Software Effort Estimation by Genetic Algorithm Tuned Parameters of Modified Constructive Cost Model for NASA Software Projects

Brajesh Kumar Singh
Department of Computer Science & Engineering,
MNNIT, Allahabad, India

A. K. Misra
Department of Computer Science & Engineering,
MNNIT, Allahabad, India

ABSTRACT

Software estimation accuracy is one of the most difficult tasks for software developers. Defining the project estimated cost, duration and maintenance effort early in the development life cycle is greatest challenge to be achieved for software projects. Formal effort estimation models, like Constructive Cost Model (COCOMO) are limited by their inability to manage uncertainties and impression in software projects early in the project development cycle. A software effort estimation model which adopts a binary genetic algorithm technique provides a solution to adjust the uncertain and vague properties of software effort drivers. In this paper, COCOMO is used as algorithmic model and an attempt is being made to validate the soundness of genetic algorithm technique using NASA project data. The main objective of this research is to investigate the effect of crisp inputs and genetic algorithm technique on the accuracy of system's output when a modified version of the famous COCOMO model applied to the NASA dataset. Proposed model validated by using 5 out of 18 NASA project dataset. Empirical results show that modified COCOMO for software effort estimates resulted in slightly better as compared with results obtained in [30]. The proposed model successfully improves the performance of the estimated effort with respect to the Variance Account For (VAF) criteria, MMRE and Pred.

Keywords

COCOMO; Effort estimation; algorithmic model, Variance Account For, MMRE, Pred.

1. INTRODUCTION

Software cost estimation is the estimation of likely amount of effort, duration and staffing levels required to build a software system. Accurate Software development effort estimations are always supposed to be a difficult task to both, software developers and customers involved in development. The most significant form of software effort estimation is the one made at an early stage during a project, starting primarily from project feasibility and requirements specification documents. However, effort estimation at the early stages of the development is the most difficult task to obtain and they are often the least accurate, because very little detail about the project and the product size, the development duration and the required facilities are known at its beginning [1]. In recent years, the development of large-scale software projects is gaining a wide range of interest [2, 3]. Accurate software effort estimation and can provide powerful assistance for software management decisions. Project manager will significantly need to identify the cost estimate so that he can evaluate the project progress and have better resource utilization [4]. It was found that the main cost driver, effort has major impact on software cost estimation. The primary element which affects the effort estimation is the developed lines of code (DLOC). The DLOC include all instructions and formal statements of the program [5].

Nowadays, many software cost estimation models have been developed. Most of these models are based on the size measure, such as Lines of Code (LOC) and Function Point (FP), obtained from size estimation. It is quite obvious that the size estimation accuracy directly impacts on cost estimation accuracy.

Based on this context, new alternative approach of evolutionary algorithms such as binary genetic algorithm can be a good choice to estimate task effort in software development.

A review of the literature depicts that there are two major types of cost estimation methods Algorithmic and Non algorithmic models as discussed in various papers [5, 6, 7, 8, 9, 10, 11, 12, and 13].

This paper provides a detailed study on the use of binary genetic algorithm as an optimization algorithm which can be used to tune the modified Constructive Cost Model (COCOMO) parameters such that a better effort estimate can be provided. The performance of the developed model was tested on NASA software project dataset provided in [2] and compared to the pre-existed model presented in [30]. The developed model was able to provide better estimation capabilities.

2. PROBLEM FORMULATION

2.1 Problem Statement

Understanding and calculation of estimation models based on historical data are difficult due to inherent complex relationships between the related attributes. Attributes and relationships used to estimate software development effort could change over time and may differ for different software development environments. In order to address and overcome to these problems, a new model with accurate estimation will be desired. The problem based on algorithmic model i.e. COCOMO, has been taken into account.

2.2 Algorithmic Models

Some other famous algorithmic models are Albrecht's Function Point [16, 17] and Putnam's [18] SLIM. All of these require inputs, accurate estimate of specific attributes, such as Line Of Code (LOC), number of user screen, interfaces and complexity, which are always difficult to acquire during the early stage of software development.

2.3 The COCOMO

One of the widely used quantitative model structures to estimate the software effort is the COCOMO which was developed by Boehm [5, 14]. The COCOMO is a regression based software cost estimation model. This model was built based on 63 software projects. The model helps in defining the mathematical relationship between the software development time, the effort in man-months and the maintenance effort [15]. One of the problems with using COCOMO today is that it does not match the development environment of recent times.

The limitations of the algorithmic models led to the exploration of the non-algorithmic techniques like genetic algorithms.

3. SOLUTION OF THE PROBLEM

Recently, many questions about the applicability of using evolutionary computation techniques to develop estimation models have been introduced [19]. The objective of this study is to focus on developing an evolutionary model for estimating software effort using genetic algorithms. GAs will be used to estimate the parameters of a COCOMO based effort estimation model.

3.1 Genetic Algorithms

Genetic algorithms are adaptive heuristic search algorithms based on the Darwin theory of natural selection. They are introduced by John Holland [21] and extensively studied by Goldberg [22], De Jong [23, 24] and back [25]. GAs search the space of all possible solutions using a population of individuals which is considered as potential solutions of the problem under consideration. These solutions are computed based on their fitness. The solutions that best fit to the objective criterion survive in the upcoming generations and produce "offspring" which are variations of their Parents [20].

GAs has been successfully used in a wide variety of difficult numerical optimization problems. They have been successfully used to solve system identification, signal processing and path searching problems [26, 27, 28, and 29]. Holland introduced the binary string representation of genetic algorithms [21].

3.2 Evolutionary Process of Genetic Algorithm

In all Evolutionary Algorithms (EAs) techniques, it is required to transfer the problem from its real domain to the domain of Evolutionary algorithms. GAs offers different kinds of representations. The evolutionary process starts by the computation of the fitness of each individual in the initial population. While stopping criterion is not yet reached, do the following;

- Select individuals for reproduction using some selection mechanisms (i.e. roulette wheel, tournament, rank, etc.).
- Create an offspring using crossover and mutation operators. The probability of crossover and mutation are selected based on the application.
- Compute the new generation.

This process will end either when the optimal solution is found or the maximum number of generations is reached.

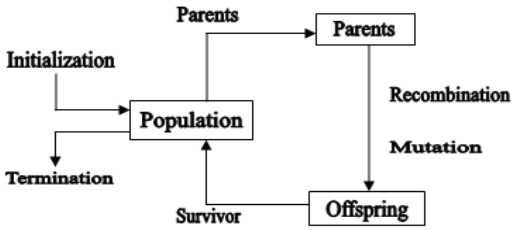

Figure 1 General Scheme of Evolutionary Process

4. Proposed approach for solving problem

To see, how the ideas of evolutionary algorithms are applied to function optimization, It is supposed that without loss of generality we want to minimize a function of n parameters $f(a_1, a_2, ..., a_n)$. A domain $D_i = [\alpha_i, \gamma_i]$, for (i=1,2,...,n) is identified as a search space for each parameter. $f(a_1, a_2, ..., a_n)$ is positive function, where a_i belongs D_i. Candidate solutions are defined as n-dimensional vectors of parameters of the form: $a_1, a_2,..., a_n$ which can be viewed as "Chromosomes" and these chromosomes consist of "genes". For each such vector of parameter values, its associated function value serves as its fitness. The small values are used for minimization problems.

The GA search process is based on using a population of individuals each of which is evaluated based on its fitness value. Individuals with higher fitness value are to the mating pool which inherits many but not all of the features of their parents. This is achieved using genetic operators like mutation and crossover [13, 14].

4.1 Evaluation criteria

4.1.1 Fitness function

The evaluation criterion to measure the performance of the developed GA based model is to calculate the Variance Account For (VAF) including Mean Magnitude of Relative Error (MMRE) and probability of a project having a relative error of less than or equal to L (PRED(L)).

The VAF is calculated as:

[1 - var (Actual Effort –Estimated Effort)/ var (Actual Effort)] × 100% (1)

Where variance is termed as var. The variance is calculated as:

$$\frac{1}{n}\sum_{i=1}^{n}(x_i)^2 - \left(\frac{1}{n}\sum_{i=1}^{n}x_i\right)^2 \quad (2)$$

Here, x is the variable and n is the number of values of that variable.

MMRE and PRED are calculated from the relative error, or RE, which is the relative size of the difference between the actual and estimated value of individual effort i :

RE_i = (Estimated $Effort_i$ – Actual $Effort_i$) /Actual $Effort_i$) (3)

The magnitude of relative error [31] can be calculated by taking the absolute value of that relative error that is,

$$MRE_i = abs(RE_i) \quad (4)$$

The MRE value is calculated for each observation i of actual and estimated effort. The aggregation of MRE over multiple observations (N) can be achieved through the Mean of MRE (MMRE) as follows:

$$MMRE = \frac{1}{N}\sum_{i}^{N} MRE_i \qquad (5)$$

A complementary criterion is the prediction at level L, Pred(L) = k/N, here k is the number of observations where MRE is less than or equal to L and N is the total number of observations.

4.2 Dataset description

Experiments have been conducted on a data set presented by Bailey and Basili[2] to develop an effort estimation model. The dataset consists of three variables. They are the Developed Line of code (DLOC), the Methodology (ME) as an element contributing to the computation of the software developed effort and the measured effort. DLOC is described in Kilo Lines of Code (KLOC) and the Effort is in person-months. The dataset is given in Table 1.

Table 1: The Dataset of NASA Software Projects

Project No.	KDLOC	ME	Actual Effort
1	90.2	30	115.8
2	46.2	20	96
3	46.5	19	79
4	54.5	20	90.8
5	31.1	35	39.6
6	67.5	29	98.4
7	12.8	26	18.9
8	10.5	34	10.3
9	21.5	31	28.5
10	3.1	26	7
11	4.2	19	9
12	7.8	31	7.3
13	2.1	28	5
14	5	29	8.4
15	78.6	35	98.7
16	9.7	27	15.6
17	12.5	27	23.9
18	100.8	34	138.3

5. Results and Discussion

The data for the first 13 projects were used to estimate the model parameters and the other 5 projects were used for testing their performance which is shown in table 3.
The tuning parameters for the GA evolutionary process, to estimate the COCOMO parameters, which include the population size, crossover, mutation types and selection mechanisms are given in the Table 2.

Table 2 Parameters of GA evolutionary process

Operator	Type
Selection Mechanism	Roulette wheel Selection
Crossover Type	Single Point Binary Crossover
Mutation Type	Non Uniform Mutation
Population Size	5
Domain Search of a	02:04
Domain Search of b	0.1:0.9
Domain Search of c	-0.5:0.5
Domain Search of d	0:20

Table 3 Showing the Effort Estimated by GA

Project No.	Actual Effort	Estimated Effort	KDLOC
1	115.8	118.7299525	90.2
2	96	73.42066986	46.2
3	79	74.1895744	46.5
4	90.8	83.35094298	54.5
5	39.6	48.14962838	31.1
6	98.4	94.60984978	67.5
7	18.9	25.07660947	12.8
8	10.3	18.00257187	10.5
9	28.5	36.4181323	21.5
10	7	7.009573391	3.1
11	9	12.2344847	4.2
12	7.3	14.43507557	7.8
13	5	3.841133884	2.1
14	**8.4**	**9.858026695**	**5**
15	**98.7**	**103.7365314**	**78**
16	**15.6**	**19.44267998**	**9.7**
17	**23.9**	**24.18104153**	**12.5**
18	**138.3**	**128.0677328**	**100.8**

Now, we will explore the proposed modeling process and describe the mathematical equations for the model. This model is proposed, based on some theoretical aspects related to linear model structure development process. Adding the ME in COCOMO will have the significant effect and improve the model prediction quality as given in proposed model. It is also found that adding a bias term d similar to the classes of regression models helps to stabilize the model and reduce the effect of noise in measurements.

5.1 Model

The proposed model structure considered the effect of ME as linearly related to the effort. The proposed model structure has their parameters a, b, c and a new bias parameter d.
The proposed model is given mathematically as follows:

$$\text{Effort} = a(\text{DLOC})^{b} + c(\text{ME}) + d \qquad (6)$$

Our goal is to find the model parameters which most suited to accurately and the software effort for project development. In table 3, the actual effort and the estimated effort based on the proposed model are shown using the same dataset given in table 1. The estimated parameters a, b, c and d for proposed model are estimated using GAs as follows:

$$\text{Effort} = 3.3602(\text{DLOC})^{0.8116} - 0.4524(\text{ME}) + 17.8025 \qquad (7)$$

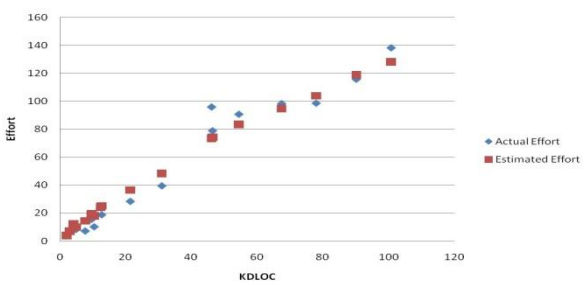

Figure 4 Actual Efforts and Estimated Effort

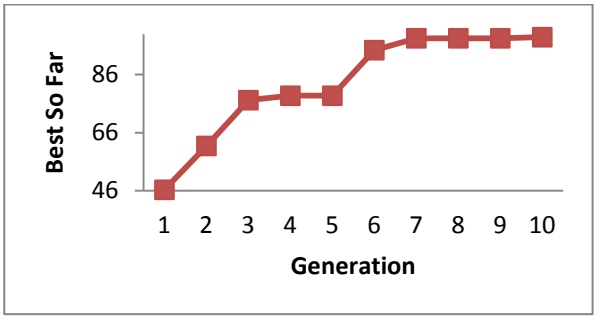

Figure 5. The best so far curve of Var

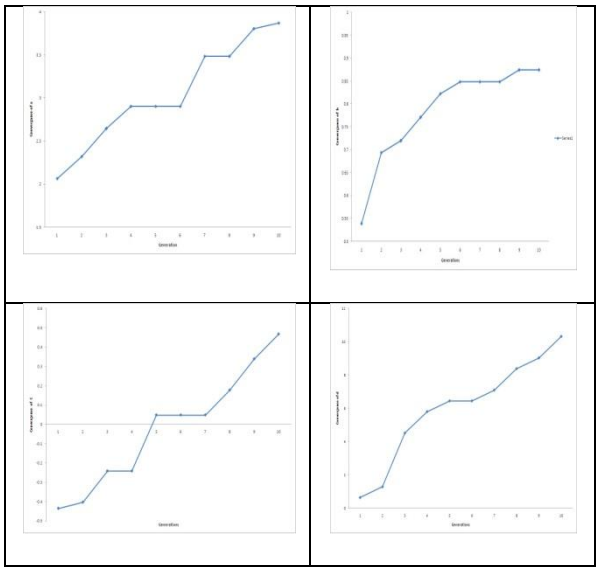

Figure 6 Convergence of the model parameters a, b, c and d

Figures 4-6 show the actual effort and estimated effort using binary GA, best so far curve for different generations and the convergence of the proposed model parameters after each generation.

Genetic Algorithms is used to estimate the COCOMO model parameters. The estimation capabilities for the model are shown in Table 4. A slightly better estimation capability was achieved using developed model as compared to other models [30]. From the Table 4, it can be observed that taking into consideration the effect of ME and adding new bias d help to improve the computed VAF. The proposed model successfully improves the performance of the estimated effort with respect to the VAF criteria.

Table 4 Estimation capabilities of the Models

Model Name	Model Input	Model Output	VAF
Proposed Model	KDLOC and ME	Effort	98.91
Model Proposed by sheta[30]	KDLOC and ME	Effort	97.565

Figure 7 depicts comparison made between 18 results produced by data for proposed model and corresponding data set for [30] as well. It can observed by the figure 7 that MRE produced by proposed model is always kept lower to the mean of MRE which shows the accuracy of the model. But in case of [30], there are few spikes with high MRE which show the inconsistency in the estiamtion of efforts.

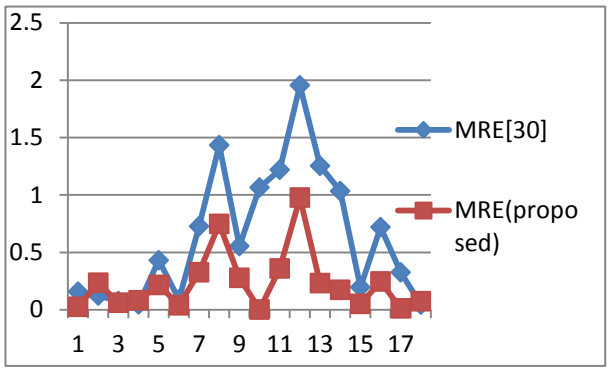

Figure 7 MRE Graph for two models

Table 5 shows the MMRE and Pred. Thus, proposed model gives around 40% improvement in performance and Pred (25) of proposed model gives the 72.22 percentage of projects which were predicted with a MRE less than or equal to 0.25.

Table 5 Comparison between the performance of two models

Model Name	MMRE	Pred(25%)
Proposed Model	0.2298287	72.222222
model proposed by Sheta[30]	0.636397	38.888889
% Improvement	40.656827	

6. CONCLUSION

In this study, new model structure is proposed to estimate the software effort for projects sponsored by NASA using binary genetic algorithm. Modified version of the famous COCOMO model was provided to consider the effect of methodology in effort estimation. The performance of the developed model was tested on NASA software project data presented in [2]. The developed models were able to provide good estimation capabilities.

7. REFERENCES

[1] Parvinder S. Sandhu, Porush Bassi, and Amanpreet Singh Brar, Software Effort Estimation Using Soft Computing Techniques, World Academy of Science, Engineering and Technology pp 46 2008.

[2] Bailey, J. W. and V. R. Basili, 1981. A meta model for software development resource expenditure. Proc. Intl. Conf. Software Engineering, pp: 107-115.

[3] Boraso, M., C. Montangero and H. Sedehi, 1996.Software cost estimation: An experimental study of model performances. Tech-nical Report TR-96-22, Departimento Di Informatatica, Uni-versita Di Pisa, Italy.

[4] B. Boehm, Software Cost Estimation with COCOMO II, Prentice Hall PTR, Upper Saddle River, New Jersey, 2000.

[5] B.W. Boehm,Software engineering economics, Englewood Cliffs, NJ: Prentice-Hall, 1981.

[6] C. E. Walston, C. P. Felix, A method of programming measurement and estimation, IBM Systems Journal, vol. 16, no. 1, pp. 54 73, 1977.

[7] G.N. Parkinson, Parkinson's Law and Other Studies in Administration, Houghton-Miffin, Boston, 1957.

[8] L. H. Putnam, A general empirical solution to the macro software sizing and estimating problem, IEEE Trans. Soft. Eng., pp. 345-361, July 1978.

[9] J. R. Herd, J.N. Postak, W.E. Russell, K.R. Steward, Software cost estimation study: Study results, Final Technical Report, RADCTR77- 220, vol. I, Doty Associates, Inc., Rockville, MD, pp. 1-10, 1977.

[10] R. E. Park, PRICE S, The calculation within and why, Proc. of ISPA Tenth Annual Conference, Brighton, England, pp. 231-240, July 1988.

[11] R.K.D. Black, R. P. Curnow, R. Katz, M. D. Gray, BCS Software Production Data, Final Technical Report, RADC-TR-77-116, Boeing Computer Services, Inc., March, pp. 5-8, 1977.

[12] R. Tausworthe, Deep Space Network Software Cost Estimation Model, Jet Propulsion Laboratory Publication 81-7, pp. 67-78, 1981

[13] W. S. Donelson, Project Planning and Control, Proc. Datamation, pp. 73- 80, June 1976.

[14] Boehm, B., 1995. Cost Models for Future Software Life Cycle Process: COCOMO2 Annals of Software Engineering.

[15] Kemere, C.F., 1987. An empirical validation of software cost estimation models. Communication ACM, 30: 416-429.

[16] Boehm B., C. Abts and S. Chulani, 2000. Software development cost estimation approaches-A survey. Ann. Software Eng., 10: 177-205. DOI: 10.1023/A: 1018991717352.

[17] Boehm, B., 1995. Cost models for future software life cycle processes: COCOMO 2.0. Ann. Software Eng. 1: 45 60.

[18] Putnam, L.H., 1978. A general empirical solution to the macro software sizing and estimating problem. IEEE Trans. Software Eng., 4: 345-361. http://portal.acm.org/citation.cfm?id=1313641.

[19] Dolado. C.J. and M. Leey, 2001. Can genetic programming improve software effort estimation? Comparative evaluation. Inform. Software Technol., 43: 863-873.

[20] Sheta. A. and K. DeJong, 1996. Parameter estimation of nonlinear systems in noisy environment using genetic algorithms. Proc. IEEE Intl. Symp. Intelligent Control (ISIC'96), pp: 360-366.

[21] Holland, J., 1975. Adaptation in Natural and Artificial Systems. Ann Arbor, MI: University of Michigan Press.

[22] Goldberg, D., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. New York, Addison-Wesley.

[23] De Jong, K.A., 1975. Analysis of Behavior of a Class of Genetic Adaptive Systems. Ph.D. Thesis. University of Michigan, Ann Arbor, MI.

[24] De Jong, K., 1992. Are genetic algorithms function optimizers? Proc. Sec. Parallel Problem Solving From Nature Conference, pp:3-14. The Netherlands: Elsevier Science Press.

[25] Back, T. and H.P. Schwefel, 1993. An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation, 1, pp: 1-

[26] Kristinsson. K. and G. Dumont, 1992. System identification and control using genetic algorithms. IEEE Transaction on Systems, Man and Cybernetics, 22: 1022-1046.

[27] Fonseca, C., E. Mendes, Fleming and S.A. Billings,1993. Nonlinear model term selection with genetic algorithms. Proc. IEE/IEEE Workshop on Natural Algorithms in Signal Process., pp: 27/1 –27/8.

[28] Schultz. A. and J. Grefenstette, 1994. Evolvingrobot behavior. Proc. Artificial Life Conf. MIT Press.

[29] Chipperfield, A.J. and P.J. Fleming, 1996. Genetic algorithms in control systems engineering. IASTED J. Computers and Control, 24: 1.

[30] Sheta, A. F., Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects, Journal of Computer Science 2 (2): 118-123, 2006

[31] Burgess, C.J. and M. Lefley, 2001, Can genetic programming improve software effort estimation? A comparative evaluation. Inform. Software Technol., 43: 863-873. DOI: 10.1016/S0950-5849(01)00192-6.