DECISION TREES FOR APPLICABILITY OF EVOLUTION RULES IN TRANSITION P SYSTEMS

Luis Fernandez, Fernando Arroyo, Ivan Garcia, Gines Bravo

Abstract: Transition P Systems are a parallel and distributed computational model based on the notion of the cellular membrane structure. Each membrane determines a region that encloses a multiset of objects and evolution rules. Transition P Systems evolve through transitions between two consecutive configurations that are determined by the membrane structure and multisets present inside membranes. Moreover, transitions between two consecutive configurations are provided by an exhaustive non-deterministic and parallel application of active evolution rules subset inside each membrane of the P system. But, to establish the active evolution rules subset, it is required the previous calculation of useful and applicable rules. Hence, computation of applicable evolution rules subset is critical for the whole evolution process efficiency, because it is performed in parallel inside each membrane in every evolution step. The work presented here shows advantages of incorporating decision trees in the evolution rules applicability algorithm. In order to it, necessary formalizations will be presented to consider this as a classification problem, the method to obtain the necessary decision tree automatically generated and the new algorithm for applicability based on it.

Keywords: Decision Tree, ID3, Evolution Rules, Applicability, Transition P System.

ACM Classification Keywords: I.2.6 Learning – Decision Tree; D.1.m Miscellaneous – Natural Computing

Introduction

Membrane computing is a new computational model based on the membrane structure of living cells [Păun, 1998]. This model has become, during last years, a powerful framework for developing new ideas in theoretical computation. Main idea was settled in the base of connecting the Biology with Computer Science in order to develop new computational paradigms.

An overview of membrane computing software can be found in literature, or tentative for hardware implementations [Fernández, 2005], or even in local networks is enough “to understand how difficult is to implement membrane systems on digital devices” [Păun, 2005].

Transition P Systems evolve through transitions between two consecutive configurations that are determined by the membrane structure and multisets present inside membranes. Moreover, transitions between two consecutive configurations are provided by an exhaustive non-deterministic and parallel application of an evolution rules subset inside each membrane of the P system. Evolution rules subset we are studding here will be composed by applicable rules. Moreover, It exist algorithms of application for evolution rules [Fernández, 2006] that, recurrently to its end, need the computation of applicable evolution rules subset. Hence, computing applicable evolution rules is critical for the whole evolution process efficiency, because it is performed in parallel inside each membrane in each one of the evolution steps.

At the present time, computation of applicable evolution rules subset falls on redundancies in a directly or indirectly way. Incorporating decision trees in this computation avoids these redundancies and improves global efficiency of P system evolution.

This work is structured as follows: firstly, evolution rules applicability over a multiset of objects problem is formalized together with its corresponding traditional algorithm. Following section, briefly describes essential elements of decision trees. Afterwards, they are presented new formalizations that permit considering applicability problem as a classification problem solvable through decision trees. In next section, it is presented the algorithm based on decision trees. Finally, efficiency between both algorithms is compared and we expose our conclusions.

Applicability of Evolution Rules

This section defines concepts about multisets, evolution rules and applicability which are needed to follow the developed work presented here. Moreover, it is presented the traditional algorithm, without decision trees, for applicability evolution rules on multisets and its complexity.

From now on, let U be a finite and not empty set of symbols with $|U| = m$.

Let ω be a multiset over U, where ω is a mapping from U to N. Hence, $\omega(u) = p / \forall u \in U \; \exists! \, p \in N$. Let us present the set of all multisets as $\mathcal{M}(U) = \{\omega / \omega \; is \; a \; multiset\}$.

Weight of a symbol $u \in U$ is defined over a multiset $\omega \in \mathcal{M}(U)$ as $\omega(u)$ and it is represented by $|\omega|_u$.

Inclusion of multiset is a binary relation defined as $\omega_1 \subset \omega_2 \Leftrightarrow |\omega_1|_u \leq |\omega_2|_u$, $\forall u \in U \;\; \forall \omega_1, \omega_2 \in \mathcal{M}(U)$.

Any $\omega \in \mathcal{M}(U)$ can be represented as the m-tuple of natural number by the Parikh vector associated to the multiset w with respect to U. The problem is that the Parikh vector representation depends on the order of the elements of U. To avoid this problem, an order over the set U is defined as an ordered succession of symbols through a one to one mapping Φ from $\{1..m\}$ to U that is:

1. $\forall i \in \{1,...,m\} \; \exists u \in U / \Phi(i) = u$
2. $\forall u \in U \; \exists i \in \{1,...,m\} / \Phi(i) = u$
3. $\forall i, j \in \{1,...,m\} / \Phi(i) = \Phi(j) \Rightarrow i = j$

This fact permits us to represent every $\omega \in \mathcal{M}(U)$ as an element of N^m in a congruent manner. Hence,

$\omega = (p_1,..., p_m) \in N^m / |\omega|_u = p_{\Phi(u)} \; \forall u \in U$.

On the other hand, let T be a finite and non empty set of targets.

Evolution rule with symbols in U and targets in T is defined by $r = (a, c, \delta)$ where $a \in \mathcal{M}(U)$, $c \in \mathcal{M}(U \times T)$ and $\delta \in$ {dissolve, not dissolve}. The set of evolution rules is defined as $\mathcal{R}(U,T) = \{r / r \; is \; a \; evolution \; rule\}$.

Antecedent of $r = (a, c, \delta) \in \mathcal{R}(U,T)$ is defined as $input(r) = a$.

Finally, it is said that $r \in \mathcal{R}(U,T)$ is applicable over $\omega \in \mathcal{M}(U)$ if and only if $input(r) \subset \omega$.

Applicability Algorithm. On the one hand, a set of useful evolution rules R and a multiset of objects ω, will be the input to the process. On the other hand, output of process will be R_A, the evolution rules subset of R that are applicable over the multiset. Traditional algorithm [Fernández, 2005] checks weights of each evolution rules antecedent symbol with the corresponding from multiset of objects.

```
(1) R_A ← ø
(2) FOR-EACH r_i IN R DO BEGIN
(3)     j ← 1
(4)     WHILE j ≤ |ω| − 1 AND |input(r_i)|_j ≤ |ω|_j DO
(5)         j ← j + 1
(6)     IF |input(r_i)|_j ≤ |ω|_j THEN
(7)         R_A ← R_A ∪ {r_i}
(8) END
```

Algorithm 1. Evolution rules applicability (without decision trees).

Complexity of algorithm 1 consider, in the worst case, situation in which every evolution rule are applicable over the multiset of objects: loop in (4) will reach as many iterations as symbols exists in U on each iteration of loop (2) to each evolution rule present in R. In the worst case, complexity order will be $O(n)$ being $n = | R | \cdot | U |$

Analysis of previous algorithm will reveal possible redundancies in checks: in a direct and indirect way. So,

- A redundant check in a direct way will occur when weight of a same symbol is equal in more than one evolution rule antecedent, executing several times the same comparison (for example, let be $input(r_1) = (3, 1, 4, 1)$, $input(r_2) = (3, 2, 4, 4)$, and $\omega = (7, 3, 5, 4)$ where comparisons for the fist and third symbol of $input(r_2)$ are redundant in a direct way with its respective symbols in $input(r_1)$).
- A redundant check in an indirect way will occur when, after result of a checking which is false, it will be performed checks between greater weights of that symbol in others evolution rules antecedent (for instance, let it be $input(r_1) = (3, 1, 3, 1)$, $input(r_2) = (5, 2, 5, 1)$, and $\omega = (1, 3, 5, 4)$ where comparison for first symbol of $input(r_2)$ is redundant in an indirect way with its respective symbol in $input(r_1)$).

Furthermore, any checking of the weight of a symbol from an evolution rule antecedent with 0 will be unnecessary because $0 \leq n \ \ \forall n \in N$.

Decision Trees

A decision tree is a tree that permits us to determine the class which one element belongs to, depending on the values that take some attributes of it. Every internal node represents one attribute and edges are possible values of that attribute. Every leaf node in the tree represents one class. So, one unknown element can be classified processing the tree: every internal node studies the value of one attribute for the element and takes the appropriate edge, depending on its value; it continues until a leaf node is reached and, therefore, to the element classification.

E	a_1	...	a_j	...	a_q	C
e_1	v_{11}	...	v_{1j}	...	v_{1q}	c_1
...	...	...	...	...	...	...
e_i	v_{i1}	...	v_{ij}	...	v_{iq}	c_j
...	...	...	...	...	...	...
e_p	v_{p1}	...	v_{pj}	...	v_{pq}	c_s

Figure 1. Example of values table for ID3 algorithm input

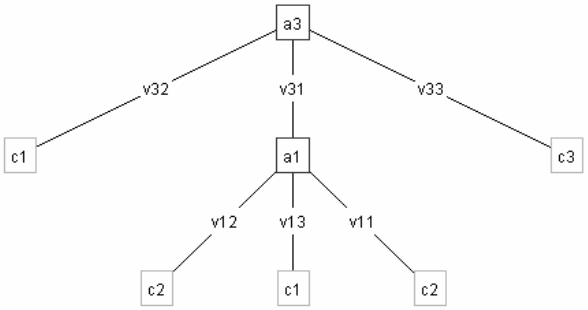

Figure 2. Decision tree generated by ID3 for values table from fig 1.

There are a lot of algorithms to generate decision trees [Rasoul, 1991]. In particular, ID3 algorithm is based on entropy information and it generates an optimum decision tree from a non incremental set of instances and without repetitions.

ID3 algorithm requires as input (Fig. 1): let E be a finite set of instances $\{e_1,...,e_p\}$; let A be a finite set of attributes $\{a_1,...,a_q\}$;let V_j be a finite set of values $\{v_{1j},...,v_{rj}\}$ for each attribute a_j, (where a_j attribute value for instance e_i fulfils $v_{ij} \in V_j$); and finally, let C be a finite set of classes for the classification $\{c_1,...,c_s\}$. On the other hand, ID3 algorithm outputs the optimum decision tree for any element classification (Fig. 2).

Decision Trees for Applicability of Evolution Rules

This section presents evolution rules applicability as a classification problem. This way, it will be posible to design a new algorithm that, being based on a decision tree, avoid direct and indirect redundancies of algorithm 1 presented above.

In order to it, we invert evolution rules applicability problem terms: for a given multiset, we compute the applicable evolution rules subset. Hence, we consider:

- Multisets of objects will be the elements to be classified: $\omega = (p_1,..., p_m) \in \mathcal{M}$(U);
- The set of attributes will be a settled as a set of checks between the objects weights from the multiset and the same object from the evolution rules antecedents having a non null weight. Hence, the finite set of attributes will be:

$$A = \{a \equiv | \omega |_u \geq k \ / \ | input(r) |_u = k \wedge k \neq 0 \ \ \exists r \in R \ \forall u \in U \ \};$$

- Consequently, the finite set of values for every attribute will be true or false, result from comparison relationship between weights.
- Finally, classes to consider will be the different applicable evolution rules subsets. Therefore, the finite set of classes will be: $C = \{c \equiv R_A / \exists R_A \subset R \ \}$.

To obtain automatic generation of decision tree from ID3 algorithm, it will be necessary a non incremental and without repetitions battery of finite instances. In order to it, domain is defined as a set of multisets having the same values for all of their attributes. Consequently, each domain is characterized because every multiset responds to the same applicable evolution rules subset, that is, to the same class. Finally, examples battery will be formed by a representative from each domain.

Fig. 3 shows an example with disjoint domains of multisets of symbols for $U = \{x, y\}$ and rules set: $R = \{r_1, r_2, r_3, r_4\}$ where *their antecedents are:* $r_1 = (y^5)$, $r_2 = (x^2, y^2)$, $r_3 = (x^6, y^2)$, $r_4 = (x^2, y^3)$.

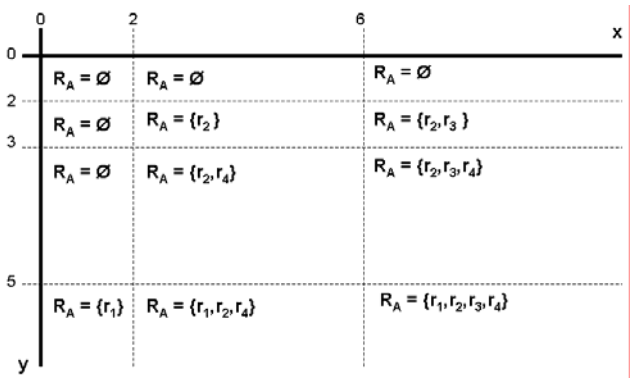

Fig. 3. Disjoint domains of multisets of objects for rules set and its corresponding applicable evolution rules.

Next, they are presented necessary definitions for formalizing the finite set of representative domains that are needed for the generation of decision trees.

It is defined projection of $u \in U$ over $R \subset \mathcal{P}(\mathcal{K}$(U,T)) as:

$$P_u(R) = \{n \in N / \exists r \in R \wedge | input \ r |_u = n\} \subset \mathcal{P}(N)$$

Hyperplane of $d \in \{1,...,m\}$ in $k \in N$ over N^m is defined as:

$$H_d^k(N^m) \rightarrow \{(x_1,..., x_d,..., x_m) / x_d = k\} \subset \mathcal{P}(N^m)$$

Thus, it is considered the grid over $R \subset \mathcal{P}(\mathcal{K}$(U,T)) as:

$$\mathcal{H}(R) = \{h / h = H^k_{\Phi(u)} \ \forall u \in U \wedge \forall k \in P_u(R)\}$$

Moreover, $\mathcal{D}(R)$ is defined as the partition N^m in disjoint subsets formed from every hyperplane of the grid $\mathcal{H}(R)$. It is named domain D to each one of the elements from partition $\mathcal{D}(R)$. Where it is fulfilled:

$$d = | \mathcal{D}(R) | = \prod_{\forall u \in U} | P_u(R) | < \infty$$

$$N^m = \bigcup_{k=1}^{d} D_k \ \wedge \ D_i \cap D_j \neq \emptyset, \forall i, j \in \{1,...,d\} \wedge i \neq j$$

Finally, it is defined representative of $D \in \mathcal{D}(R)$ as

$$\Lambda(D) = \min(\{dist(m,(0,...,0)) \ \forall m \in D\})$$

Fig. 4 shows an example obtained from values of table for the evolution rules set of figure 3. This table includes a row by each representative of the domains, its values for checking relations and applicable evolution rules subset that corresponds to each domain. Fig. 5 shows the classification tree generated by ID3 algorithm for the corresponding figure 4 values table.

Incorporation of decision trees avoids unnecessary null weights comparisons from algorithm 1 because they are not incorporated as in starting instances. Same, direct way redundancies are avoided, the weight of a symbol is compared with the same value just once. Finally, indirect way redundancies are also avoided due to the optimum decision tree ensured by ID3 algorithm, avoiding relations of transitive comparisons.

E	x≥6	x≥2	y≥5	y≥3	y≥2	C
$x^0 y^0$	no	no	no	no	no	Ø
$x^0 y^2$	no	no	no	no	yes	Ø
$x^0 y^3$	no	no	no	yes	yes	Ø
$x^0 y^5$	no	no	yes	yes	yes	$\{r_1\}$
$x^2 y^0$	no	yes	no	no	no	Ø
$x^2 y^2$	no	yes	no	no	yes	$\{r_2\}$
$x^2 y^3$	no	yes	no	yes	yes	$\{r_2,r_4\}$
$x^2 y^5$	no	yes	yes	yes	yes	$\{r_1,r_2,r_4\}$
$x^6 y^0$	yes	yes	no	no	no	Ø
$x^6 y^2$	yes	yes	no	no	yes	$\{r_2,r_3\}$
$x^6 y^3$	yes	yes	no	yes	yes	$\{r_2,r_3,r_4\}$
$x^6 y^5$	yes	yes	yes	yes	yes	$\{r_1,r_2,r_3,r_4\}$

Fig. 4. Examples battery for the evolution rules set of figure 3: in each row there is representative of each domain, values it takes for comparison relations and corresponding applicable evolution rules subset.

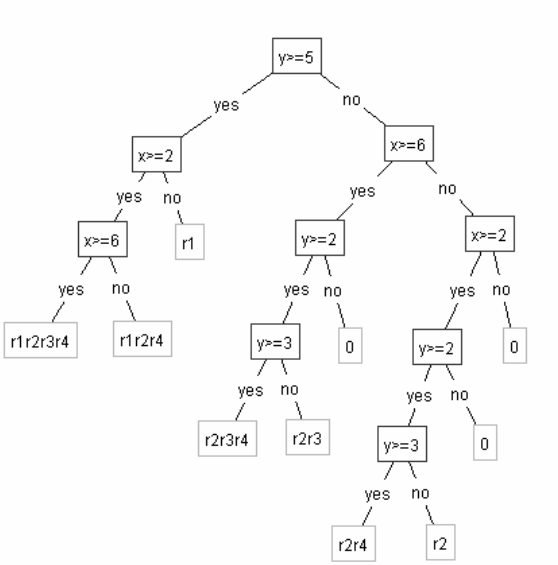

Fig. 5. Example of decision tree generated by ID3 algorithm for the examples battery from figure 4.

Applicablity Algorithm based on Decision Trees

Previously to the algorithm presentation, we will expose the appropriate data structure for supporting the decision tree.

- On the one hand, they are disposed four correlative tables *left*, *symbol*, *value* and *right* for attribute nodes, with one cell in each table by each attribute node; root node is located in position 0 cells;
- On the other, It is disposed a table *classes* for classification nodes, with one cell for each classification node;
- Correlative cells of tables *symbol* and *value* determine which object weight from the multiset of objects has to be compared with which weight. Cells of tables *left* y *right* indicate, whether or not it is respectively accomplished previous relation comparison, which cell is the following attribute node in, whether index is positive; otherwise, indicate which cell of classification nodes table is the solution in.

Figure 6 shows an example of data structures of corresponding generated decision tree from figure 3.

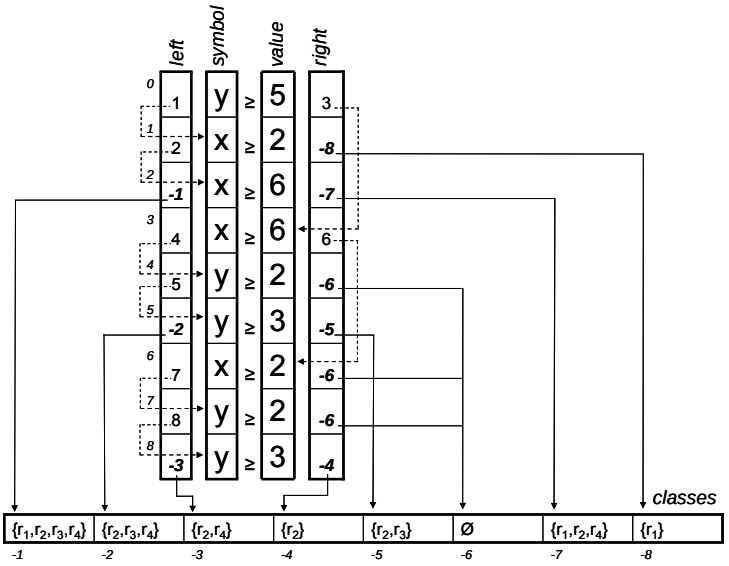

Fig. 6. Data structure for decision tree of figure 5.

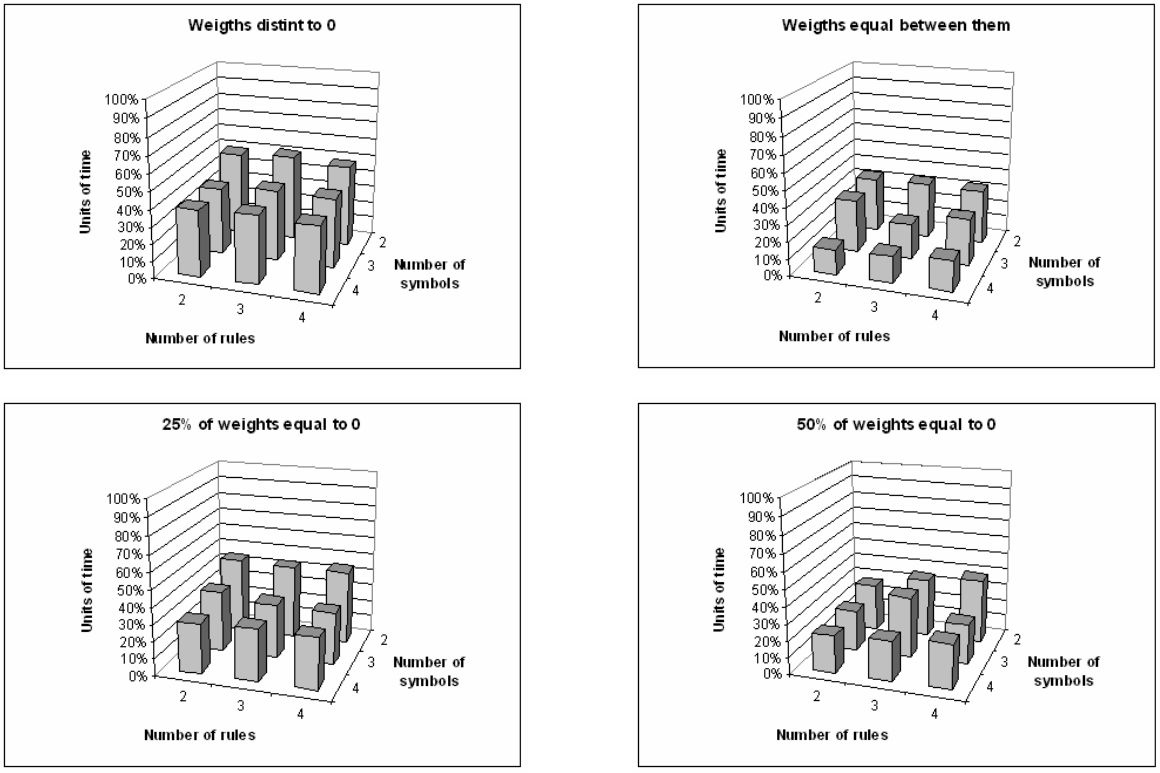

Fig. 7. Execution time reduction carried out by applicability algorithm based on decision tree respect traditional algorithm.

Then, the input to applicability algorithm is ω, multisets of objects, and the supporting decision tree data structure: *left*, *symbol*, *value*, *right* and *classes*. On the other hand, output is A, an evolution rules subset of R that is applicable over that multiset. Following code processes rows of the indexes of branches *left* or *right*, depending on the comparison of symbol indicated by *symbol* weight with established value on *value* until it is reached a classification leaf, indicated by a negative value.

(1) $f \leftarrow 0$
(2) **WHILE** $f \geq 0$ **DO**
(3) **IF** $|\omega|_{\Phi^{-1}(symbol[f])} \geq value[f]$ **THEN**
(4) $f \leftarrow left[f]$
(5) **ELSE**
(6) $f \leftarrow right[f]$
(7) $A \leftarrow applicable[f]$

Algorithm 2. Evolution rules applicability based on decision trees.

In the worst case, complexity of algorithm 2 considers to process the longest branch of the decision tree which length will always be lesser than $n = |R| \cdot |U|$. We will reach this conclusion by reduction to the absurd: in order to the longest branch of the decision tree requires n attribute nodes, It must be carried out the following a) every weight of each symbol in the antecedent of evolution rules is not null and different between them and, b) the d existing domain leads to applicable evolution rules different subsets. That is impossible because, in such circumstances, always exist more than one domain that would lead to the empty set. Specifically, a number of domains equal to $\left(\sum_{i=1}^{|U|} |P_i(R)| \right) - |U| + 1$.

Comparative

This section presents the experimental results obtained from evolution rules applicability using the two algorithms presented here. The test set has been randomly generated and it is composed by 48 different evolution rules sets (composed between 2 and 4 evolution rules composed between 2 and 4 symbols per antecedent), over these tests, it has been calculated the applicability of more than a million of randomly generated symbols multisets.

A first global analysis presents a reduction of execution time of this new algorithm based on decision trees respect to traditional algorithm in an average of 33%, with a variance of 7%.

Particularly, they has been made tests directed to four different situations to analyse the behaviour of new algorithm in extreme cases: with every different weight in antecedents of evolution rules, with every weight of same value, and with presence of 25% and 50% null weights.

In the worst case, with every weight being different between them, it has been reached at least 50% of execution time reduction. With all weights with same value, execution time is reduced to a 15% (for 4 evolution rules with 4 symbols per antecedent). In presence of 25% and 50% of null weights in antecedents of evolution rules, time is reduced to a 35% and a 29%, respectively, always in favor of the new algorithm with decision trees.

Conclusions

This work presents a new approach to the calculus of evolution rules applicability over a symbols multiset. This approach is based on decision trees generated from the set of evolution rules of a membrane. This way, they are avoided unnecessary and redundant checking in a direct or indirect way. Consequently, It is always obtained a lesser complexity than the corresponding traditional algorithm. So, execution time is optimized in the calculation of evolution rules applicability over a symbols multiset. All of this has repercussions in global efficiency of the P System evolution, because applicability calculation is carried out in parallel in each membrane in each evolution step.

Bibliography

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

[Fernández, 2006] L.Fernández, F.Arroyo, J.Castellanos, J.A.Tejedor, I.García, *New Algorithms for Application of Evolution Rules based on Applicability Benchmarks*, BIOCOMP06 International Conference on Bioinformatics and Computational Biology, Las Vegas (USA), july, 2006 (accepted).

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

[Paun, 2005] Gh.Paun, *Membrane computing. Basic ideas, results, applications*, Pre-Proceedings of First International Workshop on Theory and Application of P Systems, Timisoara, Romania, September 26-27, 2005, 1-8

[Rasoul, 1991] S.R.Safavian, D.Landgrebe, *A Survey of Decision Tree Classifier Methodology*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 21, No. 3, pp 660-674, May 1991.

Authors' Information

Luis Fernandez – Natural Computing Group of Universidad Politécnica de Madrid (UPM); Ctra. Valencia, km. 7, 28031 Madrid (Spain); e-mail: setillo@eui.upm.es

Fernando Arroyo – Natural Computing Group of Universidad Politécnica de Madrid (UPM); e-mail: farroyo@eui.upm.es

Ivan Garcia – Natural Computing Group of Universidad Politécnica de Madrid (UPM); e-mail: igarcia@eui.upm.es

Gines Bravo – Natural Computing Group of Universidad Politécnica de Madrid (UPM); e-mail: gines@eui.upm.es

A PARTITION METRIC FOR CLUSTERING FEATURES ANALYSIS

Dmitry Kinoshenko, Vladimir Mashtalir, Vladislav Shlyakhov

Abstract: A new distance function to compare arbitrary partitions is proposed. Clustering of image collections and image segmentation give objects to be matched. Offered metric intends for combination of visual features and metadata analysis to solve a semantic gap between low-level visual features and high-level human concept.

Keywords: partition, metric, clustering, image segmentation.

ACM Classification Keywords: I.5.3 Clustering - Similarity measures

Introduction

There has been a tremendous growth of the image content analysis significance in the recent years. This interest has been motivated mainly by the rapid expansion of imaging on the World-Wide Web, the availability of digital image libraries, increasing of multimedia applications in commerce, biometrics, science, entertainments etc. Visual contents of an image such as color, shape, texture and region relations play dominating role in propagation of feature selection, indexing, user query and interaction, database management techniques. Many systems combine visual features and metadata analysis to solve a semantic gap between low-level visual features and high-level human concept, i.e. there arises a great need in self-acting content-based image retrieval task-level systems.

To search images in an image database traditionally queries ‘ad exemplum’ are used. In this connection essential efforts have been devoted to synthesis and analysis of image content descriptors. However, a user’s semantic understanding of an image is of a higher level than the features representation. Low-level features with mental concepts and semantic labels are the groundwork of intelligent databases creation. Short retrieval time