Journal of Computer Science 6 (2): 117-125, 2010
ISSN 1549-3636

A Novel Algorithmic Cost Estimation Model Based on Soft Computing Technique

Iman Attarzadeh and Siew Hock Ow
Department of Software Engineering, Faculty of Computer Science and Information Technology,
University of Malaya, 50603 Kuala Lumpur, Malaysia

Abstract: Problem statement: Software development effort estimation is the process of predicting the most realistic use of effort required for developing software based on some parameters. It has always characterized one of the biggest challenges in Computer Science for the last decades. Because time and cost estimate at the early stages of the software development are the most difficult to obtain and they are often the least accurate. Traditional algorithmic techniques such as regression models, Software Life Cycle Management (SLIM), COCOMO II model and function points, require an estimation process in a long term. But, nowadays that is not acceptable for software developers and companies. Newer soft computing techniques to effort estimation based on non-algorithmic techniques such as Fuzzy Logic (FL) may offer an alternative for solving the problem. This work aims to propose a new fuzzy logic realistic model to achieve more accuracy in software effort estimation. The main objective of this research was to investigate the role of fuzzy logic technique in improving the effort estimation accuracy by characterizing inputs parameters using two-side Gaussian function which gave superior transition from one interval to another. **Approach:** The methodology adopted in this study was use of fuzzy logic approach rather than classical intervals in the COCOMO II. Using advantages of fuzzy logic such as fuzzy sets, inputs parameters can be specified by distribution of its possible values and these fuzzy sets were represented by membership functions. In this study to get a smoother transition in the membership function for input parameters, its associated linguistic values were represented by two-side Gaussian Membership Functions (2-D GMF) and rules. **Results:** After analyzing the results attained by means of applying COCOMO II and proposed model based on fuzzy logic to the NASA dataset and created an artificial dataset, it had been found that proposed model was performing better than ordinal COCOMO II and the achieved results were closer to the actual effort. The relative error for proposed model using two-side Gaussian membership functions is lower than that of the error obtained using ordinal COCOMO II. **Conclusion:** Based on the achieved results, it was concluded that, using soft computation approaches such as fuzzy logic and their advantages, good predication; adaption; understandability and the accuracy of software effort estimation can be improved and the estimation can be very close to the actual effort. This novelty model will lead researchers to focus on benefits of non-algorithmic models to overcome the estimation problems.

Key words: Software project management, software cost estimation models, COCOMO II, soft computation model, fuzzy logic

INTRODUCTION

Software development effort estimation deals with the prediction of the probable amount of time and cost required to complete the specific development task. Generally, software development effort estimations are based on the prediction of size of software, which is a very difficult task in the sense that estimates obtained at the early stages of development life cycle are inaccurate because not much information of the system is available at that time. These estimations are essential for software developers and their companies, because it can provide cost control, delivery accuracy, among many other benefits for them. To the present time, many quantitative models of software cost estimation have been developed. Most of these models are based on the size measure, such as Line of Code (LOC) and Function Point (FP), obtained from size estimation. It is obvious that the accuracy of size estimation directly impacts the accuracy of cost estimation. Based on this context, new

Corresponding Author: Iman Attarzadeh, Department of Software Engineering,
Faculty of Computer Science and Information Technology, University of Malaya,
50603 Kuala Lumpur, Malaysia

alternative such as fuzzy logic can be a good choice to estimate task effort in software development.

Software Development Effort Estimation: Software developers always interest to know the time estimation of software tasks. It could be done by comparing similar tasks that have already been developed. Although, estimating task has an uncertain nature, as it depends on several and usually not clear factors and it is hard to be modeled mathematically. Software schedule and cost estimation supports the planning and tracking of software projects. Effectively controlling the expensive investment of software development is of high importance (MacDonell and Gray, 1997; Jingzhou and Guenther, 2008; Kastro and Bener, 2008; Strike *et al*., 2001). The reliable and accurate cost estimation in software engineering is an ongoing challenge (Kastro and Bener, 2008) due to it allows for considerable financial and strategic planning. Software cost estimation techniques can be classified as algorithmic and non-algorithmic models. Algorithmic models are based on the statistical analysis of historical data (past projects) (Strike *et al*., 2001; Hodgkinson and Garratt, 1999), for example, Software Life Cycle Management (SLIM) (Schofield, 1998) and Constructive Cost Model (COCOMO) (Putnam, 1978; Boehm, 1981).

Non-algorithmic techniques are based on new approaches such as, Parkinson (Boehm, 1981), Expert Judgment, Price-to-Win and machine learning approaches (Schofield, 1998). Machine learning is used to group together a set of techniques that represent some of the facets of human mind (Schofield, 1998; Huang and Chiu, 2009), for example regression trees, rule induction, fuzzy systems, genetic algorithms, artificial neural networks, Bayesian networks and evolutionary computation. The last five of these approaches are classified as soft computing group. The importance of algorithmic and non-algorithmic estimation techniques will briefly discuss in the Algorithmic models.

Algorithmic models: Some of the famous algorithmic models are: Boehm's COCOMO'81, II (Boehm *et al*., 2000), Albrecht's Function Point (Boehm *et al*., 2000; Boehm, 1995) and Putnam's (1978) SLIM. All of them require inputs, accurate estimate of specific attributes, such as Line Of Code (LOC), number of user screen, interfaces and complexity, which are not easy to acquire during the early stage of software development. Models based on historical data have limitations. Understanding and calculation of these models are difficult due to inherent complex relationships between the related attributes, are unable to handle categorical data as well as lack of reasoning capabilities (Boetticher, 2001).

Besides, attributes and relationships used to predict software development effort could change over time and/or differ for software development environments (Srinivasan and Fisher, 1995). The limitations of the algorithmic models led to the exploration of the non-algorithmic techniques which are soft computing based.

Non-algorithmic models: In 1990's non-algorithmic models was born and have been proposed to project cost estimation. Software researchers have turned their attention to new approaches that are based on soft computing such as artificial neural networks, fuzzy logic models and genetic algorithms. Neural networks are able to generalize from trained data set. A set of training data, a specific learning algorithm makes a set of rules that fit the data and fits previously unseen data in a rational manner (Srinivasan and Fisher, 1995; Idri *et al*., 2006; Liu and Yu, 2005). Some of early works show that neural networks are highly applicable to cost estimation include those of Venkatachalam (1993) and Krishna and Satsangi (1994). Fuzzy logic offers a powerful linguistic representation that able to represent imprecision in inputs and outputs, while providing a more knowledge based approach to model building. Research shows that fuzzy logic model achieved good performance, being outperformed in terms of accuracy only by neural network model with considerably more input variables.

Hodgkinson and Garratt represented that estimation by expert judgment was better than all regression based models (Hodgkinson and Garratt, 1999). A marriage between neural networks and fuzzy logic, is named Nero-fuzzy, was introduced into cost estimation in (Hodgkinson and Garratt, 1999). Nero-fuzzy systems can take the linguistic attributes of a fuzzy system and combine them with the learning and modeling attributes of a neural network to produce transparent, adaptive systems. As it mentioned above, Fuzzy Logic has been proposed to some models to overcome the uncertainly problem. However, there is still much uncertainty as to what prediction technique appropriate to which type of prediction problem (Burgess and Lefley, 2001). Choosing a suitable technique is a difficult decision that requires the support of a well-defined evaluation scheme to rank each prediction technique as it applies to any prediction problem.

This study proposed an effective model based on fuzzy logic and COCOMO II model to overcome the uncertainly problem and acquiring the better results. Because of the importance of COCOMO Model and fuzzy logic system in our research we provide a brief overview on them in this study.

Related work: MacDonell and Gray (1997) compared popular techniques in software effort estimation as regression techniques, Function Point Analysis (FPA), fuzzy logic and neural network. Their results showed that fuzzy logic model achieved good performance. They introduced an application of fuzzy logic to effort estimation. They developed a tool, FUzzy Logic SOftware MEasuring (FULSOME) (MacDonell and Gray, 1997), to assist software managers in making estimation. In FULSOME model, the two most important variables were selected: complexity adjustment factor and unadjusted function point. Then a triangular membership functions were defined for the small, medium, large intervals of size, complexity and effort.

Fei *et al*., have tried to fuzzify some of the existing algorithmic models in order to handle uncertainties and imprecision problems in such models (Fei and Liu, 1992). They have done the first realization of the fuzziness on COCOMO model. They found it is unreasonable to assign a determinate number for it, because an accurate estimate of Delivered Source Instruction (KDSI) cannot be made before starting the project. Ryder (1998) applied fuzzy modeling technique to COCOMO and the Function-Points models. Idri *et al*. (2006); Huang *et al*. (2006) investigated the application of fuzzy logic to the cost drivers of intermediate COCOMO model.

Musflek *et al*. worked on fuzzifying basic COCOMO model without considering the adjustment factor. They introduced f-COCOMO model, the size input into the COCOMO model also the coefficients related to the development mode are assigned by a fuzzy set. In another research, Kumar *et al*. (Krishna Kumar and Satsangi, 1994) applied fuzzy logic in Manpower Buildup Index (MBI) of Putnam estimation model. MBI was based upon 64 different rules. The results showed it can be effectively applied to software project management. Fuzzy logic also had been applied to the non- algorithmic models to overcome the uncertainly of the models.

Molokken *et al*. (2003); Idir *et al*, proposed a combination of fuzzy logic and estimation by analogy. Estimation by analogy is one of the classified techniques of expert-based estimation method. It is a type of Case-based Reasoning (CBR) method. The fuzzy analogy for software cost estimation had also been applied to web base software. Venkatachalam (1993) applied artificial neural network to cost estimation. Neural network is able to generalize from trained data set. Over a set of training data, neural network learning algorithm constructs mappings that fit the data and fits previously unseen data in a reasonable way.

Research had also been done to combine fuzzy logic with neural network. A new system based on fuzzy logic, neural network and COCOMO II proposed (Huang and Chiu, 2009). This system Based on COCOMO II post architecture model, the input of neuro-fuzzy COCOMO consists of size and 22 cost drivers (5 scale factors plus 17 effort multipliers). In summary, fuzzy logic has been proposed to algorithmic and non-algorithmic models in the pursuit of achieving better estimation results. Nevertheless, there is still much uncertainty as to what estimation technique suits which type of estimation problem Huang and Chiu, 2009. Choosing between the different techniques is a difficult decision that requires the support of a well-defined evaluation method to show each estimation technique as it applies to any estimation problem.

MATERIALS AND METHODS

Problem Statement: Understanding and calculation of models based on historical data are difficult due to inherent complex relationships between the related attributes, are unable to handle categorical data as well as lack of reasoning capabilities. Besides, attributes and relationships used to estimate software development effort could change over time and differ for software development environments. In order to address and overcome to these problems, a new model with accurate estimation will be considerable.

The COCOMO II model: The COCOMO model is a regression based software cost estimation model. It was developed by Bohem (1995; 2000) in 1981 and thought to be the most cited, best known and the most plausible (Fei and Liu, 1992) of all traditional cost prediction models. COCOMO model can be used to calculate the amount of effort and the time schedule for software projects. COCOMO 81 was a stable model on that time. One of the problems with using COCOMO 81 today is that it does not match the development environment of the late 1990's. Therefore, in 1997 COCOMO II was published and was supposed to solve most of those problems. COCOMO II has three models also, but they are different from those of COCOMO 81. They are (Ryder, 1998; Huang *et al*., 2006):

- Application composition model-suitable for projects built with modern GUI-builder tools. Based on new Object Points
- 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. Based on Unadjusted Function Points or KSLOC

- Post-Architecture Model-The most detailed on the three, used after the overall architecture for the project has been designed. One could use function points or LOC as size estimates with this model. It involves the actual development and maintenance of a software product

COCOMO II describes 17 cost drivers that are used in the Post-Architecture model (Ryder, 1998). The cost drivers for COCOMO II are rated on a scale from Very Low to Extra High in the same way as in COCOMO 81. COCOMO II post architecture model is given as:

$$\text{Effort} = A \times [\text{size}]^{B} \times \prod_{i=1}^{17} \text{Effort multiplier}_i \quad (1)$$

Where:

$$B = 1.01 + 0.01 \times \sum_{j=1}^{5} \text{Scale factor}_j$$

In Eq. 1:

A = Multiplicative constant

Size = Size of the software project measured in terms of KSLOC (thousands of source lines of code, function points or object points)

The selection of Scale Factors (SF) is based on the rationale that they are a significant source of exponential variation on a project's effort or productivity variation. The standard numeric values of the cost drivers are given in Table 1.

Fuzzy Logic: In 1965, Zadeh formally developed multi-valued set theory and introduced the term fuzzy into the technical literature (Zadeh, 1994).

Table 1: COCOMO II cost drivers

Cost driver	Range
Required software reliability (RELY)	0.82-1.26
Database size (DATA)	0.90-1.28
Product complexity (CPLX)	0.73-1.74
Developed for reusability (RUSE)	0.95-1.24
Documentation match to life-cycle needs (DOCU)	0.81-1.23
Execution time constraint (TIME)	1.00-1.63
Main storage constraint (STOR)	1.00-1.46
Platform volatility (PVOL)	0.87-1.30
Analyst capability (ACAP)	1.42-0.71
Programmer capability (PCAP)	1.34-0.76
Personnel continuity (PCON)	1.29-0.81
Applications experience (APEX)	1.22-0.81
Platform experience (PLEX)	1.19-0.85
Language and tool experience (LTEX)	1.20-0.84
Use of software tools (TOOL)	1.17-0.78
Multi site development (SITE)	1.22-0.80
Required development schedule (SCED)	1.43-1.00

Fuzzy Logic starts with the concept of fuzzy set theory. It is a theory of classes with un-sharp boundaries and considered as an extension of the classical set theory (Zadeh, 2001). The membership $\mu_A(x)$ of an element *x* of a classical set *A*, as subset of the universe *X*, is defined by Eq. 2 in below:

$$\mu_A(x) = \begin{cases} 1 & \text{if} \quad x \in A \\ 0 & \text{if} \quad x \notin A \end{cases} \quad (2)$$

A system based on Fuzzy Logic has a direct relationship with fuzzy concepts (such as fuzzy sets, linguistic variables) 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 (Zadeh, 1994). 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 was first proposed by Mamdani It has been successfully applied to a variety of industrial processes and consumer products (Zadeh, 1994). The main fours components' functions are as follows:

Step #1:

- Fuzzification: It converts a crisp input to a fuzzy set

Step #2:

- Fuzzy Rule Base: Fuzzy logic systems use fuzzy IF-THEN rules
- Fuzzy Inference Engine: Once all crisp input values are fuzzified into their respective linguistic values, the inference engine accesses the fuzzy rule base to derive linguistic values for the intermediate and the output linguistic variables

Step #3:

- Defuzzification: It converts fuzzy output into crisp output

Experimental design: The new proposed model base on COCOMO II has two input's group from COCOMO II cost drivers and scale factors and one output, effort estimation. This model covers those three fuzzy steps. It shows in Fig. 1.

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.

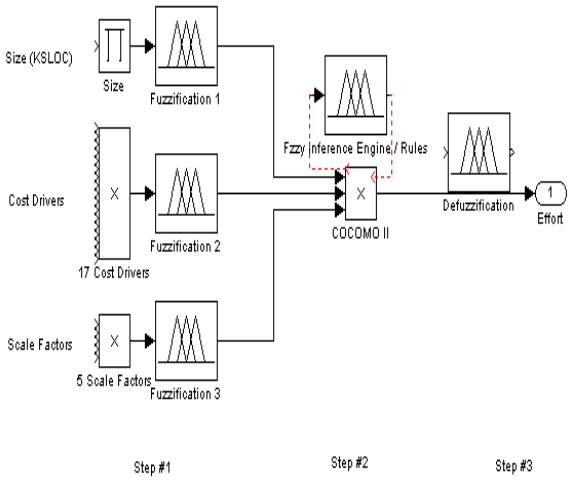

Fig. 1: The proposed model: Inputs (COCOMO II cost drivers, scale factors, Size) and Output: (effort estimation)

Traditionally, the problem of software effort estimation relies on a single (numeric) value of size and scale factors values of given software project to predict the effort. However, the size of the project is, based on some previously completed projects that resemble the current one (especially at the beginning of the project). Obviously, correctness and precision of such estimates are limited. It is of principal importance to recognise this situation and come up with a technology using which we can evaluate the associated imprecision residing within the final results of cost estimation. The technology endorsed here deals with fuzzy sets. Using fuzzy sets, size of a software project can be specified by distribution of its possible values. Commonly, this form of distribution is represented in the form of a fuzzy set. It is important that uncertainty at the input level of the COCOMO model yields uncertainty at the output (Boehm *et al*., 2000). This becomes obvious and, more importantly, bears a substantial significance in any practical endeavor. By changing input parameters using fuzzy set, we can model the effort that impacts the estimation accuracy. Obviously, a certain monotonicity property holds, which is less precise estimates of inputs give rise to less detailed effort estimates. Overlapped symmetrical two-sided Gaussian function reduces fuzzy systems to precise linear systems.

Furthermore there is a possibility when using a Two-sided Gaussian function that some attributes are assigned the maximum degree of compatibility when they should be assigned lower degrees. In order to avoid this linearity it is proposed to use more superior

Table 2: The artificial dataset generated for system validation consists of 100 data samples

No.	Mode	Size	Effort
1	1.1200	51.2500	246.5900
2	1.2000	12.5500	58.2800
3	1.0500	81.5200	550.4000
…	...	…	…
97	1.2000	56.5300	354.7300
98	1.0500	16.0400	67.1400
100	1.1200	54.1700	262.3800

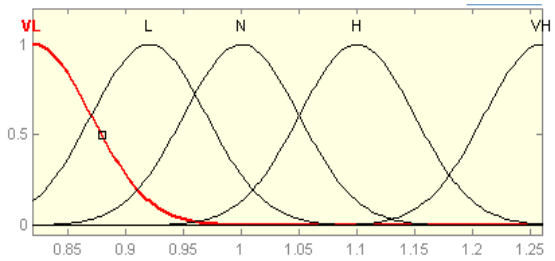

Fig. 2: Representation of RELY cost driver using Gaussian function (Input)

function i.e., Two-sided Gaussian membership function for representing inputs of the project. The Gaussian Function is represented by Eq. 3 in below:

$$\mu_{A_i}(x) = \text{Gaussian}(x, c_i, \sigma_i) = e^{\frac{-(x-c_i)^2}{2\sigma_i^2}} \quad (3)$$

Where:

c_i = The center of the ith fuzzy set

σ_i = The width of the ith fuzzy set

The processes involved in software effort estimation using FL are shown in Fig. 1. The main processes of this system include four activities: fuzzification, fuzzy rule base, fuzzy inference engine and defuzzification.

All the input variables in COCOMO II model changed to the fuzzy variables based on the fuzzification process. The terms Very Low (VL), Low (L), Nominal (N), High (H) and Very High(VH) were defined for the 22 variables, cost drivers and scale factors, in COCOMO II. For example, in the case of RELY cost driver, we define a fuzzy set for each linguistic value with a Two-sided Gaussian shaped membership function μ is shown in Fig. 2. We have defined the fuzzy sets corresponding to the various associated linguistic values for each cost driver.

In this research, a new fuzzy effort estimation model is proposed by using Two-sided Gaussian function to deal with linguistic data and to generate fuzzy membership functions and rules for cost drivers obtained from Table 2. In the next step, we evaluate the

COCOMO model using the equation 3 and cost drivers obtained from fuzzy sets (F_EM_{ij}) rather than from the classical EM_{ij}. F_EM_{ij} is calculated from Eq. 5 the classical EM_{ij} and the membership functions μ defined for the various fuzzy sets associated with the cost drivers:

$$Fuzzy_{EM_{ij}} = F(\mu^{V_i}{}_{A1}, ...\mu^{V_i}{}_{Ai}, EM_{i1} ... EM_{ij}) \quad (4)$$

For ease, F is taken as a linear function, where the μV_i Aj is the membership function of the fuzzy set Aj associated with the cost driver V_i is shown in Eq. 4:

$$Fuzzy_{EM_{ij}} = \sum_{j=1}^{k_i} \mu_{A_i}{}^{v_i} * EM_{ij} \quad (5)$$

The new fuzzy model rules contain the linguistic variables related to the project. It is important to note that those rules were adjusted or calibrated, as well as all pertinence level functions, in accordance with the tests and the characteristics of the project. In rules use the connective "and" and "or" or combination of them between input variables, as indicated in the example below. The number of rules that have used in proposed model is more than 193 rules for all input variables.

Fuzzy rules:
IF TOOL is Low TEHN effort is Low
IF PCAP is Very_Low THEN effort is Very_High
IF RESUE is Nominal THEN effort is Nominal
IF DATA is Very_High THEN effort is Very_High
…

The MATLAB Fuzzy Inference System (FIS) was used in the fuzzy calculations, in addition to the Max-Min composition operator, the Mandani implication operator and the Maximum operator for aggregation. The defuzzification of the output "effort" used the Mean Of Maximum (MOM) technique in this work because the resulting values were more appropriate when compared to the other evaluated techniques (Center Of Area (COA) and First Of Maximum (FOM)).

RESULTS AND DICUSSION

Experiments were done by taking two datasets, first one was original data from NASA dataset and second one was artificial dataset.

Datasets description: Boehm (1981) is the first researcher to look at software engineering from an economic point of view and he came up with cost estimation models from two datasets, COCOMO and COCOMO II. The COCOMO (Boehm, 1995) dataset includes 63 historical projects with 17 effort drivers and one dependent variable of the software development effort. So, the first used dataset for evaluating the proposed model is based on COCOMO model. The second attempt was to create an artificial dataset, Table 2, based on COCOMO model. The algorithm for fuzzy set learning in a Mamdani-type fuzzy system is following this four-step procedure:

- Choose a training sample and propagate the input vector across the network to get the output
- Determine the error in output and the error gradient in all the other layers
- Determine the parameter changes for the fuzzy weights and update the fuzzy weights
- Repeat until the fuzzy error is sufficiently small after an epoch is complete

Therefore, this work has used two datasets for evaluation of the proposed model. Finally, by aggregate the accuracy across all testing datasets as the mean result.

Evaluation Method: For evaluating the different software effort estimation models, the most widely accepted evaluation criteria are the Mean Magnitude of Relative Error (MMRE) and probability of a project having a relative error of less than or equal to 0.25 (Pred(l)). The Magnitude of Relative Error (MRE) is defined as follows:

$$MRE_i = \frac{|\text{Actual Effort}_i - \text{Predicted Effort}_i|}{\text{Actual Effort}_i} \quad (6)$$

The MRE value is calculated for each observation i whose effort is predicted. The aggregation of MRE over multiple observations (N) can be achieved through the Mean MRE (MMRE) as follows:

$$MMRE = \frac{1}{N}\sum_i^N MRE_i \quad (7)$$

Another measure similar to MRE, the Magnitude of error Relative to the Estimate (MER), has been proposed. Intuitively, it seems preferable to MRE since it measures the error relative to the estimate. MER uses Predicted Efforti as denominator in Eq. 6. The notation MMER is used to the mean MER in Eq. 7. However, the MMRE and MMER are sensitive to individual predictions with excessively large MREs or MERs.

Therefore, an aggregate measure less sensitive to extreme values is also considered, namely the median of MRE and MER values for the N observations (MdMRE and MdMER respectively). A complementary criterion is the prediction at level l, Pred(l) = k/N, where k is the number of observations where MRE (or MER) is less than or equal to l and N is the total number of observations. Thus, Pred(25) gives the percentage of projects which were predicted with a MRE (or MER) less or equal than 0.25.

The proposed fuzzy model was validated by two approaches. In the first approach, has used the NASA dataset that consists of 93 projects (Dataset #1). In the second approach, has used the artificial dataset that consists of 100 sample projects (Dataset #2). Then both datasets are applied to the new fuzzy model and COCOMO II model. The validation of the new fuzzy model to building trained fuzzy model for effort estimation has been done using artificial dataset and NASA dataset. The comparison between the results of NASA dataset and artificial dataset that applied on the new fuzzy model and COCOMO II model shows more accuracy in case of effort estimation by the new fuzzy model. The comparisons between results are shown in Table 3 and 4.

In this research, each dataset separately applied to the COCOMO II model and proposed model. Then for each model, the MMRE and Pred were calculated. Finally mean of those calculations are used to compare both models. The result for 193 applied projects shows the MMRE for COOCMO II model is 0.406713037 and for proposed model the value equals to 0.369637508. It shows the proposed model has MMRE less than COCOMO II model, so it means the accuracy of proposed model is better than COCOMO II. In case of Pred, the final result shows the proposed model value is 47.5% in Pred(25%) and COCOMO II value is 35% in same Pred. As it mentioned above, Pred shows the number of projects that they have MMRE lass than 25%. According to this definition, the proposed model shows better accuracy. Table 4 shows how much the proposed model is accurate than COCOMO II model.

For comparing proposed model with COCOMO model, the improvement is 12.63% based on the MMRE 0.40 and 0.36. The experimental results show that the proposed software effort estimation model shows better estimation accuracy than the other two models, i.e., COCOMO. In summary, an output with more terms or fuzzy sets provided a better performance due to the high granularity demanded from the results. Most of the sample data in the dataset with the proposed fuzzy model resulted in a more accurate estimation when compared to the COCOMO II model.

Table 3: Comparison between performance of new model and COCOMO II

		Evaluation	
Data set	Model	MMRE	Pred (25%)
Data set #1	COCOMO II	0.413812453	30%
	Proposed model	0.366545456	50%
Data set #2	COCOMO II	0.39961362	40%
	Proposed model	0.37272956	45%
Mean	COCOMO II	0.406713037	35%
	Proposed model	0.369637508	47.5%

Table 4: Accuracy of the proposed model

Model	Evaluation	MMRE
Proposed model Vs	COCOMO II	0.406713037
COCOMOII	Proposed model	0.369637508
	Improvement (%)	12.630000000

CONCLUSION

An essential issue for project managers is the accurate and reliable estimates of the required software development effort, especially in the early stages of the software development life cycle. Software effort drivers usually have properties of uncertainty and vagueness when they are measured by human judgment. A software effort estimation model utilizing fuzzy inference system can overcome these characteristics of uncertainty and vagueness exist in software effort drivers. However, the determination of the suitable fuzzy rule sets for fuzzy inference plays an important role in coming up with accurate and reliable effort estimates. Software effort estimation using fuzzy logic is an attempt in the area of software project estimation. The objective of this work is to provide a technique for software cost estimation that performs better than other techniques on a given set of test cases. This paper presented a new model for handling imprecision and uncertainty by using the fuzzy logic systems. The objective of this work is to provide a technique for software cost estimation that performs better than other techniques on the accuracy of effort estimation. This work has shown by applying fuzzy logic on the algorithmic and non-algorithmic software effort estimation models accurate estimation is achievable. The proposed fuzzy logic model showed better software effort estimates in view of the MMRE, Pred(0.25) evaluation criteria as compared to the traditional COCOMO. The above-mentioned results demonstrate that applying fuzzy logic method to the software effort estimation is a feasible approach to addressing the problem of uncertainty and vagueness existed in software effort drivers. Furthermore, the fuzzy logic model presents better estimation accuracy as compared to the NASA dataset. The utilization of fuzzy logic for

other applications in the software engineering field can also be explored in the future.

REFERENCES

Boehm B. W., 1981. Software engineering economics. Englewood Cliffs, Prentice-Hall, NJ., ISBN: 10: 0138221227, pp: 768.

Boehm, B., 1995. Cost models for future software life cycle processes: COCOMO 2.0. Ann. Software Eng. 1: 45-60.

Boehm B., C. Abts and S. Chulani, 2000. Software development cost estimation approaches-A survey. Ann. Software Eng., 10: 177-205. DOI: 10.1023/A:1018991717352

Boetticher, G.D., 2001. An assessment of metric contribution in the construction of a neural network-based effort estimator. http://sce.uhcl.edu/boetticher/SCASE01.pdf

Burgess, C.J. and M. Lefley, 2001. Can genetic programming improve software effort estimation? A comparative evaluation. Inform. Software Technol., 43: 863-873. DOI: 10.1016/S0950-5849(01)00192-6

Fei, Z. and X. Liu, 1992. f-COCOMO: Fuzzy constructive cost model in software engineering. Proceedings of the IEEE International Conference on Fuzzy Systems, Mar. 8-12, IEEE Xplore Press, San Diego, CA., USA., pp: 331-337. DOI: 10.1109/FUZZY.1992.258637

Hodgkinson, A.C. and P.W. Garratt, 1999. A neurofuzzy cost estimator. Proceedings of the 3rd International Conference on Software Engineering and Applications, (SEA'99), ePrint, pp: 401-406. http://eprints.ecs.soton.ac.uk/2659/

Huang, X., D. Ho, J. Ren and F. Capretz, 2006. A soft computing framework for software effort estimation. Soft Comput., Fusion Foundat., Methodol. Appli. J., 10: 170-177. DOI: 10.1007/s00500-004-0442-z

Huang, S. and N. Chiu, 2009. Applying fuzzy neural network to estimate software development effort. Proc. Applied Intel. J., 30: 73-83.

Jingzhou, L. and R. Guenther, 2008. Analysis of attribute weighting heuristics for analogy-based software effort estimation method AQUA+. Empiric. Software Eng. J., 13: 63-96. DOI: 10.1007/s10664-007-9054-4

Idri, A., A. Zahi and A. Abran, 2006. Software cost estimation by fuzzy analogy for web hypermedia applications. Proceedings of the International Conference on Software Process and Product Measurement, (SPPM'06), Cadiz, Spain, pp: 53-62.

Kastro, Y. and A.B. Bener, 2008. A defect prediction method for software versioning. Proc. Software Qual. J., 16: 543-562. DOI: 10.1007/s11219-008-9053-8

Kumar, S. A. Krishna and P. Satsangi, 1994. Fuzzy systems and neural networks in software engineering project management. J. Applied Intel., 4: 31-52. DOI: 10.1007/BF00872054

Liu. H. and L. Yu, 2005. Toward integrating feature selection algorithms for classification and clustering. IEEE Trans. Knowl. Data Eng., 17: 491-502. DOI: 10.1109/TKDE.2005.66

MacDonell, S.G. and A.R. Gray, 1997. A comparison of modeling techniques for software development effort prediction. Proceedings of the 1997 International Conference on Neural Information Processing and Intelligent Information Systems, Dunedin, (NIPIISD'97), Springer-Verlag, New Zealand, pp: 869-872.

Molokken, K. and M. Jorgensen, 2003. A review of software surveys on software effort estimation. Proceedings of IEEE International Symposium on Empirical Software Engineering, Sept. 30-Oct. 01, IEEE Computer Society, Washington DC., USA., pp: 223-230. http://portal.acm.org/citation.cfm?id=943636

Putnam, L.H., 1978. A general empirical solution to the macro software sizing and estimating problem. IEEE Trans. Software Eng., 4: 345-361. http://portal.acm.org/citation.cfm?id=1313641

Ryder, J., 1998. Fuzzy modeling of software effort prediction. Proceedings of IEEE Information Technology Conference, Sept. 1-3, IEEE Xplore Press, Syracuse, NY., pp: 53-56. DOI: 10.1109/IT.1998.713380

Schofield C., 1998. Non-Algorithmic effort estimation techniques. Technical Reports, Department of Computing, Bournemouth University, England. http://decgradschool.bournemouth.ac.uk/ESERG/Technical_Reports/TR98-01/TR98-01.ps

Srinivasan, K. and Fisher D., 1995. Machine learning approaches to estimating software development effort. IEEE Trans. Software Eng., 21: 126-137. DOI: 10.1109/32.345828

Strike, K., K. El-Emam and N. Madhavji, 2001. Software cost estimation with incomplete Data. IEEE Trans. Software Eng., 27: 890-908. DOI: 10.1109/32.962560

Venkatachalam, A.R., 1993. Software cost estimation using artificial neural networks. Proceedings of the 1993 International Joint Conference on Neural Networks, Oct. 25-29, IEEE Xplore Press, USA., pp: 987-990. DOI: 10.1109/IJCNN.1993.714077

Zadeh, L.A., 1994. Fuzzy logic, neural networks and soft computing. Mag. Commun. ACM, 37: 77-84. DOI: 10.1145/175247.175255

Zadeh, L.A., 2001. The future of soft computing. Proceeding of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference, (NAFIPS'01) Vancouver, Canada.