An Adaptive Strategy Applied to Memetic Algorithms

Jingfeng Yan, Meilian Li, and Jin Xu

Abstract—**Memetic algorithms(MAs) represent one of the promising areas of evolutionary algorithms. However, there are many issues to be solved to design a robust MA. In this paper, we introduce an adaptive memetic algorithm, named GADE-DHC, which combines a genetic algorithm and a differential evolution algorithm as global search methods with a directional hill climbing algorithm as local search method. In addition, a novel strategy is proposed to balance the intensity of global search methods and local search method, as well as the ratio between genetic algorithm and differential evolution algorithm. Experiments on several benchmark problems of diverse complexities have shown that the new approach is able to provide highly competitive results compared with other algorithms.**

Index Terms—**Memetic algorithm, Differential evolution, Genetic algorithm, Directive hill climb, Adaptive strategy.**

I. Introduction

MEMETIC algorithms(MAs) is a kind of evolutionary algorithm which includes one or more local phases within its evolutionary cycle [1]. Global search(GS) algorithms based on population, such as genetic algorithm [2] and differential evolution algorithm [3], have been proven to be a kind of effective techniques in obtaining good results on a variety of problems. It is easy for global search methods to explore promising area of solution space, however, the ability to exploit neighbourhood area of a solution is poor. As a consequence, it often takes a relatively long time to locate the global optima which may have a low precision at most time. On the contrary, local search methods based on individual, such as hill climbing, tabu search, greedy algorithm, can exploit the neighbourhood area of an individual well and obtain sufficient precision, but tend to find the local optima. In order to utilize the ability of exploring and exploiting of an algorithm and achieve higher performance, many search algorithms use a hybrid of dedicated global search methods and local search methods. Hybrid genetic algorithm and local search method(GA-LS) which incorporates genetic algorithm with local improvement procedure, may be used to improve the performance of GAs. Such hybrids have been proven to be an efficient way to solve many problems. Studies in [4]-[9] have shown that GA-LS can not only find better solutions than simple GAs, but also search more efficiently.

As a matter of fact, local search methods(LSs) and global search methods(GSs) by themselves are known to work very differently with different design problems, even among problems from the same design domain [9]. On basis of this reason, simple hybrid global search methods with a single local search method doesn't always work. Therefore, many recent researches adopted multi-memes in their MAs. During the running of these algorithms, MAs control the selection of a meme adaptively, which results in robustness and capability of generalization. Recently, there are many different kinds of MAs which are introduced in [10]-[16] to solve numeric optimization problems. Additionally, MAs have also been applied to solve real world engineered project successfully. Strategies proposed in [10], [17], [18] show their advantages and effects in dealing with real world problems. However, almost all these new approaches consist of a global search method and multiple local search methods, none of them uses multiple global search method. Furthermore, many of them ignore to adjust the intensity between global search and local search methods, for example, the algorithm MAS2 in [10] use LSs to improve some individuals each time after GS is applied to population. MAs in [11][12] can adjust the usage frequency of GS and LSs adaptively, but they pay little attention to the characteristic of GSs and LSs.

Motivated by these phenomena, we present a novel memetic algorithm in this paper, which adopts multiple GSs, GA and DE, to do global search operators and directional hill climbing to do some local improvement. What's more, an adaptive way is introduced to control the ratio between GSs and LS as well as the selection of genetic algorithm and differential evolution algorithm. We take best individual's fitness improvement, average fitness improvement of population, evaluation calls and the phase of evolution into account in this adaptive way. Experiments demonstrate that this approach achieves competitive results in optimizing the well known benchmark functions.

The rest of this paper is organized as follow: Section II represents some related work on adaptive memetic algorithm. In Section III , we introduce the three components of GADE-DHC: a genetic algorithm, a differential evolution algorithm(JADE) and an improved directional hill climbing algorithm. And besides, the adaptive way to combine and balance these three components is also introduced in detail. Experimental studies on benchmark functions are conducted to verify the performance of our approach in Section IV. Section VI is mainly about the conclusions of this paper.

II. Previous Work

Recently, the general practice of hybridization in MA has tended to associate hybridization with adaptation. Adaptation of parameters and operators represents one of the most promising areas of computation in memetic algorithm [19]. Adaptive algorithms are capable of acclimatizing to suit a given problem without a priori knowledge by methodically

Manuscript received January 15, 2015; revised March 15, 2015. This work was supported in part by the Foundation of Henan Important Science Research Project under Grant No. 122102210488, Young Core Teachers of Xuchang University and the Foundation of Xuchang Science Research Project under Grant No. 1101029, and the Foundation of Education Department of Henan province Project under Grant No. 13A520748.

Jingfeng Yan, Meilian Li are with the Department of School of Information engineering, Xuchang University, Xuchang, Henan, 461000, China, PR china (email: xujin_cug@163.com).

utilizing acquired information about the matching of problems to produce and adjust themselves to the problem as the search progress. Algorithm 1 is the outline of adaptive MA. Actually, it is still difficult and challengeable to design a robust and efficient MA with so many issues to be addressed, for example:

Algorithm 1 Brief framework of adaptive MA

Generate the initial population;
while (stopping criteria not met) **do**
    Evaluates all individuals in the population;
    Select individuals from the parent population;
    Apply standard GA operators to create offspring;
    Select individuals in offspring to proceed with local improvement;
    for all selected individuals **do**
        select corresponding meme;
        apply this meme to improve the individual;
    end for
    Recombination the two populations;
end while

- Where and when should local search be applied within the evolutionary cycle ?
- Which individuals in the population should be improved by local search, and how should they be chosen ?
- How much computational effort should be allocated to each local search ?
- How to balance the ratio between global search and local search ?

And in this part, several adaptive memetic algorithms are introduced to show how they address some of these issues.

A. A Stochastic Approach, Biased Roulette Wheel:MA-S2

The biased roulette wheel strategy MAS2[10] is a stochastic approach which makes use of knowledge obtained online. At the beginning, it gives each local method an equal chance to hybridize with GA to locally improve some specified individuals. After this stage, sum up all ”reward” grouping by local search method. In subsequent optimization, a local search method will be chosen to improve individuals according to its previous performance, namely, its reward. And a biased roulette wheel is used to pick the subsequent local search method based on the rewards taken over all previous local searches.

Since the choice of LS is based on its reward, the biased roulette wheel strategy is generally a competitive strategy. At the same time, it also guarantees diversity in selection LS. By ensuring diverse LS methods participating in the search, the strategy promotes joint operation and cooperation among local search methods.

B. Co-evolving Memetic Algorithms

Krasnogor proposed a simple inheritance mechanism for discrete combinatorial search[8]. In this strategy, an individual's memetic material is encoded into its genetic part, it specifies the LS that will be applied to locally improve its carrier. During the evolving, the offspring's memetic material is decided by its parents and is the same with the parents who have a larger fitness. Its core idea is simple, but efficient.

What's more, Smith also worked on co-evolving MAs which use similar mechanisms to govern the choice of LSs represented in the form of rules [20][21]. These are forms of self-adaptive MA that evolve simultaneously the genetic material and the choice of LSs during the search.

C. Cost-Benefit-Based Adaptation

The cost-benefit-based adaptation mechanism[11][12] is for the case of scheduling local search methods. Initially, all memes have an equal chance to be selected. The relative fitness gain rfg and the required evaluation $eval$ are summed up. A normalised fitness function in the range of 0 and f_{max} is used which turns every problem to be a maximisation problem. rfg is the ratio between the obtained fitness improvement and the possible one, as shown in (1)

$$rfg = \frac{f_{LS} - f_{evo}}{f_{max} - f_{evo}} \qquad \frac{\sum rfg_{i,LS1}}{\sum eval_{i,LS1}} ... \frac{\sum rfg_{i,LSn}}{\sum eval_{i,LSn}} \tag{1}$$

where f_{LS} is the fitness obtained by the LS and f_{evo} the fitness of the offspring as produced by the evolution. The probabilities of applying the local search method are adjusted, if either each LS was used at minimum $usage_{min}$ times or there have been $matings_{max}$ mating in total since the last adjustment. The new relation between the local search method $LS1, ..., LSn$ is calculated as shown in (1).

III. The Proposal: GADE-DHC

In this part, we will introduce our proposed algorithm, GADE-DHC, in detail. At first, the three parts of our approach, GA, DE, DHC, are presented, especially DHC, which is an improved hill climbing algorithm recommended detailly. It can achieve optimum more precise and more fast compared with traditional hill climbing algorithm. And then, the details of how our proposed algorithm deal with those problems introduced in last chapter is also presented.

A. Components of GADE-DHC

1) Genetic Algorithm: Genetic algorithm is inspired by Darwin's theory about evolution. It is a heuristic method that mimics the process of natural selection. As the development of genetic algorithm, it has been widely used in optimization and search problems, such as Traveling Salesman Problem(TSP) [22], Vehicle Routing Problem [23], job-shop scheduling problem [24] etc.

In our approach, we mainly focus on using multi-point crossover operator of Genetic algorithm to spread good genes quickly.

2) Differential Evolution Algorithm: DE [25] is a simple, but efficient, evolution algorithm for global numerical optimization, It creates new candidate solutions by combining the parental individual with several other individuals of current population. A candidate replaces the parent only if it performs better in terms of fitness evaluation function. In DE algorithm, the core operator is the differential mutation operator. There are many mutation operators that have been proposed in [26], [27]. And it is widely used in many field. For example, Chuang, Li-Yeh [28] applies it to operon prediction.

In our memetic algorithm, we use an adaptive DE, named JADE [29], which implements a mutation strategy

"DE/current-to-$p$best" with optional archive and controls F and CR in an adaptive way. In "DE/current-to-$p$best" with archive, a mutation vector is produced as follows:

$$v_i = x_i + F_i * (x^p_{best} - x_i) + F_i * (x_{r1} - \widetilde{x_{r2}}) \tag{2}$$

where x^p_{best} is randomly chosen as one of the top 100p% individuals in the current population with $p \in (0, 1]$, and F_i is the mutation factor which is associate with x_i and is updated at each generation. $\widetilde{x_{r2}}$ is randomly chosen from the union of the current population and archive.

3) Directional Hill Climbing Algorithm: Hill Climbing algorithm is a mathematical optimization technique which belongs to greedy algorithm. It is an iterative algorithm that starts with an arbitrary solution of a problem, then attempts to find a better solution by incrementally searching its neighborhood area. If the change produces a better solution, an incremental change is made to the new solution, repeating until no further improvements can be found. Hill climbing is good for finding a local optimum, but it is not guaranteed to find the global optimum out of the search space.

In particular, we consider that a hill climbing algorithm is used to minimize a objective function $\boldsymbol{f(x)}$, where $\boldsymbol{x}$ is decision vector. At each iteration, hill climbing algorithm will search the neighbourhood of $\boldsymbol{x}$, generally speaking, in the following manner:

$$x_i = x_i + Gaussian() \tag{3}$$

where $Gaussian()$ function can create a fluctuation around x_i, and then, if $f(\boldsymbol{x_{new}}) < f(\boldsymbol{x_{old}})$, replace $\boldsymbol{x_{old}}$. Apparently, this method is inefficient. At most time, it won't generate a good solution, as a result, it will lead to many iterations to converge to optimum. Moreover, it often tend to trip into the local optimum.

In order to exploit the neighborhood of a given solution, we propose an improved hill climbing algorithm, named Directional Hill Climbing(DHC). It's a kind of oriented search technique which can achieve not only higher performance in time consumption but also precision, compared with traditional stochastic hill climbing. To be specific, for a given vector $\boldsymbol{x}$, we firstly perturb it slightly in a deterministic direction, and then evaluate it. If the fitness of new individual is better, we know that the direction of climbing hill is right, and then, the algorithm will continue climbing along this direction until no improvement is made. Otherwise, the new approach will climb in negative direction, because climbing along the negative direction will lead to an improvement in fitness at this time. Algorithm 2 shows the process of the improved algorithm.

where $nDirect$ is the number of directions to be tested for climbing, and in GADE-DHC, we set $nDirect = 30\%D$, it is taken time consumption for local search improvement into consideration. As GA can spread good genes among the population, it is not necessary to spend too many time detecting direction of each dimension of a vector $\boldsymbol{x}$. $direct[]$ is an array of directions associating with each vector $\boldsymbol{x}$, $direct[i] = 0$ means in x_i^{th} dimension space, stay unchanged; $direct[i] = 1$ denotes climbing along the positive direction; and $direct[i] = -1$, represents climbing along a negative direction.

Algorithm 2 Directional Hill Climbing Algorithm

```
int direct[]=0, evalCout=0,scaling=0.01;
for all i = 1 : nDirect do
    index = rndint(0 , D);
    xtmp = x;evalCout++;
    xtmp_index = xtmp_index + xtmp_index*scaling ;
    if (f(xtmp) < f(x)) then
        x = xtmp;
        direct[index] = 1;
    else
        direct[index] = -1;
    end if
end for
while (evalCout < evalLimit) do
    for all i = 1 : D do
        xtmp = x;
         evalCout++;
        xtmp_i = xtmp_i + xtmp_i*scaling*direct[i] ;
        if (f(xtmp) < f(x)) then
            x = xtmp;
        else
            scaling *= 0.5;
        end if
    end for
end while
```

B. How to Adjust the Frequency of the usage of GA and DE

GA and DE are two kinds of global search methods based on population, they have many in common. However, problems suitable for them to solve are different because of their characteristics in evolutionary operators. The multi-point crossover strategy of GA can make it easy for a parental individual to pass a fragment of good genes to its offspring, while DE have already been proven to be an efficient way to solve numeric optimization problems because of it capacity of exploring potential area. And the purpose of using these two components in GADE-DHC is also different. Therefore, it is necessary to adjust the frequency of the usage of GA and DE as algorithm progress. In order to keep cooperative and competitive between these two methods, we mainly take the following issues into consideration during the procedure of evolution in GADE-DHC.

- The improvement of the best individual's fitness.
- Average fitness improvement of the population.
- Stage of evolution.

Generally speaking, the greater improvement of best individual's fitness, the better performance of the global search algorithm. Nevertheless, the situation where the best individuals' fitness in previous population and current population remain unchanged, happens just as often. We bring in the concept of average fitness improvement of population so as to ease this situation. What's more, it is known to us that the improvement of individual's fitness is often bigger at initial stage of evolution than that at latter stage. Taking all these facts into consideration, we use $weight$ to measure the contribution of an algorithm after it applies to the population, it can be calculated as follow:

$$weight = \rho_1 * \left|\frac{curBest - preBest}{preBest}\right| + \rho_1 * \left|\frac{curMean - preMean}{preMean}\right| \tag{4}$$

Where, ρ_1, ρ_2 denote the coefficient to adjust the weight of these two parts, they reflects the importance between the improvement of the best individual's fitness and the average fitness of population; $curBest$ and $preBest$ represent the best individual's fitness of current population and previous population. $curMean$ and $preMean$ are the average fitness of the all individuals in current population and previous population, namely:

$$mean = \sum_{i=1}^{NP} fitness_i / NP \tag{5}$$

Where $fitness_i$ is the fitness of i^{th} individual in the population, and NP is the size of the population. After applying DE to the population, using formula (4) to calculate the $DEWeight$; and $GAWeight$, which is obtained after GA is applied to population, is calculated in the same way. Then, update pGD using formula (6).

$$pGD = pGD + pGD * \frac{GAWeight - DEWeight}{GAWeight + DEWeight} \tag{6}$$

Where pGD is the probability to apply GA to the population, Obviously, $1 - pGD$ is the probability to apply DE to the population. if $GAWeight > DEWeight$, we know that GA is more suitable to optimize current population, as a reward, we should increase the value of pGD. otherwise the the value of pGD should be decreased. In addition, we use $(GAWeight - DEWeight)/(GAWeight + DEWeight)$ to make sure that pGD won't change sharply because of a great improvement by accident as well as the degree of improvement of different stage of evolution. The value of pGD has a significant influence on the subsequence optimization.

C. How to Balance the Ratio Between GS and LS

GSs(GA and DE) which are based on population, can explore the whole space of solution. While LS(Directional Hill Climbing) which is based on individual, can exploit the neighbourhood of a solution well. Because of their natural characteristics, we consider these issues shown as below to adjust the intensity of GS and LS.

- The improvement of the best individual's fitness.
- function evaluation calls.
- the period to use these two approaches.

Firstly, it is still the improvement of the best individual's fitness that we should pay attention to. Still the greater improvement, the better. What's more, global search method(GA and DE) calls evaluation function NP times at each generation, while that of local search method is $evalLimit$ times. We think that an algorithm is better if the same improvement is made within less evaluation calls. therefore, we bring in $evalGS/evalLS$ to eliminate the unfairness caused by different function evaluate calls. Furthermore, LS is for local optimization, there is no need to use too frequently in the early stage of the evolution, but a higher usage frequency should be given to DHC in the latter stage of evolution. Consequently, we use $1-cureval/maxeval$ as a scale factor to decrease the probability to use GSs. Above all, we use formula (7) to balance the ratio between global search and local search method.

$$pGL = pGL + pGL * \frac{LSWeight - GSWeight}{LSWeight + GSWeight} \tag{7}$$

$$LSWeight = |\frac{curBest - preBest}{preBest}| * \frac{evalGS}{evalLS} \tag{8}$$

$$GSWeight = GSReward * (1 - \frac{cureval}{maxeval}) \tag{9}$$

$$GSReward = pGD * GAWeight + (1 - pGD) * DEWeight \tag{10}$$

Where $evalGS$ is evaluation calls of global search method at each generation, $evalLS$ is evaluation calls of local search method to locally improve an individual, $cureval$ and $maxeval$ represents the current evaluation calls and max evaluation calls.

D. GADE-DHC

GADE-DHC is a kind of memetic algorithm which integrates GA and DE as global search strategies with DHC as local search method. There is a training stage at the beginning of the algorithm, in this stage, GA and DE have an equal chance to be applied to improve the population. After this stage, the choice of a global search method applied to optimize the subsequence population is decided by its previous performance, which is changeable dynamically as the overall search progress. The contribution of a global search method on the population is gained based on (4). Then, GSs and DHC will compete to optimize the population according to pGL, if GS is chosen to optimize the population, then, either GA or DE is selected, else apply DHC to locally improve some individuals, the number of which is 5% of the population size. As we just want DHC to optimize several best individuals, instead of optimizing the whole population, just as GSs do, so it will be a waster of time if DHC is applied to improve too many individuals. Afterwards, update pGL using formula(7)-(10). Algorithm 3 shows the framework of our proposed algorithm.

IV. Experimental Results And Analysis

In this section, we present a numerical study to analyze the search behavior and performance of the GADE-DHC. Several commonly used continuous benchmark functions with diverse complexity, shown in Appendix A, are used here to demonstrate the efficacy of the proposed algorithm. Moreover, in order to see how GADE-DHC improves the efficiency, we also conduct some comparison study with JADE, MAS2 and other algorithms. All results presented in this section are obtained from 30 independent runs for each benchmark function.

A. Parameters Setting

In order to compare the results of GADE-DHC with that of other algorithms, in all experiments, we use the following parameters as shown in Table I. It is remarkable that there are some differences between MAS2 implemented in this paper and MAS2 in [10] which is for adaptive selecting local search methods, it applies a best local search method to locally improve subsequent individuals according to its

Algorithm 3 GADE-DHC

```
Generate the initial population;
while (stopping criteria not met) do
    if (g < Training stage) then
        GA and DE have an equal change to be applied to population;
        Sum the Weight of each global search method;
        Update the best solutions;
    else
        if (rndreal() < pGL) then
            if (g == Training stage) then
                Update pGD;
            end if
            if (rndreal() < pGD) then
                GA−>evolve();
            else
                DE−>evolve();
            end if
            Calculate weight of the GS using formula (4);
            Update pGD in the form of formula (6);
        else
            Select Individuals to be improved locally;
            for all selected individuals do
                DHC−>search();
                Update pGL using formula (7)-(10);
            end for
        end if
    end if
end while
```

TABLE I
PARAMETER SETTING FOR EACH PART OF OUR PROPOSED ALGORITHMS

GADE-DHC and MAS2 Parameters	
Global search	GA and JADE
Local search	DHC
Population size	50
Training stage	6
Prob between GS and LS	0.9 (for GADE-DCH)
ρ_1 and ρ_2	0.9 , 0.1
Genetic Algorithm parameters	
Selection scheme	Roulette wheel
Crossover probability	pCrossover = 0.9
Mutation probability	pMutation = 0.1
JADE parameters	
Top p% individual	p = 0.06
Other parameters	uCR = uF = 0.5, c=0.1
DHC parameters	
Evaluation Limit	15,20,30,40 for D=10,30,50,100
Scaling	0.01
nDirect	30%D

η which stands for its previous performance and can be calculated by formula (11)

$$\eta = \beta \frac{| pf - cf |}{\mu} \tag{11}$$

but there is only one local search method(DHC) in our algorithm, the MAS2 in our study is used to adaptively select global search methods(JADE and GA), and the η is obtained using formula (4). Except that, MAS2 in this paper is same with MAS2 in [10].

B. The Comparative Study

In this experiments, results obtained by GADE-DHC, MAS2, JADE, GADE, GADHC and DEDHC are shown in Table II-IV. Where GADE, GADHC, DEDHC consist of GA+DE, GA+DHC, DE+DHC respectively, they are different kind of combinations of these three components. The best result for each benchmark function is highlighted in boldface. And ”1vN” in Table II-IV means comparing the first algorithm(GADE-DHC) with the N^{th} algorithm, for example, the column of ”1v2” is the comparative results of GADE-DHC and MAS2. In addition, the Wilcoxon signed-rank test is used to compare the significance between two algorithms. In Table II-IV, the results are summarized as ”$w/t/l$” which denotes that our proposed GADE-DHC wins w functions, ties in t functions and loses in l functions, compared with the other five algorithms.

With respect to overall performance, from table II-IV we can see that our proposed approach can significantly obtain better results compared with other five algorithms. For example, for all test functions at $D = 30$, GADE-DHC wins in 11 out of 21 functions, ties in 4 functions and loses in 6 functions compared with MAS2, and wins in 15 functions, ties in 4 functions and loses in 2 functions compared with JADE. For all benchmark functions at $D = 50$, GADE-DHC wins in 16 functions, ties in 3 functions, loses in 2 functions and wins in 20 functions, ties in 1 function and loses 0 functions when compared with MAS2 and JADE respectively. Seen from Table II-IV, we can observe that:

- for f01-f04 these simple unimodal functions, DEDHC and MAS2 perform best, especially DEDHC. the reason is that our improved DHC can exploit the neighborhood area around best solutions with high efficiency in these unimodal functions. Every time after global search is applied to population, DHC is used to locally improve some better individuals in MAS2. And compared with GADE-DHC, GA is absent in DEDHC, which makes DHC have a higher chance to optimize best individuals. As a result, MAS2 and DEDHC can achieve a higher precision than that GADE-DHC do.
- For the multimodal functions(f10-f21), our proposed algorithm obtains the best results. For the reason that GADE-DHC can balance not only the intensity between global search method and local search method but also the usage frequency between GA and DE well. In MAS2, the reward is accumulative, as improvement in earlier stage of evolution is greater than that of the latter period, the ratio between GA and DE tends to be a constant or even a simple MA just with a single GS and a fixed LS. Obviously, the adaptation is not enough.

TABLE II

COMPARATIVE STUDY FOR FUNCTION F01-F21 AT D=30. THE MEAN AND DEVIATION FOR THESE SIX FUNCTIONS ARE PRESENTED. AND "+", "-", AND "=" INDICATE OUR APPROACH IS RESPECTIVELY BETTER THAN, WORSE THAN, OR SIMILAR TO ITS COMPETITOR ACCORDING TO THE WILCOXON SIGNED-RANK TEST AT LEVEL 0.05

	NFFES	GADE-DHC Mean(Dev)	MAS2 Mean(Dev)	JADE Mean(Dev)	GADE Mean(Dev)	GADHC Mean(Dev)	DEDHC Mean(Dev)	1v2	1v3	1v4	1v5	1v6
f01	100000	1.67e-025(3.73e-025)	1.31e-037(4.12e-037)	2.25e-023(2.38e-023)	2.08e-023(1.84e-023)	5.63e+000(5.41e+000)	**1.80e-070(6.74e-070)**	-	+	+	+	-
f02	100000	2.86e-024(6.86e-024)	2.76e-051(1.20e-050)	5.56e-022(6.68e-022)	4.36e-022(2.79e-022)	9.49e+001(1.07e+002)	**4.10e-064(1.78e-063)**	-	+	+	+	-
f03	100000	2.73e-021(5.90e-021)	5.35e-055(1.94e-054)	1.50e-019(1.70e-019)	6.90e-019(4.34e-019)	1.23e+005(1.21e+005)	**2.34e-045(1.02e-044)**	-	+	+	+	-
f04	100000	1.78e-028(5.34e-028)	9.30e-058(4.05e-057)	9.55e-024(1.03e-023)	3.39e-023(2.22e-023)	5.84e+000(6.73e+000)	**4.42e-054(1.92e-053)**	-	+	+	+	-
f05	100000	2.40e-014(3.39e-014)	2.43e-003(1.80e-003)	**2.00e-015(1.95e-015)**	3.64e-015(3.72e-015)	3.2e+001(4.22e+001)	4.10e-011(4.31e-011)	+	-	-	+	+
f06	200000	**5.29e-009(5.64e-009)**	1.31e+000(1.52e-001)	1.15e-001(8.82e-003)	1.13e-004(6.02e-005)	1.09e+001(4.16e+000)	2.82e-002(1.39e-002)	+	+	+	+	+
f07	100000	2.72e-040(7.33e-040)	**1.01e-066(4.40e-066)**	2.34e-010(3.54e-010)	7.78e-011(6.66e-011)	5.97e-001(2.97e-001)	1.29e-047(5.03e-047)	-	+	+	+	-
f08	100000	**0(0)**	**0(0)**	**0(0)**	**0(0)**	2.85e+001(3.35e+001)	**0(0)**	=	=	=	+	=
f09	200000	6.64e-008(2.14e-007)	**1.71e-021(3.60e-021)**	6.71e+000(2.66e+000)	6.37e+000(2.54e+000)	4.41e+001(3.71e+001)	5.02e-001(1.24e+000)	-	+	+	+	+
f10	150000	**0(0)**	5.09e-008(1.34e-007)	**0(0)**	7.40e-004(2.22e-003)	7.97e-001(3.27e-001)	**0(0)**	+	=	+	+	=
f11	100000	**5.42e-015(2.61e-015)**	4.93e-009(6.07e-009)	1.59e-012(7.30e-013)	1.56e-012(9.52e-013)	1.10e+000(8.70e-001)	9.05e-014(1.12e-013)	+	+	+	+	+
f12	50000	**8.88e-016(3.49e-015)**	6.17e-013(1.19e-012)	5.46e+001(2.11e+000)	2.84e-009(4.69e-009)	2.96e+000(3.80e+000)	4.34e+001(3.76e+000)	+	+	+	+	+
f13	150000	**0(0)**	8.93e+001(2.11e-002)	8.85e+000(7.15e-001)	**0(0)**	**0(0)**	1.20e+001(6.67e-001)	+	+	=	=	+
f14	200000	**8.18e-005(1.36e-020)**	**8.18e-005(1.36e-020)**	1.37e+003(2.31e+002)	**8.18e-005(1.36e-020)**	8.18e-005(2.18e-011)	1.30e+003(2.45e+002)	=	+	=	=	+
f15	150000	**0(0)**	2.18e-005(9.28e-005)	3.92e-006(1.42e-005)	**0(0)**	2.37e+000(9.24e-001)	**0(0)**	+	+	=	+	=
f16	100000	1.19e+001(2.80e+000)	**3.21e-001(1.94e-001)**	3.98e-001(0)	1.37e+001(2.61e+000)	2.23e+002(7.61e+001)	1.34e+001(4.31e+000)	-	-	=	=	=
f17	200000	**1.57e-032(0)**	**1.57e-032(0)**	**1.57e-032(0)**	**1.57e-032(0)**	3.66e-002(4.48e-002)	**1.57e-032(0)**	=	=	=	+	=
f18	200000	**1.35e-032(0)**	**1.35e-032(0)**	**1.35e-032(0)**	**1.35e-032(0)**	5.81e-001(5.54e-001)	**1.35e-032(0)**	=	=	=	+	=
f19	200000	**5.25e-015(7.80e-015)**	8.03e-013(6.41e-013)	4.62e-004(5.21e-004)	8.22e-015(1.60e-014)	4.14e-003(1.14e-002)	3.89e-003(4.25e-003)	+	+	+	+	+
f20	200000	**3.72e-002(6.94e-018)**	7.20e-002(1.46e-002)	5.77e-002(2.05e-002)	2.16e-001(5.30e-002)	1.58e-001(6.09e-002)	7.61e-002(8.93e-003)	+	+	+	+	+
f21	200000	**5.77e-002(1.29e-002)**	2.74e-001(2.21e-002)	2.04e-001(3.64e-002)	1.05e+000(5.76e-002)	6.29e-001(8.60e-002)	2.61e-001(4.97e-002)	+	+	+	+	+
w / t / l		-	10 / 4 / 7	15 / 4 / 2 /	13 / 7 / 1	18 / 3 / 0	10 / 6 / 5					

TABLE III

COMPARATIVE STUDY FOR FUNCTION F01-F21 AT D=50. THE MEAN AND DEVIATION FOR THESE SIX FUNCTIONS ARE PRESENTED. AND "+", "-", AND "=" INDICATE OUR APPROACH IS RESPECTIVELY BETTER THAN, WORSE THAN, OR SIMILAR TO ITS COMPETITOR ACCORDING TO THE WILCOXON SIGNED-RANK TEST AT LEVEL 0.05

	NFFES	GADE-DHC Mean(Dev)	MAS2 Mean(Dev)	JADE Mean(Dev)	GADE Mean(Dev)	GADHC Mean(Dev)	DEDHC Mean(Dev)	1v2	1v3	1v4	1v5	1v6
f01	100000	6.12e-036(2.26e-035)	2.07e-033(8.91e-033)	4.49e-025(2.99e-025)	7.56e-026(7.34e-026)	3.43e+000(3.44e+000)	**9.00e-082(3.92e-081)**	+	+	+	+	-
f02	100000	9.79e-033(1.61e-032)	2.78e-038(1.10e-037)	8.81e-024(4.78e-024)	3.20e-024(4.15e-024)	1.03e+002(9.62e+001)	**2.94e-091(1.28e-090)**	-	+	+	+	-
f03	100000	3.54e-031(5.97e-031)	5.12e-013(2.19e-012)	8.23e-021(1.05e-020)	1.39e-021(1.15e-021)	2.40e+005(1.59e+005)	**4.08e-066(1.78e-065)**	+	+	+	+	-
f04	100000	3.59e-035(1.07e-034)	7.31e-029(2.21e-028)	4.64e-025(3.25e-025)	7.63e-026(4.73e-026)	6.72e+000(6.24e+000)	**8.04e-088(3.50e-087)**	+	+	+	+	-
f05	100000	**8.51e-017(9.37e-017)**	1.92e-002(1.45e-002)	1.15e-008(1.41e-008)	5.10e-011(4.17e-011)	4.69e+001(4.07e+001)	5.69e-005(5.47e-005)	+	+	+	+	+
f06	200000	3.03e-002(4.01e-002)	7.08e-002(1.00e-001)	5.23e+000(6.69e-001)	**7.90e-006(4.54e-006)**	1.63e+001(5.49e+000)	5.53e+000(6.08e-001)	+	+	-	+	+
f07	100000	**1.81e-076(4.31-076)**	1.60e-010(6.41e-010)	1.38e-012(9.90e-013)	5.00e-013(3.46e-013)	5.03e-001(2.78e-001)	6.47e-060(2.59e-059)	+	+	+	+	+
f08	100000	**0(0)**	**0(0)**	1.55e+000(9.73e-001)	**0(0)**	4.50e+001(4.15e+001)	3.1e+000(1.73e+000)	=	+	=	+	+
f09	200000	5.37e-020(1.17e-019)	**4.28e-022(5.59e-022)**	1.24e+001(7.67e+000)	6.99e+000(6.35e+000)	4.61e+001(7.54e+000)	9.77e-003(4.26e-002)	+	-	+	+	+
f10	150000	**0(0)**	1.62e-015(4.71e-015)	4.56e-003(5.30e-003)	1.48e-003(2.96e-003)	6.64e-001(3.38e-001)	**0(0)**	+	+	+	+	=
f11	100000	**4.00e-015(0)**	3.88e-007(9.15e-007)	1.09e+000(4.09e-001)	7.03e-014(3.196e-014)	1.81e+000(1.50e+000)	1.31e-014(9.16e-015)	+	+	+	+	+
f12	50000	**0(0)**	2.78e-010(6.028e-010)	1.29e+002(5.69e+000)	4.09e-009(2.48e-009)	3.39e+000(3.10e+000)	1.08e+002(9.67e+000)	+	+	+	+	+
f13	150000	**0(0)**	3.13e-002(5.77e-002)	2.19e+001(1.44e+001)	**0(0)**	**0(0)**	2.76e+001(1.60e+000)	+	+	=	=	+
f14	200000	**1.36e-004(2.71e-020)**	**1.36e-004(2.71e-020)**	3.65e+003(1.47e+002)	**1.36e-004(2.71e-020)**	1.36e-004(4.00e-010)	3.48e+003(3.32e+002)	=	+	=	=	+
f15	150000	**0(0)**	3.54e-006(5.83e-006)	7.18e-001(3.29e-001)	**0(0)**	4.39e+000(1.57e+000)	7.13e-001(3.75e-001)	+	+	=	+	+
f16	100000	3.98e-001(0)	**2.49e-001(1.49e-001)**	1.53e+000(3.31e-001)	1.79e+000(4.99e-001)	7.58e+000(3.97e+000)	1.24e+000(3.48e-001)	=	+	+	+	+
f17	200000	**9.42e-033(2.74e-048)**	9.93e-033(1.08e-033)	6.22e-003(1.87e-002)	**9.42e-033(2.74e-048)**	4.23e-001(9.49e-001)	1.11e-032(2.85e-033)	=	+	=	+	+
f18	200000	**1.35e-032(0)**	1.47e-032(2.47e-033)	**1.35e-032(0)**	**1.35e-032(0)**	1.30e+000(1.69e+000)	**1.35e-032(0)**	=	=	=	+	=
f19	200000	3.96e-014(3.51e-014)	2.29e-012(1.799e-012)	1.20e-007(5.22e-007)	**2.03e-022(3.95e-022)**	5.00e-002(6.74e-002)	4.41e-002(4.05e-002)	+	+	-	+	+
f20	200000	**3.72e-002(6.94e-018)**	1.85e-001(2.87e-002)	1.83e-001(3.14e-002)	4.16e-001(3.23e-002)	2.59e-001(9.18e-002)	1.83e-001(3.52e-002)	+	+	+	+	+
f21	200000	**9.65e-002(3.10e-002)**	3.59e-001(3.14e-002)	6.35e-001(5.69e-002)	1.42e+000(4.54e-001)	9.84e-001(1.54e-001)	6.21e-001(5.91e-002)	+	+	+	+	+
w / t / l		-	15 / 5 / 1	19 / 1 / 1 /	13 / 6 / 1	19 / 2 / 0	15 / 2 / 4					

TABLE IV
COMPARATIVE STUDY FOR FUNCTION F01-F21 AT D=100. THE MEAN AND DEVIATION FOR THESE SIX FUNCTIONS ARE PRESENTED. AND "+", "-", AND "=" INDICATE OUR APPROACH IS RESPECTIVELY BETTER THAN, WORSE THAN, OR SIMILAR TO ITS COMPETITOR ACCORDING TO THE WILCOXON SIGNED-RANK TEST AT LEVEL 0.05

	NFFES	GADE-DHC Mean(Dev)	MAS2 Mean(Dev)	JADE Mean(Dev)	GADE Mean(Dev)	GADHC Mean(Dev)	DEDHC Mean(Dev)	1v2	1v3	1v4	1v5	1v6
f01	100000	4.99e-057(7.31e-057)	1.43e-075(6.22e-075)	4.73e-018(5.35e-018)	1.28e-025(5.95e-026)	5.47e+000(3.84e+000)	**3.34e-197(0)**	-	+	+	+	-
f02	100000	6.26e-054(1.28e-053)	2.38e-063(1.04e-062)	4.28e-016(4.38e-016)	6.34e-024(4.18e-024)	5.12e+002(4.60e+002)	**6.84e-140(2.98e-139)**	-	+	+	+	-
f03	100000	1.76e-051(2.60e-051)	3.49e-058(1.52e-057)	4.69e-014(5.48e-014)	6.92e-021(4.06e-021)	1.45e+006(1.12e+006)	**3.89e-143(1.69e-142)**	-	+	+	+	-
f04	100000	6.46e-057(1.41e-056)	1.12e-056(4.74e-056)	2.22e-018(2.19e-018)	1.01e-025(7.44e-026)	8.46e+000(7.71e+000)	**2.96e-186(0)**	+	+	+	+	-
f05	100000	**8.20e-017(1.30172e-016)**	2.36e-001(3.22e-001)	5.49e+000(3.18e+000)	1.00e-004(4.59e-005)	8.45e+001(8.20e+001)	3.63e+001(1.78e+001)	+	+	+	+	+
f06	200000	**2.93e-001(2.32e-001)**	9.46e-001(1.19e+000)	1.86e+001(1.10e+000)	1.07e+001(2.12e-001)	3.93e+001(6.09e+000)	1.80e+001(9.74e-001)	+	+	+	+	+
f07	100000	6.78e-102(2.33e-101)	**4.06e-178(0)**	7.58e-007(1.04e-006)	7.31e-015(3.80e-015)	6.23e-001(3.37e-001)	8.28e-123(3.61e-122)	-	+	+	+	-
f08	100000	**0(0)**	**0(0)**	7.17e+001(2.03e+001)	**0(0)**	5.50e+001(4.97e+001)	3.10e+001(2.75e+001)	=	+	=	+	+
f09	200000	**1.88e-028(2.54e-028)**	1.32e-027(2.12e-027)	1.32e+002(2.85e+001)	5.76e+001(3.51e+001)	1.81e+002(1.34e+002)	2.14e-022(5.71e-022)	=	+	+	+	+
f10	150000	**0(0)**	3.68e-014(9.43e-014)	4.68e-003(6.69e-003)	1.85e-003(3.20e-003)	7.45e-001(3.89e-001)	**0(0)**	+	+	+	+	=
f11	100000	**4.35e-015(1.07e-015)**	4.35e-009(6.89e-009)	3.46e+000(2.69e-001)	1.60e-013(7.91e-014)	6.97e-001(7.42e-001)	2.37e+000(6.10e-001)	+	+	+	+	+
f12	50000	**0(0)**	4.14e+000(7.49e+000)	3.68e+002(9.36e+001)	7.25e-006(7.49e-006)	7.16e+000(7.46e+000)	2.76e+002(2.79e+001)	+	+	+	+	+
f13	150000	**0(0)**	6.80e-001(7.49e-001)	9.50e+001(2.96e+000)	**0(0)**	**0(0)**	1.33e+002(4.78e+000)	+	+	=	=	+
f14	200000	**2.73e-004(1.08e-019)**	**2.73e-004(1.08e-019)**	1.29e+004(2.58e+002)	**2.73e-004(1.08e-019)**	**2.73e-004(1.08e-019)**	1.33e+004(9.64e+002)	=	+	=	=	+
f15	150000	**0(0)**	4.11e-004(1.36e-003)	1.21e+001(1.43e+000)	**0(0)**	8.41e+000(3.18e+000)	1.30e+001(1.45e+000)	+	+	=	+	+
f16	100000	**3.98e-001(6.19e-005)**	4.40e-001(2.53e-001)	9.71e+000(2.50e+000)	7.49e+000(1.15e+000)	9.14e+000(6.86e+000)	3.58e+000(8.82e-001)	=	+	+	+	+
f17	200000	**4.71e-033(1.37e-048)**	2.40e-030(5.50e-030)	9.50e-002(1.13e-001)	4.79e-033(1.55e-034)	7.84e-001(1.16e+000)	2.64e-002(2.99e-002)	+	+	=	+	+
f18	200000	**1.35e-032(0)**	9.71e-032(5.34e-032)	7.85e-001(1.14e+000)	**1.35e-032(0)**	7.15e+000(1.13e+001)	1.57e-029(6.82e-029)	=	+	=	+	+
f19	200000	2.41e-014(2.34e-014)	8.80e-014(6.17e-014)	1.20e-014(6.90e-015)	**2.38e-027(4.63e-027)**	7.05e-004(3.07e-003)	3.17e+000(3.33e+000)	+	=	-	+	+
f20	200000	**4.54e-002(1.64e-002)**	4.29e-001(2.41e-002)	4.46e-001(1.53e-002)	4.95e-001(1.00e-002)	3.07e-001(8.42e-002)	4.49e-001(1.02e-002)	+	+	+	+	+
f21	200000	**1.92e-001(5.55e-002)**	4.89e-001(3.42e-002)	1.13e+000(4.26e-002)	1.87e+000(5.30e-002)	2.18e+000(9.52e-002)	1.15e+000(4.58e-002)	+	+	+	+	+
w / t / l		-	12 / 5 / 4	20 / 1 / 0 /	14 / 6 / 1	19 / 2 / 0	15 / 1 / 5					

TABLE V
FITNESS OF BEST SOLUTIONS WHEN FES=LE3, FES=LE4, FES=1E5 FOR BENCHMARK FUNCTIONS F01-F11 AT D=50,EACH FUNCTION RUNS 30 TIMES INDEPENDENTLY

FES \ Prob		f01	f02	f03	f04	f05	f06	f07	f08	f09	f10	f11
1e3	5th	1.7e+004	3.52e+005	2.68e+008	1.58e+004	2.98e+004	5.87e+001	2.02e+002	1.38e+004	1.4e+009	1.18e+002	1.54e+001
	15th	1.92e+004	4.52e+005	3.46e+008	2.16e+004	3.94e+004	6.25e+001	3.05e+002	1.84e+004	1.89e+009	1.83e+002	1.65e+001
	25th	2.32e+004	5.38e+005	5.15e+008	2.51e+004	5.43e+004	6.96e+001	4.12e+002	2.07e+004	2.96e+009	2.38e+002	1.77e+001
	mean	2.01e+004	4.62e+005	3.91e+008	2.08e+004	4.04e+004	6.34e+001	3.52e+002	1.89e+004	2.21e+009	1.83e+002	1.65e+001
	std	4e+003	1.34e+005	1.44e+008	5.22e+003	1.11e+004	5	1.78e+002	6.14e+003	8.62e+008	5.37	9.93e-001
1e4	5th	1.07e-009	9.72e-007	1.76e+001	9.51e-004	3.86	4.26	7.51e-011	0	6.08e+001	1.09e-010	5.81e-006
	15th	2.26e-003	7.85e-002	3.99e+001	3.97e-003	8.17	6.35	2.31e-008	0	1.44e+002	1.81e-003	3.52e-003
	25th	1.08e-002	2.33e-001	8.67e+001	6.39e-003	2.72e+001	1.13e+001	4.81e-007	1	2.81e+004	7.92e-003	4.71e-002
	mean	4.89e-003	1.16e-001	5.25e+001	4.6e-003	1.37e+001	8.18	2.12e-005	6e-001	1.73e+004	4.04e-003	2.92e-002
	std	6.08e-003	1.17e-001	4.78e+002	3.6e-003	1.26e+001	5.03	7.74e-005	1.2	1.04e+004	5.39e-004	6.71e-002
1e5	5th	1.1e-192	2.32e-222	2.02e-129	7.3e-157	1.93e-042	6.32e-005	1.38e-188	0	2.93e-028	0	3.99e-015
	15th	1.13e-139	2.35e-102	2.11e-075	6.33e-090	6.91e-041	9.48e-003	7.28e-179	0	7.26e-024	0	3.99e-015
	25th	1.47e-086	3.24e-080	2.92e-069	1.39e-078	2.09e-0.38	8.28e-002	5.08e-172	0	9.49e-020	0	3.99e-015
	mean	3.8e-077	3.41e-070	1.47e-068	2.69e-075	5.64e-037	2.02e-001	1.09e-164	0	2.75e-015	0	3.99e-015
	std	2.06e-076	1.65e-069	4e-068	1.23e-074	2.96e-036	8.05e-001	0	0	1.48e-014	0	0

While the ratio in GADE-DHC is different in each generation, it always changes dynamically. Furthermore, the intensity between global search method and local search method in MAS2 stays unchanged all the time. DHC is used to improve some better individuals each time after GS is applied to optimize the population. However, an algorithm tends to trap into the local optimum if local search method is used too frequently in the early stage of evolution. As a consequence, the performance of GADE-DHC is something better than that of MAS2.

- For these 21 functions, MAs(GADE-DHC, MAS2 and DEDHC which is a simple MA) outperform JADE, because MAs can govern both exploration which is important to ensure global reliability and exploitation which concentrates the search effort around neighborhood of best solutions. In addition, the competition between GA and DE can guarantee a right global search method at each generation of evolution. According to the performance of DEDHC and JADE, we know that the improved DHC is effective.
- Any part of GADE-DHC is necessary. GADE, GADHC, DEDHC consist of two parts of these three components, but they are all worse than GADE-DHC. It will lead to loss in performance, if any part is absent.
- There is little loss in performance of MAs for majority

TABLE VI
FITNESS OF BEST SOLUTIONS WHEN FES=LE3, FES=LE4, FES=1E5 FOR BENCHMARK FUNCTIONS F12-F21 AT D=50,EACH FUNCTION RUNS 30 TIMES INDEPENDENTLY

FES \ Prob		f12	f13	f14	f15	f16	f17	f18	f19	f20	f21
1e3	5th	3.81e+002	2.97e+002	1.15e+004	4.99e+002	1.56e+003	6.18e+006	2.12e+007	4.74e+001	5e-001	1.67
	15th	4.12e+002	3.64e+002	1.25e+004	5.43e+002	1.95e+003	1.34e+007	4.32e+007	5.24e+001	5e-001	1.75
	25th	4.61e+002	4.04e+002	1.31e+004	5.68e+002	2.3e+003	2.24e+007	7.3e+007	5.67e+001	5e-001	1.84
	mean	4.16e+002	3.61e+002	1.24e+004	5.39e+002	1.99e+003	1.43e+007	4.65e+007	5.23e+001	5e-001	1.75
	std	3.7e+001	2.15e+001	8.36e+002	3.65	5.27e+002	8.92e+006	2.46e+007	5.15	4.78e-005	6.73e-002
1e4	5th	8.24e-008	0	7.06e-002	1.42	3.97e-001	6.73e-004	9.07e-003	3.29e-003	3.46e-001	1.28
	15th	1.46e-002	0	2.04e-001	2.31	8.95e-001	1.35e-003	3.65e-002	2.31e-002	3.96e-001	1.36
	25th	3.95e-002	0	3.8e-001	3.87	8.95e-001	5.25e-003	7.94e-002	5.62e-002	4.15e-001	1.44
	mean	1.42e-001	4.21e-002	2.57e-001	2.62	7.86e-001	3.83e-003	6.65e-002	3.63e-002	3.82e-001	1.38
	std	4.28e-001	1.16e-001	2.03e-001	1.21	3.29e-001	5.83e-003	8.33e-002	3.46e-002	3.8e-002	9.49e-002
1e5	5th	0	0	1.36e-004	0	3.97e-001	9.42e-033	1.34e-032	9.49e-183	3.72e-002	1.01e-001
	15th	0	0	1.36e-004	0	3.97e-001	9.42e-033	1.34e-032	3.05e-014	3.72e-001	1.36e-001
	25th	0	0	1.36e-004	0	3.97e-001	9.42e-033	1.34e-032	9.44e-014	3.72e-001	2.51e-001
	mean	0	0	1.36e-004	1.81e-003	3.75e-001	9.42e-033	1.34e-032	6.62e-014	3.86e-002	1.72e-001
	std	0	0	0	9.49e-003	8.78e-002	2.74e-048	0	9.92e-014	7.35e-003	7.82e-002

functions with the increment of vector's dimension, while the loss aggrandize as the dimension increases in JADE. It is because that GA is integrated into MAs, which can spread good genes among the population quickly.

C. Parameter study

In the previous comparative study, ρ_1 , ρ_2 are set to 0.9 and 0.1. Nevertheless, it is worth mentioning that, the values of ρ_1 , ρ_2 may have an influence on the results of our proposed approach. they are the coefficient of $weight$ which directly affects the pGD and pGL, that is to say, the value of ρ_1 and ρ_2 can impact on the performance of our algorithm. Therefore, in order to evaluate the influence to our approach, we set different ρ_1 and ρ_2. To save space, we only compare the results for f01-f21 at D = 50. All other parameters are set the same as shown in Table I.

Table-VII shows the influence of different values of (ρ_1, ρ_2) to our approach, The mean and the standard deviation of the results obtained by each algorithm for f01-f21 are summarized in this table. ρ_1 , ρ_2 is the weight that we assign to the improvement of average fitness of best individuals and the whole population. In general, the improvement of the former is smaller than the latter, thus, we set these different pairs of (ρ_1, ρ_2). We can see that the value of (ρ_1, ρ_2) indeed have some influence to the algorithm, when (ρ_1 , ρ_2) is (1,0), that is to say, we just focus on the improvement of best individuals, it is somewhat worse than that we take both two into consideration. And besides, it has only little impact when neither of them is 0.

In order to represent the precision as well as convergence speed of these algorithms, we exhibit the searching trace of algorithms for multimodal benchmark functions. From Fig.1 to Fig.12, we can see that the GADE-DHC performs best both in search precision and convergence speed. At the aspect of searching precision, it just loses 1 function(Alpine

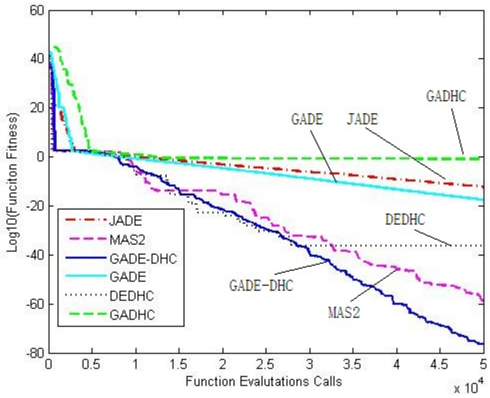

Fig. 1. Search trace for minimizing 50-D Schwefel2.22 function.

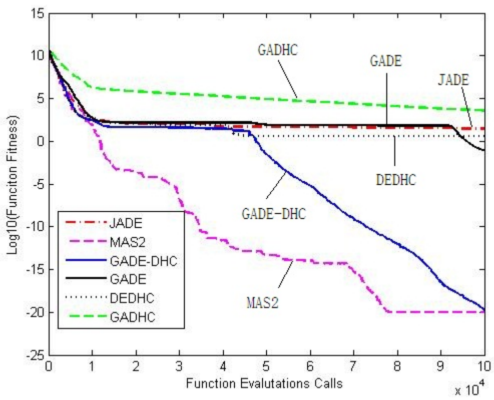

Fig. 2. Search trace for minimizing 50-D Rosenbrock function.

TABLE VII
THE INFLUENCEMENT OF DIFFERENT VALUES OF (ρ_1, ρ_2) FOR BENCHMARK FUNCTIONS F01-F21,EACH FUNCTION RUNS 30 TIMES INDEPENDENTLY, AT D=50

Prob \ (ρ_1, ρ_2)	(1 , 0)	(0.99 , 0.01)	(0.98 , 0.02)	(0.9 , 0.1)	(0.8 , 0.2)
f01	1.57e-033(2.40e-033)	**1.69e-038(5.96e-038)**	1.48e-035(4.63e-035)	1.89e-035(5.90e-035)	2.00e-037(7.38e-037)
f02	3.83e-032(7.69e-032)	**5.23e-035(2.09e-034)**	2.91e-033(6.10e-033)	1.39e-032(2.23e-032)	2.80e-033(8.36e-033)
f03	**6.96e-032(1.99e-031)**	1.08193e-029(1.48592e-029)	4.42e-030(8.70e-030)	5.19e-031(8.71e-031)	1.04e-029(1.46e-029)
f04	4.53e-034(8.25e-034)	8.29e-035(2.31e-034)	2.36e-034(4.87e-034)	6.2e-035(1.51e-034)	**2.94e-039(7.37e-039)**
f05	1.19e-016(1.58e-016)	3.91e-017(5.36e-017)	**1.00e-017(1.24e-017)**	1.15e-016(1.44e-016)	1.24e-016(1.76e-016)
f06	2.69e-002(2.87e-002)	3.10e-002(4.47e-002)	3.45e-002(4.33e-002)	3.86e-002(4.91e-002)	**1.37e-002(2.34e-002)**
f07	1.54e-035(6.70e-035)	3.08e-075(6.54e-075)	5.72e-077(1.98e-076)	6.55e-075(2.77e-074)	**3.23e-077(7.96e-077)**
f08	0(0)	0(0)	0(0)	0(0)	0(0)
f09	7.44e-020(2.32e-019)	**6.73e-021(1.70e-020)**	9.41e-019(2.57e-018)	3.69e-019(1.36e-018)	9.22e-020(2.24e-019)
f10	0(0)	0(0)	0(0)	0(0)	0(0)
f11	5.95e-015(1.77e-015)	4.35e-015(1.07e-015)	**4.00e-015(0)**	4.00e-015(0)	4.00e-015(0)
f12	1.32e-014(2.95e-014)	**0(0)**	0(0)	0(0)	0(0)
f13	0(0)	0(0)	0(0)	0(0)	0(0)
f14	1.36e-004(2.71e-020)	1.36e-004(2.71e-020)	1.36e-004(2.71e-020)	1.36e-004(2.71e-020)	1.36e-004(2.71e-020)
f15	0(0)	0(0)	0(0)	0(0)	0(0)
f16	5.22e-001(2.15e-001)	**3.98e-001(0)**	3.98e-001(0)	3.98e-001(0)	3.98e-001(0)
f17	9.42e-033(2.74e-048)	9.42e-033(2.74e-048)	9.42e-033(2.74e-048)	9.42e-033(2.74e-048)	9.42e-033(2.74e-048)
f18	1.35e-032(0)	1.35e-032(0)	1.35e-032(0)	1.35e-032(0)	1.35e-032(0)
f19	**1.73e-014(2.18e-014)**	2.69e-014(2.64e-014)	3.32e-014(3.21e-014)	4.65e-014(4.00e-014)	3.84e-014(3.61e-014)
f20	6.79e-002(1.77e-002)	4.13e-002(1.23e-002)	**3.72e-002(6.94e-018)**	3.72e-002(6.94e-018)	3.72e-002(6.94e-018)
f21	1.23e-001(2.30e-002)	**8.08e-002(2.05e-002)**	8.61e-002(2.30e-002)	1.01e-001(3.23e-002)	1.02(4.08e-002)

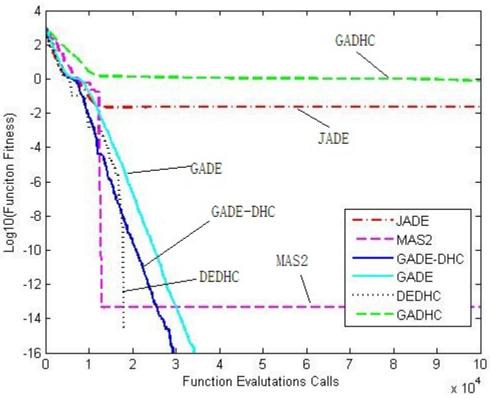

Fig. 3. Search trace for minimizing 50-D Griewank function.

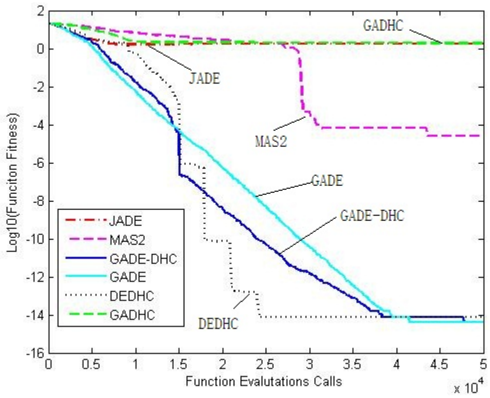

Fig. 4. Search trace for minimizing 50-D Ackley function.

function), and at the aspect of searching speed, it just has a narrow loss in a few functions, but it doesn't lose in precision. And besides, we can also see that DEDHC which is a simple MA consists of JADE and DHC outperforms JADE, this demonstrates that hybridize a global search method with a right local search method can be more efficient than global search alone. Of course, it also proves that the improved DHC is resultful. Furthermore, the performance of GADE-DHC, GADE, DEDHC and GADHC indicate that the three components of our proposed algorithm is essential.

V. CONCLUSIONS

When we design a memetic algorithm to solve some optimization problems, we should often take exploration, exploitation, cooperation and competition into consideration. Therefore, we often encounter some problems, for example: which local search method and global search method are suitable to solve a given problem, when to use local search method to locally improve an individual, and how to balance the ratio between global search method and local search method, etc.

In this paper, a novel memetic algorithm, named GADE-

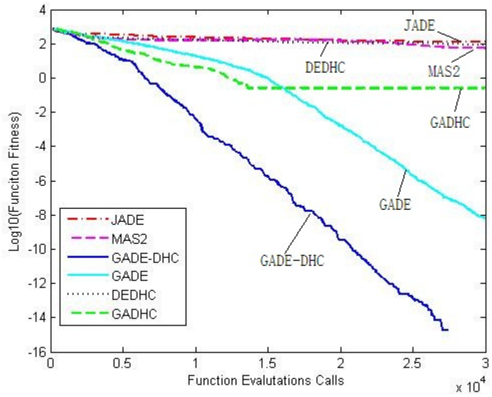

Fig. 5. Search trace for minimizing 50-D Rastrigin function.

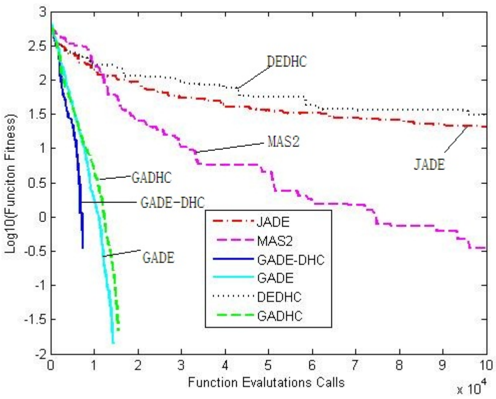

Fig. 6. Search trace for minimizing 50-D Noncontinuous Rastrigin function.

DHC which use GA and DE as global search methods and DHC as local search method, is proposed. In this new memetic algorithm, GA is mainly used to quickly spread good genes in the population, DE is used to create competitive individuals and exploring promising area of solution space and DHC can search the neighborhood area of best individuals directionally which enhance the success rate of searching and reduce the time consumption. The frame based on its previous generation performance is adopted to adjust the usage frequency of these three components. Furthermore, experimental studies on 21 benchmark functions have demonstrated that the new strategy can obtain highly

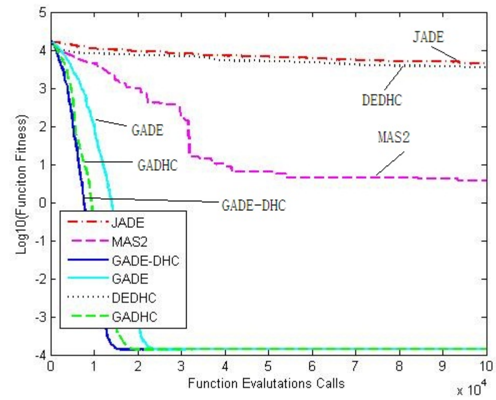

Fig. 7. Search trace for minimizing 50-D Schwefel2.26 function.

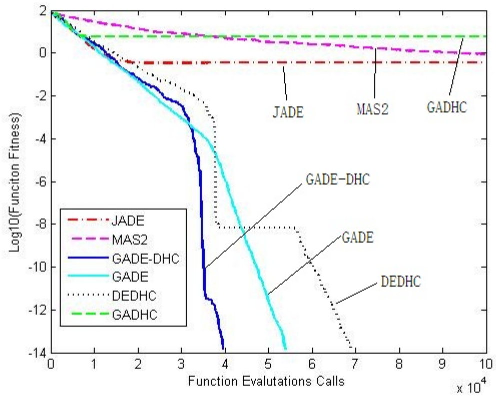

Fig. 8. Search trace for minimizing 50-D Weierstrass function.

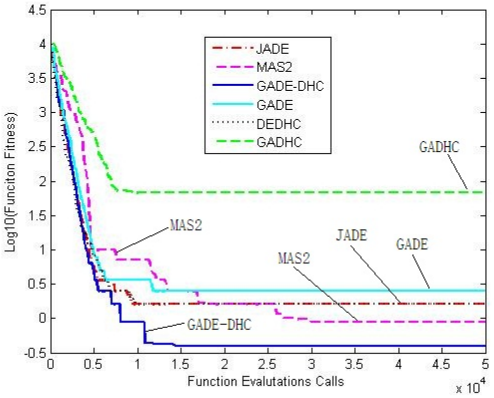

Fig. 9. Search trace for minimizing 50-D Salomon function.

competitive results.

Appendix A
Benchmark Funcitons

1, Sphere function

$$f_{01} = \sum_{i=1}^{D} {x_i}^2 \qquad -100 \leq x_i \leq 100$$

2, Axis parallel hyper-ellipsoid function

$$f_{02} = \sum_{i=1}^{D} i{x_i}^2 \qquad -100 \leq x_i \leq 100$$

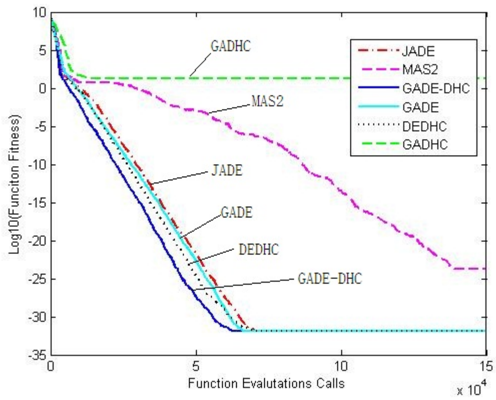

Fig. 10. Search trace for minimizing 50-D Penalized2 function.

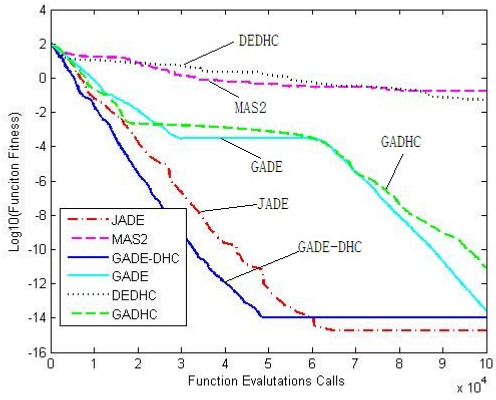

Fig. 11. Search trace for minimizing 50-D Alpine function.

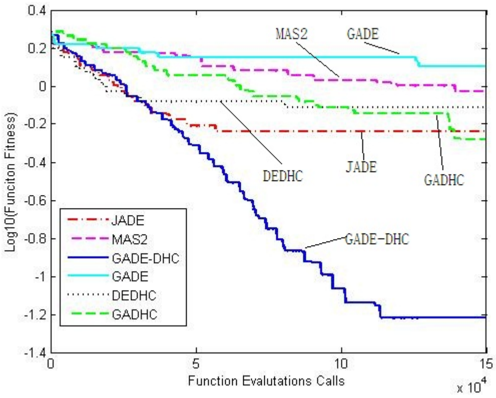

Fig. 12. Search trace for minimizing 50-D Schaffer7 function.

3, Elliptic function

$$f_{03}=\sum_{i=1}^{D}10^{6\frac{i-1}{D-1}}{x_i}^2 \qquad -100\le x_i\le 100$$

4, Schwefel's Problem 1.2

$$f_{04}=\sum_{i=1}^{D}(\sum_{j=1}^{i}x_j)^2 \qquad -100\le x_i\le 100$$

5, Schwefel's Problem 1.2 with noisy

$$f_{05}=(\sum_{i=1}^{D}(\sum_{j=1}^{i}x_j)^2)*(1+0.4|N(0,1)|) \quad -100\le x_i\le 100$$

6, Schwefel's Problem 2.21

$$f_{06}=max\{|x_i|,1\le i\le D\} \qquad -100\le x_i\le 100$$

7, Schwefel's Problem 2.22

$$f_{07}=\sum_{i=1}^{D}|x_i|+\prod_{i=1}^{D}|x_i| \qquad -32\le x_i\le 32$$

8, Step function

$$f_{08}=\sum_{i=1}^{D}\lfloor x_i+0.5\rfloor^2 \qquad -100\le x_i\le 100$$

9, Generalized Rosenbrock's function

$$f_{09}=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2] \quad -100\le x_i\le 100$$

10, Generalized Griewank function

$$f_{10}=\sum_{i=1}^{D}\frac{{x_i}^2}{4000}-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1 \qquad -600\le x_i\le 600$$

11, Ackley's function

$$f_{11}=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}{x_i}^2)} \qquad -32\le x_i\le 32$$

$$-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))+20+e$$

12, Generalized Rastrigin's function

$$f_{12}=\sum_{i=1}^{D}[{x_i}^2-10\cos(2\pi x_i)+10] \qquad -5.12\le x_i\le 5.12$$

13, Noncontinuous Rastrigin function

$$f_{13}=\sum_{i=1}^{D}({y_i}^2-10\cos(2\pi y_i)+10) \qquad -5.12\le x_i\le 5.12$$

$$y_i=\begin{cases} x_i & |x_i|<\frac{1}{2}\\ \frac{round(2x_i)}{2} & |x_i|\ge\frac{1}{2}\end{cases}$$

14, Generalized Schwefel's Problem 2.26

$$f_{14}=418.98289D+\sum_{i=1}^{D}(x_i\sin(\sqrt{|x_i|})) \quad -500\le x_i\le 500$$

15, Weierstrass function

$$f_{15}=\sum_{i=1}^{D}(\sum_{k=0}^{kmax}a^k\cos[2\pi b^k(x_i+0.5)]) \quad -0.5\le x_i\le 0.5$$

$$-D\sum_{k=0}^{kmax}[a^k\cos(2\pi b^k 0.5)]$$

$$a=0.5,b=3,kmax=20$$

16, Salamon function

$$f_{16}=1-\cos(2\pi\sum_{i=1}^{D}x_i)+0.1\sum_{i=1}^{D}x_i \qquad -100\le x_i\le 100$$

17, Penalized 1 function

$$f_{17}=\frac{\pi}{D}\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]$$

$$+(y_D-1)^2\}+\sum_{i=1}^{D}u(x_i,10,100,4) \qquad -50\le x_i\le 50$$

$$y=1+\frac{1}{4}(x_i+1)$$

18, Penalized 2 function

$$f_{18}=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}x_i-1^2[1+\sin^2(3\pi x_{i+1})]$$

$$+(x_D-1)^2\}[1+\sin^2(2\pi x_D)]\}+\sum_{i=1}^{D}u(x_i,5,100,4) \qquad -50\le x_i\le 50$$

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \leq x_i \leq a \\ k(-x_i - a)^m & x_i < a \end{cases}$$

19, Alpine function

$$f_{19} = \sum_{i=1}^{D} |x_i \sin(x_i) + 0.1x_i| \qquad -10 \leq x_i \leq 10$$

20, Schaffer F6 function

$$f_{20} = \frac{\sin^2 \sqrt{\sum_{i=1}^{D} {x_i}^2} - 0.5}{[1 + 0.001 \sum_{i=1}^{D} {x_i}^2]^2} + 0.5 \qquad -100 \leq x_i \leq 100$$

21, Schaffer F7 function

$$f_{21} = (\sum_{i=1}^{D} {x_i}^2)^{0.25} [\sin^2(50(\sum_{i=1}^{D} {x_i}^{0.1})) + 1.0] \qquad -100 \leq x_i \leq 100$$

ACKNOWLEDGMENT

The authors would like to thank Prof Ong for providing his source code of MAS2 as well as those who give them constructive suggestions on writing this paper.

REFERENCES

[1] Natalio Krasnogor and Jim Smith, ”A Tutorial for Competent Memetic Algorithms:Model, Taxonomy, and Design Issues” *IEEE Transactions On Evolutionary Computing* , Vol.9, NO.5, October 2005.
[2] DeJone K A. Genetic Algorithms: A 25 Year Perspective, In: Computational Intelligence Imitating Life. Eds. J M Zurada et al, New York: *IEEE Press*, 1994.83(11):1588-1602
[3] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, 1st ed. New York: Springer-Verlag, Dec. 2005.
[4] W. E. Hart, ”Adaptive Global Optimization With Local Search” Ph.D. dissertation, Univ. California, San Diego, CA, May 1994.
[5] H. Bersini and B. Renders, ”Hybridizing genetic algorithms with hill-climbing methods for global optimization: two possible ways,” in Proc.*IEEE Int. Symp. Evolutionary Computation*, Orlando, FL, 1994, pp.312-317.
[6] C. Houck, J. Joines, and M. Kay, ”Utilizing Lamarckian Evolution and the Baldwin Effect in Hybrid Genetic Algorithms”, Meta-Heuristic Res. Appl. Group, Dept. Ind. Eng., North Carolina State Univ., NCSU-IE Tech. Rep. 96-101, 1996.
[7] P. Merz, ”On the performance of memetic algorithms in combinatorial optimization”, presented at the 2nd Workshop on Memetic Algorithms, *GECCO* 2001, San Francisco, CA.
[8] N. Krasnogor, ”Studies on the Theory and Design Space of Memetic Algorithms,” Ph.D. dissertion, Faculty Comput., Math. Eng., Univ.West of England, Bristol, U.K, 2002.
[9] Y. S. Ong and A. J. Keane, ”A domain knowledge based search advisor for design problem solving environments”, *Eng. Appl. Artif. Intell.*, vol.15, no. 1, pp. 105-116, 2002.
[10] Y. S. Ong and A. J. Keane, ”Meta-Lamarckian Learning in Memetic Algorithms”, *IEEE Transactions On Evolutionary Computing*, vol.8, no.2, pp. 99-110, 2004.
[11] Wilfried Jakob. ”A general cost-benifit-based adaptation framework for Multimeme Algorithm”, Karlsruhe Institute of Technology (KIT), Institute of Applied Computer Science (IAI), P.O. Box 3640, 76021 Karlsruhe, Germany
[12] Wilfried Jakob. ”Towards an Adaptive Multimeme Algorithm for Parameter Optimisation Suiting the Engineers Needs”, T. P. Runarsson et al. (Eds.): Conf. Proc. PPSN IX, LNCS 4193, Springer-Verlag, Berlin, pp.132-141, 2006
[13] N.Krasnogor,Pablo A.Mocciola. ”A Runnable Functional Formal Memetic Algorithm Framework”,Red de Universidades con Carrerasen Informtica (RedUNCI), Oct,1998.
[14] Ernesto Mininno and Ferrante Neri. ”A memetic Differential Evolution approach in nois optimization”,*Memetic Comp.* (2010) 2:111-135 DOI 10.1007/s12293-009-0029-4.
[15] Daniel Molina, Manuel Lozano, Francisco Herrera et al. ”Memetic Algorithms for Continuous Optimisation Based on Local Search Chains”, *Evolutionary Computation* 18(1): 27-63, 2010
[16] Jianyong Sun and Jonathan M. Garibaldi. ”A Novel Memetic Algorithm for Constrained Optimization”, *Evolutionary Computation(CEC)*,2010.
[17] XianShun Chen, LiangFeng and Yew Soon Ong. ”A self-adaptive memeplexes robust search scheme for solving stochastic demands vehicle routing problem”, *International Journal of Systems Science*, DOI:10.1080/00207721.2011.618-646.
[18] Xianshun Chen and Yew Soon Ong. ”A Conceptual Modeling of Meme Complexes in Stochastic Search”, *IEEE Transactions on systems,man and cybernetics-part* C: Applications and Reviews, VOL. 42, NO. 5, Septemb 2012.
[19] Y. S. Ong, M. H. Lim, N. Zhu, and K. W. Wong, ”Classification of adaptive memetic algorithms: A comparative study”,*IEEE Trans. Syst. Man Cybern.* Part B: Cybern. vol. 36, no. 1, pp. 141-152, Feb. 2006.
[20] J. E. Smith et al. , Co-evolution of memetic algorithms: Initial investiga-tions, inParallel Problem Solving From Nature?PPSN VII, G. Guervos et al. , Eds. Berlin, Germany: Springer, 2002, vol. 2439,Lecture Notes in Computer Science, pp. 537-548.
[21] J. E. Smith, Co-evolving memetic algorithms: A learning approach to robust scalable optimization, in IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE Press, 2003, vol. 1, pp. 498-505.
[22] Cai Zhihua, Peng Jin'guo, Gao Wei et al. ”An Improved Evolutionary Algorithmforth Traveling Salesman Problem”. *Chinese Journal of Computers*, Vol. 28 No. 5. pp.823-828.May 2005.
[23] Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau et al. ”A Hybrid Genetic Algorithm for Multidepot and Periodic Vehicle Routing Problems”, *Operations Research* 60(3):611-624. http://dx.doi.org/10.1287/opre.1120.104
[24] Guohui Zhang, Liang Gao, Yang Shi. ”An effective genetic algorithm for the flexible job-shop scheduling problem”. *Expert Systems with Applications* 38 (2011) 3563-3573.
[25] R Storn, K Price, ”Differential evolutionCA simple and efficient heuristic for global optimization over continuous spaces,” *J. of Global Optim.*, vol. 11, no. 4, pp. 341-359, Dec 1997.
[26] K. Price, R. Storn, and J. Lampinen, ”Differential Evolution: A Practical Approach to Global Optimization”. Berlin: Springer-Verlag, 2005.
[27] R. Storn and K. Price, ”Home page of differential evolution,” 2010.[Online]. Available: http://www.ICSI.Berkeley.edu/?storn/code.html
[28] Chuang Li-Yeh, Chiang Yi-Cheng, Yang, Cheng-Hong.”Differential evolution with application to operon prediction”. IAENG International Journal of Computer Science, v 40, n 3, p 200-206, 2013.
[29] Jingqiao Zhang and Arthur C. Sanderson. ”JADE: Adaptive Differential Evolution with Optional External Archive”. *IEEE Transactions on evolutionary computation*, VOL. 13, NO. 5, pp.945-957 OCTOBER 2009

Jingfeng Yan(1982 -) , School of Information Engineering, Xuchang University, Xuchang, Henan, 461000, China, PR china. 2004-2007: M.S.in School of Information Engineering at China University of Geosciences, Wuhan, China, Thesis title: Differential evolution algorithm and its application research. 2001-2004: B.S. in School of Information Engineering at Hubei university of technology, whuan, China.

He is now the Associate Professor of Xu Chang University, working in the field of Computer Science and algorithm. His present interests lie in: Differential Evolution Algorithm, Artificial colony algorithm, Memetic Algorithm. And he has presented some papers in Journal of Software, Journal of Information And Computer Science.

Prof. Yan has gotten Sigma Xi graduate award, and China University of Geosciences (2007),The national college students challenge cup outstanding instructor(2013).