Modélisation de réseaux biologiques discrets en Programmation Logique par Contraintes (PLC)

Fabien Corblin [*] [**] — **Eric Fanchon** [**] — **Laurent Trilling** [*]

[*] *IMAG-LSR, Université Joseph Fourier BP 53, 38041 Grenoble Cedex 9, France*
Fabien.Corblin@imag.fr Laurent.Trilling@imag.fr

[**] *LCM, Institut de Biologie Structurale Jean Pierre Ebel,*
CEA-CNRS-Université Joseph Fourier,
41, rue Jules Horowitz, 38027 Grenoble Cedex 1, France
Eric.Fanchon@ibs.fr

RÉSUMÉ. *Des outils informatiques sont nécessaires au développement de la biologie systémique pour analyser qualitativement la dynamique des réseaux d'interaction. Dans ce contexte, notre objectif est de développer un outil autour d'une spécification unique qui permette aux biologistes dans un contexte de connaissances incomplètes et qualitatives, (i) d'inférer des modèles à partir de propriétés comportementales observées; (ii) de déterminer des caractéristiques dynamiques qualitatives comme les états stationnaires, les séparatrices entre bassins d'attraction, les cycles; (iii) d'effectuer des simulations sur la base de modèles partiellement connus. Notre travail est basé sur les réseaux logiques multivalués asynchrones proposés par R. Thomas et E. Snoussi (1989, 1993). Ce formalisme, qui peut être vu comme une abstraction discrète d'une classe spéciale d'équations différentielles affines par morceaux, permet une analyse qualitative du comportement dynamique de tels systèmes. Récemment, ce formalisme a été étendu par de Jong et al. (2004) pour tenir compte de trajectoires qui glissent le long des seuils de discrétisation ("modes glissants"). Nous montrons qu'une telle description formelle d'un réseau biologique peut être facilement exploitée par l'intermédiaire d'une réalisation en Programmation Logique avec Contraintes (PLC) afin d'obtenir les différentes fonctionnalités désirées. Dans cet article, nous présentons la formalisation des réseaux de Thomas-Snoussi étendus et la mise en œuvre de celle-ci en langage déclaratif Prolog IV. Nous illustrons cette approche à l'aide de deux applications à des systèmes biologiques.*

ABSTRACT. *Computer tools are needed in systems biology to analyse qualitatively the dynamics of interaction networks. In this context, our objective is to develop a tool built on a single specification allowing (i) to infer models from observed properties which can be incomplete*

and qualitative; (ii) to determine qualitative dynamical features like steady states, separatrix between basins of attraction, cycles; (iii) to perform simulations on the basis of partially known models. Our work is based on the multivalued asynchronous networks proposed by R. Thomas and E. Snoussi (1989, 1993). This formalism, which can be seen as a discrete abstraction of a special class of piecewise affine differential equations, allows a qualitative analysis of the dynamic behavior of such systems. This formalism has been recently extended by de Jong et al. (2004) to take into account trajectories which slide along discretization thresholds ("sliding modes"). We show that such a formal description of a biological switching network can be easily exploited through an implementation in Constraint Logic Programming (CLP) in order to obtain the variety of functionalities desired. We present here the formalization of extended Thomas-Snoussi networks and their implementation in the declarative language Prolog IV. Then we illustrate the flexibility of the approach with two biological applications.

MOTS-CLÉS : *Programmation Logique avec Contraintes, inférence, simulation, réseaux biologiques, réseaux logiques multivalués asynchrones, abstraction discrète.*

KEYWORDS: *Constraint Logic Programming, inference, simulation, biological networks, multivalued asynchronous logical networks, discrete abstraction.*

1. Introduction

Il est à présent clair que la recherche en biologie est entrée dans une nouvelle ère dans laquelle les composants moléculaires doivent être assemblés au sein d'une description systématique afin d'atteindre de nouveaux niveaux de compréhension. L'élucidation du fonctionnement des réseaux d'interaction passe par des moyens informatiques aidant le biologiste à raisonner, inférer et concevoir des expériences.

Notre objectif est de développer un outil informatique offrant une large palette de fonctionnalités, allant de l'inférence de modèles à partir de données comportementales jusqu'à la simulation, en passant par la vérification de propriétés et la proposition d'expériences significatives.

Nous adoptons pour représenter la dynamique qualitative des systèmes biologiques un type de modèle bien établi : les "réseaux logiques multivalués asynchrones" proposé par R. Thomas et *al.* (Snoussi *et al.*, 1993, Thomas *et al.*, 2001). Ces réseaux, adaptés au manque de données quantitatives précises fournissent une abstraction discrète de systèmes d'équations différentielles affines par morceaux et permettent ainsi une étude qualitative rigoureuse. Cependant ces modèles de Thomas doivent être étendus pour admettre une classe de comportements particuliers mais essentiels pour que les modèles représentent des abstractions d'équations différentielles affines par morceaux. Il s'agit de prendre en compte les états frontières entre les domaines (états dits "singuliers") et les trajectoires confinées à l'intérieur de ces frontières ("modes glissants"). Nous utilisons pour ce faire les travaux de H. de Jong et *al.* (2004).

Pour obtenir la variété de fonctionnalités désirée (inférence, simulation, vérification, proposition d'expériences) à partir d'une spécification formelle unique, nous adoptons comme moyen de mise en œuvre une approche déclarative typique de l'intelligence artificielle, la programmation logique avec contraintes (PLC). Il s'agit donc de construire un système en PLC mettant en œuvre les modèles dit LMAES (pour modèles Logiques Multivalués Asynchrones avec Etats Singuliers). Cette mise en œuvre doit permettre d'obtenir les valeurs recherchées en "intention", plus précisément comme des solutions d'un système d'équations. A cette fin, ce système est structuré selon la méthodologie constituée de la pose des contraintes suivie d'énumérations.

Pour apprécier l'adéquation de l'approche il est nécessaire de l'appliquer à différents modèles biologiques et sur différentes questions. Nous considérons donc dans cette perspective un modèle décrivant un système intervenant dans l'adhésion cellulaire (cf. (Hermant *et al.*, 2003)) à la conception duquel nous avons travaillé avec des biologistes de l'IBS (Institut de Biologie Structurale, Grenoble), et un modèle sur la segmentation de l'embryon de drosophile (cf. (Sánchez *et al.*, 2001)).

Nous présentons tout d'abord (section 2) les modèles LMAES avec une formalisation adaptée à une implémentation en PLC. Ensuite nous abordons les problèmes de mise en œuvre en PLC (section 3). Il s'agit essentiellement d'exprimer par programme une spécification logique. En particulier, nous montrons comment écrire un programme déclaratif minimisant le non-déterminisme à l'exécution et nous exposons

les principales difficultés rencontrées sur le plan de l'expression et sur celui de l'efficacité. Nous appliquons ensuite les différentes fonctionnalités du logiciel sur les deux modèles biologiques que nous avons considérés (section 4).

2. Modèles de Thomas étendus aux "modes glissants"

Nous présentons ici brièvement les fondements du formalisme de Thomas-Snoussi, puis l'extension aux "modes glissants" proposée par de Jong et *al.* (2004).

2.1. *Présentation informelle dans le cadre des réseaux génétiques*

Thomas et Snoussi (Snoussi, 1989, Snoussi *et al.*, 1993) ont développé un formalisme logique pour décrire des systèmes d'interaction qui a été appliqué dans de multiples contextes : réseaux génétiques, neurones, réponse immunologique (Kaufman *et al.*, 1987). Nous en rappelons brièvement les principes en utilisant la terminologie des réseaux génétiques transcriptionnels pour faciliter l'exposé.

Chaque gène i ($i = 1, ..., n$) produit une protéine donnée. Les variables du système sont les concentrations x_i de ces protéines. L'énoncé "i active j" (resp. inhibe) signifie que lorsque la concentration x_i de la protéine i (produite par le gène i) est au dessus (resp. en dessous) d'un certain seuil, alors la protéine i se lie au promoteur du gène j et active (resp. réprime) la transcription de ce dernier. Les gènes peuvent exister dans de multiples états d'expression auxquels correspondent des taux d'expression distincts. Les concentrations des protéines x_i sont discrétisées selon un ou plusieurs seuils : si x_i est la concentration discrétisée de x_i, $\mathsf{x}_i = 0$ signifie que la concentration est en dessous du premier seuil, $\mathsf{x}_i = 1$ signifie que la concentration est entre le premier et le deuxième seuil, etc... L'état d'un système est représenté par un vecteur d'entiers $\mathsf{x} = (\mathsf{x}_1, ..., \mathsf{x}_n)$. De manière générale, les quantités discrètes seront indiquées par des caractères "droits" (x au lieu de x par exemple).

L'évolution du réseau est décrite par des équations dites logiques dans la terminologie de Thomas. Nous préférons les appeler *équations discrètes* étant donné que nous considérons le cas général des systèmes multivalués plutôt que booléens. La forme de ces équations est la suivante : $\forall j = 1, ..., n,\ \Phi_j(\mathsf{x}) = \mathsf{F}_j(\mathsf{x})$ où F_j est une fonction à valeurs entières. La fonction $\mathsf{F} = (\mathsf{F}_1, ..., \mathsf{F}_n)$ associe donc à chaque état x un état "image" $(\Phi_1(\mathsf{x}), ..., \Phi_n(\mathsf{x}))$ qui est appelé *état focal*. Dans la description asynchrone une seule composante x_i peut changer de valeur (de $+1$ ou de -1) lors d'une transition. L'état focal peut être vu comme un état attracteur qui définit la tendance d'évolution du système. Par exemple, si l'état focal de l'état $(0, 0)$ est $(1, 1)$, cela signifie que les deux composantes sont destinées à augmenter, et qu'à un moment donné une des deux passera un seuil. Le système peut donc passer de $(0, 0)$ à l'un des deux états discrets $(1, 0)$ ou $(0, 1)$ selon que la première variable passe un seuil avant la deuxième ou l'inverse. La probabilité pour que les deux variables passent simultanément leur seuil respectif est considérée comme extrêmement faible. C'est la raison pour laquelle on

néglige les transitions en diagonale. Si les paramètres cinétiques et les positions des seuils ne sont pas connus précisément il n'est pas possible de dire quelle transition se produit en premier (dans le cas d'un choix entre plusieurs transitions). Tous les choix de transitions sont alors considérés. La description est non-déterministe à cause de ce manque de connaissance. Lorsque l'état focal d'un état est l'état lui même, l'état est dit *stationnaire*.

Thomas et Snoussi ont établi une connexion entre le formalisme logique et le formalisme différentiel (Snoussi, 1989, Snoussi *et al.*, 1993). Ils considèrent des systèmes génétiques qui peuvent être décrits par des équations différentielles ordinaires (ODEs pour Ordinary Differential Equations) de la forme suivante : $\forall j \in \{1, ..., n\} : \quad \dot{x}_j = F_j(x) - \gamma_j \, x_j$ où $\gamma_j \in \mathbb{R}_{>0}$ et les fonctions $F_j : \mathbb{R}^n_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ sont appelées *fonctions de régulation*. Le système peut être écrit sous forme matricielle $\dot{x} = F(x) - G.x$, où la matrice G est diagonale (les équations sont indépendantes). La fonction de régulation F_j représente le taux de production de la protéine j comme une fonction de la concentration de l'ensemble des protéines qui ont une influence sur le gène j. L'ensemble $\{F_j\}$ de ces fonctions décrit la logique de régulation du réseau. La forme générale considérée ici est identique à celle donnée par de Jong *et al.* (2004) : $F_j(x) = \kappa_{j0} + \sum_{l=1}^{L_j} \kappa_{jl} \, b_{jl}(x)$. Les paramètres κ_{jl} sont des taux de production associés au gène j dans différentes conditions, ces dernières étant indexées par l. L_j est le nombre, éventuellement nul, de termes dans la somme. Les b_{jl} sont des fonctions à valeurs réelles définies sur un pavé fermé de $\mathbb{R}^n$ à valeurs dans $[0, 1]$. Elles sont définies par des expressions arithmétiques à base de sigmoïdes. Chaque fonction b_{jl} définit des conditions dans lesquelles le taux d'expression du gène j contient le terme κ_{jl}. Les conditions définies par deux fonctions b_{jl} associées au même gène j n'étant pas nécessairement exclusives, le taux global d'expression du gène j dans un contexte cellulaire donné peut-être égal à une combinaison linéaire de plusieurs taux élémentaires κ_{jl}.

Le formalisme de Thomas-Snoussi s'applique tout aussi bien à des systèmes du type : $\dot{x} = F(x) - G(x).x$ où les éléments $G_j(x)$ de la matrice diagonale G sont étendus à des fonctions du même type que les F_j. L'exemple du modèle de l'adhésion cellulaire présenté en 4.1 fait partie de ce type de systèmes plus généraux.

2.2. *Des ODEs considérées aux équations différentielles affines par morceaux*

Nous présentons dans cette section de manière plus formelle les développements qui conduisent aux réseaux LMAES. Rappelons que le formalisme présenté peut être utilisé pour décrire différents types de réseaux d'interaction.

Les sigmoïdes $\sigma^{\varepsilon}(x; \theta)$, où $x \in \mathbb{R}$ et $\theta \in \mathbb{R}$, sont approximées par des *fonctions seuil* $s^{\varepsilon}(x; \theta)$ à valeur booléenne telles que :

$$\begin{array}{lllll} \varepsilon = + & : & x < \theta & : & s^+(x;\theta) = 0 \\ & & x \geq \theta & : & s^+(x;\theta) = 1 \\ \varepsilon = - & : & & & s^-(x;\theta) = 1 - s^+(x;\theta) \end{array}$$

Dans cette approximation les fonctions F_j deviennent des fonctions constantes par morceaux.

Les seuils θ_i intervenant dans l'ensemble des fonctions s^ε définissent une partition rectangulaire de l'espace des concentrations. L'ensemble de ces hyper-rectangles ou *domaines* est noté $\mathcal{D} = \{D_\alpha\}$. Dans chacun de ces domaines D_α le système d'ODE se réduit en un système affine (i.e. une partie linéaire et une partie constante). Un système affine non-dégénéré possède un unique point stationnaire (où toutes les dérivés s'annulent) qui est appelé le point focal ϕ_α du domaine D_α. Ceci définit une correspondance : $D_\alpha \in \mathcal{D} \rightarrow \phi_\alpha \in \mathbb{R}^n$.

Pour les équations définissant F_j données en 2.1, les points où la dérivée s'annule sont donnés par : $\phi_j(x) = K_{j0} + \sum_{l=1}^{L_j} K_{jl}\, b_{jl}(x)$ où $K_{jl} = \frac{\kappa_{jl}}{\gamma_j}$.

Dans la mesure où la matrice G est diagonale et où ses éléments diagonaux sont strictement négatifs ($-\gamma_j$), les trajectoires sont monotones ($\dot{x}_i$ garde un signe constant le long d'une trajectoire à l'intérieur d'un domaine donné).

2.3. *Abstraction discrète*

Suivant Thomas et *al.* (2001) nous définissons un opérateur de discrétisation d_i pour chaque axe x_i. Une variable x_i qui est associée à N_i seuils θ_i^ν ($\nu \in \{1, ..., N_i\}$) est abstraite en une variable discrète $\mathsf{x}_i = d(x_i)$ comme suit : $\mathsf{x}_i = \nu \;\Leftrightarrow\; \theta_i^\nu < x_i < \theta_i^{\nu+1}$. Les x_i ne sont pas définis pour les valeurs seuils ($x_i = \theta_i^\nu$), mais le seront dans la partie 2.5.

Les états discrets sont des vecteurs d'entiers notés x. Comme il y a une bijection ($D \leftrightarrow \mathsf{x}$) entre les domaines de $\mathbb{R}^n$ et les états discrets, $\phi(D)$ et $\phi(\mathsf{x})$ peuvent être identifiés.

On définit par extension la fonction $\mathsf{x} \rightarrow d(\phi(\mathsf{x})) = (d_1(\phi_1(\mathsf{x})), ..., d_n(\phi_n(\mathsf{x})))$ qui associe un état focal discret à un état discret x. Celle-ci est appelé fonction discrète par Snoussi (1989).

La composante discrète Φ_j est obtenue en appliquant l'opérateur de discrétisation d_j à l'expression donnée en 2.2 : $\Phi_j(\mathsf{x}) = d_j(K_{j0} + \sum_{l=1}^{L_j} K_{jl}\, b_{jl}(\mathsf{x}))$.

Pour simplifier les notations, les fonctions booléennes à arguments entiers sont notées aussi b_{jl}. En d'autres termes, nous ne distinguons pas $b_{jl}(\mathsf{x}_i) = b_{jl}(d(x_i))$ et $b_{jl}(x_i)$.

2.4. *Equations discrètes des états focaux*

Pour introduire ces équations nous considérons un exemple dans lequel la concentration x_p dépend de deux variables x_i et x_j.

Exemple : $\dot{x}_p = \kappa_{p0} + \kappa_{pi}\,\sigma^+(x_i;\theta_i) + \kappa_{pj}\,\sigma^+(x_j;\theta_j) - \gamma_p x_p$

Dans ce cas, $\Phi_p = d_p(K_{p0} + K_{pi}s_i^+ + K_{pj}s_j^+)$ où d_p est l'opérateur de discrétisation de l'axe p (comme défini précédement), et s_i^+, s_j^+ sont les fonctions seuils approximant respectivement $\sigma^+(x_i;\theta_i)$ et $\sigma^+(x_j;\theta_j)$. Etant donné que deux fonctions booléennes interviennent dans Φ_p, quatre cas exclusifs doivent être considérés conduisant à une expression contenant quatre termes :

$$\begin{aligned}\Phi_p &= d_p(K_{p0})\,(1-s_i^+)(1-s_j^+) \;+\; d_p(K_{p0}+K_{pi})\,s_i^+(1-s_j^+) + \\ &\quad\; d_p(K_{p0}+K_{pj})\,(1-s_i^+)s_j^+ \;+\; d_p(K_{p0}+K_{pi}+K_{pj})\,s_i^+s_j^+ \\ &= \mathsf{K}_{p.\{\}}(1-s_i^+)(1-s_j^+) + \mathsf{K}_{p.\{i\}}s_i^+(1-s_j^+) + \mathsf{K}_{p.\{j\}}(1-s_i^+)s_j^+ + \mathsf{K}_{p.\{i,j\}}s_i^+s_j^+\end{aligned}$$

où les paramètres discrets sont définis par $\mathsf{K}_{p.\{\}} = d_p(K_{p0})$, $\mathsf{K}_{p.\{i\}} = d_p(K_{p0}+K_{pi})$, $\mathsf{K}_{p.\{j\}} = d_p(K_{p0}+K_{pj})$, $\mathsf{K}_{p.\{i,j\}} = d_p(K_{p0}+K_{pi}+K_{pj})$,
avec $\mathsf{K}_{p.\{\}} \le \mathsf{K}_{p.\{i\}}$, $\mathsf{K}_{p.\{\}} \le \mathsf{K}_{p.\{j\}}$, $\mathsf{K}_{p.\{i\}} \le \mathsf{K}_{p.\{i,j\}}$ et $\mathsf{K}_{p.\{j\}} \le \mathsf{K}_{p.\{i,j\}}$. ∎

Dans le cas général, si une variable possède plusieurs seuils θ_i^ν (parce qu'elle influence plusieurs gènes) l'ordre entre ces seuils doit être pris en compte. Il en résulte que dans l'expression de Φ_p toutes les combinaisons de valeurs booléennes b_{jl} ne sont pas nécessairement possibles : par exemple, si x_i est au dessus du seuil maximum, il ne peut être en dessous du seuil minimum.

Il faut noter qu'il est nécessaire d'introduire de nouveaux paramètres pour représenter la discrétisation de sommes de paramètres réels, car $d_p(K_{pi}+K_{pj}) \neq d_p(K_{pi}) + d_p(K_{pj})$. Ainsi, le nombre de paramètres discrets est plus élevé que celui des paramètres réels. Mais le nombre de jeux de valeurs distincts est limité par le fait que les paramètres discrets sont contraints par des inégalités du type $d_p(K_{pi}+K_{pj}) \ge d_p(K_{pi})$ et $d_p(K_{pi}+K_{pj}) \ge d_p(K_{pj})$.

En résumé, la discrétisation des équations aux points focaux conduit à trois entités :

1) un ensemble de paramètres discrets $\mathsf{K}_{j.J}$, où $J \subset \{1,...,L_j\}$

2) un ensemble d'inégalités entre ces paramètres ($J_1 \subset J_2 \Rightarrow \mathsf{K}_{j.J_1} \le \mathsf{K}_{j.J_2}$),

3) un système d'équations discrètes paramétrées.

2.5. *Etats singuliers et règles de transition*

Pour obtenir une véritable abstraction discrète de la dynamique continue il est nécessaire de prendre en compte les états qui sont situés aux frontières de discré-

tisation (de Jong *et al.*, 2004). Ainsi, une variable x_i qui est associée à N_i seuils θ_i^ν ($\nu \in \{1, ..., N_i\}$) est abstraite en une variable discrète comme suit :

$$\mathsf{x}_i = \nu \;\Leftrightarrow\; \theta_i^\nu < x_i < \theta_i^{\nu+1} \quad \text{et} \quad \mathsf{x}_i = \nu - \frac{1}{2} \;\Leftrightarrow\; x_i = \theta_i^\nu$$

Si la composante x_i est entière elle est dite *régulière* (elle représente un intervalle compris strictement entre deux seuils), sinon elle est dite *singulière* (elle représente une valeur seuil). Un *état singulier* est un état ayant au moins une composante singulière. Un *état régulier* est un état sans composante singulière. L'*ordre* d'un état est égal au nombre de composantes singulières. La dimension d'un état x, noté $dim(\mathsf{x})$, est le nombre de composantes régulières de cet état.

Nous exposons dans ce qui suit les règles de transitions entre états (réguliers ou singuliers). Ces règles sont déduites de celles présentées par de Jong *et al.* (2004). Elles sont reformulées sous une forme équivalente propre à l'expression sous forme de contraintes.

Quelques préliminaires sont nécessaires :

– $\mathfrak{R}$ représente l'ensemble des états réguliers du système discret.

– $Reg(\mathsf{x})$ est l'ensemble des états réguliers adjacents à x. Si x est lui-même régulier alors $Reg(\mathsf{x}) = \{\mathsf{x}\}$.

– $Sing(\mathsf{x})$ est l'ensemble des états singuliers x' adjacents à x tels que $dim(\mathsf{x}') < dim(\mathsf{x})$.

– $Foc(\mathsf{x}) = \{\mathsf{x}^{\mathrm{f}} \in \mathfrak{R} \mid \exists \mathsf{x}' \in Reg(\mathsf{x}),\ \mathsf{x}^{\mathrm{f}} = \Phi(\mathsf{x}')\}$ représente l'ensemble des états focaux des états réguliers adjacents à un état x quelconque.

– L'*attracteur* de x $Att(\mathsf{x})$ est l'hyper-rectangle minimal contenant tous les états focaux des états réguliers adjacents à x. Il est représenté par le couple $(\mathsf{x}^{min}, \mathsf{x}^{max})$ où x^{min} et x^{max} sont les états extrêmes de l'hyper-rectangle. Autrement dit les composantes x_i^{min} (resp. x_i^{max}) sont les minima (resp. maxima) des ensembles $\{\mathsf{x}_i^{\mathrm{f}} \mid \mathsf{x}^{\mathrm{f}} \in Foc(\mathsf{x})\}$.

Les trois propositions qui suivent permettent d'établir l'existence d'une transition dans trois cas distincts. Ces propositions sont démontrées en (Corblin, 2005). La proposition 2.1 permet d'établir l'existence d'une transition d'un état x vers un état x' de plus basse dimension. L'idée intuitive, fondée sur la description continue, est que x doit tout d'abord être *persistant*, c'est à dire qu'il doit exister des trajectoires qui glissent sur l'état x ("mode glissant"). La deuxième condition permet de déterminer le sens de la transition en imposant qu'il existe dans $Foc(\mathsf{x})$ un état focal qui soit positionné de façon à attirer dans la direction de x' les trajectoires partant de x. Autrement dit, pour le cas où x_i est une composante régulière et $\mathsf{x}'_i = \mathsf{x} + 1$, la transition est possible si $\mathsf{x}_i < \mathsf{x}_i^{\max}$.

Proposition 2.1 *Soient deux états* x *et* x'*, avec* $\mathsf{x}' \in Sing(\mathsf{x})$*. Soit* $Att(\mathsf{x}) = (\mathsf{x}^{\min}, \mathsf{x}^{\max})$ *l'attracteur de* x*. Il existe une transition de* x *vers* x' *si et seulement si* $\forall\, i, 1 \leq i \leq n$*,*

1) $\mathsf{x}_i^{\min} < \mathsf{x}_i < \mathsf{x}_i^{\max}$*, si* x_i *est une variable singulière*

2) $((\mathsf{x}'_i = \mathsf{x}_i + 1) \wedge (\mathsf{x}_i < \mathsf{x}_i^{\max})) \vee ((\mathsf{x}'_i = \mathsf{x}_i - 1) \wedge (\mathsf{x}_i > \mathsf{x}_i^{\min}))$*, si* x_i *est régulière et* x'_i *singulière*

La définition de $Sing(\mathsf{x})$ permet d'obtenir des successeurs par des transitions en diagonale. Ces successeurs, comme on l'a vu en 2.1, apparaissent très improbables. Aussi nous adoptons une définition plus restrictive qui élimine ce type de successeurs pour la proposition 2.1 : $Sing(\mathsf{x})$ est l'ensemble des états singuliers x' adjacents à x tel que $dim(\mathsf{x}') = dim(\mathsf{x}) - 1$.

La proposition 2.2, qui ne figure pas explicitement dans le travail de de Jong et *al.* (2004), permet d'établir si x peut être son propre successeur. Ceci est important lorsque l'on veut identifier les états stationnaires. L'idée intuitive est similaire à ce que nous venons de voir. Il faut que x soit persistant et que x soit contenu dans son attracteur.

Proposition 2.2 *Soit un état* x*, d'attracteur* $Att(\mathsf{x}) = (\mathsf{x}^{\min}, \mathsf{x}^{\max})$*. Il existe une transition de* x *vers* x *si et seulement si* $\forall\, i, 1 \leq i \leq n,\ \mathsf{x}_i^{\min} \leq \mathsf{x}_i \leq \mathsf{x}_i^{\max}$

La proposition 2.3 permet d'établir l'existence d'une transition vers un état de plus haute dimension. L'idée intuitive est que les trajectoires partant de x peuvent aller vers x' si x' est persistant (première condition) et que l'attracteur de x' ne s'oppose pas à la transition (deuxième condition).

Proposition 2.3 *Soient deux états* x *et* x'*, avec* $\mathsf{x} \in Sing(\mathsf{x}')$*. Soit* $Att(\mathsf{x}') = (\mathsf{x}'^{\min}, \mathsf{x}'^{\max})$ *l'attracteur de* x'*. Il existe une transition de* x *vers* x' *si et seulement si* $\forall\, i, 1 \leq i \leq n$*,*

1) $\mathsf{x}_i'^{\min} < \mathsf{x}'_i < \mathsf{x}_i'^{\max}$*, si* x'_i *est singulière*

2) $((\mathsf{x}'_i = \mathsf{x}_i + 1) \wedge (\mathsf{x}_i < \mathsf{x}_i'^{\max})) \vee ((\mathsf{x}'_i = \mathsf{x}_i - 1) \wedge (\mathsf{x}_i > \mathsf{x}_i'^{\min}))$*, si* x_i *est singulière et* x'_i *est régulière*

Noter que c'est ici l'état destination x' qui doit être persistant.

3. Conception et mise en œuvre en PLC des modèles LMAES

Nous présentons d'abord certaines notions méthodologiques de la PLC essentielles à la bonne compréhension des travaux et nous rappelons l'objectif général de la mise en œuvre en PLC des modèles LMAES. Pour atteindre cet objectif nous montrons ensuite qu'il faut minimiser le retour arrière (backtracking), et nous expliquons donc comment le réduire de manière systématique. Nous soulignons aussi les problèmes rencontrés pour obtenir une modélisation finie sous forme de contraintes. L'organisation générale du programme est exposée essentiellement pour montrer l'adéquation

de la mise en œuvre à la spécification (voir 2.5). Enfin, nous donnons la complexité en fonction du nombre de contraintes de notre programme en fonction du modèle LMAES à analyser.

3.1. *Objectif de la mise en œuvre et notions essentielles*

Rappelons que nous cherchons à obtenir des valeurs en intention, comme des solutions d'un système d'équations : ce peut être par exemple les jeux de valeurs de paramètres définissant une classe de modèles satisfaisant certaines propriétés. Du point de vue méthodologique, la construction d'un programme en PLC répondant à cet objectif procède en deux étapes :

1) Spécification logique : dans notre cas, il s'agit d'une part de définir le type de modèle considéré (les réseaux LMAES) et d'autre part le modèle biologique à analyser. Ces définitions sont établies à l'aide de prédicats permettant de poser les contraintes auxquelles doivent satisfaire les comportements admis par le modèle biologique. Toutes ces contraintes portent sur des variables entières à domaines finis dans la mesure où les réseaux LMAES n'utilisent que ce type de valeurs.

2) Requêtes : certains paramètres (variables du programme) peuvent être très contraints et d'autres beaucoup moins. Dans notre cas, les paramètres sont typiquement les paramètres du modèle biologique et les comportements, représentés sous la forme de séquences d'états (chemins). Les requêtes peuvent être très diverses. C'est ce qui fait l'intérêt de l'approche. Par exemple, une requête portant sur l'existence d'un comportement demande que les paramètres du modèle soient plutôt fixés. A contrario, une requête portant sur l'existence d'un modèle à partir de comportements demande que ce soit les comportements qui soient plutôt fixés.

Dans la mesure où nous utilisons un résolveur sur intervalles (Colmerauer, 1996), l'ajout de contraintes permet de réduire les intervalles de valeurs possibles de certaines variables. Cependant le résolveur que nous utilisons, sur intervalles entiers avec un nombre fini de valeurs, ne fournit qu'une cohérence faible : les réponses sûres ne sont que les négatives (incohérence du système de contraintes) ou les positives associées à une solution instanciée (obtenue par énumération). Les réponses positives dans le cas général d'obtention de solutions non instanciées signifient seulement que s'il existe une solution les valeurs des variables sont dans les intervalles indiqués. Un des enjeux majeurs consiste à déterminer les meilleures heuristiques d'énumération pour un type de requête donnée. Cela conditionne les performances en temps de calcul du programme.

Un autre moyen important pour améliorer l'efficacité de la résolution de contraintes sur intervalles consiste à introduire des *contraintes redondantes*. Ces contraintes, produites ou non à la main, peuvent contribuer considérablement à une meilleure propagation des contraintes.

Enfin, un souci de taille en PLC est d'assurer la terminaison des requêtes. Dans notre cas, plusieurs difficultés relèvent de cet aspect. Nous les examinons en 3.3.

3.2. *Réduction du retour arrière*

En PLC on utilise classiquement pour exprimer des disjonctions des règles différentes dont l'application demande une mise en œuvre à base de retour arrière (backtracking). Ce type d'expression, très agréable à pratiquer et intéressant dans certaines situations, s'oppose à la méthodologie exposée en 3.1. Pour éviter les retours arrières, nous réunissons systématiquement un paquet de règles en une seule règle.

Nous exposons dans ce qui suit une méthode systématique pour y aboutir. Elle s'appuie sur la possibilité en Prolog IV d'utiliser conjointement des contraintes numériques et des contraintes booléennes.

a) Soit un prédicat `p(X)` qui est vrai si au moins un des k prédicats $\mathtt{p}_i\mathtt{(X)}$, $1 \leq i \leq k$, est vrai. La méthode systématique pour réunir ces k règles disjointes en une règle consiste à :

1) introduire k booléens $\mathtt{B}_i$ et k nouveaux prédicats $\mathtt{pp}_i\mathtt{(B}_i\mathtt{,X)}$ tels que $\mathtt{pp}_i\mathtt{(B}_i\mathtt{,X)}$ est vrai si et seulement si : $\mathtt{B}_i$ est équivalent à "$\mathtt{p}_i\mathtt{(X)}$ est vrai" ;

2) introduire un booléen `B` et un prédicat `pp(B,X)` tel que `pp(B,X)` est vrai si et seulement si : les k conditions $\mathtt{pp}_i\mathtt{(B}_i\mathtt{,X)}$ sont vraies, et si `B` est égal à la *disjonction* des k $\mathtt{B}_i$.

Exemple : Pour $k = 2$ nous opérons la transformation suivante :

$$\begin{array}{c} \mathtt{p(X)} :- \mathtt{p1(X)}. \\ \mathtt{p(X)} :- \mathtt{p2(X)}. \\ \downarrow \\ \mathtt{pp(B,X)} :- \mathtt{B} = \mathtt{B1\ bor\ B2},\ \mathtt{pp1(B1,X)},\ \mathtt{pp2(B2,X)}. \end{array}$$

avec `pp1(B1,X)` qui est vrai ssi `B1` est équivalent à "`p1(X)` est vrai", et `pp2(B2,X)` qui est vrai ssi `B2` est équivalent à "`p2(X)` est vrai". La contrainte `B1 bor B2` produit un booléen qui est la disjonction de `B1` et `B2`. Cette contrainte est traitée par le solveur sur intervalles. `p(X)` est alors vrai à condition que `pp(1,X)` le soit aussi (`p(X) :- pp(1,X)`). ∎

b) Soit un prédicat `p(X)` qui est vrai si les k prédicats $\mathtt{p}_i\mathtt{(X)}$, $1 \leq i \leq k$, sont vrais. La méthode systématique pour introduire un booléen dans le cas d'une règle dont la condition de validité est une conjonction est la suivante :

1) introduire k nouveaux prédicats $\mathtt{pp}_i\mathtt{(B}_i\mathtt{,X)}$, où $\mathtt{B}_i$ est un booléen tel que $\mathtt{pp}_i\mathtt{(B}_i\mathtt{,X)}$ est vrai ssi $\mathtt{B}_i$ est équivalent à "$\mathtt{p}_i\mathtt{(X)}$ est vrai" ;

2) spécifier que la condition pour que `pp(B,X)` soit vrai soit : la conjonction des k conditions $\mathtt{pp}_i\mathtt{(B}_i\mathtt{,X)}$, et de la condition que `B` vaille la *conjonction* des k $\mathtt{B}_i$.

Exemple : Pour $k = 2$ nous opérons la transformation suivante :

$$\begin{array}{c} \texttt{p(X)} :- \texttt{p1(X)},\ \texttt{p2(X)}. \\ \downarrow \\ \texttt{pp(B,X)} :- \texttt{B} = \texttt{B1 band B2},\ \texttt{pp1(B1,X)},\ \texttt{pp2(B2,X)}. \end{array}$$

`B1 band B2` produit un booléen qui est la conjonction de `B1` et `B2`. ∎

Cette approche permet d'exprimer le problème à résoudre sous la forme d'un système de contraintes unique (au lieu d'un ensemble de plusieurs systèmes différents). De plus, elle permet d'obtenir la négation des nouveaux prédicats `pp(B,X)` en fixant `B` à 0, ce qui se revèle intéressant si l'on veut par exemple interdire des comportements.

3.3. *Problèmes posés par la modélisation en contraintes*

Il est intéressant de relever quelques difficultés pour construire une modélisation en contraintes à partir d'une formulation telle que celle présentée par de Jong et *al.* (2004). Elles sont dues à la nécessité de produire un nombre fini de contraintes, à la représentation d'hyper-rectangles minimaux, et à la pose de contraintes précises exprimant la définition d'un état focal.

Contraintes concernant l'existence d'une solution

En suivant fidèlement la formulation mathématique proposé par de Jong et *al.* (2004), on est amené à définir un prédicat qui n'est pas exprimable sous forme de contraintes Prolog IV (ou, à notre connaissance, d'autres logiciels contraintes). Ce prédicat est par exemple de la forme

$\texttt{p}(\texttt{B1}, \texttt{X})$ vrai si $\texttt{B1} \equiv$ "$\texttt{X} = 1$ est une solution du système d"équations global"

En effet la description mathématique utilise un vecteur `X` supposé donné. Dans notre cas où `X` n'est pas a priori donné, il est normal de représenter les valeurs de `X` en intention. Ces contraintes permettent à la fois d'imposer l'existence de solution (`B1` est vrai) et d'inférer la présence de solution. Elles sont intéressantes dans notre cas pour assurer la réversibilité désirée.

Pour contourner cette difficulté, une solution consiste à déterminer une nouvelle condition $C1$ (exprimable sous forme de contraintes) exprimant des propriétés sur l'ensemble des valeurs de `X`. Par exemple "$C1$ vrai" pourrait impliquer "$\texttt{X} = 1$ est toujours une solution".

Cette approche nous a conduit à proposer les nouvelles conditions qui expriment les relations entre un état et son successeur : ce sont les propositions 2.1 à 2.3.

Contraintes définissant un hyper-rectangle minimal

Rappelons que les successeurs d'un état x sont définis à partir de l'hyper-rectangle minimal contenant les états focaux des états réguliers adjacents à x. Un tel hyper-rectangle est défini par deux états dont la $i^{ème}$ composante de chacun est respectivement le minimum et le maximum des $i^{ème}$ composantes de ces états focaux (voir 2.5).

Définir intentionnellement ces deux états se pose dans les termes suivant : étant donné dans un système de contraintes, exprimer les contraintes déterminant la solution minimale (ou maximale) de ce système selon un certain critère. Dans notre cadre de programmation par contraintes on ne peut malheureusement résoudre ce problème dans le cas général. Mais si le nombre N de solutions est fini, on peut utiliser une contrainte qui établit la relation entre trois réels, dont le premier est le minimum des deux autres : il faut alors poser de l'ordre de N contraintes de ce type. Dans notre cas, les inconnues sont les composantes des états focaux et N est le nombre d'états réguliers adjacents à x. N vaut 2^{Ord} où Ord est l'ordre de l'état x (le nombre de composantes singulières de x). Ceci implique que cette méthode ne peut être utilisée que si l'ordre de l'état x est connu. Pour pallier cette difficulté, nous utilisons un processus suspensif en attente de l'ordre pour poser les contraintes de minimisation.

Contraintes définissant un état focal

Une troisième difficulté concerne la pose des contraintes sur une composante d'un état focal en fonction des équations focales. Cela nécessite l'expression du prédicat `comp_focal(Xf, LB, LV_Xf)` qui est vrai ssi `LB` est une liste de booléens dans laquelle un et un seul élément est vrai (celui d'index i), `LV_Xf` est une liste d'entiers, et `Xf` est le $i^{ème}$ élement de `LV_Xf`. La contrainte que `LB` contienne un et un seul élément vrai est supposée déjà posée. Une implémentation semblant bien formaliser le problème est la suivante :

```
comp_focal(Xf, LB, LV_Xf) :-
                index(1, LB, I),
                index(Xf, LV_Xf, I).
```

où `index(V,L,I)` est vrai ssi `V` est la `I`^{ème} valeur de `L`. Une telle implémentation peut permettre dans certain cas d'interdire à `Xf` de prendre la $j^{ème}$ valeur de `LV_Xf` si `LB`$_j$ est faux. Malheureusement ce type de déduction très intéressante requiert en général l'utilisation du mode union d'intervalles, qui est très couteux en mémoire. Voici un exemple où le mode union d'intervalles est nécessaire :

Exemple :

```
LB = [B1, B2, B3],
LV_Xf = [4,6,5],
comp_focal(Xf, LB, LV_Xf),
B2 = 0.
```

Le résultat désiré est que `Xf` appartienne à l'intervalle fermé $[4, 5]$ puisque 6 a été éliminé de la liste. C'est ce qui est effectivement obtenu avec l'implémentation ci-dessus en mode union d'intervalles : `I` est contraint à appartenir à l'union des intervalles $[1, 1]$ et $[3, 3]$. En mode intervalle simple on obtient que `Xf` appartient à l'intervalle fermé $[4, 6]$, `I` étant juste contraint à appartenir à l'intervalle $[1, 3]$ ∎

Nous donnons en (Corblin, 2005) une solution équivalente qui utilise uniquement des intervalles simples et qui permet de telles déductions à l'aide de la pose de contraintes redondantes.

3.4. *Organisation du programme de modélisation des modèles LMAES*

Il s'agit de montrer la simplicité de la démarche de construction de ce programme. Son architecture est donnée en figure 1. Elle donne un aperçu sur la correspondance étroite entre l'implémentation et la formalisation donnée en 2.5.

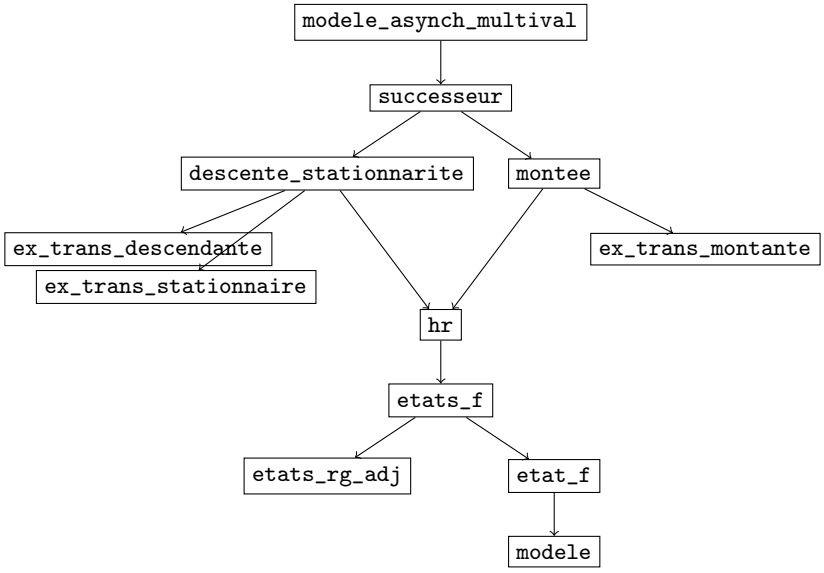

Figure 1. *Architecture informelle du programme. Les nœuds du graphes sont les noms des prédicats principaux que nous présentons. Une connexion orientée d'un prédicat $p1$ à un prédicat $p2$ signifie que les contraintes posées par $p2$ sont aussi posées par $p1$.*

Les prédicats principaux du programme sont les suivants :

– `modele` exprimant les contraintes définissant un modèle biologique.

– `modele_asynch_multival` exprimant les contraintes d'existence d'un chemin (comportement qualitatif) selon un modèle.

– `successeur` exprimant les contraintes d'existence d'une transition selon un modèle.

– `hr` définissant l'hyper-rectangle minimal contenant un ensemble d'états.

– `etats_f` définissant les contraintes sur l'ensemble des états focaux des états réguliers adjacents à un état.

– `etat_f` définissant l'état focal d'un état régulier selon le modèle considéré.

3.4.1. *modele*

Le prédicat `modele([Idf,P], Modele)` est vrai si `Modele` est la représentation complète du modèle d'identifiant `Idf` et de paramètres `P`.

Plus précisément `Modele = [Especes, Equations, Domaines, P]` avec :

– `Especes` : la liste des noms des espèces chimiques du modèle biologique

– `Equations` : la liste des équations discrètes définissant l'évolution qualitative du système

– `Domaines` : la liste des nombres de seuils pour chaque espèce du système

– `P` : la liste des paramètres se divisant en 3 sous-listes, les concentrations discrétisées des espèces en entrées (dont les concentrations sont des paramètres qui restent constants), les paramètres d'ordre de seuils (dans le cas de variables ayant plusieurs seuils) et les paramètres cinétiques discrétisés.

Les paramètres de ce prédicat sont produits automatiquement à partir des équations continues donnant les états focaux, et des contraintes éventuelles sur les ordres des seuils.

3.4.2. *modele_asynch_multival*

Le prédicat `modele_asynch_multival(B, Modele, Chemin)` définit les réseaux de Thomas avec prise en compte des états singuliers (modèles LMAES). Il est vrai si `Modele` est le modèle considéré, et `B` est équivalent à "`Chemin` est un chemin possible selon le modèle `Modele`".

C'est en utilisant ce prédicat que l'on pose des questions concernant une évolution qualitative dans son ensemble. Une évolution qualitative ou comportement qualitatif est une suite d'états dont chaque état est un successeur possible de l'état le précédant.

3.4.3. *successeur*

Le prédicat `successeur(B, Modele, Etat_i, Etat_s)` est vrai si `B` est équivalent à "`Etat_s` est un successeur de `Etat_i` selon le modèle considéré `Modele`". Ce prédicat s'écrit en distinguant trois cas (descente en dimension, stationnarité ou montée en dimension – voir respectivement propositions 2.1, 2.2 et 2.3). En utilisant la

méthode systématique pour réunir des règles disjointes décrite en 3.2, nous obtenons une unique règle correspondant à ces 3 cas :

```
successeur(B, Modele, Etat_i, Etat_s) :-
           B = B1 bor B2,
           not(B1 band B2),
           descente_stationnarite(B1, Modele, Etat_i, Etat_s),
           montee(B2, Modele, Etat_i, Etat_s).
```

où `descente_stationnarite(B, Modele, Etat_i, Etat_s)` (resp. `montee(B, Modele, Etat_i, Etat_s)`) est vrai si `B` est équivalent à "`Etat_s` est un successeur de `Etat_i` et de dimension inférieure ou égale (resp. supérieure) à celle de `Etat_i`. La deuxième condition (`not(B1 band B2)`) est une contrainte redondante qui spécifie que les solutions données par `descente_stationnarite` et `montee` sont exclusives.

`montee(B, Modele, Etat_i, Etat_s)` pose ses contraintes i) en introduisant une variable `HR` représentant l'hyper-rectangle minimal contenant $Foc(\texttt{Etat\_s})$ à l'aide du prédicat de nom `hr`, et ii) en implémentant les deux conditions sur `Etat_i`, `Etat_s` et `HR` de la proposition 2.3 à l'aide du prédicat de nom `ex_trans_montante` (pour "existence de la transition montante").

De manière similaire, les prédicats `ex_trans_descendante` et `ex_trans_stationnaire` traduisent respectivement les conditions des propositions 2.1 et 2.2.

3.4.4. *hr*

Le prédicat `hr(Modele, Etat_i, HR, Ordre)` est vrai si `HR` est l'hyper-rectangle minimal contenant les états focaux des états réguliers adjacents à `Etat_i` (attracteur de `Etat_i`) selon le modèle `Modele`.

La variable `HR` est une liste de deux états. Ces deux états sont les états extrêmes de l'attracteur (cf. 2.5).

Le principe de conception de ce prédicat est le suivant :

– contraindre une variable `Etats_f` à être la liste des états focaux des états réguliers adjacent à `Etat_i` à l'aide du prédicat `etats_f`.

– contraindre les composantes du premier (resp. du deuxième) état de `HR` à être les minima (resp. maxima) des composantes correspondantes des états de `Etats_f`.

3.4.5. *etats_f*

Le prédicat `etats_f(Modele, Etat_i, Etats_f, Ordre)` est vrai si `Ordre` est l'ordre de `Etat_i` et si `Etats_f` est la liste des états focaux des états réguliers adjacents à l'état `Etat_i` selon le modèle `Modele`. Une précondition indispensable pour la terminaison d'une requête portant sur ce prédicat est la connaissance de la valeur de `Ordre`.

En effet la connaissance de l'ordre Ord de `Etat_i` est nécessaire pour obtenir celle de la taille de la liste `Etats_f`, car le nombre d'états réguliers adjacents à un état d'ordre Ord est 2^{Ord}. Ceci permet d'assurer la terminaison de la pose des contraintes liées au prédicat `etats_f`. Il faut noter la nécessité de représenter en extension la variable `Etats_f`. En effet pour poser les contraintes définissant l'hyper-rectangle minimal contenant les états focaux des états réguliers adjacents à un état du fait que la seule contrainte de base disponible est `max(M,X1,X2)` (resp. `min(M,X1,X2)`) vrai si `M` est la maximum (resp. minimum) de `X1` et `X2`.

Le principe de conception de ce prédicat est trivial. Pour un état `Etat_i` :

– Contraindre une variable `Etats_rg_adj` à être la liste en extension des états réguliers adjacents à `Etat_i`. On utilise pour cela le prédicat de nom `etats_rg_adj`.

– Contraindre une variable `Etats_f` à être la liste des états focaux des états réguliers de la liste `Etats_rg_adj`. On utilise pour cela le prédicat de nom `etat_f` spécifié ci-après.

3.4.6. *etat_f*

Le prédicat `etat_f(B, Modele, Etat_i, Etat_f)` est vrai si `B` est équivalent à "`Etat_f` est l'état focal de `Etat_i` selon le modèle considéré `Modele`". Ce prédicat utilise pour cela le système d'équations discrètes défini dans la variable `Modele`.

3.5. *Complexité en terme de nombre de contraintes*

Le coût en espace dépend du nombre de contraintes attachées à un chemin. Ce nombre est clairement une fonction linéaire du nombre d'états du chemin (voir prédicat de nom `modele_asynch_multival` en 3.4.2).

Par ailleurs le nombre d'équations exprimant la liaison entre un état et son successeur (voir prédicat de nom `successeur` en 3.4.3) est une fonction linéaire du nombre de composantes d'un état, c'est-à-dire de la dimension du système. Les coûts sont donc très supportables.

Malheureusement, le nombre d'équations nécessaires pour déterminer la tendance d'un état singulier est exponentiel en fonction de l'ordre de cet état. Mais il faut remarquer que l'on peut espérer raisonnablement que l'ordre maximal atteignable à partir d'un état régulier reste faible. Ceci est essentiellement dû au fait que l'on néglige les transitions en diagonale (voir 2.5) vers les états de plus basse dimension que l'état initial (correspondant au cas où au moins deux variables franchissent leur seuil simultanément, ce qui est improbable). De plus, dans ce cas, un chemin débutant par un état régulier et atteignant un état singulier de dimension p doit franchir des états de toutes les dimensions intermédiaires entre n et p : pour p petit les modèles admettant de tels chemins apparaissent très spécifiques et donc rares.

D'autre part si une table (d'états focaux) donnant pour chaque état son état focal est utilisée, alors le nombre d'équations nécessaires pour définir un état focal est

exponentiel en fonction de la dimension du système. Cependant si nous utilisons un système d'équations discrètes pour définir le modèle, cela permet de définir l'état focal d'un état x à partir d'un nombre réduit de composants de x. Ce nombre de composants est au maximum le demi-degré intérieur maximal du graphe d'interaction entre les composants biologiques qui est habituellement faible. Il en ressort que dans ce cadre le nombre d'équations définissant un état focal varie, au pire, exponentiellement avec la connectivité du réseau et non avec sa dimension.

Le coût en temps dépend certes du nombre de contraintes mais principalement du nombre d'énumérations. Des heuristiques pertinentes doivent être recherchées pour réduire ce coût. En particulier, ce coût serait réductible en réduisant le nombre de modèles à considérer, grâce à la construction de classes d'équivalence regroupant des modèles admettant les mêmes graphes de transitions.

4. Examen de phénomènes biologiques

Nous présentons la modélisation de deux phénomènes biologiques : le premier porte sur l'adhésion cellulaire et le second sur la segmentation de l'embryon de Drosophile.

4.1. *Adhésion des cellules endothéliales humaines*

Le phénomène d'adhésion des cellules et son contrôle par les cellules sont complexes et notre connaissance sur ce sujet est loin d'être complète. Les parois des vaisseaux sanguins sont composés d'une monocouche de cellules qui forment une barrière entre le sang et les tissus adjacents. Les jonctions entre les cellules endothéliales (jonctions adhérentes) sont composées de protéines VE-cadhérine (VE pour *Vascular Endothelium*) qui sont intégrées à la membrane plasmique, et de protéines β-caténine qui se lient aux extrémités cytoplasmiques des VE-cadhérines. La migration des leucocytes à travers la monocouche conduit à une cassure des jonctions, par clivage des VE-cadhérines et relargage des β-caténines dans le cytoplasme (Hermant *et al.*, 2003). Notre étude porte sur le processus de réparation de ces jonctions après déstabilisation par des anticorps (perturbation). La structure biochimique du système est illustrée informellement par la figure 2.

A partir du graphe des réactions chimiques il est possible de dériver un système d'équations différentielles donnant les taux de variation de la concentration de chaque espèce chimique (Fanchon *et al.*, 2005). Après l'application de plusieurs hypothèses de simplification décrites dans la référence précédente, il est possible d'isoler un sous-système à deux variables x et z donnant l'évolution temporelle de celles-ci (les quatre autres variables sont exprimables à partir de x et z) :

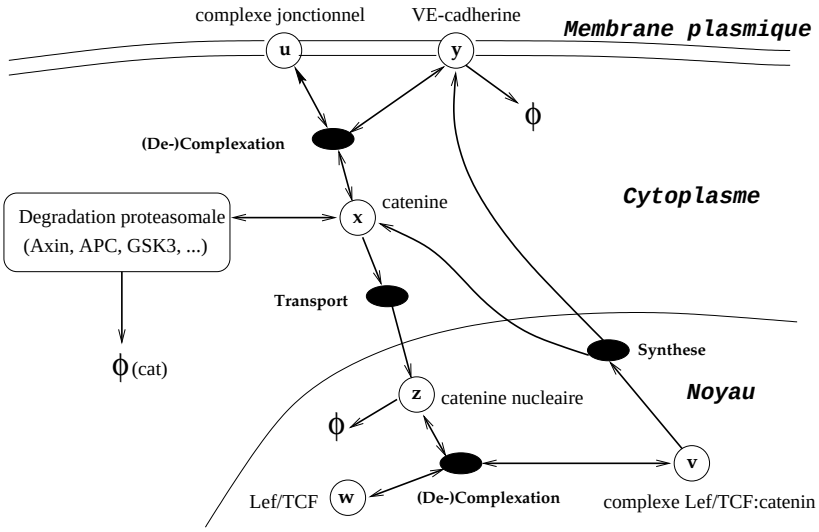

Figure 2. *Représentation informelle et schématique du système d'adhésion des cellules endothéliales. Ce graphe montre l'architecture générale du système. Les cercles représentent les espèces chimiques et les ovales noirs les réactions chimiques. ϕ représente un processus de dégradation. Les molécules sont distribuées sur trois lieux cellulaires : le cytoplasme, le noyau et la membrane plasmique. Les correspondances (variable de concentration - espèce chimique) sont les suivantes (x, β-caténine cytoplasmique non-phosphorylée), (y, VE-cadhérine dans la membrane), (z, β-caténine dans le noyau cellulaire), (u, complexe de β-caténine et trimère de VE-cadhérine), (v, complexe de Lef/Tcf et β-caténine), (w, facteur de transcription Lef/Tcf). Les flèches simples et doubles dénotent respectivement des réactions irréversibles et réversibles.*

$$\left\{ \begin{array}{l} \dot{x} = \kappa_{x0} + \kappa_{x1}.\sigma^+(z;\theta_{z1}) - \kappa_t.\sigma^+(x;\theta_{x1}) \\ \qquad -(\lambda_{x1} + \lambda_{x2}.\sigma^+(x;\theta_{x2})).x \\ \dot{z} = \kappa_t.\sigma^+(x;\theta_{x1}) - \lambda_z.z \end{array} \right. \quad [1]$$

où les paramètres κ représentent les constantes cinétiques de production ou de transport, et les paramètres λ représentent les constantes cinétiques de dégradation. L'ordre entre les deux seuils pour x est inconnu et nous devons donc considérer les deux cas suivants : $\theta_{x1} < \theta_{x2}$ (correspondant au cas où $\theta_{x1} = \theta_x^1$ et $\theta_{x2} = \theta_x^2$ avec les notations données en 2.3)et $\theta_{x2} < \theta_{x1}$ (correspondant au cas où $\theta_{x1} = \theta_x^2$ et $\theta_{x2} = \theta_x^1$). Le système complet contient un deuxième seuil θ_{z2} pour z. L'ordre entre θ_{z1} et θ_{z2} doit donc aussi être considéré.

Pour le système d'équations [1] et pour les ordres de seuils $\theta_{x2} < \theta_{x1}$ et $\theta_{z1} < \theta_{z2}$ nous obtenons le système d'équations discrètes suivant :

$$\left\{\begin{array}{lll} \Phi_x = & \mathsf{K}_{x,0}. & s^-(x,\theta_{x2}).s^-(z,\theta_{z1}) \\ + & \mathsf{K}_{x,0+1}. & s^-(x,\theta_{x2}).s^+(z,\theta_{z1}) \\ + & \mathsf{K}_{x,3}. & s^+(x,\theta_{x2}).s^-(x,\theta_{x1}).s^-(z,\theta_{z1}) \\ + & \mathsf{K}_{x,3+4}. & s^+(x,\theta_{x2}).s^-(x,\theta_{x1}).s^+(z,\theta_{z1}) \\ + & \mathsf{K}_{x,3-5}. & s^+(x,\theta_{x1}).s^-(z,\theta_{z1}) \\ + & \mathsf{K}_{x,3+4-5}. & s^+(x,\theta_{x1}).s^+(z,\theta_{z1}) \\ & & \\ \Phi_z = & \mathsf{K}_z. & s^+(x,\theta_{x1}) \end{array}\right. \quad [2]$$

Les sept paramètres discrets K dans ce modèle sont définis à partir des paramètres cinétiques réels K suivants :

$$K_{x,0} = \tfrac{\kappa_{x0}}{\lambda_{x1}}, \qquad K_{x,1} = \tfrac{\kappa_{x1}}{\lambda_{x1}}, \qquad K_{x,3} = \tfrac{\kappa_{x0}}{\lambda_{x1}+\lambda_{x2}},$$

$$K_{x,4} = \tfrac{\kappa_{x1}}{\lambda_{x1}+\lambda_{x2}}, \qquad K_{x,5} = \tfrac{\kappa_t}{\lambda_{x1}+\lambda_{x2}}, \qquad K_z = \tfrac{\kappa_t}{\lambda_z}$$

Les paramètres cinétiques discrets sont définis ainsi : $\mathsf{K}_{x,i} = d_x(K_{x,i})$, $\mathsf{K}_{x,i+j} = d_x(K_{x,i} + K_{x,j})$, $\mathsf{K}_{x,i-j} = d_x(K_{x,i} - K_{x,j})$, $\mathsf{K}_{x,i+j-l} = d_x(K_{x,i} + K_{x,j} - K_{x,l})$, $\mathsf{K}_z = d_z(K_z)$. Comme expliqué en 2.4, des inégalités déduites de ces définitions contraignent les valeurs que peuvent prendre les paramètres (par exemple $\mathsf{K}_{x,3-5} \leq \mathsf{K}_{x,3}$).

Le programme peut être utilisé aisément en simulation si l'on fixe les valeurs des paramètres, c'est-à-dire les ordre des seuils et les valeurs des K. Mais il peut aussi être utilisé pour inférer ces paramètres en imposant des contraintes comme la connaissance des états stationnaires, du nombre d'états stationnaires ou encore de la dimension minimale atteignable à partir d'un état régulier.

La requête suivante permet d'identifier tous les états stationnaires, réguliers et singuliers, de ce modèle d'identificateur `adhesion` et d'ordres de seuils $\theta_{x2} < \theta_{x1}$ et $\theta_{z1} < \theta_{z2}$:

```
modele([adhesion, P], Modele),
P = [_,[[tx,0], [tz,1]],_],  % tx, tz: ordre de seuils
modele_asynch_multival(1, Modele, [E,E]),
intsplit(E).
```

La deuxième condition impose que `Modele` est un modèle dont les ordres entre les seuils sont $\theta_{x2} < \theta_{x1}$ et $\theta_{z1} < \theta_{z2}$. La dernière condition énumère les états stationnaires possibles (les valeurs de paramètres cinétiques ne sont pas énumérées). Nous trouvons 13 états stationnaires potentiels sur les 25 = 5*5 états que possède le système. Pour la solution `E = [x,z] = [2,0]` nous obtenons sans aucune énumération une liste de paramètres complètement instanciée : $\mathsf{K}_{x,0} = \mathsf{K}_{x,0+1} = \mathsf{K}_{x,3} = \mathsf{K}_{x,3+4} =$

$\mathsf{K}_{\mathrm{x},3-5} = \mathsf{K}_{\mathrm{x},3+4-5} = 2$ et $\mathsf{K}_{\mathrm{z}} = 0$. Pour être certain de l'existence d'un état stationnaire donné, nous devons montrer par une énumération qu'il existe au moins un ensemble de paramètres acceptant cet état stationnaire. Sous cette condition d'existence, le nombre d'états stationnaires passe de 13 à 11.

Une des requêtes les plus intéressantes concerne les *chemins de réparation*. Lorsque les jonctions sont déstabilisées par des anticorps (perturbation simulant le passage des leucocytes), les VE-cadhérines sont détruites et les β-caténines sont relâchées dans le cytoplasme (Hermant *et al.*, 2003). Après cette perturbation, une reformation des jonctions est observée. Nous formalisons un chemin de réparation de la manière suivante pour notre modèle (cf. système d'équations discrètes [2]) :

1) il existe un état stationnaire sain `ES` pour lequel $\mathtt{z} < 2$. En effet une surexpression de VE-cadhérine dans l'état sain de la cellule n'est pas concevable.

2) un chemin de réparation débute par un état initial qui est un état perturbé `EP` par rapport à l'état sain `ES`. Dans la mesure où les β-caténines sont relarguées dans le cytoplasme lors de la perturbation, la valeur de concentration discrète de β-caténine cytoplasmique dans `EP` est strictement plus élevé que celle dans `ES`.

3) le chemin contient au moins un état pour lequel $\mathtt{z} = 2$. En effet, seul un tel état permet la synthèse de nouvelles VE-cadhérines nécessaires à la réparation des jonctions.

4) le chemin se termine dans l'état sain de la cellule, soit `ES`, qui est stationnaire.

La contrainte d'existence d'au moins un chemin de reconstruction pour un modèle donné est posée de la manière suivante :

– en imposant que `Modele` est le modèle considéré, que `Ch_r` est un chemin réparateur, et que `Ch_r` est un chemin existant selon `Modele`,

– puis en énumérant tous les modèles `Modele`, et enfin en énumérant au moins un chemin réparateur `Ch_r`.

Cette contrainte supprime 560 modèles sur les 600 de départ correspondant aux ordres de seuils $\theta_{x2} < \theta_{x1}$ et $\theta_{z1} < \theta_{z2}$. Nous pouvons observer par ailleurs que ces 40 modèles ont tous la propriété $\mathsf{K}_{\mathrm{z}} = 2$. La vérification formelle de cette propriété est obtenue par un échec de l'imposition à ces 40 modèles de la négation de la propriété (soit $\mathsf{K}_{\mathrm{z}} \neq 2$). Il est important de noter que si nous n'avions pas considéré les états singuliers, certains des 40 modèles répondant à la contrainte d'existence d'un chemin de réparation n'auraient pas été identifiés. Cela montre l'importance de tenir compte des états singuliers pour bien considérer tous les comportements, comme ceux décrivant des évolutions confinées sur des seuils.

Exemple : Parmi les 40 modèles qui acceptent un chemin de réparation, on trouve le modèle de paramètres $\mathsf{K}_{x,0} = \mathsf{K}_{x,0+1} = 1$, $\mathsf{K}_{x,3} = \mathsf{K}_{x,3+4} = \mathsf{K}_{x,3-5} = \mathsf{K}_{x,3+4-5} = 0$ et $\mathsf{K}_{z} = 2$ qui accepte par exemple le chemin de réparation : `[[2,0],[2,1/2],[2,1],[2,3/2],[2,2],[3/2,2],[1,2],[1/2,2],[1/2,3/2],[1/2,1],[1/2,1/2],[1/2,0],[1/2,0]]`. Ce chemin n'aurait pas été observé si nous

n'avions pas considéré les états singuliers étant donné que cette trajectoire "glisse" sur des états singuliers. En particulier, nous pouvons voir que l'état sain de la cellule considéré dans ce chemin est l'état singulier `[1/2,0]` qui est confiné sur le plus petit seuil de discrétisation de la composante x. ∎

4.2. *Segmentation de l'embryon de Drosophile*

Le système de départ analysé dans (Sánchez *et al.*, 2001) comprend les gènes répertoriés dans le tableau suivant :

identifiant de gène	nom	nombre de seuils associé
b	$bicoid$	3
c	$caudal$	2
g	$giant$	1
h	$hunchback$	3
r	$Krüppel$	2
n	$knirps$	1

Les deux gènes b et c sont des entrées du système. Les équations discrètes suivantes donnent l'évolution qualitative des concentrations des protéines associées aux quatres gènes g, h, r et n en fonction des gènes d'entrée du système b et c :

$$\begin{cases} \Phi_g = d_g(K_g + K_{g.b}.s^+(b, s_{\mathrm{b1}}) + K_{g.c}.s^+(c, s_{\mathrm{c2}}) + K_{g.h}.s^-(h, s_{\mathrm{h1}}) + K_{g.r}.s^-(r, s_{\mathrm{r1}})) \\ \Phi_h = d_h(K_h + K_{h.\varepsilon}.\varepsilon + K_{h.\delta}.\delta + K_{h.\rho}.\rho + K_{h.b}.s^+(b, s_{\mathrm{b1}}) + K_{h.r}.s^-(r, s_{\mathrm{r2}})) \\ \Phi_r = d_r(K_r + K_{r.b}.s^+(b, s_{\mathrm{b1}}) + K_{r.h}.\mu + K_{r.g}.s^-(g, s_{\mathrm{g1}}) + K_{r.n}.s^-(n, s_{\mathrm{n1}})) \\ \Phi_n = d_n(K_n + K_{n.b}.s^+(b, s_{\mathrm{b1}}) + K_{n.c}.s^+(c, s_{\mathrm{c1}}) + K_{n.g}.s^-(g, s_{\mathrm{g1}}) + K_{n.h}.s^+(h, s_{\mathrm{h2}})) \end{cases}$$

avec

$$\begin{aligned} \varepsilon &= s^+(b, s_{\mathrm{b3}}).s^+(h, s_{\mathrm{h1}}) \\ \delta &= s^+(b, s_{\mathrm{b2}}).s^+(h, s_{\mathrm{h1}}) \\ \rho &= s^+(b, s_{\mathrm{b1}}).s^+(h, s_{\mathrm{h1}}) \\ \mu &= s^+(h, s_{\mathrm{h1}}).s^-(h, s_{\mathrm{h3}}) \end{aligned}$$

L'ordre des seuils est ici connu : ce sont les numéros des seuils qui donnent leur ordre ($s_{\mathrm{c1}} < s_{\mathrm{c2}}$ par exemple).

Le long de l'axe antérieur-postérieur, l'embryon de drosophile se segmente en 4 régions nommées A, B, C et D. La segmentation se produit d'après le profil d'expression des différents gènes. Le tableau qui suit donne pour chacune des régions les valeurs de concentrations discrètes de l'état stationnaire observé :

région	A	B	C	D
b	3	2	1	0
c	0	0	1	2
g	1	0	0	1
h	3	2	1	0
r	0	2	1	0
n	0	0	1	0

La requête permettant d'imposer ces comportements (stationnarités) doit contraindre chacune des régions, définies par des valeurs de concentrations discrètes b et c, à accepter l'état stationnaire régulier associé donné dans le tableau précédent.

Exemple : Pour la région A définie par $\mathsf{b} = 3$ et $\mathsf{c} = 0$ il s'agit d'imposer la stationnarité de l'état $[\mathsf{g}, \mathsf{h}, \mathsf{r}, \mathsf{n}] = [1, 3, 0, 0]$. ∎

Il est à signaler que les 4 modèles (correspondant aux 4 régions) qui doivent être créés possèdent tous les mêmes paramètres cinétiques discrets K (que nous ne donnons pas ici), mais pas les mêmes paramètres d'entrée b et c.

Nous obtenons alors instantanément une solution donnant les valeurs en intention des différents paramètres cinétiques discrets K. Certaines de ces valeurs sont instanciées tandis que certaines sont juste contraintes à appartenir à un intervalle d'entiers.

Exemple : $\mathsf{K}_{h.\varepsilon r} = 3$, $\mathsf{K}_{r.bgh} = 1$ sont des paramètres cinétiques discrets instanciés, alors que $\mathsf{K}_{h.dr}$ doit appartenir à l'intervalle $[2, 3]$ et peut donc valoir a priori 2 ou 3. ∎

Nous pouvons aussi facilement vérifier qu'un jeu de paramètres donné répond bien aux contraintes de la requête précédente. Il suffit d'ajouter la contrainte signifiant que chaque paramètre possède la valeur donnée. Ainsi, nous avons pu confirmer aisément que les valeurs des paramètres utilisées dans les simulations de Sanchez et *al.* (2001) répondent bien à ces contraintes.

5. Discussion, bilan et perspectives

Nous présentons successivement une discussion portant sur des travaux proches, un bilan actuel et les perspectives de notre travail. Les travaux qui semblent connectés sous des aspects très différents sont les suivants. Devloo (Devloo *et al.*, 2003) utilise la technologie CSP (*Constraint Satisfaction Problem*) pour résoudre un problème très circonscrit : l'identification des états stationnaires éventuellement singuliers de modèles de Thomas complètement instanciés.

Un des problèmes important abordé est le choix du langage permettant d'exprimer des propriétés temporelles. Chabrier et Fages (Chabrier *et al.,* 2003) et Bernot (Bernot *et al.,* 2004) utilisent le langage CTL (*Computational Tree Logic*) pour exprimer des propriétés, en conjonction avec des algorithmes de *model-checking* pour vérifier ces propriétés CTL. L'étude de Fages considère que le système biologique est connu, tandis que celle de Bernot cherche à trouver des paramètres à partir de comportements en générant toutes les instances de modèle. Clairement, CTL et les méthodes de model-checking sont à considérer, et tout particulièrement le *model-checking* symbolique comme dans (Delzanno *et al.,* 1999). Adapté à notre problème, ce dernier peut nous aider à gagner en efficacité (spécialement pour des requêtes universelles), mais aussi pour la terminaison (pour éviter des chemins trop longs) et enfin et surtout pour imposer des comportements via des formules CTL.

Thieffry et *al.* (Thieffry *et al.,* 1993) et de Jong et *al.* (2004) ont conçu chacun un logiciel de modélisation et de simulation qualitative. De Jong et *al.* considèrent les états singuliers. Thieffry et *al.* ne considèrent pas les états singuliers comme élément d'un chemin, mais déterminent des valeurs de paramètres assurant l'existence d'états singuliers persistants en exploitant la présence de circuits dans le graphe d'interaction.

Au total, il nous apparait qu'à part l'étude de Cohen (Cohen, 2001), qui a suggéré l'approche initialement, on ne trouve pas de travaux proposant les mêmes objectifs ou les mêmes outils que ceux exposés ici. Le projet le plus proche est certainement (Bernot *et al.,* 2004), mais il ne considére pas les états singuliers et procède par énumération brutale plutôt que par une modélisation "exécutable" sous forme de contraintes.

On peut dire que l'objectif ambitieux de ce travail, apprendre un modèle à partir de comportements, a été partiellement atteint et qu'au moins les résultats obtenus sont encourageants. La faisabilité de l'approche contrainte pour les modèles LMAES paraît possible. Succinctement les résultats acquis sont les suivants :

– du point de vue spécification des modèles LMAES, nous avons dû approfondir une formalisation assez difficile à appréhender (de Jong *et al.,* 2004) pour obtenir une modélisation sous forme de contraintes qui, il faut le noter, ne fait aucune présupposition sur ce qui est connu et ce qui ne l'est pas. Cela nous a amené en particulier à de nouvelles formulations pour la définition d'un état successeur dans le cas singulier (voir 2.5).

– du point de vue de la mise en œuvre en PLC des modèles LMAES, nous disposons maintenant d'un logiciel déclaratif original. Il est capable par exemple à la fois de contraindre un comportement à être accepté par un modèle instancié, de contraindre un modèle à accepter un comportement connu et bien sûr de vérifier qu'un comportement est accepté par un modèle. Pour cela, nous avons dû concevoir une représentation des équations logiques, introduire des contraintes redondantes efficaces pour déterminer un état focal (voir 3.3) et concevoir une architecture générale (voir 3.4) respectant la spécification initiale et utilisant des contraintes sophistiquées pour respecter la métho-

dologie 'contrainte' (voir 3.2). Il est à noter que la complexité en terme de nombre de contraintes (voir 3.5) d'une requête concernant l'acceptation d'un comportement par un modèle apparaît acceptable.

– d'un point de vue validation de l'approche, nous avons mis en œuvre différents modèles dont deux sont présentés ici succintement. Les résultats montrent l'approche directe et automatique de questionnement. Les résultats obtenus sur l'existence de comportements réparateurs (voir 4.1) dans le cas singulier sont neufs : ils permettent une détection drastique de modèles incorrects de ce point de vue (élimination de 560 modèles sur 600).

Il reste à signaler que les difficultés rencontrées, soulevées à propos de modèles biologiques, relèvent de problèmes plus généraux : par exemple, la pose de contraintes sur des chemins de taille non fixée ou plus généralement l'imposition (et non pas uniquement la vérification) de propriétés CTL se retrouvent dans l'étude des systèmes de transition en général (Delzanno *et al.*, 1999).

Parmi les développements à entreprendre on peut citer des études sur :

– l'introduction de contraintes redondantes. Nous pensons ici à la prise en compte de résultats liant le graphe d'interaction et le graphe de transition : par exemple celui (Snoussi *et al.*, 1993) posant que seuls les circuits dans le graphe d'interaction induisent les états singuliers persistants. Ceci permettra de résoudre un des problèmes mentionnés en 3.3. Une étude est en cours pour déterminer si une certaine connaissance de l'ordre des états atteignables ne pourrait pas être obtenue au vu de certaines propriétés du graphe d'interaction.

– la définition d'un langage d'expression des propriétés. Il s'agit typiquement d'imposer des propriétés définissables dans un langage comme CTL, éventuellement universelles. La difficulté provient d'une part de l'expression sous une forme finie et minimale en termes de contraintes de la propriété et d'autre part de l'efficacité de la résolution des contraintes posées, efficacité qui est éventuellement dépendante des heuristiques d'énumération utilisées.

– la classification et la représentation en intention de modèles. L'idée ici consiste d'une part à établir des classes d'équivalence de modèles du point de vue du graphe de transitions, d'autre part à représenter à l'intention des biologistes les modèles intéressants sous une forme intentionnelle compréhensible définie à l'aide de prédicats simples (par exemple égalité, inégalité, appartenance à un intervalle). La PLI (Programmation Logique Inductive) est un des moyens d'apprentissage envisagé.

– l'extension du modèle. Il s'agit d'apporter une plus grande finesse (et vraisemblance biologique) dans la définition d'un état successeur. Par exemple, dans (Thomas *et al.*, 2001), on trouve la suggestion d'introduire la notion de "délai" favorisant tel successeur plutôt qu'un autre dans tel état. Aussi, on peut penser à restreindre les successeurs dans le cas singulier comme dans le cas régulier (où une seule composante change).

6. Bibliographie

Bernot G., Comet J.-P., Richard A., Guespin J., « Application of formal methods to biological regulatory networks : extending Thomas' asynchronous logical approach with temporal logic », *Journal of Theoretical Biology*, vol. 229, p. 339-347, 2004.

Chabrier N., Fages F., « Symbolic Model Checking of Biochemical Networks », *in* , C. Priami (ed.), *Computational Methods in System Biology, LNCS 2602*, Springer, p. 149-162, 2003.

Cohen J., « Classification of approaches used to study cell regulation : Search for a unified view using constraints and machine learning », *ETAI, Machine Intelligence 18. Linkoping Electronic Articles in Computer and Information Science ISSN*, 2001.

Colmerauer A., *Prolog – Constraints Inside, Manuel de Prolog, PROLOGIA, Case 919*, 13288 Marseille cedex 09, France. 1996.

Corblin F., « Inférence et simulation de réseaux biologiques logiques à l'aide de la Programmation Logique avec Contraintes (PLC) », 2005, Rapport de magistère d'informatique $3^{ème}$ année, Université Joseph-Fourier, Grenoble.

de Jong H., Gouzé J.-L., Hernandez C., Page M., Sari T., Geiselmann J., « Qualitative Simulation of Genetic Regulatory Networks Using Piecewise-Linear Models », *Bulletin of Mathematical Biology*, vol. 66, p. 301-340, 2004.

Delzanno G., Podelski A., « Model Checking in CLP », *Proceeding of the 5th International Conference TACAS'99, LNCS 1579*, Springer, p. 223-239, 1999.

Devloo V., Hansen P., Labbé M., « Identification of All Steady States in Large Biological Systems by Logical Analysis », *Bulletin of Mathematical Biology*, vol. 65, p. 1025-1051, 2003.

Fanchon E., Corblin F., Trilling L., Hermant B., Gulino D., « Modeling the Molecular Network Controlling Adhesion Between Human Endothelial Cells : Inference and Simulation Using Constraint Logic Programming », *in* , V. Danos, , V. Schachter (eds), *Lecture Notes in Bioinformatics – Computational Methods in Systems Biology 2004*, vol. 3082, p. 104-118, 2005.

Hermant B., Bibert S., Concord E., Dublet B., Weidenhaupt M., Vernet T., Gulino-Debrac D., « Identification of Proteases Involved in the Proteolysis of Vascular Endothelium Cadherin during Neutrophil Transmigration », *The Journal of Biological Chemistry*, vol. 278, p. 14002-14012, 2003.

Kaufman M., Thomas R., « Model analysis of the bases of multistationnarity in the humoral immune response », *J. Theo. Biol.*, vol. 129, p. 141-162, 1987.

Sánchez L., Thieffry D., « A Logical Analysis of the *Drosophila* Gap-gene System », *J. theor. Biol.*, vol. 211, p. 115-141, 2001.

Snoussi E. H., « Qualitative dynamics of piecewise-linear differential equations : A discrete mapping approach », *Dynamics and Stability of Systems*, vol. 4, p. 189-207, 1989.

Snoussi E. H., Thomas R., « Logical Identification of All Steady States : The Concept of Feedback Loop Characteristic States », *Bulletin of Mathematical Biology*, vol. 55, p. 973-991, 1993.

Thieffry D., Colet M., Thomas R., « Formalisation of Regulatory Networks : a Logical Method and Its Automation », *Math. Modelling and Sci. Computing*, vol. 2, p. 144-151, 1993.

Thomas R., Kaufman M., « Multistationarity, the Basis of Cell Differentiation and Memory. II. Logical Analysis of Regulatory Networks in Term of Feedback Circuits », *Chaos*, vol. 11, p. 180-195, 2001.

Fabien Corblin *est actuellement doctorant à l'Université Joseph Fourier (UJF - Grenoble 1). Très tôt, il s'est intéressé au monde scientifique (astronomie, logique) tout en poursuivant des études générales à dominante scientifique (mathématiques) aux lycées de Romans et de Valence. En 2002, il commence un Magistère à l'UJF, où il étudie l'informatique et rejoint une équipe de recherche en bioinformatique. En 2005, il débute une thèse et, en parallèle, assure des fonctions de moniteur à l'UJF où il enseigne la programmation algorithmique et logique.*

Eric Fanchon *a soutenu une thèse en physique de la matière condensée à l'Université Joseph Fourier de Grenoble. Il s'est ensuite orienté vers la cristallographie des protéines et les méthodes de résolution de structures moléculaires tridimensionnelles. Entré au CNRS en 1990, il a travaillé à l'Institut de Biologie Structurale et à l'European Synchrotron Radiation Facility, notamment pour concevoir et développer un logiciel de contrôle de ligne synchrotron et de collecte de données de diffraction. Depuis 2000 ses travaux sont orientés vers la biomathématique et la bioinformatique : développement de méthodes formelles pour la modélisation de réseaux biologiques tels que les réseaux génétiques ou les réseaux métaboliques.*

Laurent Trilling *est professeur à l'Université Joseph Fourier (Grenoble 1) après avoir exercé à l'Université de Montréal et à l'Université de Rennes 1. Ses intérêts ont évolués au fil des années, depuis les langages de programmation, la compilation, l'intelligence artificielle (logiques non standard, programmation logique), la géométrie dynamique, la planification jusqu'à la bioinformatique. Il aimerait se consacrer à l'élucidation et la construction de réseaux biologiques à l'aide d'outils de programmation déclarative.*

ANNEXE POUR LE SERVICE FABRICATION

A FOURNIR PAR LES AUTEURS AVEC UN EXEMPLAIRE PAPIER
DE LEUR ARTICLE ET LE COPYRIGHT SIGNE PAR COURRIER
LE FICHIER PDF CORRESPONDANT SERA ENVOYE PAR E-MAIL

1. ARTICLE POUR LA REVUE :
   Techniques et science informatiques

2. AUTEURS :
   *Fabien Corblin * ** — Eric Fanchon ** — Laurent Trilling **

3. TITRE DE L'ARTICLE :
   Modélisation de réseaux biologiques discrets en Programmation Logique par Contraintes (PLC)

4. TITRE ABRÉGÉ POUR LE HAUT DE PAGE MOINS DE 40 SIGNES :
   Modélisation en PLC

5. DATE DE CETTE VERSION :
   11 août 2006

6. COORDONNÉES DES AUTEURS :

   – adresse postale :

   * IMAG-LSR, Université Joseph Fourier BP 53, 38041 Grenoble Cedex 9, France

   Fabien.Corblin@imag.fr Laurent.Trilling@imag.fr

   ** LCM, Institut de Biologie Structurale Jean Pierre Ebel,
   CEA-CNRS-Université Joseph Fourier,
   41, rue Jules Horowitz, 38027 Grenoble Cedex 1, France

   Eric.Fanchon@ibs.fr

   – téléphone : 00 00 00 00 00

   – télécopie : 00 00 00 00 00

   – e-mail : Roger.Rousseau@unice.fr

7. LOGICIEL UTILISÉ POUR LA PRÉPARATION DE CET ARTICLE :
   LaTeX, avec le fichier de style `article-hermes.cls`,
   version 1.2 du 03/03/2005.

8. FORMULAIRE DE COPYRIGHT :
   Retourner le formulaire de copyright signé par les auteurs, téléchargé sur :
   `http://www.revuesonline.com`

SERVICE ÉDITORIAL – HERMES-LAVOISIER
14 rue de Provigny, F-94236 Cachan cedex
Tél : 01-47-40-67-67
E-mail : revues@lavoisier.fr
Serveur web : http://www.revuesonline.com