Strategies Based on Polar Coordinates to Keep Diversity in Multi-Objective Genetic Algorithm

Da Kuang
College of Information Engineering
Xiangtan University
Xiangtan, CN 411105
biscan607@163.com

Jinhua Zheng
College of Information Engineering
Xiangtan University
Xiangtan, CN 411105
jhzheng@xtu.edu.cn

Abstract- Most of the Multi-Objective Genetic Algorithms (MOGAs) can be divided into two steps, namely constructing the non-dominated set and truncation procedure. The quality of the latter directly affects the efficiency and the distribution of MOGA. In this paper, a new MOGA named PCGA (Polar Coordinates Genetic Algorithm) is proposed. The technique, which uses grids to keep diversity of solutions with polar coordinates, is introduced into PCGA. The time complexity of its truncation approach is higher than that of NSGA2, but is greatly lower than that of SPEA2. Meanwhile, though PCGA's distribution is not as good as that of SPEA2, it makes a large improvement with respect to that of NSGA2.

1 Introduction

There are many multi-objective optimization problems (MOOPs) in the real world and MOGAs can be used to solve these kinds of problems. They can obtain a number of trade-off solutions in a single simulation run among which there is not a solution that has optimal values for all objectives. These trade-off solutions are known as Pareto-optimal solutions.

Over the last decade, many MOGAs have been suggested. Among these MOGAs, NSGA2 proposed by Deb et al. in ([Deb02a]) and SPEA2 proposed by Zitler et al. in ([Zitler01]) are the most outstanding ones. NSGA2 has a fast converging speed but can not diversify its solutions well, while SPEA2 has a uniform distribution ([Jinhua04]) but an expensive time complexity. To a great extent, it is caused by the difference between their truncation approaches. In NSGA2, a concept of crowd-distance is introduced. The crowd-distances of all solutions are calculated by a sorting procedure on each of their objective values. Afterwards the solutions with larger crowd-distance are retained in the archive. It is clear that the time complexity of this truncation approach is $O(N \log N)$ (where N is the size of population). Compared with NSGA2, the time complexity of the truncation approach in SPEA2 may reach $O(N^3)$ in the worst case since this truncation approach is based on distance. Besides these two MOGAs, Corne et al. suggested PESA in ([Corne00]) and its improved version PESA2 in ([Corne01]). In their algorithms, the search space is divided into multiple hypercubes according to each objective. It can easily identify which hypercube a solution belongs to. The distribution can certainly be improved when the solution in the most crowded hypercube is kicked out. The truncation approach of PESA and PESA2 is faster than that of SPEA2. However, some shortcomings can be found:

- The whole search space must be divided.
- The scale of each unit hypercube may be difficult to decide. If we choose an improper scale, we may get a poor distribution.

Therefore, we propose a new MOGA, which we call PCGA (Polar Coordinates Genetic Algorithm). A technique using polar coordinates for diversity maintenance is applied to our algorithm. In PCGA the whole search space does not need to be taken into account for division and the only thing to do is to divide the Pareto front, which makes the identification of unit grid scale much easier. The time complexity of the truncation approach in PCGA is less than $O(N^2)$.

The remainder of this paper is arranged as follows. Some preliminary knowledge and related work is given in section 2; section 3 explains the way to divide the grid on the Pareto front; we detail the main process of PCGA and analyze its truncation approach in section 4; in section 5, according to some experiments, we compare PCGA with NSGA2 and SPEA2, then investigate the performance of the truncation approach using polar coordinates. Finally, the conclusion and future work is addressed in the last section.

2 Preliminary knowledge and related work

Laumanns et al. once analyzed several popular MOGAs and gave a unified model of MOGA ([Laum00]). He said that generally a population and an archive are maintained during the search process of an MOGA, which can be divided into two key steps namely constructing the non-dominated set and truncation procedure. The two steps iterate until the terminating condition is satisfied. To construct the non-dominated set, excellent solutions (non-dominated solutions) from population are placed into the archive. When the number of non-dominated solutions exceeds the size of archive $\bar{N}$ (generally equals the size of population), those more crowded solutions are removed from the archive during the truncation procedure. Accordingly, a well-spread distribution of solutions can be achieved and the diversity is kept.

2.1 Definition of non-dominated set

There are multiple objectives in MOGAs and a solution must be evaluated on all its objective values that are conflicting with each other. So it is difficult to find a solution with optimal values for all objectives. Hence, Pareto concept is introduced to obtain the trade-off optimal solutions in MOOPs. Here, we assume there is an MOOP that minimizes the components of a vector $F(x) = (f_1(x), f_2(x), \cdots, f_m(x))$, $i \in \{1, 2, \cdots, m\}$; m denotes the number of objectives. Some relative concepts will be explained for the reader's convenience.

Pareto Dominance: A vector $u(u_1, u_2, \cdots, u_m)$ is said to dominate $v(v_1, v_2, \cdots, v_m)$ (denoted by $u \prec v$) if and only if $\forall i \in \{1, 2, \cdots, m\}$, $u_i \le v_i$ and $\exists j \in \{1, 2, \cdots, m\}$, $u_j < v_j$.

Pareto non-dominated: A vector is said to be non-dominated if and only if $\neg\exists v, v \prec u$.

Non-dominated set: The union of all the Pareto non-dominated solutions is called non-dominated set.

2.2 Related work

In ([Most03a]), Mostaghim suggested Sigma method which transforms a vector $f(f_1, f_2, \cdots, f_m)$ into σ vector of $\binom{m}{2}$ elements. For example, a vector $f(f_1, f_2, f_3)$ of three coordinates is defined as follows:

$$\vec{\sigma} = \begin{pmatrix} f_1^2 - f_2^2 \\ f_2^2 - f_3^2 \\ f_3^2 - f_1^2 \end{pmatrix} / \left(f_1^2 + f_2^2 + f_3^2\right) \qquad (1)$$

The distance between σ vectors can be used to measure the angle between two vectors. The vectors on the same line have the same σ vectors. This method then was applied in MOPSO to find the best local guide ([Most03a]). In addition, a diversity metric using Sigma method was also proposed by Mostaghim ([Most03b]): Initially generate several reference lines, each of which has a flag (equals to 0 at the beginning), through the Pareto front. If there is an individual that has a distance less than a given value d to a reference line, set the flag of this line to 1. Then divide the number of reference lines whose flag is 1 by the total number of the reference lines. Finally regard the division result as the diversity of a given solution set.

3 Grid division using polar coordinates

From the related work mentioned above, it is known that Sigma method adopts the concept of angle to deal with the relationship between individuals. Since polar coordinates are based on angles, here we introduce polar coordinates into our algorithm for grid division. As we all know, the truncation procedure is called right after the number of non-dominated solutions exceeds $\bar{N}$. At this time, these non-dominated solutions have converged to a surface near the Pareto-optimal front. Because of this fact, we only need to divide Pareto front into grids, which is done by adopting the concept of polar coordinates.

For a vector of a solution $u(u_1, u_2, \cdots, u_m)$, its polar coordinates should be expressed like this:

$$\begin{aligned} u_1 &= \rho\cos\theta_1\cos\theta_2\cos\theta_3\cdots\cdots\cos\theta_{m-1} \\ u_2 &= \rho\cos\theta_1\cos\theta_2\cos\theta_3\cdots\cdots\sin\theta_{m-1} \\ u_3 &= \rho\cos\theta_1\cos\theta_2\cdots\cdots\sin\theta_{m-2} \\ &\cdots\cdots\cdots\cdots \\ u_m &= \rho\sin\theta_1 \end{aligned} \qquad (2)$$

ρ denotes the radius, θ_i denotes the polar angle of the solution. They can be calculated in this way:

$$\begin{aligned} \rho &= \sqrt{u_1^{\,2} + u_2^{\,2} + \cdots\cdots + u_m^{\,2}} \\ \theta_1 &= \arcsin(u_m / \rho) \\ \theta_2 &= \arcsin(u_{m-1} / \rho\cos\theta_1) \\ &\cdots\cdots\cdots\cdots \end{aligned} \qquad (3)$$

The rest of θ_i can be deduced by analogy. If the reference point is $(a_1, a_2, \cdots, a_m)$ not $(0, 0, \ldots, 0)$, we set the vector $u(u_1, u_2, \cdots, u_m)$ to be $u(u_1 - a_1, u_2 - a_2, \cdots, u_m - a_m)$ and compute $\rho, \theta_1, \theta_2, \cdots, \theta_m$ as in the procedure above. We can decide which grid the solution belongs to just by its polar angle θ_i without considering the value of ρ.

Our division of grids depends on the size of Pareto

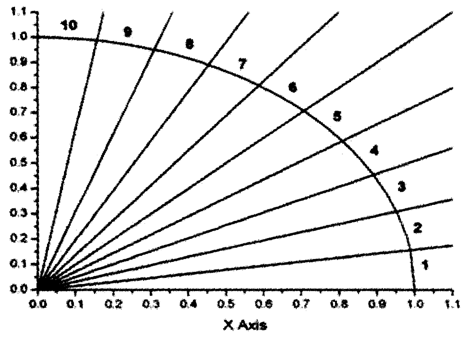


Figure 1: A 2-dimensional division case

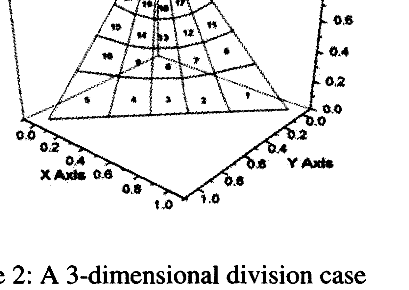


Figure 2: A 3-dimensional division case

front. Figure 1 gives a 2-dimensional case. Assume that Pareto front is the curve: $x^2 + y^2 = 1$ and we want to make a 10-grid division on the curve. In this case, the Pareto front is in the first quadrant ($0 < \theta_1 < \pi/2$), so the corresponding angle is a right angle. We divide this right angle into 10 identical angles, which forms 10 grids on the Pareto front. We can also imagine that if the Pareto front has a strong bend (e.g. the curve: $x^2 + y^2/10000 = 1$), different grids of the curve will have very different lengths. For the 3-dimensional case shown in Figure 2, the Pareto front is also in the first quadrant ($0 < \theta_1 < \pi/2$, $0 < \theta_2 < \pi/2$). Assume that the Pareto front is the plane: $x + y + z = 1$ and we want to have 25 grids on this plane. Here, two corresponding angles are considered for division. One is crossed by the X axis and the Y axis, the other is crossed by the Z axis and the XY plane. We divide each of the two right angles into 5 identical angles so that there will be 25 grids on the Pareto front. Therefore, for an m-dimensional case, we can deduce that if we want to make a division of $\bar{N}$ grids on the Pareto front, we only have to divide each corresponding angle of the Pareto front into $\sqrt[m-1]{\bar{N}}$ parts. However, it must be pointed out that the grids on the Pareto front are non-uniform. For instance, in Figure 2 grid 1 is bigger than grid 11 and grid 21.

4 Description of PCGA

Here, we set the size of archive to be $\bar{N}$ which equals population size N and assume that the number of non-dominated solutions is *count.*

4.1 Main Algorithm

The main structure of PCGA is given in the following steps:

1. Initialize the population *P*, evaluate every individual in *P* and calculate its radius ρ and polar angle θ_i ($i = 1, 2, \cdots, m-1$); then set archive to be empty.
2. Perform selection, crossover, mutation on *P* to generate a new population P' ; then evaluate every individual in P' and calculate its radius ρ and polar angle θ_i ($i = 1, 2, \cdots, m-1$).
3. Combine *P* and P' ; then copy the non-dominated individuals in *P* and P' to archive.
4. If *count* is larger than $\bar{N}$, then perform the truncation procedure; otherwise, fill the rest of archive with the individuals randomly picked up from the dominated individuals.
5. Set *P* of the next generation to be archive; if the terminating condition is satisfied, the algorithm ends, otherwise, go to step 2

4.2 Method of constructing the non-dominated set

The method of constructing the non-dominated set used in PCGA was first suggested by Deb.

```
Function establish-Nds(Pop: population)
{
  Q=Pop;
  Nds= $\varnothing$ ;
  Randomly select  $x \in Q$ ;
  $Q = Q - \{x\}$ ;
  Nds=Nds $\cup$ { x };
  for ( $y \in Q$ )
  {
    dominated=false;
    for ( $x \in Nds$ )
      if ( $x \prec y$ ) then
      {
        dominated=true;
        break;
      }
      else
        if ( $y \prec x$ ) then
          Nds=Nds − { x };
    if (not dominated) then
      Nds=Nds $\cup$ { y };
  }
}
```

Finally, *Nds* is the non-dominated set we want.

4.3 Truncation approach

1. According to the size of the Pareto front, divide each corresponding angle of the Pareto front by $\sqrt[m-1]{\bar{N}}$ to form $\bar{N}$ grids on the Pareto front, then calculate the number of individuals in every grid denoted by number[*i*] $(i = 1, 2, \cdots\cdots, \bar{N})$.
2. Find the grid which has the maximal number of individuals : number[*max*] $(max \in 1, 2, \cdots\cdots, \bar{N})$.
3. Randomly remove one individual from this grid.
4. number[*max*]= number[*max*] − 1.
5. *count*=*count* − 1.
6. If *count*> $\bar{N}$, go back to step 2, otherwise, finish the truncation procedure.

Obviously the most computational step is step 2, its time complexity is $O(\bar{N})$. The time complexity of the whole truncation procedure depends on *count* − $\bar{N}$. Since $\bar{N}$ equals N and the number of non-dominated individuals can not be more than the size of the constructing set (here, the combination of the population *P* and the new population P'), it is clear that $count < 2N$, which means $count - \bar{N} < 2N - N < N$. It can be concluded that the time complexity of this truncation approach is less than $O(N^2)$. From the approach mentioned above, we can see one individual within the most crowded grid is removed each loop. In this way more and more grids will be filled with individuals. Of course, when every grid keeps individuals, there will be only one individual in each grid.

5 Experiments

In order to test the time efficiency and the solutions' distribution of PCGA, we compare PCGA with NSGA2 and SPEA2 in the following experiments and investigate their performances. Here, we focus our experiments on 2-dimensional and 3-dimensional problems. Our test problems chosen are DTLZ1 and DTLZ2, which are proposed by Deb et al. in ([Deb02b]). The Pareto-optimal front of DTLZ1 is a hyper-plane. There are $(11^k - 1)$ local Pareto-optimal fronts paralleling in the search space, each of which may attract an MOGA. However, DTLZ2 has only one global spherical Pareto-optimal front. In all of our following experiments, we use binary coding, a crossover probability of 0.8, a mutation probability of $1/len$ (where len is the length of chromosome), and apply binary tournament selection. All the experiments are performed in this environment: a Pentium 4 2.4G CPU with a Memory of 256 Mb.

5.1 2-dimensional test problems

To do the experiments on 2-dimensional test problems, we use a population of 100 and run it for 200 generations. Certainly, the size of archive is set to be 100. The results of diversity in Table1 and Table2 are obtained using the diversity metric suggested in ([Khare03]), which can be adapted to 2-dimensional test problems. The higher the metric value, the better the distribution is. It can be observed from Table 1 that PCGA and NSGA2 get similar results for both diversity and time on DLTZ1. Nevertheless, none of their diversity is as good as SPEA2's although SPEA2 spends more time. Table 2 presents the performance of these three MOGAs on DLTZ2. Clearly, NSGA2 is the best in both of the two aspects. PCGA's diversity is inferior to SPEA2's but it

MOGA	Diversity	Time(sec)
PCGA	0.841	1.121
NSGA2	0.836	1.065
SPEA2	0.927	8.032

Table 1: Comparison of three MOGAs in terms of diversity and time on DTLZ1 for 2-dimension

MOGA	Diversity	Time(sec)
PCGA	0.785	1.906
NSGA2	0.808	1.877
SPEA2	0.807	8.453

Table 2: Comparison of three MOGAs in terms of diversity and time on DTLZ2 for 2-dimension

can run much faster than SPEA2. Undoubtedly, PCGA gets a satisfactory result, but NSGA2 is the winner on these two 2-dimensional problems. It is mainly due to the superiority of the truncation approach in NSGA2 on 2-dimensional problems. In the 2-dimensional case, the smaller a solution's crowd-distance is, the more crowded area the solution may be located in. Thus, NSGA2 can achieve an ideal distribution. However, in 3-dimensional or even higher dimensional cases, the two neighbors of a solution on one objective may be possibly far from each other on another objective. So the crowd-distance of a solution, which is the average distance of its two neighbors along each of the objectives, may not correctly reflect its crowd degree. So NSGA2's distribution becomes poorer on higher dimensional problems.

5.2 3-dimensional test problems

Here, Population 256, max generations 300 and archive 256 are selected. So the two corresponding angles of Pareto front are equally divided into 16 ($\sqrt{256}$) parts.

Figure 3 shows the distribution of solutions obtained using PCGA, NSGA2 and SPEA2 on 3-dimensional problem of DTLZ1. Apparently SPEA2 distributes its solutions most uniformly among the three, PCGA also does well while NSGA2 gets a poor distribution. Now let's see their time consumption on this problem. PCGA needs 7.773 seconds to complete its search, NSGA2 uses 7.703 seconds while SPEA2 makes a surprising result: 160.484 seconds.

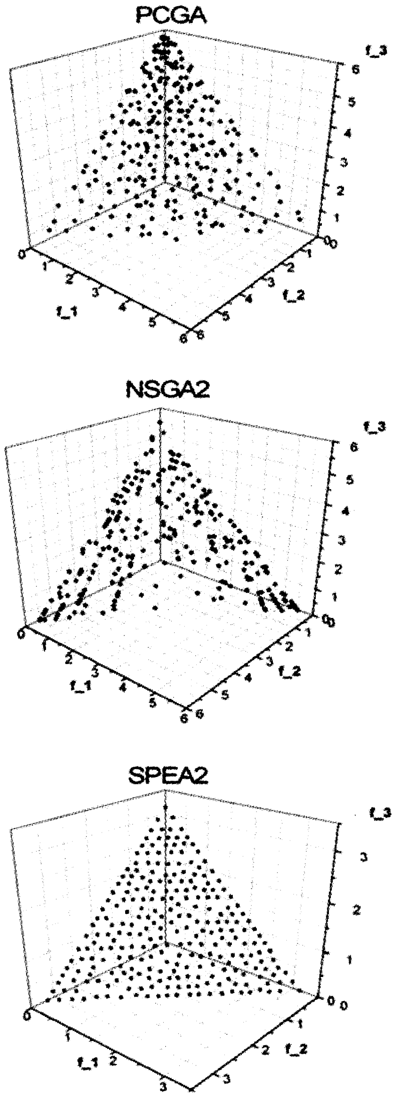


Figure 3: Comparison of PCGA, NSGA2 and SPEA2 applied on 3-dimensional problem of DTLZ1

Figure 4 shows the distribution of solutions obtained using PCGA, NSGA2 and SPEA2 on 3-dimensional problem of DTLZ2. A similar conclusion can be drawn from Figure 4, that on the aspect of distribution SPEA2 performs best, followed by PCGA and NSGA2. The time consumption for the three algorithms is 9.924, 10.003 and 135.843 seconds respectively.

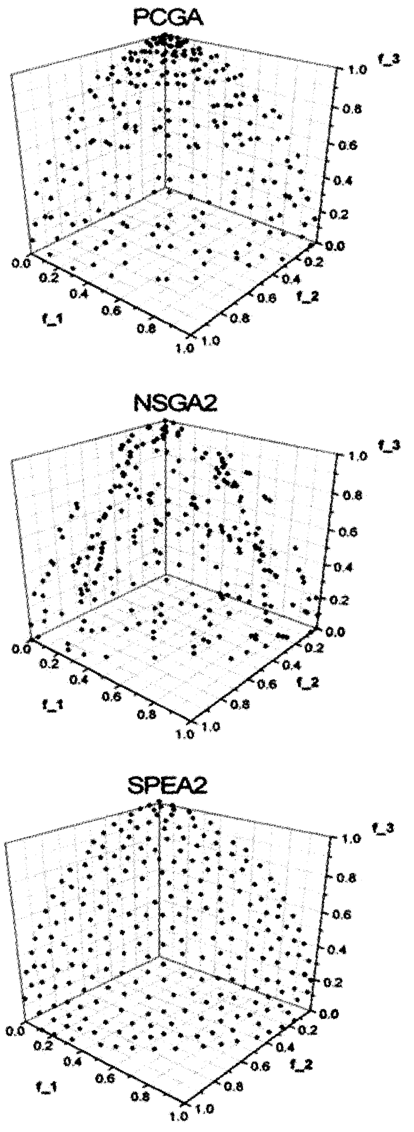


Figure 4: Comparison of PCGA, NSGA2 and SPEA2 applied on 3-dimensional problem of DTLZ1

From the two comparison results given above, PCGA runs as fast as NSGA2 while its distribution is far better than NSGA2's. Though SPEA2 obtains the most diverse solution set, it sacrifices its efficiency. Undoubtedly, PCGA achieves the best performance of the three. Interestingly, seen from Figure 3 and Figure 4, for PCGA the solutions on the top side lay close to each other while the solutions on the bottom side are sparsely spread. It is caused by the non-uniform division on the Pareto front, although we make a uniform division on each corresponding angle of Pareto front. That is why PCGA does not obtain as good of a diversity as SPEA2.

Next, let's consider the truncation approach of PCGA and check its performance.

Figure 5 illustrates the time consumption of truncation approach fluctuating with generation on 3-dimensional problem of DTLZ2. Seen from this figure, the time consumption of PCGA truncation approach is a little higher than that of NSGA2, but much lower than that of SPEA2. Since NSGA2 only needs to sort all solutions on each objective, the time consumption of its truncation approach is not very sensitive to the amount of non-dominated solutions. Thus, its curve trembles slightly. However, the number of non-dominated solutions directly relates to the efficiency of the truncation approach in PCGA and SPEA2. So their curves tremble fiercely with the number of non-dominated solutions which fluctuates from generation to generation.

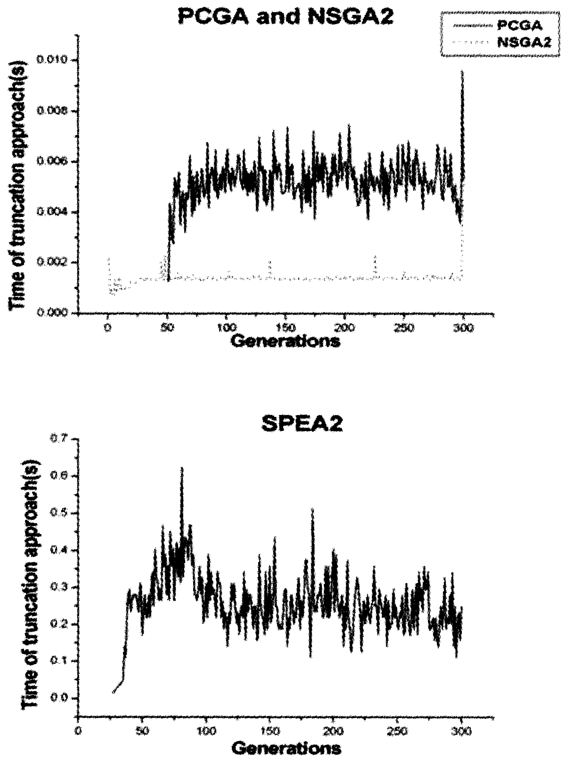
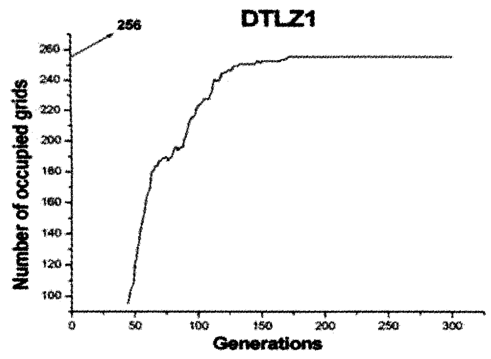


Figure 5: The run time of truncation approach for PCGA, NSGA2 and SPEA2, fluctuating with generation on 3-dimensional problem of DTLZ2

Although the grids on the Pareto front are non-uniform, finally the placement of solutions obeys this rule that each grid keeps only one solution. Perhaps the distribution of PCGA is not very satisfactory for distance-based diversity metrics, but in the sense of diversity it does not lose any representative solutions. We will prove that the distribution of PCGA is good with the experiment below.

Figure 6 shows the amount of occupied grids which fluctuates with generation. Occupied grid is the grid which has at least one solution in it. It can be seen from the figure that the amount of occupied grids increases rapidly and stabilizes after a number of generations. In the beginning generations, since the solutions are not well



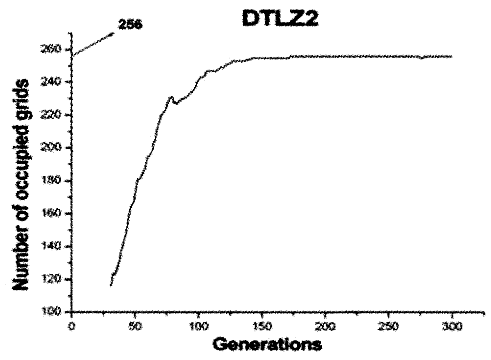


Figure 6: The number of occupied grids for PCGA fluctuating with the generation on 3-dimensional problem of DLTZ1 and DLTZ2

spread, there are some grids filled with many solutions and also some grids with no solutions in it. So the amount of occupied grids is at a low value. With the evolutionary process going on, solutions are adjusted by the truncation approach. The solution in the most crowded grid is kicked out, so that eventually each grid will be occupied with only one solution after a certain generation. Here we have set archive size to be 256; theoretically the amount of occupied grids should be 256 after some generations. Obviously we can draw the same conclusion from Figure 6 (from Figure 6 we can see that the number of occupied grids stabilizes at 256 after 170 generations).

6 Conclusion and future work

A new MOGA called PCGA is proposed in this paper as it performs satisfactorily according to our experiments. A truncation approach using polar coordinates is introduced into PCGA. It divides the Pareto front into multiple grids and removes the solution in the most crowded grid to make sure that a well spread distribution will be obtained. Because of its high efficiency and fair distribution, this truncation approach can be applied to any other MOGAs for diversity maintenance. By the definition of polar coordinates, we can surely extend this approach to the problems of any objectives. However, due to the division strategy used in PCGA, it can only be used to solve the problems with uniform Pareto-optimal front. In our future work, we wish to make some modifications on PCGA such as using a new method of constructing the non-dominated set ([Jinhua04]) or improving its truncation approach so as to apply it to the problems with higher-dimension and disconnected Pareto front.

Bibliography

Corne, D.W., Knowles, J.D. and Oates, M.J. (2000) "The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization, "*Parallel Problem Solving from Nature* - PPSN VI: 839–848.

Corne, D.W., Jerram, N.R., Knowles, J.D. and Oates, M.J. (2001) "PESA-II: Region- based Selection in Evolutionary Multiobjective Optimization, " In *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO-2001): 283-290.

Deb, K., Pratap, A., Agrawal, S. and Meyrivan, T. (2002a) "A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II, " *IEEE Transactions on Evolutionary Computation*, 6(2): 182-197.

Deb, K., Thiele, L., Laumanns, M. and Zitzler, E. (2002b) "Scalable multi-objective optimization test problems, " In *proceeding of the Congress on Evolutionary Computation* (CEC2002): 825-830.

Jinhua, Zheng., Charles, Ling., Zhongzhi, Shi. and Yong, Xie. (2004) "Some Discussions about MOGAs: Individual Relations, Non-dominated Set, and Application on Automatic Negotiation, " *Congress on Evolutionary Computation* (CEC2004), USA.

Khare, V., Yao, X. and Deb, K. (2003) "Performance scaling of multi-objective evolutionary algorithms, " In *Proceeding of the second International Conference on Evolutionary Multi-Criterion Optimization* (EMO-2003), in press.

Laumanns, M., Zitzler, E. and Thiele, L. (2000) "A unified model for multiobjective evolutionary algorithms with elitism, " In *Congress on Evolutionary Computation* (CEC 2000), volume 1: 46–53.

Mostaghim, S., Teich, J. (2003a) "Strategies for Finding Good Local Guides in Multi-objective Particle Swarm Optimization," *2003 IEEE Swarm Intelligence Symposium Proceedings*: 26-33.

Mostaghim, S., Teich, J. (2003b) "The role of e-dominance in Multi-objective Particle Swarm Optimization," *Congress on Evolutionary Computation* (CEC2003).

Zitzler, E. (2001) "SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization, " *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*: 95–100.