Reaction-Driven Membrane Systems

Luca Bianco, Federico Fontana, and Vincenzo Manca

University of Verona, Department of Computer Science,
15 strada Le Grazie - 37134 Verona, Italy
{bianco, fontana}@sci.univr.it
vincenzo.manca@univr.it

Abstract. Membrane systems are gaining a prominent role in the modeling of biochemical processes and cellular dynamics. We associate specific reactivity values to the production rules in a way to be able to tune their rewriting activity, according to the kinetic and state-dependent parameters of the physical system. We come up with an algorithm that exhibits a good degree of versatility, meanwhile it gives an answer to the problem of representing oscillatory biological and biochemical phenomena, so far mostly treated with differential mathematical tools, by means of symbolic rewriting. Results from simulations of the Lotka-Volterra's predator-prey population dynamics envision application of this algorithm in biochemical dynamics of interest.

1 Introduction

Besides their connections with formal language theory, membrane systems often use to be applied to the analysis of biological processes [1,2,3,4]. In particular, *P systems* have come useful provided their capability to represent several structural aspects of the cell along with many intra- and extra-cellular communication mechanisms: dynamic rewriting by means of P systems has already led to alternative representations of different biological phenomena and to new models of important pathological processes [3,4].

By our side we have developed a P system-based algorithm in which rules are specified along with *reactivities*, respectively denoting the "power" of a production rule to process elements such as chemical reactants, bio-molecules and so on [5]. The performances shown in the simulation of the Lotka-Volterra population dynamics foster potential practical application of this algorithm in critical open problems dealt with by computational systems biology.

2 The Algorithm

For the sake of brevity, in this paper the algorithm is formalized for the case of just one membrane [2]. So, let us consider a P system Π working on the alphabet $\mathcal{A} = \{X, Y, \ldots, Z\}$, provided with rules $r, s, \ldots, w \in R$.

The algorithm requires, firstly, to recognize the *state* of the system. This state, along with constant factors (depending, for instance, on chemical kinetic

L. Wang, K. Chen, and Y.S. Ong (Eds.): ICNC 2005, LNCS 3611, pp. 1150–1153, 2005.

parameters), is used to compute specific functions called *reaction maps*. Once we have the values assumed by such maps at hand we normalize them consistently with the available resources (i.e., *objects* in the P system), meanwhile *limiting* these values to avoid to over-consume objects. Finally, a simple stochastic method is employed to decide how to treat individual objects in the system, for which the procedure so far described cannot take a definite decision. This situation occurs when reaction maps ask for partitioning one or more objects into fractional parts of them.

About the definition of state, we postulate that at every discrete time t we can read the type and number of objects within the membrane system. More formally the state at time t is identified by a function $q_t : \mathcal{A} \longrightarrow \mathbb{N}$: for instance, $q_t(X)$ gives the amount of objects X available in the system at time t. The set of all states assumed along time by the system is given by $Q = \{q_t \mid t \in \mathbb{N}\}$: this set contains the complete information on the system evolution.

About the definition of reaction maps (one for each rule), let a reaction map give the reactivity that the corresponding rule has when the system is in a given state. In formal terms, for each rule r we define a reaction map $F_r : Q \longrightarrow \mathbb{R}$ that maps states into non-negative real numbers. Since the state is defined at any temporal step, the application of a reaction map F_r ultimately results in a non-negative real number that we will take as the *reactivity of r in q_t*.

Such maps allow for a wide choice of possible definitions depending on the biological phenomenon under analysis. As an example, consider a membrane system having an alphabet made of five symbols, $\mathcal{A} = \{A, B, C, D, E\}$, and two rules: $r : ABB \to AC$, and $s : AE \to BD$.

Possible structures of the reaction maps might be, for instance, reactivities driven by the *law of mass action* tuned by constant kinetic parameters, k_r and k_s, as well as reactivities depending on an external promoter, like an enzyme capable of activating the reaction. The two possibilities are shown in (1), respectively in the left and right column:

$$\begin{array}{ll} F_r = k_r\, q_t(A) q_t(B) & F_r = q_t(D) \\ F_s = k_s\, q_t(A) q_t(E) & F_s = \{q_t(D)\}^2 \end{array} \tag{1}$$

2.1 Reaction Weights, Limitation and Rounding, and State Transition

Reaction maps are proportionally weighted among rules by means of *reaction weights*. Every reaction weight gives, for each symbol, a population a rule applies to in order to proportionally consume the corresponding objects. By denoting with $\alpha(i)$ the ith symbol in a string α, with $|\alpha|$ the length of the same string, and with $|\alpha|_X$ the number of occurrences of X in α, then we define the reaction weight $W_r\big(\alpha_r(i)\big)$ for $r : \alpha_r \to \beta_r$ with respect to the symbol $\alpha_r(i)$.

Normalization can be expressed in quantitative terms if we think that all rules co-operate, each one with its own reactivity, to consume all available objects. Thus, it must be:

$$\sum_{\rho \in R \mid X \in \alpha_\rho} W_\rho\big(X\big) = 1 \quad \forall\, X \in \mathcal{A} \tag{2}$$

that is, for each symbol the sum of the reaction weights made over the rules containing that symbol in their left part equals unity.

Holding this constraint, we can define the reaction weights for each $r \in R$ as

$$W_r\big(\alpha_r(i)\big) = \frac{F_r}{\sum\limits_{\rho \in R \,|\, \alpha_r(i) \in \alpha_\rho} F_\rho} \quad , \quad i = 1, \ldots, |\alpha_r| \tag{3}$$

Similarly to what happens in (2), here we sum at the denominator over the rules containing the symbol $\alpha_r(i)$ in their left part.

Every rule cannot consume more than the amount of the (reactant) object, *taken with its own multiplicity in the reaction*, whose availability in the system is lowest. Thus, we have to limit the application of every rule by minimizing among all reactant symbols participating to it:

$$\Lambda_r = \min_{i=1,\ldots,|\alpha_r|} \Big\{ W_r\big(\alpha_r(i)\big) \frac{q_t\big(\alpha_r(i)\big)}{\big|\alpha_r\big|_{\alpha_r(i)}} \Big\}. \tag{4}$$

Still, Λ_r is a real number. As opposite to this, a genuine object-based rewriting system must restrict the rule application domain to integer values. We choose the following policy: for every rule, compute $\overline{\Lambda}_r$ by comparing the fractional part of Λ_r to a random variable v_r defined between 0 and 1; choose the floor of Λ_r if this fraction is smaller, the ceiling otherwise. In the simulation of the Lotka-Volterra dynamics v_r can be chosen to have a uniform distribution.

The proposed rounding policy does not prevent from potentially exceeding the available resources in the system. To avoid this we check that $\sum_{r \in R} \overline{\Lambda}_r |\alpha_r|_X \leq q_t(X) \, \forall \, X \in \mathcal{A}$, otherwise the set of minima must be computed again.

In conclusion, for every symbol $X \in \mathcal{A}$ the change $\Delta_r(X)$ in the number of objects due to r is equal to the *stoichiometric factor* of r, equal to $|\beta_r|_X - |\alpha_r|_X$, times the value $\overline{\Lambda}_r$: $\Delta_r(X) = \overline{\Lambda}_r \, (|\beta_r|_X - |\alpha_r|_X)$. It descends that for every symbol $X \in \mathcal{A}$ the state evolves according to the following formula:

$$q_{t+1}(X) = q_t(X) + \sum_{r \in R} \Delta_r(X). \tag{5}$$

3 Simulation: Lotka-Volterra Dynamics

The classic Lotka-Volterra population dynamics [5] can be described by a simple set of rewriting rules in which X are preys and Y predators: $r : X \to XX$ accounts for prey reproduction, $s : XY \to YY$ for predator reproduction, and finally $t : Y \to \lambda$ accounts for predator death.

We tune the activity of every rule by selecting proper reactivity kinetic constants k_r, k_s, and k_t, proportional to the rate of reproduction and death of both predators and preys. Moreover we postulate F_s to be proportional to the maximum number between preys and predators: $F_r = k_r$, $F_s = k_s \max\{q_t(X), q_t(Y)\}$, $F_t = k_t$. Finally we add *transparent rules* [5] accounting for preys that are not

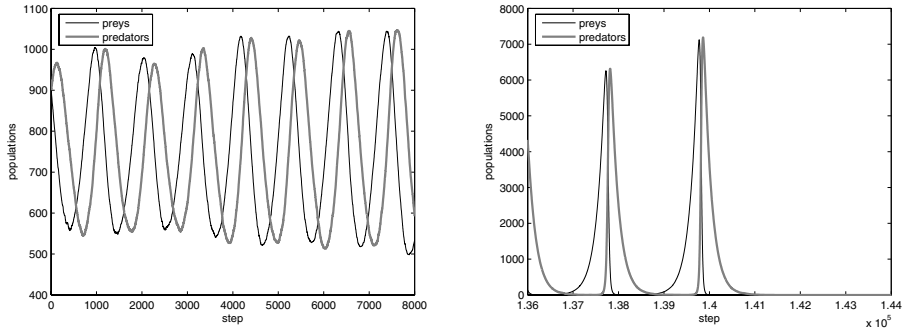

Fig. 1. Predator-prey initial dynamics (left) and after 136000 observation slots (right)

reproducing or being consumed and for predators that are not eating or dieing, respectively: $u : X \to X$ and $v : Y \to Y$, with $F_u = k_u$ and $F_v = k_v$.

Plots of the dynamic behavior of the predator-prey model are depicted in Figure 1. These plots come out when we set $k_r = k_t = 3 \cdot 10^{-2}$, $k_s = 4 \cdot 10^{-5}$, and $k_u = k_v = 5$ along with initial conditions $q_t(X) = q_t(Y) = 900$.

The oscillation can evolve to the death of both species, as in this case, or to the death of the predators solely. The long-term evolution in fact depends on single events taking place when few individuals, either preys or predators, are present in the system. Such a long-term behavior emphasizes the importance of a careful description of not only the reactivities, but also the relationships existing between individuals: the nature of these relationships can completely change the overall system evolution.

References

1. Rozenberg, G., Salomaa, A., eds.: Handbook of Formal Languages. Springer-Verlag, Berlin, Germany (1997)
2. Păun, G.: Membrane Computing. An Introduction. Springer, Berlin, Germany (2002)
3. Bianco, L., Fontana, F., Franco, G., Manca, V.: P systems for biological dynamics. In Ciobanu, G., Păun, G., Pérez-Jiménez, M.J., eds.: Applications of Membrane Computing. Springer, Berlin, Germany (2005) To appear.
4. Mauri, G., Păun, G., Pérez-Jiménez, M.J., Rozenberg, G., Salomaa, A., eds.: Membrane Computing, 5th International Workshop, WMC 2004, Milan, Italy, June 14-16, 2004, Revised Selected and Invited Papers. In Mauri, G., Păun, G., Pérez-Jiménez, M.J., Rozenberg, G., Salomaa, A., eds.: Workshop on Membrane Computing. Volume 3365 of Lecture Notes in Computer Science., Springer (2005)
5. Bianco, L., Fontana, F., Manca, V.: Metabolic algorithm with time-varying reaction maps. In: Proc. of the Third Brainstorming Week on Membrane Computing (BWMC 2005), Sevilla, Spain (2005) 43–62