

Descriptive Complexity of Tissue-Like P Systems with Cell Division

Daniel Díaz-Pernil¹, Pilar Gallego-Ortiz², Miguel A. Gutiérrez-Naranjo²,
Mario J. Pérez-Jiménez², and Agustín Riscos-Núñez¹

¹ Research Group on Computational Topology and Applied Mathematics
Department of Applied Mathematics I
sbdani@us.es

² Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla
Avda, Reina Mercedes s/n, 41012, Sevilla, Spain
pigaor@gmail.com, {magutier,marper,ariscosn}@us.es

Abstract. In this paper we address the problem of describing the complexity of the evolution of a tissue-like P system with cell division. In the computations of such systems the number of (parallel) steps is not sufficient to evaluate the complexity. Following this consideration, Sevilla Carpets were introduced as a tool to describe the space-time complexity of P systems.

Sevilla Carpets have already been used to compare two different solutions of the Subset Sum problem (both designed in the framework of P systems with active membranes) running on the same instance. In this paper we extend the comparison to the framework of tissue-like P systems with cell division.

1 Introduction

Comparing two cellular designs that solve the same problem is not an easy task, as there are many ingredients to be taken into account. Moreover, in the case of P systems where the number of membranes increases along the computation, the problem of describing the complexity of the computational process becomes specially hard. The complexity in *time* (number of parallel cellular steps) of these solutions is polynomial, but it is clear that the time is not the unique variable that we need to consider in order to evaluate the complexity of the process. This fact has been observed previously in the literature of P systems. The first paper related to this issue was [1], where Ciobanu, Păun and Ștefănescu presented a new way to describe the complexity of a computation in a P system. The so-called *Sevilla Carpet* was introduced as an extension of the notion of Szilard language from grammars to the case when several rules are used at the same time.

In [4], the problem was revisited, introducing new parameters for the study of the descriptive complexity of P systems. Besides, several examples of a graphical

representation were provided, and the utility of these parameters for comparing different solutions to a given problem was discussed. In that paper two different solutions of the Subset Sum problem, running on the same instance, were compared by using these parameters.

In this paper we adapt Sevilla Carpets to tissue-like models, in order to describe the complexity of their computations.

Note that given two Sevilla Carpets corresponding to P systems from different models designed to solve a decision problem, we can obtain detailed information about two single computations, but this is not enough to compare the efficiency of the two models in general.

Nonetheless, the numerical parameters obtained from these two Sevilla Carpets can give us some hints to compare the corresponding designs of solutions to the problem.

The paper is organized as follows. In Section 2 we recall the definition of tissue-like P systems with cell division. In Section 3 we revisit the definition of Sevilla Carpets and its associated parameters. Section 4 shows a comparison among two different solutions to the Subset Sum problem in the framework of P systems with active membranes and one solution designed with tissue-like P systems with cell division, all of them running on the same instance. Some final remarks are also provided.

2 Tissue-Like P Systems with Cell Division

Tissue-like P systems with cell division is a well-established P system model presented by Gh. Păun *et al.* in [8]. In this section we briefly recall its main features. The biological inspiration for this model is that alive tissues are not *static* network of cells, since cells are duplicated via mitosis in a natural way.

Formally, a *tissue-like P system with cell division* of degree $q \geq 1$ is a tuple of the form

$$\Pi = (\Gamma, \mathcal{E}, w_1, \dots, w_q, \mathcal{R}, i_0),$$

where:

1. Γ is a finite *alphabet*, whose symbols will be called *objects*.
2. $\mathcal{E} \subseteq \Gamma$.
3. w_1, \dots, w_q are strings over Γ representing the multisets of objects associated with the cells in the initial configuration.
4. \mathcal{R} is a finite set of rules of the following form:
 - (a) *Communication rules*: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \dots, q\}$, $i \neq j$, $u, v \in \Gamma^*$.
 - (b) *Division rules*: $[a]_i \rightarrow [b]_i[c]_i$, where $i \in \{1, 2, \dots, q\}$ and $a, b, c \in \Gamma$.
5. $i_0 \in \{0, 1, 2, \dots, q\}$.

A tissue-like P system with cell division of degree $q \geq 1$ can be seen as a set of q cells (each one consisting of an elementary membrane) labelled by $1, 2, \dots, q$. We will use 0 to refer to the label of the environment, and i_0 denotes the output region (which can be the region inside a cell or the environment).

The communication rules determine a virtual graph, where the nodes are the cells and the edges indicate if it is possible for pairs of cells to communicate directly. This is a dynamical graph, because new nodes can appear produced by the application of division rules.

The strings w_1, \dots, w_q describe the multisets of objects initially placed in the q cells of the system. We interpret that $\mathcal{E} \subseteq \Gamma$ is the set of objects placed in the environment, each one of them in an arbitrary large amount of copies.

The communication rule $(i, u/v, j)$ can be applied over two cells i and j such that u is contained in cell i and v is contained in cell j . The application of this rule means that the objects of the multisets represented by u and v are interchanged between the two cells. Note that either i or j can be equal to 0 and in this case the objects are interchanged between one cell and the environment.

The division rule $[a]_i \rightarrow [b]_i[c]_i$ is applied over a cell i containing object a . The application of this rule divides this cell into two new cells with the same label. All the objects in the original cell are replicated and copied in each of the new cells, with the exception of the object a , which is replaced by the object b in the first one and by c in the other one.

Rules are used as usual in the framework of membrane computing, that is, in a maximally parallel way (a universal clock is considered). In one step, each object in a membrane can only be used for one rule (non-deterministically chosen when there are several possibilities), but any object which can participate in a rule of any form must do it, i.e, in each step we apply a maximal set of rules. This way of applying rules has only one restriction when a cell is divided, the division rule is the only one which is applied for that cell in that step; the objects inside that cell cannot be communicated in that step.

The main features of this model, from the computational point of view, are that cells have no polarizations (the contrary holds in the cell-like model of P systems with active membranes); the cells obtained by division have the same labels as the original cell and if a cell is divided, its interaction with other cells or with the environment is blocked during the mitosis process. In some sense, this means that while a cell is dividing it closes the communication channels with other cells and with the environment.

2.1 Recognizer Tissue-Like P Systems with Cell Division

Complexity classes within Membrane Computing have been usually studied in the framework of *decision problems*. Let us recall that a decision problem is a pair (I_X, θ_X) where I_X is a language over a finite alphabet (whose elements are called *instances*) and θ_X is a total boolean function over I_X .

In order to study the computational efficiency for solving NP-complete decision problems, a special class of tissue P systems with cell division is introduced in [8]: *recognizer tissue P systems*. The key idea of such recognizer systems is the same one as from recognizer P systems with cell-like structure.

Recognizer cell-like P systems were introduced in [10] and they are the natural framework to study and solve decision problems within Membrane Computing, since deciding whether an instance of a given problem has an affirmative or

negative answer is equivalent to deciding if a string belongs or not to the language associated with the problem.

In the literature, recognizer cell-like P systems are associated with P systems with *input* in a natural way. The data encoding to an instance of the decision problem has to be provided to the P system in order to compute the appropriate answer. This is done by codifying each instance as a multiset placed in an *input membrane*. The output of the computation (**yes** or **no**) is sent to the environment, in the last step of the computation. In this way, cell-like P systems with input and external output are devices which can be seen as black boxes, in the sense that the user provides the data before the computation starts, and then waits *outside* the P system until it sends to the environment the output in the last step of the computation.

A recognizer tissue-like P system with cell division of degree $q \geq 1$ is a tuple

$$\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, \dots, w_q, \mathcal{R}, i_{in}, i_0)$$

where

- $(\Gamma, \mathcal{E}, w_1, \dots, w_q, \mathcal{R}, i_0)$ is a tissue-like P system with cell division of degree $q \geq 1$ (as defined in the previous section), $i_0 = env$ and w_1, \dots, w_q strings over $\Gamma \setminus \Sigma$.
- The working alphabet Γ has two distinguished objects **yes** and **no**, present in at least one copy in some initial multisets w_1, \dots, w_q , but not present in \mathcal{E} .
- Σ is an (input) alphabet strictly contained in Γ .
- $i_{in} \in \{1, \dots, q\}$ is the input cell.
- All computations halt.
- If \mathcal{C} is a computation of Π , then either the object **yes** or the object **no** (but not both) must have been released into the environment, and only in the last step of the computation.

The computations of the system Π with input $w \in \Sigma^*$ start from a configuration of the form $(w_1, w_2, \dots, w_{i_{in}} w, \dots, w_q; \mathcal{E})$, that is, after adding the multiset w to the contents of the input cell i_{in} . We say that the multiset w is *recognized* by Π if and only if the object **yes** is sent to the environment, in the last step of the corresponding computation. We say that \mathcal{C} is an *accepting computation* (respectively, *rejecting computation*) if the object **yes** (respectively, **no**) appears in the environment associated to the corresponding halting configuration of \mathcal{C} .

Definition 1. *We say that a decision problem $X = (I_X, \theta_X)$ is solvable in polynomial time by a family $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ of recognizer tissue-like P systems with cell division if the following holds:*

- *The family $\mathbf{\Pi}$ is polynomially uniform by Turing machines, that is, there exists a deterministic Turing machine working in polynomial time which constructs the system $\Pi(n)$ from $n \in \mathbb{N}$.*
- *There exists a pair (cod, s) of polynomial-time computable functions over I_X (called a polynomial encoding of I_X in $\mathbf{\Pi}$) such that:*

- for each instance $u \in I_X$, $s(u)$ is a natural number and $\text{cod}(u)$ is an input multiset of the system $\Pi(s(u))$;
- the family Π is polynomially bounded with regard to (X, cod, s) , that is, there exists a polynomial function p , such that for each $u \in I_X$ every computation of $\Pi(s(u))$ with input $\text{cod}(u)$ is halting and, moreover, it performs at most $p(|u|)$ steps;
- the family Π is sound with regard to (X, cod, s) , that is, for each $u \in I_X$, if there exists an accepting computation of $\Pi(s(u))$ with input $\text{cod}(u)$, then $\theta_X(u) = 1$;
- the family Π is complete with regard to (X, cod, s) , that is, for each $u \in I_X$, if $\theta_X(u) = 1$, then every computation of $\Pi(s(u))$ with input $\text{cod}(u)$ is an accepting one.

In the above definition we have defined every P system $\Pi(n)$ to be *confluent*, in the following sense: every computation of a system with the *same* input multiset must always give the *same* answer.

3 Sevilla Carpets

Sevilla Carpets were presented in [1] as an extension of the Szilard language, which consists of all strings of rule labels describing correct derivations in a given grammar (see e.g., [6,7] or [11]). The Szilard language is usually defined for grammars in the Chomsky hierarchy where only a single rule is used in each derivation step, so a derivation can be represented as the string of the labels of the rules used in the derivation (the labelling is supposed to be one-to-one). Sevilla Carpets are a Szilard-way to describe a computation in a P system. The main difference is that a multiset of rules can be used in each evolution step of a P system. In [1] a bidimensional writing is proposed to describe a computation of a P system. The (Sevilla) Carpet associated with a computation of a P system is a table with the time on the horizontal axis and the rules explicitly mentioned along the vertical axis; then, for each rule, in each step, a piece of information is given. Depending on the amount of information given to describe the evolution, Ciobanu, Păun and Ştefănescu propose five variants for the Sevilla Carpets:

1. Specifying in each time unit for each membrane whether at least one rule was used in its region or not;
2. Specifying in each time unit for each rule whether it was used or not;
3. Mentioning in each time unit the number of applications of each rule; this is 0 when the rule is not used and can be arbitrarily large when the rules are dealing with arbitrarily large multisets;
4. We can also distinguish three cases: that a rule cannot be used, that a rule can be used but it is not because of the nondeterministic choice and that a rule is actually used;
5. A further possibility is to assign a cost to each rule, and to multiply the number of times a rule is used with its cost.

They also propose two parameters (*weight* and *surface*) to study Sevilla Carpets. In [4] two new parameters (*height* and *average weight*) were proposed.

3.1 Parameters for the Descriptive Complexity

Many times we will not be interested only in the number of cellular steps of the computation, but also in other type of resources required to perform the computation. Specially if we want to implement *in silico* a P system, we need to be careful with the number of times that a rule is applied, maybe with the number of membranes and/or the number of objects present in a given configuration.

In order to describe the complexity of the computation, the following parameters are proposed:

- **Weight:** It is defined in [1] as the sum of all the elements in the carpet, i.e., as the total number of applications of rules along the computation. The application of a rule has a cost and the weight measures the total cost of the computation.
- **Surface:** It is the multiplication of the number of steps by the total number of the rules used by the P system. It can be considered as the *potential size* of the computation. From a computational point of view we are not only interested on P systems which halt in a small number of steps, but in P systems which use a small amount of resources. The *surface* measures the resources used in the design of the P system. Graphically, it represents the surface where the Sevilla Carpet lies on.
- **Height:** It is the maximum number of applications of any rule in a step along the computation. Graphically, it represents the highest point reached by the Sevilla Carpet.
- **Average Weight:** It is calculated by dividing the *weight* to the *surface* of the Sevilla Carpet. This concept provides a relation between both parameters which gives an index on how the P system exploits its massive parallelism.

4 Comparing the Solutions

In this section we compare three cellular solutions to the Subset Sum problem: Two of them in the framework of P systems with active membranes and the third one in the framework of tissue-like P systems with cell division.

The Subset Sum problem is the following one: *Given a finite set A , a weight function, $w : A \rightarrow \mathbb{N}$, and a constant $k \in \mathbb{N}$, determine whether or not there exists a subset $B \subseteq A$ such that $w(B) = k$.*

Two uniform families of P systems with active membranes solving the Subset Sum were presented in [4].

A tuple $(n, (w_1, \dots, w_n), k)$ is used in all three solutions to represent an instance of the problem, where n stands for the size of $A = \{a_1, \dots, a_n\}$, $w_i = w(a_i)$, and k is the constant given as input for the problem. All the solutions are based on a brute force algorithm implemented in the framework of P systems with active membranes. The idea of the design can be divided into several stages:

- *Generation stage:* for every subset of A , a membrane is generated via membrane division.

- *Weight calculation stage*: in each membrane the weight of the associated subset is calculated. This stage will take place in parallel with the previous one.
- *Checking stage*: in each membrane it is checked whether or not the weight of its associated subset is exactly k . This stage cannot start in a membrane before the previous ones are over in that membrane.
- *Output stage*: when the previous stage has been completed in all membranes, the system sends out the answer to the environment.

The first family can be found in [9]. Let us recall that the instance $u = (n, (w_1, \dots, w_n), k)$ is processed by the P system $\Pi_1(\langle n, k \rangle)$ with input the multiset $x_1^{w_1} x_2^{w_2} \dots x_n^{w_n}$.

This design depends on the two constants that are given as input in the problem: n and k . It consists of $5n + 5k + 18$ evolution rules, and if an appropriate input multiset is introduced inside membrane e before starting the computation, the system will stop and output an answer in $2n + 2k + 6$ steps (if the answer is *No*) or in $2n + 2k + 5$ steps (if the answer is *Yes*).

The second family is inspired by the previous one. Some modifications were made following the design presented in [3]. In this solution the instance $u = (n, (w_1, \dots, w_n), k)$ is processed by the P system $\Pi_2(n)$ with input the multiset $x_1^{w_1} x_2^{w_2} \dots x_n^{w_n}$.

The above design depends only on one of the constants that are given as input in the problem: n . It is quite similar to the previous one, the difference lies in the checking stage and the answer stage. In this case we avoid the use of counters that require knowing the constant k .

The number of evolution rules is $5n + 41$, and the number of steps of the computation depends on the concrete instance that we need to solve, but it is linearly bounded.

For the study of the Sevilla Carpet in tissue-like P systems with cell division we take the computation of one P system of the family presented in [2]. In that paper a uniform family of tissue-like P systems with cell division solving the Subset Sum problem was presented. The Subset Sum problem was solved in a linear time by a family of recognizer tissue-like P systems with cell division.

We compare these solutions with the solution described in [2], where the number of rules for a set $A = \{a_1, \dots, a_n\}$ of size n is $n \cdot \lceil \log(k + 1) \rceil + 5n + 2 \lceil \log(k + 1) \rceil + 3 \lceil \log n \rceil + 26 \in O(n \cdot \log k)$ and the number of steps is $n + \lceil \log n \rceil + \lceil \log(k + 1) \rceil + 10$ if the answer is **Yes**, and $n + \lceil \log n \rceil + \lceil \log(k + 1) \rceil + 12$ if the answer is **No**.

4.1 Descriptive Complexity

We present some detailed statistics about the previous designs, trying to compare them on a more general basis than just looking the number of steps that the computation performs. Following this scheme, we present the Sevilla Carpets associated with the computations of the three different solutions to the Subset Sum problem working on the same instance: $u = (6, (2, 3, 4, 8, 14, 5), 15)$. That

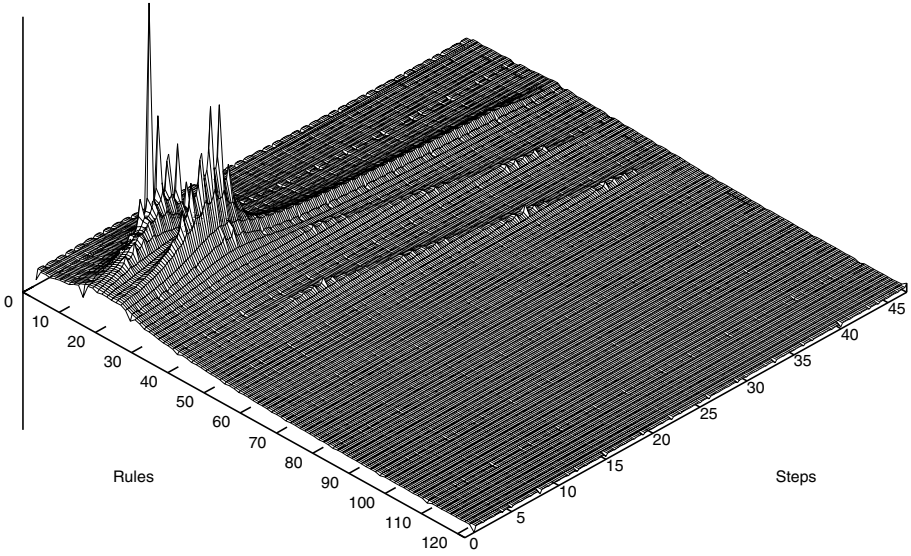


Fig. 1. Sevilla Carpet for solution 1

is, $n = 6$, $k = 15$ and the list of weights is $w_1 = 2, w_2 = 3, w_3 = 4, w_4 = 8, w_5 = 14, w_6 = 5$.

The P system $\Pi_1(\langle 6, 15 \rangle)$ has 123 evolution rules. The P system $\Pi_1(\langle 6, 15 \rangle)$ stops at step 46 and sends an object Yes to the environment.

The weight of the Sevilla Carpet (the total number of rule applications along the computation) is 7597. The height of the Sevilla Carpet (the maximal number of times that a rule is applied in one evolution step) is 281 and it is reached at Step 10. The surface of the Sevilla Carpet is 5658. The average weight of the Sevilla Carpet is 1.3427.

The P system $\Pi_2(6)$ has 66 evolution rules. The P system $\Pi_2(6)$ stops at step 51 and sends an object Yes to the environment.

	Cell-like		Tissue-like
	Sol. 1	Sol.2	Sol. 3
Rules	123	66	97
Steps	46	51	24
Surface	5658	3366	2328
Weight	7597	9491	6592
Height	281	281	755
Average Weight	1.3427	2.8197	2.8316

The weight of the Sevilla Carpet is 9491. The height of the Sevilla Carpet is 281 and it is reached at step 10. The surface of the Sevilla Carpet is 3366. The average weight of the Sevilla Carpet is 2.8197.

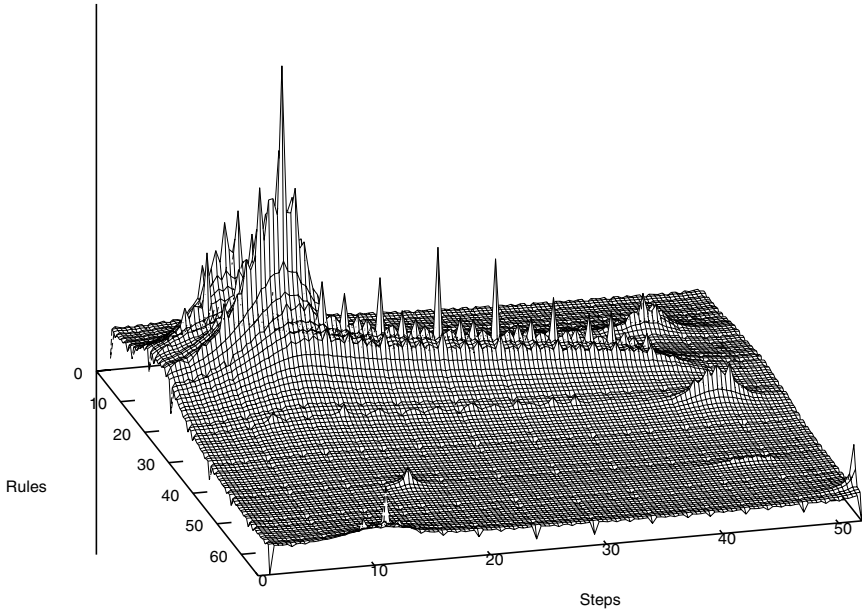


Fig. 2. Sevilla Carpet for solution 2

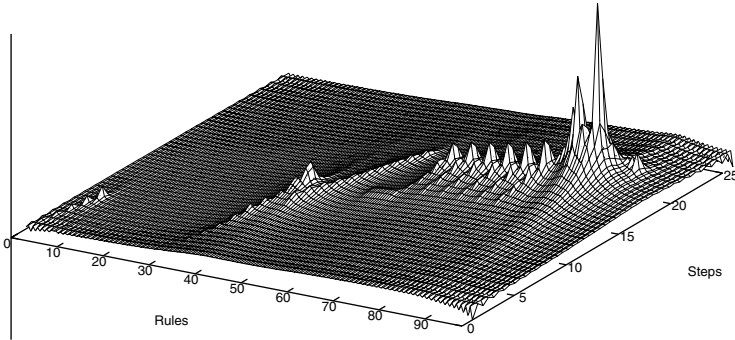


Fig. 3. Sevilla Carpet for solution 3

Finally, the solution with tissue-like P systems with cell division has 97 rules and 95 of them are applied in this computation. The P system stops at step 24. The surface of the Sevilla Carpet is 2328 and its weight is 6592. The height is 755 and it is reached at step 18. The average weight of the Sevilla Carpet is 2.8316.

The following table shows the parameters of all three solutions:

As far as the number of rules is concerned, we can observe that the second solution is the most *compact*. From a designer point of view, it is interesting

to get designs as “short” as possible, minimizing the number of rules of the P systems.

If we consider the number of (parallel) steps as a complexity measure to compare the designs, then we conclude that the third solution is *better* than the other ones, since it needs less steps.

Another relevant computational parameter is the weight of the Sevilla Carpet, that measures the total number of rules that have been applied along the computation. If we assign a *cost* to each application of a rule, then we can say that Solution 2 is the most *expensive*, while Solution 3 is the *cheapest*.

Nonetheless, the key point of a design of a solution in Membrane Computing is the use of the massive parallelism. As pointed out in [5],

a bad design of a P system consists of a P system which does not exploit its parallelism, that is, working as a sequential machine: in each step only one object evolve in one membrane whereas the remaining objects do not evolve. On the other hand, a good design consists of a P system in which a huge amount of objects are evolving simultaneously in all membranes. If both P systems perform the same task, it is obvious that the second one is a better design than the first one.

In this line, the fact that the average weight of Solution 1 is smaller than the average weights of Solutions 2 and 3 can be interpreted saying that the latter designs make a better use of the parallelism in P systems than the first one.

To sum up, according to the parameters corresponding to the computations of the three solutions running on the chosen instance, Solutions 2 and 3 are similar regarding the average weight, but the tissue-like approach is overall the best choice.

5 Conclusions and Future Work

It is important to remark that these are not asymptotical comparisons, as we focus only on the data corresponding to one instance. Indeed, due to the exponential number of membranes created during the generation stage, we believe that considering another instance with a greater size will stress the differences between the design based only on n and the other one, based on n and k . The bound on the size of the instances that can be studied is imposed by the necessity to use a P systems simulator to obtain the detailed description of the computation: number of rules, number of cellular steps, and number of times that the rules are applied in each step.

Acknowledgments

The authors acknowledge the support of the project TIN2006-13425 of the Ministerio de Educación y Ciencia of Spain, cofinanced by FEDER funds, and the support of the Project of Excellence with *Investigador de Reconocida Valía* of the Junta de Andalucía, grant P08-TIC-04200.

References

1. Ciobanu, G., Păun, G., Ștefănescu, G.: Sevilla Carpets Associated with P Systems. In: Cavaliere, M., Martín-Vide, C., Păun, G. (eds.) Proceedings of the Brainstorming Week on Membrane Computing, Tarragona, Spain, Report RGML 26/03, pp. 135–140 (2003)
2. Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Solving Subset Sum in Linear Time by Using Tissue P Systems with Cell Division. In: Mira, J., Álvarez, J.R. (eds.) IWINAC 2007. LNCS, vol. 4527, pp. 170–179. Springer, Heidelberg (2007)
3. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Towards a programming language in cellular computing. *Electronic Notes in Theoretical Computer Science* 123, 93–110 (2005)
4. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: On Descriptive Complexity of P Systems. In: Mauri, G., Păun, G., Jesús Pérez-Jiménez, M., Rozenberg, G., Salomaa, A. (eds.) WMC 2004. LNCS, vol. 3365, pp. 320–330. Springer, Heidelberg (2005)
5. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: On the Degree of Parallelism in Membrane Systems. *Theoretical Computer Science* 372(2-3), 183–195 (2007)
6. Mäkinen, E.: A Bibliography on Szilard Languages, Dept. of Computer and Information Sciences, University of Tampere, <http://www.cs.uta.fi/reports/pdf/Szilard.pdf>
7. Mateescu, A., Salomaa, A.: Aspects of Classical Language Theory. In: Rozenberg, G., Salomaa, A. (eds.) *Handbook of Formal Languages*, vol. 1. Springer, Heidelberg (1997)
8. Păun, G., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Tissue P System with cell division. *International Journal of Computers, Communications & Control* 3(3), 295–303 (2008)
9. Pérez-Jiménez, M.J., Riscos-Núñez, A.: Solving the Subset Sum Problem by Active Membranes. *New Generation Computing* 23(4), 367–384 (2005)
10. Pérez-Jiménez, M.J., Romero-Jiménez, A., Sancho-Caparrini, F.: A polynomial complexity class in P systems using membrane division. In: Csuhaj-Varjú, E., Kintala, C., Wotschke, D., Vaszyl, G. (eds.) Proceedings of the 5th Workshop on Descriptive Complexity of Formal Systems, DCFS 2003, pp. 284–294 (2003)
11. Salomaa, A.: *Formal Languages*. Academic Press, New York (1973)