An Evolutionary Algorithm Based on Decomposition for Multimodal Optimization Problems

Fangqing Gu
Department of Computer Science
Hong Kong Baptist University
Hong Kong SAR, China
Email: fqgu@comp.hkbu.edu.hk

Yiu-ming Cheung
Department of Computer Science
Hong Kong Baptist University
Hong Kong SAR, China
Email: ymc@comp.hkbu.edu.hk

Jie Luo
School of Physics & Telecommunication
Engineering, South China Normal University
Guangzhou, China
Email: 406843397@qq.com

***Abstract*—This paper presents a non-parameter method to identify the peaks of the multi-modal optimization problems provided that the peaks are characterized by a smaller objective values than their neighbors and by a relatively large distance from points with smaller objective value. Using the identified peaks as the seeds, we decompose the population into some subpopulations and dynamically allocate the computational effort to different subpopulations. We evaluate the proposed approach on the CEC2015 single objective multi-niche optimization problems. The promising experimental results show its efficacy.**

I. Introduction

An optimization problem can be defined as follows:

$$\begin{aligned}&\min_{\mathbf{x}} f(\mathbf{x})\\ &s.t.\ \mathbf{x}=(x_1,\ldots,x_D)\in\prod_{i=1}^{D}\left[l_i,u_i\right],\end{aligned}\tag{1}$$

where $\prod_{i=1}^{D}\left[l_i,u_i\right]$ is the decision space and D is the dimension of **x**. l_i and u_i are the lower and upper boundary of x_i, respectively, for $i=1,\ldots,D$. Such a problem with multiple local/global optima, namely multimodal optimization problem, has been widely studied in [1], [2], [3], [4], [5]. The evolutionary algorithm is a promising tool for multimodal optimization problems because it simultaneously finds all solutions using a population. However, the classical evolutionary algorithms fail to locate all the local/global optima. To overcome the drawback of the classical evolutionary algorithm, a number of techniques have been presented to maintain high diversity for locating multiple optima in a single run. These algorithms can be roughly divided into two categories: The first one maintains the diversity in the population by the crowding distance, such as the sharing function methods [6], [3] and Bi-objective approaches [7], [8], [9]. All the individuals are punished by the crowding distance in such an algorithm. It is clear that it is not good at protecting the local optima. The second one addresses the problem by dividing the population into some subpopulations, such as the species methods [1], [10], [2] and the classifier-based algorithm [11]. The species methods separate the population by appointing a radius, and all individuals within a given threshold distance form a species. Usually, it is difficult to select the value of the radius. Furthermore, paper [11] has presented a Gaussian classifier-based evolutionary strategy to solve the multimodal optimization problems. It has provided better and consistent performance on the benchmark functions for CEC2013 [12]. However, the clustering procedure of this algorithm only considers the distribution of the individuals in the decision space. It does not consider the objective value and the shape of the basin, which may lead to clustering the individuals within the different basins into a class.

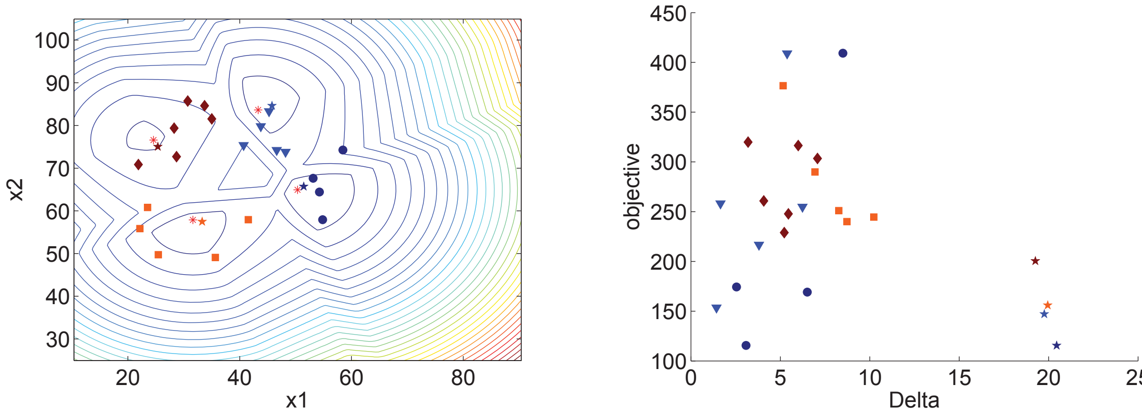
In this paper, we will propose a non-parameter method to identify the peaks of the multimodal optimization problems provided that the peaks are characterized by a smaller objective value and by a relatively large distance from points with smaller objective value. A similar idea has also been adopted in clustering [13]. Using the identified peaks as the seeds, we will decompose the population into a number of subpopulations. Such decomposition strategy considers not only the distribution of the individuals in the decision space, but also their objective values. Due to the different computational difficulties of the different peaks, a dynamic allocating strategy is therefore presented to assign different amounts of computational effort to different subpopulations. We test the proposed algorithm in the CEC2015 competition on single objective multi-niche optimization [14] to evaluate the performance of the proposed algorithm.

The remainder of this paper is organized as follows: Section II gives a detailed description of the proposed algorithm. Section III tests the proposed algorithm on the CEC2015 test instances. The results show the efficacy of the proposed algorithm. Section VI concludes this paper.

II. The Proposed Evolutionary Algorithm based on Decomposition

In this section, we systematically introduce the proposed algorithm as follows.

978-1-4799-7492-4/15/$31.00 ©2015 IEEE



(a) The contour of the function.

(b) The plots of the objective value versus δ

Fig. 1. The seeds identification in two dimensions.

A. Decompose the Population into Some Subpopulations

We first present a novel strategy to identify the seeds for the subpopulations. This mechanism is based on the idea that the peaks are characterized by a smaller objective value and by a relatively large distance from points with smaller objective value. Specifically, we first sort the individuals according to the value of the objective in ascending order. Let $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N'}\}$ be the set of the sorted individuals, where $N' \geq N$ is the number of the individuals and N is the population size. Then, δ_i is measured by computing the minimum distance between $\mathbf{x}_i$ and any other individuals with smaller objective value than $\mathbf{x}_i$. That is

$$\delta_i = \min_{j<i} d(\mathbf{x}_i, \mathbf{x}_j), \tag{2}$$

where $d(\mathbf{x}_i, \mathbf{x}_j)$ is the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$. For the individual with smallest objective value, i.e. $\mathbf{x}_1$, we take

$$\delta_1 = \max_{j} \delta_j + 0.1. \tag{3}$$

Obviously, δ_i is much greater than the typical nearest neighbor distance for the peaks. Thus, we can use δ to identify the peaks of the multimodal optimization problems. It is illustrated by an example in Fig. 1. Figure 1(a) plots the counter of the objective function, in which the red stars represent the peaks. Some points randomly generated in the domain space are shown in Fig. 1(a). We compute the value of δ_i for each point by the above-mentioned method and plot the value of the objective versus δ in Fig. 1(b). From this figure, we can see that four points, represented as pentagram, have a greater value of δ. They are the points close to the peaks in the decision space as shown in Fig. 1(a). Therefore, we can consider these points as the seeds. In this paper, we regard the individuals, whose objective value and the value of δ are all in top 30 percent of the population, as the seeds. That is, for individual $\mathbf{x}_i$, if $f(\mathbf{x}_i) < f_{ref}$ and $\delta_i > \delta_{ref}$, it is regarded as a seed, where f_{ref} is the objective value of the $\lfloor 0.3N \rfloor$th individual sorted by the objective value in ascending order and δ_{ref} is the value of δ of the $\lfloor 0.3N \rfloor$th individual sorted by the value of δ in descending order.

Suppose M seeds have been found, the remaining individual is assigned to the same subpopulation as its nearest neighbor with a smaller objective value. Then, we remove $N' - N$ individuals with the smallest value of δ from the population. The subpopulation i is formed by the seed i and the surviving individuals are assigned to this subpopulation. There may be some subpopulations having one individual only, i.e. the seed. As shown in Fig. 1, the points with the same color are assigned into a subpopulation. From Fig. 1(a), we can find that the individuals in a subpopulation are generally surrounding a peak. Such a decomposition features that: (a) it is a non-parametric decomposition technique, and (b) ideally the individuals assigned into the same subpopulation are located within a basin because it considers the objective value in the decomposition procedure.

B. Crossover and Mutation

1) Parents Selection: Generally, different peaks may have different computational difficulties. We define a utility π_i for each subpopulation i and set the initial $\pi_i = 1$ for $i = 1, \ldots, M$. Computational efforts are assigned to these subpopulations based on their utilities. That is, the subpopulation involved to generate new individual is selected by using the roulette selection based on the utilities. Suppose subpopulation i is selected to involve in the crossover and mutation operation. A new individual is generated by two parents using the crossover and mutation operation presented in [15]. The first parent $\mathbf{x}_{p1}$ is the seed of the subpopulation i. There are two cases for choosing the other individual $\mathbf{x}_{p2}$ involved in crossover. In the case of the subpopulation with only one individual, i.e. the seed, we randomly generate a point in the area centered at $\mathbf{x}_{p1}$ with the radius δ_{ref} to represent $\mathbf{x}_{p2}$. Otherwise, it is randomly selected from the individuals in this subpopulation except the seed.

2) Crossover Operation: The crossover and mutation operations are used as in [15]. That is, we perform the crossover operation between $\mathbf{x}_{p1}$ and $\mathbf{x}_{p2}$. The new individual $\mathbf{v} = (v_1, v_2, \cdots, v_D)$ is generated as:

$$v_j = x_{p1j} + rc.(x_{p1j} - x_{p2j}),\ j = 1, 2, \cdots, D \qquad (4)$$

where

$$rc = r_1 \left(1 - rand^{(1-\frac{i\_eval}{n\_eval})^{0.7}}\right) \qquad (5)$$

is the search step size, r_1 is a random number within the range of $[-1, 1]$, $rand$ is a random number within the range of $[0, 1]$, i_eval is the number of the current function evaluations, and n_eval is the maximum number of function evaluations. From equation (5), we know that rc tends to 0 as i_eval tends to n_eval, analogous to simulated annealing algorithm gradually decreasing the search range. The newly generated individual is repaired as follows:

Suppose the kth component v_k of v is out of the boundary. If v_k is smaller than the low boundary l_k, we let

$$v_k = l_k + 0.5rand(l_k - v_k).$$

If v_k is greater than the upper boundary u_k, we let

$$v_k = u_k - 0.5rand(v_k - u_k).$$

3) Mutation Operation: After crossover, every component of v is mutated with the probability of p_m, which is set at $\frac{1}{D}$ in this paper. Then, the new individual $\tilde{\mathbf{x}} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_D)$ is generated as

$$\tilde{x}_k = \begin{cases} v_k + rm(u_k - l_k) & rand < p_m \\ v_k & otherwise \end{cases} \qquad (6)$$

with $k = 1, 2, \cdots, D$, where

$$rm = 0.15r_1 \left(1 - rand^{(1-\frac{i\_eval}{n\_eval})^{0.7}}\right).$$

The definition of rm is similar to rc. Subsequently, the new individual $\tilde{\mathbf{x}}$ is repaired as: When $\tilde{x}_k < l_k$, $\tilde{x}_k = l_k + 0.5rand(\tilde{x}_k - l_k)$; When $\tilde{x}_k > u_k$, $\tilde{x}_k = u_k - 0.5rand(u_k - \tilde{x}_k)$. $\tilde{\mathbf{x}}$ is the new individual after crossover and mutation.

C. Update the Subpopulation

For the newly generated individual $\tilde{\mathbf{x}}$, we only update the subpopulation i of its parents. If $\tilde{\mathbf{x}}$ is better than the seed of subpopulation i, i.e. the value of the objective of $\tilde{\mathbf{x}}$ is smaller than that of the seed, we directly add it to this subpopulation. It can protect the seed and avoid deleting the local optima because we only consider the objective value in such updating procedure. Otherwise, if $\tilde{\mathbf{x}}$ is better than the worst individual in this subpopulation, it replaces the worst one.

Then, the utility π_i of the subpopulation i is updated as follows: If $\tilde{\mathbf{x}}$ is better than the seed of subpopulation i, we should assign more computational effort to this subpopulation because it can generate good offspring. Therefore, we set $\pi_i = \pi_i + 1$; If $\tilde{\mathbf{x}}$ is better than the worst one of this subpopulation, let $\pi_i = \pi_i + 0.1$; Otherwise, $\pi_i = max(\pi_i - 0.5, 0.5)$.

Algorithm 1: The Steps of the Proposed Algorithm

input :
- The test instance;
- N: the population size;
- n_eval: the maximum number of the function evaluations.

output: The seed of each subpopulation.

Step 1 Initialization
- ◇ Step 1.1 Set the maximum number of the individuals in the population $N' = \lfloor 1.1N \rfloor$;
- ◇ Step 1.2 Generate N' initial individuals $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N'}$ by uniformly and randomly sampling from the domain space.
- ◇ Step 1.3 Set $i\_eval = N'$, $ls\_eval = \lfloor 0.8n\_eval \rfloor$.

Step 2 Decomposition
- ◇ Step 2.1 Identify the M seeds in the individuals by the above-mentioned method;
- ◇ Step 2.2 Assign the remaining individuals into the subpopulation;
- ◇ Step 2.3 Remove the redundant $N' - N$ individuals with smallest value of δ;
- ◇ Step 2.4 Set the utility $\pi_i = 1$ for $i = 1, 2, \ldots, M$;
- ◇ Step 2.5 Set $reg = 0$ and $flag = N$.

Step 3 Evolution
- ◇ Step 3.1 Select the subpopulation i for search based on the utilities;
- ◇ Step 3.2 Perform the crossover and mutation operation to generate $\tilde{x}$;
- ◇ Step 3.3 Update the subpopulation;

  if $\tilde{x}$ *is better than the seed* **then**
  Add $\tilde{\mathbf{x}}$ to subpopulation i;
  $flag = flag + 1$; $\pi_i = \pi_i + 1$.
  else if $\tilde{\mathbf{x}}$ *is better than the worst one* **then**
  $\tilde{\mathbf{x}}$ replaces the worst one;
  $\pi_i = \pi_i + 0.1$.
  else
  $\pi_i = max(\pi_i - 0.5, 0.5)$
  end
  $reg = reg + 1$; $i\_eval = i\_eval + 1$;
  if $i\_eval > ls\_eval$ **then**
  go to **Step 4**
  else if $flag == N'$ *or* $reg == 2 * N$ **then**
  go to **Step 2**
  else
  go to **Step 3**
  end

Step 4 Local Search
if $i\_eval < n\_eval$ **then**
Perform local search on the seeds.
else
Stop and output the seed of each subpopulation
end

D. Local Search

Generally, the heuristic algorithm, such as evolutionary algorithm, is good at exploration, but not exploitation. That is, its convergence speed is slow as the solution is close to the optima. Therefore, after a given number of function evaluations, we perform a one-available local search strategy for the seeds.

E. The Algorithmic Steps of the Proposed Algorithm

The details of the proposed algorithm is shown in Algorithm 1, in which i_eval is the current number of the function evaluations, ls_eval is the number of function evaluation to start performing the local search, reg is the number of function in each period, and $flag$ is the number of the individuals in the population.

III. Experimental Results

The experiments were performed on a 3.40GHz Intel PC with 16G RAM. The programming language is MATLAB.

TABLE I
THE CEC2015 MULTI-NICHE TEST FUNCTIONS

No.	Function	Dim.	Global/Local optima
f1	Shifted and Rotated Expanded Two-Peak Trap	5 10 20	1/15 1/55 1/210
f2	Shifted and Rotated Expanded Five-Uneven-Peak Trap	2 5 8	4/21 32/0 256/0
f3	Shifted and Rotated Expanded Equal Minima	2 3 4	25/0 125/0 625/0
f4	Shifted and Rotated Expanded Decreasing Minima	5 10 20	1/15 1/55 1/210
f5	Shifted and Rotated Expanded Uneven Minima	2 3 4	25/0 125/0 625/0
f6	Shifted and Rotated Expanded Himmelblaus Function	4 6 8	16/0 64/0 256/0
f7	Shifted and Rotated Expanded Six-Hump Camel Back	6 10 16	8/0 32/0 256/0
f8	Shifted and Rotated Modified Vincent Function	2 3 4	36/0 216/0 1296/0
f9	Composition Function 1	10,20,30	10/0
f10	Composition Function 2	10,20,30	1/9
f11	Composition Function 3	10,20,30	10/0
f12	Composition Function 4	10,20,30	10/0
f13	Composition Function 5	10,20,30	10/0
f14	Composition Function 6	10,20,30	1/19
f15	Composition Function 7	10,20,30	1/19

We conducted the proposed algorithm in the CEC 2015 competition on single objective multi-niche optimization problems. There are 15 test instances, 8 expanded simple test instances, and 7 composition test instances. Table I lists the characteristics, the dimensions, and the number of the global/local optima of these test instances.

TABLE II
THE MAXIMUM (BEST), MINIMUM (WORST), MEAN AND STANDARD DEVIATION (STD) OF THE NUMBER OF OPTIMA FOUND OBTAINED BY THE PROPOSED ALGORITHM FOR F1-F8 IN THE 50 INDEPENDENT RUNS

No.	Dim.	Best	Worst	Mean	Std
f1	5	16	12	14.45098	0.965686
	10	8	1	4.176471	1.997057
	20	3	0	0.666667	0.840635
f2	2	14	4	8.647059	2.448048
	5	30	24	27.01960	1.392698
	8	57	33	50.21568	4.258233
f3	2	25	23	24.84313	0.418213
	3	57	40	45.64706	3.333608
	4	77	59	68.17647	3.855935
f4	5	8	0	1.823529	1.492727
	10	1	0	0.215686	0.415390
	20	0	0	0	0
f5	2	25	24	24.90196	0.300327
	3	61	40	47.43137	5.142975
	4	59	42	48.07843	3.867005
f6	4	16	15	15.98039	0.140028
	6	53	46	49.33333	1.544884
	8	85	83	84.66666	0.588784
f7	6	8	4	6.333333	1.089342
	10	8	1	3.666667	1.492202
	16	4	0	1.058824	0.946821
f8	2	32	26	29.05882	1.759679
	3	58	42	51.11765	3.320524
	4	60	45	50.72549	4.005389

For all the test instances, the parameter settings of the algorithm were used as follows:

- The crossover and mutation operations with the same control parameters in [15] are used.
- The maximum number of the function evaluation n_eval is set at $\lfloor 2000 * D * \sqrt{q} \rfloor$, where q is the number of the optima, which is given in Table I.
- The population size N is set at $\lfloor 40\sqrt{D * q} \rfloor$ for the test instances.

In order to evaluate the performance of the proposed algorithm, the following two performance metrics were used in this paper. The number of optima found with a given level of accuracy ε is used for the first eight expanded simple test instances, i.e., $f1 - f8$. If the distance from a computed solution to a known global/local optimum is below ε, the peak is considered to be found. Obviously, the greater the value of the number of optima we find is, the better the algorithm performs. Moreover, the mean error values $f(x) - f(x^*)$ of five best optima of each run is calculated for the other seven composition test instances, i.e., $f9 - f15$.

Table II lists the maximum (i.e. best), minimum (i.e. worst), mean and standard deviation (*Std* for short) of the number of optima obtained by the proposed algorithm for the test

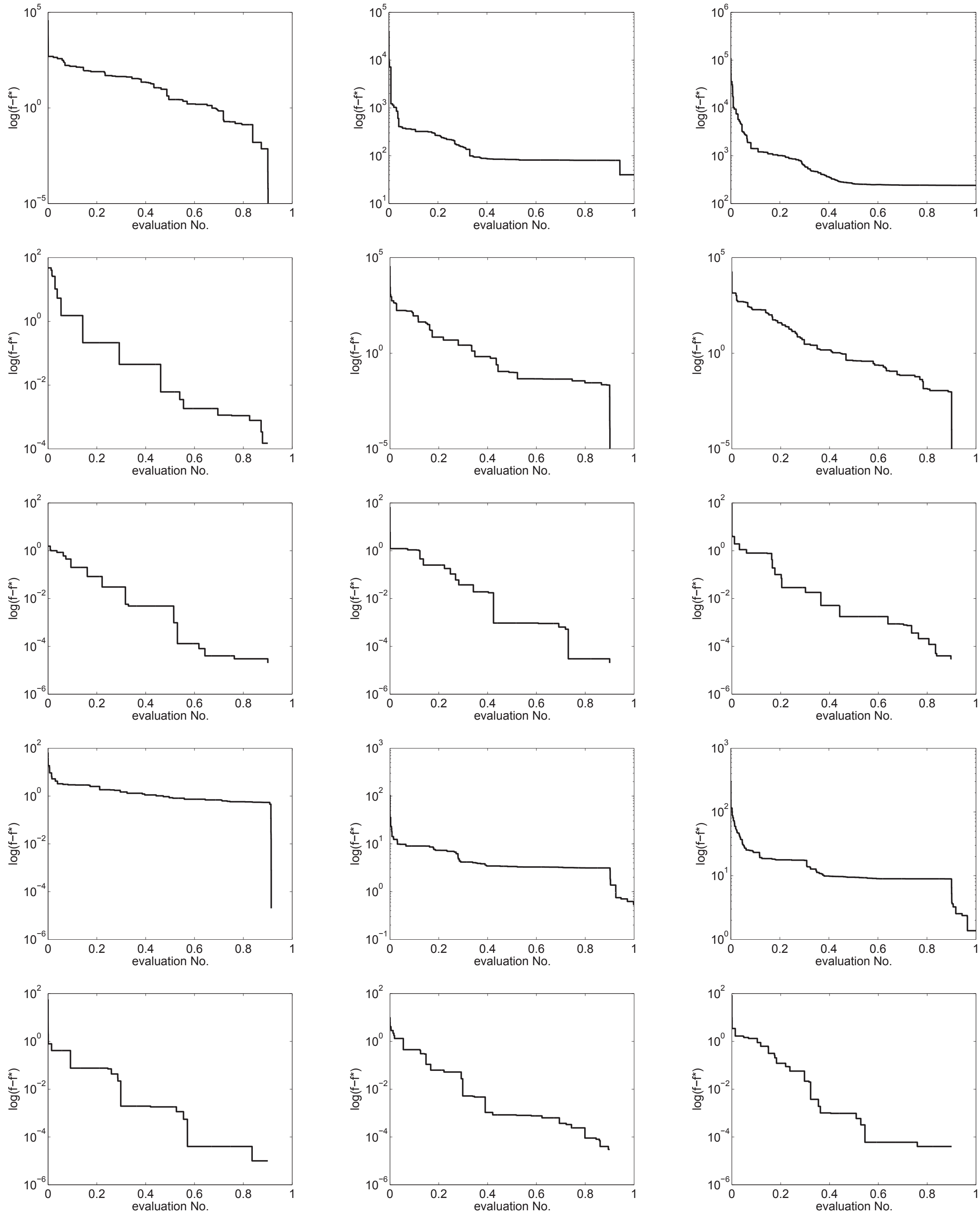


Fig. 2. The results obtained by the proposed algorithm for the expanded simple test instances $f1 - f5$, where the left column is for the test instances with smaller dimension and the middle column is for the test instances with median dimension and the right column is for the higher dimension.

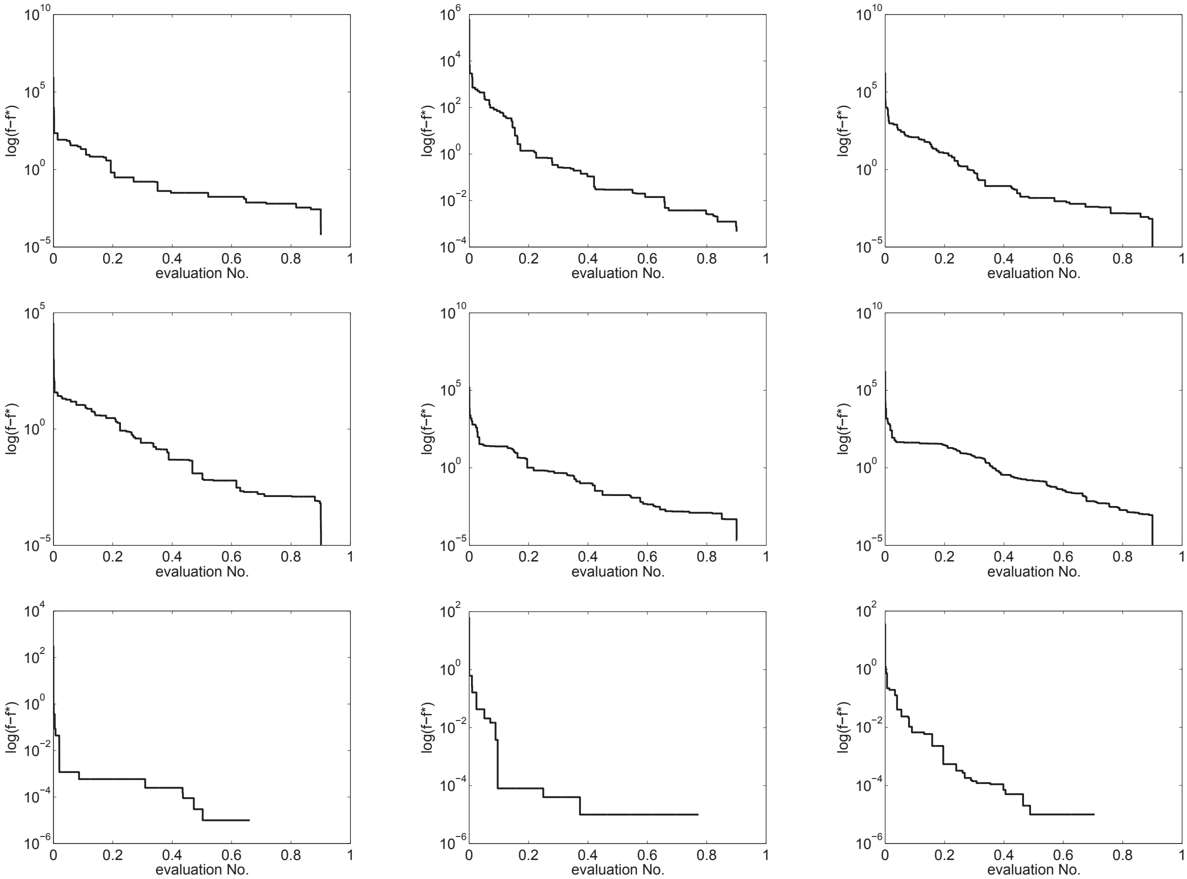


Fig. 3. The results obtained by the proposed algorithm for the expanded simple test instances $f6 - f8$, where the left column is for the test instances with smaller dimension and the middle column is for the test instances with median dimension and the right column is for the higher dimension.

instances $f1 - f8$ in the 50 independent runs. From this table, we can see that the proposed algorithm can obtain a promising results for the test instances $f2, f3, f5, f6$ and $f8$. Table III lists the minimum (i.e. best), maximum (i.e. worst), mean and Std of the mean error values of five best optima obtained by the proposed algorithm for $f9 - f15$ in the 50 independent runs. From this table, we can see that the proposed algorithm can find an ideal result for the test instances $f9$ and $f11$. Moreover, the computation complexity of the algorithm is listed in Table IV, in which $T0$ is the computing time for a given test program in [14], $T1$ is the computing time of 200000 evaluations for function $f8$ with a certain dimension, and $\hat{T}2$ is the mean time of running the proposed algorithm with 200000 evaluations on function $f8$ with the same dimension in 5 runs.

Figures 2 and 3 plot the convergence speed of the proposed algorithm on the expanded simple test instances. The horizontal axis represents the relative number of the function evaluation, i.e. $\frac{i\_eval}{n\_eval}$, while the vertical axis represents $log(f - f^*)$, where f is the objective value of the current best individual in the population in the run with the median number of the optima we have found. The left column is for the test instances with smaller dimension. The middle column is for the test instances with median dimension, and the right column is for the test instances with higher dimension. When the error value $f - f^* < 10^{-8}$, it will be taken as zeros. Therefore, the simulation curves do not approach to 1 in some sub-figures. From these figures, we can find that the proposed algorithm can achieve a very accurate solution except the test instances $f1$ with $D = 10, 20$ and $f4$ with $D = 10, 20$.

IV. Conclusion

We have proposed a non-parametric method to identify the peaks of the multi-modal optimization problems provided that the peaks are characterized by a smaller objective value than their neighbors and by a relatively large distance from points with smaller objective value. Using the identified peaks as the seeds, we have decomposed the population into some subpopulations and dynamically allocated the computational resource to different subpopulations. We have tested the proposed algorithm on the CEC2015 single objective Multi-niche optimization problems. Experiments have shown the promising results.

Acknowledgment

The work described in this paper was supported by the NSFC grant: 61272366, and by Faculty Research Grant of Hong Kong Baptist University with the Project Code: FRG2/14-15/075.

TABLE III
THE MINIMUM (BEST), MAXIMUM (WORST), MEAN AND STANDARD DEVIATION (STD) OF THE MEAN ERROR VALUES OF THE BEST OPTIMA OBTAINED BY THE PROPOSED ALGORITHM FOR F9-F15 IN THE 50 RUNS

No.	Dim.	Best	Worst	Mean	Std
f9	10	0.484430	38.54718	11.58624	9.157981
	20	2.459174	54.76658	35.82022	11.60668
	30	25.56060	79.20924	58.18674	9.187505
f10	10	4081.110	6607.264	5812.020	757.7622
	20	7429.173	17019.32	14096.66	2927.086
	30	6561.314	26154.71	20412.68	4587.097
f11	10	0.533294	4.028767	1.993306	0.756639
	20	2.259424	5.598542	3.876085	0.992146
	30	3.810548	6.345614	4.757173	0.508794
f12	10	137.0642	631.7235	460.5843	101.9687
	20	1245.886	1904.914	1653.617	146.8288
	30	2241.720	3104.029	2728.518	195.0895
f13	10	219.4170	411.1373	295.7372	39.75620
	20	621.7777	909.8633	726.9812	60.07712
	30	1038.448	1387.416	1228.889	82.54259
f14	10	650.0403	1207.844	1001.332	135.5579
	20	1899.484	2718.966	2399.059	211.9665
	30	2947.550	4707.779	4150.835	352.7010
f15	10	140.9665	258.0560	165.7476	33.16442
	20	486.7845	761.0459	673.1962	55.26076
	30	785.6872	1088.471	948.3685	73.36920

TABLE IV
COMPUTATION COMPLEXITY COMPARISON

Dim.	T0	T1	$\hat{T}2$	$(\hat{T}2 - T1)/T0$
2	3.385221	9.438060	39.379030	8.844613
3	3.385221	9.458683	40.717989	9.234051
4	3.385221	9.609661	41.127835	9.310521

REFERENCES

[1] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proceedings of the Second International Conference on Genetic Algorithms Genetic Algorithms and their Applications*, 1987, pp. 41–49.

[2] B.-Y. Qu, P. N. Suganthan, and J.-J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE transactions on evolutionary computation*, vol. 16, no. 5, pp. 601–614, 2012.

[3] X. Li, "A multimodal particle swarm optimizer based on fitness euclidean-distance ratio," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, 2007, pp. 78–85.

[4] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Multimodal optimization by means of a topological species conservation algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 6, pp. 842–864, 2010.

[5] C. L. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Disburdening the species conservation evolutionary algorithm of arguing with radii," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, 2007, pp. 1420–1427.

[6] S. W. Mahfoud, "Population size and genetic drift in fitness sharing," in *Proceedings of Foundations Genetic Algorithms*, 1994, pp. 185–223.

[7] J. Yao, N. Kharma, and P. Grogono, "Bi-objective multipopulation genetic algorithm for multimodal function optimization," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 80–102, 2010.

[8] S. Bandaru and K. Deb, "A parameterless-niching-assisted bi-objective approach to multimodal optimization," in *Proceedings of 2013 IEEE Congress on Evolutionary Computation (CEC2013)*, 2013, pp. 95–102.

[9] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 666–685, 2013.

[10] A. Della Cioppa, C. De Stefano, and A. Marcelli, "Where are the niches? dynamic fitness sharing," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 4, pp. 453–465, 2007.

[11] W. Dong and M. Zhou, "Gaussian classifier-based evolutionary strategy for multimodal optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1200–1216, 2014.

[12] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for cec2013 special session and competition on niching methods for multimodal function optimization," Tech. Rep., 11 2013. [Online]. Available: http://goanna.cs.rmit.edu.au/xiaodong/cec13-niching/top

[13] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[14] B. Y. Qu, J. J. Liang, P. N. Suganthan, and Q. Chen, "Problem definitions and evaluation criteria for the cec 2015 competition on single objective multi-niche optimization," Zhengzhou University, Tech. Rep., 11 2014. [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2015

[15] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 450–455, 2014.