SOLVING DIOPHANTINE EQUATIONS WITH A PARALLEL MEMBRANE COMPUTING MODEL

Alberto Arteta, Nuria Gomez, Rafael Gonzalo

Abstract*: Membrane computing is a recent area that belongs to natural computing.. P-systems are the structures which have been defined, developed and implemented to simulate the behavior and the evolution of membrane systems which we find in nature. Diophantine equations are those equations that have integer solutions. Currently, the extended Euclidean algorithm works to find integer solutions. .This paper shows a step by step procedure that solves a Diophantine equation by processing the extended Euclidean Algorithm*

Keywords*: Extended Euclidean Algorithm, Membrane systems .*

Introduction

Natural computing is a new field within computer science which develops new computational models. These computational models can be divided into three major areas:

1. Neural networks.
2. Genetic Algorithms
3. Biomolecular computation.

Membrane computing is included in biomolecular computation. Within the field of membrane computing a new logical computational device appears: The P-system. These P-systems are able to simulate the behavior of the membranes on living cells. This behavior refers to the way membranes process information. (Absorbing nutrients, chemical reactions, dissolving, etc)

In this paper, we design a MEIA system just by explaining the process of encrypting the information that membrane systems process.

In order to do this we will take the following steps:

- Introduction to P-systems theory.
- Introduction to encryption algorithms
- Integration of the encryption with membrane systems
- Description of MEIA
- Applications of MEIA

Introduction to P-systems theory

I. A P-system is a computational model inspired by the way the living cells interact with each other through their membranes. The elements of the membranes are called objects. A region within a membrane can contain objects or other membranes. A p-system has an external membrane (also called skin membrane) and it also contains a hierarchical relation defined by the composition of the membranes. A multiset of objects is defined within a region (enclosed by a membrane). These multisets of objects show the number of objects

existing within a region. Any object 'x' will be associated to a multiplicity which tells the number of times that 'x' is repeated in a region.

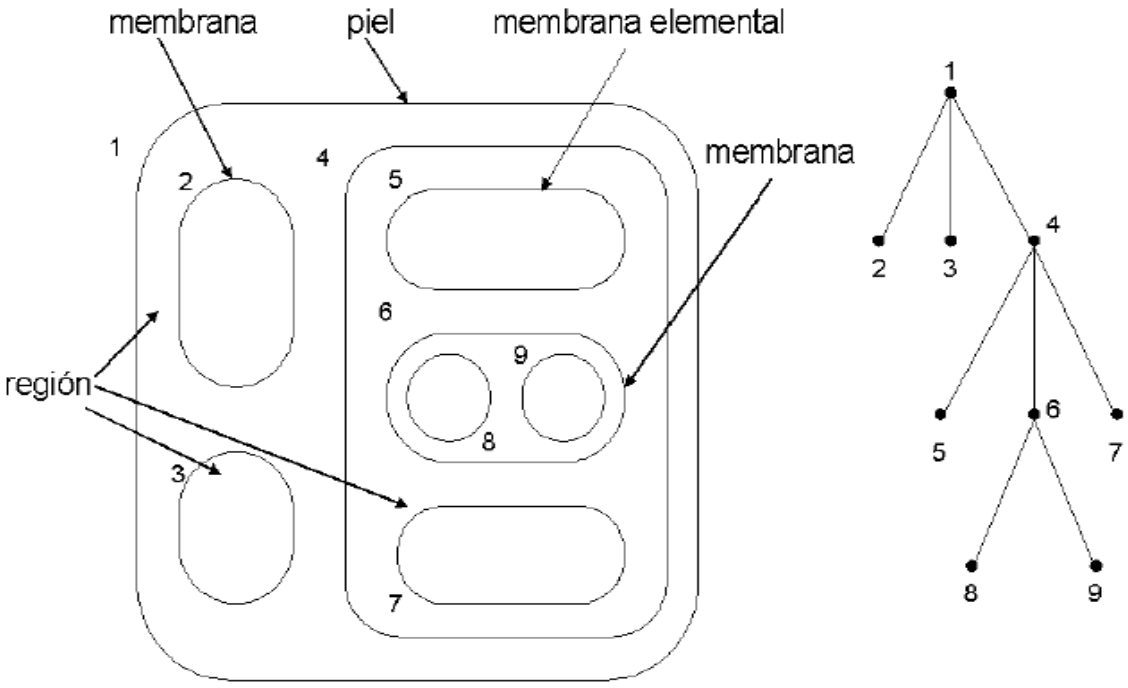


Fig. 1. The membrane's structure (left) represented in tree shape (right)

According to Păun 's definition, a transition P System of degree n, n > 1 is a construct: [Păun 1998]

$$\prod = \left(V, \mu, \omega_1, .., \omega_n, (R_1, \rho_1), .. (R_n, \rho_n), i_0\right)$$

Where:

1. *V* is an alphabet; its elements are called objects;
2. μ is a membrane structure of degree n, with the membranes and the regions labeled in a one-to-one manner with elements in a given set ; in this section we always use the labels 1,2,..,n;
3. $\omega_i \; 1 \le i \le n$, are strings from V^* representing multisets over V associated with the regions 1,2,..,n of μ
4. $R_i \; 1 \le i \le n$, are finite set of evolution rules over V associated with the regions 1,2,..,n of μ; ρ_i is a partial order over $R_i \; 1 \le i \le n$, specifying a priority relation among rules of R_i . An evolution rule is a pair (u,v) which we will usually write in the form $u \to v$ where u is a string over V and v=v' or v=v' δ where v' is a string over $\left(V \times \{here, out\}\right) \cup \left(V \times \{in_j \; 1 \le j \le n\}\right)$, and δ is a special symbol not in. The length of u is called the radius of the rule $u \to v$

5. i_o is a number between 1 and n which specifies the output membrane of Π

Let *U* be a finite and not an empty set of objects and N the set of natural numbers. A *multiset of objects* is defined as a mapping:

$$M : V \to \mathrm{N}$$
$$a_i \to u_1$$

Where a_i is an object and u_i its multiplicity.

As it is well known, there are several representations for multisets of objects.

$$M = \{(a_1, u_1), (a_2, u_2), (a_3, u_3)...\} = a_1^{u_1} \cdot a_2^{u_2} \cdot a_n^{u_n} .....$$

Evolution rule with objects in *U* and targets in *T* is defined by $r = (m, c, \delta)$ where
$m \in M(V), c \in M(VxT)\ and\ \delta \in \{to\ dissolve, not\ to\ dissolve\}$

From now on *'c'* will be referred to as the consequent of the evolution rule *'r'*

The *set of evolution rules* with *objects* in *V* and targets in *T* is represented by R *(U, T).*

We represent a rule as:

$x \to y \quad or \quad x \to y\delta$ where x is a multiset of objects in M((*V*)xTar) where Tar ={here, in, out} and y is the consequent of the rule. When δ is equal to "dissolve", then the membrane will be dissolved. This means that objects from a region will be placed within the region which contains the dissolved region. Also, the set of evolution rules included on the dissolved region will disappear.

P-systems evolve, which makes it change upon time; therefore it is a dynamic system. Every time that there is a change on the p-system we will say that the P-system is in a new transition. The step from one transition to another one will be referred to as an evolutionary step, and the set of all evolutionary steps will be named computation. Processes within the p-system will be acting in a massively parallel and non-deterministic manner. (Similar to the way the living cells process and combine information).

We will say that the computation has been successful if:

1. The halt status is reached.
2. No more evolution rules can be applied.
3. Skin membrane still exists after the computation finishes.

Extended Euclidean Algorithm

The algorithm Is used to find single solutions of Diophantine equations and great common divisor, given 2 numbers. This is the algorithm:

r_0 : = *a*, x_0 : = 1, y_0 : = 0. *Initial values*.

r_1 : = *b*, x_1 : = 0, y_1 : = 1.

i : = 1.

IF r_i = 0 RETURN r_{i-1}.. **we are done**

IF r_i > 0, DO

Dividing r_{i-1} by r_i obtaining k_i y r_{i+1} .

$r_{i-1} = k_i r_i + r_{i+1}$ $\quad 0 \le r_{i+1} < r_i$.

$x_{i+1} := x_{i-1} - k_i x_i$ e

$y_{i+1} := y_{i-1} - k_i y_i$.

i : = i + 1 and GOTO step 4.

If r_n is the last non-zero remain we are done.

$$\text{mcd}(a, b) = r_n = a x_n + b y_n$$

In fact $r_i = a x_i + b y_i, \ \forall\ i = 0, ..., n$.

By following this algorithm we can build this table.

$r_0 = a$	$r_1 = b$	r_2	...	r_{n-1}	r_n	$r_{n+1} = 0$
	k_1 =	k_2		k_{n-1}	k_n	
$X_0 = 1$	$x_1 = 0$	x_2		x_{n-1}	x_n	
$Y_0 = 0$	$y_1 = 1$	y_2		y_{n-1}	y_n	

Exercise: Find the gcd(282,84) and a linear combination that relates gcd(282,4) to 282 and 84..

i	**0**	**1**	**2**	**3**	**4**	**5**
r_i	282	84	30	24	**6**	0
k_i		3	2	1	4	
x_i	1	0	1	-2	***3***	
y_i	0	1	-3	7	***-10***	

Following the algorithm we obtain:

The last non-zero remain is $r_4 = 6$. Therefore gcd(282, 84) = 6.

Furthermore we obtain the following linear combination:

$$6 = 282 \cdot x_4 + 84 \cdot y_4$$

$$6 = 282\ (3)+84\ (-10)$$

The use of the extended Euclidean algorithm is useful for solving Diophantine equations.

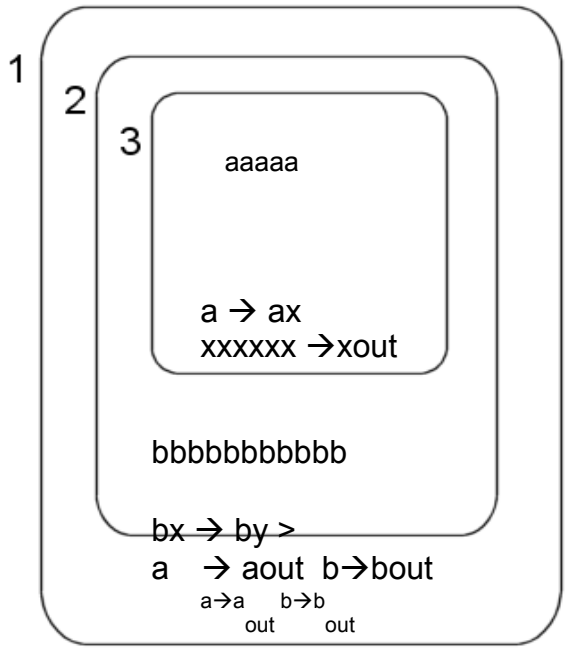
Transition P-systems solving the equation

Let us be the equation ax+by=c

The transition P-systems consist of 3 membranes:

Membrane 1 contains Membrane 2. and membrane 2 contains membrane 3.

The idea is that membrane 1 does the calculation



i

The 2 numbers involved are the number of a's and b's. The number of a's are decreasing ans passed by to membrane 2, which will do the intermediate operation to obtain y, So in the end the number of a's and b's out

The membrane 3 synchronized itself with membrane 2 . The idea is that membrane 3 is applying the euclidean algorithm step by step rising the number of 'x' until obtained the halt condition. And returns the number of a's and' b's solution (x,y) of the initial equation.

In the end:

x=aaaaaaa..aaa

y=bbbbbbb..bbbb

When this operation occurs is important to point out that it occurs in a parallel and non deterministic manner, which implies that the performance of this method is optimal in comparison with the other processing models such as the Turing machine.

Conclusions

The idea of implementing this biological model is taking advantages of the parallelism to do these operations. This theoretical model proposed here shows that it'd be possible to obtain solutions to Diophantine equations or, maybe calculate the great common divisor in a minimum number of operations. With these kind of models and optimal performance is guaranteed.

Bibliography

[Păun 1998] "Computing with Membranes", Journal of Computer and System Sciences, 61(2000), and Turku Center of Computer Science-TUCS Report nº 208, 1998.

[A. Arteta, 2008] I "Algorithm for Application of Evolution Rules based on linear diofantic equations" Synasc 2008, Timisoara Romania September 2008[1] A. Syropoulos, E.G. Mamatas, P.C. Allilones, K.T. Sotiriades "A

[Arroyo, 2001] "Structures and Bio-language to Simulate Transition P Systems on Digital Computers," Multiset Processing (. [Arroyo, 2003] "A Software Simulation of Transition P Systems in Haskell, Membrane Computing,"

[Ciobanu, Pérez-Jiménez, Ciobanu, Păun 2006] M. Pérez-Jiménez, G. Ciobanu, Gh. Păun. Applications of Membrane Computing, Springer Verlag. Natural Computing Series, Berlin, October, 2006.

[Fernández, Castellanos, Arroyo, tejedor, Garcia 2006] L. Fernández, J.Castellanos, F. Arroyo, J. tejedor, I.Garcia. New algorithm for application of evolution rules. Proceedings of the 2006 International Conferenceon Bioinformatics and Computational Biology, BIOCOMP'06, Las Vegas, Nevada, USA, 2006.

[Fernández, Martínez, Arroyo, Mingo 2005] L. Fernández, V.J. Martínez, F. Arroyo, L.F. Mingo. A Hardware Circuit for Selecting Active Rules in Transition P Systems. Proceedings of International Workshop on Theory and Applications of P Systems.Timisoara (Romania), September, 2005.

[Pan, Martin 2005] L. Pan, C. Martin-Vi de. Solving multidimensional 0-1 knapsack problem by P systems with input and active membranesl. Journal of Parallel and Distributed Computing Volume 65 , Issue 12 (December 2005)

[Păun 2000] Gh. Păun. Computing with Membranes. Journal ofComputer and System Sciences, 61(2000), and Turku Center of Computer Science-TUCS Report nº 208, 1998.

[Păun 2005] Gh. Păun. Membrane computing. Basic ideas, results, applications. Pre-Proceedings of First International Workshop on Theory and Application of P Systems,Timisoara (Romania), pp. 1-8, September , 2005.

[Qi, Li, Fu, Shi, You 2006] Zhengwei Qi, Minglu Li, Cheng Fu, Dongyu Shi, Jinyuan You. Membrane calculus: A formal method for grid transactions. Concurrency and Computation: Practice and Experience Volume18,Issue14 , Pages1799-1809. Copyright © 2006 John Wiley & Sons, Ltd.

Authors' Information

Alberto Arteta Albert *– Associate professor U.P.M Crtra Valencia km 7, Madrid-28031, Spain; e-mail: aarteta@eui.upm.es*
Research: Membrane computing, Education on Applied Mathematics and Informatics

Nuria Gomez*– Associate professor U.P.M, Crtra Valencia km 7, Madrid-28031, Spain; e-mail: ngomez@eui.upm.es*
Research: Membrane computing, Education on Informatics
Rafael Gonzalo *- Professor, faculty of informatics. Campus de Montegancedo. Boadilla e-mail: rgonzalo@fi.upm.es*
PhD on Artificial Intelligence, Education on Mathematics and Informatics