The multiobjective evolutionary algorithm based on determined weight and sub-regional search

Hai-lin Liu, Xueqiang Li

***Abstract*— By dividing the multiobjective optimization of the decision space into several small regions, this paper proposes multi-objective optimization algorithm based on sub-regional search, which makes individuals in same region operate each other by evolutionary operator and the information between the individuals of different regions exchange through their offsprings re-divided into regions again. Since the proposed algorithm utilizes the sub-regional search, the computational complexity at each generation is lower than the NSGA-II and MSEA. The proposed algorithm makes use of the max-min strategy with determined weight as fitness functions, which make it approach evenly distributed solution in Pareto front. This paper presents a kind of easy technology dealing with the constraint, which makes the proposed algorithm solved unconstrained multiobjective problems can also be used to solve constrained multiobjective problems. The numerical results, with 13 unconstrained multiobjective optimization testing instances and 10 constrained multiobjective optimization testing instances, are shown in this paper.**

I. Introduction

Multiobjective optimization is a usual problem in the fields of science and technology, and management science, etc. Different from singleobjective optimization, multiobjective optimization has more than one objective, and the objectives are often conflicting. Therefore, multiobjective optimization is to find in some sense nondominated solutions about its vector objective functions, which are often many or infinite. The model of multiobjective optimization problem is as follows:

$$\begin{cases} \min f(x) = \big(f_1(x), f_2(x), \cdots, f_m(x)\big)^{\top} \\ s.t \quad g_i(x) \geq 0 \qquad i = 1, 2, \cdots, k \\ x \in X \subseteq R^n \end{cases} \tag{1}$$

where x is decision variable, $f(x)$ is objective vector, X is super rectangular region in R^n and expresses decision space, $g_i(x)$ is constraint function $(i = 1, 2, \cdots, k)$.

The merits solving multiobjective optimization problem by evolutionary algorithm attract wide attentions of scholars, and there have been many results of research in the past 20 years [1]. Besides design of evolutionary operators, the performance of multiobjective evolutionary algorithms largely depended on how to assign fitness of individuals. The existing fitness assignment strategies mainly include fitness assignment based on nondominated sort [9], domination [7], alternative objectives [6], and aggregating objectives [2]-[5],[8],[12]. In most of algorithms based on aggregating objectives (the weighted sum of objective function and the max-min strategy with weight), the weights are generated randomly [4], [5], [8], [12]. In order to lead evolutionary algorithm better to approach evenly distributed solution in Pareto front, the literatures [2], [3] gave a method choosing weights of fitness function based on max-min strategy by spherical coordinate transformation. The complexity of MSEA algorithm in the literature [3] is $O(mKN) + O(mN \log N)$ (in which m is the number of objective, N is population size, and K is the number of weight vectors). When K is smaller, the algorithm has a faster convergence velocity. However, when K is bigger, the complexity of the algorithm will increase. In order to decrease the complexity of algorithm and improve its convergence performance, we propose an improved algorithm about MSEA. By choosing the appropriate T groups of weight vectors in max-min strategy, the objective space is divided into T regions, and corresponding decision variable space is also divided into T regions. The newborn individuals which come from crossover and mutation between two individuals in the same region are divided into T regions according to some particular rules. The next generation is selected by a group of fitness functions with the uniform designed weight vectors. A new kind of crossover and mutation operators is designed in the improved MSEA algorithm, which largely enhance performance of the algorithm and reduce its complexity. This paper presents a kind of easy technology dealing with the constraintwhich makes the proposed algorithm solved unconstrained multiobjective problem can also be used to solve constrained multiobjective problem. In this paper, 13 unconstrained multiobjective optimization testing instances and 10 constrained multiobjective optimization testing instances are simulated by computer, and the effectiveness of proposed algorithm is investigated.

The remainder of this paper is organized as follows: In section 2, the way of dividing the search area is introduced. Section 3 gives the design of evolutionary operator, section 4 shows the framework and specialty of proposed algorithm. In section 5, the numerical experiments of unconstrained and constrained multiobjective problem are introduced. The conclusion is in section 6.

II. The Division of Searching Region

In order to reduce the computing cost and improve efficiency of the searching, we divide weight vectors and individuals into T classes. In selection step, we firstly select a certain number of new individuals for each class and compare these new individuals with the exist individuals in same class.

Hai-lin Liu is with the Faculty of Applied Mathematics,Guangdong University of Technology, China (email: lhl@scnu.edu.cn). Xueqiang Li is with the Faculty of Applied Mathematics,Guangdong University of Technology, China (email: lxqlsx0001@sina.com.cn)

If the new individual is better than the existed individual, we substitute it. Meanwhile, in order to keep the selection pressure not too high, we define an external set for each class and update the individuals in external sets during the process of selection.

A. Construction of the Fitness function

In this paper, the fitness function utilizes the weighted max-min strategy which is similar to the one in literature [1], [2], the difference is that the weight vectors are calculated as follows:

$$\begin{cases} w_1(x) = \frac{1}{\cos\theta_1} \\ w_2(x) = \frac{1}{\sin\theta_1\cos\theta_2} \\ \qquad \cdots\cdots \\ w_{m-1}(x) = \frac{1}{\sin\theta_1\sin\theta_2\cdots\sin\theta_{m-2}\cos\theta_{m-1}} \\ w_m(x) = \frac{1}{\sin\theta_1\sin\theta_2\cdots\sin\theta_{m-2}\sin\theta_{m-1}} \end{cases} \tag{2}$$

where $(\rho, \theta_1, \theta_2, \cdots, \theta_{m-1})$ are the generalized spherical coordinates in objective space. Suppose the intersection point of radial $\theta = \theta^i\big(\theta^i = (\theta_1^i, \theta_2^i, \cdots, \theta_{m-1}^i)\big)$ and the unit hypersphere (when $m = 2, 3$, they are unit circumference and unit sphere respectively, hereinafter) is A_i ($i = 1, 2, \cdots, N$, N is the size of population), we can generate evenly distributed points $A_1, A_2, \cdots A_N$ on unit hypersphere in the first quadrant by appropriately selecting $\theta^i (i = 1, 2, \cdots N)$, and obtain N weight vectors from equation (2). Consequently, we have N weight vectors $W^1, W^2, ..., W^N$, where $W^i = (w_1^i, w_2^i, ..., w_m^i)$. Let

$$h_j(x) = log_2(1 + f_j(x) - f_j^*), j = 1, 2, ..., m \tag{3}$$

where f_j^* is the minimum of the objective function $f_j(x)$, it can be approximated by the minimum of $f_j(x)$ appearing in evolutionary process. The ith fitness function is defined as:

$$F_i(x) = \max_{1\le j\le m} \{W_j^i h_j(x)\}, \tag{4}$$

B. Division of weight vectors and individuals

Suppose the size of population is N, we can design N weight vectors $w^1, w^2, \cdots w^N$ from Eq.(2), and denote their corresponding ponts on the unit hypersphere as $A_1, A_2, \cdots A_N$. Similarly, we can design T weight vectors (T less than N), corresponding evenly distributed points on the unit hypersphere denoted as $\tilde{A}_1, \tilde{A}_2, \cdots \tilde{A}_T$. Through calculating Euclidean distance between each point $A_i\big(i \in \{1, 2, \cdots, N\}\big)$ with points $\tilde{A}_1, \tilde{A}_2, \cdots \tilde{A}_T$, we may find the nearest point from A_i in the group of points. Suppose the point is $\tilde{A}_t$ $\big(t \in \{1, 2, \cdots, T\}\big)$, the weight vector $w^i\big(i \in \{1, 2, \cdots, N\}\big)$ is classified into t-th class. Obviously, the total number of classes is T.

In the initial step of the algorithm, we uniformly and randomly generate $6N$ individuals and compute the objective function values of all these individuals. Through Eq.(4), we can calculate the transformed objective vector for each individual and Euclidean distances between its unit vector and points $\tilde{A}_1, \tilde{A}_2, \cdots \tilde{A}_T$. Assuming that the number of weight vectors is l_t in t-th class, we select $6l_t$ individuals from $6N$ individuals, in which the Euclidean distances calculated above to point $\tilde{A}_t$ is the minimum, and classify the selected $6l_t$ individuals into t-th class ($t = 1, 2, \cdots, T$).

In the case of unconstraint multiobjective optimization problem, since there are l_t weight vectors in t-th class as above, we can get l_t fitness functions from Eq.(3). Furthermore, we can select a best individual from the $6l_t$ individuals corresponding to t-th class with each fitness function. Selected total l_t different individuals are put in population according to l_t fitness functions. The remaining $5l_t$ individuals constitute the external set of t-th class ($t = 1, 2, \cdots, T$).

In the case of constraint multiobjective optimization problem, in order to facilitate estimating the degree of violating constraints about individuals, we deal with constraints as follows: Let

$$q(x) = \sum_{i=1}^{k} \min\big(g_i(x), 0\big).$$

When the number of individuals satisfied constraint condition are not less than the number of weight vectors l_t in t-th class, similar to the unconstraint case, we select l_t different individuals from individuals satisfied constraints condition by using fitness function corresponding to each weight vector in t-th class and put these individuals into population. The remaining $5l_t$ individuals constitute the external set of t-th class. If the number of individuals satisfying constraint condition in t-th class is less than l_t , we select l_t individuals, of which the values of $q(x)$ are maximum, from the $6l_t$ individuals in t-th class. Then according to each weight vector in the t-th class, we select l_t different individuals from above the l_t individuals and put these individuals into population. Unselected individuals constitute external set of t-th class ($t = 1, 2, \cdots, T$).

III. The Design of Evolutionary Operators

A. Crossover and mutation

Usually, the Euclidean distance between two individuals in same class is very close. Thus, the crossover operation between them can largely play a role exploring the good individuals in this area. However, the crossover operation only performing between the individuals of the same area is not very useful for exploring a wider area. The algorithm only by mutating to exploit the new area will not be very efficient. Therefore, we perform the crossover between an individual and another individual which are randomly selected from its corresponding external set. Suppose individual x^i of population belongs to the t-th class t $\big(t \in (1, 2, \cdots, T)\big)$, choosing the x^j at random in t-th class external set, after crossover, the new individual is generated as follows:

$$\tilde{x}^c(x) = x^i + rc \cdot (x^i - x^j), \tag{5}$$

where $rc = rand \cdot \big(1 - rand^{-(1-\frac{gen}{Max\_gen})^{0.7}}\big)$, $rand$ is a random number in $[-1, 1]$, gen is the current generation and Max_gen is the maximum generation. The defined rc can gradually approach to zero with increasing evolutionary generation similar to simulated annealing. If k-th component

$\tilde{x}_k^c$ of $\tilde{x}^c$ is out of the boundary of domain X, namely, if $\tilde{x}_k^c$ smaller than the low boundary $lb(k)$, $\tilde{x}_k^c = lb(k) + 0.5 \cdot rnd \cdot \big(x_k^i - lb(k)\big)$. If $\tilde{x}_k^c$ is bigger than the upper boundary $ub(k)$, $\tilde{x}_k^c = ub(k) - 0.5 \cdot rnd \cdot \big(lb(k) - x_k^i\big)$, where rnd is a random number in $[0, 1]$.

After crossover, every component in $\tilde{x}^c$ is mutated with the probability of P_m and $\tilde{x}^c$ is mutated once at least. If h-th component of $\tilde{x}^c$ turn into $\tilde{x}_h^c$ after mutation, then

$$x_h^c = \tilde{x}_h^c + rm \cdot \big(ub(h) - lb(h)\big) \tag{6}$$

where $rm = 0.15 \cdot rand \cdot \big(1 - rand^{-(1-\frac{gen}{Max\_gen})^{0.7}}\big)$ (The thought of defined rm is similar to rc.), $rand$ is a random number in $[-1, 1]$. If x_h^c is out of $[lb(h), ub(h)]$, when $x_h^c < lb(h)$, $x_h^c = lb(h) + 0.5 \cdot rnd \cdot \big(\tilde{x}_h^i - lb(h)\big)$; when $x_h^c > ub(h)$, $x_h^c = ub(h) - 0.5 \cdot rnd \cdot \big(ub(h) - \tilde{x}_h^i\big)$; rnd is a random number in $[0, 1]$, x^c is the new individual after crossover and mutation by x^i.

B. Selection operator

In the algorithm generation gen, N new individuals are generated after crossover and mutation. Calculate unit vector function values of all these individuals in the objective space and the Euclidean distances from the unit vectors to points $\tilde{A}_1, \tilde{A}_2, \cdots \tilde{A}_T$. Suppose the number of weight vectors is l_t in t-th class, select $3l_t$ individuals , of which distances to the $\tilde{A}_t$ are shortest, from the N new individuals. Suppose these individuals are $x^{i_1}, x^{i_2}, \cdots, x^{i_{3l_t}}$, then classify them into t-th classes ($t = 1, 2, \cdots, T$). Suppose there exist l_t weight vectors $W^{t_1}, W^{t_2}, \cdots, W^{t_{l_t}}$ in the t-th class and W^{t_j} correspond to the individual x^{t_j} of which fitness is the best, $W^{t_j} = (w_1^{t_j}, w_2^{t_j}, \cdots, w^{t_j})^\top (j = 1, 2, \cdots, l_t)$. Selection operator is divided into following two cases.

1) The unconstrained multiobjective optimization problem: Calculate the fitness functions $F_{t_1}(x^{i_1}), F_{t_1}(x^{i_2}), ..., F_{t_1}(x^{i_{3l_t}})$ of the individuals $x^{i_1}, x^{i_2}, \cdots, x^{i_{3l_t}}$ corresponding to weight vector W^{t_1}, and select the best individual x^*. If $F_{t_1}(x^*) \leq F_{t_1}(x^{t_1})$, let x^* exchange position with x^{t_1}, and x^{t_1} belongs to $\{x^{i_1}, x^{i_2}, \cdots, x^{i_{3l_t}}\}$. Else, x^{t_1} is unchanged. Similarly, we utilize $F_{t_j}(x)$ corresponding to weight vector W^{t_j} to select best individual and to update x^{i_t} $(j = 2, 3, ..., l_t)$. The $3l_t$ individuals randomly selected in the t-th external set are replaced by the individuals $x^{i_1}, x^{i_2}, \cdots, x^{i_{3l_t}}$.

2) The constrained multi-objective optimization problem: We combine the individuals $\{x^{t_1}, x^{t_2}, \cdots, x^{t_{l_t}}\}$ corresponding to $W^{t_1}, W^{t_2}, ..., W^{t_{l_t}}$ respectively in t-th class and new generating individuals $x^{i_1}, x^{i_2}, \cdots, x^{i_{3l_t}}$ together. Similar to method selecting initial individuals of constrained multi-objective problem, l_t different best individuals are selected from individuals satisfied constraint condition by fitness functions $F_{t_1}(x), F_{t_2}(x), ..., F_{t_{l_t}}(x)$ when the number of individuals satisfied constraint condition is not less than l_t. Otherwise, these individuals are selected from l_t individuals obtaining largest values of $q(x)$. When ending the selection of l_t best individuals in t-th class, unselected individuals randomly replace same number of individuals in the t-th class external set.

IV. The framework and characteristics of multiobjective evoluationary algorithm

A. The main framework of algorithm

We bring forward the framework of algorithm based on above chapters.

Step 1. Initialization: Set the population size N, the number of class T, the number of evolution generation gen and the maximum number of evolution generation Max_gen, give the probability P_m in mutation;

Step 2. Design the weighted vectors $w^1, w^2, \cdots, w^N$ according to Eq.(2), classify all the weighted vectors into T classes, and set the representative point $\tilde{A}_t$ in each category.

Step 3. Initialize the population, external set and the minimum value of each subobjective f_i^* $(i = 1, 2, \cdots, m)$;

Step 4. Classify N new individuals which are generated from crossover and mutation into each category in accordance with the above-mentioned manner;

Step 5. Update the population, external set and the minimum value of each subobjective f_i^* $(i = 1, 2, \cdots, m)$ by the above-mentioned selection operator in current generation;

Step 6. If $gen \leq Max\_gen$, go to step 4; otherwise, stop.

B. The characteristics of algorithm

a. In the proposed algorithm, individuals in population are divided into T classes in each generation and each class corresponds to a external set. Each individual in population crosses with randomly selected individual from the external set of its corresponding to the class. Since individuals of external set are around any individual in population corresponding to the class, the crossover between two adjacent individuals serves as local search and the crossover between two individuals whose distance is far serves as exploiting new region.

b. The proposed algorithm makes use of the max-min strategy with determined weight as fitness functions, which can maintain the diversity of population and approach the evenly distributed Pareto optimal solutions (For more information, please see literature [3]). The complexity of the algorithm is reduced by dividing the weight vectors and individuals into some regions, and the complexity of the algorithm is $O\big(\big((T + 7)m + 4(n + 1)\big)N\big)$ in each generation, where N is the size of population, m is the dimension of objective space, T is the number of class, n is the dimension of search space.

c. Since the proposed algorithm carry out selection operation in every generation after it assigns all the new individuals which are generated by crossover and mutation operation into T classes, the new individuals can be fully utilized. Furthermore, the selection operator can conveniently deal with the constraint.

V. COMPUTER SIMULATION

We simulate 13 unconstrained multiobjective test functions and 10 constrained multiobjective test functions from the literature [14] by computer. The parameters of algorithm are set as follows:

Set $P_m = \frac{1}{n}$, where n is the dimension of search space. The population size N is 100 for two objective problems, 150 for three objective problems, and 762 for five objective problems. The number of class T is 15 for two objective problems, 33 for three objective problems, and 60 for five objective problems. Set $Max\_gen = [\frac{300000-6N}{N}]$, where $[x]$ stands for the largest integer of not greater than x.

In order to measure the quality of solutions, The IGD metric [14] has been defined. Suppose P be a set of uniformly distributed points in Pareto front, and S be the approximate solutions got by different algorithm, so IGD metric is defined as follows:

$$IGD(S) = \frac{\sum_{i=1}^{n} d(v_i, S)}{n} \tag{7}$$

where $v_i \in P$, n is the number of points in P, and $d(v_i, S)$ is the minimum Euclidean distance between v_i and the solutions in A.

The Configuration of computer in simulation are System: Microsoft Windows XP Professional sp3; RAM: 1G; CPU: T2050; CPU 1.60 GHz; Computer Language: Matlab 7.0.

Table I and Table II gives the IGD value and mean CPU time used by the algorithm in this paper for each test function in 30 independently runs. The Fig.1 and Fig.2 show the the nondominated front with the lowest IGD value for each unconstrained and constrained test problems.

TABLE I

IGD VALUE AND MEAN CPU TIME(SECONDS) IN 30 INDEPENDENT RUNS FOR UNCONSTRAINED TEST PROBLEMS

Test Functions	IGD value				CPU times(s)
	Min	Max	Mean	Std	
UF1	0.004891	0.014017	0.007850	0.002088	78.1
UF2	0.008276	0.021584	0.012300	0.003317	86.7
UF3	0.004669	0.132567	0.014975	0.024045	87.6
UF4	0.041930	0.045230	0.043501	0.000650	87.5
UF5	0.100246	0.228266	0.161867	0.028244	77.0
UF6	0.060198	0.322172	0.175553	0.082933	80.3
UF7	0.005801	0.009778	0.007301	0.000890	80.9
UF8	0.066502	0.092058	0.082353	0.007330	99.2
UF9	0.047366	0.171044	0.093915	0.047064	98.2
UF10	0.240305	0.839926	0.446914	0.129614	98.1
UF11	0.123791	0.138966	0.132541	0.003624	82.3
UF12	291.471297	636.066294	444.825120	83.774592	86.00
UF13	2.028863	2.423187	2.288490	0.087217	88.1

TABLE II

IGD VALUE AND MEAN CPU TIME(SECONDS) IN 30 INDEPENDENT RUNS FOR EACH CONSTRAINED TEST PROBLEMS

Test Functions	IGD value				CPU times(s)
	Min	Max	Mean	Std	
CF1	0.000682	0.001147	0.000859	0.000110	72.4
CF2	0.002733	0.013135	0.004203	0.002635	73.6
CF3	0.090844	0.251884	0.182905	0.042127	75.1
CF4	0.008964	0.023999	0.014232	0.003293	73.7
CF5	0.058818	0.192996	0.109730	0.030676	74.4
CF6	0.009022	0.019939	0.013948	0.002586	73.7
CF7	0.053510	0.203867	0.104460	0.035116	74.2
CF8	0.047328	0.098487	0.060746	0.012972	80.4
CF9	0.046035	0.058389	0.050549	0.003357	81.5
CF10	0.105482	0.416232	0.197409	0.076004	82.2

VI. CONCLUSIONS

We define the fitness function by using a set of determined weight vectors and max-min strategy, divide the decision-making space and objective space into several small classes, and let individuals in the same class cross with each other. The information between individuals in different regions are exchanged by their offsprings which will be assigned to the region. In this paper, we propose the multiobjective evolutionary algorithm based on determined weight and sub-regional search. Because of making use of the partition searching, the complexity of algorithm is reduced and the convergence of algorithm is improved. The dealing with constraint problem becomes easier by selection operator and designing external set. The weight vector in fitness function is specially designed, so the approach lead by the fitness function is effective to obtain the evenly distributed Pareto optimal solutions on the Pareto front.

VII. ACKNOWLEDGMENT

This work was jointly supported by the Natural Science Foundation of Guangdong Province (No.07001797, No.8151009001000044), the Science and Technology Project of Guangzhou (No.2007J1-C0501).

REFERENCES

[1] K. Deb, *Multi-objective optimization using evolutionary algorithms.* New York: Wiley, 2001.

[2] Hailin Liu and Yuping Wang, *A novel multiobjective evolutionary algorithm based on min-max strategy.* Intelligent data engineering and automated learning. Vol.2690, 2003.3.

[3] Hai-lin Liu, Yuping Wang and Yiu-ming Cheung, *A multiobjective evolutionary algorithm using min-max strategy and sphere coordinate transformation.* Intelligent Automation and Soft Computing, vol.15, no.3, 2009, pp.361-384.

[4] Q. Zhang and H. Li, *A Multi-objective Evolutionary Algorithm Based on Decomposition.* IEEE Trans. on Evolutionary Computation, vol.11, no 6, pp712-731, 2007.

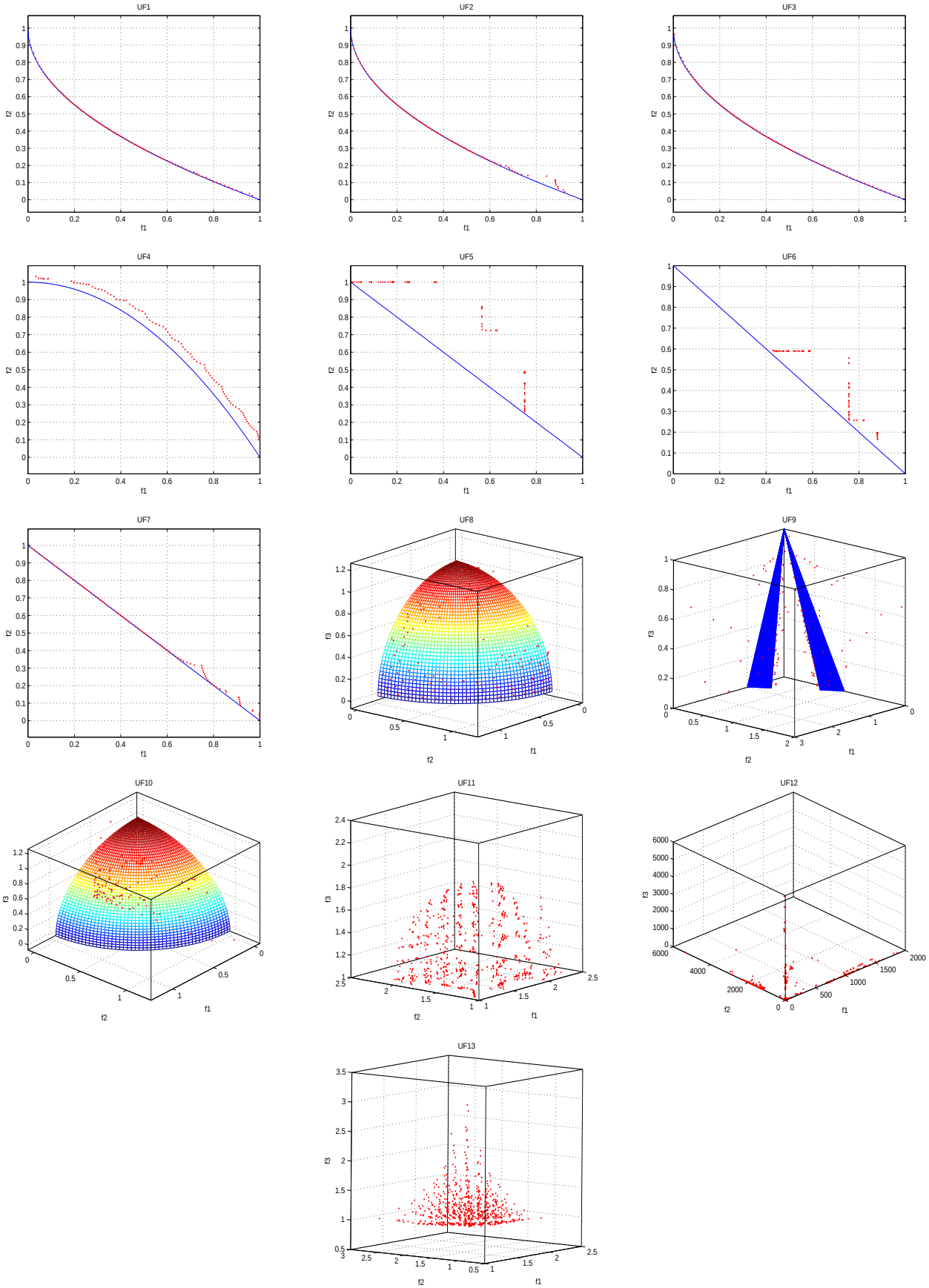

Fig. 1. Plots of the nondominated front with the lowest IGD value for each unconstrained test problems.

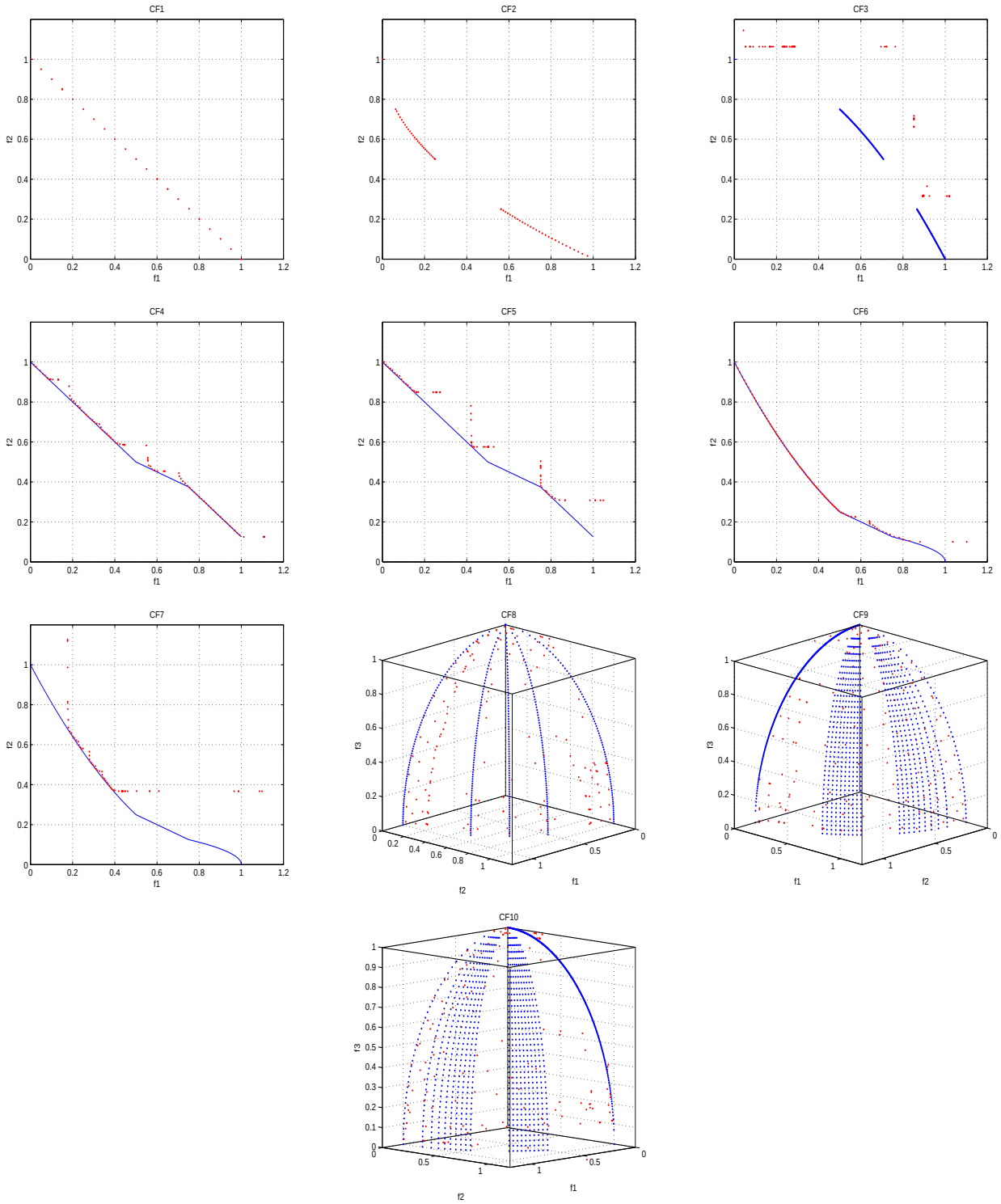

Fig. 2. Plots of the nondominated front with the lowest IGD value for each constrained test problems.

[5] Leung Y W and Wang Y P, *Multiobjective programming using uniform design and genetic algorithm. IEEE Transactions on Syst.* Man, Cybern. C, 2000, 30 (3): 293 304.

[6] J. D. Schaffer, *Multiple objective optimization with vector evaluated genetic algorithms. Proc. Of Inter.Conf.* Genetic Algorithms and Their Applications, J. J. Grefenstette (Ed.), Pittsburgh, PA, pp.93-100, 1985.

[7] Zitzler E and Thiele L, *Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach.* IEEE Transactions on Evolutionary Computation, 1999, 3(4):257-271.

[8] H. Ishibuchi and T. Murata, *Multi-objective genetic local search algorithm and its application to flowshop scheduling.* IEEE Transactions on Syst., Man, Cybern., vol.28, pp. 392-403, Aug. 2002.

[9] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II.* IEEE Transactions on Evolutionary Computation, Vol. 6, No 2, pp. 182-197, April, 2002.

[10] C. A. C. Coello, G.T. Pulido, M.S. Lechuga, *Handling multiple*

objectives with particle swarm optimization. IEEE Transactions on Evolutionary Computation, Vol. 8, No. 3, pp.256-279, Jun. 2004.

[11] C.M. Fonseca and P.J. Fleming, *Multiobjective optimization and multiple constraints handling with evolutionary algorithms Part1: Unified formulation.* IEEE Transactions on SMC, Part A, Vol. 28, No.1, pp. 38-47, January, 1998.

[12] H. Li and Q. Zhang, *Comparison between NSGA-II and MOEA/D on a Set of Multiobjective Problems with Complicated Pareto Sets.* IEEE Trans. on Evolutionary Computation 2008 accepted (http://dces.essex.ac.uk/staff/qzhang/mypublication.htm).

[13] S. Bandyopadhyay, S. Saha and U. Maulik and K. Deb, *A Simulated Annealing-Based Multiobjective Optimization Algorithm: AMOSA.* IEEE Trans. on Evolutionary Computation, Vol.12, No.3, June 2008, pp.269-283.

[14] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, Wudong Liu, and Santosh Tiwari, *Multiobjective optimization Test Instances for the CEC 2009 Special Session and Competition.* Technical Report CES-487.