Fuzzy Logic for Enhancing the Sensitivity of COCOMO Cost Model
Abeer Hamdy
Computers and Systems Department, Electronics Research Institute &
Faculty of Informatics and Computer science, British University in Egypt Egypt
Abeer.hamdy@bue.edu.eg

ABSTRACT

The accuracy of algorithmic models for software cost prediction is limited due to their inability to handle imprecision and uncertainties associated with the software project attributes like size, programmer experience, etc. This paper enhances the accuracy and sensitivity of one of a widely used models COCOMO81 intermediate by incorporating a fuzzy component into the model. The fuzzy component deals with the vagueness and imprecision of the model’s cost drivers. MATLAB fuzzy toolbox was used in implementing the proposed fuzzy model. Artificial datasets was derived from COCOMONASA2 dataset to evaluate the proposed fuzzy model. The results showed that the sensitivity of the proposed fuzzy model is superior to COCOMO81 intermediate. It’s worth mentioning that the idea of the paper isn’t restricted to COCOMO81 intermediate; it could be applied to other algorithmic models.

Keywords: Fuzzy logic, software cost, COCOMO.

1. 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. In 1978, Putnam developed an early model known as SLIM [2]. In 1981, Boehm proposed cost estimation model, COCOMO 81 (Constructive Cost Model) [3],[4]. Several other algorithmic models have been proposed in the literature like function point analysis [5] and Use case 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 Lofty Zadeh [7], provides the concept of fuzzy sets to handle vague and inaccurate data.

Enhancing Intermediate COCOMO81 using fuzzy logic is the main emphasis of this paper for two reasons: 1) It’s a widely used model and 2) To use the publicly available COCOMO81 datasets (like COCOMONNASA2 [8]) in the experiments.

The paper is organized as follows: section 2 introduces COCOMO81 cost models. Section 3 discusses the imprecision problem associated with COCOMO and the proposed fuzzy solution. Section 4 discusses the experiments and results. Related work is introduced in section 5. While, section 6 concludes the paper and introduces the future research.

2. COCOMO81 COST MODELS

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:

\[ MM_{\text{eff}} = A \times \text{Size}^B \]  

eq.1

Where,

- \( MM_{\text{eff}} \): is the nominal estimated effort in terms of person per month;
- \( \text{Size} \): 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 table1):

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

Table 1: COCOMOI project modes, A & B values

<table>
<thead>
<tr>
<th>Development mode</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organic</td>
<td>2.4</td>
<td>1.05</td>
</tr>
<tr>
<td>Embedded</td>
<td>3.6</td>
<td>1.2</td>
</tr>
<tr>
<td>Semi-detached</td>
<td>3.0</td>
<td>1.12</td>
</tr>
</tbody>
</table>

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:

a) Product: RELY - Required software reliability, DATA - Data base size, CPLX - Product complexity.

b) Platform: TIME - Execution time, STOR- Main storage constraint, VIRT - Virtual machine volatility, TURN - Computer turnaround time.


d) 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 rate 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. For each rating there is a corresponding real number (multiplier factor) that affects the value of the nominal estimated effort as given by table 2.

Table 2: COCOMO81 intermediate effort multipliers

<table>
<thead>
<tr>
<th></th>
<th>very low</th>
<th>low</th>
<th>nominal</th>
<th>high</th>
<th>very high</th>
<th>extra high</th>
</tr>
</thead>
<tbody>
<tr>
<td>acsp</td>
<td>1.46</td>
<td>1.18</td>
<td>1.00</td>
<td>0.86</td>
<td>0.72</td>
<td>0.66</td>
</tr>
<tr>
<td>aexp</td>
<td>1.29</td>
<td>1.13</td>
<td>1.00</td>
<td>0.91</td>
<td>0.82</td>
<td>0.72</td>
</tr>
<tr>
<td>modp</td>
<td>1.24</td>
<td>1.10</td>
<td>1.00</td>
<td>0.91</td>
<td>0.82</td>
<td>0.72</td>
</tr>
<tr>
<td>nexp</td>
<td>1.21</td>
<td>1.10</td>
<td>1.00</td>
<td>0.90</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lexp</td>
<td>1.14</td>
<td>1.07</td>
<td>1.00</td>
<td>0.98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sced</td>
<td>1.23</td>
<td>1.06</td>
<td>1.00</td>
<td>1.10</td>
<td>1.10</td>
<td></td>
</tr>
<tr>
<td>stor</td>
<td>1.24</td>
<td>1.03</td>
<td>1.00</td>
<td>1.10</td>
<td>1.10</td>
<td>1.10</td>
</tr>
<tr>
<td>dets</td>
<td>0.94</td>
<td>1.00</td>
<td>1.00</td>
<td>1.16</td>
<td>1.16</td>
<td>1.16</td>
</tr>
<tr>
<td>turn</td>
<td>0.87</td>
<td>1.00</td>
<td>1.00</td>
<td>1.15</td>
<td>1.15</td>
<td>1.15</td>
</tr>
<tr>
<td>virt</td>
<td>0.87</td>
<td>1.00</td>
<td>1.00</td>
<td>1.15</td>
<td>1.15</td>
<td>1.15</td>
</tr>
<tr>
<td>rely</td>
<td>0.70</td>
<td>0.85</td>
<td>1.00</td>
<td>1.25</td>
<td>1.25</td>
<td>1.25</td>
</tr>
<tr>
<td>nplx</td>
<td>0.70</td>
<td>0.85</td>
<td>1.00</td>
<td>1.15</td>
<td>1.30</td>
<td>1.65</td>
</tr>
</tbody>
</table>

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 2) is used in equation 2. 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 effort adjustment factor.
(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.

3.2 Proposed Fuzzy Model

The definitions of the cost drivers 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 effort adjustment factor (EAF) that is used in equation 3 to adjust the nominal predicted effort instead of using the effort multipliers given by table 2.

Consider the ACAP cost driver. Figure 1 shows the rule base of the ACAP. Figure 2 shows the fuzzy sets of the antecedent part which are derived using the definition of the ACAP given by table 3. Figure 3 shows the fuzzy sets of the consequent part that are derived using the effort multipliers values given by table 2. Figure 4 shows the overall architecture of the proposed fuzzy model.

![Fig 1: ACAP Rule base](image1)

![Fig 2: Antecedents fuzzy sets of ACAP attribute](image2)

![Fig 3: Consequent cost driver of ACAP attribute](image3)

![Fig 4: The proposed Fuzzy Model](image4)

4. EXPERIMENTS AND RESULTS

MATLAB2010 fuzzy toolbox has been used in the experiments.

4.1 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 contain only COCOMO81 effort multipliers (they don’t contain the real attributes of the projects). So we have generated four artificial datasets based
on the COCOMONASA2 dataset and the definitions of the effort multipliers. Each generated dataset replaces each COCOMO81 effort multiplier with a suitable real project attribute. These project attributes are used as inputs to the proposed fuzzy model.

4.2 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:

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

A value of 25% for MRE is acceptable.

Another widely used and more accurate measure [9] 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.

Pred (.25) is the metric used in this paper.

4.3 Results

Figure 5 shows the values of the effort adjustment factor (EAF) produced by the proposed fuzzy model for each project ID in each of the four artificial datasets. It is shown that each project ID has four different values of EAF as it has different attribute values in each dataset. While, applying COCOMO81 produces a single EAF value for each project ID over the four datasets.

Table 4 summarizes the values of the pred (25%) result from applying our fuzzy model on the four artificial datasets. Applying COCOMO81 produces the same value of pred (25%).

<table>
<thead>
<tr>
<th>DataSet #1</th>
<th>DataSet #2</th>
<th>DataSet #3</th>
<th>DataSet #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fuzzy</td>
<td>48.43%</td>
<td>54.85%</td>
<td>46%</td>
</tr>
<tr>
<td>COCOMO</td>
<td>47%</td>
<td>47%</td>
<td>47%</td>
</tr>
</tbody>
</table>

These results show that fuzzy model is more sensitive to changes in the project attributes than COCOMO.

5. 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 is 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.

Neural Network (NNet) is another AI technique that has proved its effectiveness in solving effort estimation problem. Dave [17] showed that NNET in general is better than regression analysis and Radial Bases NNET (RBNN) is better than Feed Forward NNET (FFNN). Du [18] and Huang [19] Used Neuro-fuzzy techniques for improving COCOMO model. Support vector machines (SVR) [20] and data mining techniques [21-24] are also candidate techniques to tackle this problem. Recently, Kazemifard [25] suggested new project attributes which are the emotions of the team and used multi-agents to model the team emotions.

A survey for effort estimation models can be found in Khatib [26] and Keung [27].

6. CONCLUSION AND FUTURE RESEARCH

Our objective is developing an adaptive fuzzy model for software effort estimation. This research is a first step towards this objective. A fuzzy logic-based component has been added to COCOMO81 intermediate model to
improve its accuracy and sensitivity. The experimental results are promising. They showed that our model is more sensitive than COCOMO.

It is known that the performance of the fuzzy models depend on parameters of the fuzzy sets, so I plan to improve the results of the proposed model by using a training algorithm like genetic algorithms (GA) to tune the fuzzy sets parameters. Moreover, a complete adaptive fuzzy model will be developed, where COCOMO formula will be replaced by a fuzzy expert system.

REFERENCES


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


