International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-2, July 2019

Time of Development and Fitness Analysis of Modified COCOMOII Model for Software Projects

Sivakumar D, Sureshkumar C

Abstract*: In the management of software projects, software cost estimate is very important. The absence of precise and premature estimates is a significant source of the inability of many software initiatives. Nevertheless, it remains a challenge for the software industry to correctly estimate the time and the cost of development. It is used to predict how much effort and time the project needs. The precision of the technology proposal usually depends on the estimated time and the strength standard of the template. In this research analysis the time of development and model fitness are analyzed as a major contribution. The optimized coefficient values for A, B, C & D is used to estimate the time of development and model fitness. This experimental analysis proved that the time of development and model fitness analysis shows the great significance in the estimation and their influence in software cost estimation.

***Index Terms*: Software cost estimation, Time of development, Fitness value, Modified COCOMOII, Coefficient optimization.**

I. INTRODUCTION

In real scenario for the software development team as well as customers engaged in the project, precise software cost estimates are important. Software cost estimation is the estimate of the effort, time plan and level of personnel involved in developing software projects [1]. In software project management it is an important task that will reduce the risk. There are many parameters that affect the assessment of the undertaking and thus many methods of assessment [2], [3]. During the past ten years, numerous efforts have been undertaken to develop principles to estimate cost in terms of total efforts in a software unit such as man-months and timing for months [4].

The demands for increasingly larger and more complex software systems have stimulated these efforts, and the realization that large-scale software production projects often come with cost and schedule, which can increasingly take on the software producer. There is growing interest in developing large scale software projects [5], [6]. The ability to characterize an item is measuring the duration of enhancement and the necessary departments are gradually becoming an arduous job. It is because the instruments and methods required by software become more complicated and the contemporary software becomes increasingly expanded in order for software development to be built up and maintained, and the quality objectives are needed to create software systems in a fast-moving evolutionary algorithm situation that leads to a variety of challenging situation [7]. Due to the intangible design of the software itself, estimates of software costs are hard. In addition, during its development the software usually tends to change [8], [9]. This change might necessitate unexpected changes to the estimate. A great deal of research effort was therefore made to get better the precision of the software estimate and a lot of methods for estimating software costs were created [10], [11]. Certain methods are based on algorithms, while others are based on expert opinions.

The software engineer Barry Boehm launched the Constructive Cost Model (COCOMO) in 1981. COCOMO is an algorithmic cost estimation software model which uses a fundamental regression method with parameters derivative beginning past project information and present project properties [12]. The coefficients involved COCOMO have four parameters: A, B, C and D by examining a big amount of software projects [13], [14]. Boehm determined the characteristics of these parameters. However, these parameters could vary from one organization to another. The parameters must be adjusted to achieve the best possible results [15]. The methods of meta-heuristic algorithms are employed to optimize the parameters in order to achieve optimized solutions. Genetic algorithms (GA) are one of the significant and potent optimization algorithms. It is heuristic natural algorithms which are applied for ruling correct and fairly accurate solutions [16], [17]. GA is focused on a number of feasible alternatives to the issue, which are improving iteratively. The creation of forecast systems and wide-ranging data processing [18] is associated with the majority of GA apps. The objective of the present study is to tone COCOMO coefficients, especially COCOMO-II Coefficients of the post Architecture model using the online GA Data Series and some new data collected from the Project.

The reaming part of this paper is prearranged as follows: Segment II literature review that provides insight on previous works. Segment III presents modified COCOMO II model and functionalities. Segment IV descries coefficient optimization with GA. Segment V presents the experiment results and discussions to evaluate the proposed optimization. Segment VI provides the conclusion and areas of the future work.

Revised Manuscript Received on 30 July 2019.
* Correspondence Author
Sivakumar D*, Department of Computer Science and Engineering, RajaRajeswari College of Engineering, Bengaluru, Karnataka, India-560074.
Sureshkumar C, Department of Information Technology, KGiSL Institute of Technology, Coimbatore, Tamilnadu, India-641035.

II. LITERATURE REVIEW

Patil et al.[19] Developed hybrid technique for optimizing NN's weights in combination with Principle Component Analysis (PCA) used for mapping precise input indicators along with NN inputs. The ANN proposal is used to translate input information and to increase the imprecision of the features of the product development item. Attarzadeh et al.[20]. The proposed model provides an increase of 8.36% prediction of estimation accuracy compared to COCOMO II.

In the year 2014 M. Madheswaran, et.al,[21] have focused on the development of a model of software effort estimation centered on artificial neural networks. The main intend of this project was framed to improve the perfection of the COCOMO model. They used neural networks with multiple layers to provide accommodation for the representation and its parameters to calculate the development effort of software. The complex system network is sophisticated by the use of background spreading algorithm through the processing and comparison of the prediction of the network with the actual effort of a number of training models. The networks have been trained and tested using the COCOMO data set and the outcome shows that the proposed model generates more precise inference than that of the COCOMO model.

The expansion and changes made in the COCOMO II model is provided by Vu. Nguyen (2014) [22] following evaluation of different popular cost estimated software maintenance models. He has shown that there are weaknesses in the different existing models regarding the limited choice of data and the limited range of maintenance activities. In order to evaluate and assess software maintenance projects, the author has evaluated and presented an extended version of the COCOMO II models. To develop the assessment model, the author has employed regression method. In the case of organizations, when data are not adequate to calibrate various evaluation models, the proposed model can be used. This expanded model of COCOMO II also takes the deleted code SLOC metric into its size metric.

In the year 2012, Syed Ali Abbas, et.al, [23] provides baselines on the development of the COCOMO suite model improvements and distinctions between the derivatives to scientists, software organizations and professionals. The authors have stated that over the years, different software cost estimation processes have been supported through a variety of models and that COCOMO (Constructive Cost Model) is considered to be one of the best schedule estimates.. This paper discusses an overview of the COCOMO suite model and provides the appropriate environment for their use, concept behind their development, and procedures for their development.

Parida et. al. [24] Analyzed designs which are normally used for estimating the magnitude and effort estimation of the back-end portion. In addition to the development of the front end, development of the back end is also crucial. The price of the back-end portion is determined on both sides by the volume of each data item and by the organizational dimension of the database. The experimental results showed how the effort estimate for the background part has to be carried out by means of ER diagrams and how the productivity of any business software can be increased in calculating the actual cost of the background part. M. Pauline et., al [25] has developed an improved system which is designed to minimize effort by improving adjustments to functional sizing techniques. This paper presents fuzzy methods for the classification of quality models as a basis. Empirical validation of multipliers of COCOMO II software development work is analyzed and ratings are defined for cost drivers.

N. Shivakumar et., al [26] examined the attempt of the algorithmic procedure and non-algorithmic procedure and demonstrated that the neuroscience-based integrated estimate is more effective for the assessment procedure than the algorithmic technique. In different procedures the achievement of the estimate is dependent on the precision and stabilization of the technique. In comparison to the methods Doty, Bailed and Halstead, they implemented the fuzzy method into existing databases. Finally, the minimum rate of error by Fuzzy was shown in comparison with other methods by experimental results.

Thamarai et., al [27] used the Differential Evolution technique, a minimum mistake percentage is determined in comparison with current techniques in the suggested technique, in order to pick the appropriate design from a number of historical activities which are comparable to a fresh venture.

The proposal of four effort metrics for the software database or background, which can be derived in line with the complexity level of the ER diagnosis, was made by Samaresh Mishra et. al. [28].

A model for estimating activity was suggested with the assistance of an ER diagram by Samaresh Mishra et. al. [29]. The model takes into account the complexity of the ER graph of the software background in order to evaluate the effort. The metrics used in the model are the relation sizes, the size of the entity and semantic integrity limitations..

III. MODIFIED COCOMO II MODEL

The software project development exertion is premeditated by means of the subsequent mathematical equation in the post architecture modelling method COCOMO II

$$Effort = AS^{E}\prod_{i=1}^{n} EM_{i} \qquad (1)$$

Where,

'A' is the constant multiplication
'S' is the software project size
'E' is an exponential variable for factors of scale and economy related to the magnitude of the software project
'EM' is the multiplier of efforts

The exponential variable 'E' is dictated through measuring of predefined scale factors (SF_i) and summing them by using the associated mathematical equation

$$E = B + 0.01\sum_{j=1}^{5} SF_{j} \qquad (2)$$

The time of development (TDEV) gets from the effort, as indicated by the accompanying equation

$$TDEV = C(Effort)^F \qquad (3)$$

The latest calibration approach shows that the ' C ' multiplier is equal to 3.67 and that the ' F ' ratio is similarly measured in the scale form of the associated mathematical statement

$$F = D + 0.02\sum_{j=1}^{5} SF_i \qquad (4)$$

$$= D + 0.2(E - B) \qquad (5)$$

When each variable and multiplier is provided with their nominal values, mathematical statement is provided for the effort and the time schedule as

$$Effort = 2.94S^{1.1} \qquad (6)$$

Time schedule,

$$TDEV = 3.67Effort^{3.18} \qquad (7)$$

The COCOMO II method with algorithmic assessment techniques is evident and strong.

IV. COEFFICIENT OPTIMIZATION WITH GA

In this research analysis the technical genetic algorithm is used as an optimisation method and the detailed step-by-step process is given below.

There are four model variables, verbalized by means of four genes, and the whole number is null to nine. The first gene is regarded null in this strategy and the three remaining genes are seen as dividing elements [30].

Algorithm: The Pseudo code for Genetic Algorithm

1: Set the required Parameters
2: Select the proper encoding methods
3: Create the initial population
4: **While** *i< IterationMax and FitnessBest<FitnessMax then*
5: Complete the Fitness estimate
6: Carry out the Selection procedure
7: Carry out the Crossover procedure
8: Carry out the Mutation process
9: **End** while
10: Based on maximum fitness value Decode the individual
11: **revisit** the finest solution

1. Create the individual arbitrary
2. For each undertaking "j" in the dataset, the project effort will be shown using optimized models coefficients of an individual ' i'.
3. Estimate the project fitness 'j' for every individual fitness 'i' through one of the subsequent equations

$$FP_{ij} = \frac{\left|TDEV_a^{\,j} - TDEV_e^{\,ij}\right|}{TDEV_a^{\,j}} \qquad (8)$$

$$FP_{ij} = \frac{\left|ASPE^{\,j} - ESPE^{ij}\right|}{ASPE^{\,j}} \qquad (9)$$

'i ' - is the quantity of individual

'j' - is the quantity of project

$TDEV_a^{\,j}$ - is an definite project 'j' development duration

$TDEV_e^{\,ij}$ - is the estimated project development 'i'

$ASPE^j$ - is definite effort of software project

$ESPE^{ij}$ - is the anticipated or predicted effort of software project, for every entity 'i' in j^{th} project

By understanding the legitimate effort for software projects is greater than the estimated or anticipated effort in the project, then fitness has an optimistic importance and if the actual effort in the project is less than that predicted or estimated, fitness has a negative value [31].

4. Every candidate fitness value is ascertained as the regular assessment of all software projects; meticulous fitness of a candidate is calculated from through the step 2 and step 3. The fitness convenience importance anticipated to be minimized [32].

$$F_i = \frac{1}{n}\sum_{i=1}^{n} F_{ij} \qquad (10)$$

'F' – represents the fitness

'I' – is the candidate numeric representation

'j' – is the project number

'n' – represents the number of projects taken

5. The situation to stop the iteration specifies when the algorithm should be completed. An alternative option is the finest (minimal) and median survival of a population.
6. This experiment uses the tournament selection process as a method for selection.
7. A uniform crossover approach is used to generate fresh people through people chosen in a preceding phase.
8. The mutant mode of intervention is the voluntary decision of none or tiny amount of people. The likelihood of choosing the person should be small is obviously around 10%. Uniform crossover approach is used to produce fresh people through people chosen in the earlier stage for each person, random scheme of the mutating genes.
9. The fresh population is established with prominent people chosen by step 6 between parents and children.

V. RESULTS AND DISCUSSION

All exertion was conducted in all the way through the methodology chosen for this research study. The updated genetic algorithm parameters, developmental technology, are used to improve the results in the COCOMO II after architecture model which include a consistent crossover approach, a choice of competition and a speed of transition of 0.15 estimated at 250, population age 50 and mutation rate 0.01. The highest outcome is achieved in countless phases and an outcome range is obtained which gives the overwhelming person the answer with the transcendent fitness importance.

Best fittest value among all available individual value is,

2 7 9 8 | 0 8 1 3 | 3 5 1 8 | 0 2 9 7

On this basis, the resulting optimized COCOMO II coefficients for the post-architectural model are evaluated and have accompanying qualities, a=2.798, b=0.813, c=3.518, d=0.297. Elderly COCOMO II post architectural model coefficients have the accompanying qualities a=2.94; b=0.91; c=3.67; d=0.28. With the assistance of the optimized coefficients of COCOMO II, the mean relative estimation value or best fitness estimation of the dataset record is 3.79. It is 1.42 times below the average comparative value of the present coefficient COCOMO II post-architecture model, which is equivalent to 5.99.

Table I provides the time of development (TDEV) measured using three different methodologies with COCOMO II data Set. This statistical data indicates that the TDEV measured with modified COCOMO II with optimized coefficient model provides enhanced prediction when compared to all other approaches considered for this experimentation. TDEV measured using modified COCOMO II with optimized coefficient model for project id 9 is 23.3 and it is very close to actual 24.1. In project id 50 the original TDEV is 20.3 the coefficient optimized model calculated TDEV is 19 which is very closer to the actual TDEV. Whereas for the same project id TDEV calculated using COCOMO II model is 12 and the modified COCOMO II model are 11.2. Time of Development (TDEV) is measured with the new project data for validation purpose. The table II clearly shows that TDEV estimated with the modified COCOMO II with optimized coefficient model produced uppermost enhancement in prediction accuracy. The effective estimation of the time necessary to build up the software project will limit the cost of the software project.

Table I. Time of development (TDEV) in modified COCOMO II with optimized coefficient model for COCOMO II data set

S.No	Project ID	KLOC	Actual TEDV	Estimated TDEV		
				COCOMO II	Modified COCOMO II	Modified COCOMO II With Optimized Coefficients
1	1	113	38.4	29.4	30.3	36.5
2	5	16	14.3	8	11	12.4
3	9	30	24.1	17.9	19.6	23.3
4	11	32	24	16	19	22.6
5	26	48	18.5	8.3	12.4	15.9
6	34	23	14.2	6	8	13
7	42	45.5	21	13.2	15	19
8	47	23	15.6	7.9	9.3	13.4
9	50	24	20.3	12	11.2	19
10	51	10	16.2	9	10.3	14
11	54	4.4	10.6	5	7.1	8.7
12	56	27	21.1	13.2	15.6	18.4
13	61	28	16.4	10	10.8	13.7

In this regard the TDEV estimated for the project with project ID P3 is 2.8 and it is closer to actual TDEV 3.4 of the same project. The TDEV estimated for the project with project ID P6 is 21.8 whereas the actual TDEV is 23. These type data are considered as the proven evident in the enhanced estimation and it helps the project administrator to maintain the price of the software project in the specified limit.

Table II. Time of development (TDEV) in modified COCOMO II with optimized coefficient model for new project data

S.No	Project ID	KLOC	Actual TDEV	Estimated TDEV		
				COCOMO II	Modified COCOMO II	Modified COCOMO II With Optimized Coefficients
1	P1	23	19.2	12.1	13.5	17
2	P2	98	57	43.4	46	82.4
3	P3	10	3.4	1.4	2.1	2.8
4	P4	130	96.4	87	89	93.2
5	P5	79	20.7	12	15.3	19
6	P6	86	23	13.4	16	21.8
7	P7	45	14.3	9	11.2	13

Individual project fitness is evaluated and is listed in Table III. The statistical data output of this experiment clearly indicates that the modified COCOMO II with optimized coefficient model produced very less fitness value when compared to the other models considered for the analysis of this experiment.

Table III. Individual project fitness based on TDEV for COCOMO II dataset

S.No	Project ID	Estimated Fitness Value		
		COCOMO II	Modified COCOMO II	Modified COCOMO II With Optimized Coefficients

1	1	23	21	5
2	5	44	23	13
3	9	26	19	3
4	11	33	21	6
5	26	55	33	14
6	34	58	44	8
7	42	37	29	10
8	47	49	40	14
9	50	41	45	6
10	51	44	36	14
11	54	53	33	18
12	56	37	26	13
13	61	39	34	16
Model Fitness		41.46	31.07	10.76

If the fitness value is less the model prediction is closer to the actual. Very less model fitness value 10.76 is achieved in modified COCOMO II with the optimized coefficient approach, whereas the same value in the COCOMO II model was 41.46. This result shows the clear improvement in the estimation of TDEV.

Table IV. Individual project fitness based on TDEV for new project data

S.No	Project ID	Estimated Fitness Value		
		COCOMO II	Modified COCOMO II	Modified COCOMO II With Optimized Coefficients
1	P1	37	30	11
2	P2	24	19	45
3	P3	59	38	18
4	P4	10	8	3
5	P5	42	26	8
6	P6	42	30	5
7	P7	37	22	9
Model Fitness		35.86	24.71	14.14

The new project data collected for testing the method is used for evaluating the individual project fitness and the model fitness. Both the fitness has very less value when compared the same with another model. The model fitness value 14.14 is measured using modified COCOMO II with optimized coefficient model and is the least value when compared to the remaining two fitness value 35.86 and 24.71. While considering the individual project fitness also the project with project id P7 has the fitness value of 9 in the modified COCOMO II model with optimized coefficients where as the COCOMO II model fitness value is 37 and the modified COCOMO II model fitness value is 22.

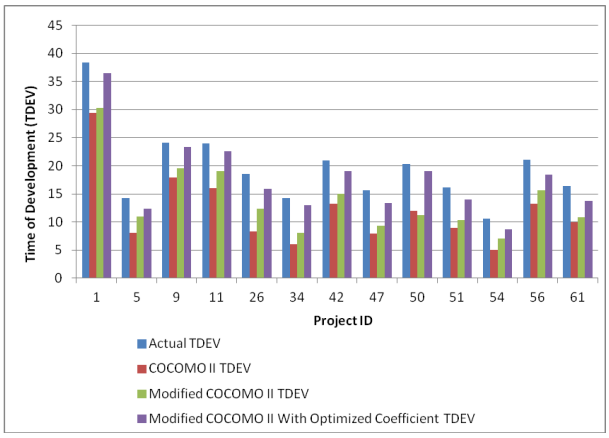

Fig. 1 Time of development (TDEV) based on modified COCOMO II with optimized coefficient model for COCOMO II dataset

Fig.1 is the pictorial representation of the time of development of the COCOMO II data set. This figure depicts that the modified COCOMO II with optimized coefficient model outperformed in producing the exact schedule of estimation when compared to the other two models considered in this experiment. When compared to the TDEV calculated with COCOMO II model and the modified COCOMO II model the coefficient optimized modified COCOMO II model predicted TDEV is closer to the actual TDEV of the software projects considered in the research study and analysis.

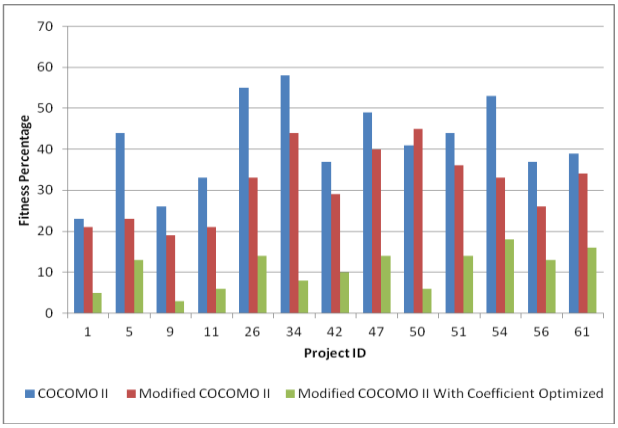

Fig. 2 Fitness representation of modified COCOMO II with optimized coefficient model for COCOMO II dataset

In view of better mechanism in representing results the pictorial representation is a more powerful tool when compared to tabular form with statistical values. The Fig.2 clearly indicates that the modified COCOMO II with optimized coefficient model produced enhanced fitness with less difference in actual and estimated TDEV values. The project id 9 has very less value of fitness this indicates that the project id 9 provides very high accuracy in its prediction of effort and TDEV. Almost all testing data shows that the fitness value is very less in coefficient optimized modified COCOMO II model

when compared to all other model fitness values.

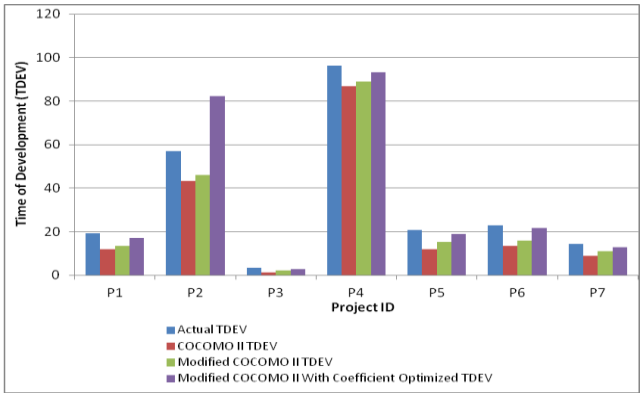

Fig. 3 Time of development (TDEV) based on modified COCOMO II with optimized coefficient model for new project data.

This modified COCOMO II with optimized coefficient model again proved with the new project data considered, that is this model provides a schedule of estimation which is very closer to the actual schedule. Even though it is not the exact schedule of estimation, it is a better result when compared to the COCOMO II based schedule estimation.

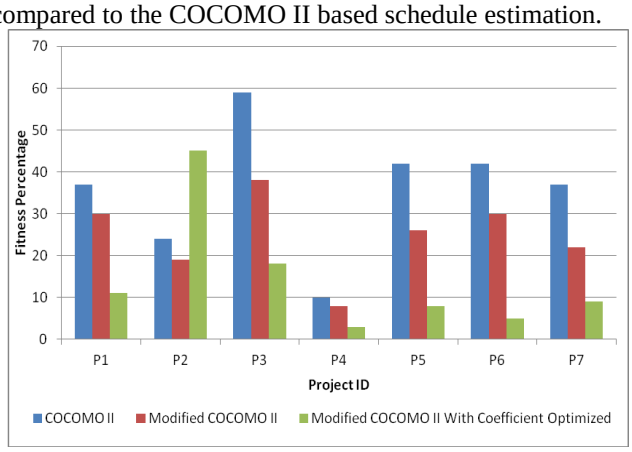

Fig. 4 Fitness representation of modified COCOMO II with optimized coefficient model for new project data

Fig.4 provides the graphical representation of individual project fitness in estimating the effort and time of development of software project. The fitness value of all the projects has very less value and hence this modified COCOMO II with optimized coefficient model produced a better fitness in estimating the effort and the schedule.

VI. CONCLUSION

This experimental assessment is based on the COCOMO II dataset and current design information, the updated COCOMO II with an optimized coefficient model. In comparison with the consequence calculated using present COCOMO II PA model equations, the assessed growth and fitness moment is capable to produce high assessment ability. The findings show that the energy and moment measured using the optimized coefficient with the genetic algorithm method in the vast bulk of instances gives superior outcomes. This practice is further extended to optimize mobile software forecast precision.

REFERENCES

1. B.W. Boehm, E. Horowitz, R. Madachy, D. Reifer, B. K. Clark, B. Steece, A. W. Brown, S. Chulani, and C. Abts, “Software cost estimation with COCOMO II”, Prentice Hall, 2000.
2. R. D. Banker, S. M. Datar, C. F. Kemerer, “Factors affecting software maintenance productivity: an exploratory study”, Proceedings of 8th international conference on Information system. Pittsburgh S, pp. 160–175.
3. J. E. Matson, B. E. Barrett, J. M. Mellichamp, “Software Development Cost Estimation Using Function Points”, IEEE Transactions on Software Engineering, vol. 20, pp. 275-287, 1994.
4. De Lucia, E. Pompella, S. Stefanucci, “Assessing effort estimation models for corrective maintenance through empirical studies”, Journal of Information and Software Technology, vol. 47, pp. 3–15, 2005.
5. P. Suri, P. Ranjan, “Article: Comparative Analysis of Software Effort Estimation Techniques”, International Journal of Computer Applications, vol. 48, pp. 12-19, 2012.
6. 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.
7. B. Jamil, J. Ferzund, A. Batool, S. Ghafoor, “Empirical Validation of Relational Database Metrics for Effort Estimation”, In 6th International Conference on Networked Computing, IEEE, ISSN: 2074-9015, pp. 1-5, May-2010.
8. G. R. Weckman, H. W. Paschold, J. D. Dowler, H. S. Whiting, W. A. Young, Using Neural Networks with Limited Data to Estimate Manufacturing Cost, Journal of Industrial and Systems Engineering 3(4): 257-274, 2010.
9. C. S. Reddy, P. S. Rao, KVSVN Raju, V. V. Kumari, A New Approach for Estimating Software Effort Using RBFN Network, IJCSNS International Journal of Computer Science and Network Secruity 8(7): 237-241. 2008.
10. A. Idri, A. Zakrani, A. Zahi, Design of Radial Basis Function Neural Networks for Software Effort Estimation, IJCSI International Journal of Computer Science Issues 7(3): 11-17, 2010.
11. S. J. Huang, C. Y. Lin, N. H. Chiu, Fuzzy Decision Tree Approach for Embedding Risk Assessment Information in to Software Cost Estimation Model , Journal of Information Science and Engineering, No. 22, pp. 297-313, 2006.
12. Mrinal Kanti Ghose, Roheet Bhatnagar and Vandana Bhattacharjee, “Comparing Some Neural Network Models for Software Development Effort Prediction”, IEEE, 2011.
13. Kaczmarek J., Kucharski M., “Size and Effort Estimation for Applications Written in Java”, Journal of Information and Software Technology, Vol. 46, No. 9, pp. 589-60, 2004.
14. Abbas S.A., et. al. “Cost Estimation-A Survey of Well-known Historic Cost Estimation Techniques”, Journal of Emerging Trends in Computing and Information Sciences, Vol. 3, No. 2, pp. 612-636, 2012.
15. Subramanian G.H., Pendharkar P.C. and Wallace M., An Empirical Study of the Effect of Complexity, Platform, and Program Type on Software Development Effort of Business Applications, Empirical Software Engineering Journal, Vol. 11, pp. 541-553, 2006.
16. Jorgen M., Sjoberg D.I.K, “The Impact of Customer Expectation on Software Development Effort Estimates”, International Journal of Project Management, Elsevier, pp. 317-325, 2004.
17. M. Padmaja, D. Haritha. “Software Effort Estimation Using Grey Relational Analysis”, International Journal of Information Technology and Computer Science (IJITCS), ISSN: 2074-9015, Vol. 9, Issue: 5, pp. 52-60, May-2017.
18. S. Goyal, A. Parashar, “Machine Learning Application to Improve COCOMO Model using Neural Networks”, International Journal of Information Technology and Computer Science (IJITCS), Vol. 3, pp. 35-51, March-2018.
19. 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).
20. 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).
21. M. Madheswaran, D. Sivakumar, “Enhancement of prediction accuracy in COCOMO model for software project using neural network”, IEEE, pp. 1-5, 2014.

22. Vu. Nguyen, “Improved size and effort estimation models for software maintenance”, An Unpublished Ph.D. Dissertation, University of Southern California, Los Angeles, CA, viewed 20 December 2014.
23. Syed Ali Abbas, Saleem Ullah Lar, Xiaofeng Liao, Raja Aftab Naseem, “Software Models, Extensions and Independent Models in Cocomo Suite: A Review”, Journal of Emerging Trends in Computing and Information Sciences, vol. 3, pp. 683-693, 2012.
24. S. Parida, S. Mishra, “Review report on Estimating the Back-End Cost of Business Software Using ER- Diagram Artifact”, International Journal of Computer Science and Engineering Technology (IJCSET), ISSN: 2229-3345, Vol. 2, Issue: 1, pp. 233-238, March-2014.
25. M. Pauline, Dr. P. Aruna, Dr. B. Shadaksharappa ,Comparison of available Methods to Estimate Effort, Performance and Cost with the Proposed Method‖, International Journal of Engineering Inventions, Volume 2, Issue 9, PP: 55-68, May 2013
26. N. Shivakumar, N. Balaji and K. Ananthakumar, A Neuro Fuzzy Algorithm to Compute Software Effort Estimation‖, Global Journal of Computer Science and Technology: Software & Data Engineering, Volume 16 Issue 1 Version 1.0 Year 2016
27. I. Thamarai and S. Murugavalli, An Evolutionary Computation Approach for Project Selection in Analogy based Software Effort Estimation‖, Indian Journal of Science and Technology, Vol 9(21), DOI: June 2016.
28. S. Mishra, P. Pattnaik, R. Mall, “Early Estimation of Back-End Software Development Effort”, International Journal of Computer Applications, ISSN: 0975–8887, Vol. 33, Issue: 2, pp. 6-11, November-2011.
29. S. Mishra, R. Mall, “Estimation of Effort Based on Back-End Size of Business Software Using ER Model”, In 2011 World Congress on Information and Communication Technologies, IEEE, ISSN: 2074-9015, pp. 1098-1103, December-2011.
30. A L I. Oliveira, P L. Braga , R M F. Lima, M L. Cornélio, “GA-based Method for Features Selection and Parameters Optimization for Machine Learning Regression applied to Software Effort Estimation”, Information and Software Technology, Elsevier, Vol. 51, Issue: 11, pp. 6129-6139, November-2010.
31. F. Leung , H. Lam, S. Ling, “Tuning of the Structure and Parameters of a Neural Network Using an Improved Genetic Algorithm”, IEEE Transactions on Neural Networks, ISSN: 1045-9227, Vol. 14, Issue: 1, pp. 79-88, February-2003.
32. S. Bilgaiyan, K. Aditya, S. Mishra, M N. Das, “Chaos-based Modified Morphological Genetic Algorithm for Software Development Cost Estimation”, Progress in Computing, Analytics and Networking, Vol. 710, pp. 31-40, April-2018.

AUTHORS PROFILE

Dr Sivakumar D received his Ph.D (CSE) in 2016, ME (CSE) in 2007, M.Tech (Power Systems) in 2005, and BE (EEE) in 1999. He is currently working as an Associate Professor in the Department of Computer Science and Engineering, RajaRajeswari College of Engineering, Bengaluru. He also worked as a Software Trainer. He is having more than 15 years of Experience in the field of engineering education. He is a reviewer for the International Journal of Technological Research & Innovative Solutions (IJTRIS). His areas of interest in Teaching, Learning & Research are: Software Engineering | Computer Networks | Big Data Analytics | Soft Computing: Neural Network, Genetic Algorithm | Data Base Management | Programming and Data Structures | Java | C | C++ | MATLAB. He is Life Time Member of ISTE.

Mr. Sureshkumar C received his Master of Engineering in Computer science and engineering from Annamalai University, Chidambaram. He is presently working as an assistant professor in the department of Information Technology, KGiSL Institute of Technology, Coimbatore. He has 13 years of industry, research and teaching experience. He has published papers in International conferences (Paper indexed in IEEE Xplore digital library) and Scopus indexed journal. His research interests includes | Data analytics | Software Engineering | Software Testing | Distributed Computing | Cloud Computing.