# Deductive databases and P systems

**Vladimir Rogozhin [1] and Miguel A. Gutiérrez-Naranjo [2]**

[1] The State University of Moldova
60 Mateevich str., MD-2009 Chişinău, Moldova

[2] Dpto. de Ciencias de la Computación e Inteligencia Artificial
E.T.S. Ingeniería Informática. Universidad de Sevilla
Avda. Reina Mercedes s/n, 41012, Sevilla, España

E-mail: `rv@math.md, magutier@us.es`

**Abstract.** In computational processes based on backwards chaining, a rule of the type $A \leftarrow B_1, \ldots, B_n$ is seen as a procedure which points that the problem $A$ can be split into the problems $B_1, \ldots, B_n$. In classical devices, the subproblems $B_1, \ldots, B_n$ are solved sequentially. In this paper we present some questions that circulated during the Second Brainstorming Week related to the application of the parallelism of P systems to computation based on backwards chaining on the example of *inferential deductive process.*

## 1 Introduction

In computational processes based on backwards chaining, a rule of the type $A \leftarrow B_1, \ldots, B_n$ is usually seen as a procedure which points that the problem $A$ can be split into the problems $B_1, \ldots, B_n$ with the hope that $B_1, \ldots, B_n$ are simpler than $A$. In the case of getting $B_1, \ldots, B_n$ solved, we also have a solution for $A$ via this rule.

This is the case of pure Prolog [2, 10] where $A \leftarrow B_1, \ldots, B_n$ is a definite clause and $A, B_1, \ldots, B_n$ are positive literals. Prolog uses SLD resolution to find an *answer* to the *goal* $A$, i.e., *L*inear resolution for *D*efinite clauses with *S*election function. This selection function considers sequentially the list of current subgoals $B_1, \ldots, B_n$ and chooses one of them (in standard Prolog the selection function always takes the leftmost literal). The process of finding an answer for the chosen subgoals generates new subgoals, hopefully simpler than the previous one. The computation ends when trivial subgoals are reached.

The selection mapping is necessary because classic computational devices work sequentially, so we need to fix an order between the tasks.

In this paper we present some questions that circulated during the Second Brainstorming Week related to the application of the parallelism of P systems to the computation based on backwards chaining.

## References

[1] Apt, K.R.: *Logic Programming*, Handbook of Theoretical Computer Science. Elsevier Science Publishers B.V., 1990

[2] Bratko, I.: *PROLOG Programming for Artificial Intelligence*, Third edition. Addison-Wesley, 2001.

[3] Doets K.: From Logic to Logic Programming. The MIT Press, 1994.

[4] Dzeroski, S. and Lavrac N.: *An Introduction to Inductive Logic Programming* in Relational Data Mining, Springer, Berlin 2001. pp.:48-73

[5] Krishna S.N., Rama R.: A Variant of P Systems with Active Membranes: Solving NP-Complete Problems. *Romanian Journal of Information Science and Technology*, 2, 4 (1999), 357–367.

[6] Lloyd J.W.: Foundations of Logic Programming. (2nd ed.) Springer, Berlin, 1987

[7] G. Metakides, A. Nerode: PRINCIPLES OF LOGIC AND LOGIC PROGRAMMING, *Studies in Computer Science and Artificial Intelligence*, 1996.

[8] Nienhuys-Cheng S.H. and de Wolf R.: Foundations of Inductive Logic Programming, LNAI 1228. Springer 1997

[9] Păun, Gh.: Computing with membranes, *Journal of Computer and System Sciences*, **61**, 1 (2000), 108–143.

[10] Logic Programming: `http://www.afm.sbu.ac.uk/logic-prog/`

[11] Deductive Databases: The DATALOG Approach:
`http://goanna.cs.rmit.edu.au/ zahirt/Teaching/subj-datalog.html`