A Membrane-Inspired Evolutionary Algorithm with a Population P System and its Application to Distribution System Reconfiguration

Gexiang Zhang[1], Miguel A. Gutiérrez-Naranjo[2], Yanhui Qin[3], Marian Gheorghe[4]

[1,3] School of Electrical Engineering,
Southwest Jiaotong University, Chengdu 610031, P.R. China
[1]zhgxdylan@126.com; [3]qinyanhui@gmail.com

[2]Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla, 41012, Spain
magutier@us.es

[4] Department of Computer Science,
University of Sheffield Regent Court,
Portobello Street, Sheffield S1 4DP, UK
M.Gheorghe@dcs.shef.ac.uk

Summary. This paper develops a membrane-inspired evolutionary algorithm, PSMA, which is designed by using a population P system and a quantum-inspired evolutionary algorithm (QIEA). We use a population P system with three cells to organize three types of QIEAs, where communications between cells are performed at the level of genes, instead of the level of individuals reported in the existing membrane algorithms in the literature. Knapsack problems are applied to discuss the parameter setting and to test the effectiveness of PSMA. Experimental results show that PSMA is superior to four representative QIEAs and our previous work with respect to the quality of solutions and the elapsed time. We also use PSMA to solve the optimal distribution system reconfiguration problem in power systems for minimizing the power loss.

Key words: Membrane computing; membrane-inspired evolutionary algorithm; population P system; distribution system reconfiguration

1 Introduction

According to the research development of interactions on membrane computing and evolutionary computation, two kinds of research topics, membrane-inspired evolutionary algorithms (MIEAs) and automated design of membrane computing models (ADMCMs), have been reported in the literature.

The automated synthesis of some types of membrane computing models or of a high level specification of them is envisaged to be obtained by applying various heuristic search methods. ADMCMs aim to circumvent the programmability issue of membrane-based models for complex systems. In this direction, Suzuki and Tanaka made the first attempts [20, 21] to introduce a genetic method to the *Artificial Cell Systems (ACS)* via a P system model called *Abstract Rewriting Systems on Multisets* [19, 22], a rewriting Membrane Computing model where the P systems have only one membrane. More recently, new attempts of using evolutionary algorithms to evolve P systems have been presented (see, e.g. [3, 5, 12, 24]). In [3], a nested evolutionary algorithm was used to tuning parameters of P system models. The automatic design of P systems for fulfilling an specific task was first discussed in [5], where genetic algorithms are used for finding simple P systems. In [5], the membrane structure is settled and the genetic evolution only corresponds to the set of rules. A *population* of P systems is considered and two genetic operations, *crossover* and *mutation* perform the evolution of the population. This work was extended from 4^2 to n^2 P systems in [12] by introducing a quantum-inspired evolutionary algorithm (QIEA), where the set of rules were encoded by a binary string and evolutionary operations (quantum-inspired gate (Q-gate) update) were performed on genotypic individuals (quantum-inspired bits (Q-bits)), instead of phenotypic individuals (binary bits) or evolution rules of P systems. The outstanding advantage of this approach is that the difficulty of designing evolutionary operators in the phenotypic space, such as crossover and mutation ones is effectively avoided. In [24], the design of P systems for generating languages and fitness functions were discussed.

A MIEA concentrates on generating new approximate algorithms for solving various optimization problems by using the hierarchical or network structures of membranes and rules of membrane systems, and the concepts and principles of meta-heuristic search methodologies [33, 34]. In [11, 17], a cell-like membrane system with a nested membrane structure (NMS) was used to combine with simulated annealing and genetic algorithms to solve traveling salesman problems and controller design problems for a marine diesel engine. In [31], a QIEA based on P systems (QEPS) was proposed by incorporating a one-level membrane structure (OLMS) with a QIEA. Knapsack problems were applied to verify that QEPS is superior to its counterpart method and OLMS has an advantage over NMS. The use of QEPS to solve sixty-five satisfiability problems with different complexities was discussed in [33]. In [34], the QEPS performance was improved by introducing a local search and the modified QEPS applied to analyze sixteen radar emitter signals. In [4,29], OLMS was integrated with differential evolution approaches and ant colony optimization to solve numeric optimization and travelling salesman problems. In [35], the use of a cell-like membrane system with active membranes to design a MIEA was designed for solving for combinatorial optimization problems. In [27], a MIEA was presented to solve the DNA sequence design problem, which has been proved to be NP-hard. In the above MIEAs, heuristic search methods, such as genetic algorithms, QIEA, differential evolution and ant colony optimization, were considered as an independent subalgorithm inside each membrane. This idea was extended by the proposal of a membrane algorithm with quantum-inspired subalgorithm (MAQIS) in [30], where each membrane contains one component of the approach and all the components inside membranes cooperate together to produce offspring in a single evolutionary generation. The effectiveness of MAQIS was tested on knapsack problems and image sparse decomposition problems. It is worth pointing out that the analysis of the dynamic behavior of MIEAs in the process of evolution with respect to population diversity and convergence showed that MIEAs have better capabilities to balance exploration and exploitation than their corresponding optimization algorithms used [32, 36]. Until now MIEAs have been studied in conjunction with cell-like membrane systems with fixed membrane structures and by principally considering an evolutionary computing approach as a subalgorithm put inside a membrane. Further research topics might include cell-like membrane systems with active membranes, tissue-like membrane systems and population membrane systems for exploring more real-world applications of membrane computing.

In spite of the biological inspiration of membrane computing and evolutionary computation, in the literature there are only a few examples of papers bridging them. We continue to push this work forward. The main motivation of this work is to use a population membrane system to design a MIEA for distribution system reconfiguration. The algorithm called PSMA is designed by appropriately considering a population P system and three variants of QIEAs, where the communications between cells are performed at the level of genes, instead of the level of individuals reported in the existing membrane algorithms in the literature. This is the first attempt to apply a population P system to design an approximate optimization approach. Knapsack problems and distribution system reconfiguration are applied to test the effectiveness and the application of PSMA, respectively. Experimental results show that PSMA can obtain better solutions that four types of QIEAs and QEPS (a MIEA reported in [31]) and is competitive to five types of optimization algorithms for solving distribution system reconfiguration problems in power systems.

This paper is organized as follows. Section 2 introduces briefly QIEA and population P systems, and then describes PSMA in detail. Section 3 presents experiments conducted on knapsack problems for testing the PSMA performance. The application of PSMA to distribution system reconfiguration is discussed in Section 4. Concluding remarks follow in Section 5.

2 PSMA

PSMA uses the network framework of a population P system to organize the objects consisting of quantum-inspired bits (Q-bits) and classical bits, and rules made up of several quantum-inspired gate (Q-gate) evolutionary rules like in QIEA and evolution rules like in membrane systems. To clearly and concisely describe

PSMA, we first give brief introductions on QIEAs and population P systems, and then turn to a detailed presentation of the introduced MIEA.

2.1 QIEA

Inspired by concepts and principles of quantum computing such as quantum bits, quantum gates and a probabilistic observation, Han and Kim [9] proposed a new evolutionary algorithm, QIEA, for a classical computer instead of quantum one. In QIEA, a Q-bit representation is applied to describe individuals of a population; a Q-gate is introduced to generate the individuals at the next generation; a probabilistic observation is employed to link Q-bit representation with binary solutions [28]. A Q-bit is defined by a pair of numbers (α, β) represented as $[\alpha\ \beta]^T$, where $|\alpha|^2$ and $|\beta|^2$ are probabilities that the observation of the Q-bit will render a '0' or '1' state and $\xi = \arctan(\beta/\alpha)$ is the phase of the Q-bit [9,28]. Normalization requires that $|\alpha|^2 + |\beta|^2 = 1$. The evolution of QIEA depends on the operation of Q-gates on Q-bit individuals. The basic pseudocode algorithm for a QIEA is shown in Fig. 1 and a brief description for each step is as follows (here we just list the outline of QIEA algorithm and details will be provided in the algorithm description of PSMA (Section 2.3).).

```
        Begin
          t ← 1
(i)       Initialize Q(t)
          While (not termination condition) do
(ii)          Make P(t) by observing the states of Q(t)
(iii)         Evaluate P(t)
(iv)          Update Q(t) using Q-gates
(v)           Store the best solutions among P(t)
              t ← t+1
          End
        End
```

Fig. 1. Pseudocode algorithm for a QIEA [9,28]

(i) In the "initialize $Q(t)$" step, a population $Q(t)$ with n Q-bit individuals is generated, $Q(t)=\{\boldsymbol{q}_1^t, \boldsymbol{q}_2^t, \cdots, \boldsymbol{q}_n^t\}$, at generation t, where $\boldsymbol{q}_i^t (i = 1, 2, \cdots, n)$ is an arbitrary individual in $Q(t)$ and denoted as

$$\boldsymbol{q}_i^t = \begin{bmatrix} \alpha_{i1}^t | \alpha_{i2}^t | \cdots | \alpha_{il}^t \\ \beta_{i1}^t | \beta_{i2}^t | \cdots | \beta_{il}^t \end{bmatrix} \tag{1}$$

where l is the number of Q-bits, i.e., the string length of the Q-bit individual.

(ii) By observing the states $Q(t)$, binary solutions in $P(t)$, where $P(t) = \{\boldsymbol{x_1^t}, \boldsymbol{x_2^t}, \cdots, \boldsymbol{x_n^t}\}$ are produced at step t. According to the current probability, either $\left|\alpha_{ij}^t\right|^2$ or $\left|\beta_{ij}^t\right|^2$ of $\boldsymbol{q_i^t}$, $i = 1, 2, \cdots, n, j = 1, 2, \cdots, l$, a binary bit 0 or 1 is generated. Thus, l binary bits can construct a binary solution $\boldsymbol{x_i^t}(i = 1, 2, \cdots, n)$. More details can be referred to Step 2 *Observation* in the algorithm description of PSMA (Section 2.3).

(iii) Binary solutions $\boldsymbol{x_j^t}(j = 1, 2, \cdots, n)$ are evaluated and assigned fitness values with respect to a criterion.

(iv) In this step, Q-bit individuals in $Q(t)$ are updated by applying the current Q-gates. The details will be expounded in Step 5 *Q-gate update* in the algorithm description of PSMA (Section 2.3).

(v) The best solutions among $P(t)$ are selected and stored.

2.2 Population P Systems

A population P system is a special kind of tissue P systems except for two important differences that the structure can be dynamically changed by using bond making rules and cells are allowed to communicate indirectly by means of the environment [2].

A population P system with degree n is formally defined as follows [2]

$$\mathcal{P} = (V, \gamma, \alpha, \omega_e, C_1, C_2, \ldots, C_n, c_o),$$

where

(i) V is a finite alphabet of symbols called objects;

(ii) $\gamma = (\{1, 2, \ldots, n\}, E)$, with $E \subseteq \{\{i, j\} | 1 \leq i \neq j \leq n\}$, is a finite undirected graph;

(iii) α is a finite set of bond making rules $(i, x_1; x_2, j)$, with $x_1, x_2 \in V^*$, and $1 \leq i \neq j \leq n$;

(iv) $\omega_e \in V^*$ is a finite multiset of objects initially assigned to the environment;

(v) $C_i = (\omega_i, S_i, R_i)$, for each $1 \leq i \leq n$, with
   (a) $\omega_i \in V^*$ a finite multiset of objects,
   (b) S_i is a finite set of communication rules; each rule has one of the following forms: $(a; b, in)$, $(a; b, enter)$, $(b, exit)$, for $a \in V \bigcup \{\lambda\}, b \in V$,
   (c) R_i is a finite set of transformation rules of the form $a \to y$, for $a \in V$, and $y \in V^+$;

(vi) c_o is the (label of the) output cell, $1 \leq c_o \leq n$.

A population P system $\mathcal{P}$ is defined as a collection of n cells where each cell C_i corresponds in a one-to-one manner to a node i in a finite undirected graph γ, which defines the initial structure of the system. Cells are allowed to communicate alongside the edges of the graph γ, which are unordered pairs of the form $\{(i, j)\}$, with $1 \leq i \neq j \leq n$. The cells C_i, $1 \leq i \leq n$, are associated in a one-to-one manner with the set of nodes $\{1, 2, \ldots, n\}$. Each cell C_i gets assigned a finite multiset of objects ω_i, a finite set of communication rules S_i, and a finite set of transformation rules R_i. Each set R_i contains rules of the form $x \to y$ that allow cell i to consume a multiset x in order to produce a new multiset y inside cell i. Communication rules in S_i of the form $(a; b, in)$ are instead used by cell i to receive objects from its neighboring cells if the object a is placed in the cell i. The rules of the forms $(a; b, enter)$ mean that objects from the environment can enter the cell i if an object a is present in it. The rules of the forms $(b, exit)$ allow the cell i to release an object b in the environment.

Cell capability of moving objects alongside the edges of the graph is influenced by particular bond making rules in α that allow cells to form new bonds. A bond making rule $(i, x_1; x_2, j)$ specifies that, in the presence of a multiset x_1 in the cell i and a multiset x_2 inside the cell j, a new bond can be created between the two cells. This means that a new edge $\{i, j\}$ can be added to the graph that currently defines the structure of the system. Thus the structure of a population P system can be dynamically changed in the process of evolution of the system.

A step of a computation in a population P system $\mathcal{P}$ is defined as being performed in two separate stages: the content of the cells is firstly modified by applying the communication rules in S_i, and the transformation rules in R_i, for all $1 \leq i \leq n$; the structure of the system is then modified by using the bond making rules in α. A successful computation in $\mathcal{P}$ is defined as a finite sequence of configurations from processing the initial multisets ω_i to the final state where the content of the cells cannot be modified anymore by means of some communication rules and transformation rules after a last bond making stage. The result is given by the number of objects that are placed inside the output cell c_o in the final configuration.

2.3 PSMA

In this subsection, we design PSMA by applying the dynamic network structure of a population P system with three cells and three representative variants of QIEAs that have good performance in terms of the investigations in [9, 10, 28, 37, 38]. Specifically, three QIEAs, QIEA02 [9], QIEA04 [10] and QIEA07 [38], are placed inside three cells of the population P system in a common environment. The objects consist of Q-bits and classical bits. The rules are composed of observation and Q-gate update rules of QIEAs, transformation rules in the population P system, evaluation rules for candidate solutions, communication rules for exchanging information between the three cells and bond making rules for modifying the structure of the system. Q-bits, organized as a Q-bit individual which is a special string of Q-bits, are processed as multisets of objects. Classical bits, which are obtained from their corresponding Q-bits by applying a probabilistic observation process, are arranged as a binary string and are treated with also as multisets of objects. Inside each cell, the processes of initialization, observation, evaluation and Q-gate update processes for producing offspring are performed independently. Information exchange between individuals are executed through communications between cells at the level of genes. In PSMA, a binary string corresponds a candidate solution of a problem. The set of rules are responsible for evolving the system. The framework of the population P system used in PSMA is shown in Fig. 2, where ovals represent the cells and dashed lines indicate the links. The population P system can be described as the following construct

$$\mathcal{P} = (V, \gamma, \alpha, \omega_e, C_1, C_2, C_3, c_e),$$

where

(i) V is a finite alphabet that consists of all possible Q-bits and classical bits (*objects*)(It is worth noting that the alphabet used in this paper is finite because the number of possible Q-bits equals the product of the number of Q-bit individuals and the length of a Q-bit individual);
(ii) $\gamma = (\{1,2,3\}, E)$, with $E = \{(1,2),(1,3),(2,3)\}$, is a finite undirected graph;
(iii) α is a finite set of bond making rules $(i, \lambda; \lambda, j)$ or $\emptyset$ if no new bond can be added;
(iv) $\omega_e = \lambda$;
(v) $C_i = (\omega_i, S_i, R_i)$, for each $1 \leq i \leq 3$, with
  (a) $\omega_i = \boldsymbol{q_1 q_2} \cdots \boldsymbol{q_{n_i}}$, where $\boldsymbol{q_i}, i = 1, 2, \cdots, n_i$, is a Q-bit individual as shown in (1); n_i is the number of individuals in cell C_i and satisfies $\sum_i^3 n_i = N$, where N is the total number of individuals in this system;
  (b) S_i is a finite set of communication rules; each rule has one of the following forms: $(\lambda; b, in)$, $(b, exit)$, for $b \in V$,
  (c) R_i is a finite set of transformation rules of the form $a \to y$, for $a \in V$, and $y \in V^+$;
(vi) c_e means that the result is collected in the environment.

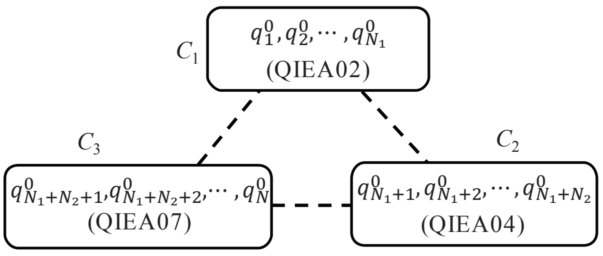

C_1
$q_1^0, q_2^0, \cdots, q_{N_1}^0$
(QIEA02)
C_3
$q_{N_1+N_2+1}^0, q_{N_1+N_2+2}^0, \cdots, q_N^0$
(QIEA07)
C_2
$q_{N_1+1}^0, q_{N_1+2}^0, \cdots, q_{N_1+N_2}^0$
(QIEA04)

Fig. 2. The framework of the population P system involved in this paper

To clearly understand PSMA, in what follows we describe its algorithm step by step.

Step 1 *Initialization*: a membrane structure of a population P system with three cells in a common environment is created. An initial population with N individuals is generated. Each individual is composed of a certain number of Q-bits. The N individuals are randomly scattered across the three cells so that $n_i > 1$ and $\sum_i^3 n_i = N$, $i = 1, 2, 3$.

Step 2 *Observation*: a probabilistic observation process occurring in step (ii) of QIEA is applied to establish a link between genotypes and phenotypes, i.e., between Q-bits and classical bits. To be specific, as for the Q-bit $[\alpha\ \beta]^T$, if a random number r between 0 and 1 is less than $|\beta|^2$, i.e., $r < |\beta|^2$, the observed classical bit equals 1, otherwise, it is 0. Thus, given a Q-bit individual, we can get a corresponding binary solution. The observation process is shown in Fig. 3

```
Begin
    If  $random[0,1) < |\alpha|^2$
    Then  $x \leftarrow 0$
    Else  $x \leftarrow 1$
End
```

Fig. 3. Observation process in PSMA

Step 3 *Evaluation*: a specific criterion with respect to a problem is used to evaluate all the binary solutions obtained in Step 2. This step is identical with step (iii) of QIEA.

Step 4 *Communication*: Suppose that P_k represents the binary individuals obtained in Step 2 inside cell C_k, $P_k = \{\boldsymbol{x_1^k}, \boldsymbol{x_2^k}, \cdots, \boldsymbol{x_N^k}\}$, where $k = 1, 2, 3$; $\boldsymbol{x_i^k} = b_{i1}^k b_{i2}^k \cdots b_{im}^k$, where b_{ij}^k is a gene of $\boldsymbol{x_i^k}$ and m is the number of genes in an individual; the fitness of the individual $\boldsymbol{x_i^k}$ is $f(\boldsymbol{x_i^k})$. In this step, as for each gene b_{ij}^k in the binary individual $\boldsymbol{x_i^k}$, a random number r_c following a uniform distribution in the range [0, 1] is produced; if $r_c < p_c$, we randomly choose two binary individuals, $\boldsymbol{x_{c_1}^{k_1}}$ and $\boldsymbol{x_{c_2}^{k_2}}$, from the whole population (N individuals) except for the individual $\boldsymbol{x_i^k}$, where k_1 and k_2 are the labels of cells, k_1,k_2=1,2,3; p_c is a parameter denoting a communication rate and will be discussed in the next section. If $f(\boldsymbol{x_{c_1}^{k_1}})$ is better than $f(\boldsymbol{x_{c_2}^{k_2}})$, we use the gene $b_{c_1j}^{k_1}$ to replace b_{ij}^k, otherwise, we use the gene $b_{c_2j}^{k_2}$ to replace b_{ij}^k. Thus we can obtain another binary individual $\bar{\boldsymbol{x}}_i^k$ corresponding to $\boldsymbol{x_i^k}$. In the process of replacement, the values of k_1, k_2 and k decide what structure will be created and used to perform the communication for information exchange between cells k and k_1 or k_2. As for the values of k_1, k_2 and k, there are three cases: (1) $k_1 = k_2 = k$ means that no communication will be performed, i.e., the dashed lines in Fig. 2 do not work and the three cells are separate; (2) $k_1 = k_2 \neq k$ or $k_1 = k \neq k_2$ or $k_1 \neq k_2 = k$ means that communication is performed between two cells, i.e., only one of the dashed lines in Fig. 2 works and the communication rule $(\lambda; b, in)$ is performed between the two cells having the channel that works; (3) $k_1 \neq k_2 \neq k$ means that the three dashed lines in Fig. 2 work and the three cells communicate with each other. Thus the communication rule $(\lambda; b, in)$ is performed between each pair of cells.

Step 5 *Q-gate update*: transformation rules of the form $a \to y$ is utilized to evolve the objects in each of the three cells. The rules considered here are applied according to evolutionary mechanisms of QIEAs, instead of the semantics of P systems. The Q-gate update procedure

$$\begin{bmatrix}\alpha^{t+1}\\ \beta^{t+1}\end{bmatrix} = G(\theta)\begin{bmatrix}\alpha^{t}\\ \beta^{t}\end{bmatrix} = \begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}\alpha^{t}\\ \beta^{t}\end{bmatrix} \tag{2}$$

is used to transform the current Q-bit $[\alpha^t\ \beta^t]^T$ into the corresponding Q-bit $\left[\alpha^{t+1}\ \beta^{t+1}\right]^T$ at generation t+1. The rotation angle θ in the Q-gate $G(\theta)$ in different cells has different definitions.

To be specific, in cell 1, the rotation angle is defined as $\theta = s(\alpha,\beta)\cdot\varDelta\theta$, where $\varDelta\theta$ is the value of θ determining the convergence speed of the algorithm and $s(\alpha,\beta)$ is the sign of θ deciding the search direction. The approach for looking up the rotation angle θ in [9] is shown in Tables 1, where $f(.)$ is the fitness function; α and β are the probabilities of the current Q-bit.

Table 1. Q-gate update approach in cell 1, where $f(.)$ is the fitness function, $\varDelta\theta$ and $s(\alpha,\beta)$ are the value and the sign of θ, x and b are the bits of the binary individuals $\boldsymbol{x}_i^1$ and $\boldsymbol{x}_i^1$, respectively [9]

x	b	$f(\boldsymbol{x}) \geq f(\boldsymbol{b})$	$\varDelta\theta$	$s(\alpha,\beta)$	
				$\alpha=0$	$\beta=0$
0	0	False	0	–	–
0	0	True	0	–	–
0	1	False	0.01π	+1	-1
0	1	True	0	–	–
1	0	False	0.01π	+1	-1
1	0	True	0	–	–
1	1	False	0	–	–
1	1	True	0	–	–

In cell 2, the Q-gate update procedure in (2) and the approach in Table 1 are firstly used. Then an additional process is applied to modify the Q-bit $\left[\alpha^{t+1}\ \beta^{t+1}\right]^T$. The modification method is as follows.

(i) If $|\alpha^{t+1}|^2 \leq \epsilon$ and $|\beta^{t+1}|^2 \geq 1-\epsilon$, $\left[\alpha^{t+1}\ \beta^{t+1}\right]^T = \left[\sqrt{\epsilon}\ \sqrt{1-\epsilon}\right]^T$;
(ii) If $|\alpha^{t+1}|^2 \geq 1-\epsilon$ and $|\beta^{t+1}|^2 \leq \epsilon$, $\left[\alpha^{t+1}\ \beta^{t+1}\right]^T = \left[\sqrt{1-\epsilon}\ \sqrt{\epsilon}\right]^T$.

According to the investigation in [10], the parameter ϵ is usually assigned as 0.01.

In cell 3, the approach for choosing the quantum rotation angle was defined by using the ratio of the probabilities of Q-bits [38]. The rotation angle θ is defined as

$$\theta = \theta_0 s(\alpha, \beta) f(\gamma_\alpha, \gamma_\beta) \tag{3}$$

where α and β represent the probabilities of a Q-bit; θ_0 is an initial rotation angle and is usually set to 0.05π; $s(\alpha, \beta)$ is a function determining the search direction of the algorithm; $f(\gamma_\alpha, \gamma_\beta)$ is a function of γ_α or γ_β, where $\gamma_\alpha = |\beta|/\alpha$, $\gamma_\beta = 1/\gamma_\alpha$. The values of $s(\alpha, \beta)$ and $f(\gamma_\alpha, \gamma_\beta)$ can be obtained in Table 2.

Table 2. Q-gate update approach in cell 3, where $f(.)$ is the fitness function, x and b are the bits of the binary individuals $\boldsymbol{x}_i^1$ and $\bar{\boldsymbol{x}}_i^1$, respectively [38]

x	b	$f(\boldsymbol{x}) \geq f(\boldsymbol{b})$	$s(\alpha, \beta)$			$f(\gamma_\alpha, \gamma_\beta)$
			$\alpha\beta \geq 0$	$\alpha\beta < 0$	$\alpha\beta = 0$	
0	0	false	-1	+1	± 1	$\exp(-\gamma_\beta)$
0	0	true	-1	+1	± 1	$\exp(-\gamma_\beta)$
0	1	false	+1	-1	± 1	$\exp(-\gamma_\alpha)$
0	1	true	-1	+1	± 1	$\exp(-\gamma_\beta)$
1	0	false	-1	+1	± 1	$\exp(-\gamma_\beta)$
1	0	true	+1	-1	± 1	$\exp(-\gamma_\alpha)$
1	1	false	+1	-1	± 1	$\exp(-\gamma_\alpha)$
1	1	true	+1	-1	± 1	$\exp(-\gamma_\alpha)$

Step 6 *Halting*: the algorithm stops when a prescribed number of evolutionary generations is attained.

Step 7 *Output*: the communication rule $(b, exit)$ is responsible for sending the best solutions out to the environment at the end of the computation. To be specific, each cell send the best solution inside it out to the environment at the end of the computation; thus there are three solutions coming from three cells in the environment; through a comparison, we collect the best one among the three solutions as the final solution of the computation.

3 Experiments

In this section, a well-known NP-hard combinatorial optimization problem, knapsack problem, is used to test the PSMA performance. The knapsack problem can be described as selecting from among various items those items that are most profitable, given that the knapsack has limited capacity [7, 9]. The knapsack problem is to select a subset from the given number of items so as to maximize the profit $f(x)$:

$$f(x) = \sum_{i=1}^{k} p_i x_i \tag{4}$$

subject to

$$\sum_{i=1}^{k} w_i x_i \leq C \tag{5}$$

where k is the number of items; p_i is the profit of the i-th item; w_i is the weight of the i-th item; C is the capacity of the given knapsack; and x_i is 0 or 1.

In the following experiments, strongly correlated sets of unsorted data are used

$$\omega_i = uniformly\, random[1, 50]$$

$$p_i = w_i + 25$$

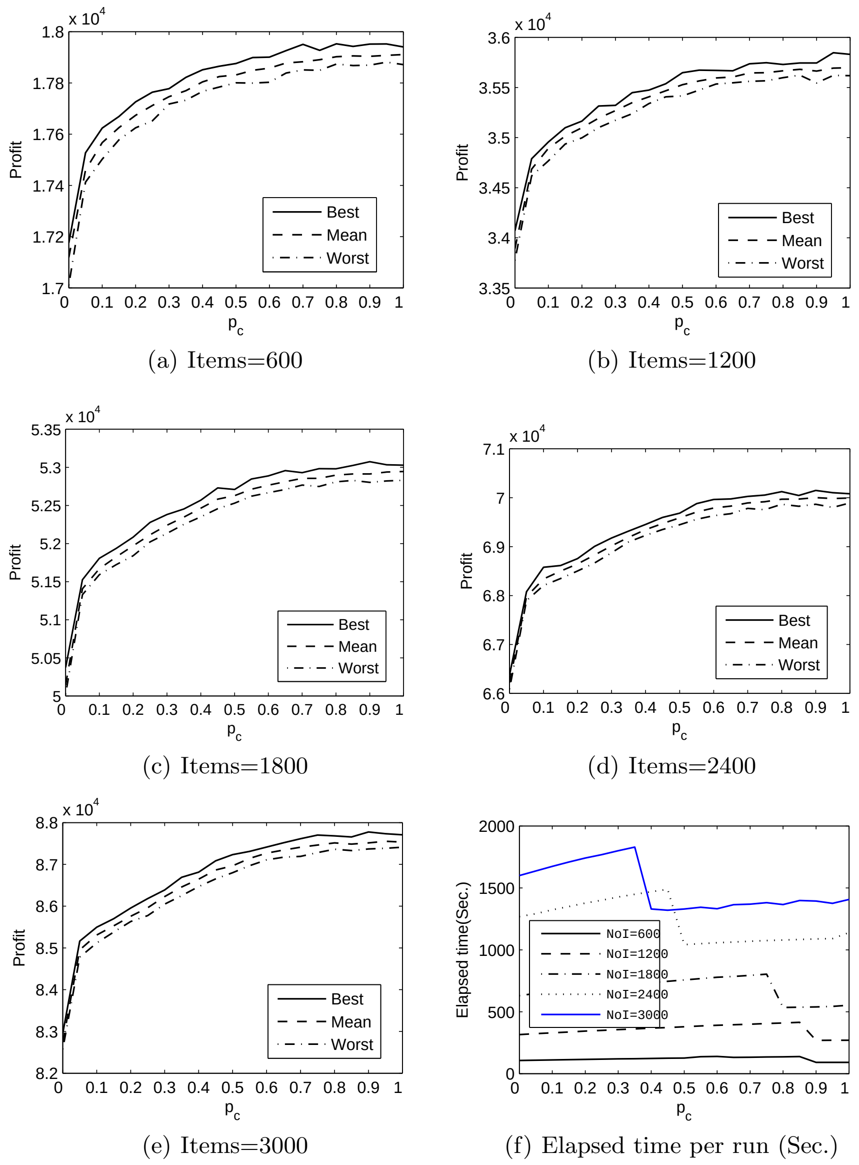
and the average knapsack capacity C is applied.

$$C = \frac{1}{2}\sum_{i=1}^{k} w_i \tag{6}$$

First of all, we use five knapsack problems with respective 600, 1200, 1600, 1800, 2400 and 3000 to discuss the choice of the parameter p_c in PSMA. The population size N=20. The the numbers, 20000, 30000, 40000, 60000 and 60000, of function evaluations are used as the stopping conditions for knapsack problems with 600, 1200, 1600, 1800, 2400 and 3000, respectively. Let p_c vary from 0 to 1 with interval 0.05, i.e., there are 21 cases. In the experiment, we perform 30 independent runs for each of 21 values of p_c of each knapsack problem. We record the best, mean and worst solutions over 30 runs and the elapsed time per run. Experimental results are shown in Fig. 4. It can be seen from these results that the parameter p_c could be assigned as the value ranged between 0.9 and 0.95 in terms of the quality of solutions and the elapsed time. Thus we set p_c to 0.9 in the following experiments.

To test the effectiveness of PSMA, Fifteen knapsack problems that have the items varied from 200 to 3000 items with interval 200 are used to conduct comparative experiments. Benchmark algorithms are considered to be composed of four types of QIEAs and the membrane algorithm QEPS in [31]. The four QIEAs include QIEA02 in [9], QIEA04 in [10], QIEA07 in [38] and QIEA08 [25]. 20 individuals are used in the six algorithms and 30 independent runs are performed for each of the 15 cases of each algorithm. The stopping condition for the six algorithms is set as follows: 20000 function evaluations for the first 4 knapsack problems; 30000 function evaluations for the 3 knapsack problems with 1000, 1200 and 1400; 40000 function evaluations for the 4 knapsack problems with 1600, 1800, 2000 and 2200; 60000 function evaluations for the last 4 knapsack problems. The best, mean and worst solutions over 30 independent runs and the elapsed time per run are recorded and listed in Tables 3 and 4.

As shown in Tables 3 and 4, we can conclude that PSMA is superior to QIEA02, QIEA04, QIEA07, QIEA08 and QEPS in terms of the the quality of best, mean and worst solutions and the elapsed time. To show that PSMA really outperforms



(a) Items=600

(b) Items=1200

(c) Items=1800

(d) Items=2400

(e) Items=3000

(f) Elapsed time per run (Sec.)

Fig. 4. Experimental results for p_c

Table 3. Experimental results of the first 8 knapsack problems. Best, Mean, Worst and Time represent the best, mean and worst solutions over 30 independent runs and the elapsed time per run, respectively (to be continued)

items		200	400	600	800	1000	1200	1400	1600
QIEA02	Best	5885	11650	17403	22940	28673	34399	39560	45277
	Mean	5786	11553	17173	22659	28333	33984	39149	44864
	Worst	5359	11396	16851	22010	27954	33424	38488	44423
	Time	24	48	72	96	182	221	259	413
QIEA04	Best	5749	11272	16561	21684	27024	32429	37329	42892
	Mean	5674	11081	16327	21499	26812	32210	37134	42596
	Worst	5627	10666	15680	20809	26524	31213	36028	42096
	Time	29	57	89	115	225	272	320	547
QIEA07	Best	5935	11850	17749	23390	29204	35099	40490	46403
	Mean	5893	11760	17627	23286	29080	34949	40267	46184
	Worst	5859	11700	17527	23139	28929	34822	40114	46002
	Time	26	52	78	104	197	249	402	420
QIEA08	Best	5456	10699	15734	20956	26073	31419	36384	41750
	Mean	5367	10615	15659	20747	25901	31244	36081	41499
	Worst	5325	10536	15591	20634	25775	31071	35942	41387
	Time	29	58	92	126	249	309	376	599
QEPS	Best	5959	11873	17702	23403	29531	35441	40886	47242
	Mean	5945	11837	17647	23257	29373	35292	40722	47018
	Worst	5909	11778	17575	23109	29198	35061	40364	46672
	Time	22	44	70	92	178	215	246	398
PSMA	Best	5984	11975	18000	23859	29822	35845	41412	47470
	Mean	5963	11965	17945	23782	29750	35782	41301	47365
	Worst	5959	11946	17902	23729	29687	35727	41214	47300
	Time	32	64	97	129	240	288	337	512

the other five algorithms, we go further to employ statistical techniques to analyze the behavior of the six algorithms over the 15 knapsack problems. Both parametric and non-parametric methods are considered. The parametric statistical analysis, also called single-problem analysis [6], is used to analyze whether there is a significant difference over one optimization problem between two algorithms. The non-parametric statistical test, also called multiple-problem analysis [6], is applied to compare different algorithms whose results represent average values for each problem. In Tables 5, a 95% confidence t-test is applied to check whether the mean solutions of the two pairs of algorithms, PSMA vs. QIEA02, QIEA04, QIEA07, QIEA08 and QEPS, are significantly different or not. Two non-parametric tests, Wilcoxon's and Friedman's tests, are employed to check whether there are signifi-

Table 4. Experimental results of the last 7 knapsack problems. Best, Mean, Worst and Time represent the best, mean and worst solutions over 30 independent runs and the elapsed time per run, respectively (continued)

items		1800	2000	2200	2400	2600	2800	3000
QIEA02	Best	50784	56453	61645	66683	72546	77511	83294
	Mean	50163	55879	61175	65984	71992	76734	82608
	Worst	49506	55129	59820	64981	71497	75924	82020
	Time	475	538	619	1056	1176	1310	1454
QIEA04	Best	47920	53276	58723	62952	68858	73355	79068
	Mean	47513	53018	58278	62523	68448	72938	78548
	Worst	46444	51889	56942	62250	66910	71360	76743
	Time	569	636	708	1268	1356	1448	1560
QIEA07	Best	51882	57579	63199	68351	74531	79471	85343
	Mean	51669	57414	62985	68093	74237	79215	85073
	Worst	51459	57277	62768	67932	73998	78685	84753
	Time	475	530	587	964	1049	1133	1222
QIEA08	Best	46507	52127	57221	61294	67228	71600	77142
	Mean	46293	51816	57008	61063	66950	71308	76867
	Worst	46155	51618	56811	60894	66817	71121	76709
	Time	702	815	983	1706	1950	2230	2515
QEPS	Best	52772	58775	64513	70402	76621	81918	88207
	Mean	52600	58543	64230	70015	76245	81486	87657
	Worst	52395	58065	63680	69726	75296	80683	87044
	Time	464	523	605	1051	1170	1289	1441
PSMA	Best	53201	59091	64955	70244	76569	81806	87899
	Mean	53071	58968	64785	70134	76442	81664	87740
	Worst	52982	58819	64669	70024	76311	81505	87565
	Time	576	643	709	1156	1253	1352	1451

cant differences between the two pairs of algorithms, PSMA vs. QIEA02, QIEA04, QIEA07, QIEA08 and QEPS. The level of significance considered is 0.05. The results of Wilcoxon's and Friedman's tests are shown in Table 6. In Tables 5 and 6, The symbols "+" and "–" represent significant difference and no significant difference, respectively.

As shown in Tables 5 and 6, the t-test results demonstrate that there are significant differences between the two pairs of algorithms, PSMA vs. QIEA02, QIEA04, QIEA07, QIEA08 and QEPS. The p-values of the Wilcoxon's and Friedman's tests in Table 6 are far smaller than the level of significance 0.05, which indicates that PSMA really outperforms QIEA02, QIEA04, QIEA07, QIEA08 and QEPS.

Table 5. The results of *t*-test for the algorithms in Tables 3 and 4. The symbols "+" and "–" represent significant difference and no significant difference, respectively

PSMA vs.	QIEA02	QIEA04	QIEA07	QIEA08	QEPS
200 items	3.41e-14 (+)	4.57e-49 (+)	3.12e-24 (+)	1.41e-65 (+)	7.83e-08 (+)
400 items	6.71e-40 (+)	2.81e-49 (+)	9.19e-38 (+)	1.93e-81 (+)	1.81e-33 (+)
600 items	7.30e-36 (+)	6.41e-53 (+)	7.29e-35 (+)	1.99e-92 (+)	1.72e-43 (+)
800 items	4.39e-38 (+)	1.66e-60 (+)	7.09e-47 (+)	6.87e-83 (+)	1.24e-44 (+)
1000 items	5.93e-44 (+)	3.49e-77 (+)	1.71e-48 (+)	4.70e-94 (+)	1.77e-28 (+)
1200 items	1.67e-43 (+)	8.17e-64 (+)	7.48e-50 (+)	5.28e-89 (+)	4.92e-38 (+)
1400 items	5.58e-49 (+)	6.68e-66 (+)	3.65e-51 (+)	1.06e-92 (+)	8.74e-33 (+)
1600 items	4.75e-54 (+)	6.57e-82 (+)	6.17e-53 (+)	2.12e-98 (+)	4.43e-21 (+)
1800 items	1.43e-48 (+)	1.32e-71 (+)	1.61e-59 (+)	3.02e-98 (+)	1.06e-32 (+)
2000 items	3.98e-50 (+)	5.76e-73 (+)	6.57e-63 (+)	1.23e-94 (+)	3.71e-19 (+)
2200 items	9.07e-51 (+)	2.02e-69 (+)	6.06e-59 (+)	4.94e-97 (+)	7.45e-22 (+)
2400 items	9.34e-50 (+)	8.21e-88 (+)	6.78e-63 (+)	7.97e-101 (+)	2.70e-03 (+)
2600 items	6.98e-62 (+)	3.20e-72 (+)	2.24e-62 (+)	4.07e-101 (+)	6.71e-04 (+)
2800 items	8.78e-59 (+)	3.91e-73 (+)	1.45e-58 (+)	7.47e-102 (+)	1.60e-03 (+)
3000 items	5.86e-64 (+)	9.18e-73 (+)	1.36e-64 (+)	1.08e-102 (+)	1.16e-01 (–)

Table 6. The *p*-values of Wilcoxon's and Friedman's tests for the algorithms in Tables 3 and 4. The symbol + represents significant difference

PSMA vs.	QIEA02	QIEA04	QIEA07	QIEA08	QEPS
Wilcoxon	6.1035e-5(+)	6.1035e-5(+)	6.1035e-5(+)	6.1035e-5(+)	6.1035e-5(+)
Friedman	0.0142(+)	0.0142 (+)	0.0142 (+)	0.0142 (+)	0.0142 (+)

4 Distribution System Reconfiguration

Power network reconfiguration is an important process in the improvement of operating conditions of a power system and in planning studies, service restoration and distribution automation when remote-controlled switches are employed [1, 15]. The optimal distribution system reconfiguration problem is to minimize the power loss of the system by changing the topology of distribution systems through altering the open/closed status of sectionalizing switches. Because there are many candidate-switching combinations in a distribution system, the distribution system reconfiguration is a complex combinatorial problem with a large number of integer and continuous variables and various constraints such as power flow equations, upper and lower bounds of nodal voltages, upper and lower bounds of line currents, feasible conditions in terms of network topology. As usual the problem can be formulated as a minimization cost function f [1, 23], i.e.,

$$\min f = \sum_{i=1}^{L} r_i \frac{P_i^2 + Q_i^2}{V_i^2} \tag{7}$$

Subject to

$$g(x) = 0 \tag{8}$$

$$V_{min} < V_n < V_{max} \tag{9}$$

$$I_i^{min} < I_i < I_i^{max} \tag{10}$$

$$\det(A) = 1 \text{ or } -1 (\text{for radial systems}) \tag{11}$$

$$\det(A) = 0 (\text{for not radial systems}) \tag{12}$$

where

f is the objective function (kW);
L is the number of branches;
P_i is the active power at sending end of branch i;
Q_i is the reactive power at sending end of branch i;
V_n is the voltage at node n;
I_i is the line current at branch i;
$g(x)$ is the power flow equations;
V_{min} and V_{max} are the lower and upper voltage limits, respectively;
I_i^{min} and I_i^{max} are the lower and upper current limits, respectively;
A is the bus incidence matrix;
r_i is the resistance of branch i.

The PSMA described above is used to solve the IEEE 33-bus and PG&E 69-bus distribution system reconfiguration problems. The IEEE 33-bus and PG&E 69-bus systems are shown in Fig. 5 and Fig. 6, respectively. Both of them are widely used as examples to test the performance of various optimization approaches. As shown in Fig. 5, the IEEE 33-bus system has 33 buses, 37 branches and 5 tie-lines. The normally open switches are 33, 34, 35, 36 and 37. The initial real power losses (before reconfiguration) are 202.68 kW. The PG&E 69-bus systems consists of 69 buses, 68 sectionalizing switches and 5 tie switches. The normally open switches are 69, 70, 71, 72 and 73. The initial real power losses (before reconfiguration) are 226.4419 kW. The algorithm for solving the distribution system reconfiguration problem by using PSMA is the same as the description in Section 2.3 except that Step 3 Evaluation considers (7) as the candidate solution criterion. In the experiment, PSMA uses 10 individuals as a population and 0.9 as the value of p_c. After 100 evolutionary generations, we obtain the optimal result reported in the literature. The experimental result of the IEEE 33-bus system is listed in Table 7, where results obtained by five optimization approaches, a heuristic approach [15], SA+TS [13], MTS [16], PSO [1] and ACO [23], reported in the recent literature, are also provided to be as a comparison. The experimental result from the PG&E 69-bus system is shown in Table 8, where ACS [8], HPSO [14], VSHDE [18] and ACO [23] are considered as benchmark optimization approaches.

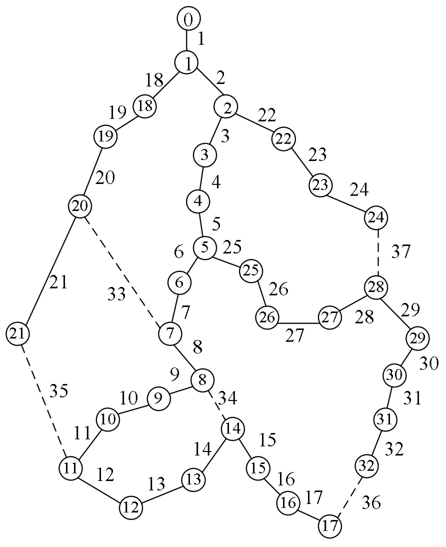


Fig. 5. IEEE 33-bus system

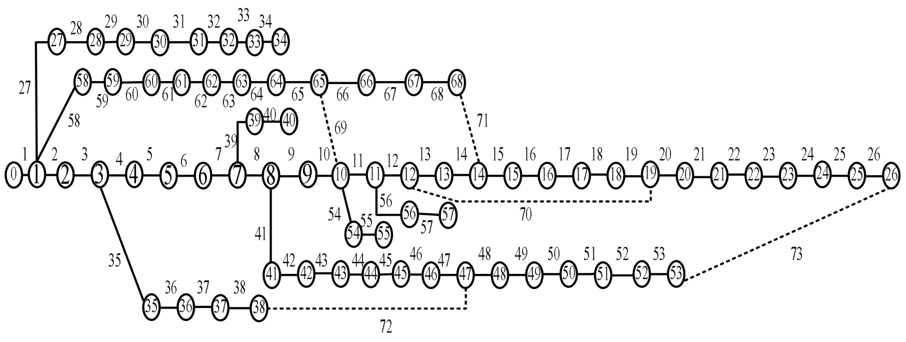


Fig. 6. PG&E 69-bus system

Table 7 shows that PSMA is competitive to the nine optimization approaches, a heuristic approach, SA+TS, MTS, PSO and ACO, due to the optimal solution obtained. The experimental results in Table 8 show that PSMA achieves lower real power losses and higher minimum node voltage than ACS, HPSO, VSHDE and IIGA. These results indicate that the better solution PSMA can obtain, the more complex the power system is.

Table 7. Results provided by PSMA for the IEEE 33-bus test system. MNV represents minimum node voltage

Methods	Optimal configuration	Real power loss (kW)	MNV (pu)
Before reconfiguration	33, 34, 35, 36, 37	202.68	0.9378
Heuristic approach [15]	7, 9, 14, 32, 37	139.55	0.9378
SA + TS [13]	7, 9, 14, 32, 37	139.55	0.9378
MTS [16]	7, 9, 14, 32, 37	139.55	0.9378
PSO [1]	7, 9, 14, 32, 37	139.55	0.9378
ACO [23]	7, 9, 14, 32, 37	139.55	0.9378
PSMA	7, 9, 14, 32, 37	139.55	0.9378

Table 8. Results provided by PSMA for the PG&E 69-bus test system. MNV represents minimum node voltage

Methods	Optimal configuration	Real power loss (kW)	MNV (pu)
Before reconfiguration	69, 70, 71, 72, 73	226.4419	0.9089
ACS [8]	61, 69, 14, 70, 55	99.519	0.943
HPSO [14]	69, 12, 14, 47, 50	99.6704	0.9428
VSHDE [18]	11, 24, 28, 43, 56	99.6252	0.9427
IIGA [26]	69, 14, 70, 47, 50	99.618	0.9427
PSMA	47, 12, 50, 14, 69	99.4944	0.9441

5 Conclusions

This paper is a continuous work on how to appropriately combine membrane computing models and evolutionary algorithms. This is the first attempt to use a population P system to design an approximate optimization algorithm. Extensively comparative experiments conducted on knapsack problems show that PSMA has a good performance with respect to the search capability and elapsed time. We also use PSMA to solve the distribution system reconfiguration problem in the area of power systems and experimental results are also attractive. Further work will focus on more and complex distribution system reconfiguration problems.

Acknowledgements

GZ and YQ acknowledges the support by the National Natural Science Foundation of China (61170016), the Program for New Century Excellent Talents in University (NCET- 11-0715) and SWJTU supported project (SWJTU12CX008), the Project-sponsored by SRF for ROCS, SEM, the Scientific and Technological Funds for Young Scientists of Sichuan (09ZQ026-040), the Fund for Candidates of Provincial Academic and Technical Leaders of Sichuan and the Fundamental Research Funds for the Central Universities (SWJTU09ZT10). MAGN acknowledges the support of the project TIN-2009-13192 of the Ministerio de Ciencia e Innovación of Spain and the support of the Project of Excellence with *Investigador de Reconocida Valía* of the Junta de Andalucía, grant P08-TIC-04200. MG acknowledges the support of CNCSIS-UE-FISCSU project number PNII-IDEI 643/2008.

References

1. Abdelaziz, A., Mohammed, F., Mekhamer, S., Badr, M.: Distribution systems reconfiguration using a modified particle swarm optimization algorithm. Electric Power Systems Research 79(11), 1521–1530 (2009)
2. Bernardini, F., Gheorghe, M.: Population P systems. Journal of Universal Computer Science 10(5), 509–539 (2004)
3. Cao, H., Romero-Campero, F.J., Heeb, S., Cámara, M., Krasnogor, N.: Evolving cell models for systems and synthetic biology. Systems and Synthetic Biology 4(1), 55–84 (2010)
4. Cheng, J., Zhang, G., Zeng, X.: A novel membrane algorithm based on differential evolution for numerical optimization. International Journal of Unconventional Computing 7(3), 159–183 (2011)
5. Escuela, G., Gutiérrez-Naranjo, M.A.: An application of genetic algorithms to membrane computing. In: Martínez del Amor, M.A., Păun, Gh., Pérez Hurtado, I., Riscos-Núñez, A. (eds.) Eighth Brainstorming Week on Membrane Computing. pp. 101–108. Fénix Editora, Sevilla, Spain (2010)
6. García, S., Molina, D., Lozano, M., Herrera, F.: A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. Journal of Heuristics 15(6), 617–644 (2009)
7. Garey, M.R., Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York (1979)
8. Ghorbani, M.A., Hosseinian, S.H., Vahidi B.: Application of ant colony system algorithm to distribution networks reconfiguration for loss reduction. In: Proceedings of the 11^{th} International Conference on Optimization of Electrical and Electronic Equipment, pp. 269–273 (2008).
9. Han, K.H., Kim, J.H.: Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. IEEE Transactions on Evolutionary Computation 6, 580–593 (2002)
10. Han, K.H., Kim, J.H.: Quantum-inspired evolutionary algorithms with a new termination criterion, H_ϵ gate, and two-phase scheme. IEEE Transactions on Evolutionary Computation 8(2), 156–169 (2004)
11. Huang, L., Suh, I.H.: Controller design for a marine diesel engine using membrane computing. International Journal of Innovative Computing, Information and Control 5(4), 899–912 (2009)
12. Huang, X.L., Zhang, G.X., Rong, H.N., Ipate, F.: Evolutionary design of a simple membrane system. In: Gheorghe, M., Păun, G., Rozenberg, G., Salomaa, A., Verlan, S. (eds.) International Conference on Membrane Computing. Lecture Notes in Computer Science, vol. 7184, pp. 203–214. Springer (2011)

13. Jeon, Y.J., Kim, J.C.: Application of simulated annealing and tabu search for loss minimization in distribution systems. International Journal of Electrical Power & Energy Systems 26(1), 9–18 (2004)
14. Li, Z.K., Chen, X.Y., Yu, K., Sun, Y., Liu, H.M.: A hybrid particle swarm optimization approach for distribution network reconfiguration problem. In: Proceedings of Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century, pp. 1–7 (2008).
15. Martín, J.A., Gil, A.J.: A new heuristic approach for distribution systems loss reduction. Electric Power Systems Research 78(11), 1953–1958 (2008)
16. Mekhamer, S., Abdelaziz, A., Mohammed, F., Badr, M.: A new intelligent optimization technique for distribution systems reconfiguration. In: Proceedings of the 12^{th} International Middle-East Power System Conference, 2008. MEPCON 2008. pp. 397–401. IEEE (2008)
17. Nishida, T.Y.: Membrane algorithm with brownian subalgorithm and genetic subalgorithm. International Journal of Foundations of Computer Science 18(6), 1353–1360 (2007)
18. Nournejad, F., Kazemzade, R., Yazdankhah, A.S.: A multiobjective evolutionary algorithm for distribution system reconfiguration. In: Proceedings of the 16_{th} Conference on Electrical Power Distribution Networks, pp. 1–7 (2011).
19. Suzuki, Y., Fujiwara, Y., Takabayashi, J., Tanaka, H.: Artificial life applications of a class of P systems: Abstract rewriting systems on multisets. In: Calude, C., Păun, G., Rozenberg, G., Salomaa, A. (eds.) Workshop on Multiset Processing. Lecture Notes in Computer Science, vol. 2235, pp. 299–346. Springer, Berlin Heidelberg (2001)
20. Suzuki, Y., Tanaka, H.: Chemical evolution among artificial proto-cells. In: Bedau, M.A., McCaskill, J.S., Rasmussen, S. (eds.) Artificial Life VII: Proceedings of the Seventh International Conference on Artificial Life. pp. 54–63. MIT Press, Cambridge, MA, USA (2000)
21. Suzuki, Y., Tanaka, H.: Computational living systems based on an abstract chemical system. In: Proceedings of the 2000 Congress on Evolutionary Computation. pp. 1369–1376. IEEE (2000)
22. Suzuki, Y., Tanaka, H.: Modeling p53 signaling pathways by using multiset processing. In: Ciobanu, G., Păun, G., Pérez-Jiménez, M.J. (eds.) Applications of Membrane Computing, pp. 203–214. Natural Computing Series, Springer Berlin Heidelberg (2006)
23. Swarnkar, A., Gupta, N., Niazi, K.: Efficient reconfiguration of distribution systems using ant colony optimization adapted by graph theory. In: Power and Energy Society General Meeting, 2011 IEEE. pp. 1–8. IEEE (2011)
24. Tudose, C., Lefticaru, R., Ipate, F.: Using genetic algorithms and model checking for P systems automatic design. In: Pelta, D.A., Krasnogor, N., Dumitrescu, D., Chira, C., Lung, R.I. (eds.) NICSO. Studies in Computational Intelligence, vol. 387, pp. 285–302. Springer (2011)
25. Vlachogiannis, J., Lee, K.: Quantum-inspired evolutionary algorithm for real and reactive power dispatch. IEEE Transactions on Power Systems 23(4), 1627–1636 (2008)
26. Wang, C.X., Zhao, A.J., Dong, H., Li, Z.J.: An improved immune genetic algorithm for distribution network reconfiguration. In: Proceedings of International Conference on Information Management, Innovation Management and Industrial Engineering, 218–223 (2009).

27. Xiao, J.H., Zhang, X.Y., Xu, J.: A membrane evolutionary algorithm for DNA sequence design in DNA computing. Chinese Science Bulletin 57(6), 698–706 (2012)
28. Zhang, G.X.: Quantum-inspired evolutionary algorithms: a survey and empirical study. Journal of Heuristics 17(3), 303–351 (2011)
29. Zhang, G.X., Cheng, J.X., Gheorghe, M.: A membrane-inspired approximate algorithm for traveling salesman problems. Romanian Journal of Information Science and Technology 14(1), 3–19 (2011)
30. Zhang, G.X., Li, Y.Q., Gheorghe, M.: A membrane algorithm with quantum-inspired subalgorithms and its application to image processing, Natural Computing DOI: 10.1007/s11047-012-9320-2 (2012)
31. Zhang, G.X., Gheorghe, M., Wu, C.Z.: A quantum-inspired evolutionary algorithm based on P systems for knapsack problem. Fundamenta Informaticae 87(1), 93–116 (2008)
32. Zhang, G.X., Liu, C.X., Gheorghe, M.: Diversity and convergence analysis of membrane algorithms. In: Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), 2010 IEEE. pp. 596–603 (2010)
33. Zhang, G.X., Liu, C.X., Gheorghe, M., Ipate, F.: Solving satisfiability problems with membrane algorithms. In: Fourth International Conference on Bio-Inspired Computing, 2009. BIC-TA '09. pp. 29–36 (2009)
34. Zhang, G.X., Liu, C.X., Rong, H.N.: Analyzing radar emitter signals with membrane algorithms. Mathematical and Computer Modelling 52(11-12), 1997–2010 (2010)
35. Zhang, G.X., Liu, C.X., Gheorghe, M., Ipate, F.: An approximate algorithm using P systems with active membranes. Mathematics and Computers in Simulation (2012), Available: `http://staffwww.dcs.shef.ac.uk/people/M.Gheorghe/research/paperlist.html`
36. Zhang, G.X., Gheorghe, M., Cheng, J.X.: Dynamic behavior analysis of membrane algorithms. MATCH Communications in Mathematical and in Computer Chemistry (2012), Accepted paper.
37. Zhang, H., Zhang, G.X., Rong, H.N., Cheng, J.X.: Comparisons of quantum rotation gates in quantum-inspired evolutionary algorithms. In: Sixth International Conference on Natural Computation (ICNC), 2010. pp. 2306–2310. IEEE (2010)
38. Zhang, R., Gao, H.: Improved quantum evolutionary algorithm for combinatorial optimization problem. In: International Conference on Machine Learning and Cybernetics, 2007. pp. 3501–3505. IEEE (2007)