Application of Variable Neighborhood Search (VNS) in Configuring Membrane Systems Communications Architectures

Sandra Gómez Canaval, Abraham Gutiérrez Rodríguez and Soledad Delgado Sanz
Natural Computing Group
Universidad Politécnica de Madrid
Madrid, Spain
{sgomez,abraham,sole}@eui.upm.es

ABSTRACT

Nowadays, it is possible to find out different viable communication architectures that implements P-Systems in a distributed cluster of processors. These proposed architectures have reached a certain compromise between the massively parallelism character of the system and the evolution step times. They are based in the distribution of several membranes in each processor, the use of proxies to control the communication between membranes and mainly, the suitable distribution of the architecture in a balanced tree of processors. For a given P-System and *K* processors, there exists a great volume of possible distributions of membranes over these. The main disadvantage related with these architectures is focused in the selection of the distribution of membranes that minimizes the external communications between them and maximizes the parallelism grade. Recently has been demonstrated the usefulness of Self-Organizing Neural Networks (SONN) to help in the selection process of balanced distributions of membranes for a given P-System. The aim of this study is to test the feasibility of using a heuristic search based on Variable Neighborhood Search (VNS) to adjust the results generated by the SONN.

Keywords

Membrane computing, Transition P-System, Self-Organizing Neural Networks, Variable Neighborhood Search.

1. INTRODUCTION

Membrane computing are a distributed highly parallel computational model introduced by G. Păun [1] inspired on basic features of biological membranes and the observation of biochemical processes that offers multiples possibilities by for solving NP-problems. Transition P-Systems are membrane systems defined like a computational device which is an abstract representation of a complex hierarchical structure of membranes. One membrane defines a region where there are a series of chemical components (*multisets*) that are able to go through chemical reactions (*evolution rules*) to produce other elements. Inside the region delimited by a membrane can be placed other membranes defining a tree as is illustrated in the Figure 1 b).

Generated products by chemical reactions can remain in the same region or can go to another region crossing a membrane. Transition P-Systems evolves transiting from an initial configuration to the halting one (in the case of obtaining a successful computation) [1].

The system configurations are determined by the membrane structure and *multisets* present inside membranes. The rules are applied simultaneously by observing the so-called maximal parallelism principle, that is the rules are selected in such a way that only "optimal" output is yielded. When it is not possible to apply any rule, the P-System halts. A designated compartment, called the output compartment, contains the output of the computation, which is equal to the cardinality of the *multiset* contained in it.

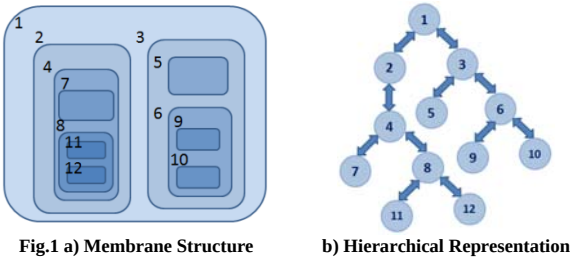

Fig.1 a) Membrane Structure **b) Hierarchical Representation**

In the formal Transition P-Systems model can be distinguished two phases in each evolution step: rules application and communication. Once application rules phase is finished, then it begins communication phase, where those generated *multisets* travel through membranes towards their destination in case it is another region. These systems carry out computations through transitions between two consecutive configurations, what turn them into a computational model with the same capabilities as Turing machines. Power of this model lies in the fact that the evolution process is massively parallel in application rules phases as well as in communication phase. The challenge for researchers is to achieve hardware and/or software implementations of P-Systems respecting the massively parallelism in both phases.

In this sense, in [2] its presented an hardware architecture based on microcontrollers for the implementation of the distributed architectures proposed in [3][4]. This hardware solution is interesting because allows a high degree of parallelism at a cheaper cost than other hardware/software implementation solutions.

Recently has been demonstrated in [5] that independent of the hardware architecture, an important disadvantage related with the communication architectures is the great volume of possible combinations of membrane distributions over a number of processors that can vary from one to the number of membranes. To resolve this problem, in [5] is suggested the use of a Self-Organizing Map (SOM) with growing capability [6], to assist in the search and selection of membranes distributions between processors that obtain a reduction of run times of each step of evolution in the P-System.

The process of finding solutions by the SOM results in a set of possible distributions whose volumes of internal and external communications are balanced. The set of results should be studied to select the final solution to be implemented. This paper suggests the use of a metaheuristic algorithm (VNS), which performs a local search over the set of solutions to obtain automatically the best solution possible.

This paper is structured in the following way: first, the related works is presented; second, the Variable Neighborhood Search (VNS) metaheuristic is illustrated and third, experiments and results are presented. Finally, future works and conclusions are introduced.

2. RELATED WORKS

Nowadays, it is possible to find out at least three different viable architectures that implements P Systems in a distributed cluster of processors: P2P [9], Hierarchical P2P [3] and Master-Slave [4]. These proposed architectures have reached a certain compromise between the massively parallelism character of the system and evolution step times. In particular, they have focused in the second phase of an evolution step and have obtained good results in the throughput in the external communication among processors and parallelization levels of the system. These architectures are based in the distribution of several membranes in each processor, the use of proxies to control the communication between membranes and mainly, the suitable distribution of the architecture in a balanced tree of processors. These solutions avoid communication collisions, and reduce the number and length for communication among membranes. All this facts allows obtaining a better step evolution time than in others suggested architectures congested quickly by the network collisions when the number of membranes grows.

In [2] a tree topology is generated like the distributed communications architectures proposed, but adapted to use a component based on microcontrollers. The designed prototype demonstrates that it is possible to implement the communication architectures. This hardware solution is interesting because it allows a high degree of parallelism with a very low cost respect the others hardware implementation solutions. This implementation presents a number of specific components: a basic unit and several architectonics elements (bus, master/slaves nodes, bridges, hubs, etc).

Self-Organizing Map (SOM) is an artificial neural network model with competitive and unsupervised training. SOM network has two main characteristics: it makes possible obtaining a simplified model of the training data (normally high dimensional) and it has the capacity to project them on a two dimensional map that shows the existing relations among them. Growing Cell Structure (GCS) is a SOM model proposed in [10] where this static structure limitation of the early models was eliminated. In addition, the flexibility that offers the possibility of inserting and removing neurons on the output layer of the network during the training phase causes that the GCS network receives better value associated to the topology preserving term, understanding this like the grade that defines the quality of the simplified model that the network represents.

GCS networks have demonstrated in [5] to be a useful tool to P-System in the searching of membrane balanced distributions. The feature of simplified model associated to GCS networks allows working with high volumes of membranes where the distribution possibilities go off. In this work, is tested the model with thirty P-System generally used in the literature of P-System. Different membrane distributions between different numbers of processors have been generated for every P-System, observing how the resulting communications of the distribution affect to the parallelization grade. For each data set associated to a concrete P-System diverse GCS networks have trained with the intention of visualize the output layer and establish the optimal distribution, which will be the one that balances the degrees of external communications and parallelization.

In [11] a Self-Organizing Neural Network (SONN) which allows the search and selection of the balanced distribution of membranes for a given P-System is implemented. In [12] this SONN is integrated into a developed IDE enabling the automatic injection of a balanced distribution configuration into the hardware architecture introduced in [2].

3. VARIABLE NEIGHBORHOOD SEARCH (VNS)

Local search methods for combinatorial optimization proceed by performing a sequence of local changes in an initial solution which improve each time the value of the objective function until a local optimum is found. That is, at each iteration an improved solution x' in the neighborhood N(x) of the current solution x is obtained, until no further improvements are found. In VNS a simple and effective metaheuristic is obtained by proceeding to a systematic change of neighborhood within a local search algorithm. VNS is based in the next principles [14]:

Fact 1. A local minimum with respect to one neighborhood structure is not necessary so for another;
Fact 2. A global minimum is a local minimum with respect to all possible neighborhood structures.
Fact 3. For many problems local minima with respect to one or several neighborhoods are relatively close to each other.

Let us denote a finite set of pre-selected neighborhood structures with N_k (k= 1 ..... k_{max}), and with N_k (x) the set of solutions in the k^{th} neighborhood of x. Local search heuristics usually use one neighborhood structure, i.e.. k_{max}= 1. The basic VNS algorithm has this rules:

Initialization. Select the set of neighborhood structures N_k (k= 1 ..... k_{max}), that will be used in the search; find initial solution x;
Main step. (1) Set k: = I.
(2) Until k =k~x, repeat the following steps:
(a) generate a point x' at random from the k^{th} neighborhood of x ($x' \in N_k(x)$>);

(b) apply some local search method with x' as the initial solution; denote with x'' the obtained local optimum;
(c) if the solution thus obtained is better than the incumbent, move there ($x := x''$), and continue the search with N_1 (k:= 1); otherwise, set k: =k+ 1;

These rules are represented graphically by the following figure:

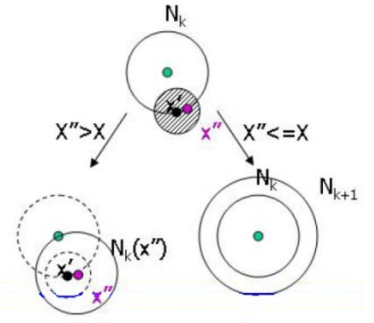

Fig. 2: VNS algorithm structure

GUIDED VNS

In this paper we use an objective function that allows balancing the columns of internal and external communications that occur during the performance of P-System. In particular, there will be a Guided Local Search (GLS) that aims to increase the effectiveness of local search by changing the objective function with penalties.
In GLS, a given objective function g that maps a candidate solution to a numerical value, GLS defines a function h (that replaces g) and that is used by local search as follows:

$$h(s) = g(s) + \lambda \times \sum(p_i \times I_i(s))$$

where S is the candidate solution, λ is a parameter for GLS, i varies over all the attributes, p_i is the penaltie for the i-ésimo attribute (p_i are initialized to 0) and indicates whether it has the i-ésimo attribute as is illustrated below:

$$I_i(S) = \begin{cases} 1 & \text{If } s \text{ has the i-ésimo attributte} \\ 2 & \text{otherwise} \end{cases}$$

For local search may leave optimum locals, GLS adds penalties for certain attributes. The utility to penalize i attribute for a given local optimum $S*$ is:

$$util_i(s^*) = I_i(s^*) \times \frac{c_i}{1 + p_i}$$

where c_i is the cost of the attribute and p_i is the current penalization for i attribute. The most useful attribute is the penalty (the current penalty is increased). The algorithm is shown below.

```
k←0
s0← initial solution is generated
for i=1 until M do
   pi=0
h(s)=g(s) + λ × ∑ (pi × Ii(s))
while criterion do halt do
   sk+1 = local search (sk,h)
   for i=1 until M do
      utili(s*) = Ii (s*)× ci/1+pi
   for each i such that utili is maximun
   do
      pi ← pi +1
   k←k+1
return better solution found
```

4. EXPERIMENTS AND RESULTS

To test the system we have selected the same thirty P-System models used in [5]. Different membrane distributions between different number of processors have been generated for every P-System, observing how the resulting external and internal communications of the distribution affect to the parallelization grade. For each data set associated to a concrete P-System diverse GCS networks was trained with the intention of visualize the output layer and establish the optimal distribution, which will be the one that balances the degrees of external communications and parallelization. In order to present the results of applying GCS has chosen the P-System structure presented in the Fig 1. The neurons are grouped into three criterions: bad (from 1 to 11), medium (from 14 to 18) and good (from 19 to 28). Within each of these three groups the neurons was ordered from highest to lowest level of external communication and for those with a similar value, from highest to lowest level of internal communication. Based on this information has been determined that the best and distributions and therefore the solutions are in the medium region, as is illustrated in Fig. 3.

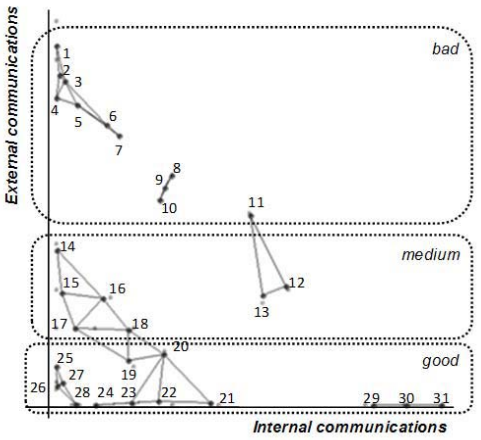

Fig. 3. Left: Scattergram of a GCS network. Points represent neurons and lines between them neighbor connections. Units are grouped in *bad*, *medium* and *good* classes on the basis of the external communication degree.

The elite solutions in the medium region obtained by GCS network are used as the initial neighborhood set (N_k) for the guided VNS algorithm. The selection criterion to application of the algorithm is the same that in the GCS network: to maintain a balanced volume of internal and external communications for a concrete membrane-processor-distribution. The VNS search technique is utilized to implement intelligent random recombination toward elite solutions to locate the optimality regions. The change in structure of environments is made using these rules:

i. The cost of the solution is the sum of the volume of the communications,
ii. attributes are all possible weights of internal and external communications for each membrane ,
iii. the cost of each attribute is its relative distance to the average volume of communications, and
iv. every possible communication is associated with a penalty (initial value =0)

These rules ensure that the resulting solution improves the overall execution times for the P-System and therefore increases the quality of the obtained solution.
The result of the searching process indicates that C13, C15 and C16 distributions are local optimums that can be considered as high-quality solutions as is illustrated in Fig. 4.

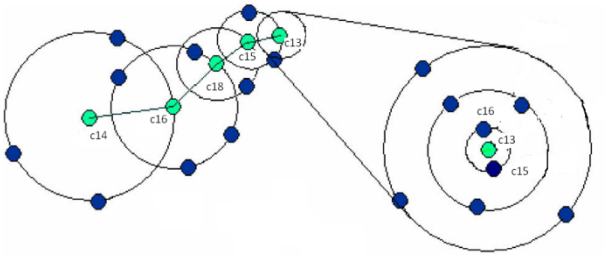

Fig. 4 The shaking trajectory-based search concepts: (a) A distributions center search along the elite solutions.

As we can see in Figures 5 and 6, the combinations internal and external communications load profiles are very similar, which states that all of them can be considered as valid solutions.

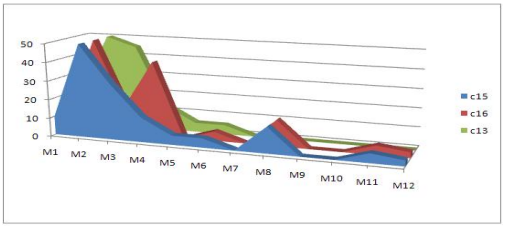

Fig. 5: Distributions results for the external communication degree

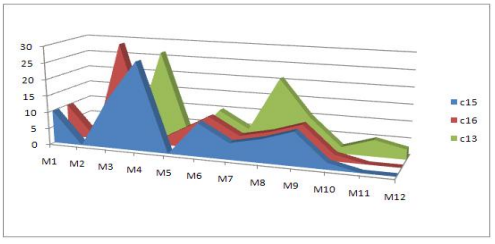

Fig. 6: Distributions results for the internal communication degree

5. CONCLUSIONES AND FUTURE WORK

This paper demonstrates that the combination of using a GCS network for classification of all possible distributions of membrane between processors in a P-System and the subsequent treatment with a VNS search algorithm provides a quick way to obtain the best distributions with a balanced internal and external communications, which implies an improvement in execution times of the P-System.
Although the example that has been used to document the VNS application has a small volume of membranes, the feature of simplified model associated to allows working with high volumes of membranes where the distribution possibilities go off guaranteeing an optimal solution within the set of feasible solutions.
The resulting combinations from the searching process through the VNS algorithm can be exported to the tool implemented in [11] to automate configuration process of the P-System.
Given the good results obtained in the experiments, as future extensions the possibility of working with vectors of greater dimension is considered, what will allow to fit the search of suitable balanced P-System, for example generating a single vector for each membrane distribution or working with fuzzy values for determining the degree of internal and external communications of a processor.

REFERENCES

[1] Păun, Gh., "Computing with membranes". *Journal of Computer and System Sciences*, 61 (2000), and Turku Center for Computer Science-TUCS Report No 208, (1998).
[2] Gutiérrez, A., Fernández, L., Arroyo, F., and Alonso S., "*Suitability of using microcontrollers in implementing new P System communications architectures*", Proceedings of the 13th International Symposium on Artificial Life and Robotics (AROB 13th 08), B-Con Plaza, Beppu -Oita (Japan), no. 31, January 2008.
[3] Bravo, G., Fernández L., Arroyo, F., et al, "*A hierarchical architecture with parallel communication for implementing P-Systems*", ITA 2007 Joint International Scientific Events on Informatics, Varna, Bulgaria (2007).
[4] Bravo, G., Fernández L., Arroyo F., et al, "*Master-Slave Distributed Architecture for Membrane Systems Implementation*", 8th WSEAS Int. Conf. on Evolutionary Computing (EC'07), Vancouver, Canada (2007).
[5] Gutiérrez, A., Delgado, S., Fernández, L, "*Suitability of Using Self-Organizing Neural Networks in Configuring P-System Communications Architectures*", 15th International Conference on Neural Information Processing of the Asia-Pacific Neural Network Assembly.
[6] Kohonen, T., "*Self-Organized Formation of Topologically Correct Feature Maps*", Biological Cybernetics, Vol. 4359--69 (1982).
[7] Mladenović, N., "*A variable neighborhood algorithm - a new metaheuristic for combinatorial optimization*", Abstracts of Papers Presented at Optimization Days, page 112, 1995.
[8] Hansen, P., Mladenović, N., "*An introduction to variable neighborhood search*", Voss et al., editor, Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization, pages 433– 458. Kluwer, 1999.
[9] Tejedor, J., Fernández, L., Arroyo, F., Bravo, G., "*An architecture for attacking the bottleneck communication in P Systems*". M. Sugisaka, H. Tanaka (eds.), *Proceedings of the 12th Int. Symposium on Artificial Life and Robotics*, Beppu, Oita, Japan, 500--505 (2007).
[10] Fritzke, B., "*Growing Cell Structures – A Self-organizing network for Unsupervised and Supervised learning*". Neural Networks, Vol. 7, no.1. 1441--60 (1994).
[11] Gómez, S., Gutiérrez, A., Delgado, S., "*Implementing Self-Organizing Neural Networks for P-Systems Communications Architectures using Microcontrollers*", "4th Indian International Conference on Artificial Intelligence" - IICAI-09, Tumkur – India (Submitted).
[12] Gómez, S., Gutiérrez A., Alonso S,: "*Hardware implementation of P Systems using microcontrollers. An Operating Environment for implementing a Partially Parallel Distributed Architecture*". Proceedings of the 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNACS'08), ISBN No. 978-0-7695-3523-4, pp. 489-495, (2009).
[13] Hakimi-Asiabar M., Ghodsypour, S., Kerachian, R., "*Multi-objective genetic local search algorithm using Kohonen's neural map*", Computers & Industrial Engineering 56 (2009) 1566–1576.
Contents lists
[14] Hansen P., Mladenović N., Moreno J., "*Variable Neighbourhood Search*", Revista Iberoamericana de Inteligencia Artificial. No.19 (2003),pp. 77-92. ISSN: 1137-3601.