A NEW SEARCH VIA PROBABILITY ALGORITHM FOR SOLVING ENGINEERING OPTIMIZATION PROBLEMS

NGUYEN HUU THONG*

ABSTRACT

In this paper, we calculate probabilities of the appearance of a better solution than the current one on each iteration, and on the performance of the algorithm we create good conditions for its appearance. We improve the algorithm of [3], [6] to search and determine the correct values for each digit from left to right of variables of a solution for solving numerical optimization problems with long narrow feasible domains.

***Keyworks**:* Optimization, Probability, Stochastic, Random, Algorithm.

TÓM TẮT

Một giải thuật tìm kiếm theo xác suất mới giải bài toán tối ưu kĩ thuật

Trong bài báo này chúng tôi tính các xác suất của việc xuất hiện các lời giải tốt hơn lời giải hiện hành trong mỗi lần lặp, và trong khi thực thi giải thuật chúng tôi tạo điều kiện tốt để các lời giải này xuất hiện. Chúng tôi đã phát triển giải thuật của [3], [6] để tìm kiếm và xác định các giá trị đúng của mỗi chữ số từ trái sang phải của các biến của một lời giải để giải các bài toán tối ưu số có miền khả thi hẹp và dài.

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

1. Introduction

The random computing algorithm was invented to solve optimization problems with very large-scale feasible domain. However, it is not efficient in sovling the optimization problems with long and narrow feasible domain because of the two following reasons:

- Ability to be randomly generated from the initial feasible solution of the random algorithm is very poor. Actually the feasible solution does not even arise.
- The ability to generate a feasible solution from the previous feasible solution is very low; which causes the convergent speed to decrease rapidly when the optimization process reaches a certain solution close to the optimal solution.

The Search Algorithms have been introduced in the paper [3], [6] under the name 'Search via Probability Algorithm'. These optimization techniques converge very fast and are very efficient for solving optimization problems with very large-scale feasible domains. But these optimization techniques are not effective in solving the numerical optimization problems with long narrow feasible domains. In this paper, we calculate probabilities of the appearance of a better solution than the current one on each iteration, and on the performance of the algorithm we create good conditions for its appearance. We improve the algorithm of [3], [6] to search and determine the correct

* MSc., HCMC University of Education

values for each digit from left to right of variables of a solution for solving numerical optimization problems with long narrow feasible domains

This paper introduces a new approach for solving numerical optimization problems. We rely on the following remarks:

Remark 1.

When we calculate a numerical optimization problem by the type of random computing, in certain iteration with the current solution, there may be some variables that have values very close to or as precise as the values of some variables of an optimal solution. So we only need to change and find the correct values for a number of remaining variables, it is sufficient to have a new solution better than the current one. The change of all n variables often has benefits in the first time to look for a good position and to avoid a local optimum. So when the current solution is in the vicinity of an optimal solution, it is often difficult to create new and better solutions with the techniques that change the values of all n variables, which causes the convergence speed of the current solution to the optimal solution to decrease rapidly.

Remark 2.

The role of left digit is more important than the role of right digit of the same variable in computing the value of the objective function. The evolutionary algorithms do not focus in the role of each digit in a variable of the solution, which means that the roles of the digits in a variable are the same. It also means that the evolutionary algorithms ignore the important role of the left digits, even though the left digits have the role to create the preconditions for the algorithms that are able to find good values for the right digits.

Remark 3.

Because the values of right digits depend on the values of left digits in evaluating the value of an objective function, the algorithm can only search the good values of right digits after the good values of left digits have been found by the algorithm. The algorithm in turn performs the following tasks:

- Step 1: If the values of left digits are not good enough, the algorithm must search for the better values of left digits.
- Step 2: If the values of left digits are good enough, then the algorithm must keep the current values of left digits and search for the good values of right digits.

However the algorithm does not know when to start Step 1 or Step 2, so it uses the probability to control the execution of two jobs that are mentioned.

Remark 4.

We consider a class of optimization problems with n variables and that class has the following properties: In each iteration of the algorithm exists a number k_0 ($1 \le k_0 < n$) which does not depend on the size n of the problem, and the algorithm just selects k ($1 \le k \le k_0$) variables to change their values then it is possible to find a better solution

than the current one. **Notice that in each iteration of the algorithm, the value of k changes and depends on the current solution**. In particular, the optimization problem for a variable (n = 1), we choose k = 1.

Remark 5.

We do not know the value of k in each iteration of the algorithm, so we should apply probability to search for the optimal solutions. We can speed up the implementation of the algorithm by computing the probabilities for controlling the algorithm and select the values of digits when changes are made.

We can use more than one set of probabilities and it is important to note that the probabilistic sets have the following characteristics simultaneously:

- It is possible to search the values of leftmost digits.
- It is possible to keep the values of left digits in search of the values of right digits.

2. The model of single-objective optimization problem

We consider two models of single-objective optimization problem as follows:

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

or

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

Without loss of generality, we design an algorithm to solve the problem (I), the problem (II) is a similar calculation by reversing the inequality signs.

3. The sets of probabilities of the algorithm

3.1. The sets of probabilities for controlling the algorithm

- *The set of probabilities for finding the good value of leftmost digits*

However, many problems have the constraints that are too tight and their feasible domains are very narrow, so the feasible solutions can only be generated if two, three or four leftmost digits are determined simultaneously. The following table shows the probabilities for finding the values of many leftmost digits simultaneously.

Table 1. *Probabilities for finding the values of many leftmost digits simultaneou*sly

Searching for leftmost digits	first digit	Second digit	third digit	fourth digit	fifth digit	sixth digit	seventh digit
probability of searching for 7 digits	1	1	1	1	1	1	1
probability of searching for many leftmost digits	1/2	1	1	1	1	1	1
	1/2	1/2	1	1	1	1	1

or a next digit	1/2	1/2	1/2	1	1	1	1
	1/2	1/2	1/2	1/2	1	1	1
	1/2	1/2	1/2	1/2	1/2	1	1
	1/2	1/2	1/2	1/2	1/2	1/2	1
Average probabilities	0.57	0.64	0.71	0.79	0.86	0.93	1

- *The set of probabilities for finding the good value of right digits*

According to the results of the paper [3][6], we have the set of probabilities for finding the good value of middle digits as follows:

(0.37, 0.41, 0.46, 0.52, 0.61, 0.75, 1)

We now provide a summary of the sets of probabilities. We have three sets of probabilities for controlling the algorithm as follows:

1. The set of probabilities for finding the good value of leftmost digits:

(0.57, 0.64, 0.71, 0.79, 0.86, 0.93, 1)

2. The set of probabilities for finding the good value of right digits:

(0.37, 0.41, 0.46, 0.52, 0.61, 0.75, 1)

According to some tests, the best rate is 30% for using the first set, 40% for the second set.

3.2. The set of probabilities for selecting the value of digits

According to the results of the paper [3][6], we have the set of probabilities for selecting the value of digits as follows:

- r_1=0.5: Probability of selecting a random integer from 0 to 9
- r_2=r_3=0.25: Probability of increasing or decreasing a random integer from 1 to 5

4. The expanded search via probability (ESVP) algorithm

4.1. The Changing Procedure

Without loss of generality we suppose that a solution of the problem has n variables, every variable has m digits, one digit is displayed to the left of the decimal point and m-1 digits are displayed to the right of the decimal point. We use a function *random (num)* that returns a random number between 0 and (num-1). The Changing Procedure that changes the value of a solution x under the control of 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←x;

S2. Select one of the following three cases according to the probabilities (0.3, 0.7) and set the probabilities (P_1, P_2, P_3, P_4, P_5, P_6, P_7) with the following values:

Case 1: (0.57, 0.64, 0.71, 0.79, 0.86, 0.93, 1)

Case 2: (0.37, 0.41, 0.46, 0.52, 0.61, 0.75, 1)

S3. Select randomly k variables of solution y and call these variables y_i ($1 \le i \le k$). Let E1 and E2 be the random events that are independent of each other. The technique for changing 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<4) then a=random(3) else a=random(5);
If (the probability of a random event E1 is P_j) then
If (the probability of a random event E2 is r_1) then $y_i=y_i+random(10)*10^{1-j}$;
Else
If (the probability of a random event E2 is r_2) then $y_i= y_i+(x_{ij} +a)*10^{1-j}$;
Else $y_i= b*y_i+(x_{ij} -a)*10^{1-j}$;
Else $y_i= y_i +x_{ij}*10^{1-j}$;
End_2;
If ($y_i<a_i$) then $y_i=a_i$; If ($y_i>b_i$) then $y_i=b_i$;
End_1;
S4. Return y;
S5. The end of Changing Procedure;

4.2. The general steps of the expanded search via probability algorithm

We apply the Changing Procedure to build the algorithm for solving single-objective optimization problems. The algorithm is described with general steps as follows:

S1. Select a random feasible solution x;
S2. Use the Changing Procedure to transform the solution x into a new solution y;
S3. If y is not a feasible solution then return S2;
S4. If $f(y) \le f(x)$ then $x \leftarrow y$;
S5. If the condition of stop is not satisfied then return S2;
S6. The end of the algorithm;

Stop condition is the desired value of the objective function or after a number of certain iterations.

4.3. The characteristic of the expanded search via probability algorithm

The Changing Procedure has the following characteristics:

- The central idea of the Changing Procedure is that variables of the solution x are separated into discrete digits, and then the values of these digits are changed with the guide of probabilities and combined to a new solution y.
- Because the role of left digit is more important than the role of right digit for assessing values of objective functions, the Procedure 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 <10, k is an integer chosen randomly from 1 to 5.
- n>10, k is chosen as follows: k is an integer chosen randomly from 1 to n / 2 with probability of 20% (finding the best peak of a hill to prepare to climb), k is an integer chosen randomly from 1 to 5 with probability of 80% (climbing the hill or optimizing the solution).

5. Examples

Using PC, Celeron CPU 2.20 GHz, Borland C++ 3.1. We performed 30 independent runs for each example. The results for all test problems are reported in Tables. The following test problems have no real-time data as shown in papers [1], [2], [4].

5.1. Test Problem 1: three-bar truss design

Minimize

$$f(x) = (2\sqrt{2}x_1 + x_2) \times l$$

subject to

$$\begin{aligned} g_1(x) &= \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} p - \sigma \le 0; \\ g_2(x) &= \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} p - \sigma \le 0; \\ g_3(x) &= \frac{1}{x_1 + \sqrt{2}x_2} p - \sigma \le 0; \\ 0 \le x_1 \le 1;\ 0 \le x_2 \le 1; \\ l = 100\,cm;\ P = 2\,KN/cm^2;\ \sigma = 2\,KN/cm^2; \end{aligned} \qquad \text{(III)}$$

T. Ray and K. M. Liew [4] used an optimization algorithm based on the simulation of social behaviour for solving the problem.

Table 2. *The best solutions found for Three-Bar Truss Design problem*

	ESVP algorithm	**Ray & Liew [4]**
x_1	0.7886753505	0.7886210370
x_2	0.4082476798	0.4084013340
$g_1(x)$	-0.0000000000059739	-0.0000000082754600
$g_2(x)$	-1.4641023093510375	-1.4639276500479634
$g_3(x)$	-0.5358976906549365	-0.5360723582274965
f(x)	263.8958433772909300	263.8958466196

However just using 6 digits after the decimal point, ESVP algorithm found a solution that was better than the solution of [4] as follows:

x=(0.788706, 0.408161)

$g_1(x)$=-0.0000000023386963;

$g_2(x)$=-1.4642008532399351;

$g_3(x)$=-0.5357991490987614;

f(x)=263.8958443850068530;

Table 3. *Statistics of 30 times by running ESVP algorithm for Three-Bar Truss Design problem (running time is 3 seconds for each experiment)*

Min	263.8958433772909300
Max	263.8958433772909300
Average	263.8958433772909300
Median	263.8958433772909300
Standard deviation	0

5.2. Test Problem 2: design of a hydrostatic thrust bearing

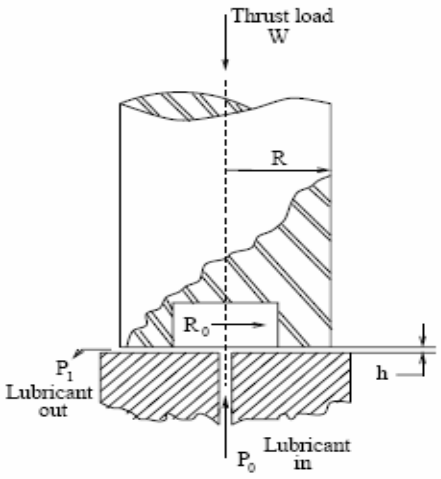

Figure 1. *The hydrostatic thrust bearing*

The following paragraph that describes hydrostatic thrust bearing problem is obtained from [1][2]:

In this problem we want to minimize the power loss during the operation of a hydrostatic thrust bearing which has to withstand a specified load while providing an axial support. Four design variables are considered: R (bearing step radius), R0 (recess radius), μ (viscosity), and Q (flow rate). The optimization problem can be stated as follows:

$$Minimize \quad f(x) = \frac{\left(\frac{QP_0}{0.7} + E_f\right)}{12.0}$$

subject to

$$g_1(x) = \frac{\pi P_0}{2} \times \frac{R^2 - R_0^2}{\ln\left(R/R_0\right)} - W_s \geq 0; \quad g_2(x) = P_{\max} - P_0 \geq 0;$$

$$g_3(x) = \Delta T_{\max} - \Delta T \geq 0; \quad g_4(x) = h - h_{\min} - P_0 \geq 0;$$

$$g_5(x) = R - R_0 \geq 0; \quad g_6(x) = 0.001 - \frac{\gamma}{gP_0}\left(\frac{Q}{2\pi Rh}\right)^2 \geq 0;$$

$$g_7(x) = 5000 - \frac{W}{\pi(R^2 - R_0^2)} \geq 0; \qquad \text{(IV)}$$

$$P_0 = \frac{6\mu Q}{\pi h^3}\ln\frac{R}{R_0}; \quad W = \left(\frac{PI \times P_0}{2.0}\right) \times \left(\frac{R^2 - R_0^2}{\ln\left(R/R_0\right)}\right);$$

$$E_f = 93336.0 Q\gamma C\Delta T; \quad \gamma = 0.0307\, lb/in^3 (849.5755\, kg/m^3);$$

$$C = 0.5\, Btu/lb^0F\,(0.5\, cal/g^0C); \quad \Delta T = 2P_2,\ P_2 = 10^{P_3} - 560;$$

$$P_3 = \frac{\log_{10}\log_{10}(8.122\times10^6\,\mu + 0.8) - C_1}{n}; \quad C_1 = 10.04;\ n = -3.55;$$

$$h = \left(\frac{2\pi N}{60}\right)^2 \frac{2\pi\mu}{E_f}\left(\frac{R^4}{4} - \frac{R_0^4}{4}\right); \quad W_s = 101000\, lb\,(45804.99\, Kg);$$

$$P_{\max} = 1000\, psi\,(6.89655\times10^6\, Pa); \quad \Delta T_{\max} = 50^0F\,(10^0C);$$

$$h_{\min} = 0.001 in\,(0.0025\, cm);\ g = 386.4\, in/seg^2\,(981.456\, cm/seg^2);\ N = 750;$$

$$1 \leq R \leq 16;\ 1 \leq R_0 \leq 16;\ 10^{-6} \leq \mu \leq 16.10^{-6};\ 1 \leq Q \leq 16.$$

Table 4. *Values of C1 and n for various grades of oil*

Oil	C_1	N
SAE5	10.85	-3.91
SAE 10	10.45	-3.72
SAE 20	10.04	-3.55
SAE 30	9.88	-3.48
SAE 40	9.83	-3.46
SA 50	9.82	-3.44

To solve this problem, there was a solution of Siddal (1982). Coello used a method to handle constraints as the objective function and the approach used multi-objective optimization based on genetic algorithm. Deb used two algorithms, a binary genetic algorithm (BGA), and a combined genetic search technique (GeneAS).

Table 5. *The solutions of the test problem in the original papers [1] [2]*

Design Variables	**Best solution found**			
	Coello	**GeneAS**	**BGA**	**Siddall**
$\mathbf{x_1(R)}$	6.271	6.778	7.077	7.155
$\mathbf{x_2(R_0)}$	12.901	6.231	6.549	6.689
$\mathbf{x_3(\mu)x10^{-6}}$	5.605	6.096	6.619	8.321
$\mathbf{x_4(Q)}$	2.938	3.809	4.849	9.168
$\mathbf{g_1(x)}$	2126.86734	8329.7681	1440.6013	-11086.7430
$\mathbf{g_2(x)}$	68.0396	177.3527	297.1495	402.4493
$\mathbf{g_3(x)}$	3.705191	10.684543	17.353800	35.057196
$\mathbf{g_4(x)}$	0.000559	0.000652	0.000891	0.001542
$\mathbf{g_5(x)}$	0.666000	0.544000	0.528000	0.466000
$\mathbf{g_6(x)}$	0.000805	0.000717	0.000624	0.000144
$\mathbf{g_7(x)}$	849.718683	83.618221	467.686527	563.644401
f(x)	**1950.2860**	**2161.4215**	**2296.2119**	**2288.2268**

Remark.

- We check the solution of Coello and there are 4 violated constraints (in bold) as follows:

g_1(x)=-100983.773300;
g_2**(x)**=999.941374;
g_3**(x)**= 3.705191;
g_4(x)= -0.073873;
g_5(x)= -6.630000;
g_6(x)= -0.000419;
g_7**(x)**=5000.040635;
f=1624.342273 (ft-lb/s);

- We check the solutions of the GeneAS, BGA, Siddall and their results are the same as the results listed in Table 5.

We use 6 digits after the decimal point of the solution x, ESVP algorithm can find a solution which has an objective-function value better than that of other authors as follows:

x=(5.955782, 5.389014, 5.365305, 2.271058)
$g_1(x)$ =0.0070179975171740;
$g_2(x)$=0.0003615996265580;

$g_3(x)$=0.0000293538797393;
$g_4(x)$=0.0003251798841270;
$g_5(x)$=0.5667680000000006;
$g_6(x)$=0.0008333632160746;
$g_7(x)$=0.0055866538696425;
f(x)=1626.4459513996814600;

Table 6. *Statistics of 30 times by running ESVP algorithm for hydrostatic thrust bearing problem (running time is 5 seconds for each experiment)*

Min	1626.445951399680
Max	1626.448425542740
Average	1626.447472035290
Median	1626.447482542920
Standard deviation	0.000994413

6. Conclusion and further development

The Search Algorithms have been introduced in the paper [3][6] under the name 'Search via Probability Algorithm'. These optimization techniques converge very fast and are very efficient for solving optimization problems with very large-scale feasible domains. But these optimization techniques are not effective in solving the numerical optimization problems with long narrow feasible domains. In this paper we have extended the algorithms of [3][6] by adding a set of probabilities for controlling the algorithm and finding the optimal solution. The algorithm does not use the population or the swarm. During each execution, the algorithm only uses a solution and changes the current solution under the guidance of probability until the algorithm finds the global optimal solution. We test the improved algorithm on some engineering optimization problems with long narrow feasible domains, and obtained better results than those found by other algorithms. The complexity of the algorithm will be presented in the next paper.

Based on the idea of SVP algorithm for solving numerical optimization problems, we can develop the algorithm for tuning parameters in the other problems or solving optimization problems with a non-numeric search objective. We consider the optimization problem and its solution with n components, the algorithm is developed based on the following rules:

• **Rule 1:** In each iteration of the algorithm, a number k (1<=k<n) is chosen randomly and the algorithm searches k components.

• **Rule 2:** The component that is most important and decisive is searched first. If we can sort the components in descending order of importance and decision, then we will search the components each in turn using that order. In case of components in the

same level for which they are not sorted in order of importance, their search potential opportunities are divided equally.

• **Rule 3:** If a component is analyzed into smaller components then the rules 1 and 2 will be applied again.

• **Rule 4:** Probabilities are applied to perform the rules mentioned above.

Finally, we can summarize the ideas of the ESVP algorithm according to the ideas of the Oriental Tao as follows: The algorithm must be soft and flexible as the water, it is sometimes as the large waves at the sea, it is sometimes as the spring water flowing through a slit in the stone. The algorithm must be reasonable and natural as the water flowing down from high to low. The algorithm does not have certain shapes, but it is frequently changed and extremely transformed.

REFERENCES

1. Coello C. A. C. (1999), "The use of a multiobjective optimization technique to handle constraints", Second International Symposium on Artificial Intelligence, track on Adaptive Systems, Edited by Alberto A. Ochoa Rodríguez, Marta R. Soto Ortiz and Roberto Santana Hermida, pp. 251-256, La Havana, Cuba.
2. Deb K., Goyal M. (1995) "Optimizing Engineering Designs Using a Combined Genetic Search", In L. J. Eshelman, editor, Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kauffman Publishers, San Mateo, California, pp. 521-528.
3. Hao T. V. and Thong N. H. (2007), "A New Stochastic Algorithm for Engineering Optimization Problems", in Proceedings of The 7th International Conference on Optimization: Techniques and Applications (ICOTA7), Kobe International Conference Center, Japan.
4. Ray T. and Liew K. M. (2003), "Society and Civilization: An Optimization Algorithm Based on the Simulation of Social Behaviour", IEEE Trans. On Evolutionary Computing, Vol 7(4), pp. 386-396.
5. Siddall J. N. (1972), "Analytical Design-Making in Engineering Design", Prentice-Hall.
6. Thong N. H. and Hao T. V. (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, May 29, 2007. In book: Recent Advances in Stochastic Modelling and Data Analysis, editor: Christos H. Skiadas, publisher: World Scientific Publishing Co Pte Ltd.

(*Received: 21/6/2012; Revised: 25/7/2012; Accepted: 24/9/2012)*