Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 372768, 6 pages
http://dx.doi.org/10.1155/2014/372768

Research Article

Solving Multidimensional 0-1 Knapsack Problem with Time-Free Tissue P Systems

Xiangrong Liu,[1,2] Ziming Li,[1,2] Juan Suo,[1,2] Ying Ju,[1] Juan Liu,[3] and Xiangxiang Zeng[1]

[1] *Department of Computer Science, Xiamen University, Xiamen 361005, China*
[2] *Shenzhen Research Institute of Xiamen University, Shenzhen 518057, China*
[3] *Department of Mechanical and Electrical Engineering, Institute of Physical and Mechanical and Electrical Engineering, Xiamen University, Xiamen 361005, China*

Correspondence should be addressed to Xiangxiang Zeng; xzeng@xmu.edu.cn

Received 4 December 2013; Revised 2 March 2014; Accepted 2 March 2014; Published 27 March 2014

Academic Editor: Ying Hu

Tissue P system is a class of parallel and distributed model; a feature of traditional tissue P system is that the execution time of certain biological processes is very sensitive to environmental factors that might be hard to control. In this work, we construct a family of tissue P systems that works independently from the values associated with the execution times of the rules. Furthermore, we present a time-free efficient solution to multidimensional 0-1 knapsack problem by timed recognizer tissue P systems.

1. Introduction

Membrane computing is one of the recent branches of natural computing, which has developed rapidly (already in 2003, ISI considered membrane computing as fast emerging research area in computer science; see http://esi-topics.com/). The aim is to abstract computing ideas (data structures, operations with data, ways to control operations, computing models, etc.) from the structure and the functioning of a single cell and from complexes of cells, such as tissues and organs including the brain. The various types of membrane systems are known as P systems after Păun who first conceived the model in 2000 [1] (the paper was circulated first as a Turku Center for Computer Science (TUCS) Report 208, 1998). There are three main classes of P systems investigated: cell-like P systems [2, 3], tissue-like P systems [4–6], and neural-like P systems [7, 8]. Many variants of these systems have been also investigated during the past years [9–14], and most of them are proved to be computationally universal (equal in power to Turing machines) [8, 15–17]. A series of applications of P systems, in biology [11, 18–22], economics [23], graphics [24, 25], optimization [26–30], fault diagnosis [31], and so forth, were reported. An overview of the field can be found in [32, 33], with up-to-date information available at the membrane computing website (http://ppage.psystems.eu/).

Tissue P systems are inspired from the cell intercommunication in tissues [4], where the membranes are placed in the nodes of a graph and all of the membranes are at the same level. In such a framework, intercellular communication is mainly through the protein channels established among the membranes of the neighboring cells and the membrane structure did not change along the computation. Tissue P systems with cell division rules of the same form as in P systems with active membranes (without using polarizations) were introduced by Păun et al. in 2008 [34]. This kind of systems can generate exponential workspace in linear time by trading space for time. Thus, it provides the possibility of solving NP-hard problems in polynomial or even linear time.

A feature of traditional tissue P systems is that each rule is executed in exactly one time unit, but this feature does not have a counterpart in cell biology. In cell biology, the execution time of a reaction is difficult to know precisely and the execution time of certain biological processes is very sensitive to environmental factors that might be hard to control. Therefore, there is a need to construct tissue P systems that work independently from the values associated with the execution time of the rules. Tissue P systems with a time mapping (called timed tissue P systems) can specify the execution time of each rule. A timed tissue P system that generates the same

results, independently from the time mapping, is called time-free tissue P system. The concept of time-free was first introduced in [35]. In [36], a time-free solution to Hamilton Path Problems using cell-like P systems was investigated. In this work, we construct a family of time-free tissue P systems to solve another well-known NP-complete problem: the multidimensional 0-1 knapsack problem.

This paper is organized as follows. Firstly, we recall the definition of tissue P systems. Then, we introduce timed tissue P systems with cell division. Finally, a uniform time-free solution for multidimensional 0-1 knapsack problem is proposed, and the informal verification of the solution is presented.

2. Tissue P Systems with Cell Division

It is useful for readers to have some familiarity with (basic elements of) language theory, for example, from [37], as well as basic membrane computing [32]. In what follows we briefly recall some concepts of formal language theory used later in this paper.

For an alphabet V, V^* denotes the set of all strings (ordered finite sequences) of symbols from V, while the empty string is denoted by λ, and the set of all nonempty strings over V is denoted by V^+.

By $\mathbb{N}$ we denote the set of nonpositive integers. Let U be an arbitrary set. A multiset (over U) is a mapping $M : U \to \mathbb{N}$. The multiplicity of a in the multiset M can be denoted by $M(a)$ with any $a \in U$. It can be expressed by the pair $(a, M(a))$. If the set $U = \{a_1, a_2, \ldots, a_n\}$ is finite, a multiset M over U, represented by the set of mappings $\{(a_1, M(a_1)), (a_2, M(a_2)), \ldots, (a_n, M(a_n))\}$ can also be represented by a string $w = a_1^{M(a_1)} a_2^{M(a_2)} \cdots a_n^{M(a_n)}$ or by any of its permutations. We denote by $M(U)$ the set of all multisets over U. In [34], Păun et al. presented a new model of tissue P systems with cell division. The biological inspiration is that alive tissues are not static network of cells, since cells are duplicated via mitosis in a natural way.

Formally, a *tissue P system with cell division* of degree $q \geq 1$ is a tuple of the form [34]

$$\Pi = \left(\Gamma, \Omega, w_1, \ldots, w_q, R, i_{\text{out}}\right), \tag{1}$$

where

(1) $q \geq 1$ is the initial degree of the system, which means the system contains q cells labeled with $1, 2, \ldots, q$, respectively. We use 0 to refer to the environment;

(2) Γ is the *alphabet* of *objects*;

(3) $w_1, \ldots, w_q$ are multisets over Γ, describing the objects placed in the cells of the system at the beginning of the computation;

(4) $\Omega \subseteq \Gamma$ is the set of objects initially located in the environment of the system, all of them available in an arbitrary number of copies;

(5) R is a finite set of rules of the following forms:

(a) *communication rules*: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \ldots, q\}$, $i \neq j$, $i \neq i_{\text{out}}$, $u, v \in M(\Gamma)(|u| + |v|$ is called the length of the communication rule $(i, u/v, j)$);

(b) *division rules*: $[a]_i \to [b]_i[c]_i$, where $i \in \{1, \ldots, q\}$, $i \neq i_{\text{out}}$, $a \in \Gamma$, and $b, c \in \Gamma \cup \{\lambda\}$;

(6) $i_{\text{out}} \in \{0, 1, 2, \ldots, q\}$ is the output cell if $i_{\text{out}} \neq 0$ or the environment if $i_{\text{out}} = 0$.

Rules are used as usual in the framework of membrane computing, that is, in a maximally parallel way. In each step, all cells which can evolve must evolve in a maximally parallel way (in each step we apply a multiset of rules which is maximal, no further rule can be added), with the following important mentioning: if a cell is divided, then the division rule is the only one which is applied for that cell in that step, its objects do not evolve by means of communication rules. Their labels precisely identify the rules which can be applied to them.

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. When applying a rule $(i, u/v, j)$, the objects of the multiset represented by u are sent from cell i to cell j and simultaneously the objects of the multiset v are sent from cell j to cell i.

The division rule $[a]_i \to [b]_i[c]_i$ can be applied over a cell i containing object a. When applying a rule $[a]_i \to [b]_i[c]_i$, the cell with label i is divided into two cells with the same label; in the first copy the object a is replaced by b, in the second copy the object a is replaced by c; all other objects are replicated and copies of them are placed in the two new cells.

A configuration of Π at an instant t is described by the multisets of objects over Γ associated with all the cells present in the system at the moment and the multiset over $\Gamma \setminus \Omega$ associated with the environment at the instant t. The computation starts from the initial configuration and proceeds as defined above; only halting computations give a result, and the result is encoded (usually by the number of objects) in the output region i_{out} (a cell if $i_{\text{out}} \in \{1, \ldots q\}$ or the environment if $i_{\text{out}} = 0$) in the halting configuration. The set of numbers computed in this way by the various halting computations in Π is denoted by $N(\Pi)$.

3. Timed Tissue P Systems with Cell Division

A timed tissue P system can be constructed by adding to the tissue P system a time mapping $e : R \to \mathbb{N} \setminus \{0\}$, which specifies the execution time of each rule of the system. We suppose to have an external clock (that does not have any influence on the system) that marks time units of equal length, starting from time 0. In each step of the rules, all cells which can evolve must evolve in a maximally parallel way. There is a need to define the concept of a valid step. If in one step there exists at least one rule of system $\Pi(e)$ which can start, we say that the step is *valid*. The execution of the rules does not take the same time unit. When a rule r is started, the occurrences of objects used by this rule are not available for other rules during the entire execution of r. We denote $e(r)$ as the time which rule r lasts, that is, execution of rule r takes $e(r)$ time units to be completed. The computation stops when no rule can be applied in any membrane and there are no rules in execution (*the systems has reached a halting configuration*). The output of a halting computation is always

defined based on the objects in the output membrane in the halting configuration.

We can construct a family of timed tissue P systems by adding different time mappings. The same family of tissue P systems may produce different results for having different time mappings. A timed tissue P system that generates the same computation result, independently from any time mapping, is called time-free tissue P system.

3.1. A Timed Recognizer Tissue P System. In this subsection, we introduce a variant of timed tissue P systems with cell division, namely, timed recognizer tissue P system following the definitions of complexity classes in terms of membrane computing (see [38]). Such a system of degree $q \geq 1$ has the form

$$\Pi(e) = \left(\Gamma, \Sigma, \Omega, w_1, \ldots, w_q, R, i_{\text{in}}, i_{\text{out}}, e\right). \quad (2)$$

There are two points that should be noted. First, the working alphabet Γ has two distinguished objects *yes* and *no*, present in at least one copy in some initial multisets $w_1, \ldots, w_q$ but not present in Ω. Second, if $\mathscr{C}$ is a computation of Π, then either the object *yes* or the object *no* (but not both) must exist in the environment when the computation halts.

The computations of the system Π with input multiset $w \in M(\Gamma)$ start from a configuration of the form $(w_1, w_2, \ldots, w_{\text{in}}w, \ldots, w_q, \emptyset)$, 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, and only at the last step of the corresponding computation. We say that $\mathscr{C}$ is an accepting computation (resp., rejecting computation) if the object *yes* (resp., *no*) appears in the environment associated with the corresponding halting configuration of $\mathscr{C}$.

Next, we will define the concepts of time-free soundness, time-free completeness, and time-free polynomially bounding for timed recognizer tissue P systems. Let $\prod = \{\Pi(n, e) \mid n \in \mathbb{N}\}$ be a family of timed recognizer tissue P systems. There exists a pair (cod, s) of polynomial-time computable functions over X such that

(i) for each instance $u \in I_x$, $s(u)$ is a natural number and $\text{cod}(u)$ is an input multiset of the system $\Pi(s(u))$;

(ii) the family $\prod$ is *time-free sound* with regard to (X, cod, s); that is, for each instance of the problem $u \in I_x$ and any mapping e, if there exists an accepting computation of the time-free system $\Pi(s(u), e)$ (this means $\Pi(s(u))(e)$) with input $\text{cod}(u)$, then we have $\Theta_x(u) = 1$;

(iii) the family $\prod$ is *time-free complete* with regard to (X, cod, s); that is, for each $u \in I_x$ and any mapping e, if $\Theta_x = 1$, then every computation of $\Pi(s(u), e)$ with input $\text{cod}(u)$ is an accepting one;

(iv) the family $\prod$ is *time-free polynomial bounded* with regard to (X, cod, s), that is, there exists a polynomial function p, such that for each $u \in I_x$ and any time mapping e, every computation of the time-free system $\Pi(s(u), e)$ with input $\text{cod}(u)$ is halting and, moreover, it performs at most $p(|u|)$ steps.

A family $\prod = \{\Pi(s(u), e) \mid u \in I_x\}$ of timed recognizer tissue P systems is a uniform time-free solution to decision problem $X = (I_X, \theta_X)$ if the following holds:

(i) the family $\prod$ is *polynomially uniform* by Turing machines; that is, there exists a deterministic Turing machine working in polynomial time which constructs the system $\Pi(e)$ with knowledge involving only the size of the problem X for every instance of X;

(ii) the family $\prod$ is time-free polynomially bounded, time-free sound, and time-free complete.

Given a decision problem X, if there exists a family of timed P system $\prod = \{\Pi(s(u), e) \mid u \in I_x\}$ which computes a solution of the instance u of X and the correctness of the solution does not depend on the execution time of the rules involved in the computation, then the solution is called a time-free solution to problem X computed by the system $\prod$.

4. Solving Multidimensional 0-1 Knapsack Problem by Time-Free Recognizer Tissue P Systems

4.1. Problem Formulation. The *multidimensional 0-1 knapsack problem* (MKP) is a well-known NP-complete combinatorial problem [39]. The decision MKP can be formulated as follows: given an integer k, an objective function $f(x_1, \ldots, x_n) = \Sigma_{i=1}^{n} c_i x_i$, and constraints $\Sigma_{i=1}^{n} w_{ij} x_i \leq b_j$, for $j = 1, \ldots, m$, $x_i \in \{0, 1\}$, for $i = 1, \ldots, n$, where c_i, w_{ij}, and b_j are nonnegative integers, decide whether or not there exists an assignment of variables x_i such that it satisfies the constraints and the objective function is greater than or equal to k.

The special case of MKP with $m = 1$ is the classical knapsack problem (KP). In [40], a solution to MKP by recognizer P systems with active membranes has been proposed. In the next subsection, we will construct a uniform time-free tissue P system for the general MKP.

4.2. A Uniform Time-Free Solution to MKP. We present a uniform time-free solution to MKP in the framework of timed recognizer tissue P systems.

Let us consider an instance of MKP as shown in the above subsection, we call $\Sigma_{i=1}^{n} w_{ij} x_i \leq b_j$ $(1 \leq j \leq m)$ the jth constraint inequality and $\Sigma_{i=1}^{n} c_i x_i \geq k$ the $(m+1)$th inequality. We construct a family of timed recognizer tissue P system of degree 2

$$\Pi(\langle n, m \rangle, e) = \left(\Gamma, \Omega, w_1, w_2, R, i_{\text{in}}, i_{\text{out}}, e\right), \quad (3)$$

where

(i) $\Gamma = \{X_i, Y_i, N_i \mid 1 \leq i \leq n\} \cup \{a_{ijh} \mid 1 \leq i \leq n, 1 \leq j \leq m, 1 \leq h \leq w_{ij}$ (only when $w_{ij} \geq 1)\} \cup \{b_{lj} \mid 1 \leq j \leq m, 0 \leq l \leq b_j + 1\} \cup \{c_{ih} \mid 1 \leq i \leq n, 1 \leq h \leq c_i\} \cup \{k_s \mid 0 \leq s \leq k\} \cup \{p_1, p_2, p_3, \text{yes}, \text{no}\}$;

(ii) $\Omega = \Gamma - \{\text{yes}, \text{no}\}$;

(iii) $w_1 = \{\text{yes}, \text{no}\}$, $w_2 = \{X_1, \ldots, X_n, p_1, p_2\} \cup \{a_{ijh} \mid 1 \leq i \leq n, 1 \leq j \leq m, 1 \leq h \leq w_{ij}\} \cup \{c_{ih} \mid 1 \leq i \leq n, 1 \leq h \leq c_i\} \cup \{b_{0j}, k_0 \mid 1 \leq j \leq m\}$;

(iv) $i_{\text{in}} = 2$ is the input cell;

(v) the output region i_{out} is the environment;

(vi) $e : R \to \mathbb{N}$ is a time mapping from R to natural numbers;

(vii) R is the set of rules:

(1) $r_{1,i} \equiv [X_i]_2 \to [Y_i]_2[N_i]_2$, for $1 \le i \le n$;

(2) $r_2 \equiv (1, \text{no}/\lambda, 0)$;

(3) $r_{3,ijhl} \equiv (2, Y_i a_{ijh} b_{lj}/Y_i b_{l+1\,j}, 0)$, for $1 \le i \le n$, $1 \le j \le m$, $1 \le h \le w_{ij}$ (only when $w_{ij} \ge 1$), $0 \le l \le b_j$;

(4) $r_{4,ihs} \equiv (2, Y_i c_{ih} k_s/Y_i k_{s+1}, 0)$, for $1 \le i \le n$, $1 \le h \le c_i$, $0 \le s \le k-1$;

(5) $r_{5,j} \equiv (2, b_{b_j+1\,j} p_j/\lambda, 0)$, for $1 \le j \le m$;

(6) $r_6 \equiv (2, k_k/p_{m+1}, 0)$;

(7) $r_7 \equiv (2, p_1 p_2 \cdots p_{m+1}/\text{yes}, 1)$;

(8) $r_8 \equiv (2, \text{yes}/\text{no}, 0)$.

In what follows, a detailed explanation of how this system $\Pi(\langle n, m\rangle, e)$ works to MKP is presented. There are three stages in the computation process: generating, checking, and outputting.

In the generating stage, rules $r_{1,i}$ and r_2 start at the same time. By using the rule $r_{1,i}$, the system generates 2^n membranes labeled by 2 after $e(r_{1,i})$ time units. It takes n valid steps to complete rule $r_{1,i}$. When the generating stage is ended, every membrane labeled by 2 represents one assignment of variables x. Object Y_i means that $x_i = 1$, and N_i means 0. Object *no* can be ejected to the environment after $e(r_2)$ steps by the rule r_2.

In the checking stage, the rules $r_{3,ijhl}$, $r_{4,ihs}$, $r_{5,j}$, and r_6 are started in the same time and in the nondeterministic maximally parallel manner. Let us explain how these rules work. The rule $r_{3,ijhl}$ means that if Y_i is in one membrane with label 2, object a_{ijh} will be sent to the environment. The range of subscript h is from 0 to w_{ij} (the case of $w_{ij} = 0$ does not have any influence on the systems) and the subscript j means the jth constraint. In the same time, object b_{lj} is replaced by object $b_{l+1\,j}$. Rule $r_{3,ijhl}$ will stop when there are no objects a_{ijh} in all the membranes with label 2 or there are no objects b_{lj} in the environment. The rule $r_{3,ijhl}$ is used to check the jth constraint inequality $\Sigma_{i=1}^n w_{ij} x_i \le b_j$ $(1 \le j \le m)$. Similarly, the rule $r_{4,ihs}$ is in charge of the checking of the $(m+1)$th inequality $\Sigma_{i=1}^n c_i x_i \ge k$. If there appears object $b_{b_j+1\,j}$ in the membrane, the jth constraint inequality $\Sigma_{i=1}^n w_{ij} x_i \le b_j$ $(1 \le j \le m)$ is not satisfied and object p_j will be sent to the environment by using rule $r_{5,j}$. On the contrary, the rule r_6 means that if there exists object k_k, the $(m+1)$th inequality is satisfied and object p_{m+1} will be sent to the membrane.

In the last stage, outputting stage, rules r_7 and r_8 are used to send the right answer to the environment. After checking stage, if objects $p_1, \ldots, p_{m+1}$ are all in the membrane, we can come to a conclusion that the assignment of variables x in this membrane satisfies the constraints and the objective function is greater than or equal to k. By using rule r_7, objects $p_1, \ldots, p_{m+1}$ are replaced by object *yes*. This process needs one valid step. By using rule r_8, object *yes* will be sent to the environment to exchange for object *no*. The system halts with object *yes* in the environment, which means that the answer to the decision problem is positive. On the contrary, if some of the objects p_j $(1 \le j \le m+1)$ are not in the membrane, it means that this assignment of x does not meet the requirements and object *yes* will remain in the membrane labeled by 1, object *no* will exist in the environment when the computation halts. In this case, the answer to the decision problem is negative.

4.3. Verification. Next, we prove that the family of timed recognizer tissue P systems built above is a uniform time-free solution to MKP.

First, according to the definition of uniform time-free solution, we need to show that the family $\Pi(\langle n, m\rangle, e)$ is polynomially uniform by deterministic Turing machines. It is easy to check that the rules of the system $\Pi(\langle n, m\rangle, e)$ of the family are defined recursively from the values n and m. Besides, the necessary resources to build an element of the family are of a polynomial order, as shown below.

(i) Size of the alphabet: $3n + \Sigma_{j=1}^m \Sigma_{i=1}^n w_{ij} + \Sigma_{i=1}^n c_i + \Sigma_{j=1}^m b_j + 3m + k + 4 \in \Theta(nm)$.

(ii) Initial number of cells: $2 \in \Theta(1)$.

(iii) Initial number of objects: $n + \Sigma_{j=1}^m \Sigma_{i=1}^n w_{ij} + \Sigma_{i=1}^n c_i + m + 5 \in \Theta(nm)$.

(iv) Number of rules: $n + \Sigma_{j=1}^m \Sigma_{i=1}^n w_{ij} b_j + k\Sigma_{i=1}^n c_i + m + 4 \in \Theta(nm)$.

(v) Number of execution time associated with rules: $n + \Sigma_{j=1}^m \Sigma_{i=1}^n w_{ij} b_j + k\Sigma_{i=1}^n c_i + m + 4 \in \Theta(nm)$.

(vi) Maximal length of a rule: $5 \in \Theta(1)$.

Therefore, a deterministic Turing machine can build $\Pi(\langle n, m\rangle, e)$ in a polynomial time with respect to n and m.

It is clear that the family of timed tissue P systems constructed above is time-free sound, time-free complete, and time-free polynomially bounded. And the total number of valid steps to complete the whole computation process is no more than $n + \text{Max}\{k+1, b_j + 2 \mid 1 \le j \le m\} + 2$. From all the above we have the following result.

Theorem 1. *A family of timed tissue P system* $\prod = \{\Pi(s(u), e) \mid u \in I_x\}$ *can be constructed as a uniform time-free solution to multidimensional 0-1 knapsack problem. For any time mapping e, the correctness of the solution does not depend on the execution time of the rules.*

5. Conclusions and Remarks

In this paper, we have proposed a time-free solution to a NP-complete problem, the multidimensional 0-1 knapsack problem. The feature of time-free tissue P systems is that they can work independently from the values associated with the execution times of the rules. The solution above can be used as a scheme for designing solutions to other NP-complete problems by a space-time trade off strategy. In the future work, we will give direct time-free solutions to other NP-complete problems or even PSPACE-problems.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The work is supported by National Natural Science Foundation of China (61202011, 61272152, 61071151, 60971085, and 61272385), Natural Science Foundation of Fujian Province (2011J01334), and Ph.D. Programs Foundation of Ministry of Education of China (20120121120039).

References

[1] G. Păun, "Computing with membranes," *Journal of Computer and System Sciences*, vol. 61, no. 1, pp. 108–143, 2000.

[2] L. Pan and T.-O. Ishdorj, "P systems with active membranes and separation rules," *Journal of Universal Computer Science*, vol. 10, no. 5, pp. 630–649, 2004.

[3] A. Păun and G. Păun, "The power of communication: P systems with symport/antiport," *New Generation Computing*, vol. 20, no. 3, pp. 295–305, 2002.

[4] C. Martín-Vide, G. Păun, J. Pazos, and A. Rodríguez-Patón, "Tissue P systems," *Theoretical Computer Science*, vol. 296, no. 2, pp. 295–326, 2003.

[5] X. Zhang, Y. Niu, L. Pan, and M. J. Pérez-Jiménez, "Linear time solution to prime factorization by tissue P systems with cell division," *International Journal of Natural Computing Research*, vol. 2, no. 3, pp. 49–60, 2011.

[6] X. Zhang, S. Wang, Y. Niu, and L. Pan, "Tissue p systems with cell separation: attacking the partition problem," *Science China Information Sciences*, vol. 54, no. 2, pp. 293–304, 2011.

[7] M. Ionescu, G. Păun, and T. Yokomori, "Spiking neural P systems," *Fundamenta Informaticae*, vol. 71, no. 2-3, pp. 279–308, 2006.

[8] T. Song, L. Pan, and G. Păun, "Asynchronous spiking neural P systems with local synchronization," *Information Sciences*, vol. 219, pp. 197–207, 2013.

[9] M. A. Colomer, A. Margalida, and M. J. Pérez-Jiménez, "Population Dynamics P System, (PDP) Models: a standardized protocol for describing and applying novel bio-inspired computing tools," *PLoS ONE*, vol. 8, no. 4, Article ID e60698, 2013.

[10] D. Pescini, D. Besozzi, G. Mauri, and C. Zandron, "Dynamical probabilistic P systems," *International Journal of Foundations of Computer Science*, vol. 17, no. 1, pp. 183–204, 2006.

[11] F. J. Romero-Campero and M. J. Pérez-Jiménez, "A model of the quorum sensing system in Vibrio Fischeri using P systems," *Artificial Life*, vol. 14, no. 1, pp. 95–109, 2008.

[12] L. Pan, J. Wang, and H. J. Hoogeboom, "Spiking neural P systems with astrocytes," *Neural Computation*, vol. 24, no. 3, pp. 805–825, 2012.

[13] L. Xu and P. Jeavons, "Simple neural-like P systems for maximal independent set selection," *Neural Computation*, vol. 25, no. 6, pp. 1642–1659, 2013.

[14] X. Zhang, B. Luo, X. Fang, and L. Pan, "Sequential spiking neural P systems with exhaustive use of rules," *BioSystems*, vol. 108, no. 1–3, pp. 52–62, 2012.

[15] M. Cavaliere, O. H. Ibarra, G. Păun, O. Egecioglu, M. Ionescu, and S. Woodworth, "Asynchronous spiking neural P systems," *Theoretical Computer Science*, vol. 410, no. 24-25, pp. 2352–2364, 2009.

[16] L. Pan and X. Zeng, "Small universal spiking neural P systems working in exhaustive mode," *IEEE Transactions on Nanobioscience*, vol. 10, no. 2, pp. 99–105, 2011.

[17] T. Song, L. Pan, J. Wang, I. Venkat, K. Subramanian, and R. Abdullah, "Normal forms of spiking neural P systems with antispikes," *IEEE Transactions on Nanobio-Science*, vol. 11, pp. 352–359, 2012.

[18] J. M. Cecilia, J. M. García, G. D. Guerrero, M. A. Martínez-del-Amor, I. Pérez-Hurtado, and M. J. Pérez-Jiménez, "Simulation of P systems with active membranes on CUDA," *Briefings in Bioinformatics*, vol. 11, no. 3, pp. 313–322, 2009.

[19] S. Cheruku, A. Păun, F. J. Romero-Campero, M. J. Pérez-Jiménez, and O. H. Ibarra, "Simulating FAS-induced apoptosis by using P systems," *Progress in Natural Science*, vol. 17, no. 4, pp. 424–431, 2007.

[20] M. À. Colomer, A. Margalida, D. Sanuy, and M. J. Pérez-Jiménez, "A bio-inspired computing model as a new tool for modeling ecosystems: the avian scavengers as a case study," *Ecological Modelling*, vol. 222, no. 1, pp. 33–47, 2011.

[21] F. J. Romero-Campero and M. J. Pérez-Jiménez, "Modelling gene expression control using P systems: the Lac Operon, a case study," *BioSystems*, vol. 91, no. 3, pp. 438–457, 2008.

[22] L. Valencia-Cabrera, M. García-Quismondo, M. J. Pérez-Jiménez, Y. Su, H. Yu, and L. Pan, "Modeling logic gene networks by means of probabilistic dynamic P systems," *International Journal of Unconventional Computing*, vol. 9, no. 5-6, pp. 445–464, 2013.

[23] G. Păun and R. Păun, "Membrane computing and economics: numerical P systems," *Fundamenta Informaticae*, vol. 73, no. 1-2, pp. 213–227, 2006.

[24] R. Ceterchi, R. Gramatovici, N. Jonoska, and K. G. Subramanian, "Tissue-like P systems with active membranes for picture generation," *Fundamenta Informaticae*, vol. 56, no. 4, pp. 311–328, 2003.

[25] B. Nagy and L. Szegedi, "Membrane computing and graphical operating systems," *Journal of Universal Computer Science*, vol. 12, no. 9, pp. 1312–1331, 2006.

[26] J. Cheng, G. Zhang, and X. Zeng, "A novel membrane algorithm based on differential evolution for numerical optimization," *International Journal of Unconventional Computing*, vol. 7, no. 3, pp. 159–183, 2011.

[27] S. Elias, V. Gokul, K. Krithivasan, M. Gheorghe, and G. Zhang, "A variant of distributed P systems for real time cross layer optimization," *Journal of Universal Computer Science*, vol. 18, no. 13, pp. 1760–1781, 2012.

[28] T. Y. Nishida, "An approximate algorithm for NP-complete optimization problems exploiting P systems," in *Proceedings of the Brainstorming Workshop on Uncertainty in Membrane Computing*, pp. 185–192, 2004.

[29] S. Yang and N. Wang, "A novel P systems based optimization algorithm for parameter estimation of proton exchange membrane fuel cell model," *International Journal of Hydrogen Energy*, vol. 37, no. 10, pp. 8465–8476, 2012.

[30] G.-X. Zhang, M. Gheorghe, and C.-Z. Wu, "A quantum-inspired evolutionary algorithm based on P systems for knapsack problem," *Fundamenta Informaticae*, vol. 87, no. 1, pp. 93–116, 2008.

[31] H. Peng, J. Wang, M. J. Pérez-Jiménez, H. Wang, J. Shao, and T. Wang, "Fuzzy reasoning spiking neural P system for fault diagnosis," *Information Sciences*, vol. 235, pp. 106–116, 2013.

[32] G. Păun, *Membrane Computing. An Introduction*, Springer, Berlin, Germany, 2002.

[33] G. Păun, G. Rozenberg, and A. Salomaa, *The Oxford Handbook of Membrane Computing*, Oxford University Press, 2010.

[34] G. Păun, M. J. Pérez-Jiménez, and A. Riscos-Núñez, "Tissue P systems with cell division," *International Journal of Computers, Communications & Control*, vol. 3, no. 3, pp. 295–303, 2008.

[35] M. Cavaliere and D. Sburlan, "Time-independent P systems," in *Membrane Computing*, Lecture Notes in Computer Science, pp. 239–258, 2005.

[36] T. Song, X. Wang, and H. Zheng, "Time-free solution to Hamilton path problems using P systems with d-division," *Journal of Applied Mathematics*, vol. 2013, Article ID 975798, 7 pages, 2013.

[37] G. Rozenberg, *Handbook of Formal Languages: Word, Language, Grammar*, vol. 1, Springer, 1997.

[38] M. J. Pérez-Jiménez, "An approach to computational complexity in membrane computing," in *Membrane Computing*, vol. 3365 of *Lecture Notes in Computer Science*, pp. 85–109, 2005.

[39] R. G. Michael and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, New York, NY, USA, 1979.

[40] L. Pan and C. Martín-Videc, "Solving multidimensional 0-1 knapsack problem by P systems with input and active membranes," *Journal of Parallel and Distributed Computing*, vol. 65, no. 12, pp. 1578–1584, 2005.