DOI: http://dx.doi.org/10.26483/ijarcs.v8i7.4113

ISSN No. 0976-5697

Volume 8, No. 7, July – August 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

OPTIMIZED MODEL FOR SOFTWARE EFFORT ESTIMATION USING COCOMO-2 METRICS WITH FUZZY LOGIC

Rahul Kumar Yadav
Department of Computer Science and engineering
Scholar, Mewar University Gangrar
Chittorgarh, India

Dr. S. Niranjan
Department of Computer Science and engineering
Professor, Mewar University Gangrar
Chittorgarh, India

Abstract: Effort estimation is the crucial activity during the planning phase of any project. The successful delivery of the software project is directly dependent on the accuracy of software effort estimation in planning phase. As effort multiplier have significant influence on the COCOMO-II and this research proposed the model for improving the precision of effort estimation using fuzzy logic on COCOMO-II effort multipliers. Fuzzy Logic is a rule based architecture which runs on binary pattern. It has Input set, associated with rule-sets based on the membership function. There are three membership functions i.e. Triangular Membership function, Trapezoidal Membership Function and Bell Membership Function which has been utilized in the proposed architecture.

Keywords: Software effort estimation, COCOMO-II, Fuzzy Logic, Membership Function.

1. INTRODUCTION

The most substantial activity in software project management is Software development effort prediction. Software effort prediction at early stages of project development holds great significance for the industry to meet the competitive demands of today's world. Accuracy, reliability and precision in the estimates of software effort are highly desirable. The globalization result in high competition between software industries. And so, estimation task has become one of the most crucial tasks inside software development course of action. Due to different issues in computer software development global computer software estimates are not exact, which leads to your great loss. Estimation of software effort sometime causes overestimation or underestimation. In both the cases software effort estimation carried out is imprecise, so companies have grown more unwavering in calculating accurate computer software effort estimation. This reflects that software cost estimation is often a complex task.

Estimation models may be introduced for dealing with such problems, it is available in three categories [1] [2]: - algorithmic model consisting of COCOMO model, perform points etc., non-algorithmic style expert and machine learning. Number of estimation models may be developed but none has appeared perfect. The most popular algorithmic estimation models which include Boehm's COCOMO [3], Putnam's SLIM [4] and Albrecht's Function Point [5]. These models require as inputs, accurate estimate of some attributes such as line of code (LOC), complexity and so on which are difficult to obtain during the early stage of a software project development. The models also have difficulty in modelling the inherent complex relationships between the contributing factors, are unable to handle categorical data as well as lack of reasoning capabilities [6]. By this paper, I have focused on the fuzzy- logic model useful for estimation process, which provides much more accurate and sensitive results as compared with other estimation models. Fuzzy logic focused COCOMO II models are highly made for software effort estimation specially when there are uncertain or imprecise data. Fuzzy logic can be put under machine studying estimation model [7].

2. THE COCOMO FRAMEWORK

The Constructive Cost Model (COCOMO) is an algorithmic software cost estimation model developed by Barry W. Boehm. The model uses a basic regression formula with parameters that are derived from historical project data and current project characteristics. Boehm proposed 3 modes of projects:

1. Organic mode – simple projects that engage small teams working in known and stable environments.
2. Semi-detached mode – projects that engage teams with a mixture of experience. It is in between organic and embedded modes.
3. Embedded mode – complex projects that are developed under tight constraints with changing requirements.

According to the Boehm's, the basic COCOMO equation takes the following form:

$$\text{Effort} = a_b * (\text{KLOC})\, b_b$$

$$D = c_b * (\text{KLOC})\, d_b$$

Where,
D is estimated development time in months. The coefficients a_b, b_b, c_b, d_b are given in table these coefficients are constraints for different category for software products.

Project	a_b	b_b	c_b	d_b
Organic Mode	2.4	1.05	2.5	.38
Semidetached Mode	3.0	1.12	2.5	.35
Embedded Mode	3.6	1.20	2.5	.32

The COCOMO II MODEL

The COCOMO II model is a COCOMO 81 update to address software development practices in the 1990's and 2000's [8].

The COCOM O II model is a regression based software cost estimation model and thought to be the most cited, best known and the most acceptable of all traditional cost prediction models.

COCOM O II comprises of the following models [8] [9]:-

Application Composition Model— this model assumes that systems are created from reusable components, scripting or database programming. This model involves prototyping efforts to resolve potential high-risk issues such as user interfaces, software/system interaction, performance, or technology maturity. It is used during the early stages of development when prototype of user interface is available. Software size estimates are based on application points / object points, and a simple size/productivity formula is used to estimate the effort required. Object points include screens, user interface, reports, and components that are likely to be used.

Early Design Model-To get rough estimates of a project's cost and duration before have determined its entire architecture. It uses a small set of new cost drivers and new estimating equations. It uses Unadjusted Function Points (UFP) as the measure of size.

Post Architecture Model: Once the system architecture has been designed, a more accurate estimate of the software size can be made. – It involves the actual development and maintenance of a software product. This model proceeds most cost effectively if a software life-cycle architecture has been developed; validated with respect to the system's mission, concept of operation, and risk; and established as the framework for the product. One could use function points or LOC as size estimates with this model. COCOMO II describes 17 cost drivers that are used in the Post Architecture model. The cost drivers for COCOM O II are rated on a scale from Very Lo w to Extra High. COCOMO II post architecture model is given as:

$$Effort = A \times [SIZE]^{B} \times {}_{i=1}\Pi^{15} EM_i$$

$$B = 1.01 + 0.01 \times \sum SF_i$$

Where A = 2.45

Cost Drivers	Range
Reliability required (RELY)	0.82-1.26
Database size (DAT A)	0.9-1.28
Product complexity (CPLX)	0.73-1.74
Required reusability (RUSE)	0.95-1.25
Documentation (DOCU)	0.81-1.23
Execution time constraint (TIME)	1-1.63
Main storage constraint (ST OR)	1-1.46
Platform volatility (PVOL)	0.87-1.3
Analyst capability (ACAP)+	1.42-0.72
Programmers capability (PCAP)	1.34-0.76
Personnel continuity (PCON)	1.29-0.81
Analyst experience (AEXP)	1.22-0.81
Programmer experience (PEXP)	1.19-0.85
Language & Tool experience (LTEX)	1.2-0.84
Use of software tool (TOOL)	1.17-0.78
Multisite development (SITE)	1.22-0.8
Schedule (SCED)	1.43-1

Scale Factors Values For COCOMO II Model

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	6.2	4.96	3.72	2.48	1.24	0
FLEX	5.07	4.05	3.04	2.03	1.01	0
RESL	7.07	5.65	4.24	2.83	1.4.	0
TEAM	5.48	4.38	3.29	2.19	1.1	0
PMAT	Level 1 [7.80]	Level 1+ [6.24]	Level 2 [4.68]	Level 3 [3.12]	Level 4 [1.56]	Level 5 [0.00]

3. FUZZY LOGIC

Since fuzzy logic foundation by Lotfi Zadeh in 1965, it has been the subject of important investigations [10]. It is a mathematical tool for dealing with uncertainty and also it provides a technique to deal with imprecision and information granularity [11]. The fuzzy logic model uses the fuzzy logic concepts introduced by Lotfi Zadeh [10].The membership! (") of an element x of a classical set A, as subset of the universe X, is defined by (2), as follows:

μ A (x) = 1 if x € A

μ A(x) = 0 if x € A

A system based on FL has a direct relationship with fuzzy concepts (such as fuzzy sets, linguistic variables, etc.) and fuzzy logic. The popular fuzzy logic systems can be categorized into three types: pure fuzzy logic systems, Takagi and Sugeno's fuzzy system and fuzzy logic system with fuzzifier and defuzzifier [12]. Since most of the

engineering applications produce crisp data as input and expects crisp data as output, the last type is the most widely used one fuzzy logic system with fuzzifier and defuzzifier. It was first proposed by Mamdani. It has been successfully applied to a variety of industrial processes and consumer products [13].

3.1 MEMBERSHIP FUNCTIONS

Below three membership functions are used [14]:

1. Trimf - Triangular-shaped built-in membership function

Syntax

y = trimf(x, params)

y = trimf(x, [a b c])

Description: The triangular curve is a function of a vector, x, and depends on three scalar parameters a, b, and c, as given by

$$f(x;a,b,c)=\begin{cases}0, & x\le a\\ \dfrac{x-a}{b-a}, & a\le x\le b\\ \dfrac{c-x}{c-b}, & b\le x\le c\\ 0, & c\le x\end{cases}$$

or, more compactly, by

$$f(x;a,b,c)=\max\left(\min\left(\frac{x-a}{b-a},\frac{c-x}{c-b}\right),o\right)$$

The parameters a and c locate the "feet" of the triangle and the parameter b locates the peak.

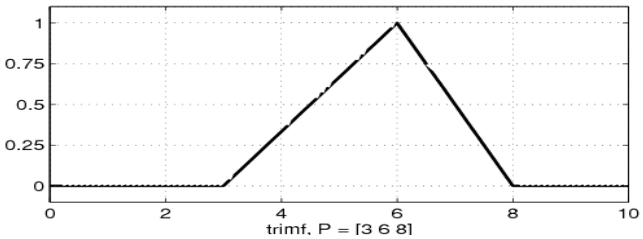


Figure: Triangular membership function

3.2 GBELLMF - GENERALIZED BELL-SHAPED BUILT-IN MEMBERSHIP FUNCTION

Syntax: y = gbellmf(x, params)

Description: The generalized bell function depends on three parameters *a*, *b*, and *c* as given by

$$f(x;a,b,c)=\frac{1}{1+\left|\dfrac{x-c}{a}\right|^{2b}}$$

Where the parameter *b* is usually positive. The parameter *c* locates the center of the curve. Enter the parameter vector params, the second argument for gbellmf, as the vector whose entries are *a*, *b*, and *c*, respectively.

A. Examples

x=0:0.1:10;

y=gbellmf(x, [2 4 6]);

plot (x, y)

x label ('gbellmf, P=[2 4 6]')

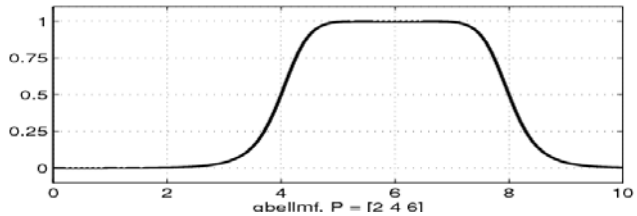


Figure: Gbell membership function.

3.3 TRAPMF - TRAPEZOIDAL-SHAPED BUILT-IN MEMBERSHIP FUNCTION

B. Syntax

y = trapm f(x, [a b c d])

*C. Description: The trapezoidal curve is a function of a vector, **x**, and depends on four scalar parameters **a**, **b**, **c**, and **d**, as given by*

$$f(x;a,b,c,d)=\begin{cases}0, & x\le a\\ \dfrac{x-a}{b-a}, & a\le x\le b\\ 1, & b\le x\le c\\ \dfrac{d-x}{d-c}, & c\le x\le d\\ 0, & d\le x\end{cases}$$

or, more compactly, by

$$f(x;a,b,c,d)=\max\left(\min\left(\frac{x-a}{b-a},1,\frac{d-x}{d-c}\right),o\right)$$

The parameters *a* and *d* locate the "feet" of the trapezoid and the parameters *b* and *c* locate the "shoulders."

D. Examples

x=0:0.1:10;

y=trapmf (x,[1 5 7 8]);

plot (x, y)

x label ('trapmf, P=[1 5 7 8]')

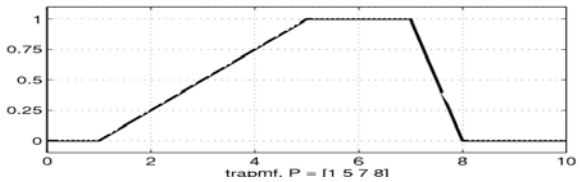


Figure: Trapezoidal membership function.

3.4 EVALUATION CRITERIA

The evaluation consists in comparing the accuracy of the estimated effort with the actual effort. There are many evaluation criteria for software effort estimation introduced in the literature, among them we will apply the most frequent evaluation criteria [15] such as:

1) Variance Accounted For (VAF)

$$\% \text{ VAF} = \left[1 - \frac{\text{var}(\textit{Measured Effort} - \textit{Estimated Effort})}{\text{var}(\textit{Measured Effort})}\right] \times 100$$

2) Mean Absolute Relative Error (MARE)

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

3) Variance Absolute Relative Error (VARE)

$$\% \text{ VARE} = \text{Var}\left[\frac{abs(\textit{Measured Effort} - \textit{Estimated Effort})}{(\textit{Measured Effort})}\right] \times 100$$

4) Prediction (n)
Prediction at level n is defined as the % of projects that have absolute relative error less than n.

5) Balance Relative Error (BRE),

$$\text{BRE} = \frac{\left|E - \hat{E}\right|}{\min(E, \hat{E})}$$

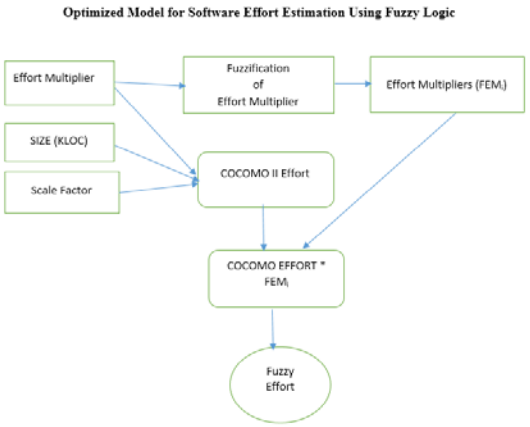
Where E = Estimated Effort, $\hat{E}$ =Actual Effort.

Absolute Relative Error (RE) =

$$\frac{\left|E - \hat{E}\right|}{E}$$

4. PROPOSED COCOMO II FUZZY MODEL

The proposed framework developed is an optimized fuzzy logic based framework and reconstruct the COCOMO II model for software effort estimation. To evaluate development effort we will use COCOMO NASA data set on proposed developed models. This research is used to handle the inaccuracy and vagueness present in the early stages of the project to predict the effort more accurately by including total transparency in the prediction system



The new proposed model is based on COCOMO II input's group and scale factors and one output, effort estimation. In COCOMO effort is expressed as Person Months (PM). It determines the efforts required for a project based on software project's size in Kilo Source Line of Code (KSLOC) as well as other cost drivers known as scale factors and effort multipliers. It contains 17 effort multipliers and 5 scale factors.

The below figures show that the fuzzy logic framework of proposed Fuzzy COCOMO II model in Matlab. These three includes upload of NASA project data, calculation of COCOMO II effort and calculation of fuzzy effort multipliers.

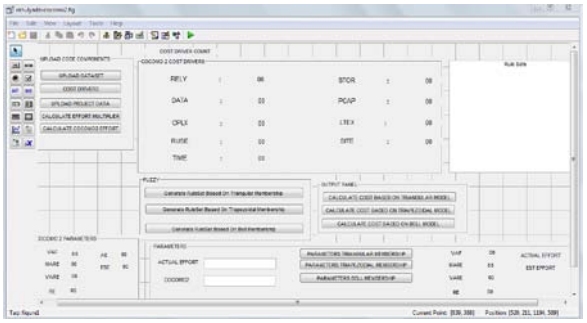


Figure: Proposed model framework in Matlab

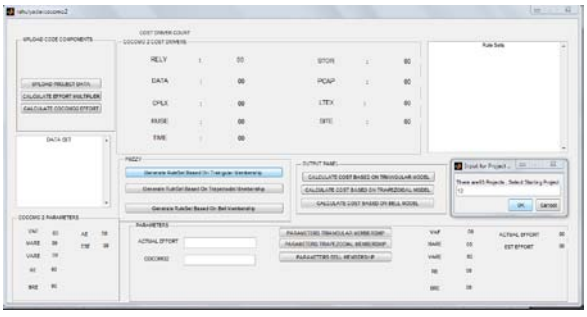


Figure: Upload Project Data

Figure: Calculation of Effort Multiplier for COCOMO II

5. CONCLUSIONS

Many researchers contributed towards the software effort estimation and presented the different model to minimize the gap between estimated and actual effort. But no single model reach to the satisfactory level due to the influence of different factors till the completion of the project. In this paper we presented a COCOMO II fuzzy model of software effort estimation. Due to the Fuzzification of 17 effort multiplier, the model will minimize the imprecision in estimated effort. The implementation of this model includes Fuzzification, fuzzy rule generation and defuzzification. Our future work will present the comparison of estimated effort with actual effort based on this model using the data set that contains records of 14 different years, starting from 1971 and ending at 1987, 93 NASA projects.

6. REFERENCES

[1]. M. Jørgensen, M. Shepperd, A systematic review of software development cost estimation studies, IEEE Transactions on SE 33 (1) (2007) 33–53.

[2]. Srichandan, Srimam, "A new approach of software effort estimation using radial basis function neural networks," International Journal on Advanced Computer Theory and Engineering (IJACTE) 1.1 (2012): 113-120.

[3]. Bohem, B.W, "Software engineering economics," prentice hall, 1981.

[4]. L. H. Putnam, "A General Empirical Solution to the Macro software Sizing and Estimating Problem," IEEE Transactions on Software Engineering, SE-4(4), 1978, pp345-361.

[5]. Attar Software, "Fuzzy Logic in Knowledge Builder", White Paper. http://www.in tellicrafters.com/fuzzy.htm, 2002

[6]. M. O. Saliu, M. Ahmed and J. Al Ghamdi, "Owards Adaptive Soft Computing based Software Effort Prediction, "Fuzzy Information, 2004. Processing NAFIPS '04.IEEE Annual Meeting of the North American Fuzzy Information Processing Society, 27-30, June 2004, 1, pp.16-21.

[7]. Chulani, Sunita, Barry Boehm, and Bert Steece. "Bayesian analysis of empirical software engineering cost models," IEEE Transactions on Software Engineering 25.4 (1999): 573-583.

[8] . Bohem, B.W, Chris Abts, A. Winsor Brown, Sunita Chulani, "Software Cost Estimation with COCOMO II," Prentice-Hall, 2000. ISBN 0-13-026692-2

[9]. Boehm B.W, B. Clark, E. Horwitz, R. Madachy, C. Abts, S.Chulani, A.W.Brown and B. Steece,"COCOMO II model definition manual, University of South California Center for Software Engineering, 2000.

[10]. Uddin, M. Nasir, Tawfik S. Radwan, and M. Azizur Rahman, "Performances of fuzzy- logic-based indirect vector control for induction motor drive." IEEE Transactions on Industry Applications 38.5 (2002): 1219-1225.

[11]. Zhao, Jin, and Bimal K. Bose. "Evaluation of membership functions for fuzzy logic controlled induction motor drive." IECON 02, Industrial Electronics Society, IEEE 2002 28th Annual Conference, Vol. 1. IEEE, 2002.

[12]. L. A. Zadeh. "Fuzzy Sets," Information and Control, **8**, 1965, pp. 338-353.

[13]. Rahul Kumar Yadav and Dr. S. Niranjan, "Software effort estimation using fuzzy logic : A Review," International journal of Engineering research and Technology, vol. 2, no. 5, pp 1377 – 1384, 2013.

[14]. Chang J, Zhao Y, Wei C (2006) Research on optimization of fuzzy membership function based on ant colony algorithm. In: The 25th Chinese control conference, pp 7

[15]. Kitchenham, Barbara A., et al. "What accuracy statistics really measure." IEE Proceedings-Software 148.3 (2001): 81-85.