Genetic Fuzzy System for Enhancing Software Estimation Models

Abeer Hamdy

***Abstract*—Imprecise estimation of software development cost is one of the major factors that contributes in the failure of software projects. Several algorithmic models have been devised for cost estimation; but they lack the *ability* to handle imprecision and uncertainties associated with the software project attributes. Embedding a fuzzy component in the algorithmic model enables it to deal with the imprecision and uncertainty problem; consequently improves its accuracy. However, the performance of any fuzzy system depends on the settings of its parameters. This paper proposes a genetic fuzzy model for effort estimation. Genetic algorithm is used in tuning the fuzzy sets of the model to optimize the estimation accuracy. MATLAB 2012 was used in implementing the proposed model. The model was evaluated using artificial datasets derived from COCOMONASA2 dataset. The experimental results showed that the accuracy and sensitivity of the proposed model is superior to COCOMO. It's note worthy to mention that the idea of the paper is not restricted to COCOMO; it could be applied to other algorithmic models.**

***Index Terms*—COCOMO81, genetic algorithms, fuzzy systems, genetic fuzzy, effort estimation.**

I. Introduction

Software cost estimation refers to the prediction of the human effort (typically measured in man-months) and time needed to develop a software artifact. The accurate estimation of the development effort and cost of a software system is one of the important and challenging tasks for software project management. It helps in contract negotiations, project scheduling and efficient allocation of resources. However, estimates at the preliminary stages of the project are the most difficult to obtain because the primary source to estimate the cost comes from the requirement specification documents [1]. Considerable research has been carried out in the past, to come up with a variety of effort prediction models. Putnam developed an early model known as SLIM in 1978 [2]. Boehm proposed cost estimation model, COCOMO 81 (COnstructive Cost MOdel) in 1981 [3], [4]. Several other algorithmic models have been proposed in the literature like function point analysis [5] and Usecase point [6]. All these models are derived by applying regression techniques to data from past projects. They lack the ability to handle the vagueness and inaccuracy associated with the different projects attributes. Fuzzy logic, introduced by Lotfi Zadeh [7], provides the concept of fuzzy sets to handle vague and inaccurate data. However, optimizing the parameters of the fuzzy sets is one of the challenging problems in the fuzzy experts systems. Genetic algorithms (GAs) are general purpose search algorithms which have proved a great success in search and optimization problems. GAs is inspired by natural genetics to evolve solutions to problems. Their basic idea is to maintain a population of chromosomes that represent candidate solutions to the problem being solved. The chromosomes evolve over time, towards better chromosomes, through the mechanisms of natural evolution such as selection, mutation and reproduction. A fitness function is associated with each chromosome in the population to rate the chromosomes and determine (in the selection process) which chromosomes are used to form the new generation. Genetic operators such as crossover and mutation are applied to the new generation to explore the solution space. Over the last years enormous publications have integrated fuzzy logic and genetic algorithms in different fashions to optimize the performance of the fuzzy expert systems. GAs can be used in generating the rule base of the fuzzy system or tuning the parameters of the membership functions.

This paper proposes a fuzzy model to enhance the accuracy and sensitivity of COCOM81 intermediate model. GA is used in tuning its parameters to optimize its performance. It's worth mentioning that the work presented in this paper isn't restricted to COCOMO81 intermediate; it could be applied to other algorithmic models. Intermediate COCOMO81 model is selected for two reasons: 1) It's a widely used model and 2) To use the publicly available COCOMO81 datasets (like COCMONNASA2 [8]) in the experiments.

The paper is organized as follows: Section II introduces COCOMO models. Section III discusses the imprecision problem associated with COCOMO and the proposed genetic fuzzy system. Section IV discusses the experiments and results. Related work is introduced in Section V. While, Section VI concludes the paper and introduces the future research.

II. COCOMO81 Cost Model

COCOMO81 was published by Barry Boehm in 1981[3]. It was developed from the analysis of sixty three software projects. COCOMO81 has three versions called Basic COCOMO81, Intermediate COCOMO81 and Detailed COCOMO81 [3], [4]. The used version depends on the available information. Basic COCOMO81 is the simplest and least accurate one. It is used for quick and rough estimate of effort. The Basic COCOMO81 model is based on the following formula:

$$MM_{est} = A\, Size^{B} \tag{1}$$

where,

Manuscript received October 9, 2013; revised December 12, 2013.

Abeer Hamdy is with the Electronics Research Institute and British University, Egypt (e-mail: Abeer.hamdy@bue.edu.eg)

DOI: 10.7763/IJMO.2014.V4.378

MM_{est}: is the nominal estimated effort in terms of person per month;

Size: is the software size measured in KLOC.

The constants A, B are dependent upon the 'mode' of the development project. Boehm proposed 3 modes of projects (shown in Table I):

1) *Organic mode* – simple projects that engage small teams working in known and stable environments.
2) *Semi-detached mode* – projects that engage teams with a *mixture* of experience. It is in between organic and embedded modes.
3) *Embedded mode* – complex projects that are developed under *tight* constraints with changing requirements.

TABLE I: COCOMOI PROJECT MODES, A & B VALUES.

Development mode	A	B
Organic	2.4	1.05
Semi-detached	3.0	1.12
Embedded	3.6	1.2

The accuracy of Basic COCOMO is limited because it does not consider factors like hardware, personnel, use of modern tools and other attributes that affect the project cost. Boehm proposed the Intermediate COCOMO that adds accuracy to the Basic COCOMO by multiplying the nominal estimated effort ,derived from equation 1, by the product of 15 'Cost Drivers'. The 15 cost drivers can be classified into four categories:

1) *Product*: RELY – Required software reliability, DATA – Data base size, CPLX – Product complexity.
2) *Platform*: TIME – Execution time, STOR- Main storage constraint, VIRT – Virtual machine volatility, TURN – Computer turnaround time
3) *Personnel*: ACAP – Analyst capability, AEXP – Applications experience, PCAP – Programmer capability, VEXP – Virtual machine experience, LEXP – Language experience.
4) *Project*: MODP – Modern programming, TOOL – Use of software tools, SCED – Required development schedule.

Each cost driver in the intermediate COCOMO81 has a *definition*, and is measured using a certain scale of six linguistic values: "very low", "low", "nominal", "high", "very high", "extra high" (some cost drivers don't cover the whole scale). The assignment of a linguistic value (rating) to a cost driver depends on its definition as given by Table II. For each rating there is a corresponding real number (multiplier factor) that affects the value of the nominal estimated effort as given by Table III. Depending on the software project attributes, effort multipliers of the cost drivers will vary. The product of all effort multipliers results in an *effort adjustment factor (EAF)* that increases or decreases the value of the nominal estimated effort. Typical values for EAF range from 0.9 to 1.4. The predicted effort using intermediate COCOMO81 is given by the following formulas:

$$MM_{est} = A\ Size^{B} * EAF \tag{2}$$

$$MM_{est} = A\ Size^{B} \prod_{i=1}^{15} C_i \tag{3}$$

where, C_i is the effort multiplier associated with the i[th] cost driver.

TABLE II: DEFINITIONS OF COCOMO 81 COST DRIVERS [3]

	Project Attributes Ranges					
	Very Low	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
RELY	50%	60%	70%	80%	90%	
DATA	-	10	100	1000	>1000	
CPLX	15%	30%	45%	60%	70%	90%
TIME			50%	70%	85%	95%
STOR			50%	70%	85%	95%
VIRT		12 Months	6 Months	2 Months	2 Weeks	
TURN		0	4 Hours	12 Hours	12 Hours	
ACAP	15%	35%	55%	75%	90%	
PCAP	15%	35%	55%	75%	90%	
AEXP	4 Months	1 Year	3 Years	6 Years	12 Years	
TOOL	15%	35%	55%	75%	90%	
MODP	15%	35%	55%	75%	85%	
LEXP	1 Month	4 Month	1 Year	3 Years		
VEXP	1 Month	4 Month	1 Year	3 Years		
SCED	75%	85%	100%	130%	160%	

TABLE III: COCOMO81 INTERMEDIATE EFFORT MULTIPLIERS [3]

	Very Low	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
RELY	.75	.85	1.00	1.15	1.4	
DATA		.94	1.00	1.08	1.16	
CPLX	.7	.85	1.00	1.15	1.3	1.65
TIME			1.00	1.11	1.30	1.66
STOR			1.00	1.06	1.21	1.56
VIRT		0.87	1.00	1.15	1.30	
TURN		0.87	1.00	1.07	1.15	
ACAP	1.46	1.19	1.00	.86	.71	
PCAP	1.42	1.17	1.00	0.86	0.70	
AEXP	1.29	1.13	1.00	0.91	0.82	
TOOL	1.24	1.10	1.00	0.91	0.83	
MODP	1.24	1.10	1.00	0.91	0.82	
LEXP	1.14	1.07	1.00	0.95		
VEXP	1.21	1.10	1.00	0.90		
SCED	1.23	1.08	1.00	1.04	1.10	

III. RESEARCH METHODOLOGY

This section discusses both the problem of imprecision and vagueness that exists with the COCOMO81 cost drivers; and the proposed genetic fuzzy model to handle this problem.

A. COCOMO Imprecision Problem

Consider the ACAP cost driver as an example to explain the imprecision problem exists with cost drivers. ACAP linguistic values are defined according to Table II. So, if the ACAP attribute of a project is in the range 15 to 35 percentile; the rating "low" is assigned to this cost driver for this project and consequently an effort multiplier factor equals to 1.19 (according to Table III) is used in equation3 to compute EAF. While, if the ACAP attribute of a project equals to 36; the rating nominal is assigned to the ACAP cost driver for this project and an effort multiplier equals to 1 is used which leads to a different value for EAF. From this example we can come up with two problems:

1) COCOMO applies the traditional quantization

method to the intervals. i.e. a range of values is dealt as a singleton.

2) The transition from an interval (linguistic value) to the contiguous one is sudden.

Fuzzy modeling is a good candidate to handle these problems by using fuzzy sets to represent the linguistic values of each cost driver as discussed in the following subsection.

B. *Proposed Fuzzy Model*

The definitions of the cost drivers (listed in Table II) have been studied and a fuzzy inference system (FIS) for each cost driver is developed. Trapezoidal and triangular fuzzy sets are defined for the linguistic values of each cost driver, based on its definition. The defuzzified values result from the FISs are multiplied to form the EAF that is used in equation 2 to adjust the nominal predicted effort instead of using the effort multipliers given by Table III.

Consider the ACAP cost driver. Fig. 1 shows the rule base of the ACAP. Fig. 2 shows the fuzzy sets of the antecedent part which are derived using the definition of the ACAP given by Table II. Fig. 3 shows the fuzzy sets of the consequent part that are derived using the effort multipliers values given by Table III. Fig. 4 shows the overall architecture of the proposed fuzzy model. The following subsection discusses how GA is integrated with the proposed fuzzy model to tune the parameters of the fuzzy sets.

If ACAP-Attribute is Very-law Then ACAP-Multiplier is Very-low.
If ACAP-Attribute is low Then ACAP-Multiplier is low.
If ACAP-Attribute is Nominal Then ACAP-Multiplier is Nominal.
If ACAP-Attribute is High Then ACAP-Multiplier is High.
If ACAP-Attribute is Very-High Then ACAP-Multiplier is Very-High.

Fig. 1. ACAP rulebase.

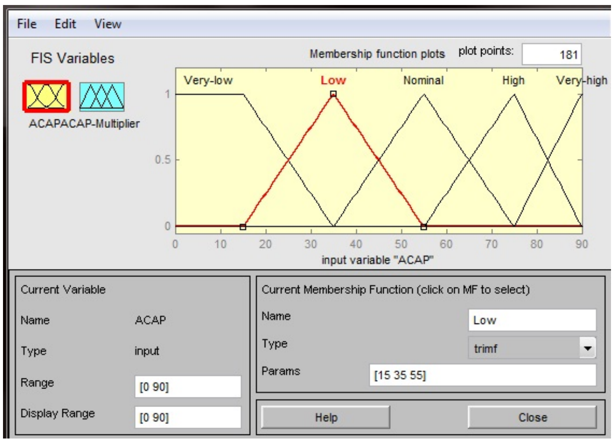


Fig. 2. Antecedents fuzzy sets of ACAP cost driver (ACAP project attribute).

C. *Proposed Genetic Fuzzy Model*

In order to use GA in tuning the fuzzy sets parameters, an initial population of potential solutions is generated, a set of evolution operators, that search for new and better solutions, should be defined. A fitness function that is used as a performance index for population individuals should be set. It drives the evolution of the population towards better solutions.

The evolution process proceeds as follows:

1) Generates an initial population of solutions p(0).
2) Evaluate each solution using the proposed fitness function.
3) While (not termination condition) do
   a) Select p(t) from p(t-1)
   b) Recombine p(t)
   c) Evaluate p(t)

The population of potential solutions: All chromosomes are real coded and have fixed length, each chromosome contains a coding of the whole set of membership functions, i.e. each chromosome represents different parameters definitions for the fuzzy model. Population size is set to 40.

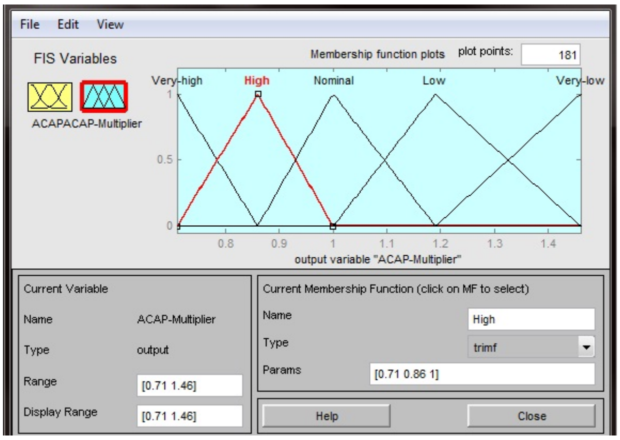


Fig. 3. Consequent fuzzy sets of ACAP cost driver (ACAP effort multiplier).

Selection mechanism: Several selection mechanisms are proposed in the literature, *Roulette-wheel selection* mechanism is used in this paper. It's a form fitness-proportionate selection in which the chance of an individual's being selected is proportional to the amount by which its fitness is greater or less than its competitors' fitness.

Evolution operators: New potential solutions (generation) are obtained by a applying the evolution operators to the chromosomes of the intermediate population that is produced from the selection mechanism. This process is called recombination process. The operators should make balance between exploiting the best solutions and exploring the search space aiming at new solutions. The basic operators are mutation and crossover.

Crossover operator: Crossover produces artificial "offsprings" by selecting two chromosomes and swap some of their gene segments. *Single-point crossover* is the crossover form that is used in this paper. In this crossover a point of exchange is set at a random location in the two chromosomes, and one individual contributes all its genes from before that point and the other contributes all its genes from after that point to produce offsprings. The crossover rate in this paper is set to be 70-80%.

Mutation operator: Chromosomes with worst fitness function are selected for mutation. Mutation is achieved by adding random numbers within some ranges, based on the definition of the gene, to all genes of the selected chromosomes. The mutation rate in this paper is set to be 10%.

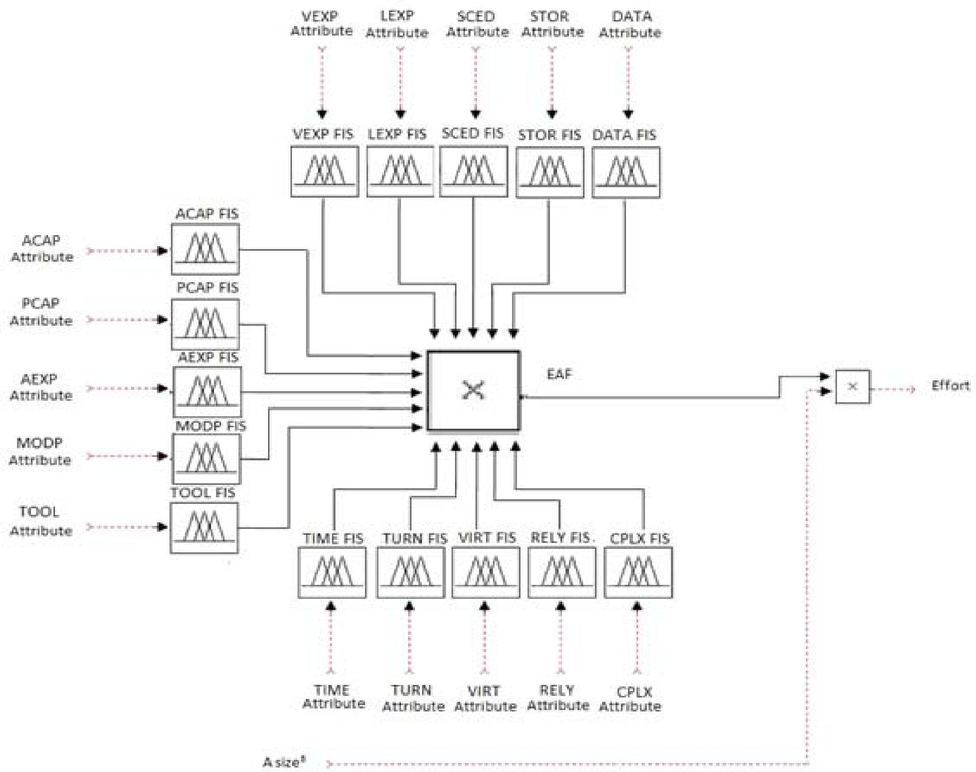


Fig. 4. The proposed fuzzy model.

The fitness function: Fitness function is a key factor in the evolution process. Fitness function decided in this paper is based on error measurements that characterize the difference between the actual effort and the estimated one by the proposed model, as given by the following equation:

$$Fitness\ function = \frac{1}{N}\sum \frac{|Actual\ Effort_i - Predicted\ Effort_i|}{Actual\ Effort_i}$$

where, N : number of the software projects used in the optimization process;

$Actual\ Effort_i, Predicted\ Effort_i$: Actual and estimated efforts of a project i.

IV. Experiments and Results

MATLAB2012 toolboxes were used in the experiments. The following subsections introduce the datasets used in both of tuning the fuzzy model using genetic algorithms and evaluating the performance of the genetic fuzzy model (Gfuzzy) against both of the untuned fuzzy model (Fuzzy) and the COCOMO81 intermediate. The criteria that are used in performance evaluation are introduced in subsection B. Subsection C summarizes and discusses the results of the testing experiments.

A. Data Sets

COCOMONASA2 dataset [8] is one of the publicly available COCOMO81 datasets. It was collected from six NASA centers and covers a wide range of software domains, development process, languages and complexity, as well as fundamental differences in culture and business practices between each center. All of these factors contribute to the large variances observed in this data set. The problem with this dataset and all the other publicly available COCOMO81 datasets is that, they only record the effort multipliers of the projects (they don't include real project attributes (values of the project cost drivers)). An example of a project in the COCOMONASA2 dataset is:

"High,Low,High,Nominal,Nominal,Low,Low,Nominal,Nominal,Nominal,Nominal,High,High,Nominal,Low,25.9,117.6,semidetached"[8]

In order to overcome this problem, the NASA data was passed by a preprocessing stage. Through this stage each linguistic effort multiplier was replaced by a value represents the corresponding project attribute. These values are generated randomly based on the definition of the cost drivers listed by Table II. For example preprocessing the project sample above may result in the following:

"78,7,50,48,45,8,0,51,15,39,11,29,56,46,80,25.9,117.6,semidetached"

Eight artificial datasets were generated by preprocessing COCOMONASA2. The generated datasets contain project attributes instead of effort multipliers. Four data sets are used in the evolution process, while the other four data sets are used in performance evaluation.

B. Performance Assessment Criteria

Several criteria to assess and compare effort estimation models are proposed in the literature [9]. One of these criteria is the magnitude of relative error (MRE) which is defined for a project i as follows:

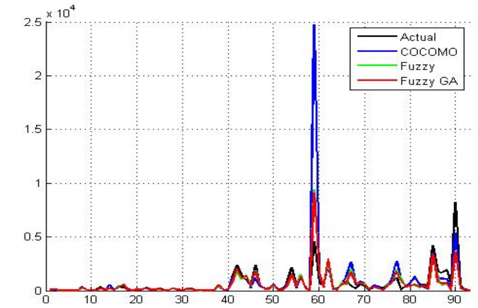
$$MRE_i = \frac{|Actual\ Effort_i - Predicted\ Effort_i|}{Actual\ Effort_i} * 100\%$$

A value of 25% for MRE is acceptable.

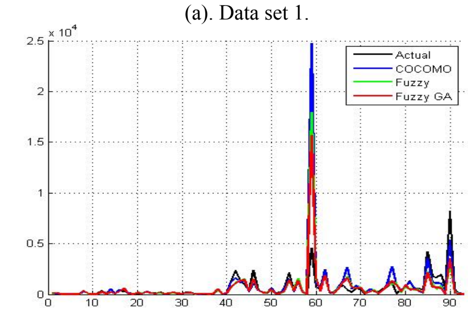
Another widely used measure is the pred(l) which is defined as follows:

$$pred(l) = \frac{k}{N} \times 100$$

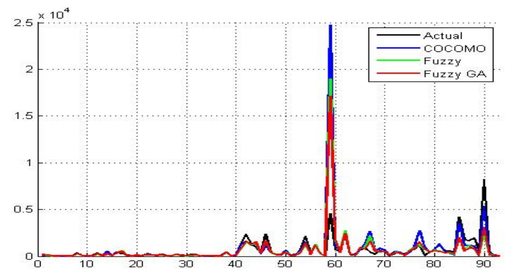
where, *N:* is the total number of projects, and k is the number of projects whose MRE is less than or equal to l. A common value for l is 0.25. The Pred (0.25) represents the percentage of projects whose MRE is less than or equal to 25%. The accuracy of any estimation technique is proportional to the **pred**. This metric is used in this paper.



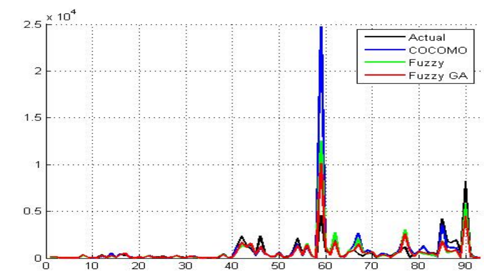
(a). Data set 1.



(b). Data set 2.



(c). Data set 3.



(d). Data set 4.

Fig. 5. Each of the four figures show, for each artificial dataset, a comparison between the actual effort of each project and estimated efforts using COCOMO81, fuzzy model and genetic fuzzy model.

C. Results

Fig. 5 shows a comparison (for each project in each of the four artificial datasets) between the actual effort and the estimated effort using each of the three models COCOMO81, fuzzy and GFuzzy.

Table IV and V summarizes the values of the MMRE and the pred(25%) when using each of COCOMO81, the proposed fuzzy model and the GFuzzy model in estimating the effort required for each project in the four artificial datasets. It is noteworthy that each of the fuzzy model and the GFuzzy model result in different values for the pred (25) and MMRE across the four artificial datasets, while applying COCOMO81 produces the same pred (25%) and MMRE values over the four datasets. These results show that both fuzzy models are more sensitive for the values of the project attributes than COCOMO81 even though the COCOMO81 estimations outperform the fuzzy model estimations. COCOMO81 results in higher values for pred (25) and lower values for MMRE than the unturned fuzzy model , over the four data sets. It should also be noted that the G Fuzzy model outperforms both of the COCOMO81 and the fuzzy model. It results in the lowest MMRE across the four data sets. The values of the pred (25) over the first two data sets are lower in case of using the G Fuzzy model than in case of using COCMO81, this is due to the selection of the fitness function. It is based on the MMRE and doesn't include any component that represents the Pred (25).

TABLE IV: MMRE DUE TO EFFORT ESTIMATION USING COCOMO81, FUZZY MODEL AND GENETIC FUZZY MODEL

MMRE	DATASET 1	DATASET 2	DATASET 3	DATASET 4
COCOMO	59.497	59.497	59.497	59.497
FUZZY	52.908	56.168	50.519	52.999
GFUZZY	47.901	51.087	48.449	47.644

TABLE V: PRED (25) DUE TO EFFORT ESTIMATION USING COCOMO81 , FUZZY MODEL AND GENETIC FUZZY MODEL

Pred(25)	DATASET 1	DATASET 2	DATASET 3	DATASET 4
COCOMO	48.387	48.387	48.387	48.387
FUZZY	45.161	41.935	41.935	39.784
GFUZZY	46.236	46.237	43.011	45.161

V. RELATED WORK

Artificial intelligence techniques have attracted the attention of software engineers to tackle the problem of software effort estimation. Fuzzy modeling is one of the techniques that are widely applied in this area. Mittal [10] and Reddy [11] enhanced COCOMO by presenting the size attribute as a fuzzy number. Attarzadeh *et al.*, [12] proposed a fuzzy model for cost estimation. Their model takes only two software attributes: complexity and size as inputs. They didn't compare their results with any other models. Parasad *et al.*, [13] proposed another fuzzy model for effort prediction. In this model fuzzy was applied on the effort and two software attributes which are size and mode. Vishal *et al.*, [14] went further; they proposed a fuzzy model that fuzzifies the size, mode and cost drivers. Azzah [15] and Malathi [16] have used fuzzy analogy for effort estimation and they found that it outperforms COCOMO. Noel *et al.*, [17] conducted a study to compare the estimation accuracy of the Mamdani and Takagi-Sugeno fuzzy systems with that of a linear regression model, they used 125 small projects from 37 developers for evaluation. The input to each of their models is one variable, new and changed source line of code (N&C SLOC). They found that Takagi-Sugeno outperforms both of Mamdani and linear regression.

Neural Network (NNet) is another AI technique that has proved its effectiveness in solving effort estimation problem. Dave [18] showed that NNET in general is better than regression analysis and Radial Bases NNET (RBNN) is better than Feed Forward NNET (FFNN). Du [19] and Huang [20] Used Neuro-fuzzy techniques for improving COCOMO model. Support vector machines (SVR) [21] and data mining techniques [22]-[25] are also candidate techniques to tackle this problem. Recently, Kazemifard [26] suggested new project attributes which are the emotions of the team and used multi-agents to model the team emotions.

VI. Conclusion and Future Research

The objective of this research is developing an adaptive fuzzy model for software effort estimation. In [27] a fuzzy logic component was embedded into COCOMO81 intermediate model to improve its sensitivity. But, the parameters of these fuzzy components were tuned manually based on observation. This paper proposed A genetic fuzzy model to be embedded into the COCOMO81 intermediate. GA was used in tuning the parameters of the fuzzy model. The experimental results were promising; they showed the superiority of the genetic fuzzy model over the COCOMO81 and the fuzzy model.

Currently, different forms of fitness function are considered for the tuning process. Also, a complete adaptive fuzzy model will be developed, where COCOMO formula will be replaced by a fuzzy expert system.

References

[1] A. Idri, T. M. Khoshgoftaar, and A. Abran, "Investigating soft computing in case-based reasoning for software cost estimation," *Engineering Intelligent Systems for Electrical Engineering and Communications*, vol. 10, no. 3, pp. 147-157, 2002.

[2] L. H. Putnam, "A general empirical solution to the macro software sizing and estimation problem," *IEEE Transactions on Software Engineering*, vol. SE-4, no. 4, July, 1978.

[3] B. W. Boehm, *Software Engineering Economics*, Prentice-Hall, New York, 1981.

[4] B. W. Bohem, C. Abts, A. W. Brown, *Software Cost Estimation with COCOMOII*, Englewood Cliffs, NJ, USA: Prentice Hall, 2000.

[5] Y. H. Zheng, B. Z. Wang, Y. L. Zheng, and L. Shi, "Estimation of software projects effort based on function point," in *Proc. 4th International Conference on Computer Science and Education*, pp. 941-943, 2009.

[6] A. B. Nassif, D. Ho, and L. F. Capretz, "Regression model for software effort estimation based on the use case point method," in *Proc. International Conference on Computer and Software Modeling*, Singapore, pp. 117-121, 2011.

[7] Z. Lotfi, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, 1965.

[8] COCOMONASA2 DataSet: [Online]. Available: http://promise.site.uottawa.ca/SERepository/datasets/cocomonasa_2.arff

[9] D. Port and M. Korte, "Comparative studies of the model evaluation criterions mmre and pred in software cost estimation research," in *Proc. the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, pp. 51-60, 2008.

[10] A. Mittal, K. Parkash, and H. Mittal, Software cost estimation using fuzzy logic," in *Proc. ACM SIGSOFT Software Engineering Notes*, vol. 35, no. 1, 2010.

[11] C. S.Reddy and K. Raju, "Improving the accuracy of effort estimation through fuzzy set representation of size," *Journal of Computer Science*, pp. 451-455, 2009.

[12] I. Attarzadeh and S. H. Ow, "Software development effort estimation based on a new fuzzy logic model," *International Journal of Computer Theory and Engineering*, vol. 1, no. 4, pp. 1793-8201, October, 2009.

[13] P.V. G. D. P. Reddy, K. R, and R. S. P. Sudha. "Application of fuzzy logic approach to software effort estimation," *International Journal of Advanced Computer Science and Applications*, vol. 2, no. 5, 2011.

[14] V. Sharma and H. K. Verma, "Optimized fuzzy logic based framework for effort estimation in software development," *Computer Science Issues*, vol. 7, no. 2, pp. 30-38, 2010.

[15] M. Azzeh, D. Neagu, and P. I. Cowling, "Analogy based software effort estimation using fuzzy numbers," *The Journal of Systems and Software*, vol. 84, pp. 270-284, 2011.

[16] S. Malathi and S. Sridhar, "Performance evaluation of software effort estimation using fuzzy analogy based on Complexity," *International Journal of Computer Applications*, vol. 40, no. 3, February, 2012.

[17] G. D. Noel, L. Martin, and C. C. Arturo, "A comparative study of two fuzzy logic models for software development effort estimation," in *Proc. 3rd Iberoamerican Conference on Electronics Engineering and Computer Science*, CIIECC 2013, Procedia Technology, Elsevier, vol. 7, pp. 305-314.

[18] D. S. Vanchik and D. Kamlesh, "Comparison of regression model, feed-forward neural network and radial basis neural network for software development effort estimation," in *Proc. ACM Sigsoft Software Engineering Notes*, vol. 36, no. 5, September, 2011.

[19] W. L. Du, D. Ho, and L. F. Capretz, "Improving software effort estimation using neuro-fuzzy model with SEER-SEM," *Global Journal of Computer Science and Technology*, vol. 10, no. 12, pp. 52-64, Oct. 2010.

[20] X. S. Huang, D. Ho, J. Ren, and L. Capretz, "Improving the COCOMO model with a neuro fuzzy approach," *Computer Journal of Applied Soft Computing Journal*, vol. 7, no. 3, pp. 29-40, 2007.

[21] J. k. Lee and K. T. Kwon, "Software cost estimation using SVR based on immune algorithm," in *Proc. 10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing*, pp. 462-466, 2009.

[22] A. S. Andreou and E. Papatheocharous, "Software cost estimation using fuzzy decision trees," in *Proc. 23rd IEEE/ACM International Conference on Automated Software Engineering*, pp. 371-374, 2008.

[23] M. Azzeh, "Software effort estimation based on model tree," in *Proc. the 7th International Conference on Predictive Models in Software Engineering*, New York, NY, USA , ACM, pp. 328-335, 2011.

[24] M. Azzeh, P. I. Cowling, and D. Neagu, "Software stage-effort estimation based on association rule mining and fuzzy SET theory," in *Proc. IEEE 10th International Conference on Computer and Information Technology* (CIT), pp. 249 – 256, 2010.

[25] H. S. Hota and R. P. Singh, "A min-max Approach for Improving the accuracy of effort estimation of COCOMO," *International Journal of Soft Computing and Engineering* (IJSCE), vol. 1, no. 3, 2011.

[26] M. Kazemifard, A. Zaeri, N. G. Aghaee, M. A. Nematbakhsh, and F. Mardukhi, "Fuzzy emotional COCOMO II software cost estimation (FECSCE) using multi-agent systems," *Journal of Applied Soft Computing*, Elsevier Science Publishers ,vol. 11, no. 2, 2011.

[27] A. Hamdy, "Fuzzy Logic for enhancing the sensitivity of COCOMO cost model," *Journal of Emerging Trends in Computing and Information Sciences (CIS)*, vol. 3, no. 9, 2012.

Abeer Hamdy is an associate professor in the Faculty of Informatics and Computer Science at the British University in Egypt. She earned her B.Sc. degree with honors, M.Sc. and Ph.D. degrees in electronics and electrical communications from the Faculty of Engineering, Cairo University in 1992, 1998, and 2003 respectively. Her research focuses on real time systems and software engineering.