International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Parameter Estimation of COCOMO II using Simulated Annealing

Aashima Kundu[1], Vikas Sethi[2]

[1]Research Scholar, Department of Computer Science, Panipat Institute of Engineering & technology, Samalkha, Haryana, India

[2]Assistant Professor, Department of Computer Science, Panipat Institute of Engineering & technology, Samalkha, Haryana, India

Abstract: ***Software cost estimation is process of forecasting the effort required to build up a software engineering project. This process becomes one of the principal challenges and most expensive component in the field of software .The precise results are must for proper project planning because any error may result in huge losses. The cost estimation is usually reliant upon the size estimate of the project. There are a number of diverse techniques for executing software cost estimation amid which COCOMO II is commonly used because of its lucidity and simplicity. Effort estimation models are based on various soft computing techniques such as neural network, tabu search, genetic algorithm, fuzzy logic modeling etc. for finding the precise predictive software development effort and time estimation. But the author intends to use the Simulated Annealing approach to handle these models and will show potential advantages in solving the problem. This technique will minimize functions of various variables. This technique will be applied to arbitrary combinatorial problems. The COCOMO II model predicts software development effort in Person Months (PM) and project duration in months. This work aims to propose simulated annealing for optimizing current coefficients of COCOMO II model to achieve more accuracy in estimation of software development effort.***

Keywords: COCOMO II; Effort Multipliers; Scale Factors; Size; Simulated Annealing.

1. Introduction

The process of prediction of the effort required to develop a Software Engineering project is called Software cost Estimation. Software cost estimation appears to be a simple concept but it is difficult and complex in reality [1]. Cost estimations are required throughout the lifecycle of software projects. It is known that cost of software project depends on the nature & characteristics of project where as the accuracy of estimation purely depends upon the amount of reliable information available about the product to be developed [3]. Various researchers are constantly working for the development of new techniques for software cost estimation. Most of the Software cost estimates are based on algorithmic models, expert judgment (machine learning methods) [2]. Estimation accuracy is affected by modelling accuracy. To find a good model for the software estimation is the most important objective of software engineering community. In the midst of those methods, COCOMO II (Constructive Cost Model) is the most frequently used since of its simplicity for estimating the effort in person-month for the project at diverse stages [4].

In this paper, COCOMO II model used the most commonly and broadly used simulated annealing approach for optimizing the current coefficients that estimate the optimized predictive effort required for the development of software project. Simulated annealing is a heuristic algorithm that exploits similarity between the way in which a metal cools and freezes into a minimum energy crystalline structure (annealing process) and the search for a minimum in a more general system. In this heuristic approach, the solution arbitrarily goes to in its neighbourhood with the probability which is determined by Metropolis principle while the temperature of the system decreases slowly and when the annealing temperature is closing to zero, the solution will stay at the global best solution in a high probability [5].

2. COCOMO II Model

The COCOMO'81 model is a regression based software estimation model. It was developed by the Barry Bohem in 1981 and it was thought to be the most popular of all traditional cost estimation models [4]. The acronym COCOMO stands for Constructive Cost Model. Here Constructive word states that the model gives a helping hand to the estimator for the better insight of the complexities of software job to be done & also with the openness of the model estimator can know exactly why the model has given the estimate it does . With the growing development environment, cocomo'81 failed to match the necessities in late 1990's. There when in 1997 COCOMO II was published [4]. The competency of COCOMO II is size measurement (in KLOC), Function Points, or Object Points. COCOMO II model adjusts for software reuse and reengineering. This new model served as structure for an immense current data collection and analysis effort to further refine and calibrate the model's estimation capabilities [6]. This model has three sub-models given below:

APPLICATION COMPOSITION MODEL: Suitable for early phases or spiral cycles.

EARLY DESIGN MODEL: This model is for next phases or spiral cycles. It includes discovery of architectural alternatives. Here one can get rough estimate of cost and duration of project before the determination of entire architecture.

POST- ARCHITECTURE MODEL: It is the most detailed model & is used after the overall architecture of the project has been designed. It provides more accurate information on inputs of cost driver and facilitates more precise cost estimations [7].

COCOMO II illustrates 17 Effort Multipliers (EMs) and 5 scale factors that are used in Post architectural model. Table 1 summarizes COCOMO II cost drivers/effort multipliers [1].

Table 1: COCOMO II 17 effort multipliers [1]

Cost Driver	Description
SCED	Schedule
SITE	Multi site development
RUSE	Reusability
DOCU	Documentation needs
CPLX	Product complexity
TIME	Execution time
STOR	Storage
PVOL	Platform volatility
ACAP	Analyst capability
APEX	Application experience
PLEX	Platform experience
LTEX	Language and tool experience
PCON	Personnel continuity
TOOL	Software tools used
PCAP	Programmer capability
DATA	database size
RELY	Reliability

Table 2: COCOMO II 5 Scale Factors [1]

Scale factor	Description
PREC	Precedentedness
FLEX	Development flexibility
RESL	Risk resolution
TEAM	Team cohesion
PMAT	Process maturity

According to the paper [9], COCOMO II post architecture model calculates the software development effort (in PERSON MONTH) with the help of following equation:

$$\text{Effort (PM)} = A \times (\text{SIZE})^{E} \times \Pi_i \, EM_i \quad \text{..... [9](1)}$$

Where, A - Multiplicative constant having value 2.94 that scales the effort according to specific project conditions.

Size - estimated size of the project in Kilo Source Lines of Code (KSLOC) or Unadjusted Function Points (UFP).

E - is an exponential factor that accounts for the relative economies or diseconomies of scale encountered as the software project increases its size.

EM_i - are Effort Multipliers where i =1, 2, 3….17.

The coefficient E is decided by weighing the predefined scale factors & adding them using following equation:

$$E = B + 0.01 \sum_j SF_j \quad \text{..... [9] (2)}$$

Where, B = 0.91

SFj = are Scale Factors where j = 1, 2…5.

The development time TDEV is calculated from the effort according to the following equation:

$$\text{TDEV} = C \times (\text{Effort})^{F} \quad \text{..... [9] (3)}$$

Where, C = 3.67

The coefficient F is determined in a similar way as the scale exponent by using following equation:

$$F = D + 0.2 \times 0.01 \times \sum_j SF_j \quad \text{..... [9](4)}$$

Or

$$F = D + 0.2 \times (E - B) \quad \text{..... [9](5)}$$

Where, D= 0.28

The equations for effort and schedule are given as follows:

$$\text{Effort} = 2.94 \times (\text{Size})^{1.1} \quad \text{..... [9](6)}$$

$$\text{Duration: TDEV} = 3.67 \times (\text{Effort})^{3.18} \quad \text{..... [9](7)}$$

COCOMO II is a clear and competent calibration procedure by combining Delphi technique with algorithmic cost estimation techniques. This model supports tools and has objective approach [9].The model has novel as well as accurate methods to use qualitative inputs in order to produce quantitative result [10]. But its limiting condition is that most of additions made are still testing and not fully calibrated till now [9].

3. Simulated Annealing

Today's effort estimation models are based on soft computing techniques as neural network, genetic algorithm, the fuzzy logic modelling, simulated annealing, tabu search etc. for finding the accurate predictive software development effort and time estimation. As there is no clear guideline for designing neural networks approach and also fuzzy approach is hard to use. Genetic Algorithm can offer some significant improvements in accuracy but due to its problem of premature convergence. Premature convergence means that a population for an optimization problem converged too early, resulting in being suboptimal. Simulated annealing technique can provide good results by overcoming this problem.

Simulated annealing *(SA)* is a random-search technique which utilizes an analogy between the way in which a metal cools down and freezes into a minimum energy crystalline structure (called as the annealing process) and the search for a minimum in a more general system, it forms the foundation of an optimisation technique for combinatorial problems[11]. It is the most widely accredited heuristic algorithm. In the process of optimization, the, solution randomly walks in its neighbourhood with a probability determined by Metropolis principle , as the temperature of system decreases slowly and when it is closing to zero, the solution will stay at the global best solution in a high probability[5]. The key feature of simulated annealing is that it provides a means to escape local optima by allowing hill-climbing moves i.e. moves which worsen the objective function value in hopes of finding a global optimum.

4. Proposed Work

A. Objective: The main aim of this research is to employ the concept of simulated annealing in order to optimize the COCOMO II PA model coefficients for achieving accurate software effort estimation and to reduce the

uncertainty of COCOMO II post architecture model coefficients i.e. a, b, c and d using SA.

B. Dataset Decription: Experiments have been conducted on Turkish and Industry data set presented by Ekananta Manalif [12] to optimize effort. The dataset consists of three variables i.e. Size in Kilo Line of code (KLOC), Actual effort and the predicted effort using COCOMO II PA model. The dataset is given in Table 3. Effort multipliers and scale factors rating from Very Low to Extra High related to fifteen projects are taken from Appendix B of [12].

Table 3: Data sets with their size and effort values [12]

Pr. No.	*Size (KLOC)*	*Actual Effort (PM)*	*COCOMO II Model Predicted Effort (PM)*
1	002.00	002.00	002.90
2	114.28	018.00	294.00
3	064.10	332.00	256.70
4	023.11	004.00	063.20
5	001.37	001.00	000.90
6	001.61	002.10	002.00
7	031.85	005.00	147.10
8	131.00	619.90	745.20
9	010.00	003.00	036.20
10	015.00	004.00	063.20
11	004.25	004.50	009.30
12	004.05	002.00	002.30
13	019.90	074.60	092.70
14	003.00	001.20	003.60
15	040.53	002.00	028.60

C: PROPOSED ALGORITHM

Simulated Annealing algorithm is proposed to optimize the COCOMOII PA model coefficients. The main steps of Simulated Annealing algorithm are:

Statement Algorithm

Select an initial solution $\omega \varepsilon \Omega$

Select temperature change counter $k=0$

Select a temperature cooling schedule, t_k

Select an initial temperature $T = t_0 >= 0$

Select a repetition schedule, M_k that defines the number of iterations executed at each Temperature, t_k

Repeat

Set repetition counter m= 0

Repeat

Generate a solution $\omega' \varepsilon N(\omega)$

Calculate $\Delta_{\omega\omega'} = f(\omega') - f(\omega)$

If $\Delta_{\omega\omega'} <= 0$, then $\omega \leftarrow \omega'$

If $\Delta_{\omega\omega'} > 0$, then $\omega \leftarrow \omega'$ with probability $\exp(-\Delta_{\omega\omega'} / t_k)$

$m \leftarrow m+1$

Until $m = M_k$

$k \leftarrow k+1$

Until stopping criterion is met

This formulation results in $M_0 + M_1 + \ldots\ldots\ldots + M_k$ total iterations being executed, where k relates to the value for t_k at which the stopping criteria is met. In supplement if $M_k = 1$ for all k then the temperature changes at each iteration.

Here Ω be the *solution space* (the set of all possible solutions). The *objective function is* defined on the solution space. Our goal is to find a global minimum ω^*, such that for all the objective function must be bounded to ensure that ω^* exists. $N(\omega)$ is defined to be the *neighbourhood function* for $\omega \varepsilon \Omega$, therefore are associated with every solution, $\omega \varepsilon \Omega$ are neighbouring solutions, $N(\omega)$ that can be reached in a single iteration of a local search algorithm. Simulated annealing starts with an initial solution, $\omega \varepsilon \Omega$. Neighbouring solution $\omega' \varepsilon N(\omega)$ is then generated (either randomly or by using some pre-specified rule). Simulated annealing is based upon the Metropolis acceptance criterion, which prototypes how a thermodynamic system moves from the current solution (state) to a candidate solution in which the energy content is being lessened. The candidate solution, is accepted as the current solution based on the probability of acceptance [13].

P {Accept ω' as next solution} =
$\exp\{-(f(\omega') - f(\omega)) / t_k\}$ if $f(\omega') - f(\omega) > 0$
$\{1\}$ if $f(\omega') - f(\omega) <= 0$

D: RESULT ANALYSIS

The main purpose of the Experiment performed is to lessen the vagueness of current COCOMO II post architecture coefficients i.e. a, b, c and d and get the best software effort estimation results alike to actual effort using simulated annealing algorithm. The anticipated algorithm is tested on Turkish and industry dataset on 15 different projects. The seventeen cost drivers/effort multipliers and five scale factors are taken from [13]. The working is implemented on NetBeans IDE 8.0.

Current COCOMO II PA coefficients are:
a= 2.94, b= 0.91, c= 3.67, d= 0.28.

After much iteration we get the best result. Simulated annealing is a stochastic algorithm, which means that it uses random numbers in its execution. Therefore every time one runs the program, one might turn up with a different result. It generates a sequence of solutions, each one derived by trivially amending the previous one, or by declining a new solution and falling back to the previous one devoid of any change This technique is better than GA as this technique also accepts worst solution which will in turn responsible for the best solution.

A solution set of individuals is received from which best individual with best fitness function value is chosen. The resulting optimized COCOMO II PA coefficients by simulated annealing are as:
a=2.786, b=0.624, c=3.506, d=0.237.

The following Table 4 illustrates the contrast among the Actual Effort values and Estimated Effort values for the last five project dataset using the Simulated Annealing Algorithm optimized and current COCOMO II PA model coefficients. At the same time, in the results obtained using the coefficients optimized by simulated annealing algorithm, the error is much lower, but it still perseveres.

It can be seen from the Table 4 that effort which is calculated by optimizing the coefficients of COCOMO II by using simulated annealing technique provides the results which are much nearer to actual effort and much better than current COCOMO II PA predicted effort.

Table 4: Predicted values of effort using Simulated Annealing

Pr. No.	*Project Size (KSLOC)*	*Actual Effort (PM)*	*Calculated Effort (PM) using coefficients optimized by Simulated annealing*	*Calculated Effort (PM) using COCOMO II model current coefficients*
11	4.25	4.50	05.23	09.30
12	4.05	2.00	01.88	02.30
13	19.90	74.6	79.44	92.70
14	3.00	1.20	01.05	03.60
15	40.53	2.00	01.77	28.60

The graphical contrast among three effort values described in Table 4 is given away in following Figure 1. It clearly reveals that optimized coefficients by simulated annealing produces more accurate results than the old coefficients. So, Simulated Annealing technique can offer some significant improvements in precision and has the aptitude to be a legitimate additional tool for the software effort estimation.

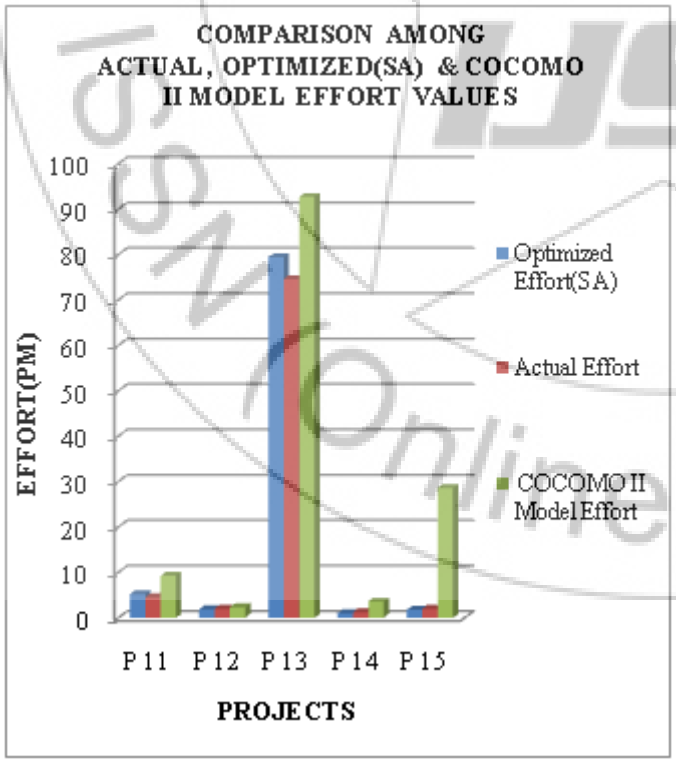

Figure 1: Graph showing comparison

5. Conclusion

Precise software cost estimation is an imperative issue in project planning. It has been seen that simulated annealing aims to optimize the COCOMO II PA model coefficients for achieving accurate software effort estimation and to reduce the vagueness of COCOMO II post architecture model coefficients i.e. a, b, c and d. The above stated results clearly depict that applying simulated annealing method to software effort estimation is a viable approach to deal with the problem of uncertainty. Furthermore, the Simulated Annealing approach presents better estimation accuracy as contrasted to COCOMO II model. Some of the advantages of SA include -

(1) Its convergence to the optima is 'good' even if the initial guesswork is distant from optima.

(2) It statistically guarantees attaining an optimal solution.

6. Future Scope

This research indicates directions for further research. The proposed framework can be analyzed in terms of feasibility and acceptance in the industry. Trying to improve the performance of existing methods and introducing the new methods for estimation based on today's software project requirements can be future works in this area. So the research is on the way to combine different techniques like Tabu Search & Simulated Annealing for calculating the best estimate. The deployment of Simulated Annealing for other applications in the software engineering field can also be investigated in the future.

References

[1] Maged A. Yahya, Rodina Ahmad, Sai Peck Lee, "Effects of Software Process Maturity on COCOMOII"s Effort Estimation from CMMI Perspective" In 2008 IEEE, Department of Software Engineering, University of Malaya, pp. 255-256

[2] Nasser Tadayon, "Neural Network Approach for Software Cost Estimation" In 2005 IEEE, Department of Computer and Software Engineering, Embry Riddle Aeronautical University,*pp.1-6.*

[3] Petr Musilek, Witold Pedrycz, Nan Su, "On the Sensitivity of COCOMO II Software Cost Estimation Model" In 2002 IEEE COMPUTER SOCIETY**,** *Dept. of Electrical and ComputerEngineering, University of Alberta, pp1-7.*

[4] Iman Attarzadeh, Siew Hock Ow, "Proposing a New High Performance Model for Software Cost Estimation" Second International Conference on Computer and Electrical Engineering, University of Malaya, In 2009 IEEE, pp no. 34-37.

[5] Mitat Uysal, "Estimation of the Effort Component of theSoftware Projects Using Simulated Annealing Algorithm", In 2008 World Academy of Science, Engineering and Technology, Department of Computer Engineering, DoğuşUniversity, Acıbadem, İstanbul, Turkey,pp 234-237

[6] Barry Boehm, Chris Abts, Sunita Chulani, "Software development cost estimation approaches –A survey" Annals of Software Engineering 10, 2000, pp. 182-190

[7] Nancy Merlo–Schett, "Seminar on Software Cost Estimation" Requirements Engineering Research Group, Department of Computer Science, WS 2002/2003, pp. 3-19.
[8] Chander Diwaker, Astha Dhiman, Sanjeev Dhawan, "Size and Effort Estimation Techniques for Software Development" International Association of Scientific Innovation and Research, Vol. 1 & 2, No. 4, ISSN: 2279-0071, pp. 35-37
[9] Bogdan Stępień, "Software Development Cost Estimation Methods and Research Trends" Computer Science, Vol. 5, 2003, pp. 68-82.
[10] Bradford K. Clark, "Cost Modeling Process Maturity-COCOMO 2.0",In 1996 IEEECenter for Saftware Engineering Department of Computer ScienceUniversity *o€* Southem Califarnia ,pp 355-357
[11] Franco Busetti, "Simulated annealing overview" pp-1-10.
[12] Ekananta Manalif, "Fuzzy Expert-COCOMO Risk Assessment and Effort Contingency Model in Software project Management" For the degree of Master of Engineering Science, The School of Graduate and Postdoctoral Studies, The University of Western Ontario London, Ontario, Canada
[13] Darrall Henderson, Alan W. Johnson, "THE THEORY AND PRACTICE OF SIMULATED ANNEALING", *Department of Mathematical Sciences,United States Military Academy,chapter 10, pp 287-291*