2012 International Conference on Network and Computational Intelligence (ICNCI 2012)
IPCSIT vol. 46 (2012) © (2012) IACSIT Press, Singapore

A Self-Adaptive Differential Evolution Algorithm Based on Cloud Model

Jingjing Sun [1], Hongbin Dong [1+]and Yue Pan[2]

[1] Department of Computer Science and Technology, Harbin Engineering University, Harbin, China
[2]Science and Technology on Underwater Acoustic Antagonizing Laboratory, Beijing, China

Abstract: A self-adaptive Differential Evolution (CMSaDE) is proposed which can dynamically change mutation strategies and control parameters instead of manual settings. The performance of CMSaDE is investigated and compared with other well-known approaches and the results show that CMSaDE generally outperforms other DE algorithms nearly in all the benchmark functions.

Keywords: Evolution computation, Cloud model, Differential evolution, Numerical optimization.

1. Introduction

Differential Evolution(DE)is a general-purpose stochastic search method simulating natural selection and biological evolution, using the differences between randomly selected individuals as the source of random variations for a third individual[1].The performance of DE is sensitive to the choices of mutation strategies and control parameters that strongly affect the optimal solutions.

Cloud model chooses natural language as cut-in point,doing transformation between a qualitative concept and a set of quantitative numerical values,which simultaneously reflects randomness and fuzziness. Normal cloud model is the most important one, which is mainly used to combine with geneticalgorithmin the field of evolution computation.

Based on the outstanding characteristics of the cloud model,aiming at adjustments of mutation strategies and control parameters for DE,this paper proposed a self-adaptive differential evolution algorithm based on cloud model.

2. Cloud Model

2.1. Forward Normal Cloud Generator

The transformation of a qualitative concept to a set of quantitative numerical values is referred as forward cloud generator. Forward normal cloud generator produces drops according to the digital characters (Ex, En, He), respectively, expected value Ex, Entropy En and hpter entropy He,mainly described as follows.

Produce $En_i^{'}$ with normal distributions of mean En and standard deviation He, $En_i^{'} = NORM(En, He^2)$.

Produce x_i with normal distributions of mean Ex and standard deviation $En_i^{'}$ $x_i = NORM(Ex, En_i^{'2})$

Calculate $\mu_i = e^{-\frac{(x_i - Ex)^2}{2(En_i^{'})^2}}$.

+ Corresponding author. Tel.: + 13904646378.
E-mail address: donghongbinbjtu@gmail.com.

x_i is a drop with certainty μ_i.

Repeat the above steps until producing n drops.

2.2. Atomized Feature

While He is smaller, drops ,drops distribution looks like normal distribution; While He is larger, the drops of the cloud spread around, but many drops still stand in the central area of the cloud. This character is called atomized feature.

3. Classical DE

DE uses special difference operator to produce offspring from parents,mainly including mutation,crossover and selection, the detailed is not described any more here.

Price and Storm proposes about ten mutation strategies[7],we combine DE/rand/1 with DE/current to best/2 in this paper.DE/rand/1 is good at keeping population diversity, while DE/current to best/2 is good at convergence.

4. Self-adaptive Differential Evolution Algorithm Based on Cloud Model

The original DE only applies one mutation strategy and keep the control parameters constant,which easily cause DE to stagnate before finding a globally optimal solution. The proposed algorithm focuses on the adjustment of mutation strategies and control parameters in order to make DE be self-adaptive**.**

4.1. Self-adaptive mutation strategy

Inspired by evolutionary game theory,CMSaDE applies two deferent mutation strategies. In each generation,each individual chooses one of the two mutation strategies according to a certain probability distribution for producing offspring. The main operations are summarized as follows.

Probability Distribution Initialization

For each individual,the probability distribution is initialized:

$$\vec{\rho}_{i,0} = (\rho_{i,0}(1), \rho_{i,0}(2)), \quad i = 1, ..., NP$$

Where $\rho_{i,0}(1)$, $\rho_{i,0}(2)$ represents the probability of DE/rand/1and DE/current to best/2,generally initialized $(0.5, 0.5)$.

Mutation Strategy Selection

Each individual selects one mutation strategy h according to the probability distribution.

$$h = \begin{cases} DE/rand/1 & if\ rand(0,1) \le \rho_{i,G}(1) \\ DE/current\ to\ best/2 & else \end{cases}$$

Probability Distribution update

If individual $x_{i,G+1}$ comes from the offspring population, then we enhance the strategy h which used in the mutation.

$$\begin{cases} \rho_{i,G+1}(h) = \rho_{i,G}(h) + (1 - \rho_{i,G}(h))\gamma \\ \rho_{i,G+1}(l) = \rho_{i,G}(l) - \rho_{i,G}(l)\gamma, \ \forall l \ne h \end{cases}$$

If individual $x_{i,G+1}$ comes from the parent population, then we weaken the strategy h which used in the mutation.

$$\begin{cases} \rho_{i,G+1}(h) = \rho_{i,G}(h) - \rho_{i,G}(h)\gamma \\ \rho_{i,G+1}(l) = \rho_{i,G}(l) + \dfrac{1}{3}\rho_{i,G}(h)\gamma, \ \forall l \ne h \end{cases}$$

Where the parameter $\gamma \in (0,1)$ is used to control the mixed strategy distribution, $\sum_{k=1}^{4} \rho_i(k) = 1$ and

$\rho_i(k) \geq 0$.

4.2. Self-adaptive F

F mainly controls search foot, we can dynamically adjust F according to the present circumstance. For facilitating description, some necessary definitions we use can be found in [4].

F is adjusted on the basis of continuous normal generations and continuous abnormal generations , the main operations is as follows.

In each generation, F_i is generated by forward normal cloud generator for each individual x_i ,

$$F_i = CG(Ex, En, He), F_i \in [0,2]$$

where Ex is the mean of successful scale factors,that can produce offspring that can go into the next generation. Ex is initialized as 0.5,and adjusted following

$$Ex = (1-c) \cdot Ex + c \cdot mean_L(S_F)$$

where c is a constant in the range of $[0,1]$, $mean_L$ is Lehrer mean[7].

$$mean_L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F}$$

En and He respectively controls breadth and stability. When there appears cross-age elite, we can shrink breadth (decrease En)and strengthen the stability(decrease He),making the drops of cloud look like normal distribution ,which can increase accuracy and convergence; A simple way is to narrow K-fold to En , He adjusted according to[9]; When there no appears cross-age elite, that is Continuous normal generations more than λ_{local} , this demonstrates that the algorithm is stagnated, when it should skip the present neighborhood; A simple way is to Expand L times to En , He adjusted according to[9].When there is still no cross-age elite after local adjustment, that is Continuous normal generations more than $\lambda_{\mathrm{g}loal}$ $\lambda_{\mathrm{g}loal} \succ \lambda_{local}$,a reset of Ex, En, He is necessary.

4.3. Self-adaptive CR

In each generation, CR_i is randomly generated through normal distributions for each individual x_i

$$CR_i = N_i(CR_m, 0.1), CR_i \in [0,2]$$

where CR_m is the mean of successful crossover rates S_{CR} ,which can produce offspring that can go into the next generation. CR_m is initialized as 0.5,and adjusted following

$$CR_m = mean_A(S_{CR})$$

Where $mean_A$ is arithmetic mean.

5. Experiments and Analysis

CMSaDE is tested on 12 classical benchmark functions, described fully in[8], $f_1 - f_{12}$ in this paper correspond with f_{15} , f_{16} and $f_1 - f_{10}$ in [8]. The experiment is divided into three sections,respectively to test the accuracy, the convergence and the affection of F .

Experiment settings: CMSaDE is to compare with classical DE, SaDE[6];Classical DE uses DE/rand/1,F=0.5,CR=0.9;The settings of SaDE is the same with [6];NP=100,the dimension and the max generation is the same with[8].For the sake of fair comparisons, All the results are the average results of 50 independent runs.

Experiment 1 Accuracy Experiment

From the results, it can be seen that MMSDE outperforms other three algorithms on all test functions. Especially, the“Mean” and the “StdDev” is smaller than the other algorithms,except for f_{10} ,maybe,it is

because that f_{10} is a high dimension function and has one more minimums.

Experiment 2 Convergence Experiment

The convergence comparison of classical DE, SaDE and MMSDE on function f_1、f_3、f_8、f_9 is presented in Fig.1- Fig.4.It is obvious that MMSDE converges faster than other algorithms in the whole evolution, especially in the beginning.

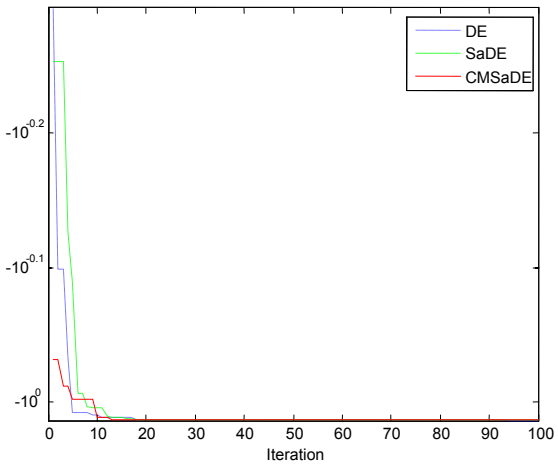

Fig. 1:The convergence comparison on function f1

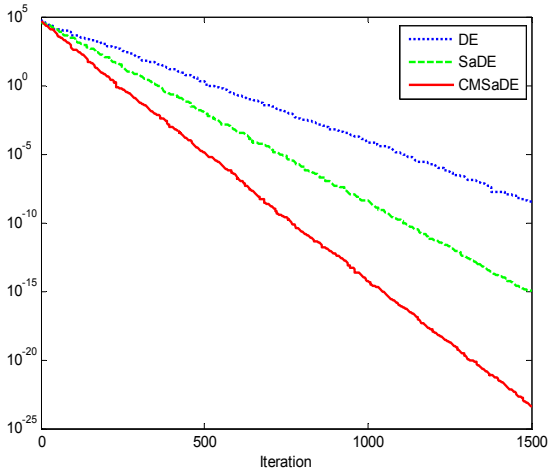

Fig. 2:The convergence comparison on function f3

Table.1:The average results achieved by DE, SaDE and CMSaDE

Functions	DE			SaDE			CMSaDE		
	Best	Mean Best	StdDev	Best	Mean Best	StdDev	Best	Mean Best	StdDev
f1	-1.0316	-1.0316	0	-1.0316	-1.0316	0	-1.0316	-1.0316	0
f2	0.398	0.398	0	0.398	0.398	0	0.398	0.398	0
f3	2.01E-14	5.10E-14	3.43E-14	5.26e-16	6.12e-016	1.19e-016	4.96e-25	3.10e-024	2.87e-024
f4	3.25E-10	4.95E-10	2.05E-10	1.04e-12	1.09e-012	4.54e-014	4.81e-19	8.12e-019	2.38e-019
f5	1.28E-11	2.61E-11	1.51E-11	1.05 E-18	1.12E-18	7.62E-19	1.86e-21	6.33e-020	1.28e-019
f6	0.09	0.1	0.21	0.0088	0.0090	1.82e-004	2.89e-4	0.0031	0.0045
f7	12.79	14.49	1.21	0.0023	0.0089	0.0120	6.07e-27	0.7973	1.6809
f8	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f9	3.95E-03	4.68E-03	9.78E-04	1.15E-0.2	1.24E-02	1.30E-03	2.17E-03	3.80E-03	1.20E-03
f10	-12550.69	-12539.88	78.34	-12596.5	-12596.5	0	-12569.5	-1.25e+004	100.6532
f11	39.26	56.01	16.14	**0**	**0**	**0**	**0**	**0**	**0**
f12	4.53E-08	7.04E-08	2.32E-08	1.30e-08	1.52e-008	2.07e-009	8.68e-13	2.47e-012	1.07e-012

Experiment 3 Effection of F Experiment

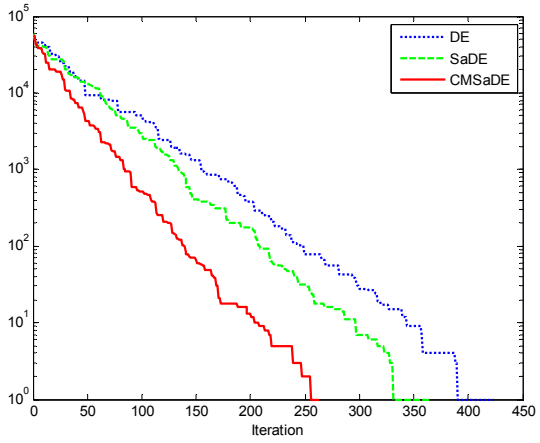

Fig. 3:The convergence comparison on function f8

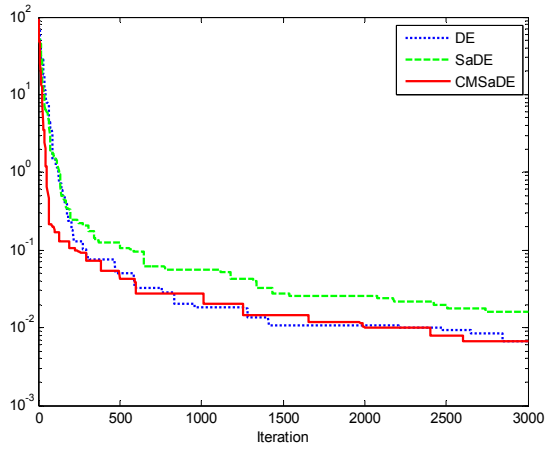

Fig. 4:The convergence comparison on function f9

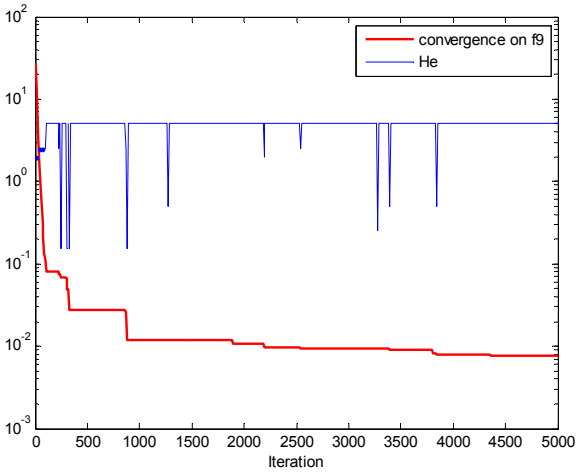

Fig. 5:He values during the iteration on function f_9

In order to prove that the adjustment of control parameters dispositive effect on the algorithm,Fig. 5 shows He values during the iteration on function f_9.It is obvious that ,when He changes,a new optimal solution is found.

6. Conclusion

This paper proposes a self-adaptive differential evolution algorithm based on cloud model and control parameters and mutation strategies can dynamically adjust. The experimental results prove that the algorithm has better performances than the other compared algorithms.

7. Acknowledgements

This work is partially supported by the National Natural Science Foundation of China under Grant (60973075), the Natural Science Foundation of Heilongjiang Province of China under Grant (F200937), and Ministry of Industry and Information Technology (B0720110002).

8. References

[1] R. Storn and K. V. Price, “Differential evolution–A simple and efficient adaptive scheme for global optimization over continuous spaces,” Institute of Company Secretaries of India, Chennai, Tamil Nadu. Tech. Report TR-95-012, 1995.

[2] H. A. Abbass, “The self-adaptive pareto differential evolution algorithm,” in Proc. IEEE Congr. Evol. Comput., vol. 1. Honolulu, HI, May2002, pp. 831–836.

[3] LI De-yi. No certainty of artificial intelligence [M]. National Defence Industry Press, 2005,PP.143.

[4] ZHANG Guang-wei, HE Rui and LIU Yu. Evolutionary algorithm Based on Cloud Model [J]. Chinese Journal of Computers. 2008, 31(7): 1082-1091.

[5] LI De-yi, SHI Xue-mei,WARDP,et a1. Soft inference mechanism based on cloud models[C] //Proc of the 1st International Workshopon Logic Programming and Soft Computing.1996:38-63.

[6] A. K. Qin and P. N. Suganthan, “Self-adaptive differential evolution algorithm for numerical optimization,” in Proc. IEEE Congr. Evol.Comput., vol. 2. Sep. 2005, pp. 1785–1791.

[7] Price, K., Storn, R., 2005. DE Web site. http://www.ICSI.Berke-ley.edu/~storn/code.html (visited 9 July 2005).

[8] Yao xin. Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation. 3(2):82-102, July, 1999.

[9] LI Yu, LI De-yi, ZHANG Guang-wei et al. Atomize Feature in Cloud Based Evolutionary Algorithm [J]. Chinese Journal of Electronic, 2009, 37(8):1651-1658.