Multi Objective Particle Swarm Optimization for Software Cost Estimation

Prasad Reddy P.V.G.D
Department of CS&SE,
Andhra University,
Visakhapatnam, India

Hari CH.V.M.K.
Department of IT,
Gitam University,
Visakhapatnam, India

Srinivasa Rao T.
Department of CSE,
Gitam University ,
Visakhapatnam, India

ABSTRACT

Software Project Management activities are classified as planning, monitoring-control and termination. Planning is the most important activity in project management which defines the resources required to complete the project successfully. Software Cost Estimation is the process of predicting the cost and time required to complete the project. The basic input for the software cost estimation is coding size and set of cost drivers, the output is Effort in terms of Person-Months (PM's). In this paper we proposed a model for software cost estimation using Multi Objective (MO) Particle Swarm Optimization. The parameters of model tuned by using MOPSO considering two objectives Mean Absolute Relative Error and Prediction. The COCOMO dataset is considered for testing the model. It was observed that the model gives better results when compared with the standard COCOMO model. It is also observed, when provided with enough classification among training data may give better results.

Keywords

KDLOC-thousands of delivered lines of code, PM- person months, PSO- particle swarm optimization, COCOMO-constructive cost estimation, MO- Multi Objective.

1. INTRODUCTION

Software Engineering is a systematic approach to the development, maintenance and retirement of software. The project manger is the responsible person for software development life cycle activities. The primary job of the project manager is to ensure that the project is completed with the goal: "high quality of software must be produced with low cost that is within time and budget". The responsibilities of project manger are planning, organizing, staffing, directing and controlling the activities. The first important component of software project management is effective planning of the development of the software which determines the resources required to complete the project successfully. The resources include the number and skill level of the people, and the amount of computing resources[6,8]. The cost of a software project is directly proportional to the number of people needed for the project. The problem of predicting how many people and other resources needed for a given software project known as Software Cost/Effort Estimation.

The Cost for a project is a function of many parameters. Size is a primary cost factor in most models and can be measured using lines of code (or) thousands of delivered lines of code (KDLOC) or function points. The A number of models have been evolved to establish the relation between Size and Effort for Software Effort Estimation. There are two major types of cost estimation methods: algorithmic and non-algorithmic. Algorithmic models vary widely in mathematical sophistication. Some are based on simple Arithmetic formulas using such summary statistics as means and standard deviations. Others are based on regression models and differential equations[7,20,21]. Some of the famous models are COCOMO[1], SLIM, Function Point, Price to Win and Delphi model. The parameters of the algorithms are tuned using Genetic Algorithms[17], Fuzzy models[10,19,24], Soft-Computing Techniques[9,22,25], Computational Intelligence Techniques, Heuristic Algorithms, Neural Networks[14,15], Radial Basis[23], MO Genetic Algorithm[3] and Regression. In this paper the parameters are tuned by using Multi Objective Particle Swarm Optimization for Software Effort Estimation.

2. BACKGROUND

This section discusses the COCOMO model and Multi Objective PSO for fine tuning parameters in software effort estimation.

2.1Constructive COst Model (COCOMO)

[Boehm, 1981] described COCOMO as a collection of three variants, they are Basic model, Intermediate model, and Detailed model. Boehm described three development modes and Organic is for relatively simple projects, Semidetached is for relatively intermediate projects, Embedded for a project developed under tight constraints. The **Basic COCOMO Model** computes effort E as function of program size, and it is same as single variable method [7,15,20,21]. The Effort calculated using the following equation

$$\text{Effort} = a*(\text{size})^b \quad (1)$$

Where a and b are the set of values depending on the complexity of software (for organic projects a=2.4,b=1.05,for semi-detached a=3.0,b=1.1.2 and for embedded a=3.6,b=1.2).

An **Intermediate COCOMO** model effort is E is function of program size and set of cost drivers or effort multipliers. The Effort calculated using the following equation

$$\text{Effort} = a*(\text{size})^b * \text{EAF} \quad (2)$$

where a and b are the set of values depending on the complexity of software (for organic projects a=3.2,b=1.05,for semi-detached a=3.0,b=1.1.2 and for embedded a=2.8,b=1.2) and EAF (Effort Adjustment Factor) which is calculated using 15 cost drivers. Each cost driver is rated from ordinal scale ranging from low to high. In **Detailed COCOMO** the effort E is function of program size and a set of cost drivers given according to each phase of software life cycle. The phases used in detailed COCOMO are requirements planning and product design, detailed design, code and unit test, and integration testing. The weights defined accordingly. The Effort calculated using the following equation

$$\text{Effort} = a*(\text{size})^b*\text{EAF}*\text{sum}(W_i) \quad (3)$$

Boehm and his colleagues have refined and updated COCOMO called as COCOMO II. It is a collection of three variants, Application composition model, Early design model, and Post architecture model.

2.2 Multi Objective Particle Swarm Optimization

2.2.1 Particle Swarm Optimization (PSO)

Swarm Intelligence (SI) is an innovative distributed intelligent paradigm for solving optimization problems that originally took its inspiration from the biological examples by swarming, flocking and herding phenomena in vertebrates. Particle Swarm Optimization (PSO)[Dr. Russell C. Eberhart and Dr. James Kennedy in 1995][11,12,13,16,18] incorporates swarming behaviors observed in flocks of birds, schools of fish, or swarms of bees, and even human social behavior, from which the idea is emerged. PSO is a population-based optimization tool, which could be implemented and applied easily to solve various function optimization problems, or the problems that can be transformed to function optimization problems.

PSO is a robust stochastic optimization technique based on the movement of intelligent swarms. PSO applies the concept of social interaction to problem solving. It uses a number of agents(particles) that constitutes a swarm moving around in the search space looking for the best solution. Each particle is treated as a point in an N- dimensional space which adjusts its flying according to its own flying experience (Pbest- personal best) as well as flying experience of other particles (Gbest – global best) . The basic concept of PSO lies in accelerating each particle towards its Pbest and Gbest locations with a random weighted acceleration at each time. The modifications of the particles positions can be mathematically modeled according to the following equations:

$$V^{k+1} = V_i^k + c_{1*} \text{rand}()_1 * (\text{Pbest} - S_i^k) + c_2 * \text{rand}()_2 * (\text{Gbest} - S_i^k) \quad (4)$$

$$S_i^{k+1} = S_i^k + V_i^{k+1} \quad (5)$$

Where,

S_i^k is current search point, S_i^{k+1} is modified search point., V_i^k is the current velocity , V^{k+1} is the modified velocity, V_{pbest} is the velocity based on Pbest , V_{gbest} = velocity based on Gbest, c_j is the weighting factors. rand() are uniformly distributed random numbers between 0 and 1.

In the particle swarm optimization technique, the particle searches the solutions in the problem space with a range of [-s, s]. In order to guide the particle effectively in the search space , the maximum moving distance during each iteration must be changed in between the maximum velocity [$-V_{max}$, V_{max}]. PSO variants and Applications are Binary Particle Swarm Optimizer, Standard Particle Swarm Optimizer, Particle Swarm Optimizer with Inertia, and Particle Swarm Optimizer with Constriction Coefficient.

2.2.2 Multi Objective PSO

A general single-objective optimization problem is defined as minimizing (or maximizing) f(**x**) subject to $g_i(\mathbf{x}) \leq 0$, i = {1, . . . , m}, and $h_j(\mathbf{x}) = 0$, j = {1, . . . , p} $\mathbf{x} \in \Omega$. A solution minimizes (or maximizes) the scalar f(**x**) where **x** is a n-dimensional decision variable vector **x** = (x1, . . . , xn) from some universe Ω.

Observe that $g_i(\mathbf{x}) \leq 0$ and $h_j(\mathbf{x}) = 0$ represent constraints that must be fulfilled while optimizing (minimizing or maximizing) f(**x**). Ω contains all possible **x** that can be used to satisfy an evaluation of f(**x**) and its constraints. Of course, **x** can be a vector of continuous or discrete variables as well as f being continuous or discrete [2,4,5].

Many (may be most) real-world problems involve the optimisation of two or more objectives E.g.: Minimise the cost of a product, Maximise the quality of the product, Minimise the wastage of raw materials and Maximise the Efficient usage of the Machine and the workers. A multi objective optimization is defined as $U = [U_1\ U_2\ \ldots\ U_n]$ Where U is the control variable vector, n is the no of control variable and Objective function is Min / Max $\mathbf{F} = \{ f_1(U), f_2(U), \ldots , f_m(U) \}$ Subject to $Gj(U) \leq 0$, j=1,2,…..,m; L j (U) =0 , j=1,2,…..,p. In order to make single objective, each objective has some weight, combine the objectives into single weighted formula $W_1 * f_1(U) + W_2 * f_2(U) + \ldots + Wm * fm(U)$ and normalize the weights using $W_1+W_2+\ldots+W_m=1$. The limitation of Multi Objective is instead of returning a single solution, in Multi Objective problems it might be better to return a set of solutions, representing different trade-offs among the objectives.

3. PROPOSED METHODOLOGY FOR SOFTWARE EFFORT ESTIMATION

The following section introduces the methodology that has been used on the proposed model in order to tune the parameters. The parameters are tuned by using Particle Swarm Optimization with two objectives, which are Mean Absolute Relative Error (MARE) and Prediction (n). The parameters should minimize the MARE and Maximize the Prediction accuracy. The objectives defined as

$$\%\ \text{MARE} = \text{mean}\left[\frac{\text{abs (Measured Effort} - \text{Estimated Effort)}}{\text{(measured effort)}}\right] \times 100 \quad (6)$$

Prediction (n) is number of projects having less than n% error in the measured value. (7)

3.1 Methodology

The Methodology/ Algorithm is used to tune the parameters are
Step 1: Start

Step 2: Initialize the m particles random position and velocity vectors $[P_1, P_2 \ldots P_m]$ and $[V_1, V_2 \ldots V_m]$ respectively for parameters to be tuned.

Step 3: Initialize all the particles as Pbest particles.

Step 4: Evaluate the two fitness functions $f_1(U)$, $f_2(U)$ using equations 7 and 8 for all the particles. The objective of $f_1(U)$ is minimization and objective of $f_2(U)$ is also maximization.

Step 5: This step converts Multi Objective into Single Objective by using weighted ranking method. For each two objectives assign ranks for all the particles. Add the ranks of objectives assigned to each particle. Final fitness is minimization.

Step 6: if fitness (p) better than fitness (Pbest) then Pbest = p.

Step 7: set the best of Pbest as a Gbest.

Step 8: update the particles velocities using the equations 4 and 5.

Step 9: repeat the steps 4 to 8 until particles exhaust that is no change in the objectives.

Step 10: Give the Gbest value parameters as optimal solution.

3.2 Proposed Model

We considered intermediate COCOMO model for tuning parameters. The proposed model is

$$\text{Effort} = a*(\text{Size})^b * \text{EAF} + c \qquad (8)$$

Where a, b are cost parameters and c is bias factor. Size is coding size measured in KDLOC and Effort is in terms of Person Months (PM's) In order to tune the parameters the above methodology multi objective particle swarm optimization is used.

4. IMPLEMENTATION AND PERFORMANCE MEASURES

The following section describes the experimentation part of work, and in order to conduct the study and to establish the affectivity of the models two datasets of 20 projects and 21 projects from COCOMO dataset were used. We have implemented the above methodology for tuning parameters a, b and c in "C" language. The performance measures considered is equations 6 and 7. By running the "C" implementation of the above methodology we obtain the following parameters for the proposed model.

Experiment 1:

Totally 20 projects are considered. Number of iterations 100, Number of particles considered are 50.
a=1.538113, b=1.270503 and c=2.800148. The range of a is [1, 10] b is [-5,5] and c is [-5,5] .

The following table shows estimated effort of our proposed models:

Table 1: Estimated Efforts of Proposed Models using MOPSO

Project No	Size	EAF	Measured Effort	COCOMO Effort	Estimated Effort MOPSO Model	COCOMO Error %	MOPSO Error %
1	46	1.17	240	212	237	13.207	1.699
2	16	0.66	33	39	37.2	15.384	11.26
4	6.9	0.4	8	9.8	10	18.367	19.67
10	24	0.85	79	108	77	26.851	2.690
14	1.9	1.78	9	10.7	9	15.887	0.111
21	2.14	1	7.3	7	6.8	4.2857	6.725
22	1.98	0.91	5.9	5.8	6.1	1.7241	3.752
28	34	0.34	47	44	49	6.8181	3.983
30	6.2	0.39	8	8.4	8.9	4.7619	10.01
31	2.5	0.96	8	8.9	7.5	10.112	6.241
32	5.3	0.25	6	4.7	6	27.659	0
33	19.5	0.63	45	46	45	2.1739	0
38	23	0.38	36	33	34.2	9.0909	5.293
42	8.2	1.9	41	55	45.1	25.454	9.171
43	5.3	1.15	14	22	17.5	36.363	20.09
44	4.4	0.93	20	14	12.2	42.857	63.93
49	21	0.87	70	68	66.8	2.9411	4.743
51	28	0.45	50	47	50.5	6.3829	1.048
52	9.1	1.15	38	42	32.1	9.5238	18.56
53	10	0.39	15	17	14	11.764	7.296

The following figure 1 shows the graph of measured effort versus estimated effort of COCOMO and MOPSO proposed model.

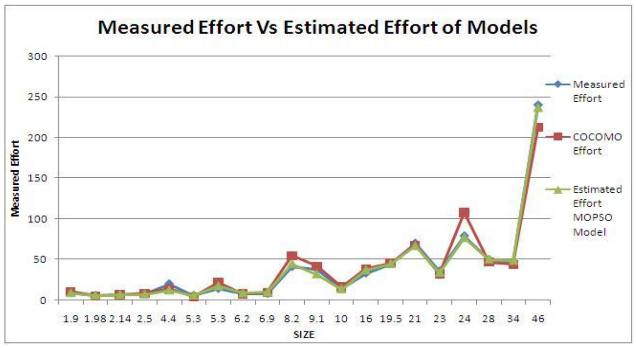

Figure 1: Measured Effort Vs Estimated Effort of Various Models

Experiment 2:

Totally 21 projects are considered. Number of iterations 100, Number of particles considered are 50.
a=3.960064, b=1.103581 and c=-5.420986. The range of a is [1, 10] b is [-5,5] and c is [-5,5] .

The following table shows estimated effort of our proposed models:

Table 2: Estimated Efforts of Proposed Models using MOPSO

Project No	Size	EAF	Measured Effort (Actual)	COCOMO Effort	Estimated Effort-MOPSO	COCOMO Error %	MOPSO Error %
1	46	1.17	240	212	311.44	8.990326	22.94
2	16	0.66	33	39	50.3	11.92624	34.40
3	4	2.22	43	30	35.17	36.95862	22.24
4	6.9	0.4	8	9.8	7.92	22.69974	0.887
5	22	7.62	107	869	908.96	22.66312	18.26
6	30	2.39	423	397	398.43	6.525573	6.166
7	18	2.38	321	214	223.44	47.88725	43.66
8	20	2.38	218	243	251.66	9.933995	13.37
9	37	1.12	201	238	233.11	15.87182	13.77
10	24	0.85	79	108	106.85	27.13894	26.06
11	3	5.86	73	60	72.58	17.90941	0.568
12	3.9	3.63	61	52	59.12	15.22093	3.164
13	3.7	2.81	40	38	41.72	4.793031	4.139
14	1.9	1.78	9	10.7	8.89	19.11694	1.207
15	75	0.89	539	443	407.98	23.53037	32.11
16	90	0.7	453	326	392.19	32.38168	15.50
17	38	1.95	523	430	422.29	22.02246	23.84
18	48	1.16	387	339	323.84	14.82191	19.50
19	9.4	2.04	88	89	90.35	1.106741	2.606
20	13	2.81	98	133	183.26	19.0982	46.52
21	2.14	1	7.3	7	3.74	8.003391	99.77

The following figure 2 shows the graph of measured effort versus estimated effort of COCOMO and MOPSO proposed model.

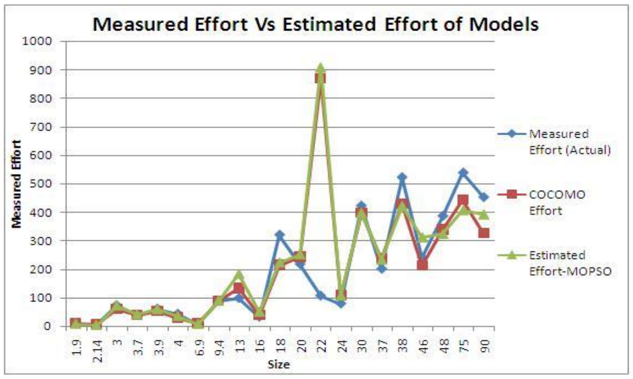

Figure 2: Measured Effort Vs Estimated Effort of Various Models

From the Figure 1 and Figure 2, one can notice that the estimated efforts are very close to the measured effort.

5. RESULTS AND DISCUSSIONS

For the Experiment-1 we considered the small projects which size less than 50 KDLOC. The MARE and Prediction accuracy is good. For the Experiment-2 we considered the large projects, the MARE and Prediction accuracy is good in some cases which is a limitation of multi objective. The results are tabulated in Table 3. It was observed that the model may gives better results and when provided with enough classification among training data set.

Table 3: Performance and Comparisons

Experiment-1	**COCOMO**	**MOPSO Proposed Model**
MARE	16.1306	9.0143
Prediction(25%)	20	24
Experiment-2		
MARE	18.1548	20.9717
Prediction(25%)	17	15

The following figure 2 shows the performance measures of COCOMO and MOPSO proposed model.

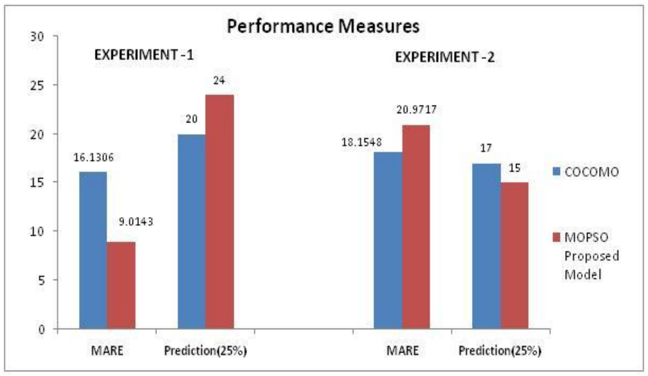

Figure 3: Performance Measure

6. CONCLUSION AND FUTURE SCOPE

Software cost estimation is based on a probabilistic model and hence it does not generate exact values. However availability of good historical data coupled with a systematic technique can generate better results. The accuracy of the model is measured in terms of its error rate. In this paper new model was proposed to estimate the software cost. In order to tune the parameters the multi objective particle swarm optimization methodology algorithm is applied. It is observed from the results that MOPSO gives better results. On testing the performance of the model in terms of the MARE and Prediction the results were found to be useful.

It is also noticed the presence of non linearity in the data items being considered during the present work for training and testing the tuning parameters and the best way to bring in some linearity among such data items is through clustering techniques. By using clustering method the data items may be divided into number of clusters and PSO be used then for parameter tuning of each cluster. These clusters and tuned parameters may be trained on Neural Networks by using efficient back propagation algorithms and results be compared for improvements as a part of future work with a view to generate even better models for software development effort estimation.

7. REFERENCES

[1]. Lawrence H.P, A general empirical solution to the macro software sizing and estimating problem, IEEE Transactions on Software Engineering, Vol. SE-4, NO. 4, , July 1978, pp. 345-361.

[2]. Hans-Dieter Joos et.al, A multi-objective optimisation-based software environment for control systems design, 2002 IEEE International symposium computer added control system design proceedings, pp: 7-14, sep 18-20.

[3]. Kavita C , GA based Optimization of Software Development effort estimation, IJCST, Vol 1, Issue 1, pp: 38-40, sep -2010.

[4]. Daniel Rodríguez, Multi Objective simulation optimization in software project management, ACM 978-1-4503-0557-0/11/07, *GECCO'11,* July 12–16, 2011.

[5]. Marco Laumanns, Evolutionary Multi Objective optimization, International Journal of Computational Intelligence Research, ISSN 0973-1873 Vol.2, 2006.

[6]. John W.B and Victor R.B, A meta model for software development resource expenditures, proceedings of the Fifth International Conference on Software Engineering,DOI:CH-1627-9/81/0000/0107500.75@IEEE,1981,pp.107-129.

[7]. Chris F.K, An Empirical Validation of Software Cost Estimation Models, Management Of Computing-Communications of ACM, Vol: 30 No. 5, , May 1987, pp.416-429.

[8]. Rajiv D.B and Chris F.K, Scale Economies in New Software Development, IEEE Transactions on Software Engineering, Vol. 15. No. 10, October 1989 pp.1199-1205.

[9]. KrishnaMurthy S and Douglas F (1995), Machine Learning Approaches to estimating software development effort, IEEE Transactions on Software Engineering, Vol. 21, No. 2, February 1995, pp. 126-137.

[10]. Pedrycz W, Peters J.F, Ramanna S, A Fuzzy Set Approach to Cost Estimation of Software Projects, proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering, May 9-12 1999, pp.1068-1073.

[11]. Wu B, Zheng Y, Liu S, and Shi Z, CSIM: A Document Clustering Algorithm Based on Swarm Intelligence, DOI:0-7803-7282-4/02@IEEE, 2002, pp.477-482.

[12]. Wei P, Kang-ping W, Chun-guang Z, Long-jiang D, Fuzzy Discrete Particle Swarm Optimization for Solving Traveling Salesman Problem, Proceedings of the Fourth International Conference on Computer and Information Technology (CIT'04), DOI:0-7695-2216-5/04, 2004, pp.1-5.

[13]. Matthew S, An Introduction to Particle Swarm Optimization, Department of Computer Science, University of Idaho, November 7, 2005, pp.1-8.

[14]. Nasser T, Neural Network Approach for Software Cost Estimation, Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05), DOI: 0-7695-2315-3/05IEE, 2005.

[15]. Xishi H, Danny H, Jing R, Luiz F. C, Improving the COCOMO model using a neuro-fuzzy approach, DOI:10.1016 /j.asoc. 2005.06.007, Elsevier-Applied Soft Computing 7 (2007) 2005, pp. 29–40.

[16]. Ajith A, He G, and Hongbo L, Swarm Intelligence: Foundations, Perspectives and Applications, Studies in Computational Intelligence, Springer-Verlag Berlin Heidelberg (SCI), 2006, pp. 3–25.

[17]. Alaa F.S, Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects, Journal of Computer Science Vol: 2, No: 2, 2006, pp.118-123.

[18]. Felix T. S. Chan and Manoj Kumar Tiwari, (2007), Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, I-TECH Education and Publishing, ISBN 978-3-902613-09-7, pp: 1- 548, 2007.

[19]. Harish M and Pradeep B, Optimization Criteria for Effort Estimation using Fuzzy Technique, CLEI Electronic Journal, Vol. 10, Number 1, Paper 2, June 2007, pp. 1-11.

[20]. Magne J and Martin S, A Systematic Review of Software Development Cost Estimation Studies, IEEE Transactions On Software Engineering, Vol. 33, No. 1, January 2007, pp. 33-53.

[21]. Rahul P and Thomas Z, Building Software Cost Estimation Models using Homogenous Data, First International Symposium on Empirical Software Engineering and Measurement, DOI: 0-7695-2886-4/07, IEEE, 2007, pp. 393-400.

[22]. Alaa S, David R and Aladdin A, Development of Software Effort and Schedule Estimation Models Using Soft Computing Techniques, IEEE Transaction, 978-1-4244-1823-7/08/IEEE, 2008, pp. 1283-1289.

[23]. Prasad Reddy P.V.G.D, Sudha, K.R., Rama S.P, Ramesh .S.N.S.V.S.C, Software Effort Estimation using Radial Basis and Generalized Regression Neural Networks, Journal of Computing, Vol.2 No.5, May 2010, pp.87-92.

[24]. Prasad Reddy P.V.G.D, Sudha, K.R., Rama Sree .P, Ramesh .S.N.S.V.S.C, Fuzzy Based Approach for Predicting Software Development Effort, International Journal of Software Engineering, Vol.1 No.1,2010, pp.1-11.

[25]. Prasad Reddy P.V.G.D, Particle Swarm Optimization In The Fine Tuning Of Fuzzy Software Cost Estimation Models, International Journal of Software Engineering, Vol.1 No.2,2010, pp. 12-23.