Accepting Evolutionary P Systems

Victor Mitrana[1,2] and José M. Sempere[2⋆]

[1] Faculty of Mathematics and Computer Science
University of Bucharest
[2] Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia,
mitrana@fmi.unibuc.ro, jsempere@dsic.upv.es

P systems were introduced as a computational model inspired by the information and biochemical entities processed in the living cells by means of membrane communication. In most of the works about P systems, information is represented as multisets of symbol/objects which can interact and evolve according to predefined rules. Nevertheless, the use of strings to represent the information and the use of rules to transform strings instead of multiset objects has been present in the literature of this scientific area from the very beginning. For an early survey of different string-based P systems the reader is referred to [3].

In this work, we propose the use of evolutionary transformations from strings to strings as the definition of P rules. The evolutionary rules that we address have been widely used in the definition of *networks of evolutionary processors*, an intensive study started in [1] and continued in a series of papers. *Accepting networks of evolutionary processors with filtered connections* (ANEPFC for short) have been introduced in [2] as a computationally complete model of computation. It is known that many string-based P systems are computationally complete. Our goal is to establish a direct simulation of ANEPFCs by string-based P systems. To this aim, we are going to consider regions with evolutionary rules. Two aspects are important in our view: (1) the permitting conditions of the filters of ANEPFCs are simulated by inner membrane structures, (2) the forbidding conditions of these filters are simulated by rule priorities. In many works devoted to P systems, the membrane structure does not play a very important role as it is reduce to only one membrane. In our approach the membrane structure plays a crucial role.

⋆ Work supported by the Spanish Ministerio de Educación y Ciencia under project TIN2007-60769

Accepting Networks of Evolutionary Processors with Filtered Connections

Here, we will informally describe an ANEPFC as it was defined in [2]. An *evolutionary processor* can be viewed as a very simple string-processing unit. It holds a finite set of strings with arbitrary many copies of each of them, and a finite set of evolutionary rules. These rules can be formally defined by (here a and b ranges over a finite alphabet):

- Insertion rules: $\lambda \rightarrow a$
- Deletion rules: $a \rightarrow \lambda$
- Substitution rules: $a \rightarrow b$

Every rule can be applied to a string in three different ways: in any position in the string, in the rightmost position, or in the leftmost position. Observe that, due to the multiplicity of copies of every string, a single rule applied in an arbitrary position could eventually produce more than one string. The processor will apply the rules to the existing strings in an *evolutionary* step. Basically, an ANEPFC consists of a finite set of evolutionary processors which are connected following a predefined underlying topology (complete, ring, star, etc.) Every connection between two processors is filtered by a pair of disjoint sets of symbols (P, F). There are two types of filters: (1) the *weak* filter (a string passes it if at least one symbol from P and none symbol from F is present in the string), (2) the *strong* filter (a string passes it if every symbol from P and none symbol from F is present in the string). The process of communicating strings between connected processors regulated by the filters associated with connections is called a *communication* step.

A *configuration* of an ANEPFC may be understood as the sets of words which are present in any node at a given moment. A configuration can change either by an *evolutionary step* or by a *communication step* which alternate with each other. When changing by an evolutionary step each component of the configuration is changed in accordance with the set of evolutionary rules associated with every processor and the way of applying these rules. When changing by a communication step, each node processor of an ANEPFC sends one copy of each word it has to every node processor connected to it, provided they can pass the filter of the edge the processors. It keeps no copy of these words but receives all the words sent by any node processor connected with it providing that they can pass the filter of the connection.

Every ANEPFC has two distinguished processors, namely the *input* and the *output* ones. Initially the input string is located in the input node and the network performs an *accepting computation*; if there exists a configuration in which the set of words existing in the output node is non-empty, then the network halts. The *language accepted* by an ANEPFC is the set of input strings that lead the network to a halting configuration.

Accepting Evolutionary P Systems

We now informally describe the P system we are going to investigate. An *Accepting evolutionary P system* of degree m (AEvoP in short) is a construct

$$\Pi = (V, U, \mu, (R_1, \rho_1), \cdots, (R_m, \rho_m)),$$

where:

- V is the input alphabet, $U \supseteq V$ is the working alphabet,
- μ is a membrane structure consisting of m membranes,
- R_i, $1 \leq i \leq m$ is a finite set of *evolutionary and/or dissolving rules* over U associated with the ith region and ρ_i is a partial order relation over R_i specifying the *priority* among the rules. An evolutionary rule is a 4-tuple (a, b, α, β) (or $a \rightarrow b^{\beta}_{\alpha}$) where $a, b \in U \cup \{\lambda\}$, $\alpha \in \{here, out, in\}$ and $\beta \in \{*, l, r\}$. A dissolving rule is 5-tuple $(a, b, \alpha, \beta, \delta)$ (or $a \rightarrow b^{\beta}_{\alpha}\delta$), where a, b, α, β have the same meaning as for evolutionary rules and $\delta \in U$ is the dissolving symbol.

The application of a rule $a \rightarrow b^{\beta}_{\alpha}$ in an arbitrary region of the system works as follows: if there exists a string w in that region, such that $w = u_1 a u_2$, then w is transformed into $u_1 b u_2$ (observe that β establishes the way of applying the evolutionary rule). Parameter α establishes where to send the new strings, namely they are sent to the outer region, to all immediate inner regions (a copy of each string is sent to all these regions), or remain in the same region, provided that β is *out*, *in*, or *here*. If a string is to be sent to an inner region that does not exist, then it remains where it is.

If the rule is a dissolving one ($a \rightarrow b^{\beta}_{\alpha}\delta$, the membrane of the region is dissolved after the rule application, provided that the membrane is different from the skin one.

The input string is initially stored in the outmost region. Then, in a fully parallel manner all the rules are applied to the strings existing in every region according to their priorities. The system halts whenever: (1) No rule can be applied, or (2) The system is reduced to only one region, namely the outmost one.

The language accepted by Π is denoted by $L(\Pi)$. A string is in $L(\Pi)$ if and only if it being initially stored in the outmost region reduces the system to only one region.

A simulation of ANEPFCs by EvoPs

In this section we give just a very brief idea how an AEvoP can simulate an ANEPFC. The membrane structure for the proposed AEvoP system will have the skin region and as many regions as connections between processors inside it. For every connection between processor i and j we will have the regions R_{ij} and R_{ji}. Inside every R_{ij} region we will have different structures depending on the filter type. Let us suppose that the set of permitting symbols for the filter on the connection between processor i and j is defined by $P_{ij} = \{b_1, b_2, \cdots, b_k\}$. Then, if

the filter acts in the weak mode, the structure is showed in the next figure to the left, while if the filter acts in a strong mode the structure is showed to the right.

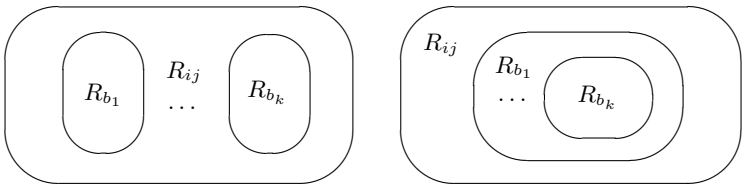

Fig. 1. Membrane structures for the filters

Inside the inner regions in illustrated membrane structures we apply the evolutionary rules similarly to those associated with the node i. In addition, the new strings are moved through the region in order to check the filter conditions. Thus evolutionary rules having the highest priority check the presence of forbidden symbols. If such a symbol is present, then the string remains blocked in an inner region. If these rules cannot be applied, then other evolutionary rules check the presence of permitting symbols. As soon as one permitting symbol is present, the string is sent to the outer region. Clearly, some special symbols are used in order to manage the string movements. When a string is going to enter a region of the form R_{ij} where i is the output node of the ANEPFC, then a new symbol is inserted; this symbol will dissolve in turn all the membranes.

References

1. J. Castellanos, C. Martín-Vide, V. Mitrana and J. M. Sempere. Networks of evolutionary processors. *Acta Informatica* Vol.39 No. 6-7, pp 517-529. 2003
2. C. Drăgoi, F. Manea, V. Mitrana, Accepting networks of evolutionary processors with filtered connections, *Journal of Universal Computer Science*, 13 pp 1598–1614 (2007).
3. Gh. Păun. *Membrane Computing. An Introduction.* Springer. 2002.