Programmes biochimiques et algorithmes mixtes analogiques-digitaux dans la cellule

François Fages, Guillaume Le Guludec

20164557

Programmes biochimiques et algorithmes mixtes analogiques-digitaux dans la cellule

François Fages et Guillaume Le Guludec

Inria Saclay-Ile de France,
EP Lifeware, Bât. Alan Turing, Campus de l'École Polytechnique,
1 rue Honoré d'Estienne d'Orves, F-91120 Palaiseau
Francois.Fages@inria.fr

RÉSUMÉ. *Dans ce chapitre nous adoptons un point de vue d'informaticien pour chercher à comprendre les calculs qu'effectue la cellule pour se maintenir dans son environnement, traiter les signaux qu'elle capte, et prendre les décisions qui déterminent son destin. L'aspect continu de beaucoup d'interactions protéiques nous conduit à considérer des modes de calcul mixtes analogiques-digitaux, pour lesquels des résultats récents en théorie de la calculabilité et de la complexité analogique établissent des liens fondamentaux avec la programmation classique. Nous dérivons de ces résultats un compilateur de spécifications comportementales en systèmes de réactions biochimiques que l'on peut comparer aux circuits naturels résultats de l'évolution. Nous illustrons cette démarche par l'exemple du module de signalisation MAPK qui a une fonction de convertisseur analogique-digital dans la cellule, et par le contrôle du cycle cellulaire.*

ABSTRACT. *In this chapter we take an IT perspective in seeking to understand how computation is carried out in the cell to maintain itself in its environment, process signals and make the decisions that determine its fate. The continuous nature of many protein interactions leads us to consider mixed analog-digital computation models, for which recent results in the theory of analog computability and complexity establish fundamental links with classical programming. We derive from these results a compiler of behavioural specifications into biochemical reactions which can be compared to natural circuits acquired through evolution. We illustrate this approach through the example of the MAPK signaling module which has a function of analog-digital converter in the cell, and through the cell cycle control.*

MOTS-CLÉS : *Programmation biochimique, calcul analogique, signalisation, cycle cellulaire*

KEYWORDS: *Biochemical programming, analog computation, cell signalling, cell cycle*

1. Introduction

"La mathématique est l'art de donner le même nom à des choses différentes" Henri Poincaré

Une des leçons de l'informatique est que le calcul digital permet de passer à l'échelle des très grands programmes, contrairement au calcul analogique. Cependant quand on observe les processus cellulaires avec un regard d'informaticien, même si on retrouve bien le caractère discret de l'activation en tout ou rien des gènes, on est frappé par la prépondérance des activations graduelles des complexes protéiques, et par l'importance du temps que prennent ces transformations au sein de grands réseaux d'interactions. Dans cet article nous cherchons à assumer l'importance du calcul analogique dans la cellule, et jetons les bases d'une théorie du calcul biochimique dans le but d'analyser les circuits naturels en se dotant d'outils de spécification de leurs fonctions, de compilation de ces spécifications en circuits synthétiques, et de comparaison de leurs complexités de calcul.

2. Programmes biochimiques

2.1. *Syntaxe*

Formellement, une réaction biochimique est ici une règle de la forme

$$a_1x_1 + a_2x_2 + \ldots \xrightarrow{f} b_1x_1 + b_2x_2$$

où les a_i et b_j sont les *coefficients stœchiometriques*, et les x_i sont les espèces chimiques du système. f est la *fonction cinétique* de la réaction. Les *réactions élémentaires* ont au plus deux réactants, et sont de cinq formes notoires : liaison ou complexation $x + y \to z$, déliaison $z \to x + y$, transformation ou transport $x \to y$, synthèse $x \to x + y$ (ou $\_ \to y$) et dégradation passive $x \to \_$ ou active $x + y \to y$.

On supposera que les réactions respectent la *loi d'action de masse*, c'est-a-dire que la fonction cinétique de chaque réaction est de la forme $f = kx_1^{a_1} \ldots x_n^{a_n}$ avec k une constante. On note alors la réaction sous la forme $R \xrightarrow{k} P$ (et omettons k si $k = 1$). Les autres fonctions cinétiques classiques s'obtiennent par réduction de systèmes de réactions élémentaires sous des hypothèses d'états quasi-stationnaires, par exemple, cinétique de Michaelis-Menten en $\frac{v*x}{c+x}$ pour une réaction enzymatique de transformation d'un substrat x par une enzyme en plus faible quantité, ou cinétique de Hill en $\frac{v*x^n}{c^n+x^n}$ pour une réaction enzymatique allostérique coopérative (Segel, 1984).

Notons que la conservation de la matière peut ne pas être respectée comme par exemple dans le cas des réactions de synthèse ou dégradation. De telles réactions sont physiquement impossibles mais doivent s'interpréter comme des abstractions de réactions physiques qui oublient certaines espèces moléculaires. Par exemple, une réaction formelle de synthèse $\_ \to x$ peut représenter en fait la transcription d'un gène supposé actif en ARN x ou directement en protéine x, ou bien encore l'activation d'une protéine $\underline{x}$ par une réaction de phosphorylation $\underline{x} + y \to x + y$, en oubliant sa forme

inactive $\underline{x}$ non phosphorylée ainsi que l'enzyme kinase y toutes deux supposées présentes. Dans ce dernier cas, la réaction de dégradation $x \rightarrow \_$ peut alors dénoter en fait la réaction de déphosphorylation $x + z \rightarrow \underline{x} + z$ de retour à la forme inactive sous l'effet d'une phosphatase z également négligée.

Ces réactions formelles ne préjugent donc pas de leur implémentation physique.

2.2. *Sémantiques*

De tels systèmes de réactions peuvent s'interpréter de différentes manières que l'on peut relier par des relations d'approximation (Gillespie, 1977) ou d'abstraction (Fages *et al.,* 2008a), pour former une hiérarchie de sémantiques, notamment dans le cadre informatique de l'interprétation abstraite (Cousot *et al.,* 1977)

La sémantique différentielle associe des concentrations continues aux espèces moléculaires et le système d'équations différentielles

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = \sum_j (b_i^j - a_i^j) f_j(x_1, \ldots, x_n)$$

où l'indice j parcourt l'ensemble des règles de réaction du système. On dit qu'un système est *à l'état stationnaire* sur x_i lorsque $\frac{\mathrm{d}x_i}{\mathrm{d}t} = 0$.

La sémantique stochastique associe un nombre entier de molécules (ou niveau de concentration) à chaque espèce moléculaire et une chaîne de Markov à temps continu, dans laquelle les probabilités de transition sont définies à partir des fonctions cinétiques par normalisation, et le temps de la prochaine réaction est donné par une loi exponentielle.

La sémantique de réseau de Petri travaille sur les mêmes états que la sémantique stochastique mais oublie la cinétique, les probabilités et le temps continu (abstraction par foncteur d'oubli). Les notions d'invariants des réseaux de Petri donnent néanmoins des informations sur les sémantiques différentielles et stochastiques, comme des lois de conservation (P-invariants) et les flux extrêmes (T-invariants) qui sont un outil essentiel d'analyse des réseaux métaboliques.

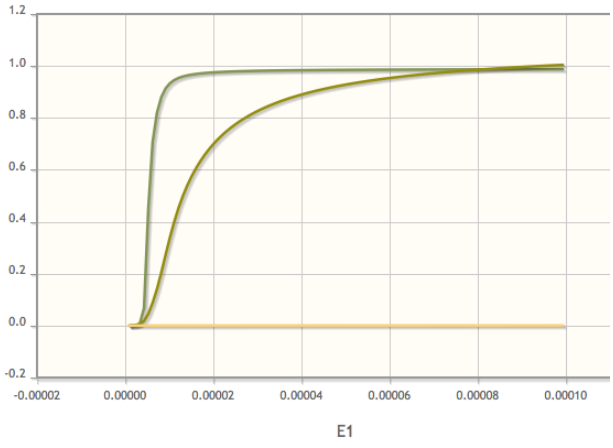
La sémantique booléenne abstrait encore les états discrets en états booléens concernant la présence/absence des espèces moléculaires (abstraction $(> 0) : \mathbb{N} \rightarrow \{0, 1\}$, ou $(> \theta)$ pour un seuil θ) et associe un système de transition booléen asynchrone au système de réactions. Les outils de *model-checking* permettent alors de vérifier des propriétés d'accessibilité et de trajectoires dans des grands systèmes de réaction dont on ne connaît pas la cinétique (Chabrier-Rivier *et al.,* 2004). Ils permettent notamment de montrer que si un comportement n'est pas possible dans la sémantique booléenne alors il ne l'est pas non plus dans la sémantique stochastique quelles que soient les fonctions cinétiques (Fages *et al.,* 2008a).

2.3. *Exemple des réseaux de signalisation MAPK*

Les circuits MAPK (pour *mitogen-activated protein kinases*) sont des modules de signalisation extrêmement courants que l'on retrouve en plusieurs copies dans les organismes eucaryotes. Dans ces voies de signalisation les protéines activées par phosphorylation sont elle-mêmes des kinases qui catalysent en cascade d'autres phosphorylations. Ainsi, la cascade MAPK a trois étages de phosphorylation pour un total de 30 réactions élémentaires : l'entrée `E1` de la cascade, directement liée au récepteur membranaire, catalyse la phosphorylation de la kinase `KKK` du premier étage, qui à son tour phosphoryle deux fois la kinase `KK` du second étage, qui sous cette forme doublement phosphorylée phosphoryle la protéine `K` du dernier étage de la cascade, qui elle-même doublement phosphorylée `PP_K`, est capable de migrer dans le noyau et d'activer ou inhiber la transcription de certains gènes.

(Huang *et al.,* 1996) ont proposé une explication pour cette structure en montrant que les cascades MAPK exhibent une réponse (à l'état stationnaire) en forme de fonction de Hill (similaire donc à la réponse d'une chaîne de réactions enzymatiques allostériques coopératives), c'est à dire qu'en notant (u, y) le couple entrée sortie du système, on pouvait approximer le diagramme dose-réponse par une équation de la forme $y(u) \approx \lambda \frac{u^\alpha}{c^\alpha + u^\alpha}$ avec α de l'ordre de 4.9 au troisième niveau `PP_K` $\alpha \sim 1.7$ au second `PP_KK` et $\alpha = 1$ au premier niveau `P_KKK` qui est Michaelien :

```
biocham: present(E1=3.0e-5, KKK=0.003, KK=1.2, K=1.2,
                 E2=0.0003, KKPase=0.0003, Kpase=0.12).
biocham: dose_response(E1, 1e-6, 1e-4, 500, {PP_K, PP_KK, P_KKK}).
```



La cascade MAPK agit donc comme un *convertisseur analogique-digital* qui transforme un signal continu en entrée (concentration des recepteurs activés `E1`) en signal digital en sortie (activation en tout ou rien de la protéine `PP_K`).

3. Spécifications logiques des comportements

La logique temporelle CTL fournit un langage d'expression très puissant pour analyser la dynamique booléenne d'un système de réactions (Chabrier-Rivier *et al.,* 2004). Dans ce langage logique, une formule propositionnelle définit un ensemble d'états (l'ensemble des états qui la satisfont), et les opérateurs modaux définissent la valeur de vérité des formules dans un ou tous les états futurs, sur une ou toutes les branches du système de transition booléen associé aux réactions. Au dela des propriétés d'accessibilité à partir d'un ensemble d'états initiaux, on peut exprimer l'atteinte d'un état stationnaire (dont on peut ne pas sortir), d'un état stable (dont on ne peut pas sortir), des points de passage obligés (*checkpoints*) pour atteindre un autre état, de possibilités d'oscillations, etc.

L'exemple MAPK vérifie par exemple des propriétés d'accessibilité d'états stationnaires ou stables mais aussi d'oscillation vérifiées automatiquement par les méthodes de *model-checking*.

```
biocham: generate_ctl.
reachable(steady(K))
reachable(steady(P_K))
reachable(steady(PP_K))
...
checkpoint(E1,P_KKK)
...
oscil(KKK)
oscil(KK)
oscil(K)
```

De plus, si on ne retient comme spécification du comportement que la propriété d'activation de la sortie Kpp, la commande `biocham: reduce_model(reachable(PP_K))` détermine par *model-checking* que les 15 réactions inverses de déphosphorylation ne sont pas utiles et peuvent être automatiquement éliminées du modèle.

Les propriétés d'oscillations de la sémantique booléenne abstraite de MAPK n'impliquent pas qu'elles se produisent dans la sémantique différentielle. L'intuition dirait plutôt qu'elles ne peuvent s'y produire car MAPK est une cascade de réactions dirigées de l'étage d'entrée vers l'étage de sortie sans réaction inverse. On peut néanmoins bien obtenir de telles oscillations dans la sémantique différentielle pour certaines valeurs (8%) des paramètres et des concentrations initiales (Qiao *et al.,* 2007). Cela peut se vérifier en exprimant la condition d'oscillation en logique temporelle du premier-ordre avec contraintes quantitatives (Fages *et al.,* 2014) et en utilisant des algorithmes d'optimisation stochastique pour trouver des valeurs de paramètres satisfaisant ces contraintes (Rizk *et al.,* 2011) . Le caractère contre intuitif de ces oscillations tient au fait qu'il n'y a pas de réaction de retrocontrôle négatif, mais une réaction de complexation entre la kinase de l'étage supérieur et son substrat de l'étage inférieur qui crée une *influence négative* du substrat inférieur vers la kinase supérieure (par séquestration), et donc un circuit négatif dans le *graphe d'influence* (et non pas de réaction)

ce qui est bien une condition nécessaire pour les oscillations (Thomas, 1981, Snoussi, 1998, Fages *et al.,* 2008b, Fages *et al.,* 2015).

4. Spécifications analogiques

4.1. *Théorie de la calculabilité et de la complexité analogique*

"The varied titles of Turing's published work disguise its unity of purpose. The central problem with which he started, and to which he constantly returned, is the extent and the limitations of mechanistic explanations of nature." Max Newman

La thèse de Church-Turing énonce qu'il n'y a qu'une seule notion de calculabilité, et de fait, tous les procédés de calcul mécanistes imaginés jusqu'à présent se sont toujours avérés codables dans des machines de Turing. On peut ainsi donner un sens informatique aux calculs analogiques en considérant la notion de calculabilité de l'analyse computationnelle qui repose sur celle de calcul sur les réels en précision arbitraire, mais finie, par des machines de Turing :

Définition 1. *Un nombre réel $r \in \mathbb{R}$ est calculable (resp. en temps polynomial) au sens de l'analyse computationnelle s'il existe un programme d'approximation de r en précision arbitraire, i.e. une machine de Turing qui prend en entrée une précision $p \in \mathbb{N}$ et rend en sortie (resp. en temps polynomial en fonction de p) un nombre rationnel $r_p \in \mathbb{Q}$ t.q. $|r - r_p| \leq 2^{-p}$.*

Définition 2. *Une fonction $f : [a, b] \to \mathbb{R}$ est calculable (resp. en temps polynomial) s'il existe une machine de Turing qui calcule $f(x)$ (resp. en temps polynomial) avec un oracle pour x.*

Le *General Purpose Analog Computer* (GPAC) de Shannon (Shannon, 1941) est un modèle de calcul analogique par circuits blocs. On se donne un ensemble d'entrées dont le temps $t, x, y, z, \ldots$ et quatre types de blocs : constantes, sommes, produits, et intégrale de Stieltjes d'une variable par rapport à une autre variable. Toutefois dans cette présentation originelle, certains circuits peuvent ne pas avoir de solution, ou bien en avoir plusieurs. Ce problème a été résolu par (Graça *et al.,* 2003) qui ont donné une définition satisfaisante de fonction générable par un GPAC en terme de solution aux problèmes aux valeurs initiales dans les équations différentielles polynomiales (PIVP) :

Définition 3. *(Graça* et al., *2003) Une fonction $f : \mathbb{R} \to \mathbb{R}$ est* GPAC-générable *si elle est une composante de la solution $y(t)$ de l'équation différentielle ordinaire $y'(t) = p(y(t))$ pour un vecteur de polynômes $p \in \mathbb{R}^n[\mathbb{R}^n]$ et de valeurs initiales $y(0)$.*

Par exemple, le GPAC `(a = integral integral -1*a)` (où les intégrations sont effectuées par rapport au temps t) donne directement $y''(t) = -y(t)$ et *génère* la fonction $cos(t)$ avec $y(0) = 1$. Cette classe de fonctions jouit d'un certain nombre de

propriétés, comme la stabilité par addition, multiplication et composition, et contient en outre des fonctions élémentaires comme les fonctions trigonométriques, exponentielle, logarithmes, etc. Cette notion de générabilité a pendant un certain temps été considérée comme synonyme de calculabilité, ce qui faisait du GPAC un modèle de calcul plus faible que l'analyse computationnelle. On peut cependant définir en terme de GPAC une notion à la fois naturelle et puissante de calculabilité, toujours par PIVP, mais par approximation du résultat sur une composante du système pour toute entrée :

Définition 4. *(Graça* et al., *2003) Une fonction* $f : [a, b] \to \mathbb{R}$ *est* GPAC-calculable *s'il existe des vecteurs de polynômes* $p \in \mathbb{R}^n[\mathbb{R}^n]$ *et* $q \in \mathbb{R}^n[\mathbb{R}]$ *et une fonction* $y : \mathbb{R}^n \to \mathbb{R}^n$ *tels que* $y(0) = q(x)$, $y'(t) = p(y(t))$ *et* $\lim_{t\to\infty} y_1(t) = f(x)$.

Le calcul de f avec l'argument x consiste donc à mettre le système dans un état dépendant polynomialement de x, puis à laisser le système évoluer selon la dynamique décrite par p. Le résultat du calcul est alors obtenu dans la première composante du système, avec une précision d'autant plus grande que l'on attend longtemps.

Le théorème suivant dû à (Bournez *et al.,* 2006) réconcilie alors parfaitement les notions de calculabilité digitale et analogique :

Théorème 1. *(Bournez* et al., *2006) Une fonction est calculable au sens de l'analyse computationnelle si et seulement si elle est GPAC-calculable.*

De plus, (Pouly, 2015) en déduit, pour la première fois, une caractérisation purement analogique de la classe de complexité Ptime. Remarquons d'abord qu'une définition naïve de la complexité en terme de temps à attendre pour obtenir une précision fixée ne peut convenir : il est en effet toujours possible de contracter le temps dans les PIVP par un changement de variable. Pouly résout ce problème en prenant comme mesure de complexité de calcul tout simplement la *longueur de la trajectoire* :

Définition 5. *(Pouly, 2015) Une fonction* $f : [a, b]^n \to \mathbb{R}^m$ *est dite* Ω*-calculable en longueur (avec* $\Omega : \mathbb{R}_+^2 \to \mathbb{R}$*) s'il existe* $p \in \mathbb{R}^d[\mathbb{R}^d]$, $q \in \mathbb{R}^d[[a, b]^n]$ *tels que pour tout* $x \in \mathrm{dom} f$*, il existe* $y : \mathbb{R}_+ \to \mathbb{R}^d$ *telle que pour tout* $t \in \mathbb{R}_+$ *:*

– $y(0) = q(x)$ *et* $y'(t) = p(y(t))$

– *pour tout* μ*, si* $\int_0^t |y'(\tau)|\mathrm{d}\tau \geq \Omega(|x|, \mu)$*, alors* $|y_{1..m}(t) - f(x)| \leq e^{-\mu}$

Théorème 2. *(Pouly, 2015) Les fonctions* Ω*-calculables en longueur, où* Ω *est un polynôme, sont exactement les fonctions calculables en temps polynomial au sens de l'analyse computationnelle.*

4.2. *Calculabilité et complexité algorithmique biochimiques*

Les résultats précédents fournissent des fondations solides pour étudier le calcul analogique biochimique. Toutefois un système biochimique est un système dynamique sur le cône $\mathbb{R}_+^n$, où l'état est la donnée des concentrations à valeurs *positives* des

espèces du système. La dynamique donnée par la loi d'action de masse conduit à un système de la forme $\frac{\mathrm{d}y}{\mathrm{d}t} = p(y(t))$ avec $p(y)_i = \sum_j (b_i^j - a_i^j) k_j y_1^{a_1^j} \dots y_n^{a_n^j}$. On voit apparaître des contraintes supplémentaires, par rapport aux GPAC généraux : les composantes y_i doivent toujours être positives, et les monônes de p_i dont le coefficient est négatif doivent avoir un exposant en y_i non nul. Ces contraintes sont des conditions nécessaires pour qu'il existe un ensemble d'espèces chimiques $\mathcal{X}$ et un ensemble de réactions physiquement réalisables $\mathcal{R}$ pour que le système $(\mathcal{X}, \mathcal{R})$ réagisse selon la dynamique $y' = p(y)$.

De façon intéressante, les résultats précédents se généralisent aux systèmes positifs et aux systèmes de réactions biochimiques. L'idée pour cela est d'encoder chaque composante y_i par la différence entre deux composantes positives y_i^+ et y_i^- :

Théorème 3. *Tout GPAC peut être encodé dans un système positif chimiquement réalisable de dimension double. Si de plus le GPAC initial a une complexité de calcul polynomiale, alors le nouveau système aura également une complexité polynomiale.*

Démonstration. Soit $p \in \mathbb{R}^n[\mathbb{R}^n]$. Chaque $y_i \in \mathbb{R}$ peut être encodé par un couple $(y_i^+, y_i^-) \in \mathbb{R}_+^2$ tel qu'a tout instant, $y_i = y_i^+ - y_i^-$, et où les $y_i^\pm$ sont soumis à des dynamiques qui correspondent à un système chimique.

On pose $\hat{p_i}(y_1^+, y_1^-, \dots, y_n^+, y_n^-) = p_i[y = y^+ - y^-]$, puis on écrit $\hat{p}_i = \hat{p}_i^+ - \hat{p}_i^-$, où les monômes de $\hat{p}_i^+$ et $\hat{p}_i^-$ ont des coefficients positifs. On définit alors un nouveau système par :

$$\forall i \leq n, \begin{cases} {y_i^+}' &= \hat{p}_i^+ - f_i y_i^+ y_i^- \\ {y_i^-}' &= \hat{p}_i^- - f_i y_i^+ y_i^- \\ y_i^+(0) &= \max(0, y_i(0)) \\ y_i^-(0) &= \max(0, -y_i(0)) \end{cases}$$

où les f_i sont des polynômes à coefficients positifs tels que $f_i \geq \max(\hat{p}_i^+, \hat{p}_i^-)$.

Les termes supplémentaires $-f_i y_i^+ y_i^-$ sont implémentés par les réaction d'annihilation $y^+ + y^- \xrightarrow{f_i} \_$ et permettent d'obtenir un système où l'un des $y_i^\pm$ demeure toujours « petit ».

On remarque qu'on a : ${y_i^+}' \leq \hat{p}_i^+(1 - y_i^+ y_i^-)$ et ${y_i^-}' \leq \hat{p}_i^-(1 - y_i^+ y_i^-)$, de telle sorte que $(y_i^+ y_i^-)' \leq q \cdot (1 - y_i^+ y_i^-)$ où q est un polynôme à coefficients positifs. Puisqu'à $t = 0$, on a $y_i^+ y_i^- = 0$, on en déduit par une inégalité de Gronwall qu'on a toujours : $y_i^+ y_i^- \leq 1$. Par suite, $|y_i^\pm| \leq |y_i| + 1$, puis $|y^\pm| \leq |y| + n$. Par conséquent, si le système d'origine est majoré en espace par un polynôme en la taille de l'entrée et du temps, alors c'est encore le cas pour le système positif obtenu par la construction qui précède.

Chaque monôme de la forme $\lambda x_1^{\alpha_1} \dots x_m^{\alpha_m}, \lambda > 0$ apparaissant dans le terme de droite d'une égalité de la forme $y = p$ est implémenté par la réaction $\alpha_1 x_1 + \dots + \alpha_m x_m \xrightarrow{\lambda} y + \alpha_1 x_1 + \dots + \alpha_m x_m$. □

On peut également se restreindre à des réactions élémentaires, i.e. ayant au plus deux réactants, car tout PIVP est équivalent à un PIVP quadratique :

Théorème 4. *(Carothers* et al., *2005) Toute solution d'un PIVP est solution d'un PIVP de degré au plus deux.*

Démonstration. La preuve consiste à introduire des variables pour chaque monôme comme suit

$$v_{i_1,\ldots,i_n} = y_1^{i_1} y_2^{i_2}, \ldots, y_n^{i_n}$$

On a $y_1 = v_{1,0,\ldots,0}$ etc. La substitution de ces variables dans les équations différentielles des y_i' donnent des équations du premier degré en les variables $v_{i_1,\ldots,i_n}$. Les équations différentielles pour les variables qui ne sont pas des y_i sont de la forme

$$v'_{i_1,\ldots,i_n} = \sum_{k=0}^{n} i_k * v_{i_1,\ldots,i_k-1,\ldots,i_n} * y'_k$$

i.e. de degré deux puisque les y'_k sont des combinaisons linéaires des variables $v_{i_1,\ldots,i_n}$. □

Ces résultats montrent que le calcul biochimique élémentaire différentiel a toute la puissance d'expression des PIVP, et on déduit du théorème 1 la complétude de Turing de ce mode de calcul :

Théorème 5. *Les systèmes de réactions biochimiques élémentaires sur des univers finis de molécules sont Turing-complets dans la sémantique différentielle.*

Notons que ce n'est pas le cas dans les sémantiques discrètes des systèmes de réactions, où la complétude de Turing nécessite d'ajouter d'autres mécanismes comme la création dynamique non bornée de membranes (Busi *et al.,* 2006, Berry *et al.,* 1992) ou des réactions de polymérisation non bornée (Cardelli *et al.,* 2010).

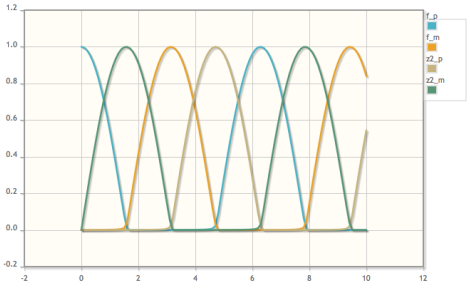
4.3. *Compilation biochimique des GPACs*

La preuve du théorème 3 montre comment implémenter biochimiquement les GPACs en doublant le nombre de variables pour les parties positives et négatives, et en implémentant chaque monôme des équations différentielles par une réaction catalytique de synthèse ou de dégradation suivant son signe. De même la preuve du théorème 4 montre comment se restreindre à des réactions élémentaires d'au plus deux réactants en augmentant le nombre d'espèces moléculaires, c'est-à-dire en sacrifiant la dimension du système à la minimisation des degrés.

Ce sont les principes de notre compilateur biochimique qui traduit une fonction mathématique définie par un PIVP en un système de réactions élémentaires. Par

exemple, l'oscillateur défini par la fonction du temps $f = cos(t)$ est compilé en six réactions élémentaires (ci-dessous les catalyseurs sont notés entre crochets, `a =[c]=> b` est une abbréviation pour `a+c => b+c`) :

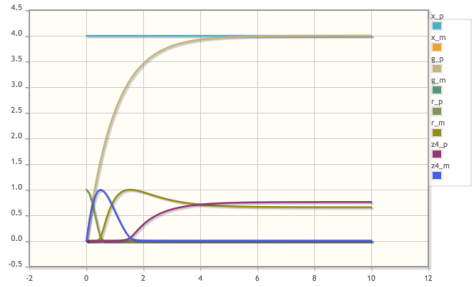
```
biocham: compile(cos,time,f).
_ =[z2_p]=> f_p.
_ =[z2_m]=> f_m.
_ =[f_m]=> z2_p.
_ =[f_p]=> z2_m.
fast*z2_m*z2_p for z2_m+z2_p=>_.
fast*f_m*f_p for f_m+f_p=>_.
present(f_p,1).

biocham: simulation(time:8).
```



La construction d'un GPAC qui *calcule* la valeur de $f(x)$ en tout point peut également se faire à partir d'un GPAC qui *génère* la fonction du temps $f(t)$, en se donnant la valeur de f en un point x_0 pour lequel $f(x)$ ne diverge pas en suivant la trajectoire $\gamma(t) = x + (x - x_0)e^{-\lambda t},\ \lambda > 0$ (Pouly, 2015). La compilation du GPAC qui calcule la fonction $cos(x)$ ajoute ainsi des réactions de calcul du résultat le long de la trajectoire vers l'argument, par exemple pour $cos(4)$:

```
biocham: compile(cos,x,r).
_ =[g_m]=> g_p.
_ =[x_p]=> g_p.
_ =[g_p]=> g_m.
_ =[x_m]=> g_m.
_ =[g_m+z4_p]=> r_p.
_ =[g_p+z4_m]=> r_p.
_ =[x_m+z4_m]=> r_p.
_ =[x_p+z4_p]=> r_p.
_ =[g_m+z4_m]=> r_m.
_ =[g_p+z4_p]=> r_m.
_ =[x_p+z4_m]=> r_m.
_ =[x_m+z4_p]=> r_m.
_ =[g_m+r_m]=> z4_p.
_ =[g_p+r_p]=> z4_p.
_ =[x_p+r_m]=> z4_p.
_ =[x_m+r_p]=> z4_p.
_ =[g_m+r_p]=> z4_m.
_ =[g_p+r_m]=> z4_m.
_ =[x_m+r_m]=> z4_m.
_ =[x_p+r_p]=> z4_m.
fast*z4_m*z4_p for z4_m+z4_p=>_.
fast*r_m*r_p for r_m+r_p=>_.
fast*g_m*g_p for g_m+g_p=>_.
fast*x_m*x_p for x_m+x_p=>_.
present(r_p,1).
```

```
biocham: present(x_p,4).
biocham: simulation(time:8).
```



4.4. *Convertisseur analogique-digital comparé à MAPK*

Nous avons vu que la structure en trois étages du circuit de signalisation MAPK pouvait s'expliquer par la fonction d'entrée-sortie qui en résulte en tout ou rien. Mais avec ce qui précède nous pouvons nous intéresser à compiler directement la fonction de Hill d'entrée-sortie. La fonction $y(t) = \frac{t^\alpha}{c^\alpha + t^\alpha}$ satisfait l'équation différentielle $y' = \frac{\alpha}{t} y(1-y)$. Cette fonction est donc facilement GPAC-générable et GPAC-calculable, par exemple par le système suivant :

$$\left\{\begin{array}{ll} \gamma \to \_ & y_2 + \alpha + x + y_1 \to \alpha + x + y_1 + 2y_2 \\ x \to x + \gamma & 2y_2 + \alpha + x + y_1 \to \alpha + x + y_1 + y_2 \\ 2y_1 + x \to y_1 + x & y_2 + \alpha + \gamma + y_1 \to \alpha + \gamma + y_1 \\ 2y_1 + \gamma \to 3y_1 + \gamma & 2y_2 + \alpha + \gamma + y_1 \to \alpha + \gamma + y_1 + 3y_2 \end{array}\right\}$$

avec les conditions initiales $(\gamma, y_1, y_2)_{t=0} = (1, 1, 1/2)$. Ce système vérifie $y_2 = \frac{x^\alpha}{1+x^\alpha}$ à l'état stationnaire, et constitue par conséquent un indicateur binaire de présence : si $x \gg 1$, alors $y_2 = 1$, et si $x \ll 1$, alors $y_2 = 0$, la discrimination étant d'autant plus forte que la valeur de α est importante. Notons que cette valeur est donnée ici par une concentration fixe de molécule mais pourrait être représentée plus simplement par une constante cinétique.

Ce convertisseur a toutefois le défaut de créer une valeur intermédiaire en $\frac{1}{\gamma}$ qui donne une amplitude exponentielle pour $x = 0$, et donc une complexité de calcul exponentielle au sens de la section précédente. Si l'on se restreint à prendre x dans un intervalle de la forme $[\varepsilon, +\infty[$, avec $\varepsilon > 0$, alors la complexité devient polynomiale. Par ailleurs, si on se restreint à l'ordre 4, notre compilateur avec la commande `compile(id^4/(1+id^4),x,o)` produit un système de 54 réactions, de complexité algorithmique polynomiale, avec toutefois une composante divergente de complexité de calcul non linéaire.

Le circuit naturel MAPK de 30 réactions apparaît donc à la fois plus concis et avec une moindre complexité de calcul (absence de divergence) que le système de réactions actuellement produit suivant nos principes génériques de compilation.

5. Compilation biochimique de la séquentialité et cycle cellulaire

De la même manière, on peut implémenter un indicateur d'absence binaire en implémentant le terme $\frac{c^\alpha}{c^\alpha + x^\alpha}$. Cela permet d'obtenir une implémentation chimique bien meilleure que celles proposées dans (Senum *et al.,* 2011) ou même (Huang *et al.,* 2012), pour lesquels des phénomènes de fuite peuvent se produire : même en l'absence relative de l'espèce x, l'indicateur de présence demeure en concentration suffisamment importante pour catalyser certaines réactions, ou bien l'effet inverse, l'indicateur d'absence peut être en quantité trop faible. Ceci est particulièrement visible dans l'implémentation de la séquentialité : étant donné des réactions R_i, si on veut que R_2 ne s'exécute qu'une fois R_1 terminée, on impose un indicateur d'absence d'une espèce consommée par R_1 comme catalyseur de R_2, idem entre R_2 et R_3, *etc.* On

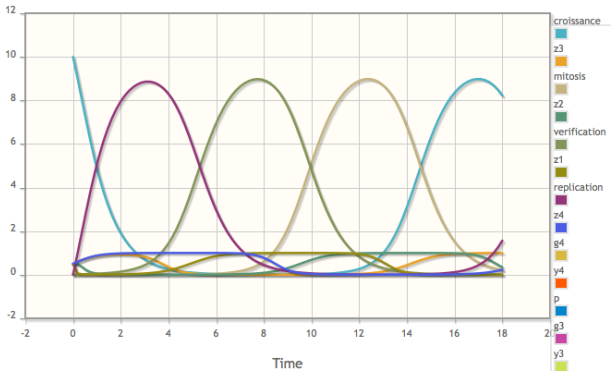
peut alors voir apparaître le phénomène suivant : les réactions se font d'autant plus lentement que i est grand, autrement dit, les réactions accumulent du retard dans leur exécution du fait de la rétention des indicateurs d'absence.

Fort d'un indicateur d'absence suffisamment puissant, il est possible d'implémenter la séquentialité, l'instruction conditionnelle, et les structures de boucles de la programmation algorithmique. (Huang *et al.,* 2012) montre ainsi comment compiler des petits programmes impératifs en un système de réactions biochimiques dans lequel les espèces moléculaires sont utilisées comme des marqueurs de la position du programme dans un graphe de flot de contrôle.

Par exemple, une spécification minimaliste du cycle de division cellulaire peut être donnée par le programme

```
while true do {croissance; réplication; vérification; mitose;}
```

La compilation de ce programme en réactions élémentaires implémente la séquentialité des quatre phases du cycle par la dégradation des marqueurs de chacune des phases, de façon similaire aux protéines cyclines dans les modèles du cycle cellulaire (Gérard *et al.,* 2009) :



6. Discussion

La vision bioinformatique qui consiste à voir les cellules comme des machines, et les réactions comme des programmes, est riche de concepts et d'outils formels pour appréhender la complexité des réseaux d'interactions moléculaires, spécifier formellement les comportements biologiques observés ou souhaités, élucider les intéractions importantes et leur rôle dans les circuits naturels, synthétiser par compilation des programmes biochimiques artificiels, et les comparer aux circuits naturels, non seulement en terme de complexité de structure (Barabási, 2016), mais aussi, et cela est nouveau, en terme de complexité de calcul.

Nous avons montré que la difficulté liée à la nature analogique des calculs protéiques, par opposition aux calculs digitaux et à l'abstraction discrète qui sied mieux à l'activité des gènes, pouvait se lever sur le plan fondamental, par les résultats d'équiva-

lence entre les notions de calculabilité et complexité des modes de calcul analogiques et digitaux. Dans cette direction, la caractérisation analogique des classes de faible complexité algorithmique semble cruciale pour analyser et contraindre les différentes façons d'implémenter une fonction.

Nous commençons donc à comprendre les systèmes de réactions biochimiques comme des programmes, mais pas encore par exemple pourquoi les circuits naturels ont cette structure et pas d'autres, quelle a été leur évolution et quelles sont leurs possibilités d'évolution. Le langage de spécification des calculs analogiques en entrée de notre compilateur biochimique peut y aider mais est encore incertain. Nous savons inférer un système de réactions équivalent à un système d'équations différentielles, compiler un GPAC en réactions biochimiques, ou un système d'entrée-sortie linéaire défini par sa fonction de transfert (Chiu *et al.,* 2015, Oishi *et al.,* 2011), et combiner ces calculs analogiques à des conditions logiques et aux structures de contrôle de la programmation algorithmique classique, mais ces différents langages de spécification ne sont pas unifiés. Même s'il est maintenant possible d'implanter dans des cellules vivantes (Nielsen *et al.,* 2016, Courbet *et al.,* 2015), ou de façon plus sure dans des vésicules artificielles non-vivantes (Courbet *et al.,* 2017), des programmes biochimiques synthétiques limités à quelques réactions satisfaisant une spécification logique, le passage à l'échelle de programmes moins triviaux demande des progrès de nature fondamentale sur les systèmes de réactions et leurs complexités.

Dans cette entreprise de décryptage et reconstruction du logiciel du vivant, la prochaine grande étape devrait être pour nous l'étude algorithmique systématique des circuits naturels et de leurs évolutions passées ou possibles, suivant les méthodes de spécification mathématique de leurs fonctions esquissées ici, et exploitables dans une théorie computationnelle de l'évolvabilité (Valiant, 2013).

7. Références

Barabási A.-L., *Network Science*, Cambridge University Press, 2016.

Berry G., Boudol G., « The chemical abstract machine », *Theoretical Computer Science*, 1992.

Bournez O., Campagnolo M. L., Graça D. S., Hainry E., « The general purpose analog computer and computable analysis are two equivalent paradigms of analog computation », *International Conference on Theory and Applications of Models of Computation*, Springer, p. 631-643, 2006.

Busi N., Gorrieri R., « On the Computational Power of Brane Calculi », *in* , G. Plotkin (ed.), *Transactions on Computational Systems Biology VI*, vol. 4220 of *Lecture Notes in BioInformatics*, Springer-Verlag, p. 16-43, November, 2006. CMSB'05 Special Issue.

Cardelli L., Zavattaro L., « Turing Universality of the Biochemical Ground Form », *Mathematical Structures in Computer Science*, vol. 20, n° 1, p. 45-73, 2010.

Carothers D. C., Parker G. E., Sochacki J. S., Warne P. G., « Some Properties of Solutions to Polynomial Systems of Differential Equations », *Electronic Journal of Differential Equations*, 2005.

Chabrier-Rivier N., Chiaverini M., Danos V., Fages F., Schächter V., « Modeling and querying biochemical interaction networks », *Theoretical Computer Science*, vol. 325, n° 1, p. 25-44, September, 2004.

Chiu T.-Y., Chiang H.-J. K., Huang R.-Y., Jiang J.-H. R., Fages F., « Synthesizing Configurable Biochemical Implementation of Linear Systems from Their Transfer Function Specifications », *PLoS ONE*, 2015.

Courbet A., Amar P., Fages F., Renard E., Molina F., « Computer aided design of programmable synthetic protocells performing multiplexed logic-gated microscale diagnostics », *In preparation*, 2017.

Courbet A., Endy D., Renard E., Molina F., Bonnet J., « Detection of pathological biomarkers in Human clinical samples via amplifying genetic switches and logic gates », *Science Translational Medicine*, 2015.

Cousot P., Cousot R., « Abstract Interpretation : A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints », *POPL'77 : Proceedings of the 6th ACM Symposium on Principles of Programming Languages*, ACM Press, New York, p. 238-252, 1977. Los Angeles.

Fages F., Gay S., Soliman S., « Inferring Reaction Systems from Ordinary Differential Equations », *Theoretical Computer Science*, vol. 599, p. 64-78, September, 2015.

Fages F., Soliman S., « Abstract Interpretation and Types for Systems Biology », *Theoretical Computer Science*, vol. 403, n° 1, p. 52-70, 2008a.

Fages F., Soliman S., « From reaction models to influence graphs and back : a theorem », *Proceedings of Formal Methods in Systems Biology FMSB'08*, number 5054 in *Lecture Notes in Computer Science*, Springer-Verlag, February, 2008b.

Fages F., Traynard P., « Temporal Logic Modeling of Dynamical Behaviors : First-Order Patterns and Solvers », *in* , L. F. del Cerro, , K. Inoue (eds), *Logical Modeling of Biological Systems*, John Wiley & Sons, Inc., chapter 8, p. 291-323, 2014.

Gérard C., Goldbeter A., « Temporal self-organization of the cyclin/Cdk network driving the mammalian cell cycle », *Proceedings of the National Academy of Sciences*, vol. 106, n° 51, p. 21643-21648, December, 2009.

Gillespie D. T., « Exact Stochastic Simulation of Coupled Chemical Reactions », *Journal of Physical Chemistry*, vol. 81, n° 25, p. 2340-2361, 1977.

Graça D., Costa J., « Analog computers and recursive functions over the reals », *Journal of Complexity*, vol. 19, n° 5, p. 644-664, 2003.

Huang C.-Y., Ferrell J. E., « Ultrasensitivity in the mitogen-activated protein kinase cascade », *Proceedings of the National Academy of Sciences*, vol. 93, n° 19, p. 10078-10083, 1996.

Huang D.-A., Jiang J.-H., Huang R.-Y., Cheng C.-Y., « Compiling Program Control Flows into Biochemical Reactions », *ICCAD'12 : IEEE/ACM International Conference on Computer-Aided Design*, ACM, San Jose, USA, p. 361-368, November, 2012.

Nielsen A. A. K., Der B. S., Shin J., Vaidyanathan P., Paralanov V., Strychalski E. A., Ross D., Densmore D., Voigt C. A., « Genetic circuit design automation », *Science*, 2016.

Oishi K., Klavins E., « Biomolecular implementation of linear I/O systems », *IET SYstems Biology*, vol. 5, n° 4, p. 252-260, 2011.

Pouly A., Continuous models of computation : from computability to complexity, PhD thesis, Ecole Polytechnique, July, 2015.

Qiao L., Nachbar R. B., Kevrekidis I. G., Shvartsman S. Y., « Bistability and Oscillations in the Huang-Ferrell Model of MAPK Signaling », *PLoS Computational Biology*, vol. 3, n° 9, p. 1819-1826, September, 2007.

Rizk A., Batt G., Fages F., Soliman S., « Continuous Valuations of Temporal Logic Specifications with applications to Parameter Optimization and Robustness Measures », *Theoretical Computer Science*, vol. 412, n° 26, p. 2827-2839, 2011.

Segel L. A., *Modeling dynamic phenomena in molecular and cellular biology*, Cambridge University Press, 1984.

Senum P., Riedel M., « Rate-Independent Constructs for Chemical Computation », *PLOS One*, 2011.

Shannon C., « Mathematical theory of the differential analyser », *Journal of Mathematics and Physics*, vol. 20, p. 337-354, 1941.

Snoussi E. H., « Necessary conditions for multistationarity and stable periodicity », *Journal of Biological Systems*, vol. 6, p. 3-9, 1998.

Thomas R., « On the relation between the logical structure of systems and their ability to generate multiple steady states or sustained oscillations », *Springer Ser. Synergetics*, vol. 9, p. 180-193, 1981.

Valiant L., *Probably Approximatively Correct*, Basic Books, 2013.