Journal of Computer Science 2 (2): 118-123, 2006
ISSN 1549-3636

Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects

Alaa F. Sheta
Computers and Systems Department, Electronics Research Institute (ERI), El-Tahrir Street, Dokky, Giza, Egypt

Abstract: Defining the project estimated cost, duration and maintenance effort early in the development life cycle is a valuable goal to be achieved for software projects. Many model structures evolved in the literature. These model structures consider modeling software effort as a function of the developed line of code (DLOC). Building such a function helps project managers to accurately allocate the available resources for the project. In this study, we present two new model structures to estimate the effort required for the development of software projects using Genetic Algorithms (GAs). A modified version of the famous COCOMO model provided to explore the effect of the software development adopted methodology in effort computation. The performance of the developed models were tested on NASA software project dataset [1].The developed models were able to provide a good estimation capabilities.

Key words: COCOMO model, NASA software, genetic algorithms, genetic programming technique

INTRODUCTION

In recent years, the development of large-scale software projects gain a growing interest [2,3]. Being able to define, the software size, the development duration and the required facilities became more and more a challenging task. The reason is software architecture, requirements, tools and techniques became more complex.

Project manager will significantly need to identify the cost estimate so that he can evaluate the project progress and have better resource utilization. It was found that the main cost driver is the effort [4]. The primary element which affects the effort estimation is the developed line of code (DLOC). The DLOC include all program instructions and formal statements.

One of the famous model structures used to estimate the software effort is the COnstructive COst Model (COCOMO). COCOMO was developed by Boehm [4,5]. 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 [6].

Soft-computing techniques were explored to build efficient effort estimation models structures. In [7], authors provided a survey on the cost estimation models using artificial neural networks. Fuzzy logic and neural networks were used for software engineering project management [8]. A fuzzy COCOMO model was developed [9].

Recently, many questions about the applicability of using evolutionary computation techniques to build estimation models were introduced [10]. The objective of this study is to focus on building an evolutionary model for estimating software effort using genetic algorithms. GAs will be used to estimate the parameters of a COCOMO type effort estimation model. Genetic algorithm is an adaptive search algorithm based on the Darwinian notion of natural selection. GAs searches the space of all possible solution using a population of individuals which is considered as potential solutions of the problem under study. 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 [24].

Stochastic algorithms: There exist many engineering and computer science problems for which no adequate, robust and global algorithms exist. Most of these problems are optimization problems [11]. There are two classes of algorithms often used to deal with such complex problems.

They are the deterministic and the stochastic algorithms. The deterministic algorithms usually provide approximate solutions and not optimal ones. A *priori* knowledge about the starting search location affects the search process. Poor starting points significantly direct the search toward local optimal solution. This represents a challenge for the deterministic search. For hard optimization problems, it is often recommended to use probabilistic algorithms. These algorithms do not assure global optimal solutions but they have the advantage of randomly generating solutions with higher level of performance accuracy.

Genetic algorithms: Genetic Algorithms (GAs) are among those stochastic search algorithms. They are adaptive search procedures which were introduced by

Corresponding Author: Alaa F. Sheta, Computers and Systems Department, Electronics Research Institute (ERI), El-Tahrir Street, Dokky, Giza, Egypt

John Holland [12] and extensively studied by Goldberg [13], De Jong [14,15] and others [16]. 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 planning problems [17-20].

Evolutionary process: The evolutionary process of GAs starts by the computation of the fitness of each individual in the initial population. While stopping criterion is not yet reached we do the following;

* Select individual for reproduction using some selection mechanisms (i.e. 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.

Representation: In all Evolutionary Algorithms (EAs) techniques, it is required to transfer the problem from its real domain to the domain of EA. GAs offer different kinds of representations. Holland introduced the binary string representation [12]. Michalewicz showed that for real-valued numerical optimization problems, floating-point representations is more efficient and can lead to faster convergence to the optimal solution domain [21]. This representation scheme is closer to the real problem domain and can achieve higher performance and accuracy.

Genetic algorithms versus conventional search algorithms: One of the major advantages of GAs compared to conventional search algorithms is that it operates on a population of solutions not only a single point. This makes GA results more robust and accurate. The solution provided by GAs is more optimal and global in nature. GAs are less likely to be trapped by local optima like Newton or gradient descent methods [22, 23]. GAs require no derivative information about the fitness criterion [13,14].This is why it is very suitable for both continuous and discrete optimization problems. In addition, GAs are less sensitive to the presence of noise and uncertainty in measurements [24,25]. There are some features which make genetic algorithms different from conventional search algorithms. Goldberg [13] stated that:

* Genetic Algorithms implement the search using a coded solution not the solutions themselves.
* Genetic Algorithms is based on a population of candidate solutions, not just a single solution.
* Genetic Algorithms evaluate individual based on their fitness function not the derivative of the function.
* Genetic Algorithms use probabilistic operators (i.e. crossover and mutation) not deterministic ones.

Problem formulation: To see how these ideas are applied to function optimization, suppose without loss of generality that we want to minimize a function of *n* parameters $f(a_1, a_2, ..., a\ _n)$. A domain $D_i = [\alpha_i\ \gamma_i]$, $(i=1,2,...,n)$ is identified as a search space for each parameter. $f(a_1, a_2, ..., a_n)$ is positive function. $a_i\ \varepsilon D_i$. 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 the individual parameters as "genes". For each such vector of parameter values, its associated function value serves as its fitness, with lower values preferred 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 are selected to produce offspring which inherit many but not all of the features of their parents. This is achieved using genetic operators like mutation and crossover [13,14].

Fitness function: The evaluation criterion to measure the performance of the developed GA based models is *selected to be* the Variance-Accounted-For (VAF).The VAF is calculated as:

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

Experimental results: Experiments have been conducted on a data set presented by Bailey and Basili [1] so that we can develop an effort estimation model. The dataset consist of two variables. They are the Developed Line of code (DLOC), the Methodology (ME) and the measured effort. DLOC is described in Kilo Line of Code (KLOC) and the Effort is in man-months. The dataset is given in Table 1.

Table 1: NASA software project data

Project No.	KDLOC	ME	Measured Effort
1.	90.2000	30.0000	115.8000
2.	46.2000	20.0000	96.0000
3.	46.5000	19.0000	79.0000
4.	54.5000	20.0000	90.8000
5.	31.1000	35.0000	39.6000
6.	67.5000	29.0000	98.4000
7.	12.8000	26.0000	18.9000
8.	10.5000	34.0000	10.3000
9.	21.5000	31.0000	28.5000
10.	3.1000	26.0000	7.0000
11.	4.2000	19.0000	9.0000
12.	7.8000	31.0000	7.3000
13.	2.1000	28.0000	5.0000
14.	5.0000	29.0000	8.4000
15.	78.6000	35.0000	98.7000
16.	9.7000	27.0000	15.6000
17.	12.5000	27.0000	23.9000
18.	100.8000	34.0000	138.3000

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.

Effort model based DLOC: The COnstructive COst Model (COCOMO) was provided by Boehm [4,5]. This model structure is classified based on the type of projects to be handled. They include the organic, semidetached and embedded projects. This model structure comes in the following form:

$$Effort = a\,(DLOC)^{b} \qquad (1)$$

Normally the model parameters are fixed for these models based on the software project type [4,5]. Our goal is to use GAs to provide a new estimate of the COCOMO model parameters. This will allow us to compute the effort developed for the NASA software projects. The estimated parameters will significantly generalize the computation of the developed effort for all projects. We used GAs to develop the following model.

$$Effort = 4.9067(DLOC)^{0.7311} \qquad (2)$$

In Table 2, we show the actual measured effort over the given 18 projects and the effort estimated based the GAs model.

Table 2: COCOMO: Measured and Estimated Effort Values using GAs

Project No.	Measured Effort	GAs Estimated Effort
1.	115.8000	131.9154
2.	96.0000	80.8827
3.	79.0000	81.2663
4.	90.8000	91.2677
5.	39.6000	60.5603
6.	98.4000	106.7196
7.	18.9000	31.6447
8.	10.3000	27.3785
9.	28.5000	46.2352
10.	7.0000	11.2212
11.	9.0000	14.0108
12.	7.3000	22.0305
13.	5.0000	8.4406
14.	8.4000	15.9157
15.	98.7000	119.2850
16.	15.6000	25.8372
17.	23.9000	31.1008
18.	138.3000	143.0788

The tuning parameters for the GA evolutionary process, to estimate the COCOMO model parameters, which include the population size, crossover, mutation types and selection mechanisms are given in the Table 3. We used the GAOT Matlab Toolbox to produce our results [26].

Table 3: The tuning parameters for the GA

Operator	Type
Selection Mechanism	normGeomSelect
Crossover type	arithXover
Mutation Type	nonUnifMutation
Population size	10
Maximum generation	100
Domain of search for a	0:10
Domain of search for b	0.3:2

The computed VAF criterion was 96.3138. Figures 1-3 show the measured and estimated GA effort, the convergence process for GAs (i.e. the best so far curve of the VAF) and the convergence of the GA model parameters after each generation.

Proposed effort models based DLOC and ME: To consider the effect of methodology (ME), as an element contributing to the computation of the software developed effort, we proposed two new models structures. We will call them model 1 and model 2. They are variations of the famous COCOMO model.

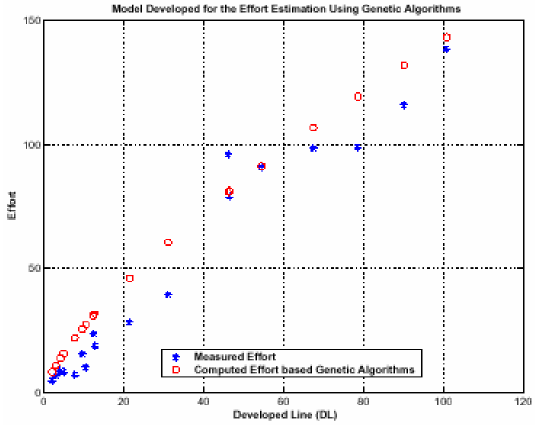

Fig. 1: Measured effort and estimated effort using genetic algorithms

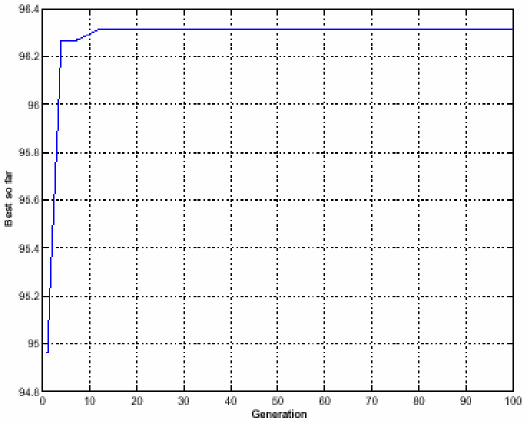

Fig. 2: Best so far curve-fitness function (VAF)

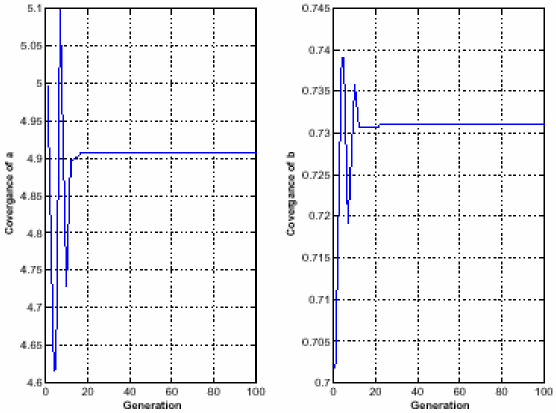

Fig. 3: Convergence of the model parameters *a* and *b*

Now, we will explore the modeling process of the proposed models and describe the mathematical equations for the two models. We proposed these models based on some theoretical aspects related to linear model structure development process. Adding the effect of ME will improve the model prediction quality as given in model 1. It was also found that adding a bias term similar to the classes of regression models helps to stabilize the model and reduce the effect of noise in measurements.

Model 1: The proposed model structure considered the effect of ME as linearly related to the effort. The proposed model structure have there parameters a, b and c.

$$Effort = a(DLOC)^{b} + c(ME) \qquad (3)$$

Our goal is to find the model parameters which most suited to accurately and the software effort for project development. In Table 4, we show the actual measured effort and the estimated effort based on the proposed model 1 using the same dataset. The model parameters were estimated and the developed model was as follows:

$$Effort = 3.1938(DLOC)^{0.8209} - 0.1918(ME) \qquad (4)$$

Table 4: Model 1: Measured and estimated effort values using GAs

Project No.	Measured Effort	GAs Estimated Effort
1.	115.8000	124.8585
2.	96.0000	74.8467
3.	79.0000	75.4852
4.	90.8000	85.4349
5.	39.6000	50.5815
6.	98.4000	99.0504
7.	18.9000	24.1480
8.	10.3000	18.0105
9.	28.5000	37.2724
10.	7.0000	4.5849
11.	9.0000	8.9384
12.	7.3000	13.5926
13.	5.0000	1.5100
14.	8.4000	8.2544
15.	98.7000	110.5249
16.	15.6000	18.2559
17.	23.9000	23.3690
18.	138.3000	135.4825

Model 2: A slightly better estimation capabilities was achieved using developed model 1. This is why we decide to modify the model by adding a new bias parameter to the above model and re-estimate the new model parameters, model 2, using GAs.

The proposed model 2 is given mathematically as follows:

$$Effort = a(DLOC)^{b} + c(ME) + d \qquad (5)$$

The estimated parameters a, b,c and d for model 2 were estimated using GAs as follows:

$$Effort = 3.3602\,(DLOC)^{0.8116} - 0.4524(ME) + 17.8025 \quad (6)$$

Figures 4-6 show the measured effort and estimated effort based the GA model 2, the convergence process for GAs and the convergence of the GA model parameters after each generation. We computed the fitness function of the developed GA model (VAF) as 97.5648.

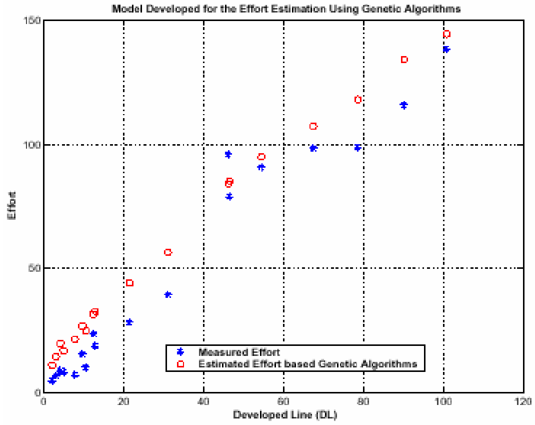

Fig. 4: Measured effort and estimated effort using genetic algorithms

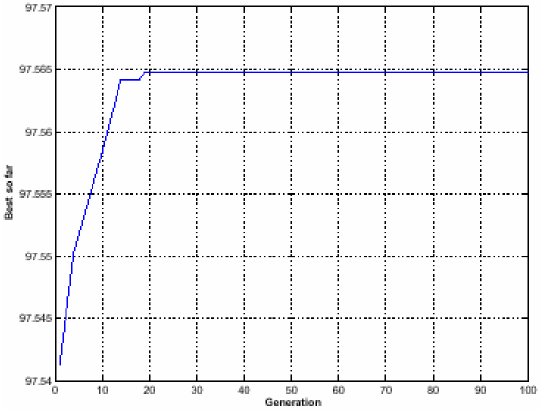

Fig. 5: Best so far curve-fitness function (VAF)

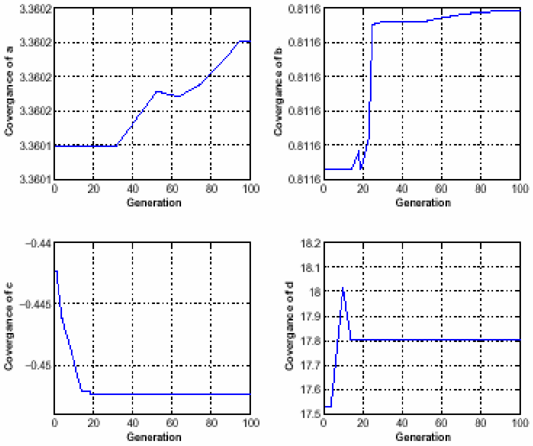

Fig. 6: Convergence of the model parameters *a, b, c* and *d*

In Table 5, we show the actual measured effort and the estimated effort based on proposed model 2.

Table 5: Model 2: Measured and estimated effort values using GAs

Project No.	Measured Effort	GAs Estimated Effort
1.	115.8000	134.0202
2.	96.0000	84.1616
3.	79.0000	85.0112
4.	90.8000	94.9828
5.	39.6000	56.6580
6.	98.4000	107.2609
7.	18.9000	32.6461
8.	10.3000	25.0755
9.	28.5000	44.3086
10.	7.0000	14.4563
11.	9.0000	19.9759
12.	7.3000	21.5763
13.	5.0000	11.2703
14.	8.4000	17.0887
15.	98.7000	118.0378
16.	15.6000	26.8312
17.	23.9000	31.6864
18.	138.3000	144.4587

The tuning parameters for the GA evolutionary process which includes the search space for the model parameters, population size, crossover probability and mutation probability are given in the Table 6.

Table 6: The tuning parameters for the GA

Operator	Type
Selection Mechanism	normGeomSelect
Crossover type	arithXover
Mutation Type	nonUnifMutation
Population size	10
Maximum generation	100
Domain of search for a	0:10
Domain of search for b	0.3:2
Domain of search for c	-0.5:0.5
Domain of search for d	0:20

RESULTS

We developed two new model structures, as variation of the COCOMO model to compute the effort required for each of the 18 projects. Our intention concern the development of model structures which can generalize the effort computed for all projects under study.

Genetic Algorithms were used to estimate the COCOMO model parameters. Two models, model 1 and 2, were provided. The prediction capabilities for the three models are shown in Table 7. From the Table, it can be seen that taking into consideration the effect of ME helps to improve the computed VAF. The two proposed models successfully improved the performance of the estimated effort with respect to the VAF criteria.

Table 7: The computed variance-accounted-for (VAF) criterion

Model Input	Model Output	VAF
KDLOC	Effort	96.3138
KDLOC and ME: Model 1	Effort	96.8496
KDLOC and ME: Model 2	Effort	97.5648

CONCLUSION

In this study we proposed two new model structures to estimate the software effort for projects sponsored by NASA using genetic algorithms. Modified versions of the famous COCOMO model were provided to consider the effect of methodology in effort estimation. The performances of the developed models were tested on NASA software project data presented in [1]. The developed models were able to provide good estimation capabilities. We suggest the use of Genetic Programming (GP) technique to build suitable model structure for the software effort. GP can find a more advanced mathematical function of both the DLOC and ME such that the computed effort will be more accurate.

REFERENCES

1. Bailey, J. W. and V. R. Basili, 1981. A meta model for software development resource expenditure. Proc. Intl. Conf. Software Engineering, pp: 107-115.
2. 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.
3. Dolado, J.J., 2001. On the problem of the software cost function. Information and Software Technology, 43: 61-72.
4. Boehm, B., 1981. Software Engineering Economics, Englewood Cliffs, NJ. Prentice-Hall.
5. Boehm, B., 1995. Cost Models for Future Software Life Cycle Process: COCOMO2 Annals of Software Engineering.
6. Kemere, C.F., 1987. An empirical validation of software cost estimation models. Communication ACM, 30: 416-429.
7. Shepper. M. and C. Schoeld, 1997. Estimating software project effort using analogies. IEEE Tran. Software Engg., 23: 736-743.
8. Kumar, S., B.A. Krishna and P. Satsangi, 1994. Fuzzy systems and neural networks in software engineering project management. J. Applied Intelligent. 4: 31-52.
9. Ryder, J., 1995. Fuzzy COCOMO: Software Cost Estimation. Ph.D. Thesis, Binghamton University.
10. Dolado. C.J. and M. Leey, 2001. Can genetic programming improve software effort estimation? A comparative evaluation. Inform. Software Technol., 43: 863-873.
11. Michalewicz,, Z., 1994. Genetic Algorithms+Data Structures=Evolution Programs. New York, Springer-Verlag.

12. Holland, J., 1975. Adaptation in Natural and Artificial Systems. Ann Arbor, MI: University of Michigan Press.
13. Goldberg, D., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. New York, Addison-Wesley.
14. De Jong, K.A., 1975. Analysis of Behavior of a Class of Genetic Adaptive Systems. Ph.D. Thesis. University of Michigan, Ann Arbor, MI.
15. 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.
16. Back, T. and H.P. Schwefel, 1993. An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation, 1, pp: 1-24.
17. Kristinsson. K. and G. Dumont, 1992. System identification and control using genetic algorithms. IEEE Transaction on Systems, Man and Cybernetics, 22: 1022-1046.
18. 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.
19. Schultz. A. and J. Grefenstette, 1994. Evolving robot behavior. Proc. Artificial Life Conf. MIT Press.
20. Chipperfield, A.J. and P.J. Fleming, 1996. Genetic algorithms in control systems engineering. IASTED J. Computers and Control, 24: 1.
21. Michalewicz, Z., 1994. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, New York.
22. Kristinsson. K. and G. Dumont, 1988. Genetic algorithms in system identification. Third IEEE International Symposium on Intelligent Control, pp: 597-602. IEEE Press.
23. Chipperfield, A., P.J. Fleming and C. Fonseca, 1994. Genetic algorithms tools for control systems engineering. Proc. First Intl. Conf. Adaptive Computing in Engineering Design and Control, pp: 128-133, UK.
24. 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.
25. Sheta. A. and K. De Jong, 2001. Time-series forecasting using gatuned radial basis functions. Inform. Sci. J., 133: 221-228.
26. Houck, C., J. Joines and M. Kay, 1996. A genetic algorithm for function optimization: A Matlab implementation. ACM Transactionson Mathmatical Software.