Fuzzy Analogy: A New Approach for Software Cost Estimation

Ali Idri
ENSIAS, BP. 713, Agdal, Rabat, Morocco
idri@enisas.um5souissi.ac.ma

Alain Abran
École de Technologie Supérieure
Montréal, Canada
abran.alain@uqam.ca

Taghi M. Khosgoftaar
Florida Atlantic University
Boca Raton, USA
taghi@cse.fau.edu

Abstract

Estimation models in software engineering are used to predict some important attributes of future entities such as software development effort, software reliability and programmers productivity. Among these models, those estimating software effort have motivated considerable research in recent years. Estimation by analogy is one of the most attractive technique in software effort estimation field. However, the procedure used in estimation by analogy is not yet able to handle correctly categorical data such as 'very low', 'complex' and 'average'. In this paper, we propose a new approach based on reasoning by analogy, fuzzy logic and linguistic quantifiers to estimate effort when the software project is described either by categorical or numerical data.

1. Introduction

Estimating the work-effort and the schedule required to develop and/or to maintain a software system is one of the most critical activities in managing software projects. This task is known as Software Cost Estimation. During the development process, the cost and time estimates are useful for the initial rough validation and monitoring of the project's progress; after completion, these estimates may be useful for project productivity assessment for example. Several cost estimation techniques are used within an organisation; these techniques may be grouped into two major categories [26]:

- algorithmic models, and
- non-algorithmic models

The first is the most popular techniques (at least in the literature), and is illustrated by estimations models such as COCOMO [5,6,8], PUTNAM-SLIM[24], and function points analysis [2, 21]. Algorithmic models are derived from the statistical or numerical analysis of historical projects data (simple/multiple/stepwise regression, bayesian approach, Principal Components Analysis, polynomial interpolation, …). The disadvantages of these models are:

- They make assumptions about the form of the prediction function, that almost is: $Effort = a \times size^{b}$ where α represents a productivity coefficient and β an economies (or diseconomies) of scale coefficient
- They need to be adjusted or calibrated to local circumstances (an example of calibrating and reformulating COCOMO'81 model is published in [10])
- They are not very understandable because there are no effective natural interpretations to their behaviour.

Next, the non-algorithmic models are developed to avoid the above weaknesses. Recently, many researchers have begun to turn their attention to this alternative, and in particular to a set of approach based on neural networks, regression trees, rule induction, and case-based reasoning. These alternatives have some advantages:

- Capabilities to adequately model the complex set of relationship between factors (cost drivers) or between effort (the independent variable) and the cost drivers (dependent variables)
- Capabilities to learn from historical projects data (specifically for neural networks)
- Contrary to the algorithmic models, their behaviour is easier to understand; an exception can be made for neural networks which are considered as 'black boxes'. Recently, this significant weakness can be avoided by establishing not just the equivalence but the equality between Artificial Neural Networks (ANN's) and Fuzzy Rule-Based Systems (FRBS's) [4].

This paper concerns the case-based reasoning. This approach is an enhanced form of estimation by analogy [8]. Boehm has suggested an informal use of analogies between software projects as one of seven possible techniques for software cost estimation [5]. Recently, it has been presented by Shepperd et al. in the form of a formal detailed methodology and has been applied on a number of software projects data sets [19, 25, 26]. After years of application of the estimation by analogy, it has not seemed to generate often more accurate results than traditional regression based techniques. Indeed, Shepperd et al., Niessink and van Vliet found that estimation by analogy generated better results than stepwise regression [23, 25, 26]; by contrast, Briand et

al., Stensrud and Myrtveit reported the reverse, namely that regression based analysis generated more accurate models than using analogy [7, 22]. Recent research has been initiated to explain the relationship which exists between different properties of the dataset (size, number of attributes, …) and the accuracy of a prediction system [27]. Beyond this interesting issue, in this work we deal with an important limitation of estimation by analogy which arises when software projects are described by categorical data (nominal or ordinal scale). Here the principal problem is in the evaluation of the similarity between two software projects when theirs attributes are measured by qualifications such as 'very low', 'low' and 'high'; these qualifications are called linguistic values in fuzzy logic [1]. To overcome this limitation, we have developed and validated a set of candidate measures for software projects similarity. These measures are based on fuzzy sets, fuzzy reasoning and linguistic quantifiers [12, 13, 14]. Consequently, the purpose of this paper is to provide a new approach to estimate effort by analogy when software projects are described either by numerical (interval, ratio or absolute scale) or categorical (nominal and ordinal scale) data

This paper is organized as follows: In the first section, we briefly outline the principles of fuzzy logic and linguistic quantifiers. In the second section, we present the classical procedure of estimation by analogy; which procedure cannot handle categorical data. To overcome this limitation, we propose, in the fourth section, a new approach which can be seen as a fuzzification of the classical approach of estimation by analogy. We illustrate, by means of the COCOMO'81 dataset, the computing process of each step composing the life cycle of our approach. A conclusion and overview of future work conclude this paper.

2. Fuzzy Logic and Linguistic Quantifiers

2.1 Fuzzy Logic

Since its foundation by Zadeh in 1965 [32], Fuzzy Logic (FL) has been the subject of important investigations. At the beginning of the nineties, fuzzy logic was firmly grounded in terms of its theoretical foundations and its application in the various fields in which it was being used (robotics, medicine, image processing, etc.).

According to Zadeh [34], the term "fuzzy logic" is currently used in two different senses. In a narrow sense, FL is a logical system aimed at a formalization of approximate reasoning. In a broad sense, FL is almost synonymous with fuzzy set theory. Fuzzy set theory, as its name suggests, is basically a theory of classes with unsharp boundaries. It is considered as an extension of classical set theory. The membership function $\mu_A(x)$ of x in a classical set A, as a subset of the universe X, is defined by:

$$m_A(x) = \begin{cases} 1 & iff \ x \in A \\ 0 & iff \ x \notin A \end{cases}$$

This means that an element x is either a member of set A ($\mu_A(x)=1$) or not ($\mu_A(x)=0$). Classical sets are also referred to as crisp sets. For many classifications, however, it is not quite clear whether x belongs to a set A or not. For example, in [16], if set A represents PCs which are too expensive for a student's budget, then it is obvious that this set has no clear boundary. Of course, it could be said that a PC priced at $2500 is too expensive, but what about a PC priced at $2495 or $2502? Are these PCs too expensive? Clearly, a boundary could be determined above which a PC is too expensive for the average student, say $2500, and a boundary below which a PC is certainly not too expensive, say $1000. Between those two boundaries, however, there remains an interval in which it is not clear whether a PC is too expensive or not. In this interval, a grade could be assigned to classify the price as partly too expensive. This is where fuzzy sets come in: sets in which the membership has grades in the interval (0,1). The higher the membership x has in fuzzy set A, the more true it is that x is A.

The fuzzy set, introduced by Zadeh, is a set with graded membership in the real interval (0,1). It is denoted by:

$$A = \int_X m_A(x) / x$$

where $\mu_A(x)$ is known as the membership function and X is known as the universe of discourse. Figure 1 shows two representations of the linguistic value 'too expensive'; the first using a fuzzy set (Figure 1 (a)) and the second using a classical set (Figure 1 (b)). The major advantage of the fuzzy set representation is that it is a gradual function rather than an abrupt-step function between the two boundaries of $1000 and $2500.

Among the other branches of fuzzy set theory are fuzzy arithmetic, fuzzy graph theory and fuzzy data analysis.

2.2 Linguistic Quantifiers

A large number of linguistic quantifiers are used in human discourse. In [33], Zadeh distinguishes between two classes of linguistic quantifiers: absolute and proportional. Absolute quantifiers such as 'about 10' and 'about 20' can be represented as a fuzzy set Q of

[1] In the rest of this paper, categorical or linguistic values will be used as synonymous.

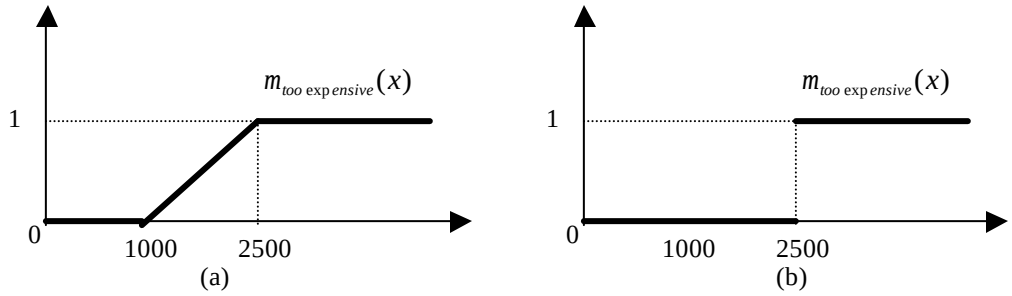

Figure 1. Fuzzy set (a) and Classical set (b) for the linguistic value 'too expensive'

the non-negative reals. In this work, we are concerned with proportional quantifiers.

A proportional linguistic quantifier indicates a proportional quantity such as 'most', 'many' and 'few'. Zadeh has suggested that proportional quantifiers can be represented as fuzzy set Q of the unit interval I. In this representation, for any $r \in I$, Q(r) is the degree to which the proportion r satisfies the concept represented by the term Q. Furthermore, Yager distinguished three categories of proportional quantifier [29, 30]:

(a) A Regular Increasing Monotone (RIM) quantifier such as 'many', 'most' and 'at least α' is represented as fuzzy subset Q satisfying the followings conditions:
   1- Q(0)=0,
   2- Q(1)=1, and
   3- $Q(x) \geq Q(y)$ if x>y

(b) A Regular Decreasing Monotone (RDM) Quantifier such as 'few' and 'at most α' is represented as fuzzy subset Q satisfying the followings conditions:
   1. Q(0)=1,
   2. Q(1)=0, and
   3. $Q(x) \leq Q(y)$ if x>y

(c) A Regular UniModal quantifier, such as 'about α', is represented as fuzzy subset Q satisfying the followings conditions:.
   1- Q(0)=0,
   2- Q(1)=1, and
   3- There exist two values a and $b \in I$, where a<b, such that:
      i. For y <a, $Q(x) \leq Q(y)$ if x<y
      ii. For $y \in [a,b]$, Q(y)=1
      iii. For y>b, $Q(x) \geq Q(y)$ if x<y

Two interesting relationships exist between these three categories of proportional quantifiers:

- If Q is an RIM quantifier, then its antonym is an RDM quantifier and vice versa: Examples of these antonym pairs are 'few' and 'many', and 'at least α' and 'at most α'.
- Any RUM quantifier can be expressed as the intersection of an RIM and an RDM quantifier.

3. Estimation by analogy

Estimation by analogy is essentially a form of Case-Based Reasoning. Case-Based Reasoning has four steps [1]:

1- Retrieve the most similar case or cases
2- Reuse the information and knowledge in that case to solve the problem
3- Revise the proposed solution
4- Retain the parts of this experience likely to be useful for future problem solving

In the situation of effort estimation, CBR is based on the following affirmation: *similar software projects have similar cost*. It has been deployed as follows: First, each project must be described by a set of attributes which must be relevant and independent. Second, we must determine the similarity between the candidate project and each project in the historical database. Third, we use the known effort values from the historical projects to derive an estimate for the new project; this later step is known as case adaptation.

Since it was first used by Vancinanza et al. [28] who suggested that CBR might be usefully adapted to make accurate software effort predictions, estimation by analogy has been the subject of studies aimed at to evaluating, enhancing, reformulating and adapting the CBR life cycle according to the features of the software effort prediction context. Shepperd et al. has been involved in the development of CBR techniques and tools to build software effort prediction systems for five years ago [25, 26]. In their recent work, they tried to explain why different research teams have reported widely different results by using CBR technology; other that the characteristics of the historical software projects database being used, Sheppered et al.

examined the impact of the choice of number of analogies and adaptation strategies when making predictions by using a dataset of software projects collected by a Canadian software house. They found that, first, choosing analogies is important; more specifically, three analogies seemed to be optimal although a fixed value for k (number of analogies) was more effective for the larger dataset while distance based analogies selection appeared more effective for the smaller dataset. Second, case adaptation strategies seemed to have little impact on the accuracy of the estimation by analogy [19].

Angelis and Stamelos, when studying the estimation by analogy method for Albrecht's software projects, explored the problem of determining the parameters for configuration of the analogy procedure before its application to a new software project. Indeed, they studied three parameters. First, the choice of distance measure that will be used to evaluate the similarity between software projects. Second, the number of analogies to take into account in the effort estimation. Third, the statistic that will be used to calculate the unknown effort from the efforts of the similar projects. They suggested that these three parameters must be configured by using the bootstrap method which consists of drawing news samples of software projects from the available dataset and testing the performance of the chosen parameters on these large numbers of samples. This is allow to the user to identify which parameters values give often accurate estimates; so, these values can be used to generate prediction for a new software project. This kind of search for optimal parameters is called calibration of the estimation procedure [3].

However, even tough it is well recognized that estimation by analogy is a promising technique to contribute to the software estimation problem, there are certain limitations that prevent it from being more popular. The most important is that until now it cannot handle categorical data such as 'very low', 'low' and 'high' whereas many factors such as experience of programmers, complexity of modules and software reliability are measured on at least an ordinal or nominal scale. For example, the well-known COCOMO'81 model uses 15 attributes out of 17 (22 out of 24 in the COCOMOII) which are measured with six linguistics values: 'very low', 'low', 'nominal', 'high', 'very high' and 'extra-high' [5, 6, 8]. Another example is the Function Points measurement method, in which the level of complexity for each item (input, output, inquiry, logical file or interface) is assigned using three qualifications ('low', 'average' and 'high'). Then there are the General System Characteristics, the calculation of which is based on 14 attributes measured on an ordinal scale of six linguistic values (from 'irrelevant' to 'essential') [15]. To overcome this limitation, we present in the next section a new method which can be seen as a fuzzification of the classical analogy to deal with categorical data. This method will be christened, in the rest of this paper, Fuzzy Analogy.

4. Estimation by Analogy using Fuzzy Logic: Fuzzy Analogy

The key activities for estimating software project effort by analogy are the identification of a candidate software project as a new case, the retrieval of similar software projects from a repository, the reuse of knowledge derived from previous software projects (essentially the actual effort) to generate an estimate for the candidate software project. Estimation by analogy has motivated considerable research in recent years. However, none has yet dealt with categorical data. We present here a new approach based on reasoning by analogy and fuzzy logic which extends the classical analogy in the sense that it can be used when the software projects are described either par numerical or categorical data.

Fuzzy Analogy is a fuzzification of the classical analogy procedure. It is also composed of three steps: identification of cases, retrieval of similar cases and case adaptation; each step is a fuzzification of its equivalent in the classical analogy procedure. In the following sub-sections, each step is further detailed.

4.1 Identification of cases

The goal of this step is the characterization of all software projects by a set of attributes. Selecting attributes which well describe software projects is a complex task in the analogy procedure. Indeed, the selection of attributes depends on the objective of the CBR system; in our case, the objective is to estimate the software project effort. Consequently, the attributes must be relevant for the effort estimation task. The problem is how to know all attributes which exhibit a significant relationship with effort in a given environment? The solution adopted by cost estimation researchers and practitioners is to test the correlation between effort and all attributes for which data in the studied environment is available. So, this solution does not take into account attributes, which can affect largely the effort, if they have not yet recorded data. Another interesting criteria that each relevant attribute must satisfy is the independence with respect to the other attributes. Shepperd et al., in the ANGEL tool, propose to resolve the attributes selection problem by applying a brute force search of all possible attributes subsets. They recognized that this is an NP-hard search problem and consequently this is not feasible solution

where the number of the candidate attributes is large. On the other hand, Briand et al. propose to use a t-test procedure to select the set of attributes. Sheppered claimed that this is not a good solution because the stepwise procedure is not efficient to model the potential interactions between the software project attributes [19]. There are two other criteria, which have not yet been the subject of in dept-study in the cost estimation literature, to which each relevant and independent attribute must obey: the attribute must be comprehensive which means that must be well defined and the attribute must be operational which means that must be easy to measure. We believe that the best way to solve the attributes selection problem is by integrating a learning procedure in the analogy approach. We are deferring here to give more details about this alternative until we develop a complete learning procedure for our Fuzzy Analogy procedure. Before the learning phase and in the training phase of our approach, we adopt a variation of Sheppered's solution by allowing to the estimators the freedom to utilise the attributes that they believe best characterize their projects and more appropriate to their environment.

The objective of our Fuzzy Analogy approach is to deal with categorical data. So, in the identification step, each software project is described by a set of selected attributes which can be measured by numerical or categorical values. These values will be represented by fuzzy sets. In the case of a numerical value x_0 (no uncertainty), its fuzzification will be done by the membership function which takes the value of 1 when x is equal to x_0 and 0 otherwise. For categorical value, let us suppose that we have M attributes and for each attribute V_j, a measure with linguistic values is defined (A_k^j). Each linguistic value, A_k^j, is represented by a fuzzy set with a membership function ($m_{A_k^j}$). It is preferable that these fuzzy sets satisfy the *normal condition* as it was defined in [13]. The use of fuzzy sets to represent categorical data, such as 'very low' and 'low', mimics the way in which humans interpret these values and consequently it allows us to deal with vagueness, imprecision and uncertainty in the case identification step. Another advantage of our Fuzzy Analogy approach is that it takes into account the importance of each selected attribute in the cases identification step; indeed, it is obvious that all selected attributes have not necessarily the same influence on the software project effort. So, we are required to indicate the weights, u_k, associated with all selected attributes in the cases identification step.

To illustrate the cases identification step, we use the COCOMO'81 dataset. Each software project in this dataset is described by 17 attributes which are declared relevant and independent [5]. Among these, the DATA cost driver is measured by four linguistic values: 'low', 'nominal', 'high' and 'very high'. These linguistic values are represented by classical intervals in the original version of the COCOMO'81; because the advantage of the representation by fuzzy sets rather than classical intervals, we have proposed to use the representation given in figure 2. The weight associated to the DATA cost driver, u_{data}, is equal to 1.23. It is evaluated by its productivity ratio[2].

4.2 Retrieval of similar cases

This step is based on the choice of a software project similarity measure. This choice is very important since it will influence which analogies are found. In the literature, most researchers have used the Euclidean distance when the projects are described by numerical data and the equality distance when they are described by categorical data [26]. These two measures are not suitable when the categorical data are represented by fuzzy sets. Consequently, we have proposed a set of candidate measures for software project similarity to avoid this limitation [12]. These measures evaluate the overall similarity of two projects P_1 and P_2, $d(P_1, P_2)$, by combining the individual similarities of P_1 and P_2 associated with the various linguistic variables (attributes) (V_j) describing P_1 and P_2, $d_{v_j}(P_1, P_2)$. After an axiomatic validation of some proposed candidate measures for the individual distances $d_{v_j}(P_1, P_2)$, we have retained two measures [13]:

$$d_{v_j}(P_1,P_2) = \begin{cases} \max_k \min(m_{A_k^j}(P_1), m_{A_k^j}(P_2)) \\ \max - \min \ \textit{aggregation} \qquad (1.1) \\ \sum_k m_{A_k^j}(P_1) \times m_{A_k^j}(P_2) \\ \textit{sum} - \textit{product aggregation} \qquad (1.2) \end{cases}$$

where V_j are the linguistic variables describing projects P_1 and P_2, A_k^j are the fuzzy sets associated with V_j, and $m_{A_k^j}$ are the membership functions representing fuzzy sets A_k^j.

To evaluate the overall distance of P_1 and P_2, the individual distances $d_{v_j}(P_1, P_2)$ are aggregated by using RIM linguistic quantifiers such as 'all', 'most', 'many', 'at most α' or 'there exists'. The choice of the appropriate RIM linguistic quantifier, Q, depends on

[2] The productivity ratio is the project's productivity ration expressed in Delivered Source Instructions by Man-Months for the best possible attribute rating to its worst possible variable rating, assuming that all the ratings for all other attributes remain constant.

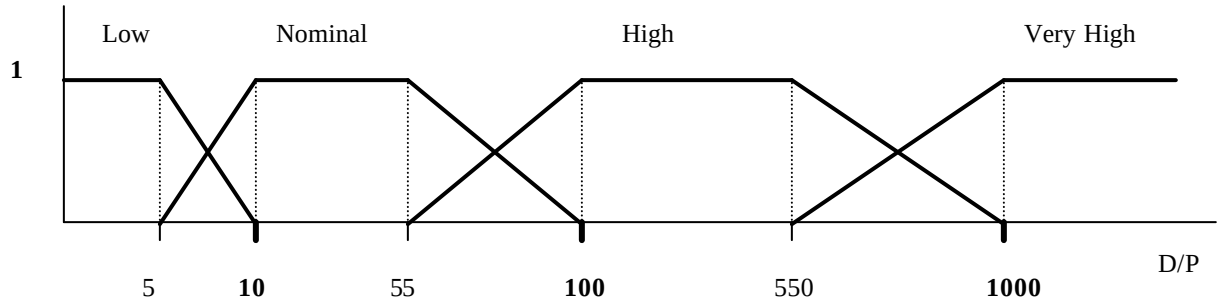

Figure 2. Membership functions of fuzzy sets defined for the DATA cost driver [13]

the characteristics and the needs of each environment. Q indicates the proportion of individual distances that we feel is necessary for a good evaluation of the overall distance. The overall similarity of P_1 and P_2, d(P_1 , P_2) is given by one of the following formulas [14]:

$$d(P_1,P_2)=\begin{cases} all\ of\ (d_{v_j}(P_1,P_2)) \\ most\ of\ (d_{v_j}(P_1,P_2)) \\ many\ of\ (d_{v_j}(P_1,P_2)) \\ .... \\ there\ exists\ of\ (d_{v_j}(P_1,P_2)) \end{cases}$$

When choosing the appropriate RIM linguistic quantifier to guide the aggregation of the individual distances, its implementation is realized by an Ordered Weight Averaging operator [14, 30]. Consequently, the overall distance, d(P_1 , P_2), is calculated by means of the following formula:

$$d(P_1,P_2)=\sum_{j=1}^{M} Q\left(\frac{\sum_{k=1}^{j} u_k}{T}\right) - Q\left(\frac{\sum_{k=1}^{j-1} u_k}{T}\right) d_{v_j}(P_1,P_2)$$

where $d_{v_j}(P_1,P_2)$ is the j^{th} largest individual distance, u_k is the importance weight associated with the k^{th} variable describing the software project, and T is the total sum of all importance weights u_k which are provided in the cases identification step.

To illustrate the retrieval of similar cases step, we suppose that 'most' is the appropriate RIM linguistic quantifier for the COCOMO'81 dataset and it is represented by the fuzzy subset $Q_{most}(r)=r^3$. Table 1 shows the results obtained for the overall similarity, between the first project and the first five projects in the COCOMO'81 dataset, using the max-min aggregation (formula 1.1) to evaluate the individual similarities.

		Max-min aggregation $d_{v_j}(P_1,P_n)$				
		$d(P_1, P_n)$				
		P_1	**P_2**	**P_3**	**P_4**	**P_5**
P_1	**Most**	0.69875	2.0948 E-02	2.9783 E-03	8.4882 E-02	4.4606 E-03

Table 1. Results obtained for $d(P_1, P_i)$ when aggregation uses the 'most' linguistic quantifier

4.3 Case adaptation

The objective of this step is to derive an estimate for the new project by using the known effort values of similar projects. There are two problems here. First, how many similar projects will be used in the adaptation? In the literature, one can notice that there is no clear rule to guide the choice of the number of analogies, *k*. Shepperd et al. have tested two strategies to calculate the number *k*: a) it can be set to some constant value, they explore values between 1 and 5; b) it can be determined dynamically as the number of cases that fall within distance d, of the new project [19]. Briand et al. have used a single analogy [7]. Angelis and Stamelos have tested a number of analogies in the range of 1 to 10 when studying the calibration of the analogy procedure for the Albrecht's dataset [3]. The results obtained from these experimentations seemed to favour the case where *k* is equal to 2 or 3.

We are not convinced by the approach fixing the number of analogies to be considered in the case adaptation step. The principle of this approach is to take only the *k* first projects which are similar to the new project. Let us suppose that the distances between the first three projects of one dataset (P_1, P_2, P_3) and the new project (P) are respectively: 3.30, 4.00 and 4.01; consequently, when we consider *k* is equal to 2, we will use only the two projects P_1 and P_2

in the calculation of an estimate of P; the project P_3 will not be considered in this case although there is no clear difference between d(P_2, P), 4.00, and d(P_3, P), 4.01!! We believe that the use of the number *k* hides behind the use of the classical logic principle: each project in the dataset is either or not similar to the new project. In our approach, we propose to use all of the projects in the dataset to derive an estimate of the new project. Each historical project will contribute, in the calculation of the effort of the new project, according to its degree of similarity with this project. This alternative seems not to be in conformity with what it is recognized within the cost estimation researchers community. Indeed, it is unanimously established that increasing number of analogies lead to extremely high level of estimation error[3, 19, 25, 26]. This is not true in our Fuzzy Analogy approach because there are significant differences between our similarity measures and Euclidean (or equality) distance [12, 13, 14]. Moreover, d(P_1 , P_2) using Euclidean distance has no clear natural interpretation; contrary to our similarity measures where d(P_1, P_2) is a membership function which expresses the truth value of the fuzzy proposition 'P_1 and P_2 are similar' [13, 14].

The second question in this step is how to adapt the chosen analogies in order to generate an estimate for the new project? The most used formulas are those using the (weighted) mean or the median of the *k* chosen analogies. In the case of weighted mean formula, the weights can be the similarity distances or the ranks of the projects. For our Fuzzy Analogy approach, we use the weighted mean of all known effort projects in the dataset; the weights are our similarity distances. The formula is then:

$$Effort(P) = \frac{\sum_{i=1}^{N} d(P, P_i) \times Effort(P_i)}{\sum_{i=1}^{N} d(P, P_i)}$$

- If d(P,P_i) is null then this implies that P_i does not influence the estimated effort of P. This is reasonable because d(P,P_i) is the truth value of the fuzzy proposition 'P and P_i are similar'.
- If d(P,P_i) is equal to 1 then this implies that P_i influences to the maximum the estimated effort of P.
- If d(P,P_i) is between 0 and 1 then this implies that P_i influences partially, according to the value of d(P,P_i), the estimated effort of P.
- If all d(P,P_i) are equal to 0 then the effort of P is indeterminate. This is the case if all projects in database differ very widely in nature from P; so, the effort of P can be any value including all the known effort values of historical software projects P_i in the dataset.

To illustrate the case adaptation step, we calculated the estimated effort for the first project, P1, in the COCOMO'81 dataset by considering only the first five projects (P_1, P_2, P_3, P_4, P_5) for which the similarity distances are given in table 1. We found that the estimated effort is equal to 1824Man-Months whereas the actual effort is 2040Man-Months.

5. Conclusions and Future Work

In this paper, we have proposed a new approach to estimate the software project effort. This approach is based on reasoning by analogy, fuzzy logic and linguistic quantifiers. Such an approach can be used when the software projects are described by categorical and/or numerical data. Thus, our approach improves the classical analogy procedure which does not take into account categorical data. In the Fuzzy Analogy approach, both categorical or numerical data are represented by fuzzy sets. The advantage of this is to handle correctly the imprecision and the uncertainty when describing a software project. Also, by using RIM linguistic quantifier to guide the aggregation of the individual similarities between two projects, the Fuzzy Analogy approach can be easily adapted and configured according to the needs of each environment. There are two other characteristics which can be integrated in our approach:

- first, it can learn from previous situations in order to generate more accurate predictions. The learning procedure can be implemented in the three steps of our approach. For example, in the identification step, it will be possible to propose a set of attributes which have often led to accurate results. This set of attributes will be used to describe the new project and consequently the attributes selection problem can be solved by learning.
- Second, our approach must handle the uncertainty when estimating the effort of the new project. Indeed, Kitchenham and Linkman have suggested that it is safer to produce interval estimates with a probability distribution rather than a point estimates which could lead to wrong managerial decisions and project failure. The estimation by interval with probability distribution provides the basis for risk analysis [18].

When integrating these two characteristics in the Fuzzy Analogy approach, it will satisfy the famous concept 'Soft Computing' defined by Zadeh in [34].

To complete this work, the Fuzzy Analogy prediction system must be validated. According to Fenton, a

prediction system is valid if it generates accurate predictions [9]. Further research has been initiated to validate our effort prediction system.

6. References

[1] A. Aamodt, E. Plaza, "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches", AI Communications, IOS Press, Vol. 7, no 1, 1994, pp. 39-59.

[2] A. Abran, P.N. Robbiard, "Functions points analysis: an empirical study of its measurement processes", IEEE Trans. on Software Engineering, Vol. 22, no 12, 1996, pp. 895-909.

[3] L. Angelis, I. Stamelos, "A Simulation Tool For Efficient Analogy Based Cost Estimation", Empirical Software Engineering, Vol. 5, no 1, 2000, pp. 35-68.

[4] J.M. Benitez, J.L. Castro, and I. Requena, "Are Artificial Neural Networks Black Boxes?", *IEEE Transactions on Neural Networks*, Vol. 8, no 5, September, 1997, pp. 1156-1164.

[5] B.W. Boehm, *Software Engineering Economics*, Prentice-Hall, 1981.

[6] B.W. Boehm, and *al*., "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0", *Annals of Software Engineering on Software Process and Product Measurement*, Amsterdam, 1995.

[7] L. Briand, T. Langley, I. Wieczorek., "Using the European Space Agency data set: A replicated assessment and comparison of common software cost modeling", In Proc 22th IEEE International Conference on Software engineering, Limerik, Ireland, 2000.

[8] D.S. Chulani, "Incorporating Bayesian Analysis to Improve the Accuracy of COCOMO II and Its Quality Model Extension", Ph.D. Qualifying Exam Report, USC, February, 1998.

[9] N. Fenton, and S.L. Pfleeger, *Software metrics: A Rigorous and Practical Approach*, International Computer, Thomson Press, 1997.

[10] Gulezian R., 'Reformulating and Calibrating COCOMO' Journal Systems Software, Vol. 16, 1991, pp. 235-242.

[11] A. Idri, L. Kjiri, and A. Abran, "COCOMO Cost Model Using Fuzzy Logic", *7th International Conference on Fuzzy Theory & Technology*, Atlantic City, NJ, February, 2000. pp. 219-223.

[12] A. Idri, and A. Abran, "Towards A Fuzzy Logic Based Measures For Software Project Similarity", *Sixth Maghrebian Conference on Computer Sciences, Fes*, Morroco, November, 2000. pp. 9-18.

[13] A. Idri, and A. Abran, "A Fuzzy Logic Based Measures For Software Project Similarity: Validation and Possible Improvements", 7th International Symposium on Software Metrics ,*IEEE computer society*, 4-6 April, England, 2001. pp. 85-96.

[14] A. Idri, and A. Abran, "Evaluating Software Project Similarity by using Linguistic Quantifier Guided Aggregations", 9th IFSA World Congress/20th NAFIPS International Conference, 25-28 July, Vancouver, 2001.

[15] IFPUG, "Function Point Counting Practices Manual", Release 4.0, *International Function Point Users Group – IFPUG*, Westerville, Ohio, 1994.

[16] R. Jager, "Fuzzy Logic in Control", Ph.D. Thesis, Technic University Delft, Holland, 1995.

[17] R. Jeffery, M. Ruhe, I. Wieczorek, "Using Public Domain Metrics to Estimate Software Development Effort", 7th International Symposium on Software Metrics ,*IEEE computer society*, 4-6 April, London, 2001, pp. 16-27.

[18] B. Kitchenham, S. Linkman, "Estimates, uncertainty and risks", IEEE Software, Vol. 14, no 3, 1997, pp. 69-74.

[19] G. Kadoda. M. Cartwright, L. Chen, M. Shepperd, "Experiences Using Case-Based Reasoning to Predict Software Project Effort", EASE, Keele, UK, 2000, p. 23.

[20] J.L. Kolodner, *Case-Based Reasoning*, Morgan Kaufmann, 1993.

[21] Matson J., E, Barrett B., E, Mellichamp J., M, 'Software Development Cost Estimation Using Function Points', IEEE, Vol. 20, no 4, Apr., 1994, pp. 275-287.

[22] I. Myrtveit, E. Stensrud, "A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models", IEEE Transaction on Software Engineering, Vol. 25, no 4, July/August, 1999, pp. 510-525.

[23] F. Niessink, H. Van Vliet "Predicting Maintenance Effort with Function Points", in Proc Inter. Conf. on Soft. Maintenance, Bari, Italy, IEEE Computer Society, 1997.

[24] Putnam L. H, ' A General Empirical Solution to the Macro Software Sizing and Estimation Problem', IEEE Transactions on Soft. Eng., Vol. SE-4, no 4, July, 1978.

[25] M. Shepperd, C. Schofield, and B. Kitchenham, "Effort Estimation using Analogy", *ICSE-18*, Berlin, 1996, pp. 170-178.

[26] M. Shepperd, and C. Schofield, "Estimating Software Project Effort Using Analogies", *IEEE Trans. on Software Engineering*, Vol. 23, no. 12, November, 1997, pp. 736-743.

[27] S. Shepperd, G. Kadoda, "Using simulation to evaluate predictions systems", 7th International Symposium on Software Metrics ,*IEEE computer society*,4-6 April, England, 2001. pp. 349-358.

[28] S. Vicinanza, and M.J. Prietolla, "Case Based Reasoning in Software Effort Estimation", *Proceedings 11th Int. Conf. on Information Systems*, 1990.

[29] R.R. Yager, and J. Kacprzyk, *The Ordered Weighted Averaging Operators: Theory and Applications"* Kluwer: Norwell, MA, 1997.

[30] R.R. Yager, "Quantifier Guided Aggregation using OWA Operators", *International Journal of Intelligent Systems*, 11, 1996, pp.49-73.

[31] R.R. Yager, "Fuzzy Constraint Satisfaction for E-commerce Agents", *7th International Conference on Fuzzy Theory & Technology*, Atlantic City, NJ, February, 2000. pp. 111-114.

[32] L.A. Zadeh, "Fuzzy Set", *Information and Control*, Vol. 8, 1965, pp. 338-353.

[33] L.A. Zadeh, "A computational approach to fuzzy quantifiers in natural languages", *Computing and Mathematics with Applications*, 9, 1983, pp. 149-184.

[34] L.A. Zadeh, "Fuzzy Logic, Neural Networks, and Soft Computing", Comm. ACM, Vol. 37, no 3, March, 1994, pp.77-84.