Proceedings of the World Congress on Engineering and Computer Science 2015 Vol I
WCECS 2015, October 21-23, 2015, San Francisco, USA

Problem Data Based Optimization (PDBO) Algorithm for Continuous Optimization Problems

Abdulelah G. Saif, Safia Abbas, and Zaki Fayed, *Member, IAENG*

***Abstract*—Problem Data-Based Optimization (PDBO) algorithm is appeared in 2015 by Abdulelah Saif, Safia Abbas and Zaki Fayed for combinatorial optimization problems and is applied to Discrete Time, Cost and Quality Trade -off problem (DTCQTP). In this paper, Problem Data-Based Optimization (PDBO) algorithm is adapted to solve continuous optimization problems. The proposed algorithm called the PDBO-CO (PDBO for continuous optimization) is tested on few benchmark functions and on COCOMO II model coefficients by using NASA 93 Dataset. The obtained results for benchmark functions are compared with the ones obtained using IWD-CO(Intelligent Water Drops for continuous optimization) and the obtained results from the optimized COCOMO II PA model coefficients by PDBO-CO are compared with ones optimized by IWD and Genetic algorithm (GA) and with the current COCOMO II PA model coefficients. The obtained results are satisfactory, which encourage other researches in this regard.**

***Index Terms*—COCOMO II, Meta-heuristic, Numerical functions, Optimization, PDBO algorithm**

I. INTRODUCTION

PDBO algorithm is a single agent meta-heuristic algorithm that is invented for combinatorial optimization problems by applying it to DTCQTP which depends on possibility calculated from problem's data. PDBO assumes the problem is represented in the form of a graph G = (V, E), in which the set of nodes V represents the activities and modes, and the set of E represents edges that connects between activities and modes.

For optimization problems, at each iteration, PDBO selects the first node n_i then depending on the best possibility values, it moves to the next adjacent node n_k. After then, in order to increase the chance of selecting other nodes rather than node n_k in the next iteration, PDBO technique updates the $Possibility(n_i, n_k)$ to be $Npossibility(n_i,n_k)=Possibility(n_i, n_k)+(cost/\alpha)$ where $\alpha>0$. Finally, after the best iteration solution found, in order to evaporate the *Npossibilities*, PDBO considers the parameter $\beta \in [0,1]$, such that $Npossibility(n_i, n_k)= Npossibility(n_i, n_k) - \beta$, where β is the evaporation rate (reduction rate) of $Npossibility(n_i, nk)$ for virtual edge between n_i and n_k[1].

The PDBO is single agent meta-heuristic. Meta-heuristics especially nature-inspired swarm-based optimization algorithms which are being increasingly used for solving optimization problems. Several meta-heuristics are basically suitable for continuous optimization whereas the rest of them are initially defined for combinatorial optimization. Particle swarm optimization [2] and ant colony optimization [3] are among the popular meta-heuristics, which are used for optimization problems.

So far, the PDBO algorithm has been used for the Discrete Time, Cost and Quality Trade -off problem (DTCQTP). Naturally, the PDBO algorithm is appropriate for combinatorial optimization problems. In this research, the PDBO is used for continuous optimization. In a continuous optimization problem, a number of continuous variables (parameters) are needed to be obtained such that a function is minimized or maximized. Here, the proposed PDBO algorithm called the “PDBO-CO” (the PDBO algorithm for continuous optimization) encodes the real continuous variables into integer numbers. Then, the PDBO tries to optimize the given function in the integer representation. Finally, the best solution is considered as the final solution. Next section talks about PDBO-CO. For this purpose, a few benchmark functions and COCOMO II model (for software cost estimation by using NASA 93 Dataset) are utilized for testing the proposed PDBO-CO for the continuous optimization, which are given in section III. At the end, conclusion is given in section IV.

II. THE PROPOSED PDBO-CO ALGORITHM

In this section, the steps to optimize a given function by the PDBO-CO are explained. In fact, solutions are constructed with the help of a graph. The proposed PDBO-CO is shown in figure 2. The following subsections explain the components of the PDBO-CO.

A. *Problem Representation*

Given a function $f: S \rightarrow R$, *find* $X^* \in S$: $\forall X \in S$ $f(X^*) \leq f(X)$ *(minimization) or* $f(X^*) \geq f(X)$ *(maximization).* Function f is called the objective function, its domain S is called the search space, and the elements of S, are called feasible solutions. A feasible solution X is a vector of optimization variables $X = \{X_1, X_2, ..., X_n\}$. A feasible solution X*t hat minimizes/maximizes the objective function is called an optimal solution. The maximization over an objective function f is equivalent to minimization over the function –f [4].

To minimize this function by PDBO_CO, a graph of n nodes (n is number of variables) and 10 other virtual nodes

Manuscript submitted July 23, 2015; revised July 29, 2015. The authors gratefully acknowledge the support of Ain Shams University and Yemen government in supporting them.
Abdulelah Ghaleb Farhan Saif is Ph.D student at Ain Shams University, Egypt (phone: 00201154415035; abdulelah.saif1980@gmail.com).
Safia Abbas Mahmoed Abbas is lecturer at Ain Shams University, Egypt (Safia_abbas@yahoo.com).
Zaki Taha Ahmed Fayed is Emeritus Professor at Ain Shams University, Egypt (ZFayed@hotmail.com).

(digits i.e. domains) numbered from 0 to 9 which are connected to each node (variable) as in figure 1. Each of above variables is expressed by 4 digits which are chosen among 10 digits by PDBO_CO algorithm according to *minimum possibilities (Po)*. First digit is integral part of a variable and the remaining 3 are fractions part. The possibilities are placed on the edges between variables and digits as in figure 1.

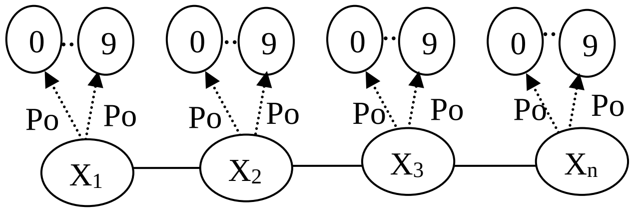

Fig. 1. Problem representation.

B. Possibilities Initialization

For each variable, $X_1,\ldots,X_n$, initialize the edges between variable node X_i and its digits nodes D_k as follow:

$$Cost(Edge(X_i, D_k))=k,\ i=1,..,n,\ k=0,..,9 \qquad (1)$$

Calculate the total cost of X_i as

$$Total\_Cost(X_i)= \sum_{k=0}^{9} \text{Cost(Edge(}X_i, D_k\text{))},\ i=,1.,.n \quad (2)$$

$$E.g.\ Total\_Cost(X_1)= \sum_{k=0}^{9} \text{Cost(Edge(}X_1, D_k\text{))}$$

$$=Cost(Edge(1,0))+ Cost(Edge(1,1))+ \ldots + Cost(Edge(1,9)).$$

$$= 0+1+\ldots.+9=45.$$

Then, calculate the possibility of choosing the digit node D_k connected to variable node X_i among others as follow:

$$Possibility(X_i, D_k)= \frac{\text{Cost(Edge(}X_i, D_k\text{))}}{\text{Total\_Cost}(X_i)}$$

,where i=1,..,n, k=0,..,9 (3)

E.g. *Possibility* $(X_1, D_0)=0/45=0.0$,
Possibility $(X_1, D_1)=1/45\approx 0.022$,
Possibility $(X_1, D_2)=2/45\approx 0.044$,
Possibility $(X_1, D_3)=3/45\approx 0.066$,
Possibility $(X_1, D_4)=4/45\approx 0.088$,
:
Possibility $(X_n, D_9)=9/45\approx 0.2$.

C. Digit Selection Mechanism

PDBO_CO starts its journey from node1 (X_1) from which selects 4 digits among 10 digits which are connected to it according to *minimum Possibility*(X_1,D_k) in order and finishes it by visiting the last node (X_n) from which selects 4 digits among 10 digits which are connected to it. This step applies for all variables, if there is an improvement in the objective function, otherwise PDBO_CO selects four digits randomly from [0,9].

The 4 digits for each variable X_i are selected as follow:

X_i= *4 digits whose Possibility*(X_i,D_k) *are the smallest* , $i=1,\ldots.,n,\ k=0,\ldots,9$ (4)

E.g. X_1= 0123(four digits); these digits are selected by PDBO_CO because 0.0, 0.022, 0.044 and 0.066 are the four smallest *Possibility*(X_1,D_k) among all in order.

Note: You can make PDBO-CO selects more than 4 digits, if you need. To obtain negative value to variable, its selected digits are multiplied by -1.

D. Updates Possibilities

PDBO_CO updates the *Possibility*(X_i,D_k) of the four selected digits for each variable X_i to be:

$$Possibility(X_i,D_k)= Possibility(X_i,D_k)+ \frac{\text{Cost(Edge(Xi, Dk))}}{\alpha} \qquad (5)$$

,where α>0 (α user selected, here α=10000).

E. Evaporate Possibilities

In this step, PDBO_CO has two choices:

1. Evaporates the *Possibility*(X_i,D_k) of the four selected digits for each variable X_i in the current iteration, if there is an improvement in the objective function in the current iteration.
2. Evaporates the *Possibility*$(X_i,D_k)$of the best selected digits for each variable X_i obtained from all iterations ,if there is no improvement in the objective function in the current iteration.

The equation used is:

$$Possibility(X_i,D_k)= Possibility(X_i,D_k)- \beta \qquad (6)$$

,where β >0 (β user selected, here β =0.00001).

1. Set α and β parameters.
2. Represent the problem in the form of graph as figure 1.
3. Determine problem dataset if exists.
4. Initialize the *Possibility*(X_i,D_k) , i=1,..,n, k=0,..,9.
5. While (termination condition not met) do
   For each variable X_i
      if there is an improvement in the objective function then
         Select 4 digits for variable X_i in the graph with Minimum Possibility in order.
      Else
         Selects 4 digits for X_i randomly from [0,9].
      End if
      Update Possibility of virtual edge between Xi and selected digit D_k by
      Possibility(X_i,D_k)= *Possibility*(X_i,D_k)+ *Cost(Edge*(X_i, D_k)*)/*α.
   End i for
6. Find iteration solution i.e. evaluate the objective function.
7. If there is an improvement in the objective function then
   evaporate the possibilities of virtual edges edge(X_i, D_k) between all variables and their selected digits at this iteration by *Possibility*(X_i,D_k)= *Possibility*(X_i,D_k)- β.
   Else
   evaporate the possibilities of virtual edges edge(X_i,D_k) between all variables and their best selected digits from all iterations by *Possibility*(X_i,D_k)= *Possibility*(X_i,D_k)- β.
8. End if
9. End while
10. Return the best solution

Fig. 2. The proposed PDBO-CO algorithm.

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

III. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed PDBO-CO, a few benchmark functions taken from [5] and COCOMO II Post Architecture model [6] are selected. The algorithm is implemented in c# and is tested and evaluated on CPU (Core(i5) 3210 M, 2.50 GHz) and 4GB RAM using Windows 7 as the operating system.

A. Benchmark Functions

The selected functions are shown in table I. For the functions f1 , f2, f3 , and f4 , the dimension of the input vectors are here selected to be ten. In contrast, the dimension of the last function f18 is originally fixed to the value of two.

TABLE I: THE BENCHMARK FUNCTIONS, WHICH ARE USED FOR TESTING THE PDBO-CO ALGORITHM

Benchmark Function	Ranges	Dim.	Minimum value (f_{min})
$f1(X)=\sum_{i=0}^{n-1} x_i^{\ 2}$	$-5.12 \le x_i \le 5.12$	$n \ge 1$	0
$f2(X)=\sum_{i=0}^{n-1} \lvert x_i \rvert + \prod_{i=0}^{n-1} \lvert x_i \rvert$	$-10 \le x_i \le 10$	$n \ge 1$	0
$f3(X)=\sum_{i=0}^{n-1}\left(\sum_{j=0}^{i} x_j\right)^2$	$-100 \le x_i \le 100$	$n \ge 1$	0
$f4(X)=\max \lvert x_i \rvert,\ 0 <= i < n$	$-100 \le x_i \le 100$	$n \ge 1$	0
$f18(X)= [1+(x_1+x_2+1)^2 (19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times [30+(2x_1-3x_2)^2 (18-32x_1+12x_1^2+84x_2-36x_1x_2+27x_2^2)]$	$-2 \le x_i \le 2$	$n=2$	3

For each function, the PDBO-CO is run five times and the results are compared with that of IWD-CO found in [7]. The results of PDBO-CO and IWD-CO are shown in table II. The PDBO-CO converges to optimal values of the five functions.

TABLE II. THE RESULTS OF THE PDBO-CO AND IWD-CO

Benchmark function	PDBO-CO			IWD-CO		
	Best value	Average value	Time (seconds)	Best value	Average value	Time (seconds)
1	0	0	00.0010000	1.28E-17	6.44E-16	68
2	0	0	00.0060000	2.33E-08	3.92E-08	68
3	0	0	00.0060004	2.09E-13	7.51E-11	65
4	0	0	00.0010001	1.00E-06	2.25E-03	60
18	3.00025190523812	3.208351	00.2960169	3.00E+00	3.0000e	22

B. COCOMO II Post Architecture model

COCOMO II PA model is one of software cost estimation methods which calculates the software development effort (in person months) by using the following equation:

$$Effort = A \times (SIZE)^E \times \prod_{i=1}^{17} EMi. \qquad (7)$$

A- multiplicative constant with value 2.94 that scales the effort according to specific project conditions. Size - Estimated size of a project in Kilo Source Lines of Code or Unadjusted Function Points. E - An exponential factor that accounts for the relative economies or diseconomies of scale encountered as a software project increases its size. EMi - Effort Multipliers. The coefficient E is determined by weighing the predefined scale factors (SFi) and summing them using following equation:

$$E = B + 0.01 \times \sum_{i=1}^{5} SFi \qquad (8)$$

The development time (TDEV) is derived from the effort according to the following equation:

$$TDEV = C \times (Effort)^F \qquad (9)$$

$$F = D + 0.002 \times \sum_{i=1}^{5} SFi \qquad (10)$$

B=0.91,D=0.28. The values of effort multipliers and scale factors used in the implementation are taken from [6].

C. Dataset Description Used To Evaluate COCOMO II PA Model

Experiments have been conducted on NASA 93 data set found in [8]. The dataset consist of 93 completed projects with its size in kilo line of code (KLOC), actual effort in person-month, development time in months .

IV. RESULT ANALYSIS

The best results of PDBO-CO, IWD[9] and GA [6] are achieved using many iterations and a solution set is received from which the best solution is chosen i.e. a solution with the best fitness function values (*Mean Magnitude of Relative Error (MMRE)* for effort and time).
The final best solution obtained for coefficients(variables) by PDBO-CO is: A= 3.734, B=1.006, C=04.02 and D=0.327.

The final best solution obtained for coefficients(variables) by IWD is: A=3.762, B=1.005, C=4.484 and D=0.288.

The final best solution obtained for coefficients(variables) by GA is: A=3.673, B=1.005, C=02.44 and D=0.342.

Current COCOMO II PA model coefficients are the following: A=2.94, B=0.91, C=3.67 and D=0.28.

The following tables, table III and table IV, show the comparison among the actual, effort and time, values and estimated, effort (person month) and time (months), values for the first ten project dataset using PDBO, IWD and GA algorithm optimized and current COCOMO II PA model coefficients with their estimated project size.

TABLE III: ESTIMATED DEVELOPMENT EFFORT VALUES

Pr. No	Project Size (KLOC)	Actual Effort (PM)	Calculated Effort (PM) using coefficients optimized by PDBO	Calculated Effort (PM) using coefficients optimized by IWD	Calculated Effort (PM) using coefficients optimized by GA	Calculated Effort (PM) using COCOMO II PA model current coefficients
1	25.9	117.6	103.0957	103.5313	102.104	59.39319
2	24.6	117.6	97.89073	98.30942	96.97911	56.67413
3	7.7	31.2	30.4278	30.59345	30.35525	19.6943
4	8.2	36	32.41586	32.59029	32.32637	20.85473
5	9.7	25.2	38.38427	38.58433	38.23973	24.29943
6	2.2	8.4	8.628555	8.686406	8.672928	6.29852
7	3.5	10.8	13.76554	13.8514	13.79784	9.610275
8	66.6	352.8	266.6097	267.4835	262.5532	140.2799
9	7.5	72	36.6467	36.84718	36.5651	23.77946
10	20	72	33.16783	33.31659	32.89979	19.58805

TABLE IV: ESTIMATED DEVELOPMENT TIME VALUES

Pr. No	Project Size (KLOC)	Actual Time (Months)	Calculated Time (Months) using coefficients optimized by PDBO	Calculated Time (Months) using coefficients optimized by IWD	Calculated Time (Months) using coefficients optimized by GA	Calculated Time (Months) using COCOMO II PA model current coefficients
1	25.9	15.3	18.30441	17.06107	26.35858	10.5749
2	24.6	15	17.99694	16.80866	25.89975	10.43705
3	7.7	10.1	12.28162	12.00955	17.4293	7.76328
4	8.2	10.4	12.53845	12.23024	17.80727	7.888731
5	9.7	11	13.25086	12.83961	18.85714	8.233734
6	2.2	6.6	8.133506	8.357022	11.3698	5.641787
7	3.5	7.8	9.475704	9.559073	13.3203	6.350325
8	66.6	21	24.97396	22.42472	36.37403	13.45205
9	7.5	13.6	13.05165	12.67039	18.57138	8.184016
10	20	14.4	12.63283	12.30812	17.91435	7.75153

The graphical comparison among effort values and among time values described in table III and table IV ,respectively is shown in figure 3 and figure 4 respectively.

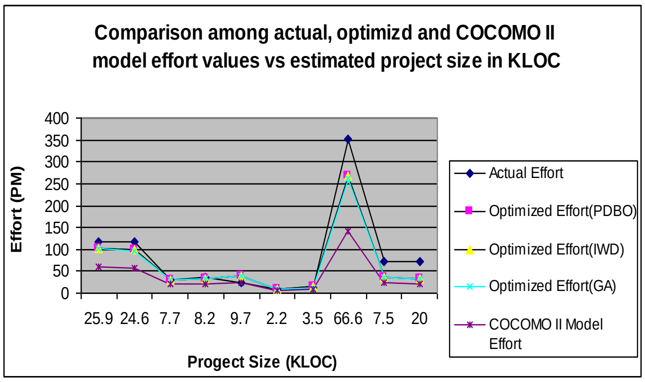

Fig. 3. Comparison among Effort values vs. Size.

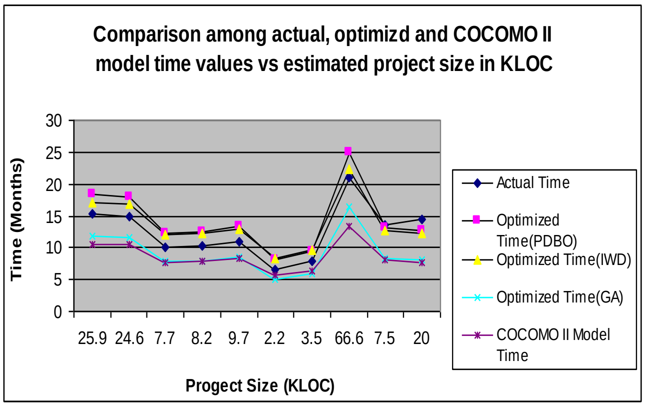

Fig. 4. Comparison among Time values vs. Size.

Table V and figure 5 compare the *MMRE (Mean Magnitude of Relative Error) and PRED (.25)* which show the performance of PDBO,IWD, GA and COCOMO II PA model in estimating the effort and time for the whole dataset.

$$MMRE = 1/n * \sum_{j=1}^{n} |\, Actual - Estimated \,| \,/\, Actual \quad (11)$$

$$MREj = |\, Actualj - Estimatedj \,| \,/\, Actualj \quad (12)$$

$$PRED\ (p) = k \,/\, n \quad (13)$$

k is the number of projects where MRE is less than or equal to p, and n is the total number of projects.

TABLE V: PERFORMANCE MEASURE COMPARISON

Results	PDBO	IWD	GA	COCOMO II
MMRE for Effort	0.474929	0.474806	0.4752	0.6
MMRE for Time	0.093497	0.095901	0.092301	0.43
PRED (.25) for Effort	0.419355	0.419355	0.387097	0.09
PRED (.25) for Time	0.989247	0.95	1	0.06

ISBN: 978-988-19253-6-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

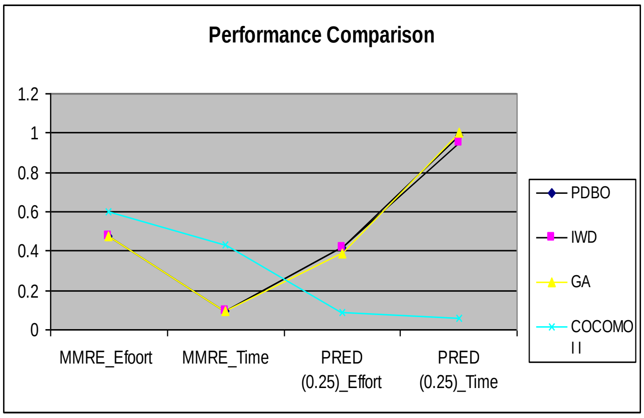

Fig. 5. Performance measure comparison.

From table V and figure 5, the *MMRE* of PDBO, IWD and GA for effort and time is equal and lower than that of COCOMO II PA model ,whereas *PRED(0.25)* of PDBO and of IWD for effort is equal and for time PDBO is larger than that of IWD. *PRED(0.25)* of GA for effort is smaller than that of PDBO and IWD, but is the largest for time. COCOMO II is the worst among all.

It shows clearly that optimized coefficients by PDBO, IWD and GA algorithm produces more accurate results than the old coefficients.

V. CONCLUSION

This paper adapts PDBO algorithm to solve continuous optimization problems. The proposed PDBO algorithm, PDBO-CO, is tested on few well known benchmark functions and on COCOMO II PA model coefficients by using NASA 93 dataset. The obtained results for benchmark functions are compared with the ones obtained using IWD-CO and the obtained results from the optimized COCOMO II PA model coefficients by PDBO-CO are compared with ones optimized by IWD and GA and with the current COCOMO II PA model coefficients. The obtained results are satisfactory. In the future, other coding methods may be used instead of integer numbers.

REFERENCES

[1] Abdulelah G. Saif, Safia Abbas and Zaki Fayed, “The PDBO Algorithm for Discrete Time, Cost and Quality Trade -off in Software Projects with Expressing Quality by Defects”, to be published in Proceedings of International Conference on Communication, Management and Information Technology (ICCMIT 2015).
[2] Kennedy and Eberhart, “Particle Swarm Optimization”, Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ. (1995) 1942-1948.
[3] M. Dorigo, M. Birattari and T. Stuzle, “Ant Colony Optimization, Artificial Ants as a Computational Intelligence Technique”, IEEE Computational Intelligence Magazine, November 2006.
[4] Krzysztof Socha, “Ant Colony Optimization for Continuous and Mixed-Variable Domains”.
[5] Yao, X. and Liu Y, “Fast evolutionary programming ” In L. J. Fogel, P. J. Angeline & T. Back (Eds.), Proceedings of the 5th Annual Conference on Evolutionary Programming (pp. 451–460). SanDiego, CA: MIT Press.
[6] Astha Dhiman, Chander Diwaker, “Optimization of COCOMO II Effort Estimation using Genetic Algorithm”, American International Journal of Research in Science, Technology, Engineering & Mathematics, 2013.
[7] Hamed Shah-Hosseini, “An approach to continuous optimization by the Intelligent Water Drops algorithm”, 4th International Conference of Cognitive Science (ICCS 2011).
[8] Nasa 93 dataset contains effort and defect information available at http://promisedata.googlecode.com/svn/trunk/effort/nasa93-dem/nasa93-dem.arff.
[9] Abdulelah G. Saif, Safia Abbas and Zaki Fayed, "Intelligent Water Drops (IWD) Algorithm for COCOMO II and COQUAMO Optimization", to be published.

,