A Population-P-Systems-Inspired Membrane Algorithm for Multi-objective Optimization

Jixiang Cheng[1], Gexiang Zhang[1], and Yanhui Qin[2]

[1]School of Electrical Engineering, Southwest Jiaotong University, Chengdu 610031, P.R. China
[2]State Grid Xinjiang Electric Power Corporation Electric Power Research Institute, Urumqi, 830011, P.R. China
{chengjixiang0106,zhgxdylan}@126.com

Abstract. This paper proposes a Population-P-Systems-inspired Membrane Algorithm (PPSMA) for multi-objective optimization. In the algorithm, the cells of population P systems are divided into two groups to implement different functions and the communications among cells are performed at two levels in order to obtain well converged and distributed solution set. Moreover, differential evolution is employed as search operator in PPSMA. Twelve multi-objective benchmark problems are utilized to test algorithm performance. Experimental results show that PPSMA performs better than five compared algorithms.

Keywords: Membrane Algorithm, Population P Systems, Multi-objective Optimization.

1 Introduction

In recent years, membrane algorithms (MAs), which hybridize membrane computing with evolutionary computation, have attracted much attention in both methodological aspects and practical applications. At present, several membrane structures, such as nested membrane structure [1], one level membrane structure [2], dynamic membrane structure [3], statistic network structure [4] and dynamic network structure [5], have been combined with various evolutionary algorithms to design many MAs. A comprehensive review of MAs can be found in the survey paper by Zhang et, al. [6].

To date, the research on MAs mainly focuses on the use of structures of cell-like P systems. In our previous work [5], population P systems [7] were applied to construct a MA for single-objective optimization. In this paper, we push the work forward and propose a Population-P-Systems-inspired MA for multi-objective optimization called PPSMA. In the algorithm, population P systems are used to organize individuals; cells in the system are divided into two groups with different functions; communications among cells are performed at local and global levels; and differential evolution (DE)[8] is applied as search operator. Experimental results show that PPSMA outperforms five peer algorithms in terms of three performance metrics.

2 PPSMA

To obtain uniformly distributed solutions, we can first define a set of uniformly distributed directions and then search for one solution that converges well along each

L. Pan et al. (Eds.): BIC-TA 2014, CCIS 472, pp. 40–44, 2014.

direction. Hence, the problem is transformed to a set of sub-problems with different goals. To ensure good convergence performance, efficient search operators like DE should be applied and the information of a sub-problem should be shared by others. Population P systems are quite suitable for solving multi-objective optimization problems (MOPs). In PPSMA, the cells are divided into two groups. The first group called evolving cells, including cells C_k, $k = 1,2,\ldots,NP$, focuses on searching individuals toward Pareto front; while the second group called surviving cell, consisting of only one cell C_{NP+1}, aims at surviving individuals and re-scattering them across the evolving cells for next generation, where NP is the population size. Hence, there are $NP+1$ cells involved in the system.

In PPSMA, we first generate NP uniformly distributed direction vectors, one for each evolving cell, and then define the neighboring cells of each evolving cell. Specifically, each direction vector $\mathbf{r} = (r_1, r_2, \ldots, r_M)$ should satisfy

$$r_m \in \left\{\tfrac{0}{H}, \tfrac{H}{H}, \ldots, \tfrac{H}{H}\right\}, m = 1,2,\ldots,M, \sum_{m=1}^{M} r_m = 1 \tag{1}$$

where M is the number of objectives and H is a positive integer. The total number of available direction vectors is C_{H+M-1}^{M-1}. Then, each evolving cell i is associated with a unique direction vector, denoted as $\mathbf{r}_i$. The neighboring cells of evolving cell i are the evolving cells whose direction vectors are the $\lfloor R \cdot NP \rfloor$ closest vectors to $\mathbf{r}_i$, denoted as $N(i)$, where $R \in (0,1)$ is a pre-defined parameter.

Based on the above description, the population P systems in PPSMA can be formulated as the following construct

$$P = (V, \gamma, \alpha, \omega_e, C_1, C_2, \ldots, C_{NP}, C_{NP+1}, c_o)$$

where

(i) $V = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{NP}\}$, where $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$ is a real-valued string;
(ii) $\gamma = (\{1,2,\ldots,NP+1\}, E)$ with $E = E_1 \cup E_2$ is a finite directed graph, where $E_1 = \{(i,j) | 1 \le i \le NP, j \in N(i)\}$, $E_2 = \{(i,j) | 1 \le i \le NP, j = NP+1, \text{or } i = NP+1, 1 \le j \le NP\}$;
(iii) α is a finite set of bond making rule $(i, x_1; x_2, j), (i,j) \in E$;
(iv) $\omega_e = \lambda$;
(v) $C_k = (\omega_k, S_k, R_k)$, for each evolving cell k, $1 \le k \le NP$,
  (a) $\omega_k = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n_k}\}$, where n_k is the number of individuals in cell C_k, satisfying $\sum_{k=1}^{NP} n_k = NP$;
  (b) S_k is a finite set of communication rules of the form $(\lambda; b, in), b \in V$;
  (c) R_k is a finite set of transformation rules of the form $x \to y$, consisting of the mutation, crossover, and selection operators of DE;
  and for surviving cell k, $k = NP+1$,
  (a) $\omega_k = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{NP}\}$;
  (b) S_k is a finite set of communication rules of the forms $(\lambda; b, in)$ and $(b, exit)$, $b \in V$;
  (c) $R_k = \emptyset$;
(vi) $c_o = NP+1$ is the label of the output cell.

In what follows we describe the procedures of PPSMA step by step.

Step 1: *Initialization*. A population P system with $NP+1$ cells is created. Each evolving cell k is associated with a direction vector $\mathbf{r}_k$. An initial population with NP individuals is generated and each individual is sent to a random selected evolving cell. Hence, each evolving cell has n_k individuals with $0 \leq n_k \leq NP$ and $\sum_{k=1}^{NP} n_k = NP$.

Step 2: *Parameter setting*. In PPSMA, DE is applied as search operator. As scaling factor F and crossover rate Cr of DE have great influence on algorithm performance, we adapt the two parameters according to their recent successful values using the strategy in [9]. At generation g, the values F_i and Cr_i for the ith individual are generated by

$$F_i = randc(\mu_F, 0.1), \mu_F = \sum_{F \in sF} F^2 \Big/ \sum_{F \in sF} F \tag{2}$$

$$Cr_i = randn(\mu_{Cr}, 0.1), \mu_{Cr} = \sum_{Cr \in sCr} Cr \Big/ W \tag{3}$$

where, $randc(\mu_F, 0.1)$ is a cauchy distribution with location parameter μ_F and scale parameter 0.1; $randn(\mu_{Cr}, 0.1)$ is a normal distribution of mean μ_{Cr} and standard deviation 0.1; sF and sCr are the sets of most recent W values of F and Cr which produce offspring entering the next generation.

Step 3: *Individual evolution*. Transformation rules of the form $x \rightarrow y$ is utilized to evolve the objects in each of non-empty evolving cells (i.e., $n_k \neq 0$). The rules are the mechanisms of DE. Instead of evolving individuals in each cell independently, individuals from neighboring cells are used through local communication. Specifically, for each individual $\mathbf{x}_i$ in an evolving cell $k, i = 1, 2, \ldots, n_k$, three individuals $\mathbf{x}_{r1}$, $\mathbf{x}_{r2}$ and $\mathbf{x}_{r3}$ randomly selected from evolving cells k_{r1}, k_{r2} and k_{r3} are used to generate a trail vector $\mathbf{v}_i$, where k_{r1}, k_{r2} and $k_{r3} \in \{k\} \bigcup \{N(k) | n_k \neq 0\}$. If $\mathbf{f}(\mathbf{v}_i)$ dominates $\mathbf{f}(\mathbf{x}_i)$, $\mathbf{x}_i$ is replaced by $\mathbf{v}_i$; otherwise, $\mathbf{x}_i$ is kept unchanged and $\mathbf{v}_i$ is stored in the current cell temporally. The values of k_{r1}, k_{r2} and k_{r3} determine what membrane structure will be created and used to perform local communication.

Step 4: Global communication. In this step, surviving cell c_o receives all individuals (including $\mathbf{x}_i$ and temporally stored $\mathbf{v}_i$) from all evolving cells using rule $(\lambda; b, in)$, forming a temporary population P' with size N, $N \in [NP, 2NP]$. The population P' is then divided into fronts $\mathscr{F}_1, \mathscr{F}_2, \ldots, \mathscr{F}_L$ through non-dominated sorting. The individuals in $\mathscr{F}_1, \mathscr{F}_2, \ldots, \mathscr{F}_{l-1}, l \leq L$ are first survived, satisfying $\sum_{i=1}^{l-1} |\mathscr{F}_i| \leq NP$ and $\sum_{i=1}^{l} |\mathscr{F}_i| > NP$. Then for each individual $\mathbf{x}$ in fronts $\mathscr{F}_1, \mathscr{F}_2, \ldots, \mathscr{F}_l$, we determine an evolving cell k that $\mathbf{x}$ could enter according to the closest perpendicular distance between $\mathbf{f}(\mathbf{x})$ and $\mathbf{r}_i$, i.e.,

$$k = \underset{i \in \{1,\ldots,NP\}}{\operatorname{argmin}} \; d\left(\mathbf{f}(\mathbf{x}), \mathbf{r}_i\right) = \underset{i \in \{1,\ldots,NP\}}{\operatorname{argmin}} \; \mathbf{f}(\mathbf{x}) - \mathbf{r}_i^T \mathbf{f}(\mathbf{x}) \mathbf{r}_i \tag{4}$$

Thirdly, we send all individuals in $\mathscr{F}_1, \mathscr{F}_2, \ldots, \mathscr{F}_{l-1}$ to corresponding cells. Afterwards, we scan all evolving cells in ascending order of the number of individuals in evolving cells. Specifically, if an evolving cell has the fewest individuals in it and there exists an individual in F_l that could enter into this evolving cell, the corresponding individual is sent to this cell and the number of individuals in this cell is incremented by 1; otherwise, the next evolving cell with the fewest individuals is checked. If all evolving cells with

the fewest individuals have been checked and there are still some individuals need to be survived, then the evolving cells with the second fewest individuals are checked. The scanning procedure is repeated until $NP - \sum_{i=1}^{l-1} |\mathscr{F}_i|$ individuals have been survived from $\mathscr{F}_l$. At the end of re-scattering procedure, the most recently adopted F_i and Cr_i values which produce the survived individuals are stored in sF and sCr for updating μ_F and μ_{Cr} at next generation.

Step 5: Termination condition. The algorithm stops when a prescribed number of evolutionary generations or function evaluations is attained. Otherwise, go to Step 2.

Step 6: Output. For cell c_o, rule $(\lambda; b, in)$ is performed to collect all individuals from evolving cells. These individuals constitute the final solution set.

3 Experimental Results

Twelve benchmark MOPs, including five two-objective problems (ZDT1-ZDT4, ZDT6) and seven three-objective problems (DTLZ1-DTLZ7) are employed to conduct experiment. To quantify algorithm performance, three performance metrics including generational distance (GD), inverted generational distance (IGD) and hypervolume (HV) are used. To show the advantage of PPSMA, five compared algorithms are considered, i.e., NSGA-II [10], GDE3 [11], DEMO [12], ε-MyDE [13] and MOEA/D-DE [14]. The maximal number of function evaluations is set to 30,000 as the stopping criterion. The population size NP is set to 100, except for PPSMA and MOEA/D-DE when solving three-object problems, where NP is set to 105 because it is impossible to generate exact 100 uniformly distributed weight vectors or direction vectors for the two algorithms when solving three-objective problems. In PPSMA, R and W are set to 0.3 and 80, respectively, according to a parameter analysis procedure. The other parameters for each compared algorithms are set the same as in their original papers. In addition, 25 independent runs for each algorithm on each test problem are performed.

Due to space limitation, we do not list the mean and standard deviation values of GD, IGD and HV. Instead, we give the statistical test results with respect to Wilcoxon's rank sum test at a 0.05 significance level, shown in Table 1. From the table, we can see that the numbers of problems where PPSMA performs significantly better than five compared algorithms are much larger than the number of problems where PPSMA performs significantly worse, which demonstrates PPSMA is superior to five compared algorithms.

Table 1. Statistical test results in terms of GD, IGD and HV

Algorithms	GD			IGD			HV		
	+	=	−	+	=	−	+	=	−
NSGA-II	11	0	1	10	1	1	9	1	2
GDE3	8	2	2	7	2	3	8	2	2
DEMO	6	2	4	7	1	4	8	2	2
ε-MyDE	9	0	3	10	1	1	10	1	1
MOEA/D-DE	12	0	0	11	1	0	11	1	0

4 Conclusion

This paper proposed a multi-objective MA called PPSMA. In the algorithms, two kinds of cells with different functions are used and the communications are performed at both local and global levels. Experimental results show that PPSMA outperforms five peer algorithms in terms of three performance metrics. Our future work will focus on designing multi-objective MAs by exploiting the features of other advanced P systems and applying these algorithms to solving real-word problems.

Acknowledgement. This work was supported by the National Natural Science Foundation of China (No. 61170016, No.61373047), the Program for New Century Excellent Talents in University (No.NCET-11-0715) and SWJTU supported project (No.SWJTU12CX008), and the 2014 Doctoral Innovation Funds of southwest Jiaotong University.

References

1. Nishida, T.Y.: An Approximate Algorithm for NP-complete Optimization Problems Exploiting P Systems. In: BWMC, pp. 185–192 (2004)
2. Zhang, G., Gheorghe, M., Wu, C.Z.: A Quantum-Inspired Evolutionary Algorithm based on P Systems for Knapsack Problem. Fund. Inform. 87, 93–116 (2008)
3. Liu, C., Zhang, G., Liu, H., Gheorghe, M., Ipate, F.: An Improved Membrane Algorithm for Solving Time-Frequency Atom Decomposition. In: Păun, G., Pérez-Jiménez, M.J., Riscos-Núñez, A., Rozenberg, G., Salomaa, A. (eds.) WMC 2009. LNCS, vol. 5957, pp. 371–384. Springer, Heidelberg (2010)
4. Zhang, G., Cheng, J., Gheorghe, M., Meng, Q.: A Hybrid Approach based on Differential Evolution and Tissue Membrane Systems for Solving Constrained Manufacturing Parameter Optimization Problems. Appl. Soft Comput. 13, 1528–1542 (2013)
5. Zhang, G., Rong, H., Cheng, J., Qin, Y.: A Population-Membrane-System-Inspired Evolutionary Algorithm for Distribution Network Reconfiguration. Chinese J. Electron. 23, 437–441
6. Zhang, G., Gheorghe, M., Pan, L., Pérez-Jiménez, M.J.: Evolutionary Membrane Computing: A Comprehensive Survey. Inform Sciences 279, 528–551 (2014)
7. Bernardini, F., Gheorghe, M.: Population P systems. J. Univers. Comput. Sci. 10, 509–539 (2004)
8. Storn, R., Price, K.: Differential evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. J. Global Optim. 11, 341–359 (1997)
9. Zhang, J.Q., Sanderson, A.C.: JADE: Adaptive Differential Evolution with Optional External Archive. IEEE T. Evolut. Comput. 13, 945–958 (2009)
10. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE T. Evolut. Comput. 6, 182–197 (2002)
11. Kukkonen, S., Lampinen, J.: GDE3: The Third Evolution Step of Generalized Differential Evolution. In: IEEE CEC, pp. 443–450 (2005)
12. Robic, T., Filipic, B.: DEMO: Differential Evolution for Multiobjective Optimization. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 520–533. Springer, Heidelberg (2005)
13. Deb, K., Mohan, M., Mishra, S.: Evaluating the ϵ-domination based Multi-objective Evolutionary Algorithm for a Quick Computation of Pareto-optimal Solutions. Evol. Comput. 13, 501–525 (2005)
14. Li, H., Zhang, Q.F.: Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. IEEE T. Evolut. Comput. 13, 284–302 (2009)