http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (3): 444-448, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.444.448

An Efficient Differential Evolution Algorithm For Function Optimization

Chao-Xue Wang, Chang-Hua Li, Hui Dong and Fan Zhang
School of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China

Abstract: An efficient differential evolution algorithm (AEDE) for function optimization is proposed. First, with population evolution, AEDE divides population into three groups by the fitness's normal distribution and the three groups adopt different mutation operators. Second, the selection of the individuals involved in mutation operation uses alternatively a random method and a roulette wheel method based on affinity matrix. To validate the superiority of AEDE, AEDE and some state-of-the-art DE variants proposed in pertinent literatures are compared as regards nine benchmark functions. The simulation results show that ANDE promises competitive performance not only in the convergence speed but also in the quality of solution.

Key words: Differential evolution algorithm, function optimization, normal distribution, affinity matrix, roulette wheel selection

INTRODUCTION

Differential Evolution (DE), proposed by Storn and Price (Storn and Price, 1995), is a relatively new optimization technique. It is very easy to implement, requires little or no parameter tuning and has been successfully applied in many problems domain. It has been investigated that DE is faster and more robust in a plethora of problems than many other evolutionary algorithms (Storn and Price, 1995; Price *et al*., 2005; Vesterstrom and Thomsen, 2004).

However, it has been observed that the convergence rate of DE do not meet the expectations in cases of highly multimodal problems. Several DE variants have been proposed to improve its performance (Qin and Suganthan, 2005; Wang *et al*., 2007; Noman and Iba, 2008; Epitropakis *et al*., 2008; Zhang *et al*., 2010; Montgomery and Chen, 2010; Wang *et al*., 2011; Epitropakis *et al*., 2011; Mininno *et al*., 2011). The exploration and exploitation are two kind of different search mechanism in DE like other intelligent optimization algorithm and the balance between them is the key factor to the performance of algorithm (Epitropakis *et al*., 2008; Macready *et al*., 1998). In this paper, a efficient DE algorithm (AEDE) is proposed which is expected to achieve better balance of exploration and exploitation by multi-strategy differential mutation mechanism. Extensive experiments have been conducted to compare AEDE with the classic DE (Storn and Price, 1995), SaDE (Qin and Suganthan, 2005), CDE (Wang *et al*., 2007), DeahcSPX (Noman and Iba, 2008), DMSDELS (Zhang *et al*., 2010), ProDE (Epitropakis *et al*., 2011) on nine well-known benchmark functions. The test results show that AEDE is very competitive.

THE CLASSIC DE AND MULTI-STRATEGY DIFFERENTIAL OPERATOR

The classical DE: As a stochastic method, DE algorithm borrows the idea from Nelder-Mead's method and uses the randomly generated initial population, differential mutation, probability crossover and greedy criterion based selection to find the global optimum of objective function. The pseudo-code of classic DE algorithm (Storn and Price, 1995) is given as following:

```
/* Initialize parameters */
Set mutation scale factor F∈[0, 2],
crossover parameter CR∈(0, 1),
population size NP, counter g = 0
/* Initialize a population */

while budget condition do
g = g+1
 for i = 1 to NP do
 /* Differential Mutation */
 select randomly 3 individuals

   x^g_p1, x^g_p2, x^g_p3 ∈ p^g

   v_i = x^g_p1 + F×(x^g_p2 + x^g_p3)
   p1,p2,p3∈{1,2,...,NP}/{i}
 /* Probability Binomial Crossover */
 generate randomly j_rand∈{1,2,...,D}
 for j = 1 to D do
 generate randomly rand(0,1)
```

Corresponding Author: Chao-Xue Wang, School of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China

```
    if rand(0,1)≤CR or j = j_rand
      u_i^{j,g} = v_i^j

    else

      u_i^{j,g} = x_i^j

  end
  end for
  /* Greedy Selection */
  if f(u_i^g)<f(x_i^g)

  else

  x_i^{j,g} = x^g_i

  end if
  end for
end while
```

Multi-strategy differential operator: Many variants of the classic DE have been proposed which use different mutation strategies and/or recombination in the reproduction stage (Price *et al*., 2005; Wang *et al*., 2011; Epitropakis *et al*., 2011; Das and Suganthan, 2011). The originally proposed and most frequently used differential mutation operators in the literature are as follows (Epitropakis *et al*., 2011; Mininno *et al*., 2011; Das and Suganthan, 2011). In order to distinguish them, the notation DE/a/b is used, where "a" specifies the base vector to be mutated (which can be random, the best or the current vector); "b" is the number of difference vector used.

- **DE/rand/1:**

$$V_i = X_{p1}+F\times(X_{p2}-X_{p3}) \quad (1)$$

- **DE/rand/2:**

$$V_i = X_{p1}+F\times(X_{p2}-X_{p3})+F\times(X_{p4}-X_{p5}) \quad (2)$$

- **DE/best/1:**

$$V_i = X_{best}+F\times(X_{p1}-X_{p2}) \quad (3)$$

- **DE/best/2:**

$$V_i = X_{best}+F\times(X_{p1}-X_{p2})+F\times(X_{p3}-X_{p4}) \quad (4)$$

- **DE/current-to-best/1:**

$$V_i = X_i+F\times(X_{best}-X_i)+F\times(X_{p1}-X_{p2}) \quad (5)$$

- **DE/current-to-best/2:**

$$V_i = X_i+F\times(X_{best}-X_i)+F\times(X_{p1}-X_{p2})+F\times(X_{p3}-X_{p4}) \quad (6)$$

In an attempt to rationalize the mutation strategies, Eq. 1-6, we observe that Eq. 3 is similar to the crossover operator employed by some genetic algorithms. Eq. 1 is derived from Eq. 3, by substituting the best member of the previous generation, X_{best}, with a random individual X_{p1}. Eq. 2 and Eq. 4-6 are modifications obtained by the combination of Eq. 1 and Eq. 3.

These mutation operators can be divided into three categories. The first is the exploratory mutation operator which includes DE/rand/1 and DE/rand/2 and shows a strong exploring performance but a low convergence speed. The second is the exploitative mutation operator which includes DE/best/1 and DE/best/2 and shows a high convergence speed but a poor performance on exploration. The third is the balanced mutation operator which includes DE/current-to-best/1 and DE/current-to-best/2 and shows harmonious on exploration and exploitation (Wang *et al*., 2011; Epitropakis *et al*., 2011).

A EFFICIENT DIFFERENTIAL EVOLUTION ALGORITHM

Here, we describe the proposed algorithm. Fig. 1 is the flowchart of AEDE and its crossover operation and selection operation is the same as the classic DE (Storn and Price, 1995).

Adaptive mutation mechanism: By extensive experiments, we have found that the fitness of population presents a similar normal distribution in DE algorithm. An adaptive mutation mechanism is adopted in AEDE. With population evolution, all the individuals of population are divided into three groups by the normal distribution of

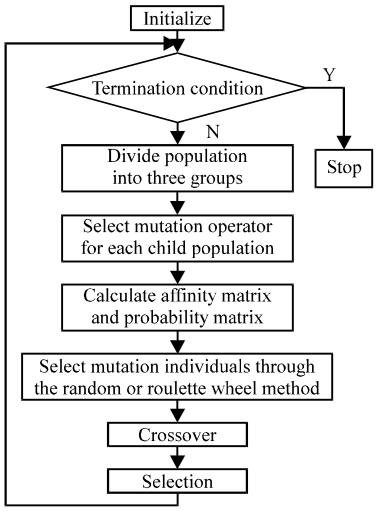

Fig. 1: The flowchart of AEDE

population fitness and the three groups adopt different mutation operators.

The probability $P(-\sigma<x-\mu<\sigma)$ is approximating 0.683 in Gaussian distribution $N(\mu,\sigma^2)$. Let, μ denote the mean of population fitness, σ denote the mean square err (MSE) of population fitness, F indicate the fitness of the ith individual. AEDE adjusts the mutation strategies as follows:

- if $F_i-\mu<-\sigma$ which shows this individual is a better one, it would be assigned to the excellent group and the exploratory mutation operators are appropriate for it (e.g., DE/rand/1 and DE/rand/2)
- if $F_i-\mu>\sigma$ which shows this individual's performance is poor, it would be assigned to the poor group and the exploitative mutation operators are appropriate for it (e.g., DE/best/1 and DE/best/2)
- if $-\sigma\leq F_i-\mu<-\sigma$ which shows this individual has a mediocre performance, it would be assigned to the middle group and the balanced mutation operators are suitable for it (e.g., DE/current-to-best/1 and DE/current-to-best/2)

The selection of individuals involved in mutation operation: A novel framework based on the proximity characteristics (Epitropakis *et al*., 2011) was proposed, where a roulette wheel method based on affinity matrix was used to select individuals involved mutation operator. This selection method is briefly introduced as follows. Firstly, the affinity matrix AM_x based on real distances between individuals is calculated by Eq. 7, whose element $AM_x(i,j)$ corresponds to the distance between the ith and jth individuals:

$$AM_x=\begin{bmatrix} 0 & \|\mathbf{x}_1^1,\mathbf{x}_2^2\| & \cdots & \|\mathbf{x}_1^g,\mathbf{x}_{NP}^g\| \\ \|\mathbf{x}_2^g,\mathbf{x}_1^g\| & 0 & \cdots & \|\mathbf{x}_2^g,\mathbf{x}_{NP}^g\| \\ \|\mathbf{x}_3^g,\mathbf{x}_1^g\| & \|\mathbf{x}_3^g,\mathbf{x}_2^g\| & 0 & \|\mathbf{x}_3^g,\mathbf{x}_{NP}^g\| \\ \vdots & \vdots & \ddots & \vdots \\ \|\mathbf{x}_{NP}^g,\mathbf{x}_1^g\| & \|\mathbf{x}_{NP}^g,\mathbf{x}_2^g\| & \cdots & 0 \end{bmatrix} \quad (7)$$

where, $\|x,y\|$ is a distance measure between the x and y individuals. In the case of decision variables with different search ranges, a scale-invariant distance measure e.g., the Mahalanobis distance (Theodoridis and Koutroumbas, 2008) needs to be used to avoid any dependence on the scale of the variables. Here Euclidean distance is used since all the variables have equal ranges in all the considered problems. Secondly, based on the AM_x matrix, a probability matrix P_x is calculated, where each element $P_x(i,j)$ Eq. 8 represents a probability between the ith and jth individual with respect to the ith row. The probability of the ith individual is inversely proportional to the distance of the jth individual, i.e., the individual of the row with the minimum distance has the maximum probability:

$$P_x(i,j)=1-\frac{AM_x(i,j)}{\sum_j^{NP}AM_x(i,j)} \quad (8)$$

where, i,j = 1,2,...,NP. Lastly, the simple roulette wheel selection according to the probability matrix P_x is used to obtain three individuals involved in mutation operation.

By theoretical analysis and experiments, we find that this roulette wheel method makes the current individual' nearest individual have the highest selection probability and the corresponding mutation operation could be viewed as a minor fluctuation to the current individual. Although this method can significantly enhance DE's ability of exploitation, there exists a risk of decreasing DE's ability of exploration. In order to avoid excessive exploitation and balance the exploration and exploitation capabilities of algorithm, we make some improvements to it. When the strategy of DE/best/1 or DE/best/2 is adopted in mutation operation, the selection of difference vector individual uses the random method. Meanwhile, when other mutation operators are adopted, the random method and the simple roulette wheel method are used alternately.

SIMULATION

The performance of AEDE is tested on 9 well-known benchmark problems f_1-f_9 from literature (Wang *et al*., 2007; Zhang *et al*., 2010) and is compared with several DE variants proposed in pertinent literatures. The parameters setting of these algorithms are consistent with their original literatures. In AEDE, the parameters NP, CR and F are fixed to 100, 0.9 and 0.5 and the affinity matrix are calculated in every 50 generations.

The comparison of evolution curve between AEDE and the classic DE (Storn and Price, 1995) for f_8 is shown in Fig. 2. It can be seen that AEDE performs a better convergence to more accurate results and a faster convergence speed.

Table 1 reports the results that AEDE is compared with the classic DE (Storn and Price, 1995), SaDE (Qin and Suganthan, 2005), CDE (Wang *et al*., 2007), DeahcSPX (Noman and Iba, 2008), DMSDELS (Zhang *et al*., 2010), ProDE (Epitropakis *et al*., 2011) in terms of the best solution, mean solution and the Std.Dev. of population on f_1~f_9.

It can be observed from Table 1 that at the same preset maximum number of iterations, AEDE outperforms classic DE, SaDE, CDE, DEahcSPX, DMSDLS and ProDE in most cases. In more detail, AEDE obtains better Best, Mean and Std.Dev. than DMSDLS on all the nine functions and meanwhile it is better than classic DE, CDE, SaDE, DeahcSPX and ProDE on most test functions,

Table 1: Comparisons between the classic DE, SaDE, CDE, DEahcSPX, DMSDLS, Pro DE and AEDE on functions f_1-f_9 with dimension 30

Function (generations)	Algorithm	Best	Mean	SD
f_1(1500)	DE	5.22e-14	3.70e-13	3.90e-13
	SaDE	2.70e-37	1.90e-35	2.30e-35
	CDE	1.00e-06	1.00e-06	2.00e-06
	DEahcSPX	3.24e-32	1.75e-31	4.99e-31
	DMSDLS	1.40e-45	8.40e-45	7.70e-45
	Pro DE	1.62e-31	2.54e-30	2.88e-30
	AEDE	1.28e-70	2.09e-68	2.37e-68
f_2(2000)	DE	6.20e-10	3.70e-09	2.20e-09
	SaDE	2.00e-14	5.80e-14	3.20e-14
	CDE	0.00	0.00	0.00
	DEahcSPX	-	-	-
	DMSDLS	1.60e-23	6.10e-23	2.50e-23
	Pro DE	8.84e-21	3.27e-20	4.09e-20
	AEDE	1.11e-35	1.52e-34	1.63e-34
f_3(5000)	DE	1.10e-11	1.80e-10	1.50e-10
	SaDE	3.70e-40	3.60e-37	1.10e-36
	CDE	8.63e-06	1.10e-05	1.40e-05
	DEahcSPX	1.68e-05	6.52e-05	4.84e-05
	DMSDLS	2.50e-02	7.40e-01	1.30
	Pro DE	2.50e-19	8.15e-18	9.17e-18
	AEDE	2.58e-46	2.72e-44	1.21e-44
f_4(5000)	DE	6.80e-13	3.10e-02	8.70e-02
	SaDE	6.20e-11	2.60e-10	1.80e-10
	CDE	7.50e-05	7.50e-05	7.50e-05
	DEahcSPX	-	-	-
	DMSDLS	3.50e-24	1.70e-22	2.80e-22
	Pro DE	4.32e-16	6.24e-14	7.35e-15
	AEDE	1.47e-25	9.59e-24	3.42e-24
f_5(20000)	DE	0.00	3.50e-31	2.50e-30
	SaDE	0.00	4.50e-30	7.50e-30
	CDE	1.12	1.83	6.33
	DEahcSPX	1.40e-01	4.52	1.55e+01
	DMSDLS	4.40e-11	1.30e-03	7.30e-01
	Pro DE	1.57e-28	5.73e-27	7.81e-27
	AEDE	8.60e-27	2.19e-10	3.14e-09
f_6(3000)	DE	2.00e-03	4.70e-03	1.30e-03
	SaDE	1.50e-03	3.20e-03	8.40e-04
	CDE	3.50e-04	4.60e-04	3.60e-04
	DEahcSPX	-	-	-
	DMSDLS	8.10e-04	2.30e-03	5.10e-04
	Pro DE	1.28e-02	2.06e-02	2.45e-02
	AEDE	7.80e-04	2.10e-03	1.21e-03
f_7(5000)	DE	1.00e+01	8.10e+01	3.20e+01
	SaDE	0.00	3.30e-02	1.80e-01
	CDE	1.20e-05	1.80e-05	2.30e-05
	DEahcSPX	9.10	2.14e+01	1.23e+01
	DMSDLS	6.00	1.30e+01	3.70
	Pro DE	1.71e+01	2.47e+01	2.87e+01
	AEDE	4.68	1.28e+01	2.23
f_8(1500)	DE	3.40e-15	3.70e-14	4.10e-14
	SaDE	1.60e-32	3.50e-03	1.90e-02
	CDE	0.00	0.00	0.00
	DEahcSPX	6.39e-03	2.07e-02	8.46e-02
	DMSDLS	1.60e-32	1.60e-32	5.60e-48
	Pro DE	8.75e-31	6.43e-30	7.37e-30
	AEDE	0.00	0.00	0.00
f_9(1500)	DE	4.10e-14	2.90e-13	2.90e-13
	SaDE	8.10e-32	1.30e-32	5.60e-48
	CDE	0.00	0.00	0.00
	DEahcSPX	3.64e-32	1.71e-31	5.35e-31
	DMSDLS	1.30e-32	1.30e-32	5.60e-48
	Pro DE	0.00	1.53e-30	2.12e-30
	AEDE	0.00	0.00	0.00

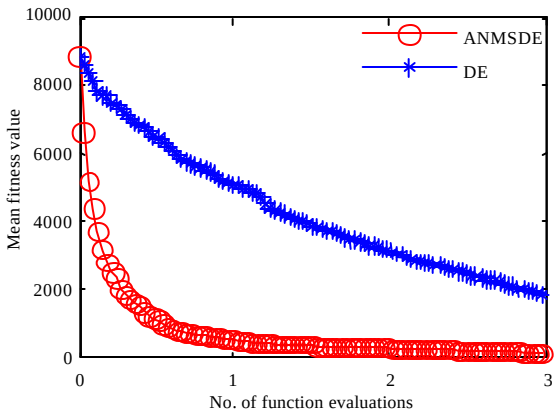

Fig. 2: Comparison of the evolution process between AEDE and classic DE on f_8

except classic DE, SaDE and ProDE are better in f_5 and CDE is better in f_6 and f_7. Thus, it can be concluded that AEDE is very competitive.

CONCLUSION

An efficient DE algorithm for function optimization is proposed in this paper. The simulations and comparisons with several state-of-the-art DE variants on nine benchmark functions demonstrate that the new algorithm can achive better balance between exploration and exploitation by multi-stratey differential mutation mechanism and promises competitive performance not only in convergence speed but also in solution quality.

ACKNOWLEDGMENT

Support from the National Natural Science Foundation of China (NO. 31170393), the Nature Science Foundation of Shaanxi Province (NO. 2012JM8023) and the Nature Science Special Foundation of Department of Education of Shaanxi Province (NO. 12JK0726) is gratefully acknowledged.

REFERENCES

Das, S. and P.N. Suganthan, 2011. Differential evolution: A survey of the State-of-the-art. IEEE Trans. Evol. Comput., 15: 4-31.

Epitropakis, M.G., D.K. Tasoulis, N.G. Pavlidis, V.P. Plagianakos and M.N. Vrahatis, 2011. Enhancing differential evolution utilizing Proximity-based mutation operators. IEEE Trans. Evol. Comput., 15: 99-119.

Epitropakis, M.G., V.P. Plagianakos and M.N. Vrahatis, 2008. Balancing the exploration and exploitation capabilities of the differential evolution algorithm. Proceedings of the IEEE Congress on Evolutionary Computation, June 1-6, 2008, Hong Kong, China, pp: 2686-2693.

Macready, W.G. and D.H. Wolpert II, 1998. Bandit problems and the exploration/exploitation tradeoff. IEEE Trans. Evol. Comput., 2: 2-22.

Mininno, E., F. Neri, F. Cupertino and D. Naso, 2011. Compact differential evolution. IEEE Trans. Evol. Comput., 15: 32-54.

Montgomery, J. and S. Chen, 2010. An analysis of the operation of differential evolution at high and low crossover rates. Proceedings of the IEEE Congress on Evolutionary Computation, July 18-23, 2010, Barcelona, Spain, pp: 1-8.

Noman, N. and H. Iba, 2008. Accelerating differential evolution using an adaptive local search. IEEE Trans. Evol. Comput., 12: 107-125.

Price, K.V., R.M. Storn and J.A. Lampinen, 2005. Differential Evolution: A Practical Approach to Global Optimization. Springer, Heidelberg, ISBN: 9783540209508, Pages: 538.

Qin, A.K. and P.N. Suganthan, 2005. Self-adaptive differential evolution algorithm for numerical optimization. Proc. IEEE Cong. Evol. Comput., 2: 1785-1791.

Storn, R. and K. Price, 1995. Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report. International Computer Science Institute, Berkley, TR-95-012, California.

Theodoridis, S. and K. Koutroumbas, 2008. Pattern Recognition. 4th Edn., Academic Press, USA., ISBN-13: 978-1597492720, Pages: 984.

Vesterstrom, J. and R. Thomsen, 2004. A comparative study of differential evolution, particle swarm optimization and evolutionary algorithms on numerical benchmark problems. Proc. Congr. Evol. Comput., 2: 1980-1987.

Wang, Y., Z.X. Cai and Q.F. Zhang, 2011. Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans. Evol. Comput., 15: 55-66.

Wang, Y.J., J.S. Zhang and G.Y. Zhang, 2007. A dynamic clustering based differential evolution algorithm for global optimization. Eur. J. Oper. Res., 183: 56-73.

Zhang, X.X., W.R. Chen and C.H. Dai, 2010. Dynamic Multi-group Self-adaptive differential evolution algorithm with local search for function optimization. Acta Electron. Sin., 38: 1825-1830.