A PROBABILITY-DRIVEN SEARCH ALGORITHM FOR SOLVING MULTI-OBJECTIVE OPTIMIZATION PROBLEMS

NGUYEN HUU THONG[*], TRAN VAN HAO[**]

ABSTRACT

This paper proposes a new probabilistic algorithm for solving multi-objective optimization problems - Probability-Driven Search Algorithm. The algorithm uses probabilities to control the process in search of Pareto optimal solutions. Especially, we use the absorbing Markov Chain to argue the convergence of the algorithm. We test this approach by implementing the algorithm on some benchmark multi-objective optimization problems, and find very good and stable results.

Keywords: multi-objective, optimization, stochastic, probability, algorithm.

TÓM TẮT

Một giải thuật tìm kiếm được điều khiển theo xác suất giải bài toán tối ưu đa mục tiêu

Bài này đề nghị một giải thuật xác suất mới để giải bài toán tối ưu đa mục tiêu, giải thuật tìm kiếm được điều khiển theo xác suất. Giải thuật sử dụng các xác suất để điều khiển quá trình tìm kiếm các lời giải tối ưu Pareto. Đặc biệt, chúng tôi sử dụng Chuỗi Markov hội tụ để thảo luận về tính hội tụ của giải thuật. Chúng tôi thử nghiệm hướng tiếp cận này trên các bài toán tối ưu đa mục tiêu chuẩn và chúng tôi đã tìm được các kết quả rất tốt và ổn định.

Từ khóa: tối ưu, đa mục tiêu, ngẫu nhiên, xác suất, giải thuật.

1. Introduction

We introduce the Search via Probability (SVP) algorithm for solving single-objective optimization problems [4]. In this paper, we extend SVP algorithm into Probabilistic-Driven Search (PDS) algorithm for solving multi-objective optimization problems by replacing the normal order with the Pareto one. We compute the complexity of the Changing Technique of the algorithm. Especially, we use the absorbing Markov Chain to argue the convergence of the Changing Technique of the algorithm. We test this approach by implementing the algorithm on some benchmark multi-objective optimization problems, and find very good and stable results.

2. The model of Multi-objective optimization problem

A general multi-objective problem can be described as follows:

[*] MSc., HCMC University of Education
[**] Asso/Prof. Dr, HCMC University of Education

$$\begin{aligned}
&\textit{Minimize} && f_k(x) \quad (k = 1,\ldots,s)\\
&\textit{subject to} && g_j(x) \le 0 \quad (j = 1,\ldots,r)\\
&\textit{where}\ \ x = (x_i),\ a_i \le x_i \le b_i\,(a_i, b_i \in R,\ 1 \le i \le n).
\end{aligned}$$

A solution x is said to dominate a solution y if

$$f_k(x) \le f_k(y), \forall k \in \{1,...,s\}\ \textit{and}\ f_i(x) < f_i(y)\ \textit{for at least one}\ i \in \{1,...,s\}$$

A solution that is not dominated by any other solutions is called a Pareto optimization solution. Let S be a set of Pareto optimization solutions, S is called Pareto optimal set. The set of objective function values corresponding to the variables of S is called Pareto front.

3. The Changing Technique of PDS algorithm and its complexity

We consider a class of optimization problems having the character as follows: there is a fixed number k ($1 \le k < n$) that is independent of the size n of the problem such that if we only need to change values of k variables then it has the ability to find a better solution than the current one, let us call it O_k. We suppose that every variable x_i ($1 \le i \le n$) has m digits that are listed from left to right $x_{i1}, x_{i2}, \ldots, x_{im}$ (x_{ij} is an integer, $0 \le x_{ij} \le 9$, $1 \le j \le m$). Let L be a number of iterations for finding correct values of j-th digits. The Changing Technique which changes a solution x into a new solution y is described with general steps as follows:

```
Input: a solution x
Output: a new solution y
S1. j←1 (determine j-th digit);
S2. i←1 (start counting variable of the loop);
S3. y←x;
S4. Randomly select k variables of solution y and randomly change values of j-th digits of these k variables;
S5. If (x is dominated by y) then x←y;
S6. If (i<L) then i← i+1 and return S3;
S7. If (j<m) then j← j+1 and return S2;
S8. The end of the Changing Technique;
```

The Changing Technique finds the value of each digit from left digit to right digit one by one. Consider j-th digit, on each iteration the Changing Technique randomly selects k variables, and randomly changes values of j-th digits of these k variables to find a better solution than the current one. Let A be the event such that the technique can find correct values of j-th digits of k variables on each iteration. The probability of A is

$$p_A = \Pr(A) = \left(\frac{k}{n} \times \frac{1}{10}\right)^k = \frac{k^k}{10^k n^k}$$

Let X be a number of occurrences of A with N of iterations. X has the binomial distribution B (N, p_A) and the probability mass function as follows:

$$\Pr(X = x) = C_N^x (p_A)^x (1 - p_A)^{N-x} \quad (x = 0,1,\dots,N)$$

Because N is sufficiently large and p_A is sufficiently small, the Poisson distribution can be used as an approximation to B (N, p_A) of the binomial distribution as follows:

$$\Pr(X = x) \approx \frac{\lambda^x}{x!} e^{-\lambda}$$

with the parameter $\lambda=Np_A$ and the expected value of X is E(X) = λ.

Because the solution has n variables, we select an average number of iterations such that the event A occurs at least n/k times. We have

$$E(X) > \frac{n}{k} \Rightarrow N.p_A > \frac{n}{k} \Rightarrow N > \frac{n}{k.p_A} = \frac{10^k n^{k+1}}{k^{k+1}}$$

Because every variable has m digits, so the average number of iterations for finding correct values of a solution the first time is

$$m\left(\frac{10^k}{k^{k+1}}\right)n^{k+1} = \left(\frac{10^k m}{k^{k+1}}\right)n^{k+1}$$

On each iteration, the technique performs k jobs with complexity O(1). Because k is a fixed number, so the complexity of the Changing Technique is $O(n^{k+1})$.

4. The absorbing Markov Chain of the Changing Technique

Without loss of generality we suppose that the solution has n variables, every variable has m=5 digits. Let E_0 be the starting state of a solution that is randomly selected, E_i (i=1,2,…,5) be the state of the solution having n variables with correct values that are found for digits from 1-th digit to i-th digit ($1 \le i \le 5$). We have (X_n; n=0,1,2,…) is a Markov chain with states {E_0, E_1, E_2, E_3, E_4, E_5}. Let p be the probability of event in which i-th digits of n variables have correct values. According to section 2, we have

$$p = \frac{k^{k+1}}{10^k n^{k+1}}$$

Set q=1-p, the transition matrix is then

$$P = \left(p_{ij}\right)_{i,j=\overline{0,5}} = \begin{bmatrix} q & p & 0 & 0 & 0 & 0 \\ 0 & q & p & 0 & 0 & 0 \\ 0 & 0 & q & p & 0 & 0 \\ 0 & 0 & 0 & q & p & 0 \\ 0 & 0 & 0 & 0 & q & p \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The states E_i ($0 \le i \le 4$) are transient states, and the state E_5 is an absorbing state. The Markov chain is an absorbing Markov Chain and its model as follows:

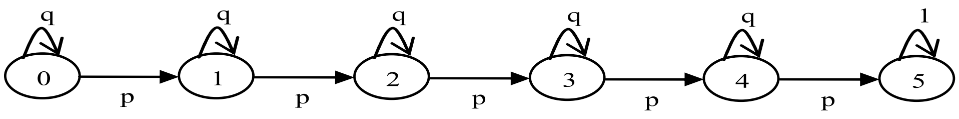

Figure 1. *The model of absorbing Markov Chain of the Changing Technique*

Absorption Probabilities: Let u_{i5} ($0 \le i \le 4$) be the probability that the absorbing chain will be absorbed in the absorbing state E_5 if it starts in the transient state E_i ($0 \le i \le 4$). If we compute u_{i5} in term of the possibilities on the outcome of the first step, then we have the equations

$$u_{i5} = p_{i5} + \sum_{j=0}^{4} p_{ij} u_{j5} \quad (0 \le i \le 4) \;\; \Rightarrow \;\; u_{05} = u_{15} = u_{25} = u_{35} = u_{45} = 1$$

Here the result tells us that, starting from state i ($0 \le i \le 4$), there is probability 1 of absorption in state E_5.

Time to Absorption: Let t_i be the expected number of steps before the chain is absorbed in the absorbing state E_5, given that the chain starts in state E_i ($0 \le i \le 4$). Then we have results as follows: $t_i = (5-i)/p$ ($0 \le i \le 4$).

5. PDS algorithm for solving multi-objective optimization problems

5.1. The Changing Procedure

Without loss of generality we suppose that a solution of the problem has n variables, every variable has m=5 digits. We use the Changing Technique of section 3 and increase the speed of convergence by using two sets of probabilities [4] to create the Changing Procedure. Two sets of probabilities [4] are described as follows:

- The changing probabilities q=(0.46, 0.52, 0.61, 0.75, 1) of digits of a variable are increasing from left to right. This means that left digits are more stable than right digits, and right digits change more than left digits. In other words, the role of left digit x_{ij} is more important than the role of right digit $x_{i,j+1}$ ($1 \le j \le m-1$) for evaluating objective functions.
- The probabilities ($r_1=0.5$, $r_2=0.25$, $r_3=0.25$) for selecting values of a digit. r_1: the probability of choosing a random integer number between 0 and 9 for j-th digit, r_2: the probability of j-th digit incremented by one or a certain value (+1,…,+5), r_3: the probability of j-th digit decremented by one or a certain value (-1,…,-5).

We use a function *random (num)* that returns a random number between 0 and (num-1). The Changing Procedure which changes values of a solution x via probability to create a new solution y is described as follows:

The Changing Procedure

Input: a solution x

Output: a new solution y

S1. $y \leftarrow x$;

S2. Randomly select k variables of solution y and call these variables y_i ($1 \leq i \leq k$). Let x_{ij} be j-th digit ($1 \leq j \leq m$) of variable x_i. The technique which changes values of these variables is described as follows:

```
For i=1 to k do
   Begin_1
      $y_i=0$;
      For j=1 to m do
         Begin_2
            If j=1 then b=0 else b=10;
            If (probability of a random event is $q_j$ ) then
               If (probability of a random event is $r_1$) then $y_i=b*y_i+random(10)$;
               Else
                  If (probability of a random event is $r_2$) then $y_i= b*y_i+( x_{ij} -1)$;
              Else $y_i= b*y_i+( x_{ij} +1)$;
            Else $y_i= b*y_i +x_{ij}$;
         End_2
      If ($y_i<a_i$) then $y_i=a_i$;   If ($y_i>b_i$) then $y_i=b_i$;
   End_1;
```

S3. Return y;

S4. The end of the Changing Procedure;

The Changing Procedure has the following characteristics:

- The central idea of PDS algorithm is that variables of an optimization problem are separated into discrete digits, and then they are changed with the guide of probabilities and combined to a new solution.
- Because the role of left digits is more important than the role of right digits for evaluating objective functions. The algorithm finds values of each digit from left digits to right digits of every variable with the guide of probabilities, and the newly-found values may be better than the current ones (according to probabilities).

- The parameter k: In practice, we do not know the true value of k for each problem. According to statistics of many experiments, the best thing is to use k in the ratio as follows:
    - $n \geq 5$, k is an integer selected at random from 1 to 5.
    - $n>6$, k is chosen as follows:
        - k is an integer chosen randomly from 1 to n / 2 with probability 20% (find the best peak of a hill to prepare to climb).
        - k is an integer selected at random from 1 to 4 with probability 80% (climbing the hill or carrying out the optimal number).

5.2. General steps of Probabilistic-Driven Search algorithm

We need to set three parameters S, M and L as follows:

- Let S be the set of Pareto optimal solutions to find
- Let M be a number of Pareto optimal solutions which the algorithm has the ability to find
- After generating a random feasible solution X, set L is the number so large that after repeated L times the algorithm has an ability to find a Pareto optimal solution that dominates the solution X.

The PDS algorithm for solving multi-objective optimization problems is described with general steps as follows:

```
S1. i←1 (determine i-th solution);
S2. Select a randomly generated feasible solution $X_i$;
S3. j←1 (create jth loop);
S4. Use the Changing Procedure to transform the solution $X_i$ into a new solution Y;
S5. If (Y is not feasible) then return S4.
S6. If ($X_i$ is dominated by Y) then $X_i \leftarrow Y$;
S7. If j<L then j←j+1 and return S4;
S8. Put $X_i$ on the set S;
Remove the solution in the set S which is dominated by another;
Remove overlapping solutions in the set S;
Set | S | = i;
S9. If i<M then i←i+1 and return S2;
S10. The end of PDS algorithm;
```

Remarks: After generating a random feasible solution x, the algorithm repeats L times to find a solution that dominates the solution x. Thus each of the solutions works independently of the other solutions. The changes of solutions are driven by probabilities. Every solution has an ability to change and directs its position to a point of the Pareto front.

6. Illustrative examples

In order to assess the performance of DPS algorithm, the algorithm will be benchmarked by using six optimization test cases developed by Deb et al. [1]. The problems are minimization problems with M=3 objectives.

DTLZ1:

$$f_1(x)=\frac{1}{2}x_1x_2(1+g(X_M)), f_2(x)=\frac{1}{2}x_1(1-x_2)(1+g(X_M)), f_3(x)=\frac{1}{2}(1-x_1)(1+g(X_M)),$$

$$g(X_M)=100\left[|X_M|+\sum_{x\in X_M}(x_i-0.5)^2-\cos\left(20\pi\left(x_i-0.5\right)\right)\right],$$

$$0\le x_i\le 1\ (i=1...7),\ x=\left(x_1,...,x_7\right),\ X_M=\left(x_3,...,x_7\right).$$

DTLZ2:

$$f_1(x)=\left(1+g(X_M)\right)\cos\left(x_1\pi/2\right)\cos\left(x_2\pi/2\right),$$

$$f_2(x)=\left(1+g(X_M)\right)\cos\left(x_1\pi/2\right)\sin\left(x_2\pi/2\right), f_3(x)=\left(1+g(X_M)\right)\sin\left(x_1\pi/2\right),$$

$$g(X_M)=\sum_{x\in X_M}\left(x_i-0.5\right)^2, 0\le x_i\le 1 (i=1...12), x=\left(x_1,...,x_{12}\right), X_M=\left(x_3,...,x_{12}\right).$$

DTLZ3: As DTLZ2, except the equation for g is replaced by the one from DTLZ1.

DTLZ4: As DTLZ2, except x_i is replaced by x_i^{α} where $\alpha>0$ (i=1,2)

DTLZ5: As DTLZ2, except x_2 is replaced by $\dfrac{1+2g\left(X_M\right)x_2}{2(1+g\left(X_M\right))}$

DTLZ6: As DTLZ5, except the equation for g is replaced by $\sum_{i\in X_M} x_i^{0.1}$

DTLZ7:

$$Min\ f_1(x)=x_1; Min\ f_2(x)=x_2; Min\ f_3(x)=\left(1+g(X_M)\right)h\left(f_1(x),f_2(x),g(X_M)\right)$$

$$g(X_M)=1+\frac{9}{|X_M|}\sum_{x_i\in X_M}x_i;$$

$$h\left(f_1(x),f_2(x),g\left(X_M\right)\right)=M-\sum_{i=1}^{M-1}\left[\frac{f_i(x)}{1+g(X_M)}\left(1+\sin\left(3\pi f_i(x)\right)\right)\right]$$

$$0\le x_i\le 1\ (i=1...22), x=\left(x_1,...,x_{22}\right),\ X_M=\left(x_3,...,x_{22}\right).$$

Because the memory of computer is limited, we divide the Pareto surface into four parts as follows:

- Part 1: $f_1(x)\le 0.5$ and $f_2(x)\le 0.5$

- Part 2: $f_1(x) \leq 0.5$ and $f_2(x) \geq 0.5$
- Part 3: $f_1(x) \geq 0.5$ and $f_2(x) \leq 0.5$
- Part 4: $f_1(x) \geq 0.5$ and $f_2(x) \geq 0.5$

Set L=30000 and M=700, we apply PDS algorithm to finding 700 Pareto optimal solutions for each part. We use two digits after decimal point for all problems. It takes 120 seconds to implement PDS algorithm for finding Pareto surface of each problem. Here the Pareto surfaces of illustrative examples are found by PDS algorithm.

We use two digits after decimal point for all problems. Here the Pareto fronts of illustrative examples are found by PDS algorithm.

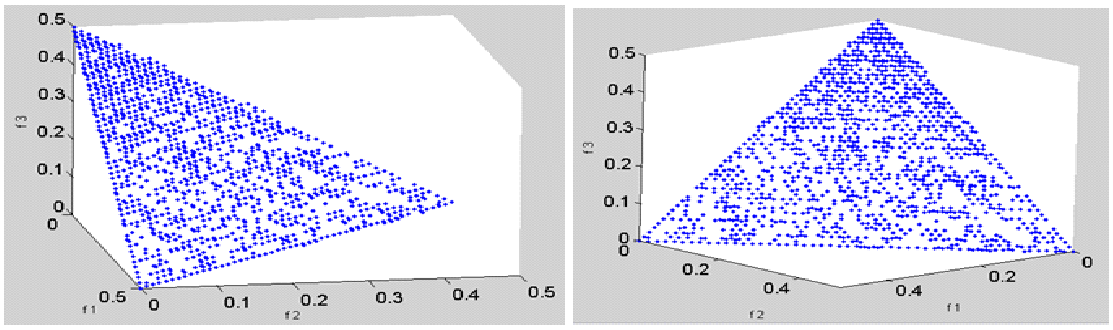

Figure 2. *DTLZ1*

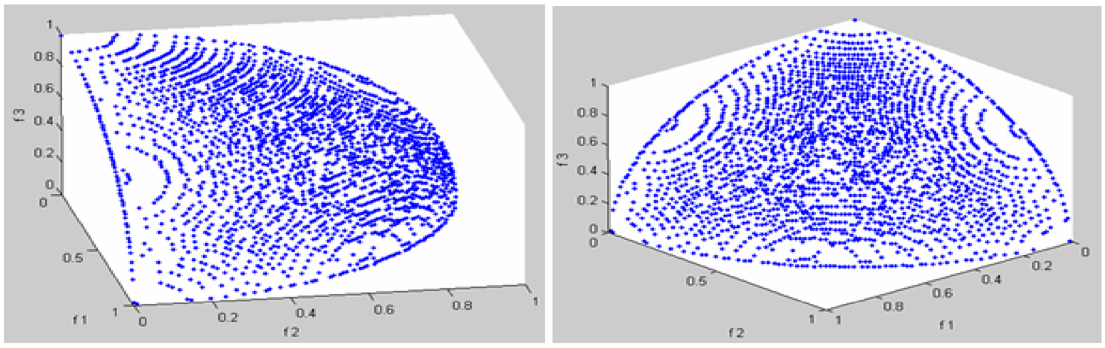

Figure 3. *DTLZ2*

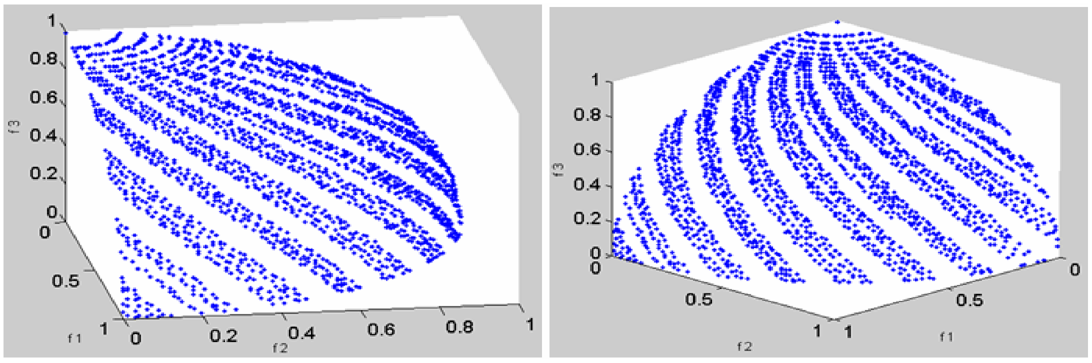

Figure 4. *DTLZ3*

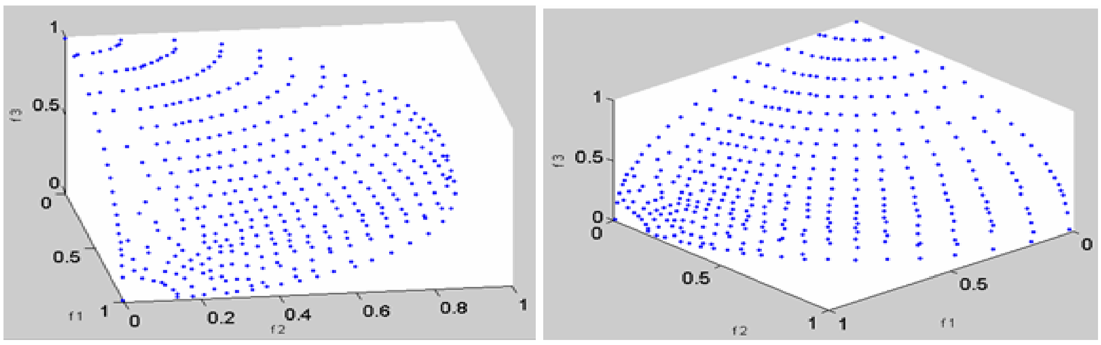

Figure 5. *DTLZ4 (α=10)*

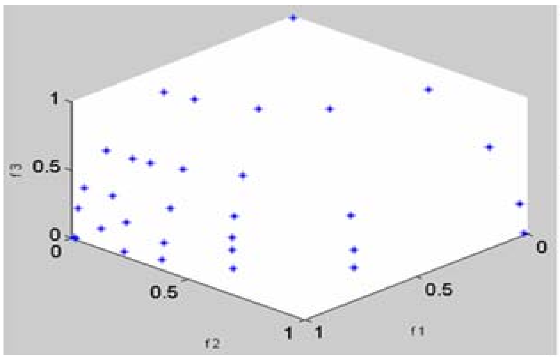

Figure 6. *DTLZ4 (α=50)*

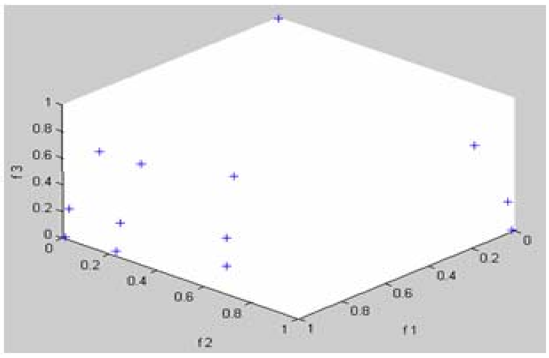

Figure 7. *DTLZ4 (α=100)*

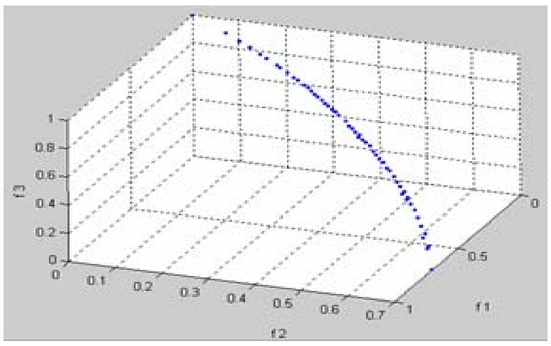

Figure 8. *DTLZ5*

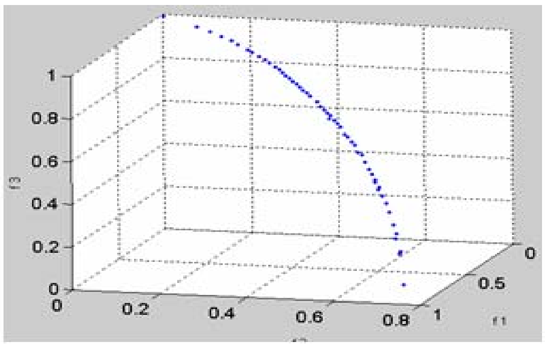

Figure 9. *DTLZ6*

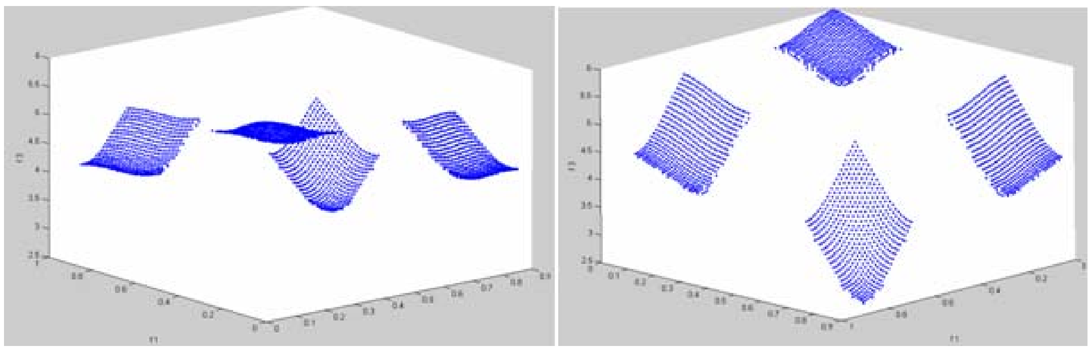

Fig. 10. *Pareto surface of DTLZ7*

Remarks:

- PDS algorithm has the ability to find a large number of Pareto optimization solutions and these solutions are able to express the concentrated regions of Pareto front.
- PDS algorithm has the ability to maintain diversity and the overall distribution of solutions on Pareto front is acceptable.

Now we use three digits after decimal point for problem DTLZ4 and the Pareto front is found by PDS algorithm with α=100 as follows:

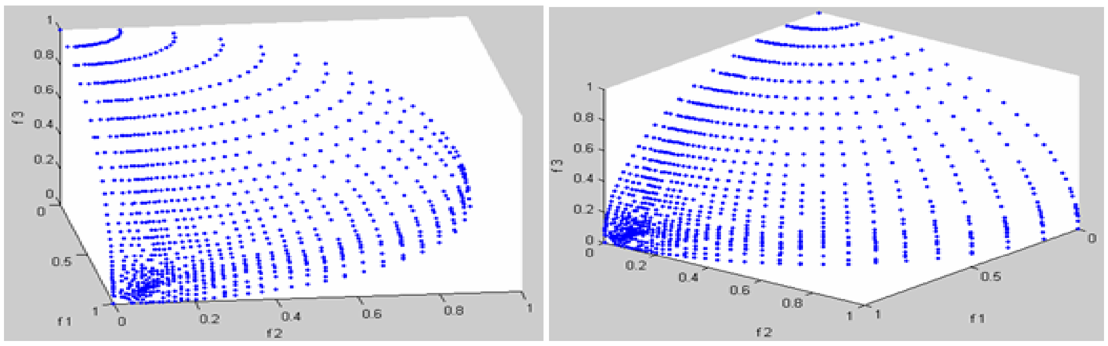

Figure 11. *DTLZ4 (α=100 with three digits after decimal point)*

7. Conclusions

In this paper, we consider a class of optimization problems having the character as follows: there is a fixed number k ($1 \le k < n$) and k is independent of the size n of the problem such that if we only need to change values of k variables then it has the ability to find a better solution than the current one, let us call it O_k. We introduce PDS algorithm for solving multi-objective optimization problems of the class O_k. We compute the complexity of the Changing Technique of PDS algorithm. Specifically, we use the absorbing Markov Chain to argue the convergence of the Changing Technique. PDS algorithm has the following advantages:

- There is no population or swarm, the algorithm is very simple and fast. The changes of solutions are driven by the probabilities and every solution has the ability to independently operate.
- The PDS algorithm has the ability to find a large number of Pareto optimization solutions and the overall distribution of solutions on Pareto front is acceptable.
- There are not many parameters to be adjusted. There is no predefined limit of objective functions and constraints, and the algorithm does not need a pre-process of objective functions and constraints.
- The parameter k: In practice, we do not know the true value of k for each problem. According to statistics of many experiments, the best thing is to use k in the ratio as follows:
    - $n \ge 5$, k is an integer selected at random from 1 to 5.

- n>6, k is chosen as follows:
    - k is an integer chosen randomly from 1 to n / 2 with probability 20% (find the best peak of a hill to prepare to climb).
    - k is an integer selected at random from 1 to 4 with probability 80% (climbing the hill or carrying out the optimal number).

In next paper, we apply PDS algorithm to solving optimization problems with equality constraints by increasing the degree of equality accuracy step by step. On the other hand, because the memory of computer is limited, we study to divide the Pareto front into several parts and apply PDS algorithm to finding a lot of solutions for each part. Especially, we apply PDS algorithm to solving multiobjective portfolio optimization problems.

REFERENCES

1. Deb K., Thiele L., Laumanns M. (2002), and Zitzler E., "Scalable Multi-Objective Optimization Test Problems". *In Congress on Evolutionary Computation (CEC 2002),* pages 825–830. IEEE Press.
2. Grinstead C. M., Snell J. L. (1997). Introduction to probability. *Published by the American Mathematical Society*.
3. Huband S., Hingston P., Barone L., and While L. (2006), "A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit". *IEEE Transactions on Evo-lutionary Computation*, 10(5):477–506.
4. Nguyễn Hữu Thông and Trần Văn Hạo (2007), "Search via Probability Algorithm for Engineering Optimization Problems", *In Proceedings of XIIth International Conference on Applied Stochastic Models and Data Analysis (ASMDA2007)*, Chania, Crete, Greece, 2007. *In book: Recent Advances in Stochastic Modeling and Data Analysis*, editor: Christos H. Skiadas, publisher: World Scientific Publishing Co Pte Ltd.
5. Zitzler E. , Deb K. , and Thiele L. (2000), "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results". *Evolutionary Computation*, 8(2):173–195.

(Received: 25/7/2011; Accepted: 09/9/2011)