MOEA/D-AMS: Improving MOEA/D by an Adaptive Mating Selection Mechanism

Tsung-Che Chiang
Department of Computer Science and Information Engineering, National Taiwan Normal University
Taipei, Taiwan, R.O.C.
tcchiang@ieee.org

Yung-Pin Lai
Department of Computer Science and Information Engineering, National Taiwan Normal University
Taipei, Taiwan, R.O.C.
697470547@ntnu.edu.tw

Abstract*—In this paper we propose a multiobjective evolutionary algorithm based on MOEA/D [1] for solving multiobjective optimization problems. MOEA/D decomposes a multiobjective optimization problem into many single-objective subproblems. The objective of each subproblem is a weighted aggregation of the original objectives. Using evenly distributed weight vectors on subproblems, solutions to subproblems form a set of well-spread approximated Pareto optimal solutions to the original problem. In MOEA/D, each individual in the population represents the current best solution to one subproblem. Mating selection is carried out in a uniform and static manner. Each individual/subproblem is selected/solved once at each generation, and the mating pool of each individual is determined and fixed based on the distance between weight vectors on the objective space. We propose an adaptive mating selection mechanism for MOEA/D. It classifies subproblems into solved ones and unsolved ones and selects only individuals of unsolved subproblems. Besides, it dynamically adjusts the mating pools of individuals according to their distance on the decision space. The proposed algorithm, MOEA/D-AMS, is compared with two versions of MOEA/D using nine continuous functions. The experimental results confirm the benefits of the adaptive mating selection mechanism.

Keywords-multiobjective optimization; evolutionary algorithm; mating selection; mating pool; subproblem; scalarization

I. INTRODUCTION

In our daily life, it is common for us to solve problems considering multiple objectives including time, cost, quantity, quality, etc. When we can define an aggregated value or a preference order for these objectives, the single-objective optimization algorithms can find the optimal solution for us. However, when we are not sure about the relationship or trade-off between the objectives, we need an approach to provide the best candidate solutions and let us to choose the favorite one. Regarding m objective functions f_i (i = 1, …, m) and two solutions x and y, we say $F(x) = [f_1(x), \ldots, f_m(x)]$ *dominates* $F(y) = [f_1(y), \ldots, f_m(y)]$ iff $f_i(x) \leq f_i(y)$ for all $i = 1, \ldots, m$ and $F(x) \neq F(y)$. Based on the definition of dominance, the best candidate solutions to a multiobjective optimization problem are those whose objective vectors are non-dominated by any other's objective vector. We call these solutions Pareto optimal, and they form the Pareto optimal set. The set of objective vectors of all Pareto optimal solutions forms the Pareto front. The goal of a multiobjective optimization approach is to generate or approximate the Pareto optimal set or Pareto front.

In the last decade, evolutionary algorithms have become one of the most popular approaches for solving multiobjective optimization problems. Evolutionary Multiobjective Optimization (EMO) has also been judged as one of the three fastest growing fields of research and application among all computational intelligence topics in the World Congress on Computational Intelligence (WCCI) in 2006. Many well-known multiobjective evolutionary algorithms (MOEA) like NSGA-II [2], SPEA2 [3], and PESA-II [4] were proposed and had been applied in various kinds of applications like shop scheduling [5], vehicle routing [6], data mining [7], and so on.

MOEA/D [1] is a recent breakthrough of design of MOEA. Its main ideas include problem decomposition, weighted aggregation of objectives, dedicated weight vectors to subproblems, and mating restriction. Mating selection in MOEA/D is carried out in a uniform and static manner. It is uniform since each individual/subproblem is selected/solved by the same times; it is static since the mating pool of each individual is determined before the evolution and fixed during the evolution. We observed the progress of MOEA/D on solving some problem instances and noticed that uniform mating selection could waste the computational effort and the mating pool determined based on the distance on the objective space is not always proper. In this paper we propose an adaptive mating selection mechanism for MOEA/D. It classifies subproblems into solved ones and unsolved ones and spends computational effort only on unsolved problems. Besides, it adjusts the mating pool of each individual dynamically according to the distance between individuals on the decision space.

The rest of this paper is organized as follows. Section II provides a review of the literature on MOEAs. Section III presents MOEA/D, MOEA/D-DE, and two proposed mechanisms to achieve adaptive mating selection. Section IV describes the experiments and shows the results. We compare the proposed MOEA/D-AMS with two versions of MOEA/D, MOEA/D-DE [8] and MOEA/D-DRA [9] using nine functions proposed in [8] in terms of the inverted generational distance (IGD). Conclusions and future work are given in Section V.

II. LITERATURE REVIEW

In the design of MOEAs, the most critical part should be fitness assignment or the way to compare the quality of individuals in terms of multiple objectives. We roughly classify the common mechanisms into three categories: dominance-based, performance metric-based, and weighted aggregation-based. Certainly, this is not the only way of classification.

Niched Pareto Genetic Algorithm (NPGA) [10] is one of the earliest dominance-based MOEAs. To determine if an individual is better than the other, NPGA compares these two individuals with a reference set of randomly picked individuals. If one candidate is non-dominated by all reference individuals and the other is dominated by at least one reference individual, the non-dominated one is said to be better. If they are both non-dominated or both dominated, a niche count is used to determine the winner. Non-dominated Sorting Genetic Algorithm (NSGA) [11] is another early dominance-based MOEA. It classifies individuals into several ranks by repeatedly identifying non-dominated individuals among the unclassified individuals. Individuals who are identified as non-dominated earlier have higher fitness. Individuals with the same rank are compared by the niche count. Strength Pareto Evolutionary Algorithm (SPEA) [12] uses the dominance relationship in a more complicated way. It stores all non-dominated individuals found during the evolutionary process in an external archive. A strength value is assigned to each archive member, and the value is proportional to the number of population members by which the archive member dominates. Then, the fitness of each population member is one plus the sum of strength values of the archive members who dominate the population member. NSGA-II [2] improves its previous version by a faster non-dominated sorting procedure and the crowding distance as a diversity measure. Omni-optimizer [13] is a further improvement on NSGA-II. It proposes a modified dominance relationship and calculates the crowding distance on both decision and objective spaces.

Evaluating the approximated Pareto front obtained by MOEAs is itself a multiobjective problem. We want the obtained solutions to be close to and evenly distributed on the true Pareto front. Defining a performance metric to measure the proximity and distribution simultaneously is not a trivial task. IGD [1], hypervolume [14], ε-indicator [15], and many metrics were proposed. Since the performance of MOEAs is evaluated by these metrics, taking the metric into consideration in fitness assignment seems a reasonable idea. Indicator-Based Evolutionary Algorithm (IBEA) [16] takes this idea and assigns fitness based on the binary ε-indicator and hypervolume. The basic idea is to measure the performance loss caused by the removal of the target individual. The more the loss, the more important the individual is. SMS-EMOA [17] is another metric-based MOEA. It considers the hypervolume metric. Non-dominated individuals are evaluated based on their contribution to the hypervolume. Since dominated individuals contribute no hypervolume, they are evaluated by the number of dominating individuals. Set Preference Algorithm for Multiobjective optimization (SPAM) [18] is an extension of IBEA. The main idea is to take the entire population as the unit of evaluation. It realizes a (1+1)-strategy on the space of Pareto set approximation.

The third category of fitness assignment is the weighted aggregation-based method. It aggregates multiple objective values as a single objective value through functions like linear weighted sum or weighted Tchebycheff function. (Note that not every Pareto optimal solution can be found by taking linear weighted sum as the aggregation function.) Given a set of weights on objectives, the MOEA searches for a single optimal solution with respect to the aggregated objective. To approximate the Pareto front, MOEAs in this category have to use multiple sets of weight vectors. Besides, these weight vectors should be evenly distributed on the objective space to obtain well-spread solutions on the Pareto front.

The multiobjective genetic local search (I-MOGLS) [19] by Ishibuchi and Murata is among the first weighted aggregation-based MOEAs. It aggregates multiple objectives by linear weighted sum and takes a random weight vector at each selection of a pair of parents in order to search toward different regions on the objective space. The weight vector used in selection of parents is recorded on the generated offspring. It is used later in the local search procedure to search along the same direction. In its extended version [20], the weight vector used in local search is generated randomly and the individual to start the search along this direction is selected by 5-tournament. Jaszkiewicz also proposed a multiobjective genetic local search (J-MOGLS) in [21]. Like I-MOGLS, J-MOGLS uses random weight vectors at each selection of parents. The difference is that I-MOGLS selects parents by roulette wheel selection but J-MOGLS selects parents randomly from the *K* best individuals in terms of the aggregated objective. The weight vector used in local search is the same as that used in parent selection. Chang *et al.* [22] proposed the gradual-priority weighting (GPW) MOEA to solve the multiobjective flowshop scheduling problem. Unlike I-MOGLS and J-MOGLS, only one weight vector is used in a generation. The weight vector changes periodically. In Sub-Population Genetic Algorithm II (SPGA-II) [23], the population is divided into sub-populations, and each sub-population is assigned a unique weight vector. A global archive stores the non-dominated solutions found by all sub-populations, and the archive members join the evolution of all sub-problems. MOEA/D [1], MOEA/D-DE [8], and MOEA/D-DRA [9] are a family of MOEAs proposed by the same authors. In these algorithms, each individual is assigned a unique and fixed weight vector. Distance between individuals is measured by the Euclidean distance between the associated weight vectors. Only individuals close to each other are allowed to mate. The algorithm proposed in this study is based on MOEA/D-DE, and we leave their details in next section.

In addition to fitness assignment, archive maintenance is another interesting topic. Laumanns *et al.* [24] proposed the concept of ε-dominance to avoid the deterioration of archive in many MOEAs. This concept was utilized by Deb *et al.* in [25]. The value of ε determines the maximum archive size and has great influence on the collection of archive members. In order to determine a proper value, Hernádez-Díaz *et al.* [26] proposed the Pareto-adaptive ε-dominance method. The model of the curve of Pareto front is approximated, and different ε values are allowed for different objectives. This method was further improved by Gong and Cai [27]. For more details on EMO, readers are suggested to the following books: [28]–[30].

III. PROPOSED ALGORITHM

The proposed algorithm is based on MOEA/D-DE, an improved version of MOEA/D. Firstly we have a brief review of MOEA/D and MOEA/D-DE Then, two proposed mechanisms for adaptive mating selection are presented.

A. Review of MOEA/D and MOEA/D-DE

The most important idea of MOEA/D is to decompose a multiobjective optimization problem into multiple single-objective optimization subproblems, solve each subproblem, and finally collect the solution to each subproblem to be the approximated Pareto optimal solution set to the original problem. The objective of each subproblem is an aggregation of the concerned objectives through some functions like linear weighted sum, weighted Tchebycheff function, and penalty boundary intersection in [1]. The difference in the aggregated objectives of the subproblems is the weights on the objectives. By varying the weights and distributing the weight vectors λ evenly on the objective space, the solutions to the subproblems are expected to be not only optimal (or near-optimal) in terms of the aggregated objective but also evenly distributed on the objective space, meeting the goal of approximating the Pareto optimal set. Figure 1 is an illustration of the main idea.

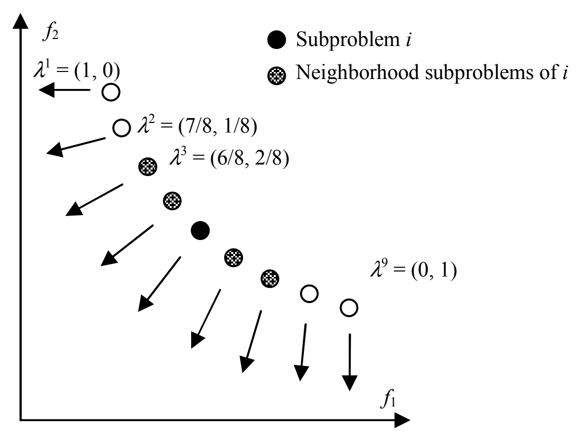


Figure 1. An illustration of the main idea of MOEA/D

Another main idea of MOEA/D is the neighborhood of subproblem. As mentioned, each subproblem i is associated with a weight vector λ^i. The distance between two subproblems is defined by the Euclidean distance between the associated weight vectors. The neighborhood $B(i)$ of a subproblem i is the set of T closest subproblems. During the evolutionary process, each subproblem i maintains a current best individual x^i. In the reproduction process, only individuals x^j of the subproblems in the neighborhood ($j \in B(i)$) are allowed to mate with the individual x^i. This is a realization of mating restriction. The rationale (or assumption) behind the idea is that mating an individual with those with similar performance is more helpful to improve the performance than mating it with those with very different performance. (For example, mating a rat which runs fast with another which also runs fast is likely to generate an offspring which runs even faster. However, mating a rat which runs fast with another which jumps high but runs slow is unlikely to improve the running speed.) Similar ideas can be seen in the algorithms in [19] and [21].

MOEA/D-DE is an improved version of MOEA/D, proposed by the same authors. There are three modifications: (1) as the name indicates, it replaces the SBX crossover operator in MOEA/D by the differential evolution (DE) operator; (2) it allows an individual to mate with any individual in the entire population with a probability $(1 - \delta)$; (3) it sets a maximum number of replacement n_r to control the population diversity and convergence speed.

Our algorithm is based on MOEA/D-DE, and the aggregation objective function is the weighted Tchebycheff function:

$$g^{te}(x \mid \lambda, z^*) = \max_{1 \le i \le m} \{\lambda_i \mid f_i(x) - z_i^* \mid\}, \qquad (1)$$

where z^* is the reference point with the i^{th} element as the minimal value ever found for the i^{th} objective (assuming minimization problem). The algorithm flow is given as follows, and more details can be found in [1] and [8].

Step 1 Initialization:

Step 1.1 Generate weight vectors of subproblems and define neighborhoods. The size of neighborhood is T.

Step 1.2 Generate the initial population and update the reference point z^*.

Step 2 Update: Do the following steps for each individual x^i of each subproblem i once.

Step 2.1 The mating pool P is the neighborhood $B(i)$ in probability δ and the entire population in probability $(1 - \delta)$.

Step 2.2 Select two individuals from P randomly. Generate a new offspring y by the DE operator, and then apply the polynomial mutation in probability p_m.

Step 2.3 Update the reference point.

Step 2.4 Check if the offspring y is better than the individuals in P in a random order. If y is better than the original individual x^j (*i.e.* $g^{te}(y|\lambda^j, z^*) < g^{te}(x^j|\lambda^j, z^*)$), replace x^j with y. At most n^r individuals can be replaced.

Step 3 Stopping criterion: If the stopping criterion is satisfied, stop; otherwise, go to Step 2.

B. Controlled Subproblem Selection (CSS)

In MOEA/D-DE, even distribution of weight vectors of subproblems on the objective space is helpful to obtain a set of solutions evenly distributed on the objective space. However, even distribution of computational effort (note. Step 2 in MOEA/D-DE) to the subproblems might not always be good since the difficulty of the subproblems could be different. Figure 2 shows the reference front and the population at generation 250 in a single run for solving a problem instance LZ2 proposed in [8]. We observed that some subproblems are almost solved optimally but others are still far away from their corresponding optimal solutions. In this condition, we should spend more computational effort on those unsolved subproblems.

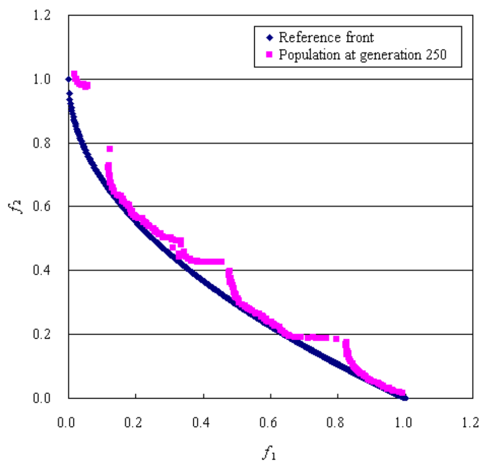


Figure 2. Reference Pareto front and the population at generation 250 in a single run for solving problem instance LZ2

Our first modification to MOEA/D-DE is at Step 2. We do not perform Steps 2.1-2.4 for each individual once at each generation. Instead, we perform these steps only for individuals whose corresponding subproblems are still unsolved. We regard a subproblem as *solved* if its solution is not improved (*i.e.* not replaced in Step 2.4) for α consecutive generations. We disable the solved subproblems and their individuals in the reproduction process. Same as MOEA/D-DE, Step 2 is executed for N times in each generation, where N is the population size. However, if a subproblem is disabled, it is skipped. In other words, given N as 100 and 20 enabled subproblems, each enabled subproblem is solved 100/20 = 5 times in a generation.

Sometimes improvement of the solution to a subproblem stops temporarily but continues later. It indicates that the subproblem is not solved yet, and we should resume the allocation of computational effort to it. Thus, we enable a disabled subproblem again if its solution is improved by the offspring generated by other subproblems. Besides, the β individuals with the largest crowding distance [2] are always enabled. The intention is to keep searching in the unexplored regions. When a hard problem is tackled, search processes for all subproblems could get stuck. In this case all subproblems are disabled, and the algorithm cannot proceed. Keeping β most-uncrowded individuals enabled can also deal with this condition. Experiments on the values of α and β are provided in Section IV-B.

C. Mating Pool Adjustment (MPA)

MOEA/D-DE realizes mating restriction by mating individuals who are close on the objective space. In our observations, however, individuals close on the objective space might be far way from one another on the decision space. In the late stage of an evolutionary process, solutions are usually close to the optimal ones and need only small change of the gene values. If two individuals who are distant from each other on the decision space are selected, the difference vector in the DE operator is large and hence the generated offspring is far away from the original individual. It is difficult to do a local fine tuning.

In our algorithm, we keep the idea of mating restriction but the individuals with which are allowed to mate are selected according to the Euclidean distance between individuals on the decision space instead of the distance between weight vectors of their subproblems on the objective space. The T closest individuals are selected into the mating pool. Starting to mate individuals with those who are close on the decision space too early might cause premature convergence. Thus, we start the adjustment of mating pool after γGen generations, where Gen is the maximum generation number. The adjustment of mating pool requires the calculation of distance between all individuals on the decision space and consumes a large amount of computation time. (Time complexity is $O(N^2 \cdot n)$, where N is the population size and n is the number of decision variables.) To save time, we do the adjustment every ε generations. Experiments on the values of γ and ε are given in Section IV-B.

IV. EXPERIMENTS AND RESULTS

A. Benchmark Instances, Benchmark Algorithms, and Performance Metric

The nine problem instances proposed in [8] are used in the experiments. They are challenging instances with complicated shapes of Pareto sets on the decision space. We denote these instances by LZ1-9 hereafter. All of them are 2-objective instances except LZ6, which is a 3-objective instance. The number of decision variables (n) is 30 in LZ1-5 and LZ9, and 10 in LZ6-8.

The two improved versions of MOEA/D, MOEA/D-DE and MOEA/D-DRA[1] [9], are considered as the benchmark algorithms. The main idea of MOEA/D-DRA and our idea of CSS are similar, but the realization methods are different. In MOEA/D-DRA, a π^i value represents the progress of solving subproblem i. A high π value means that it is worth more computational effort on the subproblem. π^i is defined by

$$\pi^i = \begin{cases} 1 & \text{if } \Delta^i > 0.001 \\ (0.95 + 0.05\dfrac{\Delta^i}{0.001})\pi^i & \text{otherwise} \end{cases},$$

where $\Delta = \dfrac{\text{old function value} - \text{new function value}}{\text{old function value}}$. (2)

The values of Δ and π are updated every 50 generations. In each generation, only the m individuals with the best objective value and the ($N/5 - m$) individuals selected by 10-tournament in terms of π value are allowed to do reproduction. (Recall that m is the number of objectives and N is the population size.)

We use the inverted generational distance (IGD) as the performance metric. It is defined by

$$\mathrm{IGD}(P^*, P) = \frac{\sum_{v \in P^*} d(v, P)}{|P^*|}, \quad (3)$$

where P^* denotes the reference front, P denotes the approximated front, and $d(v, P)$ is the minimum Euclidean

[1] MOEA/D-DRA was the winner of the unconstrained MOEA competition in IEEE Congress on Evolutionary Computation (CEC) 2009.

distance between v and the points in P. Same as the experiments in [8], 500 and 990 evenly distributed points in the Pareto front are taken as the reference front for 2-objective and 3-objective instances, respectively.

B. Parameter Setting

The two proposed mechanisms, CSS and MPA, require four additional parameters. In the first mechanism, each individual is disabled after α non-improving generations, but the β individuals with the largest crowding distance are always enabled. In the second mechanism, the adjustment is activated after γGen generations (Gen: maximum allowed generations) and the mating pool is adjusted every ε generations.

In the first experiment, we examined the performance of our algorithm with different parameter settings. The tested values are listed in Table I. There were 13 setting for each mechanism, and hence 169 (13×13) variants were tested. Values of other parameters were set identically as the experiment in [8]. The population size was 300 and 595 for 2-objective and 3-objective instances, respectively. Values of remaining parameters are listed in Table II. Each algorithm variant solved each instance for 30 times.

TABLE I. TESTED VALUES OF PARAMETERS

Parameters & Values	Levels
Controlled Subproblem Selection:	
Not used,	13 = 1 +
α = {5, 20, 50} × β = {0%, 2%, 5%, 10%}	3 × 4
Mating Pool Adjustment:	
Not used,	13 = 1 +
γ = {25%, 50%, 75%} × ε = {1, 10, 20, 50}	3 × 4

TABLE II. VALUES OF OTHER PARAMETERS

Parameter	Value	Parameter	Value
Gen	500	p_m	$1/n$
CR	1.0	T	20
F	0.5	δ	0.9
η	20	n_r	2

n: number of decision variables

We calculated the average IGD over 30 runs for each pair of algorithm variant and problem instance. Then the average IGD over all nine instances was calculated for each algorithm variant. The average IGD values of the best variant among the 13 variants using only MPA, the best variant among the 13 variants using only CSS, and the best two variants among all 169 variants are listed in Table III. The difference in average IGD of the first three variants indicates that both proposed mechanisms are beneficial to improving the solution quality. Besides, the effect of CSS is larger than that of MPA. We list the second best variant (MOEA/D-AMS-2) since its IGD is close to the best one and it requires much less computation time due to much less frequent adjustment of mating pool. It will be taken to compare with the two benchmark algorithms in next section.

We also analyzed the effect of each parameter by calculating the average IGD over the algorithm variants with the same value of a single parameter. The results are shown in Table IV. For example, the value 0.005873 means the average IGD of 52 (=1×4×(1+3×4)) variants with α equal to 5. The results show that the effect of α is the most significant, the effect of γ and ε is moderate, and the effect of β is little. The bad average performance of variants without CSS (0.008591) and without MPA (0.007546) again confirms the benefits of the proposed two mechanisms.

TABLE III. PERFORMANCE COMPARISON BETWEEN FOUR BEST-PERFORMING VARIANTS

Variants	Average IGD
Best with only MPA	0.007586
Best with only CSS	0.006099
Best with CSS+MPA (MOEA/D-AMS-1: α= 5, β= 0.05, γ= 0.25, ε= 1)	0.005291
Second best with CSS+MPA (MOEA/D-AMS-2: α= 5, β= 0.05, γ= 0.25, ε= 50)	0.005342

TABLE IV. PERFORMANCE COMPARISON BETWEEN DIFFERENT PARAMETER VALUES

Parameter	Tested values and average IGD			
α	No CSS 0.008591	5 0.005873	20 0.006771	50 0.008019
β	0% 0.006838	2% 0.006776	5% 0.006940	10% 0.006996
γ	No MPA 0.007546	25% 0.006851	50% 0.006916	75% 0.007157
ε	1 0.007119	10 0.006933	20 0.006971	50 0.006876

TABLE V. PERFORMANCE COMPARISON BETWEEN THE PROPOSED ALGORITHM AND TWO MOEA/D-BASED ALGORITHMS (AVERAGE IGD)

Problems	MOEA/D-DE	MOEA/D-DRA	MOEA/D-AMS-2
LZ1	0.001347	0.001536	**0.001328**
LZ2	0.002999	0.003685	**0.002351**
LZ3	0.012385	**0.003244**	0.003317
LZ4	0.004320	0.004092	**0.003936**
LZ5	0.010970	0.008814	**0.005847**
LZ6	0.028816	0.028705	**0.027186**
LZ7	0.003984	0.001939	**0.001340**
LZ8	0.066443	0.004343	**0.001431**
LZ9	0.004521	0.001851	**0.001346**
Average	0.015087	0.006468	**0.005342**

C. Performance Comparison

In this subsection we compare the performance of the proposed algorithm with that of MOEA/D-DE and MOEA/D-DRA. The approximated fronts of these two benchmark algorithms were generated by running the programs downloaded from the author's website[2]. The average IGD values are summarized in Table V. Our algorithm (MOEA/D-AMS-2) outperforms two benchmark algorithms for all but one instance. It improves the total average IGD of MOEA/D-DE and MOEA/D-DRA by about 65% and 17%, respectively. It is slightly worse than MOEA/D-DRA on LZ3 due to a single very bad run whose IGD value is 0.022187. Boxplots of the

[2] http://dces.essex.ac.uk/staff/qzhang/mypublication.htm

IGD of three tested algorithms over 30 runs on solving nine problem instances are shown in Figure 3. Results of Wilcoxon rank sum test show that there are significant differences between the IGD of the two benchmarks and our algorithm for all problem instances with significance level of 0.01.

Since the idea of our CSS is similar to the main idea of MOEA/D-DRA, we want to compare their performance. Referring to Table III, the average IGD of the best variant using only CSS is 0.006099, which is still better than that of MOEA/D-DRA (0.006468). Wilcoxon rank sum test shows significant differences for all instances except LZ6. A potential reason for the superior performance could be that our algorithm selects the subproblems in a more restricted way. If a subproblem is disabled, it will not be selected until its corresponding solution is replaced/improved. If most subproblems are disabled, the computational effort is concentrated on a small number of unsolved subproblems. By contrast, almost each subproblem has the possibility of being selected in MOEA/D-DRA. Besides, the selection probability is updated every 50 generations, instead of every generation in our MOEA/D-AMS. It is possible that we can tune the parameters of MOEA/D-DRA to make its selection more restricted and obtain better performance. This is left as our future work, and we also plan to compare their selection results visually.

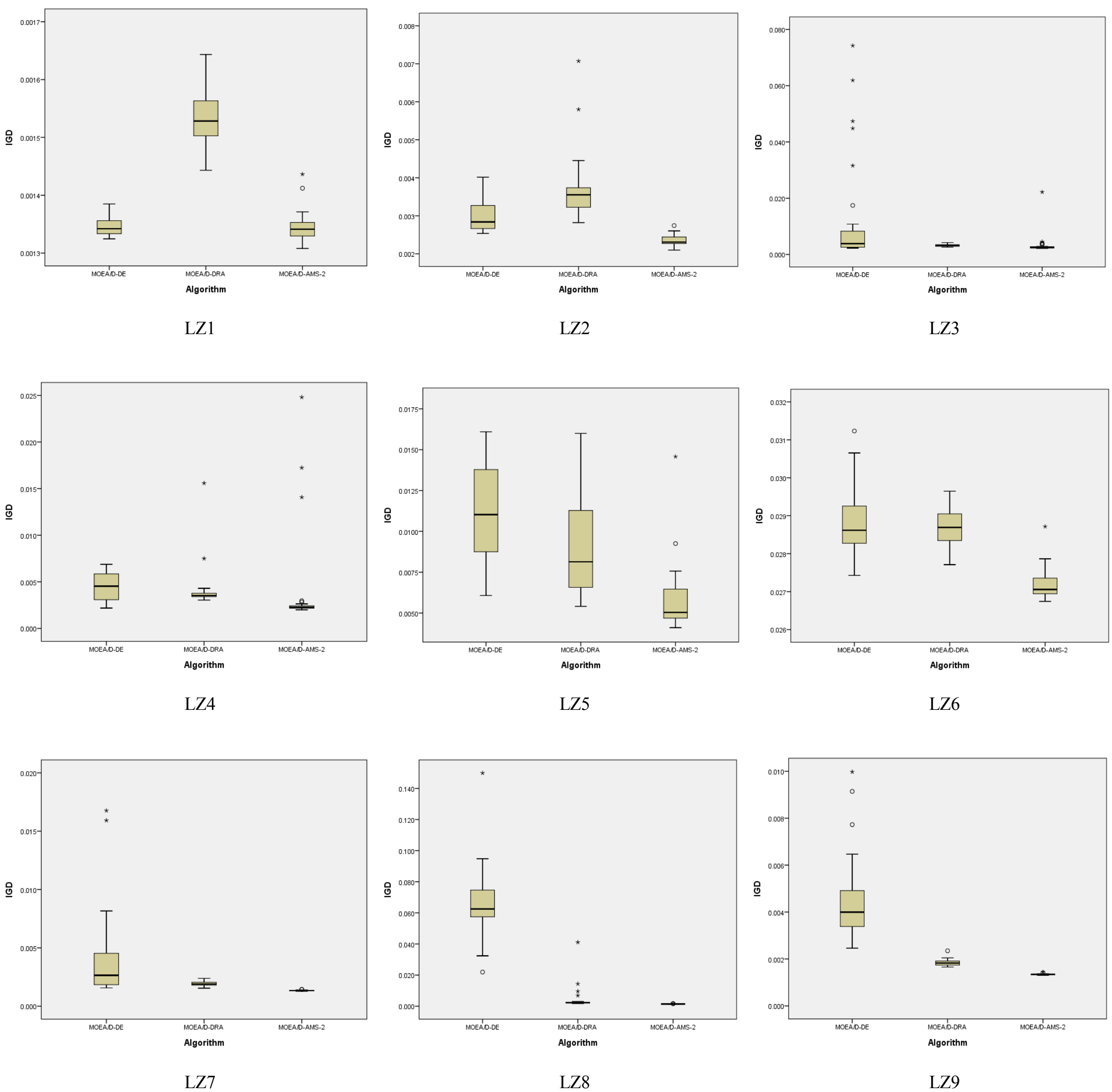
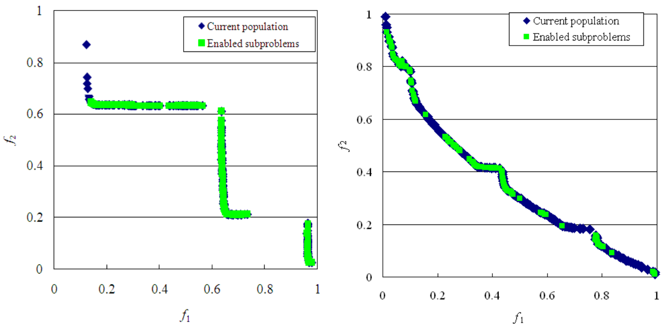


Figure 3. Boxplots of IGD of three tested algorithms over 30 runs on solving nine problem instances (LZ1-LZ9)



(a) generation 100 (b) generation 300

Figure 4. Population and enabled subproblems at generation 100 and 300 in a single run for solving problem instance LZ2

D. *Effect of Controlled Subproblem Selection*

Here we show the effect of CSS visually. We plot the individuals in a run of solving LZ2 on the objective space. Figure 4 shows the entire population and enabled subproblems at generation 100 and 300. At generation 100, most subproblems are not solved well and are enabled by the CSS mechanism. After 200 more generations, many subproblems are solved (close to the Pareto front) and disabled. The computational effort is now concentrated on the remaining unsolved subproblems, accelerating the process to solve these subproblems.

E. *Effect of Mating Pool Adjustment*

From Table III and Table IV, we have already confirmed that the MPA is effective in improving solution quality. Figure 5 illustrates the result of the MPA mechanism. It shows the final population at the last generation in a single run for solving instance LZ2. The original neighborhood and the adjusted mating pool of the subproblem with weight (0, 1) are also shown. The individual x of this subproblem is the one with the smallest f_2 value. The individuals in the original neighborhood are located contiguously at the bottom-right corner. However, these individuals are not close to x on the decision space. Those who are close to x on the decision space are far way from x on the objective space. They are located in three regions, which are distant from one another.

We also noticed that frequent adjustment of mating pool is particularly useful for instances LZ3 and LZ4. By changing ε from 50 to 1, the average IGD values are reduced from 0.003317 to 0.003067 and 0.003936 to 0.002450, respectively. Figure 6 shows the final population from 30 runs of the best two variants of MOEA/D-AMS for solving LZ4. MOEA/D-AMS-1 solves the subproblems near the bottom-right corner much better than MOEA/D-AMS-2 does. We checked the adjusted mating pools of all subproblems at the last generation. The mating pools of most subproblems are identical to the original neighborhoods. However, the adjusted mating pools of the subproblems at the bottom-right corner are very different from the original neighborhoods. The superior performance of MOEA/D-AMS-1 indicates that solving these subproblems requires repeatedly mating their individuals with those who are close on the decision space (rather than on the objective space).

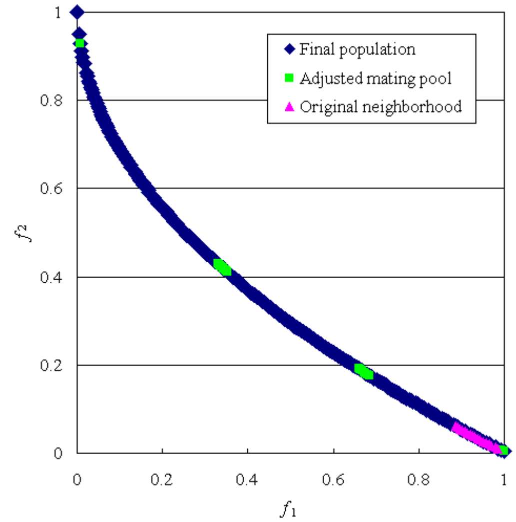
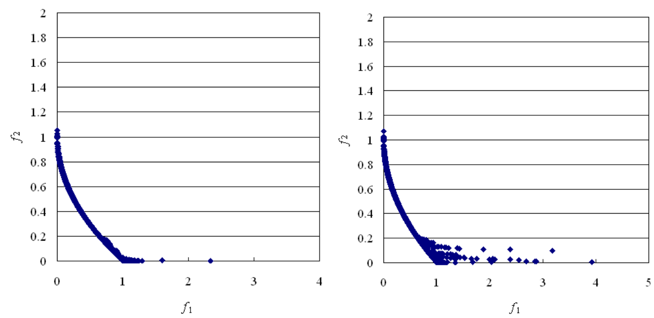


Figure 5. Population and the adjusted mating pool of the subproblem with weight (0, 1) at the last generation in a single run for solving problem instance LZ2



(a) MOEA/D-AMS-1 (b) MOEA/D-AMS-2

Figure 6. Final population from 30 runs of MOEA/D-AMS-1 (ε = 1) and MOEA/D-AMS-2 (ε = 50) for solving problem instance LZ4

V. CONCLUSION AND FUTURE WORK

We proposed an improved version of MOEA/D by introducing an adaptive mating selection mechanism. It consists of controlled subproblem selection and mating pool adjustment. The CSS saves computational effort of solved subproblems and assigns it to unsolved subproblems so that the computational effort is utilized effectively. The MPA mates individuals with those who are close on the decision space so that small change of gene values can be achieved, which is required at the late stage of evolutionary process. Comparing with two versions of MOEA/D, the proposed MOEA/D-AMS provides better solution quality in terms of IGD. The improvement percentage of overall average IGD is at least 17%.

The idea of CSS is similar to the idea of MOEA/D-DRA. CSS is discrete and deterministic, whereas MOEA/D-DRA is continuous and stochastic. CSS counts the times of improvement, but MOEA/D-DRA accumulates the amount of improvement. In the future, we want to compare them by more experiments and observations. MOEA/D has many parameters, and our version introduces four more. Another job to be done is to reduce the number of parameters or set the parameter values automatically. We are studying the techniques of parameter control [31] and will use them in our algorithm. We also plan to apply the proposed MOEA/D-AMS to applications like shop scheduling and vehicle routing.

ACKNOWLEDGMENT

The authors want to thank Prof. Qingfu Zhang for the information about MOEA/D-DRA. We also want to thank Mr. Cheng-Nan Chen for collecting the results of MOEA/D-DE and MOEA/D-DRA and making the box plots. This research was supported by the National Science Council of Republic of China under research grant No. NSC98-2221-E-003-012 and by National Taiwan Normal University under research grant No. 98091040.

REFERENCES

[1] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation,* vol. 11, no. 6, pp. 712–731, 2007.

[2] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[3] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization," In: K.C. Giannakoglu et al. (eds.) *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pp. 95–100, 2002.

[4] D.W. Corne, N.R. Jerram, J.D. Knowles, and M.J. Oates, "PESA-II: Region-based seelction in evolutionary multiobjective optimization," In: *Proceedings of the Genetic and Evolutionary Computation Congeference*, pp. 283–290, 2001.

[5] T.C. Chiang, H.C. Cheng, and L.C. Fu, "NNMA: An effective memetic algorithm for solving multiobjective permutation flowshop scheduling problems," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5986–5999, 2011.

[6] K.C. Tan, Y.H. Chew, and L.H. Lee, "A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows," *Computational Optimization and Applications*, vol. 34, pp. 115 – 151, 2006.

[7] Y.H. Chan, T.C. Chiang, and L.C. Fu, "A two-phase evolutionary algorithm for multiobjective mining of classification rules," In: *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 727 – 733, 2010.

[8] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284-302, 2009.

[9] Q. Zhang, W. Liu, and H. Li, "The peformance of a new version of MOEA/D on CEC09 Unconstrained MOP test instances," In: *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 203–208, 2009.

[10] J. Horn and N. Nafpliotis, "Multiobjective optimization using the niched Pareto genetic algorithm," *University Illinois at Urbana-Champain*, *Urbana*, *IL*, *IlliGAL Report 93005*, 1993.

[11] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.

[12] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[13] K. Deb and S. Tiwari, "Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimization," *European Journal of Operational Research*, vol. 185, pp. 1062–1087, 2008.

[14] E. Zitzler and L. Thiele, "Multiobjective optimization using evoluationary algorithms – A comparative case study," In: A.E. Eiben et al. (eds.) *Proceedings of International Conference on Parallel Problem Solving from Nature (PPSN V)*, pp. 292–301, 1998.

[15] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Comptuation*, vol. 7, no. 2, pp. 117–131, 2003.

[16] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," In: *Proceedings of International Confernce on Parallel Problem Solving from Nature (PPSN VIII)*, pp. 832–842, 2004.

[17] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669, 2007.

[18] E. Zitzler, L. Thiele, and J. Bader, "SPAM: Set preference algorithm for multiobjective optimization," In: *Proceedings of International Conference on Parallel Problem Solving from Nature* (*PPSN X*), pp. 847–858, 2008.

[19] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics – Part C*, vol. 28, no. 3, pp. 392–403, 1998.

[20] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 204–223, 2003.

[21] A. Jasckiewicz, "Genetic local search for multi-objective combinatorial optimization," *European Journal of Operational Research*, vol. 137, pp. 50–71, 2002.

[22] P.C. Chang, J.C. Hsieh, and S.G. Lin, "The development of gradual-priority weighting aproach for the multi-objective flowshop scheduling problem," *International Journal of Production Economics*, vol. 79, pp. 171–183, 2002.

[23] P.C. Chang and S.H. Chen, "The development of a sub-population genetic algorithm II (SPGAII) for the multi-objective combinatorial problems," *Applied Soft Computing*, vol. 9, no. 1, pp. 173–181, 2009.

[24] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multi-objective optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.

[25] K. Deb, M. Mohan, and S. Mishra, "Evaluating the ε-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions," *Evolutionary Computation*, vol. 13, no. 4, pp. 501–525, 2005.

[26] A.G. Hernádez-Díaz, L.V. Santana-Quintero, C.A.C. Coello, and J. Moline, "Pareto-adaptive ε-dominace," *Evolutionary Computation*, vol. 15, no. 4, pp. 493–512, 2007.

[27] W. Gong and Z. Cai, "An improved multiobjective differential evolution based Pareto-adaptive ε-domiance and orthogonal design," *European Journal of Operational Research*, vol. 198, pp. 576–601, 2009.

[28] A. Abraham and R. Goldberg, *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, Springer, 2005.

[29] J. Branke, K. Deb, K. Miettinen, and R. Slowinski, *Multiobjective Optimization: Interactive and Evolutionary Approaches*, Springer, 2008.

[30] C.K. Goh, Y.S. Ong, and K.C. Tan, *Multi-objective Memetic Algorithms*, Springer, 2009.

[31] A.E. Eiben, Z. Michaelwicz, M. Schoenauer, and J.E. Smith, "Parameter control in evolutionary algorithms," In: F.G. Lobo, C.F. Lima, and Z. Michaelwicz (eds.) *Parameter Setting in Evolutionary Algorithm*, pp. 19–46, 2007.