Volume 4, Issue 5, May 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Quantification of Strength of Relationships among Software Development effort Multipliers using FIS

Pardeep Bhandari
Doaba College, Jalandhar, India

Abstract : Software effort estimation is the most vital task in software development. Various model have been proposed for this purpose. This estimate is vital to win a contract, to allocate appropriate resources, to schedule the process and above all, to finish the project within approximated budget and time. Various factors have been proposed in constructive cost model of cost estimation commonly known as COCOMO. These factors affect each other apart from affecting the software effort. The quantification and comparison of interdependencies of these factors is the main task performed in this paper. Fuzzy inference system has been implemented in MATLAB to find the weights, which represents the strength of interconnections among various pairs of factors.

Keywords : software effort estimation, fuzzy inference system, COCOMO, Fuzzy Inference System.

I. INTRODUCTION

The software industry is always in search of a model for software development effort estimation. This estimate is vital to win a contract, to allocate appropriate resources, to schedule the process and above all, to finish the project within approximated budget and time. In the area of software engineering and project management, estimating the software development time and cost is the biggest challenge. As software development has become an essential investment for many organizations, accurate software cost estimation models are needed to effectively predict, monitor, control and assess software development. Accurate estimate means better planning and efficient use of project resources such as cost, duration and effort requirements for software projects especially space and military projects [1], [2].As such, estimation accuracy is a very significant issue for executives, managers, technical staff, and, particularly, practitioners who perform or rely on cost estimation [3]. Inaccuracy in software estimates has been identified as a root cause of a high percentage of failures in the software development [4, 5]. The reasons that software cost estimation is difficult and error prone include [6]:

- Vague and insufficient information is available at the early phases of software development process.
- Software cost estimation requires a significant amount of cost to perform correctly.
- The process is often done hurriedly, without an appreciation for the cost required to perform the estimate.
- Experience is required for developing estimates, especially for large projects.
- Human bias.

Numerous models have been proposed by various researchers over a period of time. This includes various variants of Constructive Cost Estimation Model (COCOMO), Halsted Model, Bailey Besli Model, Walston Model etc. COCOMO developed by B.Boehm has been a landmark in software estimation models. It is one of the most widely used and studied model. The software cost is amalgamation of many factors as expressed by effort adjustment factors in COCOMO. In this paper we have studied the interrelationships of the effort multipliers used in COCOMO using fuzzy cognitive maps.

II. SOFTWARE EFFORT ESTIMATION MODELS

There are several software cost estimation models which can be classified as algorithmic and non-algorithmic models. These models are discussed in [9,10,15,16,17]. The algorithmic models are based on the statistical analysis of historical data [7, 8], for example, Software Life Cycle Management (SLIM) [9] and Constructive Cost Model (COCOMO) [10]. Non-algorithmic techniques are based on new approaches such as, Parkinson, Expert Judgment, Price to Win, and Machine Learning approaches [11, 12]. The Machine learning approach is used to group together a set of techniques that represent some of the facts of human mind [13], 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.
Nevertheless, there is still much uncertainty as to what estimation technique suits which type of estimation problem [14]. 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.

III. THE CONSTRUCTIVE COST MODEL (COCOMO)

COCOMO model was researched and developed by Boehm [10], [12], at TRW. This model was evaluated using 63 TRW software projects where the projects were grouped into three different software domains, organic, semidetached and embedded. The COCOMO model is defined using mathematical economics equations that identify the developed time, the effort and maintenance effort. Boehm B. proposed three levels of the model: basic, intermediate, detailed.

Basic Model : The basic COCOMO model is a single-valued, static model that computes software development effort (and cost) as a function of program size expressed in estimated thousand delivered source instructions (KDSI).

$MM = a * KDSI^{b}$

Intermediate Model: The intermediate COCOMO model computes software development effort as a function of program size and a set of fifteen "cost drivers" that include subjective assessments of product, hardware, personnel, and project attributes.

COCOMO assumes that effort grows more than linearly on software size

$MM = a * KDSI^{b} * E$

where KDSI is estimated directly or computed from a FP analysis

E is the product of 15 effort multipliers (Table 3) i.e.

$$MM = a * KDSI^{b} * (EM1*EM2*EM3*\ldots\ldots\ldots)$$

a and b depend on the mode of the development. There are 152 hours per man month.

Advanced Model: The advanced or detailed COCOMO model incorporates all characteristics of the intermediate version with an assessment of the cost driver's impact on each step (analysis, design, etc.) of the software engineering process.

The three ways of estimating with increasing levels of accuracy are simple, intermediate and complex models. These three models are each defined using increasingly detailed mathematical relationship between the developed time, the effort and the maintenance effort. The estimation accuracy is significantly improved when adopting models such as the Intermediate and Complex COCOMO models.

IV. FUZZY COGNITIVE MAP

Fuzzy cognitive map was proposed by Kosko [18] to represent the causal relationship between concepts and analyze inference patterns. FCM have been successfully used to model complex systems and develop decision support systems. The FCM is a soft computing modeling methodology that follows a method similar to the human reasoning and human decision-making process. It utilizes concepts to illustrate the different aspects of the system's model and behavior and these concepts interact with each other showing the dynamics of the system. FCM structures can be used to represent qualitative and quantitative data. A FCM integrates the accumulated experience and knowledge on the causal relationship between factors/ characteristics/ components of the system.

The advantage of the FCM is due to the way it is constructed, i.e., using human experts which know the system and its behavior under different circumstances [19]. In fact, Fuzzy Cognitive Maps (FCM) could be regarded as a combination of Fuzzy Logic and Neural Networks. A Fuzzy Cognitive Map consists of nodes-concepts and arcs between concepts. Each concept represents a characteristic of the system; in general it stands for events, actions, goals, values, trends of the system that is modeled as an FCM. Each concept is characterized by a number *Ai* that represents its value and it results from the transformation of the real value of the system's variable, for which this concept stands, in the interval [-1,1]. Between concepts, there are three possible types of causal relationships that express the type of influence from one concept to the others. The weights of the arcs between concept *Ci* and concept *Cj* could be positive W_{ij}, which means that an increase in the value of concept *Ci* leads to the increase of the value of concept *Cj* , and a decrease in the value of concept *Ci* leads to the decrease of the value of concept *Cj* . Or there is negative causality i.e. value of W_{ij} negative, means that an increase in the value of concept *Ci* leads to the decrease of the value of concept *Cj* and vice versa.

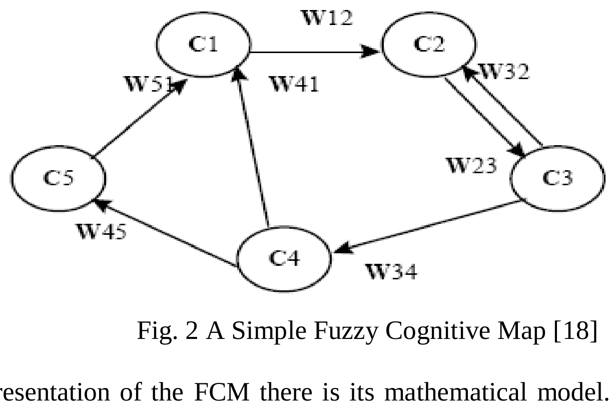


Fig. 2 A Simple Fuzzy Cognitive Map [18]

Beyond the graphical representation of the FCM there is its mathematical model. It consists of an 1x *n* state vector **A** which includes the values of the *n* concepts and an *n* x *n* weight matrix **W** which gathers the weights *Wij* of the interconnections between the *n* concepts of the FCM. The matrix **W** has *n* rows and *n* columns where *n* equals the total number of distinct concepts of the FCM and the matrix diagonal is zero since it is assumed that no concept causes itself. The value of each one concept is influenced by the values of the connected concepts with the appropriate weights and by its previous value. So the value *Ai* for each concept *Ci* is calculated by the following rule:

$$A_i = f\left(\sum_{\substack{j=1 \\ j\neq i}}^{n} A_i W_{ji}\right) + A_i^{old}$$

where *Ai* is the activation level of concept *Ci* at time t+1, *Aj* is the activation level of concept *Cj* at time t, *Ai old* is the activation level of concept *Ci* at time t, and *Wji* is the weight of the interconnection between *Cj* and *Ci* , and *f* is a threshold function.

$$A_{new} = f\left(A_{old} \circ W\right) + A_{old}$$

…

So the new state vector $\mathbf{A}_{new}$ is computed by multiplying the previous state vector $\mathbf{A}_{old}$ by the weight matrix **W**. The new vector shows the effect of the change in the value of one concept in the whole Fuzzy Cognitive Map. But, equation (2) includes also, the old value of each concept, and so the FCM possesses memory capabilities and there is a smooth change after each new cycling of the FCM.
We propose to use FCM in software effort estimation.

V. Literature Survey

J. Ryder, **"Fuzzy modeling of software effort prediction,"** Proceedings of IEEE Information Technology Conference, Syracuse, NY, pp 102-114, 1998. Author was among the initial researchers who explored the idea of using fuzzy modeling in software effort prediction. At that time two models which were in vogue were: (i) COCOMO developed by Boehm which used LOC as initial estimate and produced a nominal effort estimate using non linear equation:
Effort=a(KLOC)^b
Value of coefficients a and b depended upon the mode in which the model was applied. (shown below)

Mode	a	b
Organic	2.4	1.05
Semi-detached	3	1.12
Embedded	3.6	1.2

After nominal effort estimate, it was adjusted by multiplying with effort adjustment factors.
Second model was function points model which was based on the relative functionality of delivered system. For functionality user inputs, outputs, inquiries, logical files, external interfaces were used and given expert weights. Similar to the previous model unadjusted function point count is then adjusted by 14 weighted general characteristics.
A.F. Sheta, D. Rine, and A. Ayesh, "**Development of software effort and schedule estimation models using Soft Computing Techniques"** ; Proceedings of IEEE conference , Congress on Evolutionary Computation, pp.1283-1289, 2008.They have proposed two techniques to assist in solving problem of software effort estimation. First technique used PSO method to tune the parameters of the COCOMO model. Another model based on fuzzy logic was developed to model the relationship between the KLOC and the effort required to develop a software system. It was observed that PSO tuned COCOMO model outperformed the traditional models.
Verma H.K and Sharma V. **"Handling Imprecision's in Inputs using Fuzzy Logic to predict effort in Software Development"** Proceedings of IEEE International Advance Computing Conference, pp: 436-442, 2010. They have proposed and enhanced fuzzy logic based framework to handle the imprecision and uncertainty present in the data at the early stages of the project to predict the effort more accurately. The proposed framework is build upon COCOMO. The intermediate COCOMO has been used because it has estimation accuracy more than the basic version and comparable to the detailed version.
Mohd. Sadiq, Farhana Mariyam, Aleem Ali, Shadab Khan, Pradeep Tripathi**." Prediction of Software Project Effort Using Fuzzy Logic"** Proceedings of IEEE 3rd International Conference on Electronics Computer Technology (ICECT), pp: 353 – 358, 2011
They have developed two different linear regression models using fuzzy function point and non fuzzy function point in order to predict the software project effort for organic software i.e. projects with size between 2 to 50 KLOC. In their paper the authors have predicted the software project using linear regression model. On the basis of the data values they got the FP regression models using function points and fuzzy function points.
J.N.V.R.Swarup Kumar, T.Govinda Rao, Y. Naga Babu S.Chaitanya, K.Subrahmanyam **"A Novel Method for Software Effort Estimation Using Inverse Regression as firing Interval in fuzzy logic"** Proceedings of IEEE 3rd International Conference on Electronics Computer Technology (ICECT), pp: 177 – 182, 2011 They proposed fuzzy software cost estimation model using inverse regression as firing interval and evaluated it using MRE. After empirical evaluation they proved that proposed fuzzy logic model showed better software effort estimate as compared to traditional estimation model.
Iman Attarzadeh, Siew Hock , **"Improving Estimation Accuracy of the COCOMO II Using an Adaptive Fuzzy Logic Model"** Proceedings of IEEE International Conference on Fuzzy Systems, pp: 2458 – 2464, 2011.They propose an adaptive fuzzy logic model to improve the accuracy of software time and cost estimation. According to them use of fuzzy set and fuzzy logic can produce accurate software attributes which may result in precise software estimates. The Two-

Dimension Gaussian Membership Function (2-D GMF) was used in the fuzzy model to make software attributes smoother in terms of the range of values. Soft computing techniques such as fuzzy logic can reduce the vagueness and uncertainty of software attributes. Therefore, it may consider as alternative to decrease the inaccuracy of software estimates. This research aims to utilize an adaptive fuzzy logic model to improve the accuracy of software time and cost estimation.

Chrysostomos D. Stylios and Peter P. Groumpos, **"The challenge of modelling supervisory systems using fuzzy cognitive maps"**, Journal of Intelligent Manufacturing Vol. 9, pp. 339-345, 1998. This paper investigates a new theory, Fuzzy Cognitive Map (FCM) Theory, and its implementation in modeling systems. First the description and the methodology that this theory suggests is examined and then the application of FCM in a process control problem is described. For such systems it is extremely difficult to describe the entire system by a precise mathematical model. Thus, it is more attractive and useful to represent it, in a graphical way showing the causal relationships between states-concepts. Since this symbolic method of modeling and control of a system is easily adaptable and relies on human expert experience and knowledge it can be considered intelligent. The presentation indicates how effective FCMs are and some interesting points for further research are included.

REN Haoli, WANG Mingjun, **"A Study on Self-Organization System Based on Fuzzy Cognitive Map Method"**. ISSN 978-1-4673-0089-6/12/ IEEE 2012. In this paper, the SO systems Characteristics are studied, and some problems are brought forward related with the SO system research. It shows that Fuzzy Cognitive Map (FCM) method provides a powerful framework for developing and evaluating. In FCM, the concepts such as actions, events, and action motivators are represented as nodes connected in a causal web. The Self-Organization comes from the interactions between the concepts and is encoded in the overall architecture. The nodes are influenced between each other by their weight influence. In this paper, a new method is introduced using FCM to evaluate the SO system.

Sameera Al Shayji, Nahla El Zant El Kadhi, Zidong Wang, "**Fuzzy Cognitive Map Theory for the Political Domain".** Proceedings of the Federated Conference on Computer Science and Information Systems pp. 179–186, ISBN 978-83-60810-22-4, IEEE 2011. This paper discuss that an acceleration of regional and international events contributes to the increasing challenges in political decision making, especially the decision to strengthen bilateral economic relationships between friendly nations. Obviously this becomes one of the critical decisions. Typically, such decisions are influenced by certain factors and variables that are based on heterogeneous and vague information. A serious problem that the decision maker faces is the difficulty in building efficient political decision support systems (DSS) with heterogeneous factors. The basic concept is a linguistic variable whose values are words rather than numbers and therefore closer to human intuition. Fuzzy logic is based on natural language and is tolerant of imprecise data. Furthermore, fuzzy cognitive mapping (FCM) is particularly applicable in the soft knowledge domains such as political science. In this paper, a FCM scheme is proposed to demonstrate the causal inter-relationship between certain factors in order to provide insight into better understanding about the interdependencies of these factors. It presents fuzzy causal algebra for governing causal propagation on FCMs.

The literature survey shows that there have many trials in applying soft computing based techniques in software cost estimation. Some of these technical improvements have proved to be very beneficial also. But still there is room for improvement. This is mainly because of continuous change in the development process and tools. Also fuzzy cognitive maps have been used in various domains in which there are multiple factors affecting a single variable. In addition multiple factors affect each other also. Software development is a similar field. In software development process there are many factors affecting the software development effort. In addition these factors affect each other. In this paper we have tried to explore the application of FCM in quantification and comparison of interdependencies of factors affecting software development effort. Various factors affecting the software development effort are adopted from COCOMO developed by Boehm.

VI. IMPLEMENTATION DETAILS

After studying the various available software effort calculation models including COCOMO, the factors which highly affect the total effort are identified. According to COCOMO model there are fifteen cost/effort multipliers. These are Database Size, Process Complexity, Time Constraint for CPU, Main Memory Constraint, Scheduling Constraint, Required Software Reliability, Use Of Software Tools, Application Experience, Analyst Capability, Programmer Capability, Language Experience, Modern Programming Practices, Virtual machine Experience, Turn Around Time and Machine Volatility. Probable dependencies among these effort multipliers are identified. These are taken for further study regarding their effect on each other and hence indirectly on total development effort. A table is made to represent the dependency of Factor2 on Factor1. Based on these, a questionnaire is designed to get opinion of experts in the field of software development. This questionnaire contained 26 questions, each question is about how strongly one factor effects the other factor i.e., very low, low, nominal, high, very high, extra high. Questionnaire was responded by experts dealing with various phases of software development. Five such responses were taken.

Consistency of responses is checked by applying Mann Whitney test.

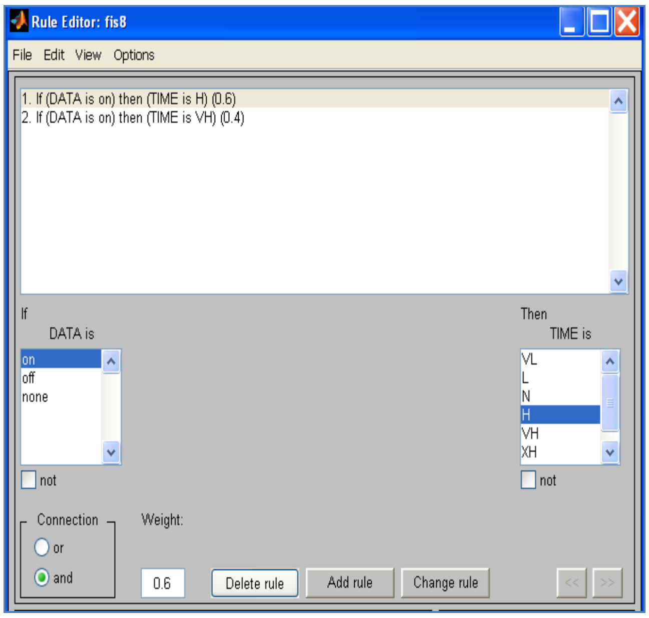
Results of Mann Whitney tests are tabulated in adjoining table. In this table p,h value for each pair responses is calculated. R1,R2 etc. implies the respective reponses.

Above table shows that populations generating two independent

TABLE **I**
RESULTS OF MANN WHITNEY TEST

	R1-R2	R1-R3	R1-R4	R1-R5	R2-R3	**R2-R4**	R2-R5	R3-R4	R3-R5	**R4-R5**
P	0.4315	0.5010	0.4704	0.3831	0.7547	**0.9454**	0.8476	0.9313	0.9845	**0.8013**
H	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

samples, Ri and Rj, are identical. The P & H values are within permissible limit so data may be used for further analysis. Matlab is used to design fuzzy inference system to ascertain the weight of impact of one factor on the other. Above file shows the .fis file used for calculating the weight which represents the strength with which Database Size affects Time Constraints for CPU. In fuzzy rules in this file value 0.6 has been taken because response about effect of Database Size on was "high" according to 3 out of 5 experts. Also value 0.4 represents that according to two out of five experts, effect of Database Size on Time Constraints for CPU is "very high".



1. If (DATA is on) then (TIME is H) (0.6)
2. If (DATA is on) then (TIME is VH) (0.4)

Fig 1: Rule Editor in FIS

VII. Rules Preparation

The relationships among the factors have been defined in the form of if-then rules of fuzzy logic. Rules are formulated after representation of input and outputs of fuzzy sets in membership functions. As in the above case:
If database size to be handled in software systems is high, it will result in higher time constraint for CPU. Rules for this are mentioned below:

1. If (Database Size is on) then (Time Constraint for CPU is High) (0.6)
2. If (Database Size is on) then (Time Constraint for CPU is Very High) (0.4)

Here the first rule is multiplied with 0.6 because this rule reflects the opinion of three out of five experts. The second rule is multiplied with 0.4 because it reflects the opinion of two out of five experts.(Fig1)

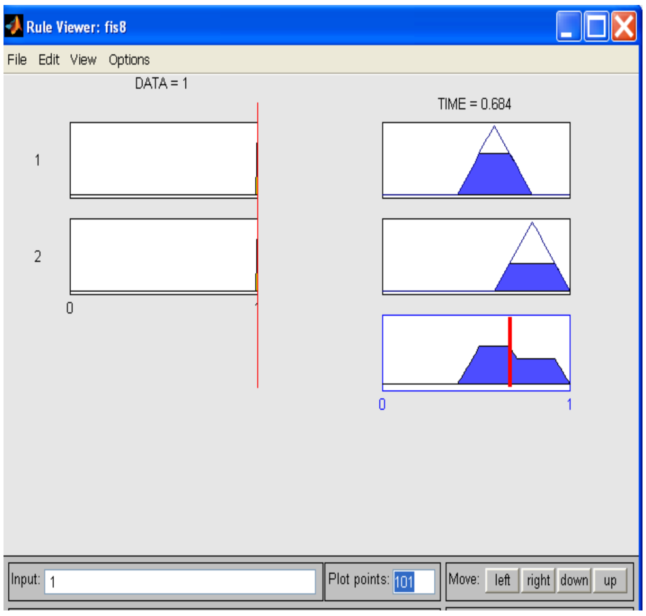


Fig 2: Weight calculation by using input value one

Similarly if complexity of the product increases, it will result in higher turnaround time.
1. If (CPLX is on) then (TURN is L) (0.6)
2. If (CPLX is on) then (TURN is N) (0.4)

Here the first rule is multiplied with 0.6 because it reflects the opinion of three out of five experts. The second rule is multiplied with 0.4 because it reflects the opinion of two out of five experts.

Similarly rules for all factors mentioned in response table as factor1 affecting factor 2 are framed in different FIS files.

The weight is calculated by considering input value as one. Fig. 2 shows the step of calculating weight by making one input as one.This weight shows the dependency of one factor on another. The weights are positive for those pair of factors which are directly proportional and negative for inversely proportional factors.

Twenty six FIS files have been created, one for each dependency factor duo. Each file formulates the corresponding rules between input and output factors.

Weights calculated for all the factors considered are as follows:

Fig.3 and table II shows that absolute weight of dependence of relationship id. 6,7,8,9 is greater than 0.684. All these relationships involve Database size as factor1, which implies that Database size is one of the important factors affecting the software development cost. The weights of relationship of Analyst Capability, Machine Volatility, Programmer Capability with software development cost is also greater than 0.684. This implies that these three factors affect the software development cost the most. As per the finding scheduling constraints affects the software development cost the least. This is an exception to real life observation. So this needs to be further investigated. The effect of other factors on each other may be explained in similar fashion. We have considered twenty six relationships only. Other possible relationships may also be invesitigated.

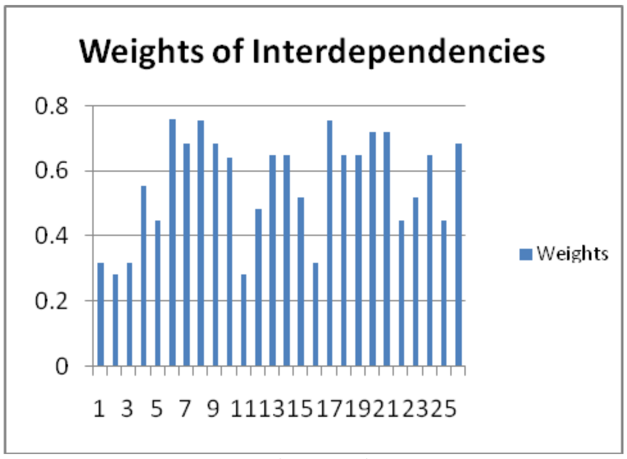


Fig.3 Comparison of weight of interdependencies

VIII. CONCLUSION AND FUTURE WORK

Relative strength of interdependencies of software effort multipliers of COCOMO model has been calculated using fuzzy inference system. These weights, representing the mutual dependence of factors may help the developers in deciding the most sensitive factors in software development. Also it will help the software industry to find, what other factors are to be studied when there is change in one factor. The terms used for depicting effect of one factor on other are low, very low, high, very high, extraordinarily high etc. Same are used in fuzzy logic to depict the effect of one factor on the other. Fuzzy logic is used since it is a tool capable of modeling complex and uncertain data using simple terminology such as if-then-else statements. This logic is perfect to deal with terms used by experts to talk about effect of factors on software development effort. These weights will be used to devise a model to estimate the software effort more accurately in future. Fuzzy cognitive maps explored in this paper may be used in development of formal model for software effort estimate. The development of model will be our direction of future research.

REFERENCES

[1] L. C. Briand, K. E. Emam, and I. Wieczorek, "*Explaining the cost of european space and military projects*," in ICSE '99: Proceedings of the 21st international conference on Software engineering, (Los Alamitos, CA, USA), pp. 303–312, IEEE Computer Society Press, 1999.

[2] R. Agarwal, M. Kumar, M. Yogesh, S. Mallick, R. M. Bharadwaj, and D. Anantwar, "*Estimating software projects*," SIGSOFT Softw. Eng. Notes, vol. 26, no. 4, pp. 60–67, 2001.

[3] C. F. Kemerer, "*An empirical validation of software cost estimation models*," Communications of the ACM, 30(5), pp. 416-429, 1987.

[4] K. Molokken and M. Jorgensen, "*A review of software surveys on software effort estimation*," Proceedings of IEEE International Symposium on Empirical Software Engineering (ISESE 2003). Rome, Italy, September 30 – October 1, 2003, pp: 223 – 230, 2003.

[5] S.G. MacDonell and A.R. Gray, "*A comparison of modeling techniques for software development effort prediction*," Proceedings of the International Conference on Neural Information Processing and Intelligent Information Systems, Dunedin, New Zealand, November 12-14, pp. 869-872, 1997.

[6] R. Agarwal, M. Kumar, M. Yogesh, S. Mallick, R. M. Bharadwaj, and D. Anantwar, "*Estimating software projects*," SIGSOFT Software Engineering Notes, 26(4), 60-67, 2001.

[7] K. Strike, K. El-Emam, and N. Madhavji, "*Software cost estimation with incomplete Data*," IEEE Transactions on Software Engineering, 27(10), pp. 215-223, 2001.

[8] A. C. Hodgkinson and P.W. Garratt, "*A neurofuzzy cost estimator*," Proceedings of the 3rd International Conference on Software Engineering and Applications (SAE-1999), NJ, USA, 999, pp. 401-406, 1999.

[9] L. H. Putnam, "*A general empirical solution to the macro software sizing and estimating problem*," IEEE Transactions on Software Engineering, 4(4), pp. 345 – 361, 1978.

[10] B. W. Boehm, *Software engineering economics*. Englewood Cliffs, NJ, Prentice-Hall, 1981.

[11] B. W. Boehm, C. Abts, and S. Chulani, "*Software development cost estimation approaches – A survey*", University of Southern California Center for Software Engineering, Technical Reports, USC-CSE-2000-505, 2000.

[12] B. W. Boehm, "*Cost models for future software life cycle processes: COCOMO 2.0*," Annals of Software Engineering Special Volume on Software Process and Product Measurement, Science Publisher, Amsterdam, Netherlands, 1(3) , pp. 45 – 60., 1995.

[13] K. Srinivasan, and D. Fisher, "*Machine learning approaches to estimating software development effort*," IEEE Transactions on Software Engineering, 21(2), 1995.

[14] X. Huang, D. Ho, J. Ren, and F. Capretz, "*A soft computing framework for software effort estimation*," Soft Computing, A Fusion of Foundations, Methodologies and Applications Journal, 10(2), pp. 170-177, 2009.

[15] C. E. Walston, C. P. Felix, *"A method of programming measurement and estimation"*, IBM systems Journal, vol. 16, no. 1, pp. 54-73, 1977.

[16] G.N. Parkinson, Parkinson's Law and Other Studies in Administration, Houghton-Miffin, Boston, 1957.

[17] R J. R. Herd, J.N. Postak, W.E. Russell, K.R. Steward, "*Software cost estimation study: Study results*", Final Technical Report, RADC-TR77-220, vol. I, Doty Associates, Inc., Rockville, MD, pp. 1-10, 1977.

[18] Kosko B., *"Fuzzy Cognitive Maps"*, International Journal Man machine Studies Studies, No 24, pp 65-75,1986.

TABLE II.
WEIGHTS CALCULATED WITH FIS

Relationship No.	**Factor 1**	**Factor 2**	**Weights**
1.	Analyst Capability	Scheduling Constraint	0.316
2.	Programmer Capability	Scheduling Constraint	0.284
3.	Modern Programming Practices	Process Complexity	0.316
4.	Use of Software Tools	Turn Around Time	0.552
5.	Use of Software Tools	Process Complexity	0.448
6.	Database Size	Main Memory Constraint	0.757
7.	Database size	Time Constraint for CPU	0.684
8.	Database size	Turn Around Time	0.752
9.	Database size	Process Complexity	0.684
10.	Process Complexity	Turn Around Time	0.284
11.	Database size	Software Development Cost	0.484
12.	Process complexity	Software Development Cost	0.648
13.	Time constraint for CPU	Software Development Cost	0.648
14.	Main memory constraint	Software Development Cost	0.516
15.	Scheduling Constraint	Software Development Cost	0.316
16	Required software reliability	Software Development Cost	0.752
17	Use of software tools	Software Development Cost	0.648
18	Analyst Experience	Software Development Cost	0.648
19	Analyst capability	Software Development Cost	0.716
20	Programmer capability	Software Development Cost	0.716
21	Language experience	Software Development Cost	0.448
22	Modern Programming Practices	Software Development Cost	0.516
23	Virtual Machine Experience	Software Development Cost	0.648
24	Turnaround Time	Software Development Cost	0.448
25	Machine Volatility	Software Development Cost	0.684