Proceedings of the International Workshop on Intelligent Information Systems
IIS2011, September 13-14, 2011, Chisinau, Republic of Moldova

Application of P system Models in Computer Linguistics

Artiom Alhazov, Elena Boian, Constantin Ciubotaru, Svetlana Cojocaru, Alexandru Colesnicov, Ludmila Malahov, Yurii Rogozhin

Abstract

We present an overview of P system models for solving classical problems in natural language processing: word search; insertion in dictionary; and inflexion for the Romanian language. These methods combine both parallel and sequential aspects of the aforementioned problems.

Keywords: bio-molecular computations, P systems, computer linguistics.

Various problems from natural language processing can be solved by using linguistic resources: collections of texts, corpora, dictionaries, etc. Typically, these resources are large, and their processing requires powerful computers. To solve this problem effectively, parallel computations were used. Bio-molecular computing is one of modern approaches where the parallelism is an intrinsic feature.

Membrane systems (also called P systems) are recently introduced models of a living cell [1]. P systems are distributed computational devices that synchronously process multisets of objects in compartments of the given membrane structure. The objects may also travel through membranes. Membranes form a hierarchical structure. They may be dissolved, divided, created, and their permeability may change. A sequence of transitions between configurations of a P system forms a computation. The result of a halting computation is the number of

objects in the specified output membrane. The objects may also be, e.g., strings over a specified alphabet. In this case, the result of the computation is a collection of strings.

We will discuss several problems of natural language processing: dictionary search and completion, and generation of new words by inflexion. The described method uses models of P systems that can be parallel or sequential. The choice of specific model of P system is made so that the massive parallelism characteristic for these systems can be effectively used.

Each **dictionary** may be represented as a function defined on a finite set of strings, e.g., $\{u_i \longrightarrow v_i \mid 1 \leq i \leq d\}$. Consider a representation of a dictionary by the skin membrane containing the membrane structure corresponding to the prefix tree of $\{u_i \mid 1 \leq i \leq d\}$, with strings v_i appearing in compartments corresponding to the nodes associated to u_i. Due to technical reasons, assume that the skin contains membranes labeled l for each $l \in A_1$ (A_1 is the input alphabet). We also assume that the input words are not empty. For instance, a dictionary of synonyms $\{\text{bar} \longrightarrow \text{local}, \text{bir} \longrightarrow \text{impozit}\}$ can be represented by

$$\Big[\; [\;]^0_a \Big[\; [\; [\; \$local\$' \;]^0_r \;]^0_a \; [\; [\; \$impozit\$' \;]^0_r \;]^0_i \; \Big]_b [\;]^0_c \cdots [\;]^0_z \; \Big]^0_0$$

The simple search in a dictionary can be implemented by the following rules: the input word propagates over the membrane structure until the corresponding place; labeling the region corresponding to the input word; replicating the translation; sending one copy to the environment; keeping the other copy in the dictionary. If the word is not found, the input is "stuck" in the membrane structure and the system gives no answer. To handle the situations of failed search, the set of rules should be extended. For instance, handling 3 situations (translation found, translation not found, position not reachable) can be done by 22 groups of rules, see [2]. Dictionary completion can be done by 8 groups of rules.

The second problem P systems were used to solve is morphological inflexion. The proposed method was used for Romanian.

All groups of **morphological inflexion of words** in the Romanian language can be divided in two parts: with alternating vowels or consonants, and without alternations.

Take a word $w = w'\alpha\sharp$, where the length $|\alpha| \geq 0$ of termination is given, and symbol $\sharp$ is the endmarker. The elements of the list $T = (f_1, f_2, ..., f_k)$ should be appended to w'. The following flections are obtained: $w'f_1$, $w'f_2$, ..., $w'f_k$.

Consider a cooperative P system with replication [1] for describing the process of the inflexion of masculine nouns without alternating vowels or consonants. Such a system contains the following components: $\Pi = (O, \Sigma, \mu, M_1, R)$, where: V is the input alphabet of the Romanian language; $O = \Sigma = V \cup \{\sharp\}$ is the alphabet of the system; the input alphabet is the same; $\mu = [_1\ ]_1$ is the structure consisting of a single membrane labeled 1; $M_1 = \varnothing$ is the initial contents of region 1. The set of rules is $R = \alpha\sharp \rightarrow (f_1, ||f_2, ||...||f_k, out)$. For example, let w='pom' (apple), which is a masculine noun without a termination, i.e., $|\alpha| = 0$, and the list of terminations is T=(–, –, –, ul, ului, ule, i, i, i, ii, ilor, ilor). In this case, the rule $\alpha\sharp \rightarrow (f_1, ||f_2, ||..||f_k)$ is applied and the following flections are obtained: *pom, pom, pom, pomul, pomului, pomule, pomi, pomi, pomi, pomii, pomilor, pomilor.*

The general model processes alterations and will require either a more complicated structure, or a more sophisticated approach [3].

The specific character of the investigated area (natural language) is reflected in the fact that many of the objects and concepts it operates, cannot be the subject to strict formalisation. Therefore we will try to distinguish certain classes in which this formalisation is possible.

Determination of inflexion group. Suppose we have the word-lemma in its graphical representation, the part of speech, and the gender for nouns. We divide the words into three categories: irregular, absolutely regular and partially regular. For all parts of speech the fact of belonging to the *irregular* class is determined by the fact of their belonging to a set of words known a priori. To simplify the statement we exclude from the examination the set of irregular words; their presence or absence does not affect the generality of the algorithm. We

distinguish between *absolutely regular* words, to which a single inflectional model corresponds, and *partially regular* words, to which two or more inflectional models correspond. We establish the criteria for belonging to last two classes and corresponding inflectional models.

We propose to obtain these criteria in parallel mode using massive parallelism which is characteristic to membrane P systems. The algorithm of inflectional group determination will be made in two steps [4]: building the sets of endings of the same length, to which the inflectional models are being put into correspondence; determination of the inflectional group in correspondence with the built sets of endings.

A few models of P systems solving certain natural language processing problems are explained. These models explain possibilities of applying P systems in computer linguistics. Such methods will be later integrated as components to solve more complex problems in this area.

References

[1] Gh. Păun. *Membrane computing: an introduction.* Published by Springer, 2002, 419 p.

[2] A. Alhazov, S. Cojocaru, L. Malahov, Yu. Rogozhin. *Dictionary Search and Update by P Systems with String-Objects and Active Membranes.* International Journal of Computers, Comunications & Control (IJCCC), Vol IV(2009), No 3, pp. 206-213.

[3] A. Alhazov, E. Boian, S. Cojocaru, Yu. Rogozhin. *Modelling Inflexions in Romanian language by P Systems with String replications.* Computer Science Journal of Moldova, v.17, 2(50), 2009, pp.160–178.

[4] S. Cojocaru, E. Boian. *Determination of inflexional group using P systems.* Computer Science Journal of Moldova, vol.18, no.1(52), 2010, pp. 70–81.

Artiom Alhazov, Elena Boian, Constantin Ciubotaru, Svetlana Cojocaru, Alexandru Colesnicov[1], Ludmila Malahov, Yurii Rogozhin

[1] Institute of Mathematics and Computer Science, Chisinau, Moldova
E–mail: *kae@math.md*