. Nucleic Acids Res 34:D394–7.
- [7]Reguly T, Breitkreutz A, Boucher L, Breitkreutz BJ, Hon GC, et al. (2006) Comprehensive curation and analysis of global interaction networks in Saccharomyces cerevisiae. J Biol 5:11.
- [8]Joyce AR, Palsbo BO (2006) The model organism as a system: integrating ‘omics’ data sets. Nat Rev Mol Cell Biol 7:198–210.
- [9]Faith JJ, Hayete B, Thaden JT, Mogno I, Wierzbowski J, et al. (2007) Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. PLoS Biology 5.
- [10]Bansal M, Belcastro V, Ambesi-Impombato A, di Bernardo D (2007) How to infer gene networks from expression profiles. Mol Syst Biol 3.
- [11]Lee TI, Rinaldi NJ, Robert F, Odom DT, Bar-Joseph Z, et al. (2002) Transcriptional regulatory networks in Saccharomyces cerevisiae. Science 298:799–804.
- [12]Harbison CT, Gordon DB, Lee TI, Rinaldi NJ, Macisaac KD, et al. (2004) Transcriptional regulatory code of a eukaryotic genome. Nature 431:99–104.
- [13]MacIsaac KD, Wang T, Gordon DB, Gifford DK, Stormo GD, et al. (2006) An improved map of conserved regulatory sites for Saccharomyces cerevisiae. BMC Bioinformatics 7:113.
- [14]Xing B, van der Laan MJ (2005) A causal inference approach for constructing transcriptional regulatory networks. Bioinformatics 21:4007–13.
- [15]Segal E, Shapira M, Regev A, Pe’er D, Botstein D, et al. (2003) Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nat Genet 34:166–76.
- [16]Nariai N, Tamada Y, Imoto S, Miyano S (2005) Estimating gene regulatory networks and protein-protein interactions of Saccharomyces cerevisiae from multiple genome-wide data. Bioinformatics 21 Suppl 2:ii206–ii212.
- [17]Ferrazzi F, Magni P, Sacchi L, Nuzzo A, Petrovic U, et al. (2007) Inferring gene regulatory networks by integrating static and dynamic data. Int J Med Inform Epub 2007 Sept 6.
- [18]S B, Eils R (2005) Inferring genetic regulatory logic from expression data. Bioinformatics 21:2706–13.
- [19]Ideker T (2004) A systems approach to discovering signaling and regulatory pathways—or, how to digest large interaction networks into relevant pieces. Adv Exp Med Biol 547:21–30.
- [20]Yeang CH, Mak HC, McCuine S, Workman C, Jaakkola T, et al. (2005) Validation and refinement of gene-regulatory pathways on a network of physical interactions. Genome Biol 6:R62.
- [21]Gutierrez-Rios RM, Rosenblueth DA, Loza JA, Huerta AM, Glasner JD, et al. (2003) Regulatory network of Escherichia coli: consistency between literature knowledge and microarray profiles. Genome Res 13:2435–2443.

- [22]Bulyk ML (2006) DNA microarray technologies for measuring protein-DNA interactions. *Curr Opin Biotechnol* 17:422–30.
- [23]Yeang CH, Ideker T, Jaakkola T (2004) Physical network models. *J Comput Biol* 11:243–62.
- [24]Radulescu O, Lagarrigue S, Siegel A, Veber P, Borgne ML (2006) Topology and static response of interaction networks in molecular biology. *J R Soc Interface* 3:185–96.
- [25]Siegel A, Radulescu O, Borgne ML, Veber P, Ouy J, et al. (2006) Qualitative analysis of the relation between DNA microarray data and behavioral models of regulation networks. *Biosystems* 84:153–74.
- [26]Veber P, Borgne ML, Siegel A, Lagarrigue S, Radulescu O (2004/2005) Complex qualitative models in biology: A new approach. *Complexus* 2:140–151.
- [27]Guziolowski C, Veber P, Borgne ML, Radulescu O, Siegel A (2007) Checking consistency between expression data and large scale regulatory networks: A case study. *Journal of Biological Physics and Chemistry* :In press.
- [28]Gasch AP, Spellman PT, Kao CM, Carmel-Harel O, Eisen MB, et al. (2000) Genomic expression programs in the response of yeast cells to environmental changes. *Mol Biol Cell* 11:4241–57.
- [29]Roberts GG, Hudson AP (2006) Transcriptome profiling of *Saccharomyces cerevisiae* during a transition from fermentative to glycerol-based respiratory growth reveals extensive metabolic and structural remodeling. *Mol Genet Genomics* 276:170–86.
- [30]DeRisi JL, Iyer VR, Brown PO (1997) Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science* 278:680–6.
- [31]Kauffman S, Peterson C, Samuelsson B, Troein C (2003) Random Boolean network models and the yeast transcriptional network. *Proc Natl Acad Sci U S A* 100:14796–9.
- [32]de Jong H (2002) Modeling and simulation of genetic regulatory systems: a literature review. *J Comput Biol* 9:67–103.
- [33]Chu S, DeRisi J, Eisen M, Mulholland J, Botstein D, et al. (1998) The transcriptional program of sporulation in budding yeast. *Science* 282:699–705.
- [34]Sudarsanam P, Iyer VR, Brown PO, Winston F (2000) Whole-genome expression analysis of *snf/swi* mutants of *Saccharomyces cerevisiae*. *Proc Natl Acad Sci U S A* 97:3364–9.
- [35]Ogawa N, DeRisi J, Brown PO (2000) New components of a system for phosphate accumulation and polyphosphate metabolism in *Saccharomyces cerevisiae* revealed by genomic expression analysis. *Mol Biol Cell* 11:4309–21.
- [36]Gasch AP, Huang M, Metzner S, Botstein D, Elledge SJ, et al. (2001) Genomic expression responses to DNA-damaging agents and the regulatory role of the yeast ATR homolog Mec1p. *Mol Biol Cell* 12:2987–3003.
- [37]Cullen PJ, Sabbagh WJ, Graham E, Irick MM, van Olden EK, et al. (2004) A signaling mucin at the head of the Cdc42- and MAPK-dependent filamentous growth pathway in yeast. *Genes Dev* 18:1695–708.
- [38]Mnaimneh S, Davierwala AP, Haynes J, Moffat J, Peng WT, et al. (2004) Exploration of essential gene functions via titratable promoter alleles. *Cell* 118:31–44.
- [39]Hong E, Balakrishnan R, Christie K, Costanzo M, Dwight S, et al. (2001). *Saccharomyces genome database*. Available: <http://www.yeastgenome.org/>.
- [40]Hughes T, Marton M, Jones A, Roberts C, Stoughton R, et al. (2000) Functional discovery via a compendium of expression profiles. *Cell* 102:109–126.
- [41]Nabil Guelzim N, Bottani S, Bourguin P, 2 Képès F (2002) Topological and causal structure of the yeast transcriptional regulatory network. *Nature Genetics* 31:60–63.
- [42]Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, et al. (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research* 13:2498–2504. Availability: <http://www.cytoscape.org>.
- [43]Margolin AA, Nemenman I, Basso K, Wiggins C, Stolovitzky G, et al. (2006) ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics* 7 Suppl 1:S7.
- [44]Friedman N, Lital M, Nachman I, Pe'er D (2000) Using Bayesian networks to analyze expression data. *J Comput Biol* 7:601–20.
- [45]Di Bernardo D, Thomson M, Gardner T, Chobot S, Eastwood E, et al. (2005) Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks. *Nat Biotechnol* 23:377–383.
- [46]Bansal M, Della Gatta G, di Bernardo D (2006) Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics* 22:815–822.
- [47]Bryant R (1986) Graph-based algorithm for boolean function manipulation. *IEEE Transactions on Computers* 8:677–691.
- [48]Gebser M, Kaufmann B, Neumann A, Schaub T (2007) clasp: A conflict-driven answer set solver. In: *Ninth International Conference on Logic Programming and Nonmonotonic Reasoning*. Tempe, AZ, USA.

Bibliographie

- [1] David Angeli and Eduardo D. Sontag. Monotone control systems. *IEEE Transactions on Automatic Control*, 48(10) :1684–1698, 2003.
- [2] M Aviv, H Giladi, G Schreiber, A B Oppenheim, and G Glaser. Expression of the genes coding for the Escherichia coli integration host factor are controlled by growth phase, rpoS, ppGpp and by autoregulation. *Mol Microbiol*, 14(5) :1021–1031, Dec 1994.
- [3] V L Balke and J D Gralla. Changes in the linking number of supercoiled DNA accompany growth transitions in Escherichia coli. *J Bacteriol*, 169(10) :4499–4506, Oct 1987.
- [4] M. Bansal, G. Della Gatta, and D. di Bernardo. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22 :815–822, 2006.
- [5] Mukesh Bansal, Vincenzo Belcastro, Alberto Ambesi-Impiombato, and Diego di Bernardo. How to infer gene networks from expression profiles. *Mol Syst Biol*, 3 :78, 2007. Comparative Study.
- [6] Ziv Bar-Joseph, Shlomit Farkash, David K Gifford, Itamar Simon, and Roni Rosenfeld. Deconvolving cell cycle expression data with complementary information. *Bioinformatics*, 20 Suppl 1 :23–30, Aug 2004.
- [7] Tanya Barrett, Dennis B Troup, Stephen E Wilhite, Pierre Ledoux, Dmitry Rudnev, Carlos Evangelista, Irene F Kim, Alexandra Soboleva, Maxim Tomashevsky, and Ron Edgar. NCBI GEO : mining tens of millions of expression profiles–database and tools update. *Nucleic Acids Res*, 35(Database issue) :760–765, Jan 2007.
- [8] Alain Barrier, Antoinette Lemoine, Pierre-Yves Boelle, Chantal Tse, Didier Brault, Franck Chiappini, Julia Breittschneider, Francois Lacaine, Sidney Houry, Michel Huguier, Mark J Van der Laan, Terry Speed, Brigitte Debuire, Antoine Flahault, and Sandrine Dudoit. Colon cancer prognosis prediction by gene expression profiling. *Oncogene*, 24(40) :6155–6164, Sep 2005.
- [9] G. Batt, D. Ropers, H. de Jong, J. Geiselman, R. Mateescu, Page M., and D. Schneider. Validation of qualitative models of genetic regulatory networks by model checking : Analysis of the nutritional stress response in escherichia coli. *Bioinformatics*, 21(Suppl 1) :i19–i28, 2005.

- [10] Richard Bonneau, David J Reiss, Paul Shannon, Marc Facciotti, Leroy Hood, Nitin S Baliga, and Vesteinn Thorsson. The Inferelator : an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo. *Genome Biol*, 7(5) :R36, 2006. Evaluation Studies.
- [11] François Boulrier, Lilianne Denis-Vidal, Thibaut Henin, and François Lemaire. Lépisme. In *Proceedings of International Conference on Polynomial System Solving*, 2004.
- [12] Paul Brazhnik. Inferring gene networks from steady-state response to single-gene perturbations. *J Theor Biol*, 237(4) :427–440, Dec 2005.
- [13] R.E Bryan. Graph-based algorithm for boolean function manipulation. *IEEE Transactions on Computers*, 8 :677–691, 1986.
- [14] Svetlana Bulashevskaya and Roland Eils. Inferring genetic regulatory logic from expression data. *Bioinformatics*, 21(11) :2706–2713, Jun 2005. Evaluation Studies.
- [15] Laurence Calzone, François Fages, and Sylvain Soliman. BIOCHAM : an environment for modeling biological systems and formalizing experimental knowledge. *Bioinformatics*, 22(14) :1805–1807, Jul 2006.
- [16] M. Chaves, T. Eissing, and F. Allgöwer. Identifying mechanisms for bistability in an apoptosis network. In *Réseaux d'interaction : analyse, modélisation et simulation. RIAMS'06, Lyon, France*, 2006.
- [17] Madalena Chaves, Reka Albert, and Eduardo D Sontag. Robustness and fragility of Boolean models for genetic regulatory networks. *J Theor Biol*, 235(3) :431–449, Aug 2005.
- [18] Vijay Chickarmane, Carl Troein, Ulrike A Nuber, Herbert M Sauro, and Carsten Peterson. Transcriptional dynamics of the embryonic stem cell switch. *PLoS Comput Biol*, 2(9) :e123, Sep 2006.
- [19] Gene Ontology Consortium. The Gene Ontology (GO) project in 2006. *Nucleic Acids Res*, 34(Database issue) :322–326, Jan 2006.
- [20] Bhaskar DasGupta, German A. Enciso, Eduardo D. Sontag, and Yi Zhang. Algorithmic and complexity results for decompositions of biological networks into monotone subsystems. In Carme Álvarez and Maria J. Serna, editors, *WEA*, volume 4007 of *Lecture Notes in Computer Science*, pages 253–264. Springer, 2006.
- [21] H de Jong, J Geiselmann, G Batt, C Hernandez, and M Page. Qualitative simulation of the initiation of sporulation in bacillus subtilis. *Bulletin of Mathematical Biology*, 66(2) :261–300, 2004.
- [22] H. de Jong, J.-L. Gouzé, C. Hernandez, M. Page, T. Sari, and J. Geiselmann. Qualitative simulation of genetic regulatory networks using piecewise-linear models. *Bulletin of Mathematical Biology*, 66 :301–340, 2004.
- [23] Hidde De Jong, Jean-Luc Guze, Celine Hernandez, Michel Page, Tewfik Sari, and Johannes Geiselmann. Qualitative simulation of genetic regulatory networks using piecewise-linear models. *Bull Math Biol*, 66(2) :301–340, Mar 2004.

- [24] Steven Eker, Merrill Knapp, Keith Laderoute, Patrick Lincoln, Jose Meseguer, and Kemal Sonmez. Pathway logic : symbolic analysis of biological signaling. *Pac Symp Biocomput*, pages 400–412, 2002.
- [25] F. Fages. Consistency of clark’s completion and existence of stable models, 1992.
- [26] Jeremiah J. Faith, Boris Hayete, Joshua T. Thaden, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, James J. Collins, and Timothy S. Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biology*, 5(1), 2007.
- [27] D. Fell. *Understanding the Control of Metabolism*. Portland Press, London, 1997.
- [28] N Friedman, M Linial, I Nachman, and D Pe’er. Using Bayesian networks to analyze expression data. *J Comput Biol*, 7(3-4) :601–20, 2000.
- [29] Christine Froidevaux, Marie-Claude Gaudel, and Michèle Soria, editors. *Types de données et algorithmes*. Ediscience, 1993.
- [30] Timothy S Gardner, Diego di Bernardo, David Lorenz, and James J Collins. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301(5629) :102–105, Jul 2003.
- [31] Martin Gebser, Benjamin Kaufmann, André Neumann, and Torsten Schaub. *clasp* : A conflict-driven answer set solver. In *LPNMR*, pages 260–265, 2007.
- [32] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert A. Kowalski and Kenneth Bowen, editors, *Proceedings of the Fifth International Conference on Logic Programming*, pages 1070–1080, Cambridge, Massachusetts, 1988. The MIT Press.
- [33] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Comput.*, 9(3/4) :365–386, 1991.
- [34] Rosa Maria Gutierrez-Rios, David A Rosenblueth, Jose Antonio Loza, Araceli M Huerta, Jeremy D Glasner, Fred R Blattner, and Julio Collado-Vides. Regulatory network of Escherichia coli : consistency between literature knowledge and microarray profiles. *Genome Res*, 13(11) :2435–2443, Nov 2003. Evaluation Studies.
- [35] Salgado H, Santos-Zavaleta A, Gama-Castro S, Peralta-Gil M, Penaloza-Spinola MI, Martinez-Antonio A, Karp PD, and Collado-Vides J. The comprehensive updated regulatory network of Escherichia coli K-12. *BMC Bioinformatics*, 7(1) :5, Jan 2006. EDITORIAL.
- [36] K R Heidtke and S Schulze-Kremer. Design and implementation of a qualitative simulation model of lambda phage infection. *Bioinformatics*, 14(1) :81–91, 1998.
- [37] R. Heinrich and S. Schuster. *The Regulation of Cellular Systems*. Chapman and Hall, New York, 1996.
- [38] Henning Hermjakob, Luisa Montecchi-Palazzi, Chris Lewington, Sugath Mudali, Samuel Kerrien, Sandra Orchard, Martin Vingron, Bernd Roechert, Peter Roepstorff, Alfonso Valencia, Hanah Margalit, John Armstrong, Amos Bairoch, Gianni Cesareni, David Sherman, and Rolf Apweiler. IntAct : an open source molecular interaction database. *Nucleic Acids Res*, 32(Database issue) :D452–5, 2004.

- [39] Aimo Hinkkanen, Karl R. Lang, and Andrew B. Whinston. A set-theoretical foundation of qualitative reasoning and its application to the modeling of economics and business management problems. *Information Systems Frontiers*, 5(4) :379–399, 2003.
- [40] B Holst, L Sogaard-Andersen, H Pedersen, and P Valentin-Hansen. The cAMP-CRP/CytR nucleoprotein complex in *Escherichia coli* : two pairs of closely linked binding sites for the cAMP-CRP activator complex are involved in combinatorial regulation of the *cdd* promoter. *EMBO J*, 11(10) :3635–3643, Oct 1992.
- [41] C Y Huang and J E Jr Ferrell. Ultrasensitivity in the mitogen-activated protein kinase cascade. *Proc Natl Acad Sci U S A*, 93(19) :10078–10083, Sep 1996.
- [42] John P Huelsenbeck, Bret Larget, and Michael E Alfaro. Bayesian phylogenetic model selection using reversible jump Markov chain Monte Carlo. *Mol Biol Evol*, 21(6) :1123–1133, Jun 2004.
- [43] T. Hughes, M. Marton, A. Jones, C. Roberts, R. Stoughton, C. Armour, H. Bennett, E. Coffey, H. Dai, and Y. He. Functional discovery via a compendium of expression profiles. *Cell*, 102(1) :109–126, 2000.
- [44] Ingenuity-Systems. Ingenuity pathways knowledge base. Available : <http://www.ingenuity.com>, 1998.
- [45] A Ishihama. Functional modulation of *Escherichia coli* RNA polymerase. *Annu Rev Microbiol*, 54 :499–518, 2000.
- [46] Andrew R Joyce and Bernhard O Palsson. The model organism as a system : integrating 'omics' data sets. *Nat Rev Mol Cell Biol*, 7(3) :198–210, 2006.
- [47] Naftali Kaminski and Nir Friedman. Practical approaches to analyzing results of microarray experiments. *Am J Respir Cell Mol Biol*, 27(2) :125–132, Aug 2002.
- [48] Minoru Kanehisa, Susumu Goto, Shuichi Kawashima, Yasushi Okuno, and Masahiro Hattori. The KEGG resource for deciphering the genome. *Nucleic Acids Res*, 32(Database issue) :D277–80, 2004.
- [49] M Kaufman, C Soule, and R Thomas. A new necessary condition on interaction graphs for multistationarity. *J Theor Biol*, 248(4) :675–685, Oct 2007.
- [50] Boris N Kholodenko. Cell-signalling dynamics in time and space. *Nat Rev Mol Cell Biol*, 7(3) :165–176, Mar 2006.
- [51] Boris N Kholodenko, Anatoly Kiyatkin, Frank J Bruggeman, Eduardo Sontag, Hans V Westerhoff, and Jan B Hoek. Untangling the wires : a strategy to trace functional interactions in signaling and gene networks. *Proc Natl Acad Sci U S A*, 99(20) :12841–12846, Oct 2002.
- [52] K W Kohn. Molecular interaction map of the mammalian cell cycle control and DNA repair systems. *Mol Biol Cell*, 10(8) :2703–2734, Aug 1999.
- [53] B.J. Kuipers. *Qualitative reasoning. Modeling and simulation with incomplete knowledge*. MIT Press, 1994.

- [54] Anshul Kundaje, Manuel Middendorf, Mihir Shah, Chris H Wiggins, Yoav Freund, and Christina Leslie. A classification-based framework for predicting and analyzing gene regulatory response. *BMC Bioinformatics*, 7 Suppl 1 :S5, 2006.
- [55] Tong Ihn Lee, Nicola J Rinaldi, Francois Robert, Duncan T Odom, Ziv Bar-Joseph, Georg K Gerber, Nancy M Hannett, Christopher T Harbison, Craig M Thompson, Itamar Simon, Julia Zeitlinger, Ezra G Jennings, Heather L Murray, D Benjamin Gordon, Bing Ren, John J Wyrick, Jean-Bosco Tagne, Thomas L Volkert, Ernest Fraenkel, David K Gifford, and Richard A Young. Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, 298(5594) :799–804, Oct 2002.
- [56] Nicola Leone, Gerald Pfeifer, Wolfgang Faber, Thomas Eiter, Georg Gottlob, Simona Perri, and Francesco Scarcello. The dlvs system for knowledge representation and reasoning. *ACM Trans. Comput. Log.*, 7(3) :499–562, 2006.
- [57] James C Liao, Riccardo Boscolo, Young-Lyeol Yang, Linh My Tran, Chiara Sabbati, and Vwani P Roychowdhury. Network component analysis : reconstruction of regulatory signals in biological systems. *Proc Natl Acad Sci U S A*, 100(26) :15522–15527, Dec 2003.
- [58] Yuliya Lierler. cmodels - sat-based disjunctive answer set solver. In Chitta Baral, Gianluigi Greco, Nicola Leone, and Giorgio Terracina, editors, *LPNMR*, volume 3662 of *Lecture Notes in Computer Science*, pages 447–451. Springer, 2005.
- [59] Konstantinos Liolios, Nektarios Tavernarakis, Philip Hugenholtz, and Nikos C Kyrpides. The Genomes On Line Database (GOLD) v.2 : a monitor of genome projects worldwide. *Nucleic Acids Res*, 34(Database issue) :332–334, Jan 2006.
- [60] Kenzie D MacIsaac, Ting Wang, D Benjamin Gordon, David K Gifford, Gary D Stormo, and Ernest Fraenkel. An improved map of conserved regulatory sites for *Saccharomyces cerevisiae*. *BMC Bioinformatics*, 7(NIL) :113, 2006.
- [61] H. Marchand, P. Bournai, M. Le Borgne, and P. Le Guernic. Synthesis of discrete-event controllers based on the signal environment. *Discrete Event Dynamic System : Theory and Applications*, 10(4) :325–346, October 2000.
- [62] Adam A Margolin, Ilya Nemenman, Katia Basso, Chris Wiggins, Gustavo Stolovitzky, Riccardo Dalla Favera, and Andrea Califano. ARACNE : an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7 Suppl 1(NIL) :S7, 2006.
- [63] V Matys, O V Kel-Margoulis, E Fricke, I Liebich, S Land, A Barre-Dirrie, I Reuter, D Chekmenov, M Krull, K Hornischer, N Voss, P Stegmaier, B Lewicki-Potapov, H Saxel, A E Kel, and E Wingender. TRANSFAC and its module TRANSCompel : transcriptional gene regulation in eukaryotes. *Nucleic Acids Res*, 34(Database issue) :108–110, Jan 2006.
- [64] Lisa M Maurer, Elizabeth Yohannes, Sandra S Bondurant, Michael Radmacher, and Joan L Slonczewski. pH regulates genes for flagellar motility, catabolism,

- and oxidative stress in *Escherichia coli* K-12. *J Bacteriol*, 187(1) :304–319, Jan 2005.
- [65] Nabil Nabil Guelzim, Samuele Bottani, Paul Bourguine, and François 2 Képès. Topological and causal structure of the yeast transcriptional regulatory network. *Nature Genetics*, 31 :60–63, 2002.
- [66] I Nachman, A Regev, and N Friedman. Inferring quantitative models of regulatory networks from expression data. *Bioinformatics*, 20 Suppl 1 :248–256, Aug 2004.
- [67] Chris J Needham, James R Bradford, Andrew J Bulpitt, and David R Westhead. A primer on learning in Bayesian networks for computational biology. *PLoS Comput Biol*, 3(8) :e129, Aug 2007.
- [68] Bela Novak and John J Tyson. A model for restriction point control of the mammalian cell cycle. *J Theor Biol*, 230(4) :563–579, Oct 2004.
- [69] M L Opel, S M Arfin, and G W Hatfield. The effects of DNA supercoiling on the expression of operons of the *ilv* regulon of *Escherichia coli* suggest a physiological rationale for divergently transcribed operons. *Mol Microbiol*, 39(5) :1109–1115, Mar 2001.
- [70] Jason A Papin and Bernhard O Palsson. Topological analysis of mass-balanced signaling networks : a framework to obtain network properties including crosstalk. *J Theor Biol*, 227(2) :283–297, Mar 2004.
- [71] Barriot R, Sherman DJ, and Dutour I. How to decide which are the most pertinent overly-represented features during gene set enrichment analysis. *BMC Bioinformatics*, 8(1) :332, Sep 2007. JOURNAL ARTICLE.
- [72] Ovidiu Radulescu, Sandrine Lagarrigue, Anne Siegel, Philippe Veber, and Michel Le Borgne. Topology and static response of interaction networks in molecular biology. *J R Soc Interface*, 3(6) :185–196, Feb 2006.
- [73] Ovidiu Radulescu, Anne Siegel, Sandrine Lagarrigue, and Elisabeth Pécou. A model for regulated fatty acid metabolism in liver ; equilibria and their changes. Arxiv q-bio.CB/0603021.
- [74] Élisabeth Remy, Paul Ruet, and Denis Thieffry. Graphic requirements for multistability and attractive cycles in a boolean dynamical framework. Technical report, Institut de Mathématiques de Luminy, 2005.
- [75] K Y Rhee, M Opel, E Ito, S p Hung, S M Arfin, and G W Hatfield. Transcriptional coupling between the divergent promoters of a prototypic LysR-type regulatory system, the *ilvYC* operon of *Escherichia coli*. *Proc Natl Acad Sci U S A*, 96(25) :14294–14299, Dec 1999.
- [76] Adrien Richard and Jean-Paul Comet. Necessary conditions for multistationarity in discrete dynamical systems. *Discrete Applied Mathematics*, 2007.
- [77] Adrien Richard, Jean-Paul Comet, and Gilles Bernot. R. thomas’ modeling of biological regulatory networks : introduction of singular states in the qualitative dynamics. *Fundamenta Informaticae*, 65(4) :373–392, 2005.

- [78] Delphine Ropers, Hidde de Jong, Michel Page, Dominique Schneider, and Johannes Geiselmann. Qualitative simulation of the carbon starvation response in *Escherichia coli*. *Biosystems*, 84(2) :124–152, May 2006.
- [79] Jean-Francois Rual, Kavitha Venkatesan, Tong Hao, Tomoko Hirozane-Kishikawa, Amelie Dricot, Ning Li, Gabriel F Berriz, Francis D Gibbons, Matija Dreze, Nono Ayivi-Guedehoussou, Niels Klitgord, Christophe Simon, Mike Boxem, Stuart Milstein, Jennifer Rosenberg, Debra S Goldberg, Lan V Zhang, Sharyl L Wong, Giovanni Franklin, Siming Li, Joanna S Albala, Janghoo Lim, Carlene Fraughton, Estelle Llamosas, Sebiha Cevik, Camille Bex, Philippe Lamesch, Robert S Sikorski, Jean Vandenhoute, Huda Y Zoghbi, Alex Smolyar, Stephanie Bosak, Reynaldo Sequerra, Lynn Doucette-Stamm, Michael E Cusick, David E Hill, Frederick P Roth, and Marc Vidal. Towards a proteome-scale map of the human protein-protein interaction network. *Nature*, 437(7062) :1173–1178, Oct 2005.
- [80] Klamt S and Stelling J. *System Modeling in Cellular Biology : From Concepts to Nuts and Bolts*, chapter Stoichiometric and constraint-based modelling, pages 73–96. Cambridge, MIT Press, 2006.
- [81] E Segal, R Yelensky, and D Koller. Genome-wide discovery of transcriptional modules from DNA sequence and gene expression. *Bioinformatics*, 19 Suppl 1 :273–282, 2003. Comparative Study.
- [82] Eran Segal, Michael Shapira, Aviv Regev, Dana Pe’er, David Botstein, Daphne Koller, and Nir Friedman. Module networks : identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat Genet*, 34(2) :166–176, Jun 2003.
- [83] A Siegel, O Radulescu, M Le Borgne, P Veber, J Ouy, and S Lagarrigue. Qualitative analysis of the relation between DNA microarray data and behavioral models of regulation networks. *Biosystems*, 84(2) :153–174, May 2006.
- [84] Patrik Simons. Extending the stable model semantics with more expressive rules. In Michael Gelfond, Nicola Leone, and Gerald Pfeifer, editors, *LPNMR*, volume 1730 of *Lecture Notes in Computer Science*, pages 305–316. Springer, 1999.
- [85] Gordon K Smyth, Yee Hwa Yang, and Terry Speed. Statistical issues in cDNA microarray data analysis. *Methods Mol Biol*, 224 :111–136, 2003.
- [86] E. H. Snoussi and R. Thomas. Logical identification of all steady states : The concept of feedback loop characteristic states. *Bulletin of Mathematical Biology*, 55 :973–991, 1993.
- [87] E.H. Snoussi. Necessary conditions for multistationarity and stable periodicity. *J.Biol.Syst.*, 6(3-9), 1998.
- [88] Christophe Soulé. Graphic requirements for multistationarity. *Complexus*, 1(3) :123–133, 2003.
- [89] Christophe Soulé. Mathematical approaches to differentiation and gene regulation. *Comptes rendus biologiques*, 329(1) :13–20, 2006.

- [90] Ralf Steuer, Thilo Gross, Joachim Selbig, and Bernd Blasius. Structural kinetic modeling of metabolic networks. *Proc Natl Acad Sci U S A*, 103(32) :11868–11873, Aug 2006.
- [91] Tommi Syrjänen and Ilkka Niemelä. The smodels system. In Thomas Eiter, Wolfgang Faber, and Miroslaw Truszczyński, editors, *LPNMR*, volume 2173 of *Lecture Notes in Computer Science*, pages 434–438. Springer, 2001.
- [92] R. Thomas and M. Kaufman. Multistationarity, the basis of cell differentiation and memory. i. structural conditions of multistationarity and other nontrivial behavior. *Chaos*, 11(1) :170–179, 2001.
- [93] L. Travé-Massuyès and P. Dague, editors. *Modèles et raisonnements qualitatifs*. Hermes sciences, 2003.
- [94] L. Travé-Massuyès, P. Dague, and Guerrin F. *Le raisonnement qualitatif pour les sciences de l'Ingénieur*. Hermes sciences, 1997.
- [95] John J Tyson, Katherine C Chen, and Bela Novak. Sniffers, buzzers, toggles and blinkers : dynamics of regulatory and signaling pathways in the cell. *Curr Opin Cell Biol*, 15(2) :221–231, Apr 2003.
- [96] E P van Someren, B L T Vaes, W T Steegenga, A M Sijbers, K J Dechering, and M J T Reinders. Least absolute regression network analysis of the murine osteoblast differentiation network. *Bioinformatics*, 22(4) :477–484, Feb 2006.
- [97] Philippe Veber, Michel Le Borgne, Anne Siegel, Sandrine Lagarrigue, and Ovidiu Radulescu. Complex qualitative models in biology : A new approach. *Complexus*, 2(3-4) :140–151, 2004.
- [98] Jean-Philippe Vert. Kernel methods in genomics and computational biology. Technical report, Centre de Bio-informatique, Ecole Nationale Supérieure des Mines de Paris, hal-00012124,arXiv :q-bio.QM/0510032, 2005.
- [99] D Weichart, R Lange, N Henneberg, and R Hengge-Aronis. Identification and characterization of stationary phase-inducible genes in Escherichia coli. *Mol Microbiol*, 10(2) :405–20, Oct 1993.
- [100] Edgar Wingender, Jennifer Hogan, Frank Schacherer, Anatolij P Potapov, and Olga Kel-Margoulis. Integrating pathway data for systems pathology. *In Silico Biol*, 7(2 Suppl) :17–25, 2007.
- [101] Y Yamanishi, J-P Vert, and M Kanehisa. Protein network inference from multiple genomic data : a supervised approach. *Bioinformatics*, 20 Suppl 1 :363–370, Aug 2004. Evaluation Studies.
- [102] Chen-Hsiang Yeang, Trey Ideker, and Tommi Jaakkola. Physical network models. *J Comput Biol*, 11(2-3) :243–262, 2004.
- [103] Chen-Hsiang Yeang, H Craig Mak, Scott McCuine, Christopher Workman, Tommi Jaakkola, and Trey Ideker. Validation and refinement of gene-regulatory pathways on a network of physical interactions. *Genome Biol*, 6(7) :R62, 2005.

- [104] Necmettin Yildirim, Moises Santillan, Daisuke Horike, and Michael C Mackey. Dynamics and bistability in a reduced model of the lac operon. *Chaos*, 14(2) :279–292, Jun 2004. Comparative Study.

Table des figures

1.1	Cycle d'analyse des données haut-débit	3
1.2	(a) Mesure d'expression sur un échantillon de poumon humain, extraite de [47]. Le même échantillon a été analysé sur deux puces de même modèle. Chaque point correspond à une sonde de la puce. Chaque sonde correspond spécifiquement à un ARN transcrit. Sur chaque axe est représentée la mesure d'expression normalisée obtenue sur chaque puce. Dans le cas idéal, tous les points devraient se trouver sur la droite d'équation $y = x$. Les droites vertes correspondent à une variation d'un facteur 2 (augmentation et diminution respectivement). (b) Même type de comparaison, mais cette fois entre deux échantillons issus de tissus différents.	5
2.1	Exemples de graphe d'interaction. Les flèches d'extrémité triangulaire (en vert) représentent des activations, les flèches d'extrémité en T (en rouge) représentent des inhibitions.	15
2.2	Graphe d'interaction pour l'opéron lactose chez <i>E. coli</i>	16
3.1	(a) Un exemple de graphe d'interaction. Les sommets sont des espèces chimiques, les flèches représentent des régulations entre espèces. Les flèches vertes (extrémités triangulaires) indiquent des activations, les flèches rouges (extrémités en T) des inhibitions, les flèches noires (extrémité en cercle) des régulations d'action indéterminée. (b) Le même graphe d'interaction, où l'on a figuré une mesure (partielle) des sommets. La couleur verte d'un sommet indique un accroissement du niveau de l'espèce entre deux conditions expérimentales, la couleur rouge une diminution, la couleur grise l'absence de variation. L'absence de coloration signifie que le sommet n'a pas été mesuré	22
3.2	Résumé des cas où une variation d'une espèce est effectivement expliquée par la variation des espèces qui la régulent. Première ligne : le sommet cible admet une variation, on doit donc parmi les prédécesseurs pouvoir trouver une influence de même signe. Deuxième ligne : si la variation du sommet est nulle, alors soit on peut trouver deux contributions de signe opposé, soit tous les régulateurs ont une variation nulle	25

4.1	L'arbre de décision associé à la fonction booléenne $F(w, x, y, z) = ((y \otimes z) \vee (x \wedge w \wedge \neg z)) \wedge (w \vee \neg y)$, où \otimes désigne le ou exclusif et $x \prec y \prec z \prec w$	46
4.2	Diagramme de décision associé à l'arbre donné en figure 4.1. Notons le gain en nombre de sommets, passé de 21 à 9.	48
4.3	Exemple de graphe d'interaction et de la contrainte de consistance associée pour la construction des diagrammes.	56
6.1	Exemples de régulations trouvées dans RegulonDB. (A) Inhibition du gène <i>fiu</i> par la protéine Fur (produite par le gène <i>fur</i>). On lui fait correspondre l'interaction $\text{fur} \xrightarrow{-} \text{fiu}$. (B) Production du dimère IHF par deux gènes <i>ihfA</i> et <i>ihfB</i> (un pour chaque sous-unité). Le complexe IHF active l'expression du gène <i>aceA</i> . On en déduit les interactions $\text{ihfA} \xrightarrow{+} \text{IHF}$, $\text{ihfB} \xrightarrow{+} \text{IHF}$ et $\text{IHF} \xrightarrow{+} \text{aceA}$	90
6.2	(a) Sous-graphe inconsistant avec les données du tableau 6.1. (b) Contrainte qualitative correspondante	92
6.3	Correction du graphe d'interaction, selon les données disponibles sur les facteurs σ	94
6.4	Sous-graphe inconsistant avec les données bibliographiques, après ajout des régulations par les facteurs σ	94
6.5	Graphe d'interaction pour le réseau transcriptionnel d' <i>E. coli</i> , avec facteurs σ	95
6.6	Prédiction de la réponse au stress nutritionnel. Les courbes montrent l'évolution des résultats en fonction du seuillage des données. On se donne un seuil en dessous duquel les variations mesurées sont jugées non significatives. Dans ce cas, la variable concernée est déclarée non observée. Plus on choisit un seuil élevé, moins on garde de données, mais plus ces données sont fiables. (a) Pourcentage de prédictions correctes en fonction du seuil. (b) Nombre de gènes conservés en fonction du seuil.	96
6.7	Statistique du nombre de signes déduits à l'aide d'un nombre donné de mesures. En abscisse, nombre de mesures disponibles; en ordonnées le nombre de signes de régulation que l'on peut en déduire.	98
6.8	Résultats de l'inférence des signes de régulation sur le réseau d' <i>E. coli</i> , à partir de données d'expression.	101
6.9	Cas typiques de défaut à la contrainte de consistance, trouvés dans les données sur <i>S. cerevisiae</i>	102
7.1	Illustrations pour la consistance de chemins. Le premier cas est présenté dans [102]. Les deux graphes ont une seule entrée : <i>A</i>	109
7.2	Contre-exemples pour la preuve du théorème 12. (a) Cas \mathcal{P} -consistant mais pas \mathcal{N} -consistant. (b) Cas \mathcal{N} -consistant mais pas \mathcal{P} -consistant. Les deux graphes ont une seule entrée : <i>A</i>	110

Résumé

Les techniques de biologie moléculaire dites haut-débit permettent de mesurer un grand nombre de variables simultanément. Elles sont aujourd'hui couramment utilisées et produisent des masses importantes de données. Leur exploitation est compliquée par le bruit généralement observé dans les mesures, et ce d'autant plus que ces dernières sont en général trop onéreuses pour être suffisamment reproduites. La question abordée dans cette thèse porte sur l'intégration et l'exploitation des données haut-débit : chaque source de données mesurant un aspect du fonctionnement cellulaire, comment les combiner dans un modèle et en tirer des conclusions pertinentes sur le plan biologique ? Nous introduisons un critère de consistance entre un modèle graphique des régulations cellulaires et des données de déplacement d'équilibre. Nous montrons ensuite comment utiliser ce critère comme guide pour formuler des prédictions ou proposer des corrections en cas d'incompatibilité. Ces différentes tâches impliquent la résolution de contraintes à variables sur domaines finis, pour lesquelles nous proposons deux approches complémentaires. La première est basée sur la notion de diagramme de décision, qui est une structure de données utilisée pour la vérification des circuits ; la deuxième fait appel à des techniques récentes de programmation logique. L'utilisation de ces techniques est illustrée avec des données réelles sur la bactérie *E. coli* et sur la levure. Les réseaux étudiés comportent jusqu'à plusieurs milliers de gènes et de régulations. Nous montrons enfin, sur ces données, comment notre critère de consistance nous permet d'arriver à des prédictions robustes, ainsi que des corrections pertinentes du modèle étudié.

Abstract

High-throughput techniques in molecular biology are able to measure a huge number of variables simultaneously. They are currently used as a routine investigation method and therefore produce large amounts of information. The analysis of high throughput data is particularly difficult, due to their usual high level of noise, and the lack of independent samples for the same experiment. This thesis deals with the integration and the analysis of high throughput data : each type of data can be thought of as measuring a certain aspect of a biological process, so how to combine these heterogeneous data in order to deduce relevant conclusions ? To this end, we introduce a compatibility criterion between a graphical model of cellular regulations and equilibrium shift data. We then show how this criterion can be used to derive predictions or to propose corrections in case of an incompatibility. These tasks require the resolution of constraints on finite domain variables, and we developed two approaches for these problems. The first one is based on the notion of decision diagrams, which is classical data structure used for circuit verification ; the second approach we propose relies on recent techniques from logical programming. Finally, we applied our approach to real data on *E. coli* and yeast. The networks we considered have up to several thousands of genes and regulations. We show on these data that our consistency criterion can effectively be used to derive robust predictions, as well as relevant corrections to the model under study.