Modélisation ASP pour l'analyse méthodique de réseaux géniques discrets

Nicolas Mobilia **Fabien Corblin** **Éric Fanchon** **Laurent Trilling**
UJF-Grenoble 1 / CNRS / TIMC-IMAG UMR 5525, Grenoble, F-38041, France
{Nicolas.Mobilia;Fabien.Corblin;Eric.Fanchon;Laurent.Trilling}@imag.fr

Résumé

Nous considérons des réseaux géniques décrits comme des réseaux de Thomas et en présentons une nouvelle modélisation à l'aide du paradigme de programmation logique ASP (Answer Set Programming). L'objectif est de mettre en évidence l'intérêt de cette modélisation pour la mise en œuvre d'une méthodologie rigoureuse d'analyse des réseaux géniques. D'une part il s'agit d'imposer des contraintes représentant les données biologiques et les hypothèses et d'autre part de disposer de fonctionnalités bien définies, telle que la correction d'incohérence ou l'inférence de nouvelles propriétés biologiques. Nous montrons que le paradigme ASP se révèle un bon candidat grâce à son pouvoir d'expression (en particulier pour la mise en place de défauts), à la minimalité des *answer set* et aux performances du solveur utilisé.

1 Introduction

Les connaissances biologiques sont de plus en plus étendues et l'utilisation d'outils informatiques pour construire, analyser et exploiter des modèles est devenue incontournable. Les réseaux de régulation géniques n'échappent pas à cette réalité.

L'approche traditionnelle consiste à proposer un modèle initial, où tous les paramètres sont instanciés et ensuite à confronter les comportements de ce modèle avec les résultats des expérimentations. En cas d'incompatibilité, il y a révision du modèle. Ce processus est répété jusqu'à obtenir des prédictions et des données expérimentales compatibles. Notre approche déclarative [10] se distingue clairement de cette démarche essai/erreur : elle consiste à formaliser l'ensemble des connaissances disponibles pour représenter en *intention* tous les modèles satisfaisant toutes les données. En cas d'incohérence, elle prévoit une démarche automatique de redressement. Sinon, l'exploitation de la classe des modèles cohérents est entreprise, en particulier la déduction automatique de propriétés nouvelles intéressantes pour le biologiste.

Nous nous plaçons dans le cadre d'une modélisation discrète des réseaux géniques : les réseaux de Thomas. Ces réseaux sont asynchrones et multi-valués et permettent de décrire facilement, et de manière qualitative, la dynamique des réseaux géniques.

Nous présentons ici une mise en œuvre pour notre approche basée sur le paradigme logique Answer Set Programming (ASP)[12]. L'objectif est de montrer l'intérêt de ce paradigme fondé sur une logique non monotone en illustrant en particulier 1) son pouvoir d'expression propre à une rédaction structurée et à la prise en compte d'informations partielles et d'hypothèses biologiques sous forme de défauts, 2) ses capacités d'inférences intéressantes dans la mesure où les modèles considérés sont minimaux, 3) les performances de résolution similaires à celle de solveurs SAT, sans restriction de pouvoir d'expression.

L'organisation de ce papier est la suivante : dans la partie 2, nous décrivons le formalisme de Thomas et, dans la partie 3, le paradigme ASP. Nous présentons dans la partie 4 la modélisation ASP de réseaux de Thomas à l'aide d'un réseau simple. Dans la partie 5, nous montrons comment modéliser en ASP des hypothèses et des observations biologiques. La partie 6 est consacrée à la méthodologie de construction de modèles et à la mise en œuvre des fonctionnalités nécessaires. La partie 7 présente les résultats obtenus sur le plan biologique et sur celui des performances pour des modélisations de réseaux géniques réels.

2 Réseaux de Thomas

Nous présentons ici une description succincte des réseaux de Thomas [16], suffisante pour notre propos.

On trouvera dans [9][10] une description plus détaillée.

Ce type de réseau fournit une formalisation discrète des réseaux géniques très acceptée, dans la mesure où l'on peut dire qu'elle représente une abstraction des systèmes d'équations différentielles non-linéaires à base de sigmoïdes, décrivant l'évolution des concentrations de protéines produites à partir des gènes [16].

La structure des réseaux de régulation géniques est habituellement représentée par un graphe dirigé dit *d'interaction*, dans lequel les nœuds représentent les gènes et les arcs représentent les interactions entre gènes. Soient deux gènes i et j, connectés par un arc allant du gène j au gène i. Cet arc indique que le gène j peut potentiellement modifier le taux d'expression du gène i. Il est étiqueté par une valeur entière identifiant le seuil discret de j auquel une modification du taux d'expression du gène cible i se produit. Une interaction est médiée par une protéine régulatrice, chaque gène produisant une seule protéine.

La dynamique du réseau de gènes est décrite par un graphe dirigé dit de *transition*. Dans ce graphe, chaque nœud est un état du système et chaque arc est un passage possible d'un état vers un autre. Un état du système est décrit par une liste $S = [x_1, \ldots, x_n]$ contenant les concentrations des protéines associées aux n gènes du réseau. Les concentrations sont représentées par des valeurs discrètes correspondant à un intervalle entre deux seuils consécutifs du gène. Par exemple, si la variable x_i vaut 0, cela signifie que la concentration de la protéine codée par le gène i est inférieure au plus petit seuil de i.

Pour un état donné du système, la concentration vers laquelle tend chaque protéine est donnée par l'*état focal* associé à l'état courant du système. Nous notons X_i la *valeur focale* du gène i. L'état focal d'un état du système est donné par une liste $X = [X_1, \ldots, X_n]$ et représente l'état vers lequel *tend* le système dans l'état courant (ce n'est pas forcément le successeur de l'état courant).

Plus précisément, la valeur focale d'un gène i est déterminée par la concentration des protéine j tel qu'il existe un arc $j \rightarrow i$ dans le graphe d'interaction. Pour chaque gène, nous définissons un ensemble de contextes cellulaires : un contexte cellulaire d'un gène i est un ensemble d'états du système dans lesquels le taux de production de i est constant. Soient $j_1, ..., j_p$ les gènes agissant sur le gène i avec les seuils $\theta_{j_1}, ..., \theta_{j_p}$. L'ensemble des contextes cellulaires du gène i constitue une partition de l'espace des concentrations où chaque contexte cellulaire est défini par un ensemble d'inégalités : $x_{j_1} \mathsf{op_1} \theta_{j_1}, \ldots, x_{j_p} \mathsf{op_p} \theta_{j_p}$ où $\mathsf{op_1} \ldots \mathsf{op_p} \in \{<\ ,\ \geq\}$, x_{j_1} est la concentration du gène j_1 et θ_{j_1} est le seuil de j_1 auquel se produit une variation du taux d'expression de i. Dans l'exemple montré par la figure 1, au gène b correspond quatre contextes cellulaires :

- un premier tel que $x_a < \theta_a^1$ et $x_b < \theta_b^2$,
- un deuxième tel que $x_a < \theta_a^1$ et $x_b \geq \theta_b^2$,
- un troisième tel que $x_a \geq \theta_a^1$ et $x_b < \theta_b^2$,
- un quatrième tel que $x_a \geq \theta_a^1$ et $x_b \geq \theta_b^2$.

Au contexte cellulaire c_i d'un gène i est associé un paramètre cinétique correspondant à la focale de i. Nous désignons ce paramètre par $K_i^{dessus(c_i)}$ où $dessus(c_i)$ est l'ensemble des noms des gènes j_r influençant i et tels que x_{j_r}, dans le contexte cellulaire c_i, soit supérieur ou égal au seuil θ_{j_r} étiquetant l'arc $j_r \rightarrow i$. Dans notre exemple, les paramètres cinétiques correspondant aux contextes cellulaires présentés plus haut sont, dans l'ordre de présentation des contextes : K_b, K_b^b, K_b^a et K_b^{ab}.

Soit $contexte(S, c'_i)$ tel que pour, tout contexte cellulaire c'_i du gène i : $contexte(S, c'_i) = 1$ si l'état du système S appartient au contexte cellulaire c'_i, 0 sinon. La valeur focale de i est définie comme :

$$X_i = \sum_{c'_i} K_i^{dessus(c'_i)} * contexte\,(S, c'_i)$$

Les équations focales des deux gènes de l'exemple 1(a) sont montrées en 1(b).

Le formalisme de Thomas est un formalisme non déterministe. Si deux gènes tendent à changer de concentration, deux successeurs sont possibles : un pour lequel le premier gène change de valeur, et un pour lequel le deuxième gène change de valeur. En effet, dans la réalité, il est très peu probable que deux gènes franchissent leur seuil exactement au même moment. Pour un état donné S, l'ensemble des successeurs de S est déterminé en comparant la valeur d'expression de chaque gène avec sa valeur focale : $S' = [x'_1, \ldots, x'_n]$ est un successeur de S si :

- Soit il existe i tel que $X_i \neq x_i$, dans ce cas, pour tout j tel que $j \neq i$, $x'_j = x_j$ et (1) $x'_i = x_i + 1$ si $X_i > x_i$ ou (2) $x'_i = x_i - 1$ si $X_i < x_i$.
- Soit pour tout j, $X_j = x_j$, alors, pour tout j, $x'_j = x_j$: S est dit *stationnaire*.

3 Answer set programming

ASP [1][12] est un langage apparu vers la fin des années 1990 comme un paradigme déclaratif. Il est basé sur une logique non monotone définie à l'aide de la notion de modèles *stables*. Nous en faisons une présentation succincte ici.

Un programme logique ASP est un ensemble fini de règles de la forme :
$a_0 \leftarrow a_1, \ldots, a_m, not\ a_{m+1}, \ldots, not\ a_n$
où $0 \leq m \leq n$ et $\forall i \mid 0 \leq i \leq n,\ a_i$ est un atome. Pour

une règle r, $tete(r) = a_0$ est la tête de la règle, et $corps(r) = \{a_1, \ldots, a_m, not\ a_{m+1}, \ldots, not\ a_n\}$ est le corps de cette règle (i.e. la partie droite de la règle). Si $tete(r)$ est vide, r est une *contrainte d'intégrité*. Si $corps(r)$ est vide, r est un *fait*.

Soit A l'ensemble des atomes, $corps^+(r) = \{a{\in}A \mid a \in corps(r)\}$ et $corps^-(r) = \{a{\in}A \mid not\ a \in corps(r)\}$. Un ensemble $X \subseteq A$ est un *answer set* ou *modèle stable* d'un programme P si X est le modèle minimal du *réduit* $P^X = \{tete(r){\leftarrow}corps^+(r) | r{\in}P, corps^-(r) \cap X = \emptyset\}$. [1]

Exemple 1 : soit le programme E suivant :

```
a :- not b,c.
b :- not a.
c.
```

Soit $X = \{a, c\}$. Nous obtenons le réduit $E^X = \{c,\ a{\leftarrow}c\}$ dont le modèle minimal est $\{a, c\}$. X est un modèle stable de E.
Soit $X' = \{a, b, c\}$. Nous obtenons le réduit $E^{X'} = \{c\}$ dont le modèle minimal est $\{c\}$. X' n'est pas un modèle stable de E.

Exemple 2 : le programme E' suivant :

```
a :- not b.
b :- not a.
```

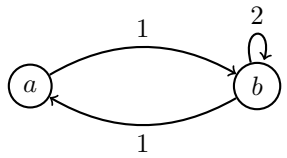
a deux modèles stables $\{a\}$ et $\{b\}$. Si nous ajoutons la contrainte d'intégrité `:- a.`, nous éliminons le modèle $\{a\}$. Si nous ajoutons la contrainte d'intégrité `:- not a.`, nous éliminons le modèle $\{b\}$ car il ne contient pas a.

Ce formalisme permet également d'exprimer des contraintes de cardinalité sur le nombre de littéraux vrais. Si nous imposons la contrainte u`{l_1,...,l_n}`v., nous n'obtenons que des modèles où le nombre de littéraux `l_i` vrais est compris entre u et v.

D'un point de vue opérationnel, le logiciel que nous utilisons [12] procède en deux étapes pour calculer les answer sets d'un programme P. Dans la première, un « grounder » qui substitue aux variables du programme des termes sans variables libres, produit un programme propositionnel $\mathbf{P}$ correspondant à P. Dans la seconde, un solveur calcule les answer sets de $\mathbf{P}$.

Le solveur que nous utilisons [12] propose l'opérateur `#maximize` (resp. `#minimize`) maximisant (resp. minimisant) le nombre de formules atomiques vraies dans un ensemble. Par exemple, si nous imposons `#maximize{f_1,...,f_n}`, nous obtenons un modèle contenant le plus grand nombre possible de formules atomiques `f_i` vraies. S'il existe plusieurs modèles, un seul d'entre eux est fourni.

1. Un ensemble de clauses de Horn ne contenant pas de littéral négatif possède toujours un modèle minimal (tel que si on lui soustrait un ou plusieurs atomes, on ne peut obtenir un modèle) et ce modèle est unique. Un réduit P^X correspond à un tel ensemble.



(a)

$$
\begin{array}{ll}
X_a = & K_a * s^-(x_b, \theta_b^1) + \\
& K_a^b * s^+(x_b, \theta_b^1) \\
X_b = & K_b * s^-(x_a, \theta_a^1)\ s^-(x_b, \theta_b^2) + \\
& K_b^a * s^+(x_a, \theta_a^1)\ s^-(x_b, \theta_b^2) + \\
& K_b^b * s^-(x_a, \theta_a^1)\ s^+(x_b, \theta_b^2) + \\
& K_b^{ab} * s^+(x_a, \theta_a^1)\ s^+(x_b, \theta_b^2)
\end{array}
$$

(b)

FIGURE 1 – (a) Exemple de graphe d'interaction d'un réseau de régulation génique. Le gène a active l'expression du gène b, tandis que le gène b inhibe l'expression du gène a. De plus, le gène b s'auto-active. L'étiquette « 1 » sur l'arc $b \rightarrow a$ indique que l'expression de a change lorsque la concentration de b traverse le premier seuil de b. (b) Les équations focales de cet exemple définissant l'état focal $[X_a, X_b]$ du système, lorsque celui-ci est dans un état $[x_a, x_b]$. $s^+(x, \theta)$ vaut 1 si $x \geq \theta$ et 0 sinon, et $s^-(x, \theta) = 1 - s^+(x, \theta)$

4 Modélisation en ASP d'un réseau de Thomas

Pour cette spécification, nous procédons en introduisant d'abord des règles donnant le domaine de valeur des atomes et, ensuite, des règles d'intégrité limitant le nombre de modèles possibles.

Par convention de nommage, les notations `predicat(X,Y)` et `predicat/2` sont équivalentes.

4.1 Graphe d'interaction

Les réseaux de régulation sont souvent représentés par un graphe d'interaction : la figure 1 en donne un exemple simple. Pour modéliser ce graphe, nous utilisons les prédicats `node(N)` vrai ssi `N` est un nœud du graphe d'interaction, `edge(N1,N2,No)` vrai ssi il existe un arc n° `No` allant du nœud `N1` au nœud `N2` (le paramètre `No` permet de modéliser des multi-arcs) et `threshold(N1,N2,No,T)` vrai ssi `T` est l'unique seuil, strictement positif, de l'arc n° `No` allant de `N1` à `N2`.

Le graphe d'interaction de l'exemple (Fig. 1) est modélisé par les formules atomiques suivantes : `node(a). node(b). edge(a,b,1). edge(b,a,1). edge(b,b,1). threshold(a,b,1,1). threshold(b,a,1,1). threshold(b,b,1,2).`

Si la valeur d'un seuil n'est pas imposée, l'ensemble des valeurs possibles pour ce seuil est considéré, chaque

valeur étant attribuée à au moins un modèle différent. Cette modélisation permet d'imposer des contraintes sur les seuils.

4.2 Dynamique du système

Un moyen de description de l'évolution des états du système est nécessaire pour saisir la dynamique du réseau. De plus, si nous possédons des données biologiques sur la dynamique, il est nécessaire de les représenter pour les inclure dans notre modèle.

L'évolution du système est traduite par des chemins de longueur finie. Pour cela, nous introduisons les prédicats `path(P)` vrai ssi `P` est un chemin, `length(L,P)` vrai ssi le chemin `P` est de longueur `L` et `step(I,P)` vrai ssi le chemin `P` possède une étape n° `I`. À chaque étape d'un chemin, le système est dans un état donné. La description d'un état se fait grâce au prédicat `species(N,V,I,P)` vrai ssi à l'étape `I` du chemin `P`, la concentration du gène `N` est `V`. Pour chaque gène, l'intervalle de concentrations possibles est défini grâce au prédicat `val(N,V)` vrai ssi `V` est une valeur possible de concentration pour `N`.

Définir la dynamique du système nécessite de spécifier l'ensemble des successeurs possibles d'un état. Comme décrit dans la partie 2, dans le formalisme utilisé, la concentration d'une protéine associée à un gène changeant de valeur varie selon la tendance donnée par la focale de ce gène. Pour cela nous introduisons le prédicat `focal(N,F,I,P)` vrai ssi, à l'étape `I` du chemin `P`, la valeur focale du gène `N` est `F`.

La valeur focale d'un gène est définie grâce aux paramètres cinétiques associés aux contextes cellulaires de ce gène. Pour modéliser ces paramètres cinétiques, nous distinguons l'identifiant d'un paramètre et sa valeur en introduisant les prédicats `param(N,Ik)` vrai ssi `Ik` est l'identifiant d'un paramètre cinétique du gène `N` et `kparam(K,Ik)` vrai ssi le paramètre d'identifiant `Ik` vaut `K`. Pour un gène donné, tous les paramètres cinétiques sont décrits par un identifiant pour exprimer des contraintes relatives aux interactions (voir § 5.2). Nous imposons (1) l'existence d'une et une seule valeur par paramètre cinétique et (2) que cette valeur soit comprise entre 0 et la valeur du plus grand seuil du gène en question. Pour imposer (1), nous utilisons la contrainte de cardinalité suivante :

```
1{kparam(K,Ik):val(N,K)}1 :-
                param(N,Ik), node(N).
```

Cette contrainte impose que, pour un identifiant `Ik`, parmi toutes les valeurs possibles de `K`, il y ait une et une seule formule atomique `kparam(K,Ik)` qui soit vraie. Pour imposer (2), nous utilisons la contrainte d'intégrité suivante :

```
:- kparam(K,Ik), param(N,Ik), val(N,K),
    not threshold(N,N2,Ne,K):edge(N,N2,Ne),
    node(N), K!=0.
```

Dans notre exemple, le gène `a` possède deux contextes cellulaires, un pour lequel `b` est en dessous de son premier seuil, et un pour lequel `b` est au dessus de ce seuil. Les identifiants des paramètres cinétiques correspondant à chaque contexte sont définis par les formules atomiques suivantes : `param(a,ka_).` et `param(a,ka_b).` Pour le gène `b`, la contrainte (2) impose que les valeurs possibles vers lesquelles tend ce gène sont comprises entre 0 et 2 puisque `b` possède deux seuils.

Grâce aux prédicats `param/2` et `kparam/2`, nous pouvons définir les règles calculant la focale d'un gène selon son contexte cellulaire. Dans notre exemple, la focale du gène `a` dans le contexte cellulaire où `b` est au dessus de son seuil est déterminée par la règle :

```
focal(a,Ka_b,I,P) :- species(b,Vb,I,P),
  above(b,a,A,Vb), step(I;I+1,P),
  param(a,ka_b), kparam(Ka_b,ka_b).
```

Où le prédicat `above(A,B,No,V)` est vrai ssi `V` est supérieur ou égal au seuil de l'arc `No` allant de `A` à `B`.

Le formalisme de Thomas étant non-déterministe, la concentration d'au plus une protéine change de valeur. De plus, si la focale d'un gène est différente de la concentration de la protéine associée, l'état du système n'est pas un état stationnaire. Dans notre modélisation, nous assurons simplement ces propriétés en utilisant le prédicat `diff(N,I,P)` vrai ssi la protéine `N` est la protéine dont la concentration change selon la tendance de sa focale à l'étape `I` du chemin `P`. La contrainte de cardinalité suivante :

```
0{diff(N,I,P):node(N)}1 :- step(I;I+1,P).
```

impose qu'au plus une seule concentration change entre un état et son successeur. En introduisant le prédicat `foceg(N,I,P)` vrai ssi la concentration de la protéine `N` est égale à sa valeur focale à l'étape `I` du chemin `P`, la contrainte d'intégrité :

```
:- 1{not foceg(N,I,P):node(N)},
    0{diff(N,I,P):node(N)}0, step(I;I+1,P).
```

permet d'assurer qu'en cas d'égalité d'un état et de son état focal, l'état en question est stationnaire.

Pour l'exemple présenté dans la figure 1, nous obtenons huit instanciations différentes des paramètres cinétiques. Ces huit instanciations correspondent à six graphes de transitions différents, nommés G_1 à G_6, présentés dans la figure 2.

5 Modélisation en ASP d'observations et d'hypothèses biologiques

Une fois le modèle de Thomas représenté en ASP, nous devons modéliser les données biologiques.

5.1 Chemins

Les données sur le comportement du réseau peuvent, en général, s'exprimer par des propriétés sur des chemins. Nous pouvons imposer des contraintes sur les concentrations des protéines dans les étapes d'un chemin de deux manières différentes :

- Nous pouvons imposer des contraintes « instanciées » : par exemple, imposer que la concentration d'une protéine pour une étape donnée d'un chemin ait une valeur donnée. Pour cela, nous utilisons une contrainte d'intégrité sur un atome `species(N,V,I,P)`.
- Nous pouvons également imposer des contraintes « non instanciées », comme dans le cas ci-dessus mais avec, par exemple, `V` restreint à `V<2` ou encore, comme l'existence d'un état stationnaire.

Pour exprimer l'existence d'un état stationnaire, nous imposons l'existence d'un chemin de longueur 2 telle que l'état du système à l'étape 1 et à l'étape 2 soit le même. Pour cela, nous définissons le prédicat `statpath(P)` vrai ssi `P` est de longueur 2 et est stationnaire. Pour implémenter ce prédicat, nous définissons le prédicat `succeg(N,P)` vrai ssi `P` est un chemin de longueur 2 et la concentration du gène `N` est la même à l'étape 1 et à l'étape 2. Grâce à ce prédicat, nous définissons `statpath/1` avec à la contrainte d'intégrité suivante :

```
:- statpath(P), 1{not succeg(N,P):node(N)}.
```

qui impose que pour un chemin `chemin`, il est impossible que `statpath(chemin)` soit vrai et qu'il existe un gène dont la concentration ne soit pas la même dans les deux étapes de `chemin`.

Nous pouvons remarquer que nous n'avons, ici, imposé aucune valeur de concentration des protéines, mais nous avons pu imposer que cette valeur soit la même dans les deux étapes du chemin grâce à la contrainte d'intégrité définissant `statpath/1`.

Ces contraintes traduisant les observations biologiques permettent de rejeter toutes les instanciations ne vérifiant pas ces observations.

Plusieurs logiques temporelles exprimant des propriétés de chemins ont été conçues, en particulier CTL (Computational Tree Logic)[7] proposée dans le cadre des réseaux biologiques [2][6]. Intuitivement, elle permet de formuler des propriétés du type : « il existe un (resp. tout) chemin possédant (resp. possède) telle caractéristique » où une caractéristique est une propriété de tous les états de chemin ou d'un au moins. Par exemple, dans le cadre de l'exemple simple (Fig. 1), la formule $(a = 0 \wedge b = 0) \Rightarrow EF(a = 0 \wedge b = 2)$ signifie qu'il doit exister (EF : Exist Future) un chemin débutant dans un état où $a = 0$ et $b = 0$ et atteignant un état où $a = 0$ et $b = 2$. Cette propriété est facilement imposée en ASP en introduisant un chemin `p` de taille maximum 5 sous la forme suivante :

```
path(p). length(5,p).
 :- not species(a,0,1,p).
 :- not species(b,0,1,p).
exist_path :- species(a,0,I,p),
        species(b,2,I,p), step(I,p).
```

et en utilisant la contrainte d'intégrité :

```
 :- not exist_path.
```

Seuls les modèles G_4 et G_6 satisfont cette formule.

On note que le nombre d'atomes de la forme `species(N,V,I,P)` reste linéaire en fonction de la taille du chemin `P`.

Vérifier une propriété CTL universelle est théoriquement possible, mais est peu intéressant, car, du fait de l'abstraction utilisée [16], si nous imposons une propriété de ce type, certains modèles peuvent être rejetés à tort.

5.2 Signes des interactions

Dans les papiers utilisant le formalisme de Thomas, tous les arcs du graphe d'interaction sont étiquetés par un signe « + » ou « − ». Si un arc $j \to i$ est étiqueté par un signe « + » (resp. un signe « − »), cela indique intuitivement que le gène j active (resp. inhibe) le gène i au moins dans un contexte cellulaire. Néanmoins, le sens exact des termes activation et inhibition n'est pas défini précisément, et plus particulièrement dans le cas où d'autres gènes régulent aussi l'expression de i : est-ce que l'activation de i par j empêche l'inhibition de i par j ? Est-ce que, quelles que soient les interactions que i subit, j active toujours i ? Nous modélisons formellement ces questions en exprimant deux types de contraintes :

Le premier type de propriété que nous considérons traduit l'observation biologique d'une activation (resp. inhibition) de i par j. Nous modélisons cette propriété de la façon suivante : nous imposons qu'il existe deux contextes cellulaires $c1_i$ et $c2_i$ de i tels que :

- dans $c1_i$, la concentration de j est inférieure au seuil θ_j,
- $c2_i$ est identique à $c1_i$ excepté pour j, dont la concentration est supérieure à θ_j,

alors, les paramètres cinétiques associés à ces contextes cellulaires sont tels que, si j active (resp. inhibe) i, la tendance de i dans $c1_i$ est strictement inférieure (resp. strictement supérieure) à la tendance de i dans $c2_i$. De plus, si un gène i s'auto-active (resp. s'auto-inhibe) avec un seuil θ_i, sa valeur focale, dans le contexte cellulaire où i est supérieur (resp. inférieur) à θ_i doit être supérieure (resp. inférieure) à ce seuil, sinon, le système arrive toujours dans un état où la concentration de i est inférieure (resp. supérieure) à θ_i et l'effet de l'auto-activation n'est pas observable.

Pour exprimer cette propriété, nous imposons des contraintes dites *contraintes d'observabilité*. Pour notre exemple, pour le gène b, les contraintes d'observabilité sont :
$(K_b < K_b^a) \vee (K_b^b < K_b^{ab})$ et
$((K_b < K_b^b) \wedge (K_b^b \geq 2)) \vee ((K_b^a < K_b^{ab}) \wedge (K_b^{ab} \geq 2))$.

Le deuxième type de propriété que nous modélisons exprime que pour tout couple de contextes cellulaires $c1_i$, $c2_i$ de i, satisfaisant aux mêmes contraintes que précédemment, alors, dans le cas d'une activation (resp. inhibition), la tendance de i dans le contexte cellulaire $c1_i$ n'est pas supérieure (resp. inférieure) à la tendance de i dans le contexte cellulaire $c2_i$. Intuitivement, cela signifie qu'il est impossible d'avoir deux paramètres cinétiques pour $c1_i$ et $c2_i$ qui vérifieraient la contrainte d'observabilité imposée par j inhibe (resp. active) i. Cela signifie également que, dans le cas ou deux gènes régulent de la même manière un troisième gène (positivement ou négativement), la régulation combinée de ces deux gènes est au moins aussi forte que la régulation de chaque gène individuellement.

Nous exprimons cette propriété par des contraintes dites *contraintes d'additivité*. Dans notre exemple, les contraintes d'additivité portant sur b sont : $K_b \leq K_b^b$, $K_b^a \leq K_b^{ab}$, $K_b \leq K_b^a$ et $K_b^b \leq K_b^{ab}$.

Les contraintes d'additivité représente un cas typique de règles « générales » en ce qu'elles s'appliquent sauf exception. Par exemple, comme il a été dit, il est assez exceptionnel que la régulation combinée de deux gènes sur un troisième ne soit pas au moins aussi forte que la régulation de chaque gène individuellement. Cependant, biologiquement, le cas est possible : par exemple, si les sites de fixation sur l'ADN des protéines correspondantes se chevauchent. La technologie ASP se prête très bien à la formalisation de ce type de connaissance dans la mesure où elle est basée sur une logique non monotone permettant l'expression de défauts [4], plus précisément ici de défauts *normaux* exprimant que « telle contrainte d'additivité est vraie jusqu'à preuve du contraire ». Un tel défaut pour la contrainte d'additivité $K_b \leq K_b^a$ se formule ainsi :
`addit(b,kb_,kb_a) :- not -addit(b,kb_,kb_a).`
où : `addit(b,kb_,kb_a)` implique $K_b \leq K_b^a$,
`-addit(b,kb_,kb_a)` et `addit(b,kb_,kb_a)`
sont exclusifs (i.e.
`:- -addit(b,kb_,kb_a), addit(b,kb_,kb_a).`),
`-addit(b,kb_,kb_a)` est impliqué par $K_b > K_b^a$. En d'autres termes, si $K_b > K_b^a$ n'est pas démontrable alors $K_b \leq K_b^a$ est vrai.

5.3 Mutants

Pour étudier des réseaux géniques, les biologistes sont amenés couramment à supprimer un gène ou à faire en sorte qu'il soit sur-exprimé. Les nouveaux réseaux obtenus sont dits *mutants* par opposition au réseau initial dit *sauvage*. Dans cette partie, nous entendons par « modèle » un réseau qui peut être sauvage ou mutant. Le problème consiste à définir un modèle mutant d'un réseau sauvage et à exprimer les propriétés des différents modèles.

La démarche procède par extension. Elle consiste à introduire les prédicats `model(M)` vrai ssi `M` est un modèle, `mutant(N,M,V)` vrai ssi, dans le modèle `M`, `N` est un gène mutant dont la valeur d'expression est forcée à `V` et `mutant(N,M)` vrai ssi `N` est un gène mutant du modèle `M`.

Pour différencier les chemins du modèle sauvage des chemins d'un modèle mutant, nous étendons le prédicat `path/1` en `path(P,M)` vrai ssi `P` est un chemin dans le modèle `M`.

Il est proposé dans [9] de définir de nouveaux paramètres cinétiques pour chaque mutant et d'imposer l'égalité des paramètres entre le modèle sauvage et le modèle mutant pour les gènes qui ne sont pas mutants. Cette approche est simple, mais présente l'inconvénient de devoir créer beaucoup de paramètres.

Le fait d'utiliser une logique non-monotone permet d'éviter de définir des paramètres cinétiques supplémentaires et de n'écrire que des équations focales spécifiques pour chaque mutant : en effet, il suffit d'utiliser les équations focales du modèle sauvage, sauf pour les gènes présentant une mutation. Nous implémentons ceci en rajoutant en partie droite des règles définissant les focales, le littéral `not mutant(N,M)`. Cela indique que nous utilisons ces règles, sauf dans les cas où le gène `N` est un mutant dans le modèle `M`. Nous définissons alors, la valeur focale d'un gène mutant comme étant sa valeur de mutation, de plus, nous imposons que, dans l'état initial des chemins, la concentration des gènes mutants soit celle imposée par la mutation.

6 Traitement d'incohérence et déduction de propriétés

La stratégie utilisée dans notre approche prévoit de définir un ensemble initial de contraintes traduisant les équations focales, les contraintes d'additivité et d'observabilité, les données biologiques, et des hypothèses sur le fonctionnement du réseau.

À partir de cet ensemble initial, soit nous obtenons au moins un modèle, soit nous n'en obtenons aucun. Dans ce dernier cas, cela signifie qu'il y a une incohérence entre le graphe d'interaction, les données biologiques et les hypothèses. Nous séparons les contraintes en deux catégories : celles qui sont soutenues par des données biologiques et donc non relâchables, et celle qui ne le sont pas (hypothèses) ou qui sont

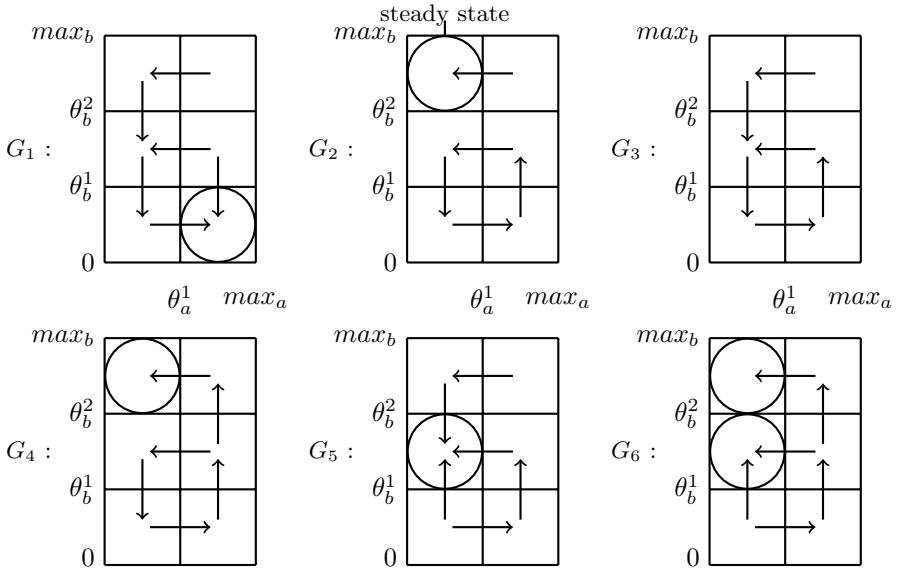


FIGURE 2 – Graphes de transition G_1, ..., G_6 satisfaisants l'ensemble des contraintes d'observabilité et d'additivité du réseau présenté par Fig. 1. Chaque case correspond à un état du système. Les états stationnaires sont indiqués par un rond. Les flèches représentent les transitions possibles. Chaque graphe correspond à une ou deux instanciations des paramètres cinétiques. Par exemple, le graphe G_4 correspond à l'instanciation suivante : $K_a = 1, K_a^b = 0, K_b = 0, K_b^b = 2, K_b^a = 2, K_b^{ab} = 2$.

moins fiables. Le but est alors de trouver le plus petit ensemble de contraintes à relâcher dans l'ensemble des contraintes relâchables pour que l'ensemble de contraintes restantes soit cohérent.

Pour cela, nous devons procéder en deux étapes :

- premièrement, la détermination du nombre minimum de contraintes relâchables. Considérons que nc contraintes soient relâchables. Soit `cv_`i, $i \in [1..nc]$ vrai ssi la i^e contrainte relâchable est vrai. La commande `#maximize{cv_1,...,cv_n}` permet d'obtenir le nombre N maximum de contraintes pouvant être respectées.
- deuxièmement, comme nous disposons du nombre N, nous posons la contrainte de cardinalité : N`{cv_1,...,cv_n}`N.

S'il existe différents ensembles de contraintes à relâcher menant à la consistance, nous procédons à une analyse manuelle de ces ensembles, afin d'évaluer leur pertinence biologique et considérons chacun de ces ensembles, sauf ceux dont la relaxation n'aurait pas de sens biologique. Il parait néanmoins difficile de fournir plus d'une dixaine d'ensembles de contraintes à relâcher à un biologiste sans fournir des éléments d'analyse supplémentaires

Il faut noter que le redressement d'incohérence est toujours gênant, car il nécessite un cadre para-logique. Ici, il peut être simplement évité en représentant les hypothèses sous la forme de défauts (voir §5.2) : par exemple, si effectivement $K_b > K_b^a$ est prouvé, un modèle différent de ceux correspondant aux graphes de la Fig. 2 est proposé. L'intérêt de rester dans le même cadre logique est clair : au lieu d'un résultat d'incohérence, nous obtenons une indication sur les hypothèses contredites et sur les conséquences de leur remise en cause.

À partir d'un ensemble de contraintes cohérentes, notre approche propose plusieurs fonctionnalités, telle que la déduction de propriétés vraies dans tous les modèles ou la recherche du nombre minimal de seuils.

La déduction automatique de propriétés est mise en œuvre facilement grâce à l'option `--cautious` proposée par [12]. Cette option renvoie l'ensemble des formules atomiques vraies dans tous les modèles. En spécifiant un langage décrivant les propriétés que nous recherchons, nous obtenons l'ensemble des propriétés vraies pour tous les modèles.

Un tel langage peut être construit à l'aide de clauses [5]. Inférer des propriétés revient alors à inférer des clauses. Il est toutefois critique de choisir pertinemment l'ensemble de formules atomiques. En [10], les auteurs proposent deux langages permettant d'exprimer des disjonctions d'inégalités entre paramètres cinétiques, comme $K_b < K_b^a \vee \neg(K_b < K_b^{ab})$.

Par exemple, supposons que nous voulions obtenir

toutes les clauses de taille 2 portant sur des atomes qui sont des inégalités strictes sur des paramètres cinétiques d'un même gène. Il suffit pour cela de définir le prédicat `clauses2(In,Ik1,Ik2,Inp,Ik1p,Ik2p)` où `In` et `Inp` représentent des opérateurs d'inégalité (non strictes) et où `Ik1` et `Ik2` (resp. `Ik1p` et `Ik2p`) sont des identificateurs de paramètres cinétiques d'un même gène, vrai si une clause formée à partir des deux atomes correspondant est démontrée.

Il faut noter la spécificité et l'intérêt de la technologie ASP en cette matière d'inférence de propriétés communes à un ensemble de modèles. On sait que les modèles obtenus sont minimaux en ce sens que soustraire un ou plusieurs atomes à un modèle ne peut donner un modèle. Par exemple, `a :- not b.` `b :- not a.` n'a que deux modèles : {`a`} et {`b`}. Le modèle {`a`, `b`} n'est pas minimal. Il en ressort que le nombre de propriétés déductibles est égal ou supérieur à celui atteint en logique classique. Ainsi l'exemple précédent permet de conclure à l'exclusion de `a et b` (en l'absence d'autre information sur `a` et `b`), ce qui ne serait pas le cas si le seul axiome était l'union classique de `a` et `b`.

La recherche du nombre minimum de seuils est mise en œuvre simplement en utilisant l'opérateur `minimize` proposé par [12]. En effet, en définissant le prédicat `threshold_max(N,T)` vrai ssi `T` est le nombre de seuils du gène `N`, nous obtenons le minimum du nombre total de seuils en utilisant cet opérateur ainsi :

```
#minimize
    [threshold_max(N,T):val(N,T):node(N)=T].
```

À chaque atome `threshold_max(N,T)` vrai, est associé le poids `T` et il s'agit de minimiser la somme de ces poids.

7 Applications et Résultats

Nous présentons dans cette section deux applications illustrant les fonctionnalités proposées par notre approche.

7.1 Stress nutritionnel de *E. coli*

Escherichia coli est une bactérie dont la population croit exponentiellement en présence de nourriture. Dans le cas d'un stress nutritionnel, la bactérie stoppe cette croissance et entre dans une phase dite stationnaire où elle ne se reproduit plus. La réponse à ce stress est réversible : si la source nutritionnelle est à nouveau disponible, la bactérie retourne en phase exponentielle. Cette adaptation à l'environnement est due à une modification de la concentration des gènes de cette bactérie en fonction de la présence ou de l'absence de nourriture.

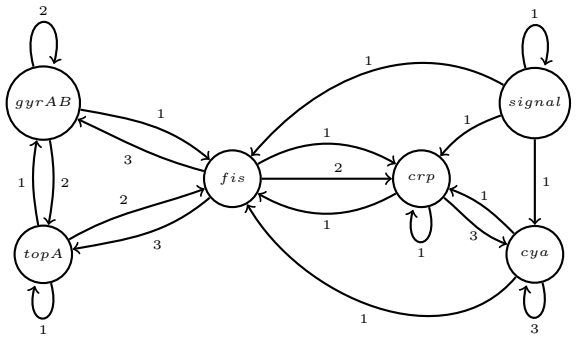


FIGURE 3 – Graphe d'interaction du réseau de régulation génique d'E. coli.

L'article [14] présente un ensemble d'observations biologiques et propose un réseau de gènes issu de ces observations. Le graphe d'interaction de ce réseau est présenté dans la figure 3. Une étude avec une approche déclarative de ce réseau est présentée en [10]. Nous avons repris cette étude avec notre mise en œuvre ASP. Nous illustrons, ici, la relaxation de contraintes due à une incohérence dans les données.

Parmi les contraintes portant sur ce réseau, on trouve, en particulier :

- l'existence de deux états stationnaires partiellement connus correspondant à l'état stressé et non-stressé de la bactérie.
- une contrainte portant sur les concentrations des gènes dans les états stationnaires
- l'existence de deux chemins, chacun menant d'un état stationnaire à l'autre

Nous modélisons ce réseau de régulation dans notre formalisme et nous constatons qu'il n'existe pas d'instanciation des paramètres rendant ce système cohérent. Il est alors intéressant de rechercher le plus petit ensemble de contraintes à relâcher pour obtenir un modèle. Les contraintes d'additivité n'étant pas soutenues par des données expérimentales, nous recherchons donc à relâcher des contraintes d'additivité.

Il apparaît que la relaxation d'une contrainte parmi $\{C_{gyrAB1}, C_{topA1}\}$ est suffisante. $C_{gyrAB} = K^{fis}_{gyrAB} \leq K_{gyrAB}$ est une contrainte d'additivité portant sur le gène *gyrAB*. $C_{topA} = K_{topA} \leq K^{fis}_{topA}$ est une contrainte d'additivité portant sur le gène *topA*. Une analyse biologique montre que cette dernière, à la différence de C_{gyrAB}, peut être tout à fait relâchée dans l'état des connaissances.

L'étude menée en [10] est basée sur une mise en œuvre réalisée à l'aide de l'outil GNBox [8] qui assure une coopération entre un solveur de contrainte

Prolog et un solveur SAT. La modélisation d'un réseau de Thomas est traduite sous forme clausale ainsi que certaines contraintes biologiques. D'un point de vue performance, on note en [10] qu'un temps très significatif (25 minutes pour la traduction en clauses et la résolution) est nécessaire pour déterminer les contraintes relâchables citées ci-dessus, dans un cas où toutes les contraintes d'additivité sur le gène *crp* sont supprimées. On trouve en [9] un temps meilleur (sept secondes ; sur un laptop, 2.4GHz, 2Go de RAM) pour la même tâche avec l'outil GNBox amélioré où toutes les contraintes sont traduites sous forme clausale. Avec notre implémentation, cette tâche est accomplie en moins de quatre secondes (avec un Core 2 Duo 3GHz, 4Go de RAM).

Il en ressort tout l'intérêt de disposer d'un langage (ASP), d'une part pourvu d'un pouvoir d'expression intéressant puisque comparable à celui de Prolog, dans lequel nous exprimons toutes les contraintes, et d'autre part, menant à des exécutions performantes, comparables à celles de solveurs SAT.

7.2 Extension du réseau de *E. coli*

En [14], les auteurs présentent un modèle instancié du réseau de gènes de *E.coli*, mais ce modèle ne reproduit pas l'ensemble des observations biologiques connues. En [13], ils étendent ce réseau pour respecter ces observations. Le nouveau réseau comporte trois gènes supplémentaires : $rpos$, $rssB$ et $gyrI$. Il comporte 9 gènes et 26 interactions. Les seuils de ces nouvelles interactions ne sont pas toujours connus, et notamment, ceux du gène $rpos$, qui agit sur trois autres gènes.

Nous avons modélisé ce réseau et adapté les contraintes imposées au réseau initial. Nous nous intéressons à l'existence d'une instanciation respectant l'ensemble des contraintes.

Si nous imposons que les chemins soient de longueur maximale 7, qui est la plus petite longueur possible, nous n'obtenons pas de modèles. Par contre, si nous permettons que les chemins soient de longueur 8, nous obtenons des modèles.

Dans le cas des chemins de longueur 7, la recherche du plus petit ensemble de contraintes à relâcher pour obtenir des modèles indique qu'il faut relâcher une contrainte parmi $\{C_{gyrAB1}, C_{topA1}, C_{gyrAB2}\}$, C_{gyrAB2} étant une contrainte exprimant que $K_{gyrAB}^{fisgyrI} \leq K_{gyrAB}^{fis}$, dont il reste à examiner la pertinence biologique. Ce résultat est obtenu en moins de cinq secondes (avec un Core 2 Duo 3GHz).

Dans ce réseau, le gène $rpos$ agit sur les gènes $gyrI$, $topA$ et $rssB$, mais, ne possédant aucune information sur les seuils de ces interactions, nous considérons que $rpos$ possède trois seuils, un pour chaque interaction. Nous nous intéressons au nombre minimum nécessaire de seuils de ce gène permettant la cohérence. Pour des chemins de longueur 8, nous trouvons que l'existence de seulement deux seuils suffit pour obtenir des modèles. Pour les chemins de longueur 7, nous obtenons les même résultats en relâchant une contrainte parmi $\{C_{gyrAB1}, C_{topA1}, C_{gyrAB2}\}$.

8 Conclusion

Nous avons présenté une nouvelle modélisation des réseaux de régulation génique utilisant le paradigme logique ASP et l'intérêt de ce paradigme pour ce faire. Le bien fondé de l'usage méthodique des règles ASP pour introduire les domaines de valeur des atomes nécessaires et ensuite celui des règles d'intégrité pour limiter l'ensemble des modèles possibles est illustré en 4. La représentation des hypothèses biologiques exposées en 5 montre l'importance de disposer de défauts normaux. Enfin, nous signalons en 6 comment, aussi à l'aide de défauts, éviter un processus para-logique dans le cas d'incohérence et comment inférer d'une façon très simple de nouvelles propriétés biologiques considérées comme des formules vraies dans tous les answer sets. Les aspects pratiques abordés en 7 confirment de bonnes performances sur des réseaux réels. Celles-ci dépendent évidemment du nombre n d'espèces et du demi-degré intérieur en moyenne f du graphe d'interaction. Les contraintes de succession sont proportionnelles à n et celles relatives à l'existence d'un chemin à la longueur du chemin. Le nombre de paramètres cinétiques est de l'ordre de 2^f, mais en général f est faible (vaut environ 3). Il reste que pour des réseaux de taille importante (plus de 20 espèces), une approche par décomposition/composition semble la plus appropriée, à la fois sur le plan des performances et de la compréhension de leur fonctionnement [3].

À notre connaissance, très peu d'équipes abordent l'analyse des réseaux de Thomas avec une approche de notre type [2][6]. Souvent, leurs travaux sont basés sur des outils de Model-Checking. Ces outils ont pour objectif de vérifier qu'un système de transition instancié satisfait des formules CTL (ou logique temporelle similaire). Aussi, de par cette origine, ils sont limités au regard des fonctionnalités présentées ici. Par exemple, du fait de la définition de CTL, il n'est pas possible d'exprimer l'existence de deux états stationnaires différents a priori non connus. Où encore, du fait de leur finalité, le relâchement automatique de contraintes ou la déduction de propriétés ne sont pas proposés par ces outils.

La technologie ASP est utilisée par ailleurs pour résoudre des problèmes d'inconsistance touchant aussi des réseaux biologiques [11][15]. Ces travaux se dis-

tinguent de ceux présentés ici dans la mesure où ils prennent seulement en compte l'aspect « statique » des réseaux : ils se consacrent à l'étude des interactions entre espèces biologiques et non pas à l'évolution temporelle d'un système biologique.

Nous nous intéressons maintenant à une formalisation étendue des réseaux biologiques intégrant des réactions métaboliques. Il s'agit de modéliser des réactions de nature différente dans un seul réseau. En effet, les réactions métaboliques possèdent des propriétés stœchiométriques que ne possèdent pas les interactions géniques. Ce formalisme étendu permet la modélisation de problèmes biologiques complexes comme l'homéostasie du fer dans les cellules mammifères. Un autre axe de recherche est l'aide au choix d'expériences.

Remerciements

Nous remercions M. Gebser et T. Schaub pour des échanges fructueux que nous avons eus et leurs conseils.

Ce travail a été soutenu par Microsoft Research à travers son programme de financement du doctorat de N.M.

Références

[1] Chitta Baral. *Knowledge Representation, Reasoning, and Declarative Problem Solving*. Cambridge University Press, New York, NY, USA, 2003.

[2] Gilles Bernot, Jean-Paul Comet, Adrien Richard, and Janine Guespin. Application of formal methods to biological regulatory networks : extending Thomas' asynchronous logical approach with temporal logic. *Journal of Theoretical Biology*, 229(3) :339 – 347, 2004.

[3] Gilles Bernot and Fariza Tahi. Behaviour preservation of a biological regulatory network when embedded into a larger network. *Fundam. Inf.*, 91 :463–485, August 2009.

[4] Philippe Besnard. *An Introduction to Default Logic*. Springer, 1989.

[5] Geneviève Bossu and Pierre Siegel. Saturation, nonmonotonic reasoning and the closed-world assumption. *Artif. Intell.*, 25 :13–63, January 1985.

[6] Nathalie Chabrier and François Fages. Symbolic model checking of biochemical networks. In *Computational Methods in Systems Biology*, Lecture Notes in Computer Science.

[7] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. *LNCS*, 131 :52–71, 1982.

[8] Fabien Corblin, Lucas Bordeaux, Eric Fanchon, Youssef Hamadi, and Laurent Trilling. Connections and integration with SAT solvers : A survey and a case study in computational biology. In Panos M. Pardalos, Pascal van Hentenryck, and Michela Milano, editors, *Hybrid Optimization*, volume 45 of *Optimization and Its Applications*, pages 425–461. Springer New York, 2011.

[9] Fabien Corblin, Éric Fanchon, and Laurent Trilling. Applications of a formal approach to decipher discrete genetic networks. *BMC Bioinformatics*, 11(1), 2010.

[10] Fabien Corblin, Sébastien Tripodi, Éric Fanchon, Delphine Ropers, and Laurent Trilling. A declarative constraint-based method for analyzing discrete genetic regulatory networks. *Biosystems*, 98 :91–104, 2009.

[11] Martin Gebser, Carito Guziolowski, Ivanchev M., Torsten Schaub, Anne Siegel, Philippe Veber, and Sven Thiele. Repair and Prediction (under Inconsistency) in Large Biological Networks with Answer Set Programming. In *Principles of Knowledge Representation and Reasoning : Proceedings of the Twelfth International Conference, KR 2010*, Toronto Canada, 2010. AAAI Press.

[12] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, and Sven Thiele. A user's guide to gringo, clasp, clingo, and iclingo (version 3.x). Octobre 2010.

[13] Pedro T. Monteiro, Delphine Ropers, Radu Mateescu, Ana T. Freitas, and Hidde de Jong. Temporal logic patterns for querying dynamic models of cellulat interaction networks. *Bioinformatics*, 24 :227–233, 2008.

[14] 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, 2006.

[15] Torsten Schaub and Sven Thiele. Metabolic network expansion with answer set programming. In Patricia Hill and David Warren, editors, *Logic Programming*, volume 5649 of *Lecture Notes in Computer Science*, pages 312–326. Springer Berlin / Heidelberg, 2009.

[16] René Thomas and Marcelle Kaufman. Multistationarity, the basis of cell differentiation and memory. II. logical analysis of regulatory networks in terms of feedback circuits. *CHAOS*, 11(1) :180–195, 2001.