Fuzzy P systems and their applications in computational biology

Jaume Casasnovas, Francesc Rosselló
Research Institute of Health Science (IUNICS)
Dept. Mathematics and Computer Science
Univ. of the Balearic Islands
{jaume.casasnovas,cesc.rossello}@uib.es

Abstract

We present a fuzzy model of P systems where the objects involved in computations are colored by means of a finite family of fuzzy sets, and we discuss several applications of this model in computational biology.

Keywords: P systems, Fuzzy sets, Computational biology

1 Introduction

Membrane computing [12] is a computational paradigm invented in 1998 by Gh. Păun. Its basic formal computational device, inspired by the living cells' membrane structures, is the *membrane*, or *P* (for Păun), *system*. Roughly speaking, a P system consists of a hierarchical arrangement of *regions* surrounded by *membranes*, all of them embedded in a *skin membrane* and, outside of it, the *environment*. These regions contain multisets over a set of *reactives* (or languages over an alphabet of reactives, depending on the nature of the intended computations) and have associated finite sets of evolution rules reminiscent of the reactions that take place inside cells. Other rules, like membrane creation, dissolution or division, have been considered. In the basic model, rules are applied in a synchronous, non-deterministic, and maximally parallel way, but many other possibilities have also been considered. Sequences of such sets of applications make the P system to evolve through *computations*, and an *output* can be read from each *halting computation*.

Most approaches to membrane computing are universal, in the sense that they generate exactly the r.e. sets of possible outputs. Moreover, several approaches provide polynomial solutions to some NP-complete problems. Besides these features, which make it attractive to theoretical computer scientists, membrane computing has several features that has made it also attractive in the field of computational systems biology. To quote a few from [13]: distribution, algorithmicity, scalability, transparency, parallelism, non-determinism, and communication. A bunch of applications of membrane computing in this area have been proposed so far: see the references given in [12, 13] and the forthcoming book [6]. A preliminary report on the possible applications of P systems in genomics can be found in [8].

The last years have witnessed an increasing interest in the development of 'uncertain' mathematical approaches to membrane computing. The reasons have been, among others, to keep closer to the non-crisp behavior of real cells, the development of new formal computational paradigms dealing with fuzzy information, and the possibility of applying P systems to model real biological processes where handling with uncertainty, errors and approximations is necessary. A first contribution to this line of research was given by A. Obtułowicz and Gh. Păun [9], by extending the classical model to several probabilistic ones. Obtułowicz [10] also discussed several possible rough set based mathematical models of uncertainty that could be used in membrane computing. In a similar vein, several fuzzy approaches have been introduced. In one of them, fuzzy mathematics are used to handle the uncertainty in the number of copies of the reactives in the mem-

branes [11, 15]. In another approach, developed by our research group, fuzzy methods are used to cope with the possibility that the objects in the membranes are imperfect, approximate copies of the reactives purportedly involved in the reactions [3, 4].

In this note we present a generalization of this second approach. In it, the objects contained in membranes, and hence involved in computations, have several properties, which are given by fuzzy subsets of the universe. The values of these properties determine whether an object can play the role of a given reactive in an application of a rule, as well as the values of the properties on the output objects. We also discuss several applications of this approach in computational biology.

2 Crisp P systems

Given an alphabet Σ, we denote by Σ^* the set of words over Σ. Given a word w, we denote by $|w|$ the length of w and, given a letter $a \in \Sigma$, by $|w|_a$ the number of occurrences of a in w.

A *membrane structure* $\mu = (M, E)$ is a rooted tree —with edges pointing to the root r— whose nodes are called *membranes*. This tree represents a hierarchical structure of nested membranes: an edge (m, m') means that m is *directly inside* m'. The root corresponds to the *skin membrane* that surrounds the membrane system. The leaves represent *elementary* membranes, without any membrane inside. To simplify the notations, we expand each membrane structure μ by adding to it a new node *env* and an arc (r, env). Let $\mu^{(env)}$ denote the resulting tree.

Each membrane m defines a *region* C_m, which we identify with the membrane. For an elementary membrane, it represents the space enclosed by it, and for any other membrane it represents the region between this membrane and those directly inside it. The *env* node also defines a region C_{env}, which represents the space outside the skin membrane: the *environment*. It is in these regions where computations take place, and where objects are created, destroyed or moved.

At each moment, every region C_m contains a set of objects that are copies of elements of a certain finite set of *reactives* V. The content of each C_m at any moment is represented by means of a *finite* (i.e., finite-valued and taking non-zero values only on a finite subset of its domain) multiset $F_m : V \to \mathbb{N}$; $F_m(v) = n$ means that there are n copies of v in C_m. If V and M are the sets of reactives and membranes, respectively, of a P system Π (see below), we shall call an M-indexed family of finite multisets $(F_m)_{m \in M}$ a *configuration* of Π.

Now, a *basic crisp P system* is a structure

$$\Pi = (V, \mu, m_{out}, (S_m)_{m \in M}, (\mathcal{R}_m)_{m \in M}),$$

where V is a finite set of *reactives*, $\mu = (M, E)$ is a membrane structure, $m_{out} \in M$ is the *output membrane*, $(S_m)_{m \in M}$ is the *initial* configuration that describes the initial content of each region C_m, and, for every $m \in M$, $\mathcal{R}_m$ is a finite set of *evolution rules* associated to the membrane m: these rules can be seen as specifications of 'reactions' that can take place in C_m, although the objects obtained through these reactions can move to regions directly inside or outside C_m. Every rule in $\mathcal{R}_m$ has the form $R = (\underline{c}; \underline{a} \to \underline{(b, m)})$, where:

– $\underline{c} \in V^*$ represents the *catalysts* of the rule: the reactives that are necessary for the reaction represented by the rule to take place, but that are not modified in any way by this reaction.

– $\underline{a} \in V^*$ represents the *active reactives* of the rule: the reactives that are processed by the reaction, being *spent*, destroyed, by this reaction.

– $\underline{(b, m)} \in (V \times (M \cup \{env\}))^*$ represents the *output reactives*, produced by the reaction, together with the region where each one of them is placed: every (b', m') in this word means that a new reactive b' is produced in the region $C_{m'}$ when the reaction represented by this rule takes place. If (b', m') appears in the word $\underline{(b, m)}$, then either $m' = m$, or they are adjacent in $\mu^{(env)}$.

An evolution rule $R \in \mathcal{R}_{m_0}$ *can be triggered* in a configuration when there are enough copies in C_{m_0} of each reactive for the corresponding reaction to take place; i.e., when $F_{m_0}(v) \geq |\underline{c}|_v + |\underline{a}|_v$ for every $v \in V$. When such a rule R can be triggered in a certain configuration, an *application* of it modifies this configuration into a new

configuration, called the *result* of this specific application, which is obtained as follows: for every $v \in V$, $|\underline{a}|_v$ copies of v are removed from C_{m_0} and, for every m, $|\underline{(b,m)}|_{(v,m)}$ copies of it are added to C_m. Copies added to C_{env} in this way are simply removed from the system.

A *transition* for a membrane system consists of a maximal simultaneous application of rules: the triggering conditions must be satisfied simultaneously for all the rules, and then the results of the applications are obtained simultaneously. The rules applied in a given transition are chosen in a non-deterministic way, and they need not be different. A sequence of transitions, starting with the initial configuration, is called a *computation*. A computation *halts* when it reaches a *halting* configuration where no rule can be triggered. For a halting computation, the *output* is the number of reactives contained in the output membrane; a computation that does not halt does not yield any output. The set of natural numbers *generated* by a given P system Π is the set of all these outputs, obtained through all possible halting computations with respect to it. Basic crisp P systems defined in this way form a universal model of computation, in the sense that they generate exactly all r.e. sets of natural numbers. See [12] for details.

This basic model admits many variations. For instance, the form of the evolution rules can be modified. A useful alternative type of rules are the *symport/antiport rules* $(\underline{a}, in; \underline{b}, out)$. The rough meaning of such a rule in $\mathcal{R}_{m_0}$ is that the reactives specified in $\underline{a}$ move from C_{m_0} to the region directly outside it, say C_{m_1}, and the reactives specified in $\underline{b}$ enter C_{m_0} from C_{m_0}, provided there are enough reactives in both membranes for these two migrations to take place simultaneously. In this model, the contents of C_{env} must be taken into account, as objects from it may enter C_r.

In other models, the objects involved in the reactions can be words over V, and then contents of the membranes are languages, the evolution rules are rewriting rules, and the outputs of computations are again languages. Moreover, a transition can consist of the application of a single rule, instead of a maximally parallel set of applications, or the rules to be applied in each transition can be chosen in a deterministic way, for instance through priorities. Rules modifying the membrane structure can also be used, like for instance the creation, duplication or destruction of membranes. And sets of membranes can be organized forming tissue-like of neural-like systems of cooperating P systems. All these models, and many more, have been introduced and studied in the literature: see [12].

3 P systems with colored objects

We assume now the existence of a *universe* X containing all objects involved in computations, and we consider a finite set $W = \{w_1, \ldots, w_s\}$ of fuzzy subsets of X, whose elements we generically call *colors*. These colors can be used to introduce uncertainty aspects in P systems.

For instance, we can understand the reactives as 'ideal' definitions of the 'chemical compounds' specified in the rules of the P system, and hence as fuzzy subsets of X. In this case W can contain a copy of V, say $\{w_v \mid v \in V\} \subseteq W$, in such a way that each $w_v : X \to [0,1]$ gives the degree of similarity of the objects $x \in X$ to the reactive v.

We can also understand the regions defined by the membranes as fuzzy. Thus, W can contain a copy of M, say $\{w_m \mid m \in M\} \subseteq W$, where each $w_m : X \to [0,1]$ gives the degree with which each $x \in X$ belongs to C_m. And even if the regions are crisp, it can be useful to introduce 'closeness' measures of objects to membranes, for instance in order to determine the plausibility of the application of symport/antiport rules. In this case, members of W could measure how close an object is to a membrane.

It can be convenient to consider other non-crisp properties of the objects; for instance, in biochemical situations, the reactivity or the hydrophobicity of the chemical compounds can be interesting. One can also consider weights of objects: for instance, normalized scores of sequence alignments. All these values can be given by members of W.

Once fixed the set W, we describe the contents of a membrane m by means of a finite multiset

$$F_m : [0,1]^W - \{(0, \ldots, 0)\} \to \mathbb{N}.$$

A value $F_m(t_1,\dots,t_s) = n$ means that in C_m there are n objects x such that $w_i(x) = t_i$, for $i = 1,\dots,s$. We exclude $(0,\dots,0)$ from the domain $Dom(F_m)$ of any F_m because we do not take into account the objects x without any $w_i(x) > 0$. For every such a finite multiset F_m, let $|F_m| = \sum_{\underline{t}\in Dom(F_m)} F_m(\underline{t})$.

If all colors w_i are crisp, these mappings are multisets over the set of all non-empty subsets of W. If all colors w_i are pairwise disjoint, i.e., if $w_i(x) > 0$ implies $w_j(x) = 0$ for every $j \neq i$, then the only elements in $[0,1]^W - \{(0,\dots,0)\}$ that may have a non-zero image under F_m are those with all entries but one equal to 0, and then we can take as the domain of F_m the set $W\times]0,1]$; cf. [3, 4].

A *configuration* for a fuzzy P system with colored objects with set of membranes M and set of colors W will be then a family of finite multisets

$$\Big(F_m : [0,1]^W - \{(0,\dots,0)\} \to \mathbb{N}\Big)_{m\in M}.$$

A *basic fuzzy P system with colored objects*, an *fuzzy P system* for short, is a structure

$$\Pi = (V, W, \mu, m_{out}, (S_m)_{m\in M}, (\mathcal{R}_m)_{m\in M}),$$

where V, μ, m_{out}, $(S_m)_{m\in M}$, and $(\mathcal{R}_m)_{m\in M}$ are as in basic crisp P systems, and W is the finite set of *colors* used in the P system. Now, each evolution rule in $\mathcal{R}_m$ has the form $R = (\underline{c}; \underline{a} \to \underline{(b,m)}, \tau, \phi)$, where:

– $\underline{c}; \underline{a} \to \underline{(b,m)}$ is a rule in the basic crisp P systems sense;

– $\tau : V \times W \to [0,1]$ is a *threshold* function that determines, for every $v \in V$, the least value $\tau(v, w_i)$ that each w_i must take on an object $x \in X$ in order for such an object to be used as the reactive v to the effect of triggering an application of this rule.

– $\phi : W \times (V \times M) \times \mathcal{F}_{|\underline{c}|+|\underline{a}|} \to [0,1]$ is a function that determines the value of the colors of each one of the objects produced by the reaction, depending on the region where they end, in terms of the values of the colors of the objects used in it as catalysts and active reactives. Here $\mathcal{F}_{|\underline{c}|+|\underline{a}|}$ denotes the set of all multisets F over $[0,1]^W - \{(0,\dots,0)\}$ such that $|F| = |\underline{c}| + |\underline{a}|$.

An evolution rule $R = (\underline{c}; \underline{a} \to \underline{(b,m)}, \tau, \phi)$ in $\mathcal{R}_{m_0}$ *can be triggered* in a configuration $(F_m)_{m\in M}$ when, for every $v \in V$, there are at least as many objects in C_{m_0} satisfying the conditions on the values of the colors given by τ corresponding to the reactive v as specified in the rule: formally, when, for every $v \in V$,

$$\sum_{\substack{t_i \geq \tau(v,w_i)\\ i=1,\dots,s}} F_{m_0}(t_1,\dots,t_s) \geq |\underline{c}|_v + |\underline{a}|_v.$$

When a rule R in $\mathcal{R}_{m_0}$ can be triggered in a configuration, an *application* of it modifies this configuration into a new configuration obtained as follows:

(1) For every $v \in V$, we choose sub-multisets $F_{m_0}^{(v,\underline{c})}, F_{m_0}^{(v,\underline{a})}$ of F_{m_0} such that: $|F_{m_0}^{(v,\underline{c})}| = |\underline{c}_v|$, and $(t_1,\dots,t_s) \in Dom(F_{m_0}^{(v,\underline{c})})$ implies $t_i \geq \tau(v, w_i)$ for each $i = 1,\dots,s$; $|F_{m_0}^{(v,\underline{a})}| = |\underline{a}|_v$, and $(t_1,\dots,t_s) \in Dom(F_{m_0}^{(v,\underline{a})})$ implies $t_i \geq \tau(v, w_i)$ for each $i = 1,\dots,s$; and $\sum_{v\in V}(F_{m_0}^{(v,\underline{c})} + F_{m_0}^{(v,\underline{a})}) \leq F_{m_0}$.

(2) We subtract $\sum_{v\in V} F_{m_0}^{(v,\underline{a})}$ from F_{m_0}; let still denote the result by F_{m_0}.

(3) For every m and for every v, we add to F_m a multiset representing $|\underline{(b,m)}|_{(v,m)}$ objects, all of them with each w_i-value equal to $\Phi(w_i, v, m, \sum_{v\in V} F_{m_0}^{(v,\underline{c})} + F_{m_0}^{(v,\underline{a})})$.

The lack of space prevents us from giving all details, we hope the interested reader will be able to reproduce them; cf. [3, 4]. Instead, we give a very simple toy example.

Example. Let Π be a fuzzy P system with set of reactives $V = \{a, b, c\}$; membrane structure μ with only two nodes: the root 1 and an output leaf $m_{out} = 2$; set of colors $W = \{w_a, w_b, w_c, w_1, w_2\}$ (where w_a, w_b, w_c represent degrees of similarity of objects to the corresponding reactives, and w_1, w_2 represent degrees of proximity of objects in C_1 to the membranes 1 and 2); initial configuration defined by $S_2 = 0$ (C_2 is initially empty) and $S_1 = 1/(0.8, 0, 0.3, 0.2, 0.9) + 1/(0.8, 0, 0.3, 0.5, 0.6) + 1/(1, 0.2, 0, 0.1, 0.8) + 1/(0.4, 0, 0.7, 0.1, 0.8)$; and sets of rules $\mathcal{R}_2 = \emptyset$ and $\mathcal{R}_1 = \{R_1, R_2, R_3, R_4\}$, where:

• $R_1 = (ac \to (a,1)(b,2), \tau_1, \Phi_1)$, with $\tau_1(a, w_a) = 0.8$, $\tau_1(a, w_2) = 0.8$, $\tau_1(c, w_c) = 0.7$,

and $\tau_1(c, w_2) = 0.8$ (henceforth, any undefined image of a mapping is assumed to be 0) and
$\Phi_1(w_a, (a, 1), 1/(t_a, t_b, t_c, t_1, t_2)+$
$1/(t'_a, t'_b, t'_c, t'_1, t'_2)) = \frac{1}{2}(\max\{t_a, t'_a\}+\max\{t_c, t'_c\})$
$\Phi_1(w_1, (a, 1), 1/(t_a, t_b, t_c, t_1, t_2)$
$+1/t'_a, t'_b, t'_c, t'_1, t'_2)) = \max\{t_1, t'_1\}$
$\Phi_1(w_2, (a, 1), 1/(t_a, t_b, t_c, t_1, t_2)$
$+1/(t'_a, t'_b, t'_c, t'_1, t'_2)) = \max\{t_2, t'_2\}$
$\Phi_1(w_b, (b, 1), 1/(t_a, t_b, t_c, t_1, t_2)+$
$1/t'_a, t'_b, t'_c, t'_1, t'_2)) = \frac{1}{2}(\max\{t_a, t'_a\}+\max\{t_c, t'_c\})$

• $R_2 = (a^2 \to (a, env)^2, \tau_2, \Phi_2)$, with $\tau_3(a, w_1) = 0.8$ and $\tau_3(a, w_2) = 0.3$; Φ_2 is nonrelevant, since the product of this rule leaves the system.

• $R_3 = ((a \to (a, 1), \tau_3, \Phi_3)$, with $\tau_3(a, w_1) = 0.9$ and $\tau_3(a, w_2) = 0.1$, and
$\Phi_3(w_x, (a, 1), \{(t_a, t_b, t_c, t_1, t_2)) = t_x$ for $x = a, b, c$,
$\Phi_3(w_1, (a, 1), \{(t_a, t_b, t_c, t_1, t_2)) = t_1 + 0.1$,
$\Phi_3(w_2, (a, 1), \{(t_a, t_b, t_c, t_1, t_2)) = t_2 - 0.1$.

• $R_4 = ((a \to (a, 1), \tau_4, \Phi_4)$, with τ_4 and Φ_4 as τ_3 and Φ_3 after interchanging w_1 and w_2.

So, R_3 and R_4 'move' objects in C_1, bringing them near the skin or the elementary membranes, while R_2 removes pairs of objects from the system, provided they are close enough to the skin membrane. Notice that these three rules do not have any threshold condition involving the similarity of objects to reactives: thus, although the reactive a is specified in them, they can also move objects similar to c. Finally, R_1 removes an object similar to a and an object similar to c, both close to membrane 2, an produces a *new* object similar to a in C_1 and an object similar to b in the output region. So, if we apply R_1 to the multiset $1/(1, 0.2, 0, 0.1, 0.9)+$ $1/(0.4, 0, 0.7, 0.1, 0.8))$, we obtain a configuration with $F_1 = 1/(0.8, 0, 0.3, 0.2, 0.9) + 1/(0.8, 0, 0.3, 0.5, 0.6) + 1/(0.85, 0, 0, 0.1, 1))$ and $F_2 = 1/(0, 0.85, 0, 0, 0)$, while if we apply it to the multiset $1/(0.8, 0, 0, 0, 0.9) + 1/(0, 0, 0.7, 0.1, 0.8)$, we obtain a configuration with $F_1 = 1/(1, 0, 0, 0.1, 0.8) + 1/(0.8, 0, 0.3, 0.5, 0.6) + 1/(0.75, 0, 0, 0.1, 0.9))$ and $F_2 = 1/(0, 0.75, 0, 0, 0)$. We cannot apply R_1 to any other submultiset of C_1.

Now, as in the basic case, a *transition* for a membrane system Π consists of a maximal simultaneous application of rules: the triggering conditions must be satisfied simultaneously, and then all steps (1), afterwards all steps (2), and finally all steps (3), corresponding to all rules being applied must be performed simultaneously.

The definitions of computation and halting computation and configuration in this model are the same as those for the basic crisp model. Now the *output* of a halting computation is some measure of the size of the multiset $H_{m_{out}}$ describing the contents of the output membrane in the corresponding halting configuration. The measure of the multiset can be given by some scalar or fuzzy cardinality [5], or by some other suitable expression. The choice will depend on the application. The set *generated* by a fuzzy P system Π will be then the set of all these outputs (or their join, if they are for instance fuzzy subsets of $\mathbb{N}$). We still have not explored the computational power of this model in detail, but we have already proved that, in two special cases, it is universal [3, 4].

Of course, the same variations as in the basic crisp case can be considered in this case. This model can also be enriched by adding a plausibility function to the rules that can be used, besides to define application priorities, to define the plausibility of a configuration obtained after a transition, and in particular the plausibility of the halting configuration of any computation. This last plausibility can be taken into account when obtaining the output value of a computation.

4 Applications in computational biology

As we mentioned in the introduction, one of the reasons of the popularity in membrane computing is its applicability in the modeling of biological complex systems. Adding to P systems the possibility to handle imprecisions, errors, uncertainty, and any kind of non-crisp properties of the objects involved in its computations, enlarges the range of its applicability.

Our approach can be used, for instance, to model in a more realistic way sodium-potassium pumps, which have been already modeled through crisp symport/antiport P systems by D. Besozzi and G. Ciobanu [2]. The colors W can be used to measure the closeness of the reactives to the pumps, and symport/antiport with a Φ mapping to re-

compute the values of the colors can be used to move reactives within a region, getting closer or farther to a pump, and to cross the pump (cf. the Example above); plausibilities can then be used to assign probabilities to rules. Since in this case the reactives are relevant and well-defined (sodium, potassium,...), the set of colors must include crisp mappings identifying the reactives. A similar model is outlined in [1] for mechanosensitive channels, with membership functions to membranes as colors, but they do not recompute *all* closeness colors after an application, only the one corresponding to the target membrane, and therefore we consider our approach more accurate.

As far as the possible applications of this models in the genomics branch of computational biology go, let us mention sequence alignment. As Fontana and Franco state in [7], it is not difficult to devise a P system solving a sequence alignment problem: first, generate all possible alignments using the capability of P systems to execute in a few steps, using its massive parallelism, procedures that performed sequentially need an exponential time; and second, extract the alignment with the best score. Unfortunately, in crisp P systems the score cannot be associated as a weight to the alignment, but only as the number of copies of the alignment in a membrane, and this makes the details of such a P system difficult to specify. Even if the P system designed to solve the alignment problem is based in an incremental or dynamical programming paradigm, the comparison of scores presents the same drawback. Fuzzy P systems, which possess the same structure as crisp P systems but admit the association of numerical fuzzy values to elements of the membranes, are more adequate to implement sequence alignment algorithms. Other applications of P systems in genomics hinted in [8] would benefit of the extra power that coloring the objects adds to this computational paradigm.

References

[1] S. Agguzoli et al, "P systems under uncertainty: the case of transmembrane proteins," in [14], pp. 107–117.

[2] D. Besozzi, G. Ciobanu, "A P Systems Description of the Sodium-Potassium Pump," in *Pre-proc. WMC5* (2004), pp. 154–160.

[3] J. Casasnovas, J. Miró, J. Moyà, F. Rosselló, "An approach to membrane computing under uncertainty," *Int. J. of Found. of Comp. Sc.* 15 (2004), pp. 841–864.

[4] J. Casasnovas, J. Miró, J. Moyà, F. Rosselló, "A fuzzy approach to membrane computing with approximate copies," in [14], pp. 121–127.

[5] J. Casasnovas, F. Rosselló, "Scalar and fuzzy cardinalities of crisp and fuzzy multisets," submitted.

[6] G. Ciobanu, Gh. Păun, M.J. Pérez-Jiménez, *Applications of membrane computing*, Springer-Verlag (2005), in press.

[7] F. Fontana, G. Franco, "Finding the Maximum Element Using P Systems," *J. Univ. Comp. Sc.* 10 (2004), pp. 567–580.

[8] S. Marcus, "Bridging P Systems and Genomics: a preliminary approach," *Lect. Notes in Comp. Sc.* 2597 (2003), pp. 371-376.

[9] A. Obtułowicz, Gh. Păun, "(In search of) probabilistic P systems", *BioSystems* 70 (2003), pp. 107–121.

[10] A. Obtułowicz, "Mathematical models of uncertainty with a regard to membrane systems," in *Proc. First Brainstorming Week on Membrane Computing* (2003), pp. 241–246.

[11] A. Obtułowicz, "General multi-fuzzy sets and fuzzy membrane systems," *Pre-proc. WMC5* (2004), pp. 316–326.

[12] Gh. Păun, *Membrane Computing. An Introduction*, Springer-Verlag, 2002.

[13] Gh. Păun, "Introduction to membrane computing," in [14], pp. 17-65.

[14] *Proceedings of the Brainstorming Workshop on Uncertainty in Membrane Computing* (Univ. of the Balearic Islands, 2004), available at `bioinfo.uib.es/~recerca/BUM`.

[15] A. Syropoulos, "Fuzzyfying P Systems", submitted (2003).