An improved Differential Evolution for Multi-Objective Optimization

Ke Li, Jinhua Zheng, Cong Zhou, Hui Lv
Institute of Information Engineering, Xiangtan University, Hunan, China 411105

Abstract

Evolutionary algorithms (EAs) are well-known optimization approaches to deal with nonlinear and complex problems. However, these population-based algorithms are computationally expensive due to the slow nature of the evolutionary process. This paper proposes an improved differential evolution algorithm (CDE). On the one hand CDE combines the advantages of DE with the mechanisms of Pareto based ranking and crowding distance sorting which are similar to the NSGA-II, on the other hand different from the previous DE, CDE compares the trial vector to its nearest neighbor to decide whether to preserve it. Experimental results confirm that CDE outperforms the other two classical multi-objective evolutionary algorithms (MOEAs) NSGA-II and SPEA2 in terms of diversity and convergence.

1. Introduction

Most real-world problems involve the simultaneous optimization of two or more (often conflicting) objectives. The solutions of such problems are different from that of a single-objective optimization problems normally have not one but a set of solutions which are all equally good.

In recent years, many algorithms for multi-objective optimization have been introduced. Among these, the NSGA-II by Deb et al. [1] and SPEA2 by Zitzler et al. [2] are most popular. Differential evolution (DE) is a simple yet powerful evolutionary algorithm (EA) for global optimization introduced by Price and Storn [5][6]. The DE algorithm has gradually become more popular and has been used in many practical cases, mainly because it has demonstrated good convergence properties and is principally easy to understand.

In this paper, we propose an improved differential evolution algorithm. Different from the previous DE，in our algorithm the way we decide whether to keep the trial vector lies in comparing to its nearest neighbor. We combine the pruning mechanism used in the NSGA-II with the advantages of DE to keep the diversity of obtained solutions. Mutation operation is an omitted evolutionary step in the previous DE. We are trying to integrate mutation operation into our algorithm, for the purpose of speeding up the convergence.

The remainder of this paper is organized as follows: Section 2 provides some basic definitions of multi-objective optimization. Afterwards, a brief introduction of DE is described in Section 3. Section 4 presents our proposed CDE. Section 5 provides the test problems used in our experiment. Our comparison of results and some discussions are provided in Section 6. Finally, Section 7 contains the conclusions and some possible extension for our future research.

2. Definitions

Different from the single objective optimization problems, MOEA goals to converge to the Pareto front fast and spread uniformly. Some related basic definitions are introduced next.

Definition 1. Multi-objective optimization problem (MOP) The multi-objective optimization can be

978-0-7695-3507-4/08 $25.00 © 2008 IEEE
DOI 10.1109/CSIE.2009.181

formally defined as the problem to find a vector $\vec{x}^* = [x_1^*, x_2^*, \cdots, x_n^*]^T$ which satisfies m inequality constraints:

$$g_i(\vec{x}) \le 0; i = 1, \cdots, m \qquad (1)$$

p equality constraints:

$$h_i(\vec{x}) \le 0; i = 1, \cdots, p \qquad (2)$$

and optimizes the vector function:

$$f_1(\vec{x}), f_2(\vec{x}), \cdots, f_k(\vec{x}) \qquad (3)$$

Definition 2. **(Pareto dominance)** Pareto dominance is formally defined as follows: A vector $\vec{u} = (u_1, \cdots, u_k)$ is said to dominate $\vec{v} = (v_1, \cdots, v_k)$ if and only if $\vec{u}$ is partially less than $\vec{v}$, i.e. $\forall i \in (1, \cdots, k), u_i \le v_i \wedge \exists i \in (1, \cdots, k) : u_i < v_i$

Definition 3. **(Pareto optimality)** The definition of Pareto optimality is provided next: A solution $\overline{x_u} \in \mathbb{F}$ (where $\mathbb{F}$ is the feasible region) is said to be Pareto optimal if and only if there is no $\overline{x_v} \in \mathbb{F}$ which makes $v = f(x_v) = (v_1, \cdots, v_k)$ dominate $u = f(x_u) = (v_1, \cdots, v_k)$, where k is the number of objectives.

Definition 4. (Pareto front) When all nondominated solutions are plotted in the objective space, the non-dominated vectors are collectively known as the Pareto front. For a given MOP $\overline{f}(x)$ and Pareto optimal set P*, the Pareto front (PF*) is defined as:

$$PF^* := \{\overline{f} = [f_1(x), \cdots, f_k] \mid x \in P^*\} \qquad (4)$$

3. Differential Evolution

Differential Evolution is a branch of evolutionary algorithms (EAs) which was designed by Price and Storn[5][6] to optimize problems over continuous domains. In DE, each decision variable is represented in the chromosome by a real number. Like other evolutionary algorithms (EAs), it starts with an initial population vector, which is randomly generated when no preliminary knowledge about the solution space is available. Let us assume that $X_{ri,G}(i = 1, 2, \cdots, N_p)$ are solution vectors in generation G (N_p is the population size). The offspring are generated by adding the weighted difference vector between two parents to a third parent. Formally, the process is described as follows. For each vector $X_{ri,G}(i = 1, 2, \cdots, N_p)$, a trial vector $U_{ri,G}$ generated according to:

$$U_{ri,G} = X_{r1,G} + F \cdot (X_{r2,G} - X_{r3,G}) \qquad (5)$$

with $r1, r2, r3 \in [0,\ N_p - 1]$, $F > 0$.

The integers r1, r2 and r3 are chosen randomly from the interval $[0,\ N_p - 1]$ which are different from ri. F is a constant factor that controls the amplification of the differential variation $(X_{r2,G} - X_{r3,G})$. Figure 1 shows a two dimensional example that illustrates the different vectors which play a role in DE.

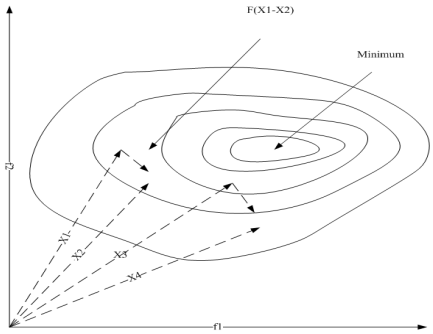

Figure 1. Two dimensional example of the process for generating offspring

4. Our proposed algorithm

Our corresponding pseudo-code of the proposed approach (CDE) is shown in the appendix.

In our algorithm, we use the same user defined parameters *CR* and *F* which are commonly used in previous researches. *CR* controls the rate of crossover and *F* is a constant factor which control the amplification of the differential variation $(X_{r2,G} - X_{r3,G})$.

During the process of the algorithm *CR* controls total rate of evolutionary operation, and *F* controls the convergence speed and robustness of DE. Experiment confirms that a small value of *F* could speed up the convergence, but it also easily lead the algorithm trapped in a local optimal [3][4]. In addition, a new parameter *MUT* is added to our CDE, it controls the rate of mutation operation, we set the *MUT* to the countdown of population size.

Different from the common DE, we compare the dominance relation between the trial vector and the nearest vector to it instead of the sign vector. That is to say, if the trial vector dominates the vector which is nearest to it, the trial vector will replace it. On the contrary we will drop the trial vector and reserve the sign vector into the next generation. But if the trial vector is non-dominated to the sign vector, we add the trial vector into a temporary population. After a generation, the temporary population and the new produced population are merged into a new hybrid population. Clearly, the size of the hybrid population may increase, so we have to remove the unexpected solutions from the hybrid population until the size of the hybrid population get the original size we have set before. The removing operation is based on the dominance level and the crowding distance which is similar to the pruning mechanism used in NSGA-II. Besides, there is no mutation operation in the traditional DE, but as we know that mutation is an indispensable step in evolutionary algorithms (EAs). So we attempt to add a mutation operator in our approach which is implemented as expression (6):

$$lower(x_i) + rand[0,1) \times (upper(x_i) - lower(x_i)) \qquad (6)$$

During the process of our algorithm, we find that the trial vector may exceed its boundary, and then such trial vector turns invalid. So we use a repair rule which makes the trial vector return to its boundary. The repair rule is shown in expression (7).

$$x^*_{i,G=g} = \begin{cases} upper(x_i) + (x^j_{i,G} - lower(x_i))/2 & if\ x^j_{i,G+1} < lower(x_i) \\ lower(x_i) + (x^j_{i,G} - upper(x_i))/2 & if\ x^j_{i,G+1} > upper(x_i) \\ x^j_{i,G+1} & otherwise \end{cases} \qquad (7)$$

5. Test problems and performance indices

We test CDE on two dimensions and three dimensions cases separately, and we compare the experimental results with two classical algorithms NSGA-II and SPEA2.

5.1. Parameter Setting

All MOEAs are given real-valued decision variables. A crossover probability of P_c=0.8 and a mutation probability P_m=*1/n* (where *n* is the number of decision variables) are used. The operators for crossover and mutation are simulated binary crossover (SBX) and polynomial mutation, with distribution indexes of η_c = 15，η_m = 20, respectively. In CDE, we set *CR* = 0.95, *F* = 0.4, $MUT = 1/N_p$.

5.2. Function used

The test bed is formed by a total of eight functions. They are ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, DTLZ-2, SCH, FON [8][9][10].

5.3. Performance metrics

All MOEAs in this paper are measured using spacing index (SP) [8]. The spacing measures the standard deviation of distances from each vector to the nearest vector in the non-dominated set as expression (8):

$$SP = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\bar{d} - d_i)^2} \qquad (8)$$

$$d_i = \min_j \left(\left| f_1^i(x) - f_1^i(x) \right| + \left| f_2^i(x) - f_2^i(x) \right| \right) \qquad i, j = 1, \cdots, n$$

$\bar{d}$ is the mean of all d_i, and *n* is the known size of the Pareto frontier.

The other metric is the Generation Distance (GD) [7] which measures the distance between the obtained non-dominated front Q and the set P^* of Pareto optimal solutions as expression (9):

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \qquad (9)$$

n is the number of individual in the obtained non-dominated front Q, where d_i is again the Euclidean distance (in the objective space) between the solution $i \in Q$ and the nearest member of P^*.

6. Simulation results and discussions

To compare these methods, we have carried out 20 independent runs, and the Table 1 which is given in the Appendix includes the average and standard deviation. The best metric values are shown bold.

The PF_{ture} of test problem SCH is an arc in [0, 4]. Table 1 shows that CDE owns the best GD and SP values. Test problem FON has a concave Pareto-optimal front. Table 1 shows that CDE owns the best SP value and NSGA-II owns the best GD value. Test problem ZDT1, ZDT2, ZDT3 all have 30 decision variables, and ZDT3 is a non-continuous problem. From Table 1 we can see that all the SP values of CDE are the best. Test problem ZDT4 is a difficult test problem as it has 21^9 local optimal, so it easily traps the MOEAs in local optimal. Table 1 also shows that CDE own the best GD. Next, we choose the test problem ZDT6, which has a concave Pareto-optimal front. In Table 1 we find that the SP value for CDE is the best. At last, we choose a three dimensional test problem DTLZ-2, which has a spherical Pareto-optimal front. Table 1 also shows that CDE don't have advantages neither in GD nor SP.

In our algorithm, there is a trial operation that we add a specific mutation operator to DE for the purpose of accelerating the speed of the convergence. However, the experiments show that such operation don't make sense in effect.

7. Conclusion and future work

In this paper, an improved differential evolution algorithm was proposed. We use a new archive mechanism that we compare the trial vector with its nearest vector, instead of the sign vector in previous DE. The diversity of obtained solutions is kept by the combination of the advantages of DE and the pruning mechanism used in the NSGA-II. The two operations above have indeed improved the effect of DE. And a mutation operator was integrated into the differential evolution for the expectation of speeding up the convergence, but this added operator hasn't reached the ideal purpose. We test our approach on eight test problems, where it outperforms a number of the state-of-art approaches in the literature. But as we can see that CDE performs not well enough in three dimensional test problem.

For future work, we intend to improve the performance of our approach in three dimensional test cases. Also, the parameters chosen in this paper were generated experimentally. It would be interesting to see the effect of these parameters on this problem.

Acknowledgment

Authors acknowledge the supports from the National Natural Science Foundation of China (60773047), the Natural Science Foundation of Hunan Province (05JJ30125) and the Keystone Science Research Project of the Education Office of Hunan Province (06A074).

8. References

[1] Deb K, Pratap A, Agarwal S, Meyarivan T. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II." *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002

[2] Zitzler E, Laumanns M, Thiele L. "SPEA2: Improving the Strength Pareto Evolutionary Algorithm." Technical Report 103, Computer Engineering and Networks

Laboratory (TIK), ETH Zurich, Zurich, Switzerland, 2001

[3] S. Kukkonen and J. Lampinen. "Comparison of Generalized Differential Evolution algorithm to other multiobjective evolutionary algorithms." In Proceedings of the 4th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS2004), page 445, Jyv¨askyl¨a, Finland, July 2004.

[4] S. Kukkonen and Jouni Lampinen. "GDE3: The third evolution step of Generalized Differential Evolution." In Proceedings of the 2005 Congress on Evolutionary Computation (CEC2005), pages 443{450, Edinburgh, Scotland, Sept 2005.

[5] K. V. Price. *New Ideas in Optimization*: An Introduction to Differential Evolution, pages 79–108. McGraw-Hill, London, 1999.

[6] K. V. Price, R. Storn, and J. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization.* Springer-Verlag, Berlin, 2005.

[7] Van Veldhuizen D A, Lamont G B. "Multiobjective evolutionary algorithm research: a history and analysis." TR–98–03[R]. Dept Elec Comput Eng, Graduate School of Eng, Air Force Inst Technol, Wright–Patterson AFB, OH,1998.

[8] J R Schott. "Fault tolerant design using single and multi-criteria genetic algorithm optimization." Master's thesis of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, 1995-05.

[9] Deb Kalyanmoy. "Multi-Objective Genetic Algorithms: Problem difficulties and Construction of Test Problems." *Evolutionary Computation*, 7(3): 205-230, 1999.

[10] Deb Kalyanmoy, Thiele L, Laumanns M et a1. "Scalable Multiobjective Optimization Test Problem." In:Congress on Evolutionary Computation (CEC'2002); 2002; 1:825~830

Appendix I

The pseudo-code of CDE is shown as follows:

Input: $D, G_{\max}, N_p \geq 4, CR \in [0,1], F \in (0,1+], q = 0, MUT = 1/N_p$

initialize the boundary of the variables: $lower(x_i), upper(x_i)$

For each individual $j \in P_{G=0}$

$x^j_{i,G=0} = lower(x_i) + rand_i[0,1] \times (upper(x_i) - lower(x_i))\ \ (i = 1, \cdots, D)$

End for each

evaluate the population in generation 0, set g to 1

While $g < G_{\max}$

Forall $j \leq N_p$

randomly choose three integers $r1, r2, r3 \in (1, \cdots, N_p)$ and $j \neq r1 \neq r2 \neq r3$

create a random integer $i_{rand} \in (1, \cdots, D)$

Forall $i \leq D$

$$u^j_{i,G=g} = \begin{cases} x^{r3}_{i,G=g-1} + F \times (x^{r1}_{i,G=g-1} - x^{r2}_{i,G=g-1}) \\ \quad if\ (rand[0,1) < CR \wedge i = i_{rand}) \\ x^j_{i,G=g-1} \quad otherwise \end{cases}$$

$$u_{i,G=g}^{j} = \begin{cases} lower(x_i) + rand[0,1) \times (upper(x_i) - lower(x_i)) \\ \qquad if\ (rand[0,1) < MUT) \\ u_{i,G=g}^{j} \quad \text{otherwise} \end{cases}$$

if the trial vector $u_{i,G=g}^{j}$ go beyond the boundary of the variable, we use the repair rule to repair this trial vector

End forall

choose the individual $x_{G=g}^{*}$ which is nearest to the trial vector $u_{i,G=g}^{j}$

$$\vec{x}_{G=g+1}^{\ j} = \begin{cases} \vec{u}_{G=g}^{\ j} \quad if\ \vec{u}_{G=g}^{\ j} \prec \vec{x}_{G=g}^{*} \\ \vec{x}_{G=g}^{\ j} \quad otherwise \end{cases}$$

$$if(\bar{x}_{G=g+1}^{j} = \bar{x}_{G=g}^{j} \wedge \bar{x}_{G=g}^{j} \prec \bar{u}_{G=g}^{j})$$

$$q = q + 1 \wedge \vec{x}_{p,G+1}' = \vec{u}_{j,G}$$

End forall

Merge these two populations into a new hybrid population. Then we truncate this over-bounded population.

$g = g + 1$

End while

Appendix II

The value of performance metrics are shown in Table 1 as follows:

Table 1. Performance comparisons of the three MOEAs for two objectives and three objectives tests.

Problem	Method	GD		SP	
		Average	Std.Dev	Average	Std.Dev
SCH	CDE	**0.00027133**	1.382E-005	**0.0128527**	0.0006038
	NSGA-II	0.00042570	1.654E-005	0.0331324	0.0068416
	SPEA2	0.00039107	1.934E-005	0.0139704	0.0010885
FON	CDE	0.00028117	1.653E-005	**0.0031244**	0.0002149
	NSGA-II	**0.00016112**	1.109E-005	0.0060988	0.0002661
	SPEA2	0.00022054	6.731E-006	0.0032458	0.0002112
ZDT1	CDE	0.00027125	1.587E-005	**0.0025789**	0.0001386
	NSGA-II	**0.00018856**	5.222E-005	0.0078044	0.0007836
	SPEA2	0.00026398	2.597W-005	0.0030282	0.0001807
ZDT2	CDE	0.00011040	3.856E-006	**0.0029994**	0.0001594
	NSGA-II	**0.00010714**	3.891E-005	0.0077056	0.0015039
	SPEA2	0.00010092	3.715E-006	0.0031558	0.0002961
ZDT3	CDE	**0.00045941**	2.612E-005	**0.0031391**	0.0017738
	NSGA-II	0.00058064	2.659E-005	0.0083976	0.0004040
	SPEA2	0.00081688	0.00026571	0.0051128	0.0013548
ZDT4	CDE	**0.00027354**	0.02176654	0.0063007	0.0002647
	NSGA-II	0.04408061	0.02876059	0.0113370	0.0003647
	SPEA2	0.07577120	0.05957841	**0.0037116**	0.0001929
ZDT6	CDE	**0.00042144**	1.687E-005	0.0057462	0.0021386
	NSGA-II	0.05306881	0.03495159	0.0063434	0.0021021
	SPEA2	0.00056205	7.473E-006	**0.0018274**	0.0001183
DTLZ-2	CDE	0.00027793	2.138E-005	0.0285187	0.0025607
	NSGA-II	0.00010799	3.456E-005	0.0573802	0.0046389
	SPEA2	**5.064E-005**	4.774E-005	**0.0232716**	0.0015607