Using hybrid concurrent constraint programming to model dynamic biological systems

Alexander Bockmayr, Arnaud Courtois

Using hybrid concurrent constraint programming to model dynamic biological systems

Alexander Bockmayr[1] and Arnaud Courtois[1]

Université Henri Poincaré, LORIA
B.P. 239, F-54506 Vandœuvre-lès-Nancy, France
{bockmayr|acourtoi}@loria.fr

Abstract. Systems biology is a new area in biology that aims at achieving a systems-level understanding of biological systems. While current genome projects provide a huge amount of data on genes or proteins, lots of research is still necessary to understand how the different parts of a biological system interact in order to perform complex biological functions. Computational models that help to analyze, explain or predict the behavior of biological systems play a crucial role in systems biology. The goal of this paper is to show that hybrid concurrent constraint programming [11] may be a promising alternative to existing modeling approaches in systems biology. Hybrid cc is a declarative compositional programming language with a well-defined semantics. It allows one to model and simulate the dynamics of hybrid systems, which exhibit both discrete and continuous change. We show that Hybrid cc can be used naturally to model a variety of biological phenomena, such as reaching thresholds, kinetics, gene interaction or biological pathways.

1 Introduction

The last decades have seen a tremendous progress in molecular biology, the most spectacular result being the announcement of a first draft of the entire human genome sequence in June 2000, with analyses published in February 2001. Current genome, transcriptome or proteome projects, whose goal is to determine completely all the genes, RNA or proteins in a given organism, produce an exponentially growing amount of data. Storing, maintaining, and accessing these data represents already a challenge to computer science. But the real work - with an enormous impact on medicine and pharmacy - consists in exploiting all these data and in understanding how the various components of a biological system (i.e. genes, RNA, proteins etc.) interact in order to perform complex biological functions.

Systems biology is a new area in biology, which aims at a system-level understanding of biological systems [16]. While traditional biology examines single genes or proteins in isolation, system biology simultaneously studies the complex interaction of many levels of biological information - genomic DNA, mRNA, proteins, informational pathways and networks - to understand how they work together, see [14] for a recent example.

The development of computational models of biological systems plays a crucial role in systems biology [3]. A number of projects like BioSpice, Cellerator, DBSolve, E-Cell, Gepasi, Jarnac, ProMot/DIVA, StochSim and Virtual Cell aim at modeling and simulating biological processes. The Systems Biology Workbench [13] is a software platform currently being developed in order to enable the different tools to interact with each other. From a programming language perspective, a fundamental question arises: what is the semantics underlying these different approaches and the possible combinations between them?

The goal of this paper is to present hybrid concurrent constraint programming [11] as a promising alternative to existing modeling and simulation approaches in systems biology. `Hybrid cc` is a very powerful framework for modeling, analyzing and simulating hybrid systems, i.e., systems that exhibit both discrete and continuous change. It is a declarative compositional programming language based on the cc paradigm. From a computer science perspective, a major advantage of `Hybrid cc` compared to other approaches is that it is a full programming language with a well-defined semantics, based on a small number of primitives. From the viewpoint of systems biology, these basic constructs may help to identify key computational concepts needed to represent and to understand biological systems at the molecular and cellular level.

The organization of this paper is as follows. We start in Sect. 2 by giving a short overview of modeling approaches for molecular and cell biology. We emphasize the role of hybrid systems, which can cover both discrete and continuous phenomena. Sect. 3 recalls some of the basic ideas underlying hybrid concurrent constraint programming and gives a short introduction into the language `Hybrid cc`. Sect. 4 is the core of the paper, explaining how `Hybrid cc` can be used to model biological systems in a high-level and declarative way. We show that various phenomena in biology, like thresholds, kinetics, gene interactions, or biological pathways. have their natural counterpart in `Hybrid cc`. Sect. 5 summarizes the discussion and points out directions for further research. The results presented in this paper were first announced in [2], see also [5].

2 Existing modeling approaches

A variety of formalisms has been proposed in the literature for modeling biological systems [3, 6]. Following [9], we may distinguish three basic approaches

- discrete,
- continuous,
- stochastic,

and various combinations between them.

Discrete models are based on discrete variables and discrete state changes. A classical example are Boolean networks for gene regulation [15]. For each gene, there is a Boolean variable indicating whether or not the gene is expressed in a given state. Boolean functions are then used to relate the variables belonging to different states. Qualitative networks [23] are an extension of Boolean networks,

based on multivalued logic. Each variable now has a finite domain of possible values, which can be used, for example, to represent different levels of gene expression. Several authors have started to apply concepts and tools from formal verification to model biological systems, like Petri nets [18] or the π-calculus [19].

Continuous models have been used in mathematical biology for a very long time [25]. They are based on differential equations that typically model biochemical reactions. In principle, any system of chemical reactions and physical constraints can be transformed into a system of nonlinear ordinary differential equations, whose variables are concentrations of proteins, RNA or other molecules.

In many models of biological systems, there exist discontinuous transitions. For example, we may want to model that expression of gene x activates expression of gene y; above a certain threshold, gene y inhibits expression of gene x. This leads to a system of conditional differential equations like

$$\begin{aligned} \text{if } (y < 0.8) \text{ then } x' &= -0.02 * x + 0.01 \\ \text{if } (y \geq 0.8) \text{ then } x' &= -0.02 * x \\ y' &= \ \ 0.01 * x \end{aligned} \quad ,$$

see Fig. 1 for an illustration.

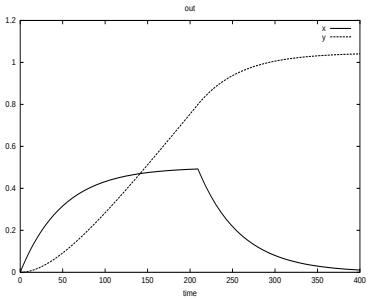


Fig. 1. Interaction between two genes

The need to capture both discrete and continuous phenomena motivates the study of hybrid dynamical systems [24]. Their relevance for biology has been pointed out, among others, in [17] and [1], where special emphasis is put on the similarity between hybrid systems encountered in engineering and genetic circuits or biomolecular networks.

Another important issue for biology are stochastic phenomena. Probabilities occur in many different ways. We may distinguish between discrete and continuous distributions. Probabilities may appear explicitly in random variables or random numbers or implicitly like in kinetic laws or models like the one presented in Sect. 4.2. Typical situations where probabilities are used in modeling include:

- Simplified representation of processes (e.g. simulating the pathway choice of λ-phages in a population [8]).
- Integration of stochastic noise in order to get more realistic models.
- Large-scale Monte Carlo simulations.

3 Hybrid concurrent constraint programming

Constraint programming started inside logic programming around 1985. In a constraint program, the user specifies a number of constraints. Each constraint defines a relation between variables that describe the state of the system under investigation. The constraint solver provides algorithms which compute solutions, i.e., valuations of the variables satisfying all the constraints, or which infer new constraints from the given ones.

In concurrent constraint programming (cc), different computation processes may run concurrently. Interaction is possible via the *constraint store*. The store contains all the constraints currently known about the system. A process may *tell* the store a new constraint, or *ask* the store whether some constraint is entailed by the information currently available, in which case further action is taken [20].

The original cc framework has been extended in various directions. In the context of this paper, we are interested in the following extensions:

- `Timed cc` [21]
- `Timed default cc` [22]
- `Hybrid cc` [12, 11, 4]

One major difficulty in the original cc framework is that cc programs can detect only the presence of information, not its absence. To overcome this problem, [21] proposed to add to the cc paradigm a sequence of phases of execution. At each phase, a cc program is executed. At the end, absence of information is detected, and used in the next phase. This results in a synchronous reactive programming language, `Timed cc`. But, the question remained how to detect negative information instantaneously. `Default cc` extends cc by a negative ask combinator `if` a `else` A, which imposes the constraints of A unless the rest of the system imposes the constraint a. Logically, this can be seen as a default. Introducing phases as in `Timed cc` leads to `Default Timed cc` [22]. Only one additional construct is needed: `hence` A, which starts a copy of A in each phase after the current one.

`Hybrid cc` is an extension of `Default cc` over continuous time. First continuous constraint systems are allowed, i.e., constraints may involve differential equations that express initial value problems. Second, the `hence` operator is interpreted over continuous time. It imposes the constraints of A at every real time instant after the current one. Tab. 1 summarizes the basic combinators of `Hybrid cc`.

The evolution of a system in `Hybrid cc` is piecewise continuous, with a sequence of alternating point and interval phases. All discrete changes take place in a point phase, where a simple `Default cc` program is executed. In a continuous

Agents	Propositions
c	c holds now
`if` c `then` A	if c holds now, then A holds now
`if` c `else` A	if c will not hold now, then A holds now
`new` X `in` A	there is an instance $A[T/X]$ that holds now
A, B	both A and B hold now
`hence` A	A holds at every instant after now
`always` A	same as (A, `hence` A)
`when(`c`)` A	same as (`if` c `then` A, `hence (if` c `then` A`)`)
`unless(`c`)` A `else` b	same as (`if` c `then` B, `if` c `else` A)

Table 1. Combinators of `Hybrid cc`

phase, computation proceeds only through the evolution of time. The interval phase, whose duration is determined in the previous point phase, is exited as soon as the status of a conditional changes, see [12] for additional details. The current implementation of `Hybrid cc` supports two types of constraints, which are handled by interval methods:

- ordinary differential equations, and
- (nonlinear) algebraic constraints.

Algebraic constraints are solved using a combination of interval propagation, splitting, Newton-Raphson method, and the Simplex algorithm. The ordinary differential equations are integrated using a 5th-order Runge-Kutta method with adaptive step size, modified for interval variables [4].

4 Modeling biological systems

4.1 Key features of biological systems and their counterpart in Hybrid cc

The goal of this section is to show that `Hybrid cc` is well-suited for modeling and simulating dynamic biological systems. The next table gives an overview of a number of important features of biological systems and their counterpart in `Hybrid cc`.

Biology	`Hybrid cc`
reaching thresholds	discrete events
time, concentration	continuous variables
kinetics	differential equations
gene interaction	concurrency
stochastic behavior	random numbers

Reaching thresholds. Thresholds can be used to determine when a process should be started or stopped. They arise in different situations. On the one hand, they may be introduced in the context of qualitative reasoning in order to discretize continuous phenomena. On the other hand, they allow one to refine a model by specifying different dynamics for different states. A typical example for this is the multimode continuous model developed in Sect. 4.3. The kind of reactivity that can be expressed in cc languages depends on the class of the language: `Timed cc` allows for synchronous, `Default cc` for instantaneous, and `Hybrid cc` for asynchronous reactions.

Time, concentrations. Time and concentrations of molecules are continuous variables in a hybrid system. If concentrations reach a certain threshold, this triggers a change of the system state.

Kinetics. The kinetics of a biological system is represented by biochemical laws (differential equations) depending on continuous variables such as concentrations and real time.

Interactions. Interactions between different agents may be direct (e.g. molecule/gene) or indirect (e.g. gene/gene, molecule/effector/target). They reflect a concurrency at the biological level, which can be expressed naturally in a concurrent programming language.

Stochastic aspects. Probabilistic choice between different pathways or stochastic noise are common techniques used in modeling biological systems [3]. Probabilities may be realized by discrete probability distributions or in a hybrid way (e.g. the Langevin approach, where differential equations are extended by a noise term that induces stochastic fluctuations of the state about its deterministic value).

In the following sections, we present a number of models of dynamic biological systems that we have developed in `Hybrid cc`. The goal is to show that `Hybrid cc` allows one to model biological systems in a very natural and declarative way. Often it is possible to represent biological phenomena directly by corresponding statements in `Hybrid cc`.

4.2 Stochastic behavior of protein-DNA complexation

Our first model represents the - unstable - binding mechanism between 2 kinds of proteins and a single DNA strand [9]. The system is composed of m M-proteins and n N-proteins. There are 4 possible states and 8 possible reactions, see Fig. 2.

Each reaction is characterized by a stoichiometric coefficient:

$$N + DNA \underset{k_{10}}{\overset{k_{01}}{\rightleftharpoons}} N \cdot DNA \qquad N + M \cdot DNA \underset{k_{32}}{\overset{k_{23}}{\rightleftharpoons}} N \cdot M \cdot DNA$$

$$M + DNA \underset{k_{20}}{\overset{k_{02}}{\rightleftharpoons}} M \cdot DNA \qquad M + N \cdot DNA \underset{k_{31}}{\overset{k_{13}}{\rightleftharpoons}} M \cdot N \cdot DNA$$

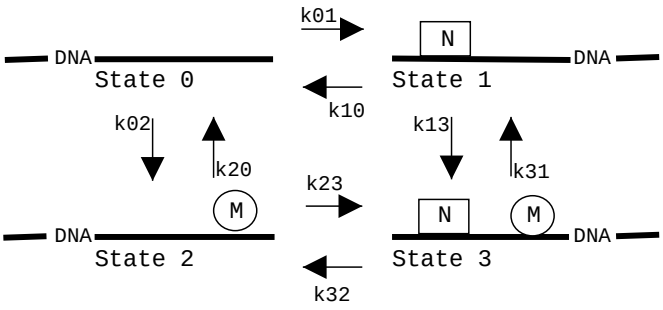


Fig. 2. Possible states and transitions

We first a consider a discrete time model. The probabilities $P_i(t+\Delta t)$ of being in state i at time $t+\Delta$ depends on the probabilities at time t. They are computed in the following way:

$$\begin{pmatrix} P_0(t+\Delta t) \\ P_1(t+\Delta t) \\ P_2(t+\Delta t) \\ P_3(t+\Delta t) \end{pmatrix} = A \cdot \begin{pmatrix} P_0(t) \\ P_1(t) \\ P_2(t) \\ P_3(t) \end{pmatrix}$$

with the transition matrix

$$A = \begin{bmatrix} 1-nk_{01}\Delta t - mk_{02}\Delta t & k_{10}\Delta t & k_{20}\Delta t & 0 \\ nk_{01}\Delta t & 1-k_{10}\Delta t - mk_{13}\Delta t & 0 & k_{31}\Delta t \\ mk_{02}\Delta t & 0 & 1-k_{20}\Delta t - nk_{23}\Delta t & k_{32}\Delta t \\ 0 & mk_{13}\Delta t & nk_{23}\Delta t & 1-k_{31}\Delta t - k_{32}\Delta t \end{bmatrix}.$$

Note that there is no direct transition between the states 0 and 3 resp. 1 and 4. This discrete stochastic process can be expressed directly in `Hybrid cc`. We use a clock for synchronization. The constant `dt` represents the time interval Δt, `prev(x)` yields the previous value of x. The constraints $P_i' = 0$ produce step functions for P_i, see Fig. 3.

```
/* Define constants  */
#define dt  1         // interval length (delta t)
#define k01 0.02      //
...
/* Define variables */
interval clock; interval P0; ... ; interval P3;
/* Initialization (no binding) */
clock=dt; clock'=-0.5;
P0=1; P1=0; P2=0; P3=0;
/* Clock update */
always { if (clock=0) { clock'=.5;}
         else {clock''=-1/dt;}
}
/* Compute probabilities */
```

```
always { if (clock=0) {
P0=((1-(n*k01*dt)-(m*k02*dt))*prev(P0))+(k10*dt*prev(P1))+(k20*dt*prev(P2));
P1=(n*k01*dt*prev(P0))+((1-(k10*dt)-(m*dt*k13))*prev(P1))+(k31*dt*prev(P3));
P2=(m*k02*dt*prev(P0))+((1-(k20*dt)-(n*dt*k23))*prev(P2))+(k32*dt*prev(P3));
P3=(m*k13*dt*prev(P1))+(n*k23*dt*prev(P2))+((1-(k31*dt)-(k32*dt))*prev(P3));
}
   else { P0'=0; P1'=0; P2'=0; P3'=0; }
}
/* Output */
sample(P0,P1,P2,P3);
```

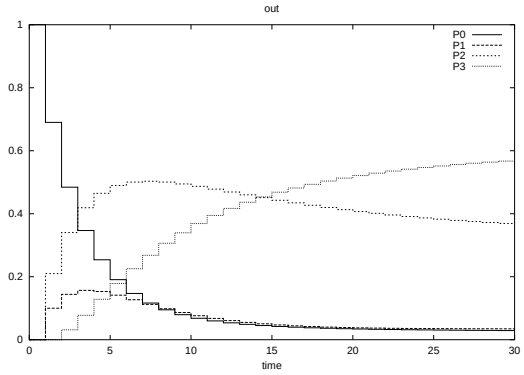


Fig. 3. Probabilities $P_0, \ldots, P_3$

The basic model can be extended in various ways. For example, we may assume that state 1 resp. 2 increase the production rate for some other proteins A and B. This may be modeled simply by adding constraints of the form

```
always {
    unless(P1>0.1)  {A'=0.0001;}  else {A'=0.03;};
    unless(P2>0.45) {B'=0.00015;} else {B'=0.01;};
}
```

resulting in the dynamics illustrated by Fig. 4.

Another possibility is to consider $\Delta t \to 0$. This leads to a continuous model similar to the one developed in the next section.

4.3 A multimode continuous model

The kinetics of a chemical reaction depends on the concentration of the reactants. At very low and very high concentrations, the normal differential equations are no longer accurate. Therefore, in the following example, the system of differential equations is modified as soon as the concentration passes certain threshold

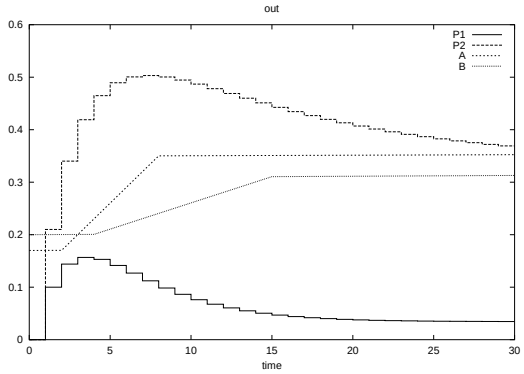


Fig. 4. Production of A and B depending on the probabilities P_i

values. This type of hybrid model can be seen as a refinement of a discrete qualitative network.

We model in `Hybrid cc` a phenomenon of bioluminescence for the bacteria V. fischeri [1]. These marine bacteria exist at low and high densities. While at low density the bacteria appear to be non-luminescent, a dramatic increase in luminescence can be observed when the density passes a given threshold. This phenomenon depends on the concentration of a certain small molecule A_i able to diffuse in and out of the cell membrane.

To describe the concentration of a molecular species x (RNA, protein, protein complex, or small molecule), we use the generic equation

$$\frac{dx}{dt} = vs - v_d \pm v_r \pm v_t.$$

Here v_s is the synthesis rate, v_d the degradation rate, v_r the reaction rate w.r.t. other molecules, and v_t the transportation rate w.r.t. the environment (diffusion etc.).

We first present a mathematical model for one bacteria in the population [1]. The differential equations depend on the concentration of the molecule A_i. We use different equations depending on whether the concentration of A_i is low, medium, or high. Let x_7 resp. x_9 denote the internal and external concentration of A_i. The concentrations of the other molecules involved in the process are described by the variables $(x_1, x_2, x_3, x_4, x_5, x_6, x_8)$. All the remaining symbols are constants.

$$
\begin{aligned}
\frac{dx_1}{dt} &= \begin{cases} \eta_1(\frac{c}{2} - x_1), & \text{if } 0 \le x_7 \le A_i^- \\ \frac{\eta_1}{4}(3c + \frac{x_8^{\nu_{81}}}{\kappa_{81}^{\nu_{81}} + x_8^{\nu_{81}}} - 4x_1), & \text{if } A_i^- < x_7 \le A_i^+ \\ -\eta_1 \cdot x_1, & \text{if } A_i^+ < x_7 \end{cases} \\
\frac{dx_2}{dt} &= \begin{cases} -\eta_2 \cdot x_2, & \text{if } 0 \le x_7 \le A_i^- \\ \eta_2(3c + \frac{x_8^{\nu_{82}}}{\kappa_{82}^{\nu_{82}} + x_8^{\nu_{82}}} - x_2), & \text{if } A_i^- < x_7 \end{cases} \\
&\vdots \\
\frac{dx_7}{dt} &= -\eta_7 \cdot x_7 + r_4 \cdot x_4 - r_{mem}(x_7 - x_9) - r_{37,R} \cdot x_3 \cdot x_7 \\
\frac{dx_9}{dt} &= -\eta_7 \cdot x_9 + r_4 \cdot x_4 - r_{mem}(x_7 - x_9) + u
\end{aligned}
$$

Again, the mathematical description can be directly translated into `Hybrid cc`. Each molecular species is represented by an independent agent, whose dynamics is described by a single - possibly conditional - differential equation. The interaction between the different agents is controlled by `Hybrid cc`.

```
#include "storeval.hcc"
/* Definition of constants */
   #define Ai_min 0.1 ... #define rmem 0.02
/* Definition of variables */
   interval x1; ... ; interval lum; interval pert;
/* Initialization */
   x1=0; ... ; lum=0; storeval(pert,0);
/* Constraints and combinators */
   always { if (x7<Ai_min) x1'=mu1*((0.5*c)-x1);
     if ((x7>=Ai_min)&&(x7<Ai_plus))
        x1'=(mu1/4)*((3*c)+((x8^ksi81)/((k81^ksi81)+(x8^ksi81)))-(4*x1));
     if (x7>=Ai_plus) x1'=-mu1*x1;}
   always { if (x7<Ai_min) x2'=-mu2*x2;
     if (x7>=Ai_min) x2'=mu2*(((x8^ksi82)/((k82^ksi82)+(x8^ksi82)))-x2);}
   ...
   always { x9'=(-mu7*x9)+(rmem*(x7-x9))+pert;}
/* Perturbation */
   when(time=200) storeval(pert,0.9);
   when(time=400) storeval(pert,0);
/* Luminescence  */
   always { lum'=x5'+x6';}
/* Output */
   sample(pert,x1,x2,x3,x4,x5,x6,x7,x8,x9,lum);
```

We assume that luminescence is proportional to x_5 and x_6. In order to simulate a population growth, we perform an artificial perturbation of the external concentration x_9 in the time interval $[200, 400]$. This is done by the function `storeval` that fixes a variable to a given value until the next change occurs.

Fig. 5 shows the evolution of x_1 depending on x_7, as well as the change in luminescence caused by the virtual population growth in the interval $[200, 400]$. Note the change in the dynamics of x_1 when x_7 reaches the lower or upper threshold.

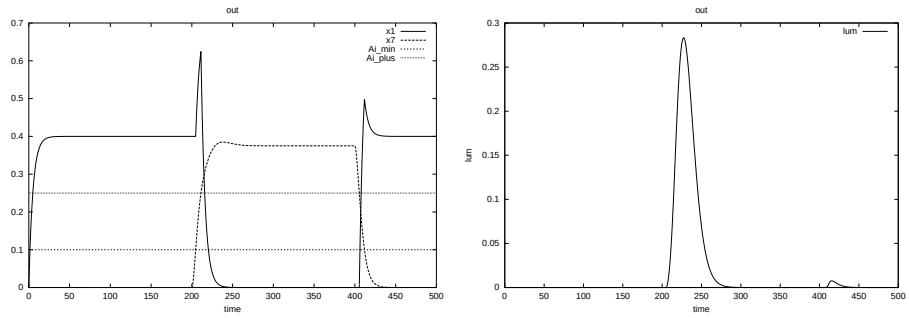


Fig. 5. Simulation of luminescence in V. Fischeri

4.4 A typical example of hybrid concurrent constraint programming

The next example combines in a single model all the features described earlier in Sec. 4.1. We consider cell differentiation for a population of epidermic *X. laevis* cells [7]. Each cell interacts with its neighbors, i.e., there are concurrent agents. The concentrations vary continuously. They are described by differential equations. Discrete transitions occur whenever certain threshold values are reached. Finally, the initialization is different for each cell (stochastic noise).

The cells are arranged in a hexagonal lattice (see Fig. 6a), so that a cell may have $2, 3, 4$ or 6 neighbors. In each cell, we consider the concentrations v_D, v_N of two proteins Delta and Notch. The Notch production capacity u_N in a cell depends on the Delta concentration of its neighbors (u_N is high if the ${v_D}^i$ are high):

$$u_N = \sum_{i=1}^{k} {v_D}^i, \text{ with } k \in \{2, 3, 4, 6\}.$$

The Delta production capacity u_D depends on the Notch concentration v_N in the same cell (u_D is high when v_N is low):

$$u_D = -v_N.$$

Depending on threshold values h_D, h_N, each cell can be in four different states:

State 1:	Delta and Notch inhibited:	$u_D < h_D$ and $u_N < h_N$
State 2:	Delta expressed, Notch inhibited:	$u_D \geq h_D$ and $u_N < h_N$
State 3:	Delta inhibited, Notch expressed:	$u_D < h_D$ and $u_N \geq h_N$
State 4:	Delta and Notch expressed:	$u_D \geq h_D$ and $u_N \geq h_N$

The production of Delta and Notch proteins depends on the state of the cell:

$$
\begin{array}{lll}
\text{State 1:} & v'_D = & -\lambda_D \cdot v_D, \qquad v'_N = -\lambda_N \cdot v_N \\
\text{State 2:} & v'_D = R_D & -\lambda_D \cdot v_D, \qquad v'_N = -\lambda_N \cdot v_N \\
\text{State 3:} & v'_D = & -\lambda_D \cdot v_D, \qquad v'_N = R_N -\lambda_N \cdot v_N \\
\text{State 4:} & v'_D = R_D & -\lambda_D \cdot v_D, \qquad v'_N = R_N -\lambda_N \cdot v_N
\end{array}
$$

Here λ_D, λ_N are degradation rate constants, R_D, R_N are production rate constants.

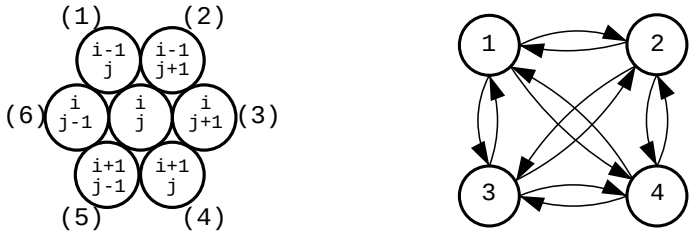


Fig. 6. a) Hexagonal lattice b) States and transitions

Given the dimensions of the hexagonal lattice, we generate automatically the following `Hybrid cc` program. To initialize the variables, we use a random number generator.

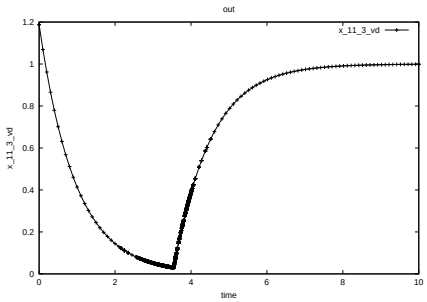
```
#include "parameters.hcc"
%module  "HCC_lib_math"
/* Declaration of variables */
   interval ud_1_1, un_1_1, vd_1_1, vn_1_1, state_1_1, ud_2_1, ...
/* Auxiliary functions */
   interval coeff_rd(interval ud) {
       if (ud<hd) return=0;
       else return=1; }
   interval coeff_rn(interval un) {
       if (un<hn) return=0;
       else return=1; }
/* Initialization (with stochastic noise) */
   vd_1_1 = HCC_noise(1,30) ; vn_1_1 = HCC_noise(1,30) ; ...
/* Constraints and Combinators */
   always {
       un_1_1  =  vd_1_2  + vd_2_1; un_1_2  = ...
       ud_1_1  = -vn_1_1;           ud_1_2  = ...

       vd_1_1' = (coeff_rd(ud_1_1)*rd) - lambda_d * vd_1_1; vd_1_2' = ...
       vn_1_1' = (coeff_rn(un_1_1)*rn) - lambda_n * vn_1_1; vn_1_2' = ...

       if ((ud_1_1 < hd) && (un_1_1 < hn)) state_1_1 = 1; ...
```

```
  }
/* Output */
sample(state_1_1, state_1_2, state_2_1, ...);
```

The indices of the variables correspond to Fig. 6. The intracellular concentrations of Delta and Notch are initialized with a stochastic noise of $\pm 30\%$ about the value 1. Fig. 7a) illustrates a transition that modifies the concentration of Delta in a border cell of a 15x15 network. Fig. 7b) shows the steady state reached at the end of the simulation. Each cell is either in State 2, marked by `*`, or State 3, marked by `o` ("salt-and-pepper" effect). Thus, in spite of its simplicity, this hybrid model provides very accurate qualitative results.



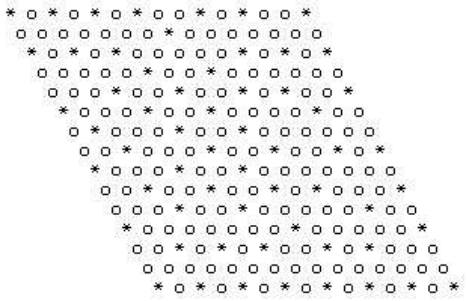


Fig. 7. a) Transition in a border cell b) Steady state

4.5 Benefits of Hybrid cc for modeling biological systems

In addition to the advantages of modeling in a language of the class `Timed Default cc` [22], the following features of `Hybrid cc` seem particularly useful in the context of biology.

Features of Hybrid cc	Interest for biology
Concurrent agents	Clear separation between different processes Possibility to refine the model
Declarative programming	Only the constraints have to be stated. Control done implicitly by `Hybrid cc`
Constraints	Express possibly incomplete knowledge Define the granularity of the analysis
Differential equations	Describe biochemical reactions and continuous variations
Modularity	Isolate the core of a model, crucial for highly context-dependent models; reusability
Asynchronous reactions	Accurate for reactive systems

5 Conclusion and Further Research

`Hybrid cc` is a promising alternative to existing modeling and simulation tools in systems biology, both from the theoretical and the practical point of view. `Hybrid cc` is a highly expressive, compositional programming language based on the constraint programming paradigm. It allows the biologist to model biological systems in a natural and declarative way, without having to know the internals of the system. For the computer scientist, it offers the advantages of a full programming language with a well-defined semantics.

There are various directions to continue this research. The examples presented in this paper were taken from the literature in order to illustrate the interest of `Hybrid cc` for modeling biological systems. They are relatively small. We have started recently a project with the Institut Pasteur in Paris in order develop with `Hybrid cc` a rather complex model of some aspects of the cell cycle in mammalian cells. We hope that this collaboration will not only help to get a better understanding of the biological phenomena, but that it will also provide additional insight into the language `Hybrid cc`. For example, it may turn out that the constraint solving capabilities of `Hybrid cc` have to be enhanced or that the language has to be extended in order to capture specific biologic phenomena. In particular, we may want to consider stochastic extensions like `Probabilistic cc` [10].

References

1. R. Alur, C. Belta, F. Ivancic, V. Kumar, M. Mintz, G. J. Pappas, H. Rubin, and J. Schug. Hybrid modeling and simulation of biomolecular networks. In *Hybrid Systems: Computation and Control, HSCC 2001, Rome, Italy*, pages 19–32. Springer, LNCS 2034, 2001.
2. A. Bockmayr and A. Courtois. Modeling biological systems in hybrid concurrent constraint programming (Abstract). In *2nd Int. Conf. Systems Biology, ICSB'01, Pasadena, CA*, 2001. `http://www.icsb2001.org/Posters/096_bockmayr.pdf`.
3. J. M. Bower and H. Bolouri, editors. *Computational modeling of genetic and biochemical networks*. MIT Press, 2001.
4. B. Carlson and V. Gupta. Hybrid cc and interval constraints. In *Hybrid Systems: Computation and Control, HSCC'98*, pages 80 – 95. Springer, LNCS 1386, 1998.
5. A. Courtois. Modélisation de systèmes biologiques en programmation par contraintes. Rapport de DEA (in French), Univ. Henri Poincaré, LORIA, July 2001.
6. H. de Jong. Modeling and simulation of genetic regulatory systems: a literature review. Technical Report RR-4032, INRIA, September 2000.
7. R. Ghosh and C. Tomlin. Lateral inhibition through Delta-Notch signaling: A piecewise affine hybrid model. In *Hybrid Systems: Computation and Control, HSCC 2001, Rome, Italy*, pages 232–246. Springer, LNCS 2034, 2001.
8. M. A. Gibson and J. Bruck. A probabilistic model of a prokaryotic gene and its regulation. In Bower and Bolouri [3], chapter 2, pages 49–71.
9. M. A. Gibson and E. Mjolsness. Modeling the activity of single genes. In Bower and Bolouri [3], chapter 1, pages 1–48.

10. V. Gupta, R. Jagadeesan, and P. Panangaden. Stochastic processes as concurrent constraint programs. In *26th ACM Conf. Principles of Programming Languages, POPL'99, San Antonio, CA*, pages 189 – 202. ACM, 1999.
11. V. Gupta, R. Jagadeesan, and V. Saraswat. Computing with continuous change. *Science of computer programming*, 30(1-2):3–49, 1998.
12. V. Gupta, R. Jagadeesan, V. Saraswat, and D. G. Bobrow. Programming in hybrid constraint languages. In *Hybrid Systems II*, pages 226–251. Springer, LNCS 999, 1995.
13. M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. Doyle, and H. Kitano. The ERATO systems biology workbench: Enabling interaction and exchange between software tools for computational biology. *Pacific Symposium on Biocomputing*, 7, 2002.
14. T. Ideker, V. Thorsson, J. A. Ranish, R. Christmas, J. Buhler, J. K. Eng, R. Bumgarner, D. R. Goodlett, R. Aebersold, and L. Hood. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science*, 292:929 – 934, May 2001.
15. S. A. Kauffman. *The origins of order*. Oxford Univ. Press, 1993. See also J. Theor. Biol. 22: 437, 1969.
16. H. Kitano, editor. *Foundations of system biology*. MIT Press, 2001.
17. H. Matsuno, A. Doi, M. Nagasaki, and S. Miyano. Hybrid petri net representation of gene regulatory network. *Pacific Symposium on Biocomputing*, 5:338–349, 2000.
18. V. N. Reddy, M. L. Mavrovouniotis, and M. N. Liebman. Petri net representation in metabolic pathways. In *Intelligent Systems for Molecular Biology, ISMB'93*, pages 328–336. AAAI Press, 1993.
19. A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical processes using the π-calculus process algebra. *Pacific Symposium on Biocomputing*, 6:459–470, 2001.
20. V. A. Saraswat. *Concurrent constraint programming*. ACM Doctoral Dissertation Awards. MIT Press, 1993.
21. V. A. Saraswat, R. Jagadeesan, and V. Gupta. Foundations of timed concurrent constraint programming. In *9th Symp. Logic in Computer Science, LICS'94, Paris*, pages 71 – 80. IEEE, 1994.
22. V. A. Saraswat, R. Jagadeesan, and V. Gupta. Timed default concurrent constraint programming. *Journal of Symbolic Computation*, 22(5/6):475–520, 1996.
23. D. Thieffry and R. Thomas. Qualitative analysis of gene networks. *Pacific Symposium on Biocomputing*, 3:77–88, 1998.
24. A. van der Schaft and H. Schumacher. *An introduction to hybrid dynamical systems*. Springer, Lecture Notes in Control and Information Sciences, Vol. 251, 2000.
25. E. O. Voit. *Computational analysis of biochemical systems*. Cambridge Univ. Press, 2000.