Software Effort Estimation Inspired by COCOMO and FP Models: A Fuzzy Logic Approach

Alaa F. Sheta and Sultan Aljahdali
Computer Science Department
College of Computers and Information Technology
Taif University
Taif, Saudi Arabia
asheta66@gmail.com, aljahdali@tu.edu.sa

***Abstract*—Budgeting, bidding and planning of software project effort, time and cost are essential elements of any software development process. Massive size and complexity of now a day produced software systems cause a substantial risk for the development process. Inadequate and inefficient information about the size and complexity results in an ambiguous estimates that cause many losses. Project managers cannot adequately provide good estimate for both the effort and time needed. Thus, no clear release day to the market can be defined. This paper presents two new models for software effort estimation using fuzzy logic. One model is developed based on the famous COnstructive COst Model (COCOMO) and utilizes the Source Line Of Code (SLOC) as input variable to estimate the Effort (E); while the second model utilize the Inputs, Outputs, Files, and User Inquiries to estimate the Function Point (FP). The proposed fuzzy models show better estimation capabilities compared to other reported models in the literature and better assist the project manager in computing the software required development effort. The validation results are carried out using Albrecht data set.**

I. Introduction

According to Dr. Patricia Sanders, Director of Test Systems Engineering and Evaluation at OUSD, in her 1998 Software Technology Conference keynote address, 40% of the DoD's software development costs are spent on reworking the software, which on the year 2000 equal to an annual loss of $18 billion. Furthermore, Sanders stated that only 16% of software development would finish on time and on budget.

The dimension and complication of computer based-systems grown noticeably during the past few decades [1]–[4] and the tendency will certainly continue in the future specially in Military Application, NASA Space Shuttle systems, Air Force and business for huge Enterprises. Some NASA and Air Force projects have estimated that the cost of software development could be up to 50% of their development cost. It was stated in [5]:

> Given that software-intensive projects are among the most expensive and risky undertakings of the 21st century, the investment in weapons from fiscal years 2003 through 2009 will exceed $1 trillion. Furthermore, many of the DoD's most important technology projects will continue to deliver less than promised unless changes are made. Improving how we acquire software-intensive systems is both long overdue and an imperative.

Although many research papers appears since 1960 providing numerous models to help in computing the effort/cost for software projects, being able to provide accurate effort/cost estimation is still a challenge for many reasons. They include: 1) the uncertainty in collected measurement, 2) the estimation methods used which might have many drawbacks and 3) the cost drivers which comes with various characteristics based on the methodology of development.

In this paper, we provide a detailed study on the algorithmic software effort estimation models. We provide our initial idea on using fuzzy models to build a Takagi Sugeno fuzzy model for the software effort. We developed two fuzzy models. They utilize both the SLOC and FP parameters. Our experimental results cover 24 software projects based Albrecht data set.

II. Literature Review

In 1994, Zadeh presented the definition of Soft Computing techniques [6]. He mentioned that soft computing is not a homogeneous body of concepts and techniques. At that time, the principal techniques which compose the domain of soft computing were fuzzy logic, neurocomputing, and probabilistic reasoning. Later on, the domain was expanded to cover techniques such as Genetic Algorithms (GAs), Swarm Intelligence (SI), Differential Evolution (DE) and many others. In the past, soft computing techniques were explored to build efficient effort estimation models structures [7], [8]. In [9], author explored the use of Neural Networks (NNs), GAs and Genetic Programming (GP) to provide a methodology for software cost estimation. Later authors in [10], provided a detailed study on using Genetic Programming (GP), Neural Network (NN) and Linear Regression (LR) in solving the software project estimation. Many data sets provided in [11], [12] were explored with promising results. In [13], authors provided a survey on the cost estimation models using artificial neural networks. Fuzzy logic and neural networks were used for software engineering project management in [14]. A fuzzy COCOMO model was developed in [7].

Recently, Soft Computing and Machine Learning Techniques were explored to handle many software engineering problems. They include the effort and cost estimation problems. In [15], author provided an innovative set of models modified from the famous COCOMO model with interesting results. Later on, many authors explored the same idea with some modification [16]–[18] and provided a comparison to the work presented in [15]. The idea of using Takagi Sugeno Fuzzy Logic was primary presented in [19] to see how a rule based system can solve the software effort estimation problem. Authors in [20] presented an extended work on the use of Soft Computing Techniques to build a suitable model structure to utilize improved estimations of software effort for NASA software projects. On doing this, Particle Swarm Optimization (PSO) was used to tune the parameters of the COCOMO model. A data set for NASA software projects [21] were used to test the developed models. Author provided a comparison between various software cost estimation models. They include COCOMO-PSO, Fuzzy Logic (FL), Halstead, Walston-Felix, Bailey-Basili and Doty models with excellent performance results.

III. COnstructive COst Model

Many software cost estimation models where proposed to help in providing a high quality estimate to assist project manager in taking best decisions for a project [22], [23]. COCOMO is one of a very famous software effort estimation models. COCOMO was introduced by Boehm in 1981 [22], [23]. This model consists of mathematical equations that identify the developed time, the effort and the maintenance effort. The model was developed based on 63 software projects. The estimation accuracy is suggestively improved when adopting models such as the Intermediate and Complex COCOMO models [23]. Equation 1 shows the basic COCOMO model:

$$E = \alpha(SLOC)^{\beta} \tag{1}$$

E presents the software effort computed in man-months. $SLOC$ stands for Source Line Of Code computed in Kilo. The values of the parameters α and β depend mainly on the class of software project. Software projects were classified based on the complexity of the project into three categories. They are: Organic, Semidetached and Embedded models [24]. Extensions of COCOMO, such as COMCOMO II, can be found [25], however, for the purpose of research reported, in this paper, the basic COMCOMO model is used. The three models are given in Table I. These models are expected to give different results according to the type of software projects.

TABLE I. Basic COCOMO Models

Model Name	Effort (E)	Time (T)
Organic Model	$E = 2.4(KLOC)^{1.05}$	$T = 2.5(E)^{0.38}$
Semi-Detached Model	$E = 3.0(KLOC)^{1.12}$	$T = 2.5(E)^{0.35}$
Embedded Model	$E = 3.6(KLOC)^{1.20}$	$T = 2.5(E)^{0.32}$

IV. Function Point Model

Software size helps in developing an initial estimate for software effort/cost estimation during software development life cycle. COCOMO model provided this estimate based on the SLOC. It was reported that SLOC produced many problems [26], [27]. For example, in modern software programming, auto-generate tools produced large number of line of codes. SLOC also changes with the developer's experience, difference in programming languages, variation in the graphical user interface (GUI) code generation, and lack of functionality. The estimation of SLOC under this condition seems uncertain to measure. This is why Albrecht proposed his idea of computing the software size based on the system functionality [28], [29].

A. Albrecht's Function Points

Albrecht's function point gained acceptance during the 1980's and 1990's because of the tempting benefits compared to the models based on the SLOC [30], [31]. Because FP is self-governing and independent of language type, platform, it can be used to identify many productivity benefits. FP is designed to estimate the time required for a software project development, and thereby the cost of the project and maintaining existing software systems.

In 1979 Albrecht [28], published his article on FP methodology while he was working at IBM. The proposed FP has no dimension. FP was computed based on the analysis of project requirements. The requirements help in identifying the number of function to be developed along with the complexity of each function. Thus, there was no need to measure the size of LOC but only concern about project functionality. Once the number of FP measured, the average number of function points per month specified and the labor cost per month is estimated; the total budget can be computed. Albrecht originally proposed four function types [28]: files, inputs, outputs and inquiries with one set of associated weights and ten General System Characteristics (GSC). In 1983, the work developed in (Albrecht and Gaffney 1983), proposed the expansion of the function type, a set of three weighting values (i.e. simple, average, complex) and fourteen General System Characteristics (GSCs) were proposed as given in Table II.

In [32], Kemerer provided a famous study reporting the results of the comparative accuracy for four software cost estimation models. They are the Function Points [28], SLIM [33], COCOMO [22], and ESTIMACS. The results were produced using data collected from 15 completed software projects. Each model was tested based on its predictive capability on computing software cost. The results showed that the models require substantial calibration. Kemer also identified the main attributes which affect software productivity. Recently, using Albrecht's Function Point analysis (FPA) method and using analogous approach, authors [34] provided a methodology they claim it is more reliable and accurate in predicting the software size at an early stage of the software life cycle. Recently, FP gain more attention as a powerful approach for estimating software effort [35]–[37].

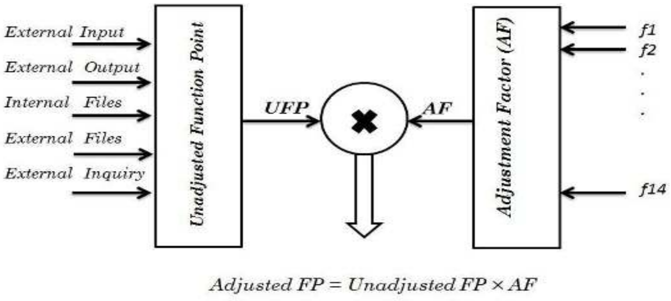

Fig. 1. Function Point Computation Model

In Albrecht FP, there are two parts in the model, which are *Unadjusted Function Point* (UFP) and *Adjusted Function Point* (AFP). The UFP consists of five components. They are:

- External Inputs (EI),

TABLE II. 1983 FUNCTION TYPES AND WEIGHTS

Function Type	Simple	Average	Complex
External Input	3	4	6
External Output	4	5	7
Internal Files	7	10	15
External Files	5	7	10
External Inquiry	3	4	6

- External Outputs (EO),
- External Inquires (EQ),
- Internal Logical Files (ILF) and
- External Interface Files (EIF).

There are also 14 GSCs factors that affect the size of the project effort, and each is ranked from ”0”- no influence to ”5”- essential. GSCs consists of 14 factors known as $f_1, f_2, \ldots, f_{14}$. These factors are listed in listed in Table III. The sum of all factors is then multiplied given in Equation 2 which constitute the Adjustment Factor (AF) defined in the range [0.65, -1.35].

$$AF = 0.65 + 0.01 \sum_{i=1}^{14} f_i \tag{2}$$

TABLE III. GENERAL SYSTEM CHARACTERISTICS (GSCS)

1	Data Communications
2	Distributed Functions
3	Performance
4	Heavily Used Configuration
5	Transaction Rate
6	Online Data Entry
7	End User Efficiency
8	Online Update
9	Complex Processing
10	Reusability
11	Installation Ease
12	Operational Ease
13	Multiple Sites
14	Facilitate Change

Then, the Unadjusted FP is then multiplied by the UFP to create the Adjusted Function Point (AFP) count as given in Equation 3. The Adjusted FP value is always within 35% of the original UFP figure. A diagram which shows the process of computing FP is given in Figure 1.

$$Adjusted\ FP = Unadjusted\ FP \times AF \tag{3}$$

V. WHAT IS FIS?

A block diagram which provide the main architecture of a fuzzy rule based system is shown in Figure 2. The proposed fuzzy logic system, used in this study, consists of number of components.

1) **Fuzzification:** In this stage, the model inputs and outputs variables are defined. These inputs and outputs are transformed to set of fuzzy domains.
2) **Inference Mechanism:** Fuzzy inference mechanism concerns on developing a relationship between the model inputs and outputs. The mapping constructs the system decision making. The process of fuzzy inference include: Membership functions, Fuzzy set operation, and If-Then rules. FIS may be summarized as two processes:
    - *Aggregation:* Compute the IF part (i.e. antecedent) of the rules. The antecedent variables reflect information about the process operating conditions.
    - *Composition:* Compute the THEN part (i.e. consequence) of the rules. The rule’s consequent is normally presented as a linear regression model [38]–[40]. This model has set of parameters usually estimated using least square minimization criterion.
3) **Defuzzification:** The computed output based on the fuzzy rules are then converted to real values.

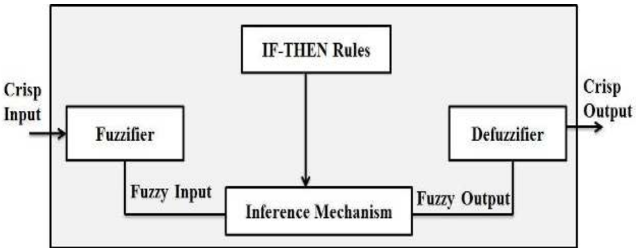

Fig. 2. The proposed fuzzy logic system

VI. PROPOSED FL MODEL

The proposed fuzzy model should be able to mathematically represents the relationship between the effort model inputs $x_1, \ldots x_n$ and the effort y; n is the number of inputs to the fuzzy model. The proposed fuzzy model is always represented by set of If-Then rules. The proposed fuzzy model equation is given as follows:

$$y = FM(x_1, \ldots, x_n) \tag{4}$$

The fuzzy region in the product space is developed based on the membership functions and the antecedent of the rule. The *antecedent* variable gives the condition of the process status now. The rule’s consequent is defined as a set of local linear regression models which relates y with $x_1, \ldots, x_4$ given as in Equation 5.

$$y = a_0 + a_1 x_1 + \cdots + a_n x_n \tag{5}$$

A rule-based fuzzy model requires the identification of the following:

1) the $antecedent$,
2) the $consequent$ structure,
3) the type of the membership functions for different operating conditions and
4) the estimation of the consequent parameters using least square estimation.

The developed fuzzy models implemented based the Takagi-Sugeno technique [38], [39]. The proposed technique does not require any *a prior* knowledge about the operating regimes. If a sufficiently number of measurements are collected which reflects the operating ranges of interest, the developed fuzzy model will be an efficient one [38], [39].

VII. Computation Criteria

The performance of the developed two models; the SLOC and the FP models based FL shall be evaluated using number of evaluation criteria. They are:

- Variance-Accounted-For (VAF):

$$VAF = [1 - \frac{var(y - \hat{y})}{var(y)}] \times 100\% \tag{6}$$

- Euclidian distance (ED):

$$ED = \sqrt{(\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{7}$$

- Manhattan distance (MD):

$$MD = (\sum_{i=1}^{n}|y_i - \hat{y}_i|) \tag{8}$$

- Mean Magnitude of Relative Error (MMRE):

$$MMRE = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{y_i} \tag{9}$$

where y and $\hat{y}$ are the actual effort and the estimated effort based on the developed fuzzy model and N is the number of measurements used in the experiments, respectively.

VIII. Experimental Results

A. The Albrecht data set

A statistical summary of the features used in the analysis of the Albrecht data set is displayed in Table IV. The feature to be predicted (i.e. goal feature or dependent variable) is effort, measured in work-hours, while the potential independent variables (i.e. descriptor features) are adjusted function points, the number of master files, the number of inputs, the number of inquiries and the number of outputs [41].

TABLE IV. Summary Statistics for Albrecht Data Set [41]

Feature	Count	Min	Max	Mean	Median
Effort	24	0.5	105.20	21.88	11.45
FP	24	199.00	1902.00	647.62	506.00
Files	24	3.00	60.00	17.38	11.50
Inputs	24	7.00	193.00	40.25	33.50
Inquiries	24	0.00	75.00	16.88	19.3
Outputs	24	12.00	150.00	47.25	39.00

B. Fuzzy Effort Model based SLOC

We developed a fuzzy model based COCOMO for the effort taking in consideration one attribute which is the SLOC. We used the FMID Matlab Toolbox [42] to develop our experimental results. The set of rules which describe the effort as a function of SLOC is given in Table V. In Table VI, we show the values of each evaluation criteria adopted in this study. In Figure 3, we show the membership function for the SLOC based model. Three membership functions are shown which reflect the relationship between the SLOC and the Effort on three sub-models. Figure 4 show the actual and estimated effort using fuzzy logic. The values of the actual and computed effort based fuzzy model is presented in Table VII. The characteristics between the two curves look very similar with high VAF criteria.

TABLE V. Fuzzy Rules for the Effort based SLOC Model

1. **If** $SLOC$ is A_1 **then**
   $E = 3.20 \cdot 10^{-1} SLOC + 1.57 \cdot 10^{-1}$
2. **If** $SLOC$ is A_2 **then**
   $E = 2.25 \cdot 10^{0} SLOC - 1.96 \cdot 10^{2}$
3. **If** $SLOC$ is A_3 **then**
   $E = -1.47 \cdot 10^{0} SLOC + 3.24 \cdot 10^{2}$

TABLE VI. Computation Criteria for the FL based SLOC Model

VAF	ED	MD	MMRE
96.158%	2.85×10^{-13}	100.27	0.4337

C. Fuzzy Effort Model based FP

We developed a fuzzy model for the effort taking in consideration four attribute inspired by the FP model. Three memberships were used. The set of rules which describe the effort as a function of FP is given in Table V. In Figure 5, we show the membership function for the FP based model. Figure 6 show the actual and estimated effort using fuzzy logic based FP model. We received a very high VAF reflecting good performance modeling. The actual and estimated values of the effort based the FP model is given in Table IX. The developed model's performance were computed using three different criteria as reported in Table X. It can be seen that the performance of the developed fuzzy models based historical data were able to achieve significant modeling results.

IX. Conclusions and Future Work

In this paper we studied the problem of effort estimation for software projects. This is a challenging problem for software project manager. We explored the use of fuzzy logic as a soft computing technique which can simplify the modeling process of the effort. Two models inspired from the COCOMO and FP were developed based fuzzy logic. The developed fuzzy models implemented based the Takagi-Sugeno technique. The developed fuzzy models were tested using the Albrecht data set reported in [41]. The models are simple and show the

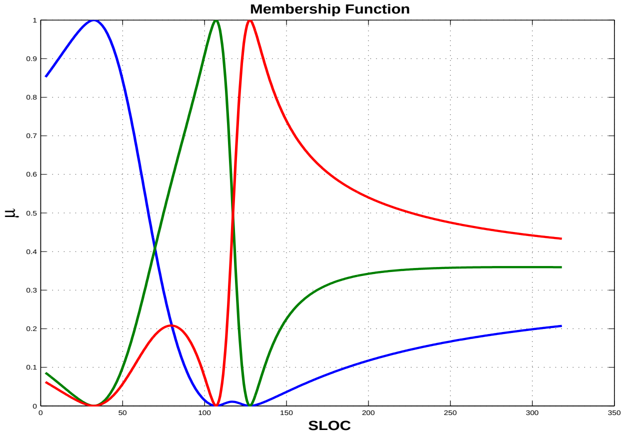

Fig. 3. Membership functions for the FL based SLOC Model

TABLE VIII. FUZZY RULES FOR THE EFFORT BASED FP MODEL

1. **If** $Inputs$ is A_{11} **and** $Outputs$ is A_{12} **and** $Files$ is A_{13} **and** $Inquiries$ is A_{14} **then**
   $FP = 5.30 \cdot 10^0 Inputs + 4.52 \cdot 10^0 Outputs - 8.10 \cdot 10^0 Files + 4.54 \cdot 10^0 Inquiries + 1.08 \cdot 10^2$

2. **If** $Inputs$ is A_{21} **and** $Outputs$ is A_{22} **and** $Files$ is A_{23} **and** $Inquiries$ is A_{24} **then**
   $FP = -3.55 \cdot 10^0 Inputs + 1.23 \cdot 10^1 Outputs + 1.12 \cdot 10^1 Files + 9.29 \cdot 10^0 Inquiries - 6.48 \cdot 10^1$

3. **If** $Inputs$ is A_{31} **and** $Outputs$ is A_{32} **and** $Files$ is A_{33} **and** $Inquiries$ is A_{34} **then**
   $FP = 7.08 \cdot 10^0 Inputs + 1.11 \cdot 10^1 Outputs + 1.45 \cdot 10^1 Files - 8.52 \cdot 10^0 Inquiries - 4.18 \cdot 10^2$

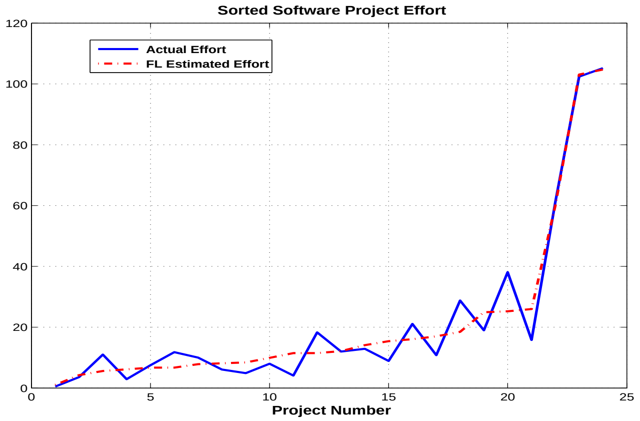

Fig. 4. Actual and estimated Effort for the FL based SLOC Model

TABLE VII. ACTUAL AND ESTIMATED EFFORT: FL-SLOC MODEL

SLOC	Effort	FL-Effort
3	0.5	1.0333
15	3.6	4.2453
20	11.0	5.6040
22	2.9	6.1542
24	7.5	6.7096
24	11.8	6.7096
28	10.0	7.8400
29	6.1	8.1276
30	4.9	8.4177
35	8.0	9.9177
40	4.1	11.4941
40	18.3	11.4941
42	12.0	12.1397
48	12.9	14.1040
52	8.9	15.4103
54	21.1	16.0532
57	10.8	16.9950
62	28.8	18.4773
93	19.0	24.9212
94	38.1	25.2247
96	15.8	25.9818
110	61.2	60.0224
130	102.4	103.0938
318	105.2	104.7296

mathematical relationship between the effort and the main model inputs. This work can be extended by exploring other forms of soft computing techniques.

ACKNOWLEDGMENTS

Dr. Sultan Aljahdali would like to acknowledge the financial support of King Abdulaziz City for Science and Technology and Taif University during his short research visit to Prof. Jayantha Herath with the Computer Science Department, St. Cloud State University, St. Cloud, MN 56301, USA on the Summer of 2013.

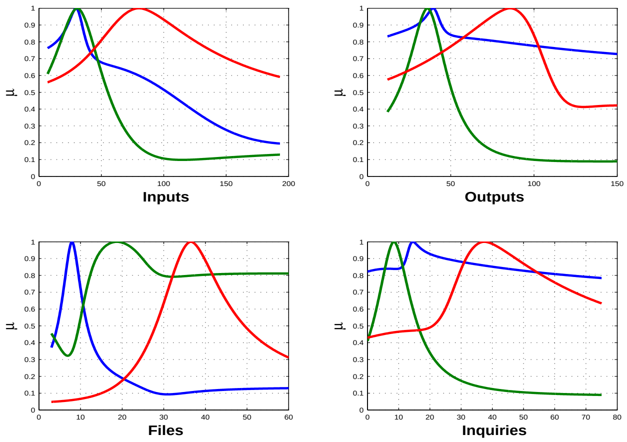

Fig. 5. Membership functions for the FL based FP Model

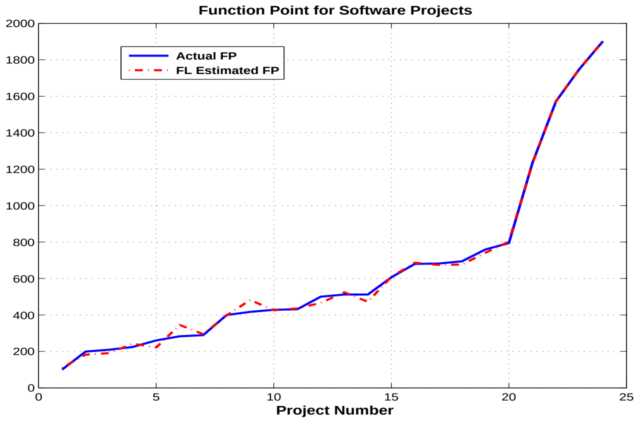

Fig. 6. Actual and estimated Effort for the FL based FP Model

REFERENCES

[1] C. F. Kemere, "An empirical validation of software cost estimation models," *Communication ACM*, vol. 30, pp. 416–429, 1987.

[2] J. W. Park R, W. Goethert, "Software cost and schedule estimating: A process improvement initiative," tech. rep., 1994.

[3] M. Boraso, C. Montangero, and H. Sedehi, "Software cost estimation: An experimental study of model performances," tech. rep., 1996.

[4] K. Pillai and S. Nair, "A model for software development effort and cost estimation," *IEEE Trans. on Software Engineering*, vol. 23, pp. 485–497, 1997.

[5] L. Pracchia, "Improving the dod: Software acquisition processes," *The Journal of Defense Software Engineering*, vol. 4, pp. 4–7, 2004.

[6] L. A. Zadeh, "Soft computing and fuzzy logic," *IEEE Softw.*, vol. 11, pp. 48–56, Nov. 1994.

TABLE IX. ACTUAL AND ESTIMATED EFFORT:FL-FP MODEL

Inputs	Outputs	Files	Inquiries	FP	FL-FP
34	14	5	0	100	110.67
15	15	3	6	199	181.88
7	12	8	13	209	190.7
33	17	5	8	224	241.97
12	15	15	0	260	221.13
13	19	23	0	283	345.08
17	17	5	15	289	294.78
27	20	6	24	400	397.29
28	41	11	16	417	482.22
70	27	12	0	428	425.33
10	69	9	1	431	435.66
25	28	22	4	500	466.19
41	27	5	29	512	524.29
28	38	9	24	512	472.31
42	57	5	12	606	609.17
45	64	16	14	680	686.46
43	40	35	20	682	674.52
61	68	11	0	694	676.44
40	60	12	20	759	741.23
40	60	15	20	794	802.39
48	66	50	13	1235	1231.9
69	112	39	21	1572	1573.2
25	150	60	75	1750	1749.9
193	98	36	70	1902	1903.3

TABLE X. COMPUTATION CRITERIA FOR THE FL BASED FP MODEL

VAF	ED	MD	MMRE
99.741%	2.78×10^{-12}	398.33	0.0495

[7] J. Ryder, *Fuzzy COCOMO: Software Cost Estimation*. PhD thesis, Binghamton University, 1995.

[8] A. C. Hodgkinson and P. W. Garratt, “A neuro-fuzzy cost estimator,” in *Proceedings of the Third Conference on Software Engineering and Applications*, pp. 401–406, 1999.

[9] M. A. Kelly, “A methodlolgy for software cost estimation using machine learning techniques,” Master’s thesis, Naval Postgratuate School, Monterey, California, 1993.

[10] J. J. Dolado and L. F. andez, “Genetic programming, neural network and linear regression in software project estimation,” in *Proceedings of the INSPIRE III, Process Improvement through training and education*, pp. 157–171, British Company Society, 1998.

[11] A. J. Albrecht and J. R. Gaffney, “Software function, source lines of code, and development effort prediction: A software science validation,” *IEEE Trans. Software Engineering*, vol. 9, no. 6, pp. 630–648, 1983.

[12] J. E. Matson, B. E. Barret, and J. M. Mellinchamp, “Software developemnet cost estimation using function points,” *IEEE Trans. Software Engineering*, vol. 20, no. 4, pp. 275–287, 1994.

[13] M. Shepper and C. Schofield, “Estimating software project effort using analogies,” *IEEE Tran. Software Engineering*, vol. 23, pp. 736–743, 1997.

[14] S. Kumar, B. A. Krishna, and P. Satsangi, “Fuzzy systems and neural networks in software engineering project management,” *Journal of Applied Intelligence*, vol. 4, pp. 31–52, 1994.

[15] A. F. Sheta, “Estimation of the COCOMO model parameters using genetic algorithms for NASA software projects,” *Journal of Computer Science*, vol. 2, no. 2, pp. 118–123, 2006.

[16] H. Mittal and P. Bhatia, “A comparative study of conventional effort estimation and fuzzy effort estimation based on triangular fuzzy numbers,” *International Journal of Computer Science and Security*, vol. 1, no. 4, pp. 36–47, 2007.

[17] M. Uysal, “Estimation of the effort component of the software projects using simulated annealing algorithm,” in *World Academy of Science, Engineering and Technology*, vol. 41, pp. 258–261, 2008.

[18] P. S. Sandhu, M. Prashar, P. Bassi, and A. Bisht, “A model for estimation of efforts in development of software systems,” in *World Academy of Science, Engineering and Technology*, vol. 56, pp. 148–152, 2009.

[19] A. Sheta, “Software effort estimation and stock market prediction using takagi-sugeno fuzzy models,” in *Proceedings of the 2006 IEEE Fuzzy Logic Conference, Sheraton, Vancouver Wall Centre, Vancouver, BC, Canada, July 16-21*, pp. 579–586, 2006.

[20] A. Sheta, D. Rine, and A. Ayesh, “Development of software effort and schedule estimation models using soft computing techniques,” in *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE CEC 2008) within the 2008 IEEE World Congress on Computational Intelligence (WCCI 2008), Hong Kong, 1-6 June*, pp. 1283–1289, 2008.

[21] J. W. Bailey and V. R. Basili, “A meta model for software development resource expenditure,” in *Proceedings of the International Conference on Software Engineering*, pp. 107–115, 1981.

[22] B. Boehm, *Software Engineering Economics*. Englewood Cliffs, NJ, Prentice-Hall, 1981.

[23] B. Boehm, *Cost Models for Future Software Life Cycle Process: COCOMO2*. Annals of Software Engineering, 1995.

[24] O. Benediktsson, D. Dalcher, K. Reed, and M. Woodman, “COCOMO based effort estimation for iterative and incremental software development,” *Software Quality Journal*, vol. 11, pp. 265–281, 2003.

[25] B. Boehm and et all, *Software Cost Estimation with COCOMO II*. Prentice Hall PTR, 2000.

[26] T. DeMarco, *Controlling Software Projects*. New York, USA: Yourdon Press/Prentice Hall, Englewood Cliffs, 1982.

[27] C. Jones, *Programming Productivity*. New York, USA: McGraw-Hill, 1986.

[28] A. J. Albrecht, “Measuring application development productivity,” in *Proceedings of the Joint SHARE, GUIDE, and IBM Application Developments Symposium*, pp. 83–92, 1979.

[29] A. J. Albrecht and J. E. Gaffney, “Software function, source lines of code, and development effort prediction: A software science validation,” *IEEE Transactions on Software Engineering*, vol. 9, no. 6, pp. 639–648, 1983.

[30] R. Rask, P. Laamanen, and K. Lyytinen, “A comparison of albrecht’s function point and symons’ mark ii metrics,” in *Proceedings of the thirteenth international conference on Information systems*, ICIS ’92, (Minneapolis, MN, USA), pp. 207–221, University of Minnesota, 1992.

[31] S. Furey, “Why we should use function points [software metrics],” *IEEE Softw.*, vol. 14, pp. 28, 30–, Mar. 1997.

[32] C. F. Kemerer, “An empirical validation of software cost estimation models,” *Commun. ACM*, vol. 30, pp. 416–429, May 1987.

[33] L. Putnam, “A general empirical solution to the macro software sizing and estimation problem,” *IEEE Transactionson Software Engineering*, vol. 4, no. 4, pp. 345–381, 1978.

[34] J. Wu and X. Cai, “A software size measurement model for large-scale business applications,” in *Proceedings of the 2008 International Conference on Computer Science and Software Engineering - Volume 02*, CSSE ’08, (Washington, DC, USA), pp. 39–42, IEEE Computer Society, 2008.

[35] L. Lavazza and S. Morasca, “Convertibility of function points into COSMIC function points: A study using piecewise linear regression,” *Inf. Softw. Technol.*, vol. 53, pp. 874–884, Aug. 2011.

[36] S. Choi, S. Park, and V. Sugumaran, “A rule-based approach for estimating software development cost using function point and goal and scenario based requirements,” *Expert Syst. Appl.*, vol. 39, pp. 406–418, Jan. 2012.

[37] L. Lavazza, S. Morasca, and G. Robiolo, “Towards a simplified definition of function points,” *Inf. Softw. Technol.*, vol. 55, pp. 1796–1809, Oct. 2013.

[38] R. Babuška, *Fuzzy Modeling and Identification*. PhD thesis, Delft Univesrsity of Technology, 1996.

[39] H. A. Babuška, R. Braake, A. J. Krijgsman, and H. B. Verbruggen, “Comparison of intelligent control schemes for real-time pressure control,” *Control Engineering Practice*, vol. 4, pp. 1585–1592, 1996.

[40] A. F. Sheta, *Modeling the Tennessee Eastman Chemical Reactor Using Fuzzy Logic*. New York, USA: Book Chapter. The ISE Book Series on Fuzzy System Engineering-Theory and Practice, published by Nova Science, ISBN: 3-540-25322-X, 2005.

[41] C. Schofield, *An Empirical Investigation into Software Effort Estimation by Analogy*. PhD thesis, Bournemouth University, 1998.

[42] R. Babuška, *Fuzzy Modeling and Identification Toolbox*. Delft University of Technology, The Netherland, http://lcewww.et.tudelft.nl/bubuska, 1998.