A NEW PROBABILISTIC ALGORITHM FOR SOLVING NONLINEAR EQUATIONS SYSTEMS

Thong Nguyen Huu[1] and Hao Tran Van[2]

[1] Department of Information Technology,
[2] Department of Mathematics-Information,
HCMc University of Pedagogy,
280, An Duong Vuong, Ho Chi Minh city, Viet Nam.

Abstract: In this paper, we consider a class of optimization problems having the following characteristics: there exists a fixed number k ($1 \le k < n$) which does not depend on the size n of the problem such that if we randomly change the value of k variables, it has the ability to find a new solution that is better than the current one, we call it O_k. We build a new set of probabilities for controlling changes of the values of the digits and build Probabilistic-Driven Search algorithm for solving single-objective optimization problems of the class O_k. We test this approach by implementing the algorithm on nonlinear equations systems, and we find very good results that are better than results of other authors

Key words: Optimization, Nonlinear Equations System, Probability, Algorithm.

1. Introduction

In the field of evolutionary computation, there are many popular approaches for solving optimization problems, such as genetic algorithm, particle swarm optimization,…. We have two following remarks:

1) We suppose that the solution of optimization problems has n variables. These approaches often simultaneously change values of n variables on each iteration. But in some cases, if we only need to change values of k ($1 \le k < n$) variables then it has the ability to find a better solution than the current one.
2) We suppose that every variable of the solution of optimization problems has m digits. The role of left digits is more important than the role of right digits for assessing values of objective functions, but evolutionary algorithms remove the difference of the roles of the digits.

In this paper, we build the Probabilistic-Driven Search (PDS) algorithm that overcomes the two drawbacks mentioned above for solving single-objective optimization problems. In the experiment we transform nonlinear equations systems into single-objective optimization problems and apply PDS algorithm to solving them.

2. The model of optimization problems

We consider a model of single-objective optimization problem as follows:

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

where g_j ($1 \le j \le r$) are real valued functions.

3. Probabilistic-Driven Search algorithm

We consider a class of optimization problems having the following characteristics: there exists a fixed number k ($1 \le k < n$) which does not depend on the size n of the problem such that just randomly changing the values of k variables; we may find a new solution that is better than the current one, we call it O_k. We have introduced Search Via Probably algorithm with probabilities of change (0.37, 0.41, 0.46, 0.52, 0.61, 0.75, 1) to resolve the problems of O_k [7]. But the probabilities of [7] are only relevant to the problems having no many local optimums. In this paper we build new probabilities to control changes of values of the solution and design the Probabilistic-Driven Search algorithm for solving single-objective optimization problems.

3.1 Probabilities of changes

We suppose that every variable x_i ($1 \le i \le n$) of a solution has m digits that are listed from left to right $x_{i1}, x_{i2}, \ldots, x_{im}$ ($0 \le x_{ij} \le 9$, $1 \le j \le m$). We consider j-digit of a variable x_i.

We suppose the values of left digits x_{ik} (k=1, 2, …, j-1) are correct, we have to fix the values of these left digits and change the value of j-th digit to find a correct value of j-th digit. Because the value of j-digit is changed, the values of digits x_{ik} (k=j+1,…, m) can be changed or can not be changed. Let A_j be an event such that the j-digit is selected to change its value ($1 \le j \le m$). We consider a following event to find a correct value of j-digit:

$$\overline{A_1}\,\overline{A_2}\ldots\overline{A_{j-1}}A_jB_{j+1}\ldots B_m \quad (1 \le j \le m)$$

We have following remarks:

Remark 1: The role of left digits is more important than the role of right digits of a variable for assessing values of objective functions. Hence we should find the values of digits from left digits to right digits one by one. We consider events

$$B_1B_2B_3\ldots B_m$$

where

$$B_j = A_j \text{ or } \overline{A_j} \quad (1 \le j \le m).$$

We classify these events according to typical events in the table below:

Table 1. Frequencies and probabilities of events

Event	Frequency	Probability
$A_1 B_2 B_3 \ldots B_m$	2^{m-1}	$\frac{2^{m-1}}{2^m} = \frac{1}{2}$
$\overline{A_1} A_2 B_3 \ldots B_m$	2^{m-2}	$\frac{2^{m-2}}{2^m} = \frac{1}{2^2}$
$\vdots$	$\vdots$	$\vdots$
$\overline{A_1}\,\overline{A_2} \ldots \overline{A_{m-1}} A_m$	1	$\frac{1}{2^m}$

The probability of selecting j-digit from n digit is

$$\frac{1}{2^j} \quad (1 \le j \le m)$$

We have a set of probabilities for selecting digits as follows:

$$\left(\frac{1}{2}, \frac{1}{4}, \ldots, \frac{1}{2^m}\right)$$

It means that number of searches for correct values of left digits is more than number of searches for correct values of the right digits.

Remark 2: Let p_j be the probability of the event A_j ($1 \le j \le m$). In some iteration we have a below event occurring:

$$\begin{aligned}
&\overline{A_1}\,\overline{A_2} \ldots \overline{A_{j-1}} A_j \quad (1 \le j \le m) \\
\Rightarrow\ &\Pr(\overline{A_1}) = \ldots = \Pr(\overline{A_{j-1}}) = 0; \\
&\Pr(A_j) = 1; \\
&\Pr(\overline{A_{j+1}}) = \ldots = \Pr(\overline{A_m}) = \frac{1}{2}
\end{aligned}$$

Hence we have probabilities of changes after selecting j-digit as follows:

$$p_1 = 0,\ \ldots,\ p_{j-1} = 0,\ p_j = 1,\ p_{j+1} = \frac{1}{2}, \ldots,\ p_m = \frac{1}{2}$$

Remark 3: According to papers [7], we consider two digits a_{j-1} and a_j ($2 \le j \le m$). Let r_1, r_2 and r_3 be probabilities of events below:

r_1: probability of choosing a random integer number between 0 and 9 for j-th digit.

r_2: probability of j-th digit incremented by one or a certain value (+1,…,+5).

r_3: probability of j-th digit decremented by one or a certain value (-1,…,-5).

We have the average probabilities r_1, r_2 and r_3 of both two cases as follows:

$$r_1=0.5,\ r_2=r_3=0.25$$

Probabilities of the other cases for finding correct values of three, four digits side by side are very small; hence we do not consider these cases. In next section we use three sets of probabilities above to build the changing procedure that transforms a solution x into a new solution y.

3.2 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 changing values 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 j-th digit according to probabilities

$$\left(\frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{2^m}\right)$$

S3. Set

$$p_1 = 0, \dots, p_{j-1} = 0, p_j = 1, p_{j+1} = \frac{1}{2}, \dots, p_m = \frac{1}{2}$$

S4. Select randomly k variables of solution y and call these variables y_i ($1 \le i \le k$).

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 (a random event with probability p_j occurs) then
          Begin_3
            Choose one of the following three cases according to the set of probabilities (0.5, 0.25, 0.25)
              Case 1: $y_i=y_i+random(10)*10^{1-j}$;
              Case 2: $y_i= y_i+(x_{ij} +1)*10^{1-j}$;
              Case 3: $y_i= b*y_i+(x_{ij} -1)*10^{1-j}$;
          End_3
        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;

S5. Return y and end the Changing Procedure;

The Changing Procedure has the following characteristics:

1) The central idea of the Changing Procedure is that variables of the solution x are separated into discrete digits, and then they are changed with the guide of probabilities and combined to a new solution y.
2) Because the role of left digits is more important than the role of right digits 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).
3) The parameter k: In practice, we do not know the true values of k for each problem. According to statistics of many experiments, the best thing is to use k in the ratio 50%-100% of n with $1 \le n \le 5$, 20%-80% of n with $5 \le n \le 10$, and 10%-60% of n with $10 \le n$.

3.3. Probabilistic-Driven Search algorithm

We use the Changing Procedure to build PDF algorithm for solving single-objective optimization problems. The PDS algorithm uses one solution in each execution of the algorithm, so the starting solution affects the rate of convergence of the algorithm. We improve the speed of convergence by implementing the algorithm in two phases. Phase 1: Search and select a solution that is able to optimize number the fastest. Phase 2: Optimize the solution of Phase 1 to find an optimal solution. Set M1=10 and M2=30000, PDS algorithm is described with general steps as follows:

PDS algorithm:

Phase 1: Generate randomly M1 solutions and each solution is optimized by M2 iterations, then we pick out a best solution for phase 2.

S1. Select a random feasible solution x;
S2. L1←1;
S3. Select a random feasible solution y;
S4. L2←1;
S5. Use the Changing Procedure to transform the solution y into a new solution z;
S6. If the solution z is not feasible then return S5;
S7. If f(z)<= f(y) then y←z;
S8. If L2<M2 then L2←L2+1 and return S5;
S9. If f(y)<=f(x) then x←y;
S10. If L1<M1 then L1←L1+1 and return S3;
S11. Return the solution x;

Phase 2: Numerical optimization.

S12. Use the Changing Procedure to transform the solution x into a new solution y;
S13. If y is not a feasible solution then return S12
S14. If f(y)<=f(x) then x←y;
S15. If the condition of stop is not satisfied then return S12;
S16. The end of PDF algorithm;

To cite a few instances of single-objective optimization problems, we consider system of equations and apply PDS algorithm to solving nonlinear Equations System.

4. Nonlinear Equations System

4.1. The model of nonlinear equations system

A general nonlinear equations system can be described as follows

$$\begin{cases} f_1(x_1, x_2, ..., x_n) = 0 \\ f_2(x_1, x_2, ..., x_n) = 0 \\ \quad\vdots \\ f_m(x_1, x_2, ..., x_n) = 0 \end{cases}$$

$$a_i \le x_i \le b_i,\ a_i, b_i \in R \quad (i = 1, ..., n)$$

where f_j (1≤j≤m) are nonlinear functions.

4.2. Popular approaches for solving nonlinear Equations System

There are several standard known techniques to solve nonlinear equations system. Some popular techniques are as follows: Newton-type techniques [4], trust-region method [2], Broyden method [1], secant method [3], Halley method [10]. It is to be noted that the techniques of Effati and Nazemi are only applied for two equations systems.

In the field of evolutionary computation, recently Grosan et al. [6] have transformed the system of equations into a multi-objective optimization problem as follows:

$$\begin{aligned} &Minimize \quad abs(f_1(x_1, x_2, ..., x_n)) \\ &Minimize \quad abs(f_2(x_1, x_2, ..., x_n)) \\ &\qquad\qquad \vdots \\ &Minimize \quad abs(f_n(x_1, x_2, ..., x_n)) \\ &a_i \le x_i \le b_i, a_i, b_i \in R \quad (i = 1, ..., n). \end{aligned}$$

and they use an evolutionary computation technique for solving this multi-objective optimization problem. It is to be noted that solutions found by this approach are Pareto optimal solutions.

4.3. PDS algorithm for solving Equations System

Because there are many equality constraints, the system of equations usually has no solution x such that $f_j(x)=0$ (1≤j≤m). Thus we find an approximate solution of simultaneous equations such that $|f_j(x)|<\varepsilon$ (1≤j≤m) with ε is an arbitrary small positive number. In order to do so, we transform the system of equations into a single-objective optimization problem as follows:

$$Minimize \ \varepsilon(x) = \max\left\{ |f_1(x)|, |f_2(x)|, \ldots, |f_n(x)| \right\}$$

$$x = (x_1, x_2, \ldots, x_n), a_i \le x_i \le b_i,\ a_i, b_i \in R \quad (i = 1, ..., n)$$

We use PDS algorithm to solve the single-object optimization problem. In next sections, we use two examples and six benchmark problems for nonlinear equations systems to examine the PDS algorithm. Using PC, Celeron CPU 2.20GHz, Borland

C++ 3.1. We performed 30 independent runs for each problem. The results for all test problems are reported in Tables.

5. Two examples

We considered two examples used by Effati and Nazemi [5]. PDS algorithm is compared with Newton's method, the Secant method, Broyden's method, and evolutionary approach [6]. Only systems of two equations were considered by Effati and Nazemi.

Example 1:

$$\begin{cases} f_1(x_1,x_2)=\cos(2x_1)-\cos(2x_2)-0.4=0 \\ f_2(x_1,x_2)=2(x_2-x_1)+\sin(2x_2)-\sin(2x_1)-1.2=0 \end{cases}$$

Example 2:

$$\begin{cases} f_1(x_1,x_2)=e^{x_1}+x_1x_2-1=0 \\ f_2(x_1,x_2)=\sin(x_1x_2)+x_1+x_2-1=0 \end{cases}$$

The evolutionary approach has the average running time of 5.14 seconds for example 1 and 5.09 for example 2 [6]. PDS algorithm has the running time of 5 seconds for both examples.

Table 2. Comparison of results for example 1 and example 2.

	Example 1		Example 2	
Method	**Solution**	**Functions values**	**Solution**	**Functions values**
Newton	(0.15, 0.49)	(-0.00168, 0.01497)		
Secant	(0.15, 0.49)	(-0.00168, 0.01497)		
Broyden	(0.15, 0.49)	(-0.00168, 0.01497)		
Effati	(0.1575, 0.4970)	(0.005455, 0.00739)	(0.0096, 0.9976)	(0.019223, 0.016776)
E. A. [6]	(0.15772, 0.49458)	(0.001264, 0.000969)	(-0.00138, 1.0027)	(-0.00276,-0.0000637)
PDS Alg.	(0.156520, 0.493376)	(-0.0000005815, -0.0000008892)	(0.0, 1.0)	(0, 0)

6. Six benchmark problems

Six problems of nonlinear equations systems considered in the following sections are as follows: Interval Arithmetic, Neurophysiology Application, Chemical Equilibrium Application, Kinematic kin2, Combustion Application and Economics Modeling Application.

6.1 Problem 1: Interval Arithmetic Benchmark

The Interval Arithmetic Benchmark [8] is described as follows:

$$\begin{cases} f_1(x) = x_1 - 0.25428722 - 0.18324757x_4x_3x_9 = 0; \\ f_2(x) = x_2 - 0.37842197 - 0.16275449x_1x_{10}x_6 = 0; \\ f_3(x) = x_3 - 0.27162577 - 0.16955071x_1x_2x_{10} = 0; \\ f_4(x) = x_4 - 0.19807914 - 0.15585316x_7x_1x_6 = 0; \\ f_5(x) = x_5 - 0.44166728 - 0.19950920x_7x_6x_3 = 0; \\ f_6(x) = x_6 - 0.14654113 - 0.18922793x_8x_5x_{10} = 0; \\ f_7(x) = x_7 - 0.42937161 - 0.21180486x_2x_5x_8 = 0; \\ f_8(x) = x_8 - 0.07056438 - 0.17981208x_1x_7x_6 = 0; \\ f_9(x) = x_9 - 0.34504906 - 0.19612740x_{10}x_6x_8 = 0; \\ f_{10}(x) = x_{10} - 0.42651102 - 0.21466544x_4x_8x_1 = 0; \end{cases}$$
$$-2 \le x_i \le 2 \ (i = 1, ..., 10)$$

There are 8 solutions found by evolutionary approach for the Interval Aithmetic Benchmark with the average running time of 39.07 seconds [6]. We choose the solution 1 that is displayed below to compare with the solution found by PDS algorithm.

Table 3. Comparison of results for Interval Aithmetic Benchmark.

	E. A. [6]	PDS algorithn
x_1	0.046491	0.257833
x_2	0.101357	0.381097
x_3	0.084058	0.278745
x_4	-0.138846	0.200669
x_5	0.494391	0.445251
x_6	-0.076069	0.149184
x_7	0.247582	0.432010
x_8	-0.017075	0.073403
x_9	0.000367	0.345967
x_{10}	0.148112	0.427326

$f_1(x)$	-0.2077959241	-0.0000003959
$f_2(x)$	-0.2769798847	-0.0000001502
$f_3(x)$	-0.1876863213	0.0000000010
$f_4(x)$	-0.3367887114	0.0000000365
$f_5(x)$	0.0530391321	-0.0000004290
$f_6(x)$	-0.2223730535	0.0000000763
$f_7(x)$	-0.1816084752	0.0000002966
$f_8(x)$	-0.0874896386	0.0000002231
$f_9(x)$	-0.3447200367	0.0000001704
$f_{10}(x)$	-0.2784227490	-0.0000002774
$\varepsilon(x)$		0.0000004290

6.2 Problem 2: Neurophysiology Application

The Neurophysiology Application [11] is described as follows:

$$\begin{cases} f_1(x) = x_1^2 + x_3^2 - 1 = 0; \\ f_2(x) = x_2^2 + x_4^2 - 1 = 0; \\ f_3(x) = x_5x_3^3 + x_6x_4^3 - c_1 = 0; \\ f_4(x) = x_5x_1^3 + x_6x_2^3 - c_2 = 0; \\ f_5(x) = x_5x_1x_3^2 + x_6x_4^2x_2 - c_3 = 0; \\ f_6(x) = x_5x_1^2x_3 + x_6x_2^2x_4 - c_4 = 0; \end{cases}$$
$$-10 \le x_i \le 10 \ (i = 1, ..., 6)$$

The constants c_i can be randomly chosen. In our experiments, we considered $c_i = 0$ ($i= 1, \dots , 4$).

There are 12 solutions found by evolutionary approach for the Neurophysiology Application with the average running time of 28.9 seconds [6]. We choose the solution 1 of [6] that is displayed below to compare with two solutions found by PDS algorithm.

Table 4. Comparison of results for Neurophysiology Application.

	E. A. [6]	Sol. 1	Sol. 2
x_1	-0.8282192996	0.703475	0.820345
x_2	0.5446434961	0.667647	0.820345
x_3	-0.0094437659	0.710720	0.571869
x_4	0.7633676230	0.744478	0.571869
x_5	0.0199325983	0.000000	-2.689698
x_6	0.1466452805	0.000000	2.689698

$f_1(x)$	-0.3139636071	-0.0000000060	0.0000000722
$f_2(x)$	-0.1206333343	0.0000000091	0.0000000722
$f_3(x)$	0.0652332757	0.0000000000	0.0000000000
$f_4(x)$	0.0123681793	0.0000000000	0.0000000000
$f_5(x)$	0.0465408323	0.0000000000	0.0000000000
$f_6(x)$	0.0330776356	0.0000000000	0.0000000000
$\varepsilon(x)$		0.0000000091	0.0000000722

6.3 Problem 3: Chemical Equilibrium Application

The chemical equilibrium system [8] is described as follows:

$$
\begin{cases}
f_1(x) = x_1x_2 + x_1 - 3x_5 = 0; \\
f_2(x) = 2x_1x_2 + x_1 + x_2x_3^2 + R_8x_2 - Rx_5 + 2R_{10}x_2^2 + R_7x_2x_3 + R_9x_2x_4 = 0; \\
f_3(x) = 2x_2x_3^2 + 2R_5x_3^2 - 8x_5 + R_6x_3 + R_7x_2x_3 = 0; \\
f_4(x) = R_9x_2x_4 + 2x_4^2 - 4Rx_5 = 0; \\
f_5(x) = x_1(x_2+1) + R_{10}x_2^2 + x_2x_3^2 + R_8x_2 + R_5x_3^2 + x_4^2 - 1 + R_6x + R_7x_2x_3 + R_9x_2x_4 = 0;
\end{cases}
$$
$$-10 \le x_i \le 10 \;\; (i = 1, \dots, 5)$$

There are 12 solutions found by evolutionary approach for the Chemical Equilibrium Application with a running time of 32.71 seconds [6]. We choose the solution 1 of [6] that is displayed below to compare with two solutions found by PDS algorithm.

Table 5. Comparison of results for Chemical Equilibrium Application.

	E. A. [6]	Sol. 1	Sol. 2
x_1	-0.0163087544	0.011212	0.010762
x_2	0.2613604709	9.155043	9.579740
x_3	0.5981559224	0.125929	0.123221
x_4	0.8606983883	0.857346	0.857893
x_5	0.0440020125	0.036662	0.036721

$f_1(x)$	-0.1525772444	0.0038723421	0.0036961619
$f_2(x)$	-0.3712483541	-0.0038723448	-0.0036961549
$f_3(x)$	-0.0265535274	0.0038688806	0.0036932686
$f_4(x)$	-0.2784694038	0.0038717720	0.0034008286
$f_5(x)$	-0.1168649340	-0.0018247861	-0.0007101592
$\varepsilon(x)$		0.0038723448	0.0036961619

6.4 Problem 4: Kinematic Application

The kinematic application kin2 [8] describes the inverse position problem for a six-revolute-joint problem in mechanics. The equations describe a denser constraint system and are given as follows:

$$\begin{cases} f_i(x) = x_i^2 + x_{i+1}^2 - 1 = 0 \\ f_{4+i}(x) = a_{1i}x_1x_3 + a_{2i}x_1x_4 + a_{3i}x_2x_3 + a_{4i}x_2x_4 + a_{5i}x_2x_7 + a_{6i}x_5x_8 + a_{7i}x_6x_7 + a_{8i}x_6x_8 + \\ \qquad a_{9i}x_1 + a_{10i}x_2 + a_{11i}x_3 + a_{12i}x_4 + a_{13i}x_5 + a_{14i}x_6 + a_{15i}x_7 + a_{16i}x_8 + a_{17i} = 0 \\ \quad 1 \le i \le 4; \end{cases}$$
$$-10 \le x_j \le 10 \ (j = 1,\ldots,8)$$

The coefficients a_{ki}, $1 \le k \le 17$, $1 \le i \le 4$, are given in the table below:

Table 6. Coefficients a_{ki} for the kinematic application kin2.

- 0.249150680	+0.125016350	-0.635550077	+1.48947730
+1.609135400	-0.686607360	-0.115719920	+0.23062341
+0.279423430	-0.119228120	-0.666404480	+1.32810730
+1.434801600	-0.719940470	+0.110362110	-0.25864503
+0.000000000	-0.432419270	+0.290702030	+1.16517200
+0.400263840	+0.000000000	+1.258776700	-0.26908494
- 0.800527680	+0.000000000	-0.629388360	+0.53816987
+0.000000000	-0.864838550	+0.581404060	+0.58258598
+0.074052388	-0.037157270	+0.195946620	-0.20816985
- 0.083050031	+0.035436896	-1.228034200	+2.68683200
- 0.386159610	+0.085383482	+0.000000000	-0.69910317
- 0.755266030	+0.000000000	-0.079034221	+0.35744413
+0.504201680	-0.039251967	+0.026387877	+1.24991170
- 1.091628700	+0.000000000	-0.057131430	+1.46773600
+0. 000000000	-0.432419270	-1.162808100	+1.16517200
+0.049207290	+0.000000000	+1.258776700	+1.07633970
+0.049207290	+0.013873010	+2.162575000	-0.69686809

There are 10 solutions found by evolutionary approach for the Kinematic Application Kin2 with the average running time of 221.29 seconds [6]. We choose the solution 1 of [6] that is displayed below to compare with two solutions found by PDS algorithm.

Table 7. Comparison of results for Kinematic Application kin2.

	E. A. [6]	Sol. 1	Sol. 2
x_1	-0.0625820337	0.953447	0.958991
x_2	0.7777446281	-0.301560	-0.283426
x_3	-0.0503725828	0.953447	0.958991
x_4	0.3805368959	0.301561	-0.283448
x_5	-0.5592587603	0.953447	0.958984
x_6	-0.6988338865	0.010363	-0.136180
x_7	0.3963927675	0.094760	0.856105
x_8	0.0861763643	-0.099564	-0.198128

$f_1(x)$	-0.3911967825	-0.0000003846	-0.0000059644
$f_2(x)$	-0.3925758964	-0.0000003846	-0.0000059644
$f_3(x)$	-0.8526542738	0.0000002185	0.0000065068
$f_4(x)$	-0.5424213099	0.0000002185	-0.0000069190
$f_5(x)$	0.7742116224	0.0000004085	0.0000069818
$f_6(x)$	-0.3828834764	-0.0000002465	0.0000069099
$f_7(x)$	-0.7843806421	0.0000005466	-0.0000070056
$f_8(x)$	0.4655985543	-0.0000004874	0.0000060584
$\varepsilon(x)$		0.0000005466	0.0000070056

6.5. Problem 5: Combustion Application

The combustion problem for a temperature of 3000 ◦C [8] is described by the system of equations:

$$\begin{cases} f_1(x) = x_2 + 2x_6 + x_9 + 2x_{10} - 10^{-5} = 0; \\ f_2(x) = x_3 + x_8 - 3.10^{-5} = 0; \\ f_3(x) = x_1 + x_3 + 2x_5 + 2x_8 + x_9 + x_{10} - 5.10^{-5} = 0; \\ f_4(x) = x_4 + 2x_7 - 10^{-5} = 0; \\ f_5(x) = 0.5140437.10^{-7} x_5 - 2x_1^2 = 0; \\ f_6(x) = 0.1006932.10^{-6} x_6 - 2x_2^2 = 0; \\ f_7(x) = 0.7816278.10^{-15} x_7 - x_4^2 = 0; \\ f_8(x) = 0.1496236.10^{-6} x_8 - x_1 x_3 = 0; \\ f_9(x) = 0.6194411.10^{-7} x_9 - x_1 x_2 = 0; \\ f_{10}(x) = 0.2089296.10^{-14} x_{10} - x_1 x_2^2 = 0; \end{cases}$$
$$-10 \le x_i \le 10 \ (i = 1, \ldots, 10)$$

There are 8 solutions found by evolutionary approach for the Combustion Application with the average running time of 151.12 seconds [6]. We choose the solution 1 of [6] that is displayed below to compare with two solutions found by PDS algorithm.

Table 8. Comparison of results for Combustion Application.

	E. A. [6]	Sol. 1	Sol. 2
x_1	-0.0552429896	0.000353	0.000003
x_2	-0.0023377533	0.000190	0.000486
x_3	0.0455880930	-0.000537	0.242296
x_4	-0.1287029472	0.000000	0.000020
x_5	0.0539771728	0.710649	1.332765
x_6	-0.0151036079	-0.030582	1.163017
x_7	0.1063159019	0.000005	-0.000005
x_8	0.0386267592	0.000567	-0.242266
x_9	-0.1144905135	-2.905380	-2.519984
x_{10}	0.0872294353	1.483182	0.096737

$f_1(x)$	0.0274133880	0.0000000000	0.0000000000
$f_2(x)$	0.0841848522	0.0000000000	0.0000000000
$f_3(x)$	0.1482418892	0.0000000000	0.0000000000
$f_4(x)$	0.0839188566	0.0000000000	0.0000000000
$f_5(x)$	-0.0030517851	-0.0000000881	0.0000000685
$f_6(x)$	-0.0000109317	-0.0000000753	-0.0000003553
$f_7(x)$	-0.0165644486	0.0000000000	-0.0000000004
$f_8(x)$	0.0025184283	0.0000001896	-0.0000007631
$f_9(x)$	-0.0001291516	-0.0000002470	-0.0000001576
$f_{10}(x)$	0.0000003019	0.0000000000	0.0000000000
$\varepsilon(x)$		0.0000002470	0.0000007631

6.6. Problem 6: Economics Modeling Application

The Economics Modeling Application [9] is described by the following system of equations:

$$\begin{cases} f_k(x) = \left(x_k + \sum_{i=1}^{n-k-1} x_i x_{i+k} \right) x_n - c_k = 0 \left(1 \le k \le n-1\right); \\ f_n(x) = \sum_{i=1}^{n-1} x_i + 1 = 0 \end{cases}$$
$$-10 \le x_i \le 10 \ (i = 1,...,n)$$

The constants c_k ($1 \le k \le n-1$) can be randomly chosen. We choose the value 0 for the constants and the case of n=20 equations in our experiments.

There are 4 solutions found by evolutionary approach for the Economics Modeling Application with the average running time of 640.92 seconds [6]. We choose the solution 1 of [6] that is listed below to compare with three solutions found by PDS algorithm. Here the solution 1 of [6]:

x=(-0.1639324, -0.3813209, 0.2242448, -0.0755094, 0.1171098, 0.0174083, -0.0594358, -0.2218284, 0.1856304, -0.2653962, -0.3712114, -0.3440810, -0.1060168, 0.0218564, -0.2028748, 0.0533728, -0.0587111, 0.0057098, -0.0149290, -0.0004102);

$f_1(x)$=0.0000194318; $f_2(x)$=0.0000973461; $f_3(x)$=-0.0001201028; $f_4(x)$=-0.0000239671; $f_5(x)$=-0.0000561734; $f_6(x)$=-0.0000389625; $f_7(x)$=0.0000390795; $f_8(x)$=0.0000931186; $f_9(x)$=-0.0001293920; $f_{10}(x)$=0.0000501015; $f_{11}(x)$=0.0001008920; $f_{12}(x)$=0.0001601619; $f_{13}(x)$=0.0000063289; $f_{14}(x)$=-0.0000079648; $f_{15}(x)$=0.0000766372; $f_{16}(x)$=-0.0000235752; $f_{17}(x)$=0.0000221321; $f_{18}(x)$=-0.0000033461; $f_{19}(x)$=0.0000061239;$f_{20}(x)$=-0.6399149000;

There are many solutions found by PDS algorithm and we choose three typical solutions and have them reported in the table below:

Table 9. Three solutions for Economics Modeling Application found by PDS algorithm.

	Solution 1	Solution 2	Solution 3
x_1	0.611228	0.027417	5.302852
x_2	1.082497	2.996639	0.035055
x_3	6.830700	-1.462483	2.212260
x_4	-5.082635	1.065133	-0.874504
x_5	3.330180	0.869460	1.236087
x_6	1.765048	1.411347	-1.646429
x_7	-3.169329	-5.308277	0.205364
x_8	5.596410	-2.958699	3.330769
x_9	2.001166	1.490692	-4.515966
x_{10}	1.731434	-0.705740	-2.944068
x_{11}	-0.880434	0.119057	1.496767
x_{12}	-5.275206	1.112881	-0.240641
x_{13}	-2.052474	-1.354802	-1.709052
x_{14}	-9.662985	-0.423468	-1.888176
x_{15}	3.184984	2.007751	-0.873979
x_{16}	1.093321	-1.565469	1.410222
x_{17}	-0.457790	0.339037	0.178445
x_{18}	-5.270496	1.009605	-0.646229
x_{19}	3.624381	0.329919	-1.068777
x_{20}	0.000000	0.000000	0.000000
$\varepsilon(x)$	0.0000000000	0.0000000000	0.0000000000

The solutions above have $g_i(x)$=0.0 ($1 \le i \le n=20$) and 30 digits after decimal point are zero.

Table 10. Statistics of results of the objective function ε(x) in 30 trials for each problem of PDS algorithm.

	Problem 1	Problem 2	Problem 3	Problem 4	Problem 5	Problem 6
Min	0.0000004290	0.0000000091	0.0036961619	0.0000005466	0.0000002470	0
Max	0.0000004290	0.0000005529	0.0052934327	0.2580684087	0.0000376137	0
Average	0.0000004290	0.0000001870	0.0042505633	0.0443614392	0.0000126718	0
Median	0.0000004290	0.0000001084	0.0040423263	0.0052189659	0.0000091598	0
St. dev.	0	0.0000001755	0.0005960323	0.079379872	0.0000110185	0

Table 11. Comparison of the running times (second) of evolutionary approach [6] and PDF algorithm.

	Problem 1	Problem 2	Problem 3	Problem 4	Problem 5	Problem 6
Evolutionary Approach [6]	39.07	28.90	32.71	221.09	151.12	640.92
PDS algorithm	30	30	30	30	30	20

Remarks:

For each problem, solutions that are found by PDS algorithm dominate solutions of [6]. That means, solutions of [6] are dominated and NOT Pareto optimal solutions!

PDS algorithm is very efficient for solving equations systems. The algorithm has the abilities to overcome local optimal solutions and to obtain global optimal solutions.

7. Conclusions

We consider a class of optimization problems having the following characteristics: there exists a fixed number k ($1 \leq k < n$) which does not depend on the size n of the problem such that just randomly changing the values of k variables; we may find a new solution that is better than the current one, we call it the class of optimization problems O_k. We have introduced Search Via Probably algorithm with probabilities of change (0.37, 0.41, 0.46, 0.52, 0.61, 0.75, 1) to resolve the problems of O_k [7], but the probabilities of [7] are only relevant to the problems having no many local optimums. In this paper we build new probabilities to control changes of values of the solution and design the PDS algorithm for solving single-objective optimization problems. For application of PDS algorithm we transform the nonlinear equations system into a single-objective optimization problem. PDS algorithm is very efficient for solving nonlinear equations systems. PDS algorithm has the abilities to overcome local optimal solutions and to obtain global optimal solutions.

Many optimization problems have very narrow feasible domains that require the algorithm having an ability to search values of two or more consecutive digits simultaneously to find a feasible solution. We study this case and the results will be reported in the next paper. We also compare Search via Probability algorithm of papers [7] with PDS algorithm of this paper for solving engineering optimization problems.

References

1. Broyden C. G. (1965), "A class of methods for solving nonlinear simultaneous equations", Math. Comput., vol. 19, no. 92, pp. 577–593.
2. Conn A. R., Gould N. I. M. and Toint P. L. (2000), Trust-Region Methods. Philadelphia, PA: SIAM.
3. Denis J. E. and Wolkowicz H. (1993), "Least change secant methods, sizing, and shifting", SIAM J. Numer. Anal., vol. 30, pp. 1291–1314.
4. Denis J. E. (1967), "On Newton's method and nonlinear simultaneous replacements", SIAM J. Numer. Anal., vol. 4, pp. 103–108.
5. Effati S. and Nazemi A. R. (2005), "A new method for solving a system of the nonlinear equations", Appl. Math. Comput., vol. 168, no. 2, pp. 877–894.
6. Grosan C., Abraham A. (2008), "A New Approach for Solving Nonlinear Equations Systems", IEEE Transactions on Systems, Man, and Cybernetics, Part A 38(3), 698-714.
7. Thong Nguyen Huu and Hao Tran Van (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. In book: Recent Advances in Stochastic Modeling and Data Analysis, editor: Christos H. Skiadas, publisher: World Scientific Publishing Co Pte Ltd, 454 – 463.
8. Hentenryck V., McAllester D. and Kapur D. (1997), "Solving polynomial systems using a branch and prune approach", SIAM J. Numer. Anal., vol. 34, no. 2, pp. 797–827.
9. Morgan A. P. (1987), Solving Polynomial Systems Using Continuation for Scientific and Engineering Problems. Englewood Cliffs, NJ: Prentice-Hall.
10. Ortega J. M. and Rheinboldt W. C. (1970), Iterative Solution of Nonlinear Equations in Several Variables, New York: Academic.
11. Verschelde J., Verlinden P. and Cools R. (1994), "Homotopies exploiting Newton polytopes for solving sparse polynomial systems", SIAM J. Numer. Anal., vol. 31, no. 3, pp. 915–930, Jun..

:

(Received: 07/3/2011; Accepted: 29/5/2011)