Application of Hybrid Techniques to Forecasting Accurate Software Cost Estimation

V. Venkataiah, Ramakanta Mohanty, M. Nagaratna

Abstract— In a competitive business world, software development is a challenging task at the primary stage of the life cycle, due to in-complete raw material and dynamically changing environment of technology in the development of the software industry. As a result inferiority software product delivered to the customer. Hence, facing a lot of problems, and wasting of time. In fact, the software quality, the budget, effort and timeliness of the development of the software product are often crucial forms of an organization to achieve success. Moreover, the interaction between vendor and development process is important. A considerable amount of models has been proposed over the most recent 3 decades. One of them the common model is the Construct Cost Mod-el in this area, and it is quite straight forward method to estimate of effort at an initial stage of software development. Shockingly, the COCOMO strategy neglected to manage the certain nonlinearity and the connection between the attributes of the project effort. In this article, we propose the application of hybrid methodology for tuning parameters of the COCOMO model which give an accurate estimated cost for project development. The COCOMO 81, IBMDPS, COCOMO NASA 2 and DESHARNAIS are used to test the performance of the proposed model.

Keywords: Particle Swarm Optimization (PSO), Construct Cost Model (COCOMO), K-Means Algorithm, Software Cost Estimation (SCE).

1. INTRODUCTION

In the present scenario modern organizations take software development as an important activity. The success of an or-ganization is achieved by which has put effort in proper order, less budget and maintain quality. SCE is the process by which predicts procedures and resources are required to complete the product within the allotted time frame. The cost of software development indirectly depends on the effort which is estimated considered the important factors are size, the project productivity and multiplier factors are cost drivers. The effort can be measured in terms of Man-Months (MM). The size can be measured as Kilo Lines of Source Code (KLOSC). Basically, Software cost influenced due to cost factors such as the level of the reliability desired, the programmer's capability, the software tools used and so on. All the above factors have incorporated an inherent uncertainty within the SCE. In order to address the inherent behavior of the factors into the estimated cost of software. Authors have been proposed various systems can be categorized into two classifications are algorithmic and non-algorithmic. The primary classification is algorithmic models are, broadly representing to in numerical and built on the factual examination of past experience. These are Software Life Cycle Management, Function Point, regression models and Constructive Cost Model (COCOMO). The COCOMO model was developed by the Barry Bohem best known and which is suitable for prediction models. The COCOMO model is used to calculate time, effort and the budget of a software product. There are different COCOMO versions of the models are Basic, Intermediate and Detailed. The mathematical representation of the basic COCOMO model equation is given below:

$$\text{Effort}(E) = c * (\text{size})d \qquad (1)$$

Where c, and d are constant parameters usually analyzed by the regression tool based on past experience. The size measured as KLOSC (Kilo Lines of Source Code) in a programming language, E is measured as Person-Months and its value directly proportional to the size and complexity of the project. These methods were not succeeding to deal with the inherent nonlinearity nature and intercommunication between the individualities of the project and effort. Reasoning, the Learning and Knowledge representation kind of methods are called non-algorithmic. Basically, these methods working principle based on the terminologies like 'learning from experience' or trail by case studies. For e.g. Expert Judgment, Analog, Price to Win, Delphi mode, Simulated Neural Networks, Fuzzy Logic, Genetic Algorithms and so on. The models were contributing work which is not filling the gap be-tween cost factors. We need an effective model for tuning the parameters of the COCOMO effort estimation model and other Software cost estimation models are presented in Table 5. These models have been produced using a substantial number of finished programming activities and applications to seek how extended sizes mapped into the project effort.

In this article, we propose a hybrid model which at first used to make groups on which tuning the constants of the COCOMO model. In area 2 Literature Survey. In the area 3 exhibits a overview of utilized strategies. In segment 4 methodology and execution measurements. In segment 5 results and discussions and in segment 6 conclusion.

2. LITERATURE SURVEY

In recent years, the software cost estimation process is emerging because of the software projects are not completed within the stipulated time bound, low quality and increase the complexity of their implementation.

Revised Version Manuscript Received on March 10, 2019.

V. Venkataiah, CMR College of Engineering & Technology, Medchal, Hyderabad, Telangana ,India
.(E-Mail:venkat.vaadaala@gmail.com)
Ramakanta Mohanty, Keshav Memorial Institute of Technology, Hyderabad, Telangana, India.
M. Nagaratna, JNTUH College of Engineering, Kukatpally, Hydreabad, Telangana, India

We need an elaborate a over-view of the literature which provides knowledge of tuning pa-rameter prediction using different techniques as well as the particle swarm optimization. Gharehchopogh et al. [1] Proposed a hybrid approach which is used to tuning the parameters of COCOMO II. This methodology was tested by evaluation metric, utilized COCOMO 81 data set collected from the literature. It was observed that the proposed method gives the best results compared to other models. Patil et al. [3] Developed hybrid technique which was used to optimize the weights of NN integrated with Principle Component Analysis (PCA) used for mapping exact input signals along with their weights as inputs to NN. Attarzadeh et al. [4] Proposed a novel ANN is utilized to translate input information and improve the imprecision of the properties of the product advancement item. The proposed model gives us 8.36% improvement in the forecast of estimation precision contrasted with the COCOMO II model. Huang et al. [5] Proposed a novel Neuro-Fuzzy approach which exhibits fuzzy behavior to interpret impression attributes of neurons and produced best results the COCOMO model. Harish [6] Estimated effort in person hours using fuzzy functional points which were employed to represent as input to the system. Andreou et al. [7] Used fuzzy decision trees for analysis on giving data to predict accuracy in software cost estimation for the project resource allocation and control. Satyananda Reddy et al. [8] RBFN used to get a functional approximation using data which collated from literature for predicting accuracy effort estimation. Pahariya J.S et al. [9] Employed computational intelligence approaches for predicting accuracy in cost estimation. Performance of proposed model tested on standard benchmark data set using RMSE metric. It was observed that 5% of accuracy improved. Attarzadeh et al. [10] Employed fuzzy logic systems to predict the software effort es-timation by using COCOMO dataset. Vishal et al. [11] Used an optimized fuzzy logic based framework for software development effort estimation prediction. Lin et al. [12] Developed a Genetic Algorithms Toolbox combined with Support Vector regression. Here, basically SVR used to make classification of observations on which Genetic Algorithms employed to optimize the parameters with the help of genetic operators like mutation, selection and crossover. Kumar et al. [13] Proposed a fuzzy system which handles ambiguous, imprecision and vagueness in order to improve the performance of the system.

In our proposed procedure, we propose a cross breed calculation to anticipate exact programming cost estimation. Essentially, the k-means display used to shape groups dependent on the mean of properties. After, the analyze can be done with speed of particles in the PSO. More than a few ages, just the most idealistic particles can transmit the data to different particles. The PSO is extremely straightforward, looking procedure is exceptionally quick and it gives improved enhancement. Subsequently, PSO can be more computationally effective than GAs at times and it is balanced to utilize this calculation in a field of programming cost estimation.

3. EMPLOYED TECHNIQUES

3.1. Particle Swarm Optimization

It is a subset of natural optimization and has capable of doing optimization. It is used to solving the non-linear and multidimensional problems effectively [16]. The PSO was proposed by James Kennedy and Russel Ebhart has the nature of stochastic optimization based on social simulation models. The basic principle of PSO is all particles of swarm traveling into the problem domain, learns it from the environment and gives better solutions. In PSO, each particle considered as a candidate solution in a problem space. All the candidate solutions are inter-preted by the fitness function which is an object function of the problem statement.

PSO initially starts with a cluster of arbitrary particles and then begin searching for an optimum solution by updating their gener-ations. Each particle in problem space is calculated with the help of the two fitness values in any iteration. Initially, best value of each particle must be memorized as Pbest. The second, best value is Gbest is choosing from the all the Pbest values which is an optimal solution for a given problem. In a PSO, each particle moving in a direction with velocity and the position of the particle is based on the velocity and inertia weight w of the particle.

The mathematical representation of the particle position

$$v[i] = w * v[i] + c_1 * r1 * (Pbest[i] - current[i]) + c_2 * r2 * (Gbest - current[i]) \quad (2)$$

$$current[i] = current[i] + v[i] \quad (3)$$

Where v is particle velocity, current is the present position, r1 and r2 random numbers are uniform distribution, w is the inertia weight, and c1, c2 are the intellectual and social parameters. The Particle speeds on each measurement clipped to a most extreme speed v_{max} .

3.2. K-Means Clustering Algorithm

The best clustering algorithm is called K-Means algorithm which is simple, efficient and faster. It is easy to understand and implement. The basic concept is randomly selected initialization centers from the given data, and then calculate the distance using the Euclidian distance between data points in each group and the cluster center. If the distance is smaller than the other cluster center that can be classified into the various group. The algorithm can be described as follows:

1. K is the numbers of clustered to be formed.
2. K initial group centers Randomly generated
3. Calculate the distance centers of K group and each data point.
4. To form cluster from the each data
5. Update clustering centers
6. Repeat 3 to 5 until the stop condition.
7. End

4. PROPOSED METHODOLOGY

In this article, we collected data sets from the literature. The first dataset COCOMO 81 (Boehm's 1981) including 63 instances and 17 attributes. 15 attributes for effort multipliers, one for the SLOC and one for the actual development effort. Second dataset is IBM Data Processing Services (DPS) prepared by Matson . Including 24 projects developed from the third generation languages. Five features which essential to play a greater role to estimate the effort required to complete any project are first one is the input count (IC), second one is the output count (OC), third the query count (QC), fourth the file count (FC) and fifth the adjustment factor (AF). Third dataset is the COCOMO NASA 2 including 60 projects and 17 attributes. 15 attributes are contributing for effort multipliers, and the rest of the attributes are SLOC and actual development effort. Fourth dataset Desharnais was collected by Desharnais (1981). This data set includes 81 projects, 24 projects developed from third generation languages and five numeric features that may affect on project effort, and then we performed data normalization by the logarithm method, thereafter on which the following proposed methodologies are applied. Here, the hybrid proposed approach employed to predict accuracy estimation for developing software cost. The proposed model uses K-Means procedure for clustering observations. Thereafter, PSO used to optimize the COCOMO model parameters.

K-Means Algorithm: K-Means step-by-step process described in the following steps.

Input: Data set contains N observations are size and actual effort.

Output: get K clusters from observations.

1. Initial centroids formed from N observations with K given values.
2. Relegate each estimation of the dataset to the bunch for which the separation between the esteem and the relating centroid is limited. A Euclidian distance formula is used for the distance evaluation.

$$D(size_i, effort_i) = \sqrt{|(size_i - size_c)|^2 + |(effort_i - effort_c)|^2} \quad (4)$$

Here $size_i$ and $effort_i$ designate the value being evaluated and $size_c$ and $effort_c$ indicate the centroids of cluster c.

3. Determine the new centroid of the cluster:

$$centriod(size_C, effort_C) = (\Sigma_{i \in cluster} \frac{size_i}{N}, \Sigma_{i \in cluster} \frac{effort_i}{N}) \quad (5)$$

4. Repeat steps 2 to 4 until the values get the fixed clusters values from the clusters.
5. End.

PSO Algorithm: The PSO implementation described below steps, which applies to the clusters of data values obtained from the K means algorithm

Input: Software Project size, Actual Effort and EAF

Output: Optimized COCOMO parameter values for Effort Calculate.

1. Initialize n particle with random velocity v and random position p of tuning parameters and initialize inertia weight with 0.5, cognitive factor c1 and social behavior factor c2 with 2.0. Range of velocities between $[-v_{max}, v_{max}]$
2. For each particle with tuning parameter values
3. Calculate the fitness value: The fitness function is MARE (Mean Absolute Relative Value)
4. Locate the nearby best position of every particle by contrasting the wellness esteem and the old neighborhood best position of the particle. In the event that the new esteem promising, at that point refreshes the nearby best position.Global best which is from the all local best particles.
5. Particle position and velocity are updated using equation (2&3)
6. Repeat 4 to 6
7. Global solution parameters considered as the optimal solution.
8. Stop

After, the following criteria are used to evaluate the performance of the proposed approach

$$MARE = Mean\left[\frac{abs(ME - EE)}{ME}\right] \quad (6)$$

Where ME: Measured Effort

EE: Estimated Effort

Root Mean Square Error (RMSE) can be expressed as follows.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(ME_i - EE_i)^2} \quad (7)$$

ME: Measured Effort, EE: Estimated Effort and 'n' are a number of instances in a given dataset.

5. RESULTS AND DISCUSSIONS

In this study, we collected COCOMO 81, IBMDPS, COCOMO NASA2, and DESHARNAIS datasets from literature. We normal-ized dataset by using the logarithm normalization method. We employed K-Means clustering techniques to all the four normalized datasets. By employing K-Means clustering, we analyzed the data sets into three different clusters using Weka 3.6.13 Tool and Java language. Thereafter, we employ PSO to find out the tuning parameters of four data sets. The parameters used in software cost estimation for PSO to find out the tuning parameters are presented in Table 2 and the PSO implemented in "c " language.

From Table 3, we employed PSO on COCOMO 81 dataset to evaluate the different tuning parameter values for three different clusters are i.e. for '0' the cluster a=0.743619, and b=0.928690 for the effort cost estimation model. Further, these values are used for measuring the 'measured efforts' and 'estimated efforts' of software cost estimation. For the

cluster '1', the results of tuning parameters obtained from the PSO i.e. a=1.153772, and b=0.773839. Similarly, for the cluster '2', the results of tuning parameters obtained by employing PSO are i.e. a= 0.300495, and b= 1.263765 respectively. The MARE values for COCOMO 81 datasets are 0.065410, 0.025226, and 0.037114. The mean value of COCOMO 81 is at 0.042583.Again, we simulated PSO on IBMDPS data set to find out the different tuning parameter values for three different clusters are as i.e. for the '0' cluster a= 0.502369, and b= 1.045276, for the clus-ter '1' as a= 1.560866, and b= 0.369047 and for the cluster '2' as a=0.019362, and b=2.271792 respectively. The MARE values for IBMDPS datasets are 0.000000, 0.057492, and 0.376338. The mean value of IBMDPS as 0.14461 is shown in Table 4.

Subsequently, we also worked PSO on COCOMO NASA 2 dataset to evaluate the different tuning parameter values for three different clusters are as i.e. for the '0' cluster a= 2.608457, and b= 0.637812, for the effort cost estimation model. Further, these values are used for measuring the 'measured efforts' and 'estimated efforts' of software cost estimation for the cluster '1' the results of tuning parameters obtained from the PSO i.e. a= 8.498737, and b= 0.134458 and for the cluster '2' as a=2.824568, and b=0.563850 respectively, for measuring the 'measured efforts' and 'estimated efforts' of software cost estimation. The MARE values for the COCOMO NASA 2 datasets are 0.000000, 0.022877, and 0.016150. The mean value of COCOMO NASA 2 is as 0.02352 presented in Table 5.

Similarly, we also employed PSO on DESHARNAIS data set to evaluate the different tuning parameter values for three different clusters are as i.e. for the '0' cluster a 9.306067 and b= 0.155009 for the cluster '1' as a=2.405843, and b=0.702451 and for the cluster '2' as a= 5.284778, and b= 0.392389 respectively, for measuring the 'measured efforts' and 'estimated efforts' of software cost estimation. The MARE values for DESHARNAIS datasets are 0.017490, 0.023404, and 0.011069. The mean value of DESHARNAIS is as 0.017321 presented in Table 6.

We compare our results with Attarzadeh [10], we found that by employing PSO, Our MARE values are outperforming the results obtained by him and the results presented in Table 7. Further, we compare our results with Sheta et. al. [15] We found that our root mean square error (RMSE) values are 0.50123, 0.60364, 0.421265 and 0.492125 in Table 8.

Table 1. Other Software Cost Estimation Models

Technique	Equation
Halstead	$E=5.2\ (KLOC)^{1.50}$
Walston-Felix	$E=0.7(KLCO)^{0.91}$
Bailey-Basili	$E=5.5+\ 0.73(KLCO)^{1.16}$
Doty (KDLOC)>9	$E=5.288+\ 0.73(KLCO)^{1.047}$

Table 2. Parameters are used in PSO

Total no. of Iterations	100
Velocity is maximized	100
Total no. of Particles	80
a is a product range	[1,10]
b is a scale factor range	[-10,10]

Table 3. The MARE values of COCOMO 81 Dataset

Clusters	Tuning parameter values a	b	MARE
Cluster0	0.743619	0.928690	0.065410
Cluster1	1.153772	0.773839	0.025226
Cluster2	0.300495	1.263765	0.037114
	Mean	0.042583	

Table 4. The MARE values of IBMDPS Dataset

Clusters	Tuning parameter values a	b	MARE
Cluster0	0.502369	1.045276	0.000000
Cluster1	1.560866	0.369047	0.057492
Cluster2	0.019362	2.271792	0.376338
	Mean	0.14461	

Table 5. The MARE values of COCOMO NASA 2 Dataset

Clusters	Tuning parameter values a	b	MARE
Cluster0	2.608457	0.637812	0.000000
Cluster1	2.405843	0.702451	0.022877
Cluster2	2.824568	0.563850	0.016150
	Mean	0.013009	

Table 6. The MARE values of DESHARNAIS Dataset

Clusters	Tuning parameter values a	b	MARE
Cluster0	9.306067	0.155009	0.017490
Cluster1	8.498737	0.134458	0.023404
Cluster2	5.284778	0.392389	0.011069
	Mean	0.017321	

Table 7. The MARE VALUES comparison

Dataset	Attarzadeh [15]	Proposed Model
COCOMO 81	0.413568	0.042483
IBMDPS	0.612654	0.14461
COCOMO NASA 2	0.468142	0.013009
DESHARNAI'S	0.312415	0.017321

Table 8. The RMSE VALUES comparison

Dataset	Sheta [31]	Proposed Model
COCOMO 81	8.131993	0.50123
IBMDPS	15.01111	0.60364
COCOMO NASA 2	8.41256	0.421265
DESHARNAI'S	10.71684	0.492125

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

6. CONCLUSION

In this article, software development is a challenging task at the primary stage of the life cycle, due to incomplete raw material and dynamically changing the environment of technology in the development of the software industry. As a result inferiority software product delivered to the customer. We propose a hybrid approach for predicting accurate cost of the software development application. The K-means clustering procedure is used to make different clusters is the first phase of hybrid approach. In the second phase, PSO used for tunning values of COCOMO on different clustered data. From our experimental results, we found that MARE and RMSE results are outperforming compared to others. This is a significant study the area of software cost estimation.

ACKNOWLEDGEMENT

We would like to thank the Dr. Hari to assist developing code and also given valuable guidelines how to write a paper.

REFERENCES

1. Gharehchopogh, F. S., Pourali, A.: A new approach based on a continuous genetic algorithm in software cost estimation. Journal of Scientific Research and Development, *2*(4), pp.87-94, (2015).
2. Kaur, M., Sehra, S. K.: Particle swarm optimization based effort estimation using Function Point analysis. IEEE 2014 International Conference on In Issues and Challenges in Intelligent Computing Techniques (ICICT), pp. 140-145, (2014).
3. Patil, L. V., Waghmode, R. M., Joshi, S. D., Khanna, V.: Generic model of software cost estimation: A hybrid approach. IEEE International Advance Computing Conference (IACC), pp. 1379-1384, (2014).
4. Attarzadeh, I., Mehranzadeh, A., Barati, A.: Proposing an enhanced artificial neural network prediction model to improve the accuracy in software effort estimation. IEEE Fourth International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN), pp. 167-172, (2012).
5. X. Huang, D. Ren, and F. L. Capretz," Improving the COCOMO model using neuro-fuzzy approach," Elsevier Journal Applied Soft Computing, vol. 7, pp. 29-40, 2007.
6. M. Harish, and B. Pradeep, "Optimization Criteria for Effort Estimation using Fuzzy Technique, " CLEL Electronic Journal, vol. 10, No.1, Paper 2, pp.1-11, June 2007.
7. A.S. Andreou, and E. Papatheocharous, "Software Cost Estimation using Fuzzy Decision Trees," 23rd IEEE/ACM International Conference on Automated Software Engineering (ASE), pp. 317-374, 15-19 Sept. 2008.
8. Ch. Satyananda Reddy, P. Sankara Rao, KVSVN Raju, and V.Valli Kumari, "A New Approach for Estimating Software Effort using RBFN network," International Journal of Computer Science and Network Security, vol.8, No.7, pp. 237-241, July 2008.
9. J. S. Pahariya, V.Ravi, and M. Car, "Software Cost Estimation Using Computational Intelligence Techniques," IEEE World Congress on Nature & Biologically Inspired Computing (NaBIC), pp. 849-854, 9-11 Dec 200
10. I. Attarzadeh, "Proposing a New Software Cost Estimation Model Based on Artificial Neural Networks," 2nd IEEE International Conference on Computer Engineering and Technology vol. 3, pp. 487-491, 2010.
11. Vishal Sharma, Hash Kumar Verma," Optimized Fuzzy Logic Based Framework for Effort Estimation in Software Development," IJCSI International Journal of Computer Science vol. 7, Issues 2, No.2, pp. 30-38, March 2010
12. Lin Jin-Cherng, Chang Chu-Ting and Shng-Yu Hung, "Research on Software Effort Estimation Combined with Genetic Algorithm and Support Vector Regression," IEEE International Symposium on Computer Science and Society vol. 4, pp. 349-352, 2011.
13. J.N.V.R. Swarup Kumar, M. Aravind, M. Vishnu Chaitanya, and G.V.S.N.R.V. Prasad, "Fuzzy logic for Software Effort Estimation Using Polynomial Regression as Firing Interval," International Journal of Computer Technology Applications, vol. 2, No.6, pp. 1843-1847, December 2011.
14. Q. Bai, "Analysis of particle swarm optimization Algorithm," Computer and Information Science, vol. 3, pp. 180-184, 2004.
15. A.F.Sheta, R. David and A.Ayesh, "Development of software Effort and Schedule Estimation models using Soft Computing Techniques," IEEE Conference on Evolutionary Computation, pp. 1283-1288, 2008.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication