Studies on Fast Pareto Genetic Algorithm Based on Fast Fitness identification and External Population Updating Scheme

Xie Qingsheng, Li Shaobo, Yang Guanci

Institute of CAD/CIMS, Guizhou University, Guiyang, China, 550003

Abstract

This paper investigates fast Pareto genetic algorithm based on fast fitness identification and external population updating scheme (FPGA) for searching Pareto-optimal set, which is based on a new approach of fast fitness identification algorithm for individual and a clustering on the basis of external population updating scheme to maintain population diversity and even distribution of Pareto solutions. Experiments on a set of multi-objective 0/1 knapsack optimization problems strongly shows that FPGA can obtain high-quality, well distributed non-dominated Pareto solutions with less computational efforts compared to other state-of art algorithms, and FPGA in convergence speed outperforms the representative SPEA.

Key words: fast genetic algorithm; Pareto optimality; fitness fast identify algorithm; fast update algorithm

1.1 Introduction

Many engineering problems involve simultaneous optimization of several incommensurable and often competing objectives. Often, there is no single optimal solution, but rather a set of alternative solutions. These solutions are optimal in the wider sense that no other solutions in the search space are superior to them when all objectives are considered. Just about that, searching Pareto-optimal is an extraordinary complicated task in high-dimensional decision-making space. Traditionally, multi-objective optimization methods transforms multi-objective functions into a single objective function through the evaluation function, However using single-objective optimization methods to searching optimal solution has shortcomings as follows .

- To constructing single-objective evaluation functions needs decision-makers to provide the profound preference knowledge or experience, but it is impossible for many engineering problems to provide such preference information;
- Most single-objective optimization methods are based on local optimal searching algorithm. Although it is possible to search a locally or globally optimal solution of single-objective optimization problems, they can not obtain Pareto-optimal solution or Pareto-optimal set or more uniform distributed optimal solution, namely parallel search is impossible, and it also can not meet the flexibility requirements of multi-objective decision-making or the requirements of environmental dynamic changes;
- When decision needs alternative Pareto-optimal solution, we must construct evaluation function and run search algorithm repeatedly.

Genetic algorithm(GA) achieves individual recombination through genetic reproduction, leads population evolution by select operator, which can deal with a set of solutions in a single population on the high efficient part for inherent parallelism of GA. References[1,2] hold that multi-objective search and optimization is the most suitable application

field of GA. Zitzler [3] put forwards the Strength Pareto EA (SPEA), and had been applied it to the multi-objective 0/1 knapsack problem successfully. SPEA employs two populations: populations P with size of N for the genetic reproduction, external populations P' whose size is N' stores non-dominated solutions found so far. According to the domination relationship between individual in populations P and member of P', SPEA determines the individual's Pareto fitness, and clustering analysis algorithm is used to maintain the population diversity. SPEA is said to be the significance Pareto-optimal search algorithm in word and deed, however, fitness evaluation and clustering analysis requires computational complexity of $O((N+N')^3)$. The reference [4] proposes a vicinity crowding algorithm, which is different from the clustering algorithm, to delete superfluous solution when current solution size exceed fixed scale of population so as to maintain the Pareto-optimal solutions' distribution uniformity, but only the "mean and variance are superior to SPEA in a certain sense ". The reference [5] bring forward a multi-objective evolutionary algorithm based on orthogonal design to solve multi-objective optimization problem, but it is just a niching evolution and segmentation iterative process materially. Other algorithms to implement niching methods uses fitness sharing method [6,7] to maintain populations diversity, which needs profound preference knowledge or experience to resolve the parameter of sharing radius, which is badly difficult to implement.

In fact, there are three main issues which must be solved when the genetic algorithm applied to solve multi-objective optimization problems. Firstly, in order to lead search to the perfect Pareto-optimal solution set front, how to assign fitness values and select effectively. Secondly, to suppress premature and to get well distributed optimal solution set and sustainable solutions, how to maintain population diversity. Thirdly, how to reserve and even improve the current Pareto-optimal solution. Based on those considerations, this paper investigates the rapid identification and evaluation of fitness algorithm, with its time complexity of $O(m(N+N')^2)$, and effective method with time complexity of $O(NN'\log N')$ to maintain population diversity.

1.2 Concept of Pareto Optimality

In essence, Multi-objective optimization is vector optimization; a general multi-objective optimization problem can be described as a vector function f that maps a tuple of m parameters (decision variables) to a tuple of n objectives. Formally:

$$\max/\min\ y = f(x) = (f_1(x),...,f_n(x))$$
$$\text{subject to}\quad x = (x_1, x_2,..., x_m) \in X\ ,\ y = (y_1, y_2,..., y_m) \in Y$$

where x is called the decision vector, X is the parameter space, y is the objective vector, and Y is the objective space. The set of solution of a multi-objective optimization problem consists of all decision vectors for which the corresponding objective vectors cannot be improved in any dimension without degradation in another—these vectors are known as Pareto optimal. Mathematically, the concept of Pareto optimality is as follows: Assume, without loss of generality, a maximization problem and consider two decision vectors $a, b \in X$. Then a is said to dominate b (also written as $a \phi b$) iff

$$\forall i \in \{1,2,...,n\} : f_i(a) \geq f_i(b) \wedge \exists j \in \{1,2,...,n\} : f_j(a) > f_j(b)$$

Additionally, in this study a is said to cover b ($a \geq b$) iff $a \phi b$ or $f(a) = f(b)$. All decision vectors which are not dominated by any other decision vector of a given set are called non-dominated regarding this set. If it is clear from the context which set is meant, we simply leave it out. The decision vectors that are non-dominated within the entire search space are denoted as Pareto optimal and constitute the so-called Pareto-optimal set or Pareto-optimal front [3].

1.3 Pareto Genetic Algorithms

1.3.1 Structure and flow of algorithm

Fast Pareto Genetic Algorithm Based on Fast Fitness Identification and External Population Update (FPGA) employs an evolution population P with size of N and an external population P' with size of N'. The former is used for crossover and mutation operator, and the latter is used for storing the identified non-dominated Pareto solutions so far. In order to implement beamed evolution search and elitist reservation strategy, binary tournament selection with replacement is used in the tow population. This study compared tow multi-objective EA's on a multi-objective 0/1 knapsack problem with nine different problem settings, each candidate solution represented by binary encode whose extent is *m*, and each binary bit $x_i \mid i \in [1, m]$ of encode figures variable's value. Thereby, the flow of FPGA is as Figure1.1, which formalization description is as follows.

1) Generate an initial population P and the empty external non-dominated set P';
2) Determine non-dominated members of P;
3) Copy non-dominated members of P to P' one by one and remove solutions within P' which are covered by any other member of P' using fast update algorithm based on clustering crowding(FUC);
4) Calculate the fitness of each individual in P as well as in P' using fast fitness identification algorithm (FIA);
5) Select individuals from $P + P'$ (multi-set union), until the mating pool is filled. In this study, binary tournament selection with replacement is used.
6) Apply problem-specific crossover and mutation operators as usual;
7) If the maximum number of generations is reached then stop, else go to step 2);

In order to produce symmetrical distributing initial population P, Gauss stochastic generator is used in the beginning. Then the comparability analysis of solution vectors has been taken to obtain more schemata so as to create prerequisite for population evolution and convergence. Determining non-dominated members of P is to find out the current Pareto-optimal curve or surface in step 2. And the FUC is used to update P' when the non-dominated individual of evolution population P copied to external population P' in turn.

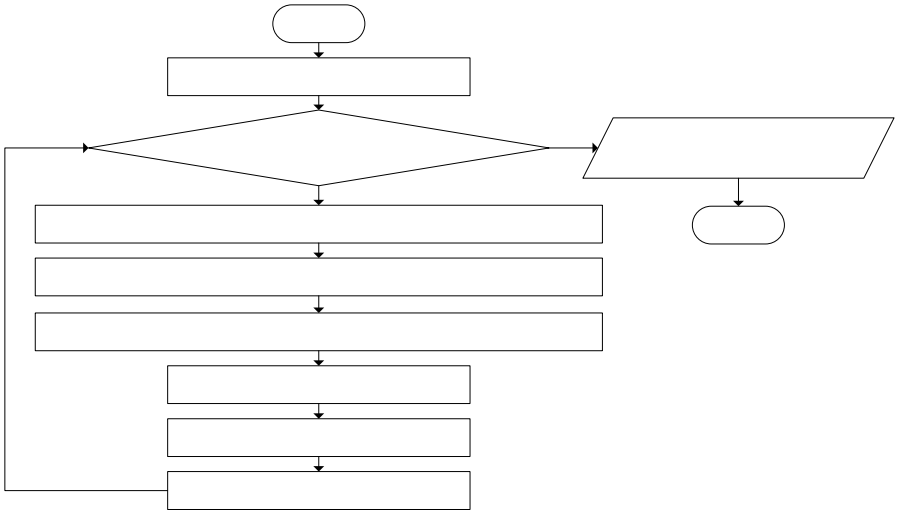

Figure 1.1 The flow of FPGA: because of the randomicity of crossover and mutation operator, there is individual, which is infeasible solution for disobeying restriction, so constraint processing has been added into the flow, and a simply repair method will be taken until all constraints are fulfilled while individual has minimum lose of object values.

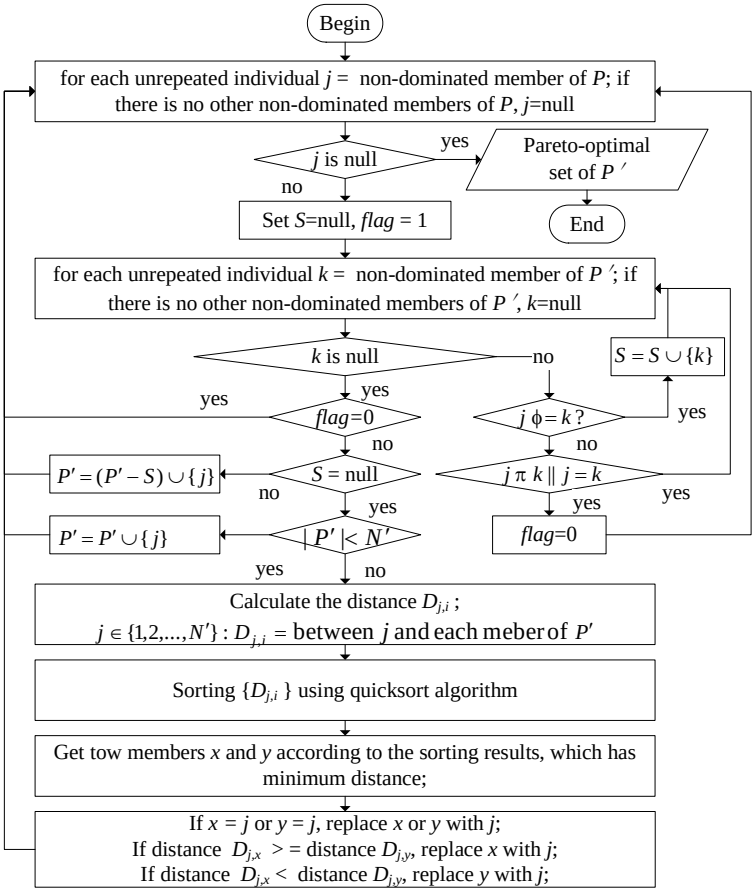

Figure 1.2 Flow of the FUC: the original intention of this algorithm is to keep the diversity of population and enlarge the differences between individuals. So, this algorithm concerns that the distribution of front edge of non-dominated solution in the objective space, we use the Euclid distance to calculate the distance between individuals.

1.3.2 Fast update algorithm based on clustering crowding (FUC)

In fact, to implement Fitness Sharing method [6, 7] needs to resolve the parameter of sharing radius. Therefore, Fitness Sharing need experiences of objective problem which are hard to obtain for many engineering problems. Meanwhile, even the experiences we have got fortunately, the relativity and subjectivity of the experiences maybe restricts the success application of Fitness Sharing. Clustering analysis [3] is complex with high computation. FPGA proposes a fast update algorithm based on clustering crowding (FUC) as the niching method, which makes the non-dominated Pareto-optimal solutions storied in the external population P' approach the optimal-optimal curve or surface. The flow of FUC is as Figure1.2, and an implementation is described by:

```
for ( ∀j ∈ {non-dominate individual of population P } )
{
    S = Φ, flag = 1 ;
    for ( k ∈ P′ && flag == 1 )
    {
        if ( j ϕ= k )   S = S Y {k} ;
        else if ( j π k || j == k )   flag=0;
        next k;
    }
```

if (*flag*==1)
if ($S \neq \Phi$) $P^{'} = P^{'} - S \,\mathrm{Y}\{j\}$;
else if($|P'| < N'$) $P' = P' \,\mathrm{Y}\{j\}$;
else
Calculating the distances between *j* and each members of P' ;
Sorting all distances in QKSORT;
Finding out one most similar member to *j*, and replace it with *j*;
next *j*;
}

The outer sentence "for" of FUC ransacks population P for N times. The inner "for" ransacks population P' for N' times. Sentence $P^{'} = P^{'} - S \,\mathrm{Y}\{j\}$ loops for N' times in worst conditions. In the deepest "else", the comparability replacement only has computational complexity of $O(N' \log N^{'})$; so computational complexity of FUC in worst condition is $O(NN' \log N^{'})$, but in the best condition, its magnitude is $O(N)$, lower than computational complexity of Fitness Sharing method and clustering analysis. Mathematically, the distance $D_{j,i}$ between members is given by the equation

$$D_{j,i} = \sqrt{\sum_{k=1}^{n} [f_k(j) - f_k(i)]^2}$$

1.3.3 Fast Fitness Identification Algorithm (FIA)

The fitness of individual expressed by the index of non-dominated individual' level in $P + P'$, the fitness of individual in level k is k. For arbitrary individual $i \in P + P'$, n_i is the number of individuals which dominate member i , S_i is the subset which member is weaker than member i , F_k is the non-dominated subset in level k. Based on those definitions, the implementation process of FIA can be described as follows.

Initialize $k = 1$;
for each solution $i \in P + P'$
$S_i = \Phi, n_i = 0$;
for each solution $j \in P + P'$
if ($j \,\pi = i$) $S_i = S_i \,\mathrm{Y}\{j\}$;
else if ($j \,\phi\, i$) $n_i = n_i + 1$;
next solution *j*;
if ($n_i = 0$) $F_k = F_k \,\mathrm{Y}\{i\}$;
next solution *i*;
while ($F_k \neq \Phi$)
$H = \Phi$;
for each solution $i \in F_k$
for each solution $j \in S_i$
$n_i = n_i - 1$;
if ($n_i = 0$) $H = H \,\mathrm{Y}\{j\}$;
next solution *j;*
next solution *i;*
$k = k + 1$;
$F_k = H$;

The outer sentence "for" determines $N' + N$ subsets S_i and $N' + N$ values of n_i , and the non-dominated solutions in level one, which computational complexity magnitude is $O((N + N')^2)$. Sentence "while" determines the level of rest

non-dominated solutions. In worst condition, there is one non-dominated solution in each rank; if so, the computation complexity to determine $N+N'-1$ ranks is $O((N+N')^2)$. So, in worst condition, the computational complexity of fitness evaluation is $O((N+N')^2)+O((N+N')^2)$ or $O((N+N')^2)$. To take FIA as fitness evaluation method, there are many individuals in each rank, which are deemed to have same competition ability. In that case, it's hard to generate an individual whose fitness is very high, which restrains the other individuals in lower fitness, and benefit to keep the diversity of population potentially in a certain sense. We have verified this conclusion in experiment.

1.4 Simulation optimization experiment

1.4.1 Test problems

Generally, a 0/1 knapsack problem consists of a set of items, weight and profit associated with each item, and an upper bound for the capacity of the knapsack. The task is to find a subset of items which maximizes the total of the profits in the subset, yet all selected items fit into the knapsack, i.e., the total weight does not exceed the given capacity. This single-objective problem can be extended directly to the multi-objective case by allowing an arbitrary number of knapsacks. Formally, the multi-objective 0/1 knapsack problem considered here is defined in the following way: Given a set of m items and a set of n knapsacks, with

$p_{i,j}$ = profit of item j according to knapsack i
$w_{i,j}$ = weight of item j according to knapsack i
c_i = capacity of knapsack i

find a vector $x=(x_1,x_2,...,x_m)\in\{0,1\}^m$,such that

$$\forall i\in\{1,2,3,...,n\}:\sum_{j=1}^{m}w_{i,j}\cdot x_j\le c_i$$

and for which $f(x)=(f_1(x),f_2(x),f_3(x),...,f_n(x))$ is maximum, where

$$f_i(x)=\sum_{j=1}^{m}p_{i,j}\cdot x_j$$

and $x_j=1$ iff item j is selected.

In this paper, in order to obtain reliable and sound results, we used nine deferent test problems where both the number of knapsacks and the number of items were varied. Two, three, and four objectives were taken under consideration, in combination with 250, 500, and 750 items. Uncorrelated profits and weights were chosen, where $p_{i,j}$ and $w_{i,j}$ are random integers in the interval [10,100]. Table 1 has shown the detailed information. The knapsack capacities were set to half the total weight regarding the corresponding knapsack

$$c_i=0.5\sum_{j=1}^{m}w_{i,j}$$

In particular, a binary string s of length m is used to encode the solution $x\in\{0,1\}^m$. Since many coding lead to infeasible solutions, a simple repair method is applied to the genotype $s:x=r(s)$. The repair algorithm removes items from the solution coded by s step by step until all capacity constraints are fulfilled. The order in which the items are deleted is determined by the maximum profit/weight ratio per item; for item j the maximum profit/weight ratio q_j is given by the equation

$$q_j=\max_{i=1}^{n}\{p_{i,j}/w_{i,j}\}$$

The items are considered in increasing order of the q_j, i.e., those achieving the lowest profit per weight unit are removed first. This mechanism intends to fulfill the capacity constraints while diminishing the overall profit as little as possible. In our testing, the probabilities of crossover (one-point) and mutation were fixed (0.8 and 0.01, respectively).

Table 1.1 Parameters that were adjusted to the problem complexity: Population *P* size (*N*), Population *P'* size (*N'*), knapsacks size (*m*), items number (*n*) and the coverage ($R_{FPGA,SPGA}$, $R_{SPEA,FPGA}$) of set

n	*m*	*N'*	*N*	$R_{FPGA,SPGA}$	$R_{SPEA,FPGA}$
2	250	30	120	0.814 95	0.048 56
	500	40	160	0.938 53	0.024 27
	750	50	200	0.959 69	0.016 43
3	250	40	160	0.996 63	0.002 03
	500	50	200	1.000 00	0
	750	75	225	1.000 00	0
4	250	50	200	1.000 00	0
	500	60	240	1.000 00	0
	750	70	280	1.000 00	0

1.4.2 Performance criteria

In order to compare the advantage of FPGA to SPEA, the coverage *R* of two sets—final Pareto-optimal set of decision vectors of running FPGA and final Pareto-optimal set of decision vectors of running SPEA—is used. Mathematically, the performance measures define is as follows:

$$R_{FPGA,SPEA} = \frac{|\{i \in P'_{FPGA}, \exists j \in P'_{SPEA} : i \phi\ j\}|}{|P'_{FPGA}|}, \quad R_{SPEA,FPGA} = \frac{|\{j \in P'_{SPEA}, \exists i \in P'_{FPGA} : j \phi\ i\}|}{|P'_{SPEA}|}$$

If Let $X', X'' \subseteq X$ be two sets of decision vectors, the function $R_{X',X''}$ maps the ordered pair (X', X'') to the interval [0,1]. The value $R_{X',X''}$ =1 means that all points in X'' are dominated by points in X'. The opposite, $R_{X',X''}$ =0, represents the situation when none of the points in X'' are covered by the set X'.

Let $X' = (x_1, x_2, ..., x_k) \subseteq X$ be a set of k decision vectors. The function $D(X')$ gives the distance enclosed by the union of the polytypic $p_1, p_2, ..., p_k$, where each p_i is formed by the intersections of the following hyperplanes arising out of x_i, along with the axes: for each axis in the objective space, there exists a hyperplane perpendicular to the axis and passing through the point $(f_1(x_i), f_2(x_i), ..., f_n(x_i))$. In the two-dimensional (2-D) case, each represents a points defined by the points $(f_1(x_i), f_2(x_i))$. The $D(X')$ is given by the equation:

$$D(X') = \sqrt{\sum_{j=1}^{k} (f_j(x_i))^2}$$

1.4.3 Experimental result and analysis

On all test milt-objective 0/1 knapsacks problems, 10000 generations were simulated per optimization run, and FPGA, SPEA runs 40 times independently at the same initial population. After the 40 times optimization run, the arithmetic average values $R_{FPGA,SPGA}$ $R_{SPEA,FPGA}$ are shown as Table 1.1. As can be seen in Table1.1, FPGA in quality of final non-dominated Pareto-optimal solutions outperform the state-of-the-art SPEA on all test 0/1 knapsack problems, and the more knapsacks and items involved, the greater the value for $R_{FPGA,SPGA}$, namely the more nakedness of FPGA's advantage. When the knapsacks n >2, FPGA covers more than 99% of the fronts computed by SPEA. In contrast, SPEA covers less than 5% of outcomes of FPGA at the best condition. So, according to the coverage values of two set, we can draw a conclusion that FPGA seems to provide the best performance comparing to SPEA.

In order to observe the distribution of non-dominated Pareto solutions in the different evolution process, two objectives problem was chose under consideration from the 40 times independent running of FPGA and SPEA randomly, in combination with 250, 500, and 750 items, and the distribution and evolution trend of non-dominated Pareto-optimal set of external populations P is shown as Figure1.3, where the tradeoff fronts obtained in two runs are plotted for the 2-D problems. As can be seen clearly in Table1.1, as the increase of evolutionary generations, the non-dominated Pareto stored in the population P' can uniformly approximate every part of Pareto-optimal front, and FPGA has more uniform distribution and more rapid convergence trends comparing to SPEA .

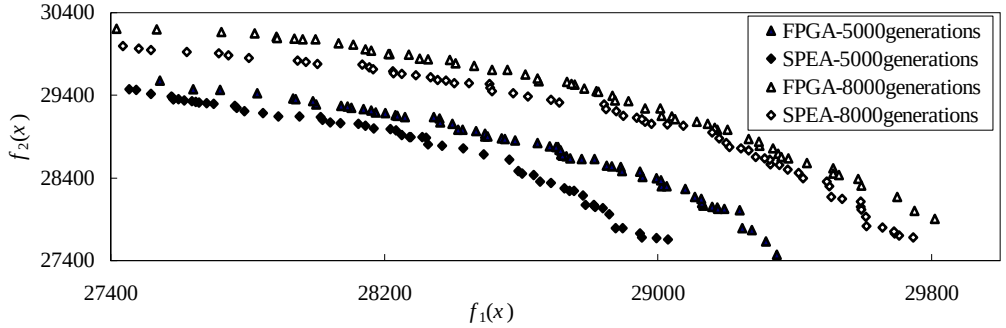

Figure 1.3 Tradeoff fronts for two knapsacks: here, the distribution and evolution trend of non-dominated Pareto-optimal set of extern populations P' are described.

Considering two, three, and four objectives, in combination with 750 items, we respectively run FPGA and SPEA 40 times respectively, and then calculate the arithmetic average values of $D(X')$. The increasing trend curve of $D(X')$, as the increase of evolutionary generations, is described as Figure1.4. The trend curve to which correspond FPGA in Fig.4 has sharper slope ratio-of-rise, which shows that FPGA has advantage in convergence speed at the beginning stage of evolutionary searching. When it achieves the relative stable stage, various curves corresponded to FPGA correspondingly locate in the top of trend curves to which belong SPEA, which indicates that FPGA can obtain high accuracy non-dominated Pareto solutions at the later part stage of evolutionary searching. Thereby, the conclusion is that FPGA in convergence speed and quality of non-dominated Pareto solution is superior to SPEA.

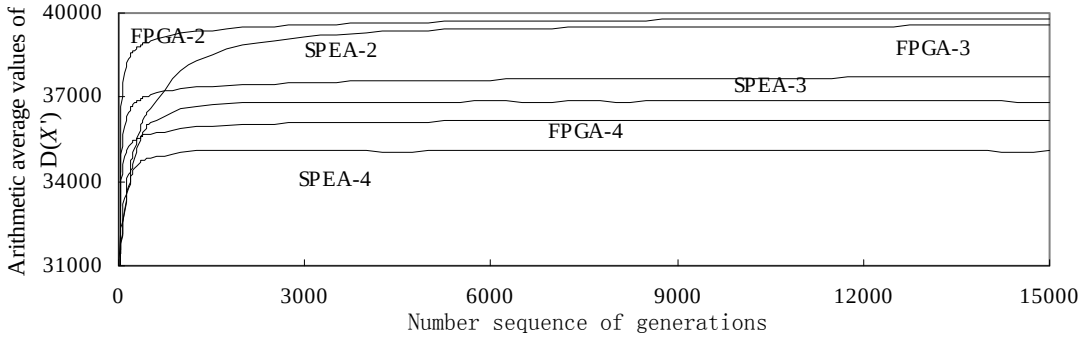

Figure1.4 The increasing trend curve of the arithmetic average values of $D(X')$: here, along with the axes: for each axis in the objective space, there exists a plane perpendicular to the axis and passing through the point $(f_1(x_i), f_2(x_i))$, each represents a point defined by the points $(f_1(x_i), f_2(x_i))$.So the trends curve are plotted by all points for each combinations

Reference [8,9] hold that genetic drift in evolutionary algorithms is the major cause of the search for global optima getting premature stagnation on single local optimum, but FUC can maintain the diversity of population at any stage of evolutionary searching, for which provides various schema for next evolution. Namely, FUC naturally suppress the search for global optima getting premature stagnation on single local optimum, and prevents genetic drift.

1.5 Conclusions

In this article, we propose a kind of fast Pareto genetic algorithm based on fast fitness identification and external population updating scheme for searching Pareto-optimal set, which supplies alternative Pareto-optimal solution set for multi-objective decision-making. FPGA is unique in two respects.Firstly, we put forward fast update algorithm based on clustering crowding (FUC) for maintaining population diversity and even uniform distribution of Pareto solutions, which realization is based on external population updating scheme by washing out the most similar individuals of external population step by step. Secondly,by analyzing the problems such as complex computation and high computation cost in the fitness evaluation of Pareto-optimal set, we propose a kind of fast fitness identification algorithm with lower computation complexity comparing to other congeneric methods.

1.6 Acknowledgements

This research is supported by the National Natural Science Foundation of China under Grant 50575047 and 50475185, 863 Project of China under Grant 2006AA04Z130, West Light Project of Chinese Academy of Science ([2005]404), Foundation of Guizhou Province in China (2006-20).

1.7 References

[1] Marínez M A, Sanchis J, Blasco X. Genetic Algorithms for Multiobjective Controller Design[C]//Proc. of the 1st International Work-conference on the Interplay Between Natural and Artificial Computation. 2005: 242.

[2] Li Bin, Chen Liping, Huang Zhengdong. Product Configuration Optimization Using a Multi-objective Genetic Algorithm[J]. International Journal of Advanced Manufacturing Technology,2006, 30(1): 20-29.

[3] Zitzler E, Thiele L. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach [R]. Technical Report: TIK43, 2002: 19-26.

[4] Zhai Yusheng, Cheng Zhihong, Chen Guangzhu, Li Liu. Multi-objective Optimization Immune Algorithm Based on Pareto[J]. COMPUTER ENGINEERING AND APPLICATIONS, 2006, 42(24): 24-27.

[5] ZENG San-You, WEI Wei, KANG Li-Shan, YAO Shu-Zhe. A Multi-Objective Evolutionary Algorithm Based on Orthogonal Design [J]. Chinese Journal of Computers, 2005, 28(7): 1153-1162.

[6] Wang Li, Liu Yushu, Xu Yuanqing. Multi-objective PSO Algorithm Based on Fitness Sharing and Online Elite Archiving[J]. Lecture Notes in Computer Science, 2006, 4113: 964-974.

[7] Horn J. Niche distributions on the Pareto optimal front[C]//Proc. of International Conference on Evolutionary Multi-criterion Optimi- zation, 2003: 365-375.

[8] Jonathan E R. Population Dynamics of Genetic Algorithms [M]. Studies in Fuzziness and Soft Computing, Berlin: Springer, 2005, 183: 19-43.

[9] Schmitt L M. Foundation study: Theory of genetic algorithms [J]. Theoretical Computer Science, 2001, 259(1): 1-61.