Eng Int Syst (2002) 3: 147–157

Engineering
Intelligent Systems

Investigating soft computing in case-based reasoning for software cost estimation

Ali Idri*, Taghi M Khoshgoftaar† and Alain Abran‡

*ENSIAS, BP. 713, Agdal, Rabat, University Mohamed V, Morocco. Email: idri@ensias.ma
†Empirical Software Engineering Laboratory, Florida Atlantic University, Boca Raton, FL, USA. Email: taghi@cse.fau.edu
‡Ecole de technologie Supérieure, 1100, Notre-dame Ouest, Montréal, Canada. Email: aabran@ele.etsmtl.ca

Software cost estimation has been the subject of intensive investigations in the field of software engineering. As a result, numerous software cost estimation techniques have been proposed and investigated. To our knowledge, currently there are no cost estimation techniques that can incorporate and/or tolerate the aspects of imprecision, vagueness, and uncertainty into their predictions. However, software projects are often described by vague information. Furthermore, an estimate is only a probabilistic assessment of a future condition. Consequently, cost estimation models must be able to deal with imprecision and uncertainty, the two principal components of soft computing. To estimate the cost of software projects when they are described by vague and imprecise attributes, in an earlier study we have proposed an innovative approach referred to as Fuzzy Analogy. In this paper, we investigate the uncertainty of cost estimates generated by the Fuzzy Analogy approach. The primary aim is to generate a set of possible values for the actual software development cost. This set can then be used to deduce, for practical purposes, a point estimate for the cost, and for analyzing the risks associated with all possible estimates.

Keywords: case-based reasoning, soft computing, software cost estimation

1. INTRODUCTION

Estimation models in software engineering are used to predict some important attributes of future entities such as software development effort, software reliability, and productivity of programmers. Among these models, the ones that estimate software effort have motivated considerable research in recent years. The prediction procedure used by these software-effort models can be based on a mathematical function, such as $Effort = \alpha \times size^{\beta}$ or other techniques such as analogy-based reasoning, neural networks, regression trees, and rule induction models. Estimation by analogy is one of the most attractive techniques in the software effort estimation field, and is essentially a form of Case-Based Reasoning with four steps [1]:

1. Retrieve the most similar case(s).
2. Reuse the information and knowledge in the selected case(s) to solve the problem.
3. Revise the proposed solution.
4. Retain the parts of this experience likely to be useful for future problem solving.

For effort estimation, CBR is based on the following assumption: *similar software projects have similar costs*, and it has been deployed as follows. First, each project must be described by a set of attributes that must be relevant and independent. Second, we must determine the similarity between the candidate project and each project in the historical database. In the third step, known as case adaptation, the known effort values from the historical

projects are used to derive an estimate for the new project.

Recently, many researchers have begun to turn their attention to the CBR-based effort estimation alternative [2,5,14,21,23,24,28]. Consequently, many experiments have been conducted to compare estimation accuracy of the case-based reasoning approach with that of other modelling techniques. The obtained results have shown that analogy-based estimation does not out-perform other cost estimation techniques in every situation. Indeed, Shepperd *et al.*, Niessink, and Van Vliet found that estimation by analogy generated better results than stepwise regression [22, 23,24]. However, Briand *et al.* and Myrtveit and Stensrud reported contradicting results [5, 21]. Recent research has been initiated to explain the relationship between different properties of data sets (such as size, number of attributes and presence of outliers) of historical projects and the accuracy of a prediction system [25].

There are two main advantages of analogy-based estimation: first, it is easy to understand and to explain its process to the users, and second, it can model a complex set of relationships between the dependent variable (such as, cost or effort) and the independent variables (cost drivers). However, its deployment in software cost estimation still warrants some improvements. Human intelligence comes in large part from the ability to reason by analogy. But human reasoning by analogy is always approximate rather than precise. Indeed, the human mind can handle imprecision, uncertainty, partial truth, and approximation to achieve tractability, robustness, and low solution cost. According to Zadeh, "the exploitation of these criteria underlies the remarkable human ability to understand distorted speech, decipher sloppy handwriting, drive a vehicle in dense traffic and, more generally, make decisions in an environment of uncertainty and imprecision" [33,34,35].

Zadeh's idea is to mimic the ability of the human mind in order to increase the Machine Intelligence Quotient (MIQ) of the new industrial products (microwave, washing machines, software, etc.). Thus, for more than 30 years, Zadeh has been involved in the foundation of a new strategy of computing that is different from the traditional (hard) computing. Zadeh's effort in this area begins with his paper on fuzzy sets followed by the paper on possibility theory, the paper on soft computing, and more recently his papers on computing with words [31,33,34,35,36]. Our work concerns reasoning by analogy to which we integrate the concept of soft computing in order to deal with uncertainty and imprecision (like the human brain) in the analogy process. Specifically, we study the use of reasoning by analogy and soft computing in cost estimation for software projects.

Currently, no cost estimation model has integrated in its modelling process, the three components of soft computing (see Section 2.1). However, cost estimation models must be able to deal with vague information. Indeed, most of the software project attributes are measured on a scale composed of linguistic values such as *low* and *high*. For example, the well-known COCOMO'81 model has 15 attributes out of 17 (22 out of 24 in the COCOMO II) that are measured with six linguistics values: *very low, low, nominal, high, very high, and extra-high* [3,4,6]. Cost estimation models must learn from previous situations because software development technology is continuously evolving. Cost estimation models must appropriately handle the uncertainty in estimates. To deal with vague information, we have proposed an innovative approach, referred to as Fuzzy Analogy for software cost estimation. The purpose of this paper is to improve our approach by integrating the uncertainty criterion of soft computing. Managing the uncertainty in Fuzzy Analogy means that it can produce a set of estimates, rather than only one, with a possibility distribution. Thus, software projects managers in an organization may use this cost possibility distribution for risk assessment of the estimation results.

The remaining of this paper is organized as follows: In Section 2, we briefly present the concepts of fuzzy sets and soft computing. In Section 3, we present our Fuzzy Analogy approach that deals with vagueness and imprecision when describing software projects. In Section 4, we discuss two sources of uncertainty when estimating software project cost by Fuzzy Analogy. These two sources are related to the measurement error and the accuracy of our approach. To deal with uncertainty in Fuzzy Analogy, we present in Section 5, a strategy based on the assumption that a CBR affirmation is non-deterministic in software cost estimation. In Section 6, we summarise our findings and suggest future work in this area.

2. FUZZY SETS AND SOFT COMPUTING

Zadeh is known as the founding father of fuzzy sets, soft computing, and more recently computing with words. The aim of these three components is to solve complex problems that have not been solved (yet) by any theory. However, such problems become very simple when the human brain tackles them. Zadeh has noticed that the key of this remarkable human ability is the human brain's crucial ability to manipulate perceptions rather than measures. The fundamental difference between perceptions and measures is that, in general, perceptions are fuzzy granules, whereas measures are crisp numbers. The goal of this section is not to discuss fuzzy sets in-depth, soft computing, and computing with words, but rather to explain those parts for a better understanding in the context of this work, especially fuzzy sets and soft computing.

2.1 Fuzzy sets

A fuzzy set, introduced by Zadeh in 1965, is a set with a graded membership function, μ, in the real interval [0,1]. This definition extends the one for a classical set, in which the membership function is in the couple {0,1}. Fuzzy sets can be used to represent linguistic values such as *low, young*, and *complex*. The advantage of this representation is that it mimics the way in which humans interpret these linguistic values. However, fuzzy sets can be used in the following two ways [15]:

- Fuzzy sets that model the gradual nature of properties, i.e., the higher the membership x has in a fuzzy set A, the more it is true that x is A. Here the fuzzy set is used to model the vagueness of the linguistic value represented by the fuzzy set A.
- Fuzzy sets that represent incomplete states of knowledge. Here the fuzzy set is a possibility distribution of the variable X, and consequently, it is used to model the uncertainty. When considering that it is only known that x is A, and x is not precisely known, the fuzzy set A can be considered as a possibility distribution, i.e., the higher the membership x' has in A, the higher the possibility that x = x'.

2.2 Soft computing

In 1994, Zadeh distinguished soft computing from traditional (hard) computing in that, unlike hard computing, the point of departure in soft computing is the hypothesis that precision and certainty carry a cost and that computation, reasoning, and decision-making should exploit (whenever possible) the tolerance of imprecision and uncertainty. Indeed, the role model of soft computing is the human mind. Zadeh's example concerning the problem of parking an automobile illustrates clearly the advantages of the use of soft rather than traditional computing. According to Zadeh, the principal constituents of soft computing are [34]:

- fuzzy logic (FL) which is primarily concerned with imprecision,
- neural networks (NN) for learning, and
- probabilistic reasoning (PR) to account for uncertainty, ex., Bayesian belief networks.

According to Zadeh, it important to note is that although there are substantial overlapping areas between FL, NN and PR, in general they are complementary rather than competing. A striking example of a particularly effective combination is what has become to be known as 'neuro-fuzzy' systems.

3. COST ESTIMATION BY FUZZY ANALOGY

It is well recognized that estimation by analogy is a promising technique for software cost and effort estimation. However, until now its use in cost estimation does not appropriately handle (unlike the human brain) vague information. But, most software project attributes, such as experience of programmers, complexity of modules, and software reliability are measured on an ordinal or nominal scale composed of linguistic values such as *low* and *high*. Indeed, most investigations of analogy in cost estimation represent linguistic values by crisp rather than fuzzy sets [2,5,17,21,22,23,24]. To overcome this limitation in order to explore the benefits of using overlapping fuzzy set membership functions, we have developed a new approach referred to as Fuzzy Analogy [13]. We have validated this approach with the COCOMO'81 data set. The results obtained were compared with three other models. Fuzzy Analogy performs better in terms of accuracy and adequacy in dealing with linguistic values [14]. In the following sections, we present a summary of our approach.

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 cases adaptation. Each step is a fuzzification of its equivalent in the classical analogy procedure. In the following sub-sections, each step will be further detailed.

3.1 Identification of cases

The goal of this step is the characterization of all software projects by a set of attributes. Selecting attributes that accurately describe software projects is a complex task, and in machine-learning literature, it is known as the Feature Subset Selection (FSS) problem [7,19,26]. It is known that feature subset selection can enhance the performance of CBR systems [1]. However, few studies have investigated this issue in cost estimation by analogy [18]. Currently, in our F_ANGEL[1] prototype, we do not yet provide intelligent solution for FSS problem; however, we adopt a simple solution by allowing the estimators to use the attributes that they believe best characterize their projects and are more appropriate in their environment. Further research work has been initiated to develop an appropriate technique for FSS problem in Fuzzy Analogy.

The objective of our Fuzzy Analogy approach is to deal with linguistic values. In the identification step, each software project is described by a set of selected attributes that can be measured by numerical or linguistic values. Let us assume that for linguistic values, 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 ($\mu_{A_k^j}$). It is preferable that these fuzzy sets satisfy the *normal condition*, i.e. they form a fuzzy partition and each of them is convex and normal [11]. The use of fuzzy sets to represent linguistic values allows us to deal with vagueness, imprecision and uncertainty in the cases identification step. Another advantage of our Fuzzy Analogy approach is that it takes into account, the importance of each selected attribute in the 'identification of cases' step. It is obvious that all the selected attributes do not necessarily have the same influence on the software project effort. Hence, we are required to indicate the weights, $\mu_{k'}$ associated with all the selected attributes in the cases identification step.

3.2 Retrieval of similar cases

This step is based on the choice of a software project

[1] F_ANGEL is a software prototype that we have developed to automate the fuzzy analogy approach. It can be seen as a fuzzification of the software ANGEL developed by Shepperd *et al.*

similarity measure. This choice is very important since it will influence which analogies are found. In the cost estimation literature, most researchers have used the Euclidean distance when projects are described by numerical data and the Equality distance when they are described by linguistic values [23,24]. In the fuzzy logic literature, a candidate similarity measure is a function, whose values are in [0,1] and must satisfy at least two criteria: it must be reflexive and symmetric [8,32]. However, the concept of similarity has also been discussed in other disciplines of cognitive science (Philosophy, Psychology, Artificial intelligence, etc.) [20,27]. In our case, we are looking for similarity measures for software projects. These measures must be able to evaluate the similarity between projects when they are described by linguistic values. Consequently, we have proposed a set of candidate measures for software project similarity [10]. These measures evaluate the overall similarity, $d(P_1, P_2)$, of two projects P_1 and 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 [11]:

$$d_{v_j}(P_1, P_2) = \begin{cases} \max_k \min(\mu_{A_k^j}(P_1), \mu_{A_k^j}(P_2)) \\ \textit{max – min aggregation} \end{cases} \quad (1.1)$$

$$\begin{cases} \sum \mu_{A_k^j}(P_1) \times \mu_{A_k^j}(P_2) \\ \textit{sum – product aggregation} \end{cases} \quad (1.2)$$

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 $\mu_{A_k^j}$ are the membership functions representing fuzzy sets A_k^j. In this axiomatic validation, we have considered four axioms. The first two are about reflexivity and symmetry. Our 'fuzzy' reflexivity is defined by $d(u,v) \leq d(u,u)$, and not by $d(u,u) = 1$ as in the literature. This latter definition is very close to classical logic, whereas 'classical' reflexivity implies 'fuzzy' reflexivity, however the opposite is not true.

To evaluate the overall distance of P_1 and P_2 the individual distances $d_{v_j}(P_1, P_2)$ are aggregated using Regular Increasing Monotone (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 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 [12]:

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

The use of the RIM quantifiers is very interesting, because it allows us to satisfy Tversky's affirmation that considers the similarity attribute as dependent on the context [27]. Consequently, the similarity between two projects depends on the environment in which it is evaluated.

After choosing the appropriate RIM linguistic quantifier to guide the aggregation of the individual distances, its implementation is realised by an Ordered Weight Averaging operator [29,30]. The overall distance $d(P_1, P_2)$ is calculated by means of the following formula:

$$\left.\begin{aligned} d(P_1, P_2) &= \sum_{j=1}^{M} w_j \times d_{v_j}(P_1, P_2) \\ w_j &= Q\left(\frac{\sum_{k=1}^{j} u_k}{T}\right) - Q\left(\frac{\sum_{k=1}^{j-1} u_k}{T}\right) \end{aligned}\right\} \quad (2)$$

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 that are provided in the 'identification of cases' (Section 3.1) step. Here we stress the meaning of u_k and w_j. Tversky defines the saliency of a feature in terms of its intensity and its diagnosticity. In our approach, u_k represents the intensity, while w_j represents diagnosticity.

3.3 Cases adaptation

The objective of this step is to derive an estimate for the new project by using the known effort values of similar projects. In this step, two issues must be addressed. First, the choice of how many similar projects should be used in the adaptation, and second, how to adapt the chosen analogies to generate an estimate for the new project. In the literature, one can notice that there is no clear rule to guide the choice of the number of analogies, K. The results obtained from many experimentations seemed to favour the case where K is lower than 3 [2,17,23,24]. We are not convinced by the approach of fixing the number of analogies to be considered in the case adaptation step. The principle of this approach is to take only the first K projects that are similar to the new project. The choice of the number K relies on the use of the classical logic principle: the transition from one situation (contribution in the estimated cost) to the following (no contribution in the estimated cost) is abrupt rather than gradual.

In Fuzzy Analogy, we have proposed a new strategy to select projects to be used in the adaptation step. This strategy is based on the distances $d(P, P_i)$ and the definition adopted in the studied environment for the proposition, 'P_i is a closely similar project to P'. Intuitively, P_i is closely similar to P if $d(P, P_i)$ is in the vicinity of 1. The only way to represent correctly the value *vicinity of 1* is by using a fuzzy set defined in the unit interval [0, 1]. Indeed, this fuzzy set defines the *closely similar* qualification adopted in the environment. Figure 1 shows a possible representation for the value *vicinity of 1*. In this example all projects that have $d(P, P_i)$ higher than 0.5 contribute to the estimated cost of P: the contribution of each P_i is weighted by $\mu_{\textit{vicinity of 1}}(d(P, P_i))$.

The second issue in this step is to adapt the chosen analogies in order to generate an estimate for the new project. The most common solutions use the (weighted) mean or the median of the K chosen analogies. In the case of weighted mean, 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 data set. The weights are the values of the membership function defining the fuzzy set *vicinity of 1*. The formula is then given by:

$$Effort(P) = \frac{\sum_{i=1}^{N} \mu_{vicinity\ of\ 1}(d(P, P_i)) \times Effort(P_i)}{\sum_{i=1}^{N} \mu_{vicinity\ of\ 1}(d(P, P_i))} \quad (3)$$

The main advantage of our adaptation approach is that it can be easily configured by defining the value *vicinity of 1* according to the needs of each environment.

Until now, Fuzzy Analogy dealt with vague information. However, it is not able to handle uncertainty and risks inherent in estimates. Indeed, in its case adaptation step, it provides only one numerical estimate. Relying on a single estimate may easily lead to wrong managerial decisions. In the following section, we present a new strategy that can be used in the adaptation step in order to let Fuzzy Analogy generate a set of possible values for the real cost.

4. SOURCES OF UNCERTAINTY IN FUZZY ANALOGY

In order to satisfy the third criterion of Soft Computing, Fuzzy Analogy must be able to handle and incorporate the uncertainty factor when estimating the cost or development effort of the new project. The need of managing uncertainty in Fuzzy Analogy is justified by three reasons:

- An estimate is a probabilistic assessment of a future condition [16].
- We cannot include all the factors that affect the cost in the identification step of Fuzzy Analogy. In practice, we consider only the attributes for which data in the studied environment is available. Hence, the factors that affect the cost, which are not included explicitly in our model contribute to the uncertainty in the predicted cost.

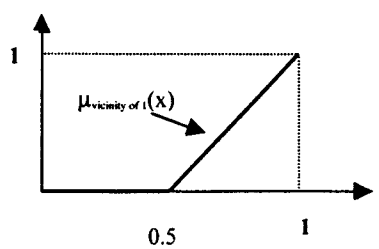

Figure 1 A possible definition of the value vicinity of 1

- Software projects managers prefer cost estimation models to provide a set of possible values rather than a unique value of the actual cost. A unique estimate may easily lead to wrong managerial decisions and project failure.

Managing the uncertainty in Fuzzy Analogy means that it can produce a set of estimates with a possibility distribution. Thus, the software projects managers in an organization may use this cost possibility distribution for risk assessment of the estimation results. Kitchenham and Linkman have examined four sources of estimate uncertainty and risk: measurement error, model error, assumption error, and scope error [16]. The two first sources are related respectively to the accuracy of the measurements of the model's input and the accuracy of the model itself. The other two sources are concerned with the assumptions that we can make about a model's input parameters, which are not known when the estimate is required. These inputs must not be considered in the model in order to avoid double-counting the effects of uncertainty. These assumptions will be used to manage project risks. According to Kitchenham and Linkman there is no way to manage uncertainties and risks for a single one-off project. The only way to do that is to manage risks across an organization's entire project portfolio, as it is the case for insurance companies. We found that it is an interesting issue that must be explored in software cost estimation. However, in this work we only deal with the two first sources of uncertainty in the Fuzzy Analogy approach.

4.1 Measurement error

In Fuzzy Analogy, the software project attributes are measured either by numerical or linguistic values. The use of numerical values expresses that there is no uncertainty and no imprecision when evaluating the attribute. However, in software engineering most of the software project attributes are qualitative rather than quantitative. The overwhelming number of linguistic attributes reflects that software engineering is a 'young' science and still needs further maturing. Moreover, in the software measurement process, humans are directly involved and consequently the results of measurements may be highly influenced by their own judgement. To deal with uncertainty and imprecision when evaluating software project attributes, we suggest that software measurement values should be linguistic rather than numeric. According to Zadeh, the use of linguistic values instead (or in addition) of numbers serves many major purposes, such as [35]:

- Compared to numbers, linguistic values are easy to understand,
- Allowance for imprecision,
- Linguistic values generalize numbers. Indeed, numbers are only used when we have precise information,
- Acceptance of the finite ability of the human mind to resolve details and store precise information.

Here the most important advantage of using linguistic

rather than numeric values is to reduce the effect of measurement errors on the estimates generated by Fuzzy Analogy. When using numerical values, measurement error is evaluated by the difference between the given and the actual values, whereas when using linguistic values, it is evaluated by the difference between the given and the actual membership degrees to these linguistic values. Consequently, the effect of measurement error will be most likely higher when software project attributes are measured by numerical data than that when they are measured by linguistic values. Indeed, there are an infinite number of possible numeric values whereas there are only a small number of linguistic values to evaluate one attribute. For example, let us consider that the attribute, 'experience of programmer', is measured by the number of years, which is a numerical value, and suppose also that we can assign to each programmer one of the three qualifications (*low, average, high*) according to its experience (Figure 2).

There are many situations in which, although we can have imprecision in the evaluation of the 'experience of programmer' attribute, it will not have any effect on the cost estimated by Fuzzy Analogy. For example, let us consider that the programmer P_1, who is in position (*), must be in position (+) (see Figure 2). P_1 belongs in both cases to the 'low' qualification with degree equal to 1. Consequently, this measurement error has no effect on the estimate. However, if we use the number of years instead of linguistic values, it will affect the estimated cost. Also, let us consider that the programmer P_3, who is in position (*), must be in position (+) (Figure 2). In this case P_3 belongs to the 'average' and 'high' qualifications with different membership degrees. When evaluating individual similarity between P_3 and P_2 by the max-min aggregation (Equation 1.1) we obtain in both cases (P_3 is in position (*) or (+)) the same value. Consequently, there is no effect of this measurement error on the estimated cost.

The use of linguistic quantifiers to combine individual similarities may also avoid the effect of measurement error in the estimate. Let us consider that we have chosen the linguistic quantifier 'at least four' to combine the individual similarities, and all project attributes have the same weights u_k. In this case, the overall similarity between two projects is exactly the fourth largest individual similarity between these two projects. Consequently, if the attribute, for which we have a measurement error, has the associated individual distance lower than the fourth largest individual distance, then this measurement error has no effect on the estimate. Although these examples show that the effects of measurement error may be avoided in our approach, we recognize that there are some situations in which measurement error can modify the estimates generated by Fuzzy Analogy. For example, the individual similarity between P_1 and P_4 (P_4 is in position (*)) is different than that of between P_1 and P_4 (P_4 is in position (+)). As consequence, if the linguistic quantifier used in the evaluation of the overall similarity cannot avoid this error, it will affect the estimate made using Fuzzy Analogy

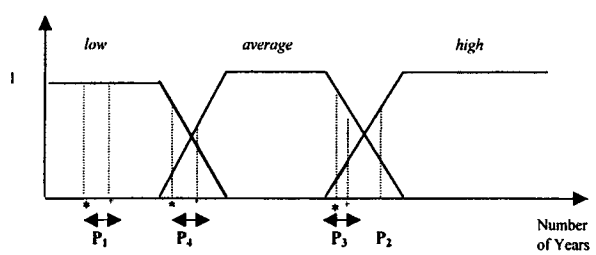

Figure 2 Examples of situations where measurement error can (or not) affect the estimate generated by Fuzzy Analogy

In the end of this section, we conclude that using linguistic values to describe software projects and linguistic quantifiers to combine individual distances may avoid or reduce the effect of measurement errors. Consequently, we do not consider this source of uncertainty in our model. We focus, in the next section, on the second source of uncertainty that is related to the accuracy of Fuzzy Analogy.

4.2 Model error

This type of error is related to the accuracy of the model used in the cost estimation task. The accuracy of a model depends on three parameters:

- The approach adopted by the model (linear regression, Neural networks, CBR, etc.)
- The characteristics of data set from which the model has been developed (size, number of attributes, presence of outliers, etc.)
- The characteristics of data set on which the model is validated

The Mean Magnitude of the Relative Error (MMRE) is the most used indicator to evaluate the accuracy of cost estimation models. For example, if we have a model with MMRE equal to 25% and the estimated cost for one project by this model is equal to 300 person-days, our estimate is likely to be between 225 and 375 person-days. When we validated the Fuzzy Analogy approach on the COCOMO'81 data set, it is observed that the accuracy of our approach is monotonously increasing according to the RIM linguistic quantifiers used in the evaluation of the overall similarity between two projects. A higher accuracy was obtained when using the 'all' linguistic quantifier (MMRE = 21). However, as Kitchenham and Linkman have noted, the MMRE assesses underestimates and overestimates to be of equal size. They point out that an underestimate is bounded naturally: the minimum effort required by an activity is zero, and in contrast, an overestimate is not bounded. Consequently, they proposed the use of a probability distribution like Gamma distribution for example.

We are convinced by the idea discussed above, but it still needs two improvements. First, the probability distribution must be developed and tailored according to the characteristics of each organization. As a consequence, different organizations can use different probability distributions. Second, an overestimate may also be bounded by the maximum budget that an organization can provide for

its software projects. In the following paragraphs, we discuss how our approach may generate a cost distribution for each software project by using the possibility theory.

The Fuzzy Analogy approach is based on the well-known CBR affirmation 'similar projects have similar costs'. We can notice that there are two sources of uncertainty in this affirmation. First, the consequence of this affirmation is imprecise. Second, it seems that this affirmation is not always deterministic in software cost estimation. Thus, we may find two software projects that are similar according to a predefined similarity measure but their costs are completely different. Dubois *et al.* have proposed two approaches to deal with these two situations depending on whether the CBR affirmation is deterministic or non-deterministic [8]. Here, the most important question that we must highlight is, 'Is CBR affirmation deterministic or non-deterministic in cost estimation?' Intuitively, we can say that the CBR affirmation is deterministic in cost estimation. There is no reason for two similar projects not to have similar costs. However, in practice, there are two reasons that can lead to the opposite situation:

- Selecting the attributes that describe accurately software projects is a complex task. Often, practitioners use only attributes for which data is available.
- The projects similarity measures (^{d}) and the cost similarity measure (C) may be invalid, i.e., they are not measuring what they claim to measure. Indeed, validation of software measures is a crucial step and there is no common definition regarding what constitutes a valid measure.

To check whether the CBR affirmation is deterministic or not, we have conducted an experiment based on the COCOMO'81 data set. CBR affirmation is deterministic in the COCOMO'81 data set implies that for each couple (P_i, P_j) of software projects, we must have $d(P_i, P_j)$ lower or equal to $C(c_i, c_j)$ where c_i and c_j are respectively, the costs of P_i and P_j. This implies that the truth-value of the proposition 'c_i and c_j are similar', must be at least equal to the truth-value of the proposition 'P_i and P_j are similar'. The COCOMO'81 data set contains 63 software projects. Each project is described by 17 attributes. The first two are the software size, a unique numerical attribute measured in KDSI (Kilo Delivered Source Instructions) and the project mode, defined as either 'organic', 'semi-detached' or 'embedded'. The remaining 15 attributes are measured on a scale composed of six linguistic values: *very low, low, nominal, high, very high,* and *extra-high*. Among these 17 attributes, we have retained 12 attributes that we had already represented by fuzzy sets [9]. The other attributes are not studied because their descriptions proved insufficient.

For the individual project similarity measure, we choose the max-min aggregation (Equation 1.1) because it satisfies all established axioms [11]. The overall project similarity measure depends on the RIM linguistic quantifier used to combine the individual similarities. We use various α-RIM quantifiers. An α-RIM is defined by a fuzzy set in the unit interval with membership function $Q(x) = x^{\alpha}$. The project cost is a numerical data and each organization, according to its financial politics, can define the set of similar values to a given cost. Below, we give two examples of cost similarity measures. Figure 3(a) shows a cost similarity measure, C_R, that considers all values of the interval $[C_0 - pC_0, C_0 + pC_0]$ as similar to the cost C_0, 'p' is a given percentage. Figure 3(b) shows a cost similarity measure, C_A, that considers all values of the interval $[C_0 - cst, C_0 + cst]$ as similar to the cost C0, 'cst' is a predetermined constant. The C_R measure, on the contrary of C_A, takes into account the cost C_0 to define the interval of its similar values and consequently the length of this interval depends on C_0. Also, the C_R cost similarity measure is asymmetric whereas C_A is symmetric.

According to Dubois *et al*, the symmetry property is required for any similarity measure whereas Tversky suggested that it depends on the context [8,27]. We agree with Tversky's suggestion. For example, in our case software projects similarity measure must be symmetric, but cost similarity measure may be asymmetric. Indeed, we can consider, for business purposes, that \$15K (\$1K = \$1000) is more similar to \$20K than \$20K is to \$15K. The C_R measure respects this empirical relation and it seems to be useful for cost similarity. Also, the C_R measure is equivalent to the classical principle used in the validation of cost estimation models. Indeed, the evaluation of the accuracy of cost estimation models is based on the affirmation: 'An estimate is accurate if its MRE is lower than or equal to 25'. This affirmation is equivalent to the following: 'An estimate is accurate if it is C_R similar to the actual cost', with 'p' is equal to 25.

Table 1 shows the results obtained when evaluating the CBR affirmation on the COCOMO'81 data set. In this evaluation, we have used various linguistic quantifiers for the project similarity measure (α-RIM column) and the two cost similarity measures: C_R and C_A. For each couple of software project similarity measure and cost similarity measure, we have calculated the following quantities:

- NB_CBR_ND: The number of cases where the CBR affirmation is non-deterministic ($d(P_i, P_j) > C(c_i, c_j)$). In such situations, E_{ij} denotes the difference between $d(P_i, P_j)$ and $C(c_i, c_j)$, i.e. $E_{ij} = d(P_i, P_j) - C(c_i, c_j)$.
- The min, max, and average of E_{ij}

By analyzing the results of the evaluation of the CBR affirmation (Table 2), we noticed that the number NB_CBR_ND depends on similarity measures used both

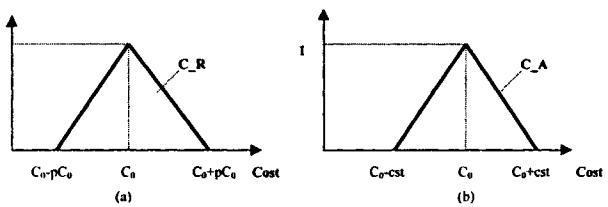

Figure 3 Two examples of cost similarity measures: (a) C_R measure, (b) C_A measure

Table 1 Results of the evaluation of the CBR affirmation on COCOMO'81 data set

	C_A measure: cts=6 (6 is the lower cost in COCOMO'81data set)				*C_R measure: P=25*			
α-RIM	***NB_CBR_ND***	***Min E_{ij}***	***Max E_{ij}***	***Average E_{ij}***	***NB_CBR_ND***	***Min E_{ij}***	***Max E_{ij}***	***Average E_{ij}***
Max	3899	0.1666	1	0.9825	3899	0.0446	1	0.9652
1/10	3899	0.0839	0.9956	0.9088	3897	0.0083	0.9924	0.9133
1/7	3899	0.0507	0.9937	0.8813	3889	0.0039	0.9893	0.8938
1/5	3899	0.0826	0.9913	0.8813	3887	0.0062	0.9851	0.8668
1/3	3887	0.0003	0.9858	0.8239	3875	0.0043	0.9755	0.8092
1	3857	0.0041	0.9606	0.5953	3799	0.0085	0.9313	0.5887
10	3769	2.5E-08	0.8109	0.8109	3634	2.5E-08	0.6526	0.0401
20	3767	6.23E-16	0.7633	0.0120	3625	6.23E-16	0.5889	0.0114
100	3765	9.41E-77	0.7276	0.0047	3623	9.41E-77	0.5769	0.0042
500	3755	8.3E-229	0.7276	0.0047	3613	2.6E-305	0.5769	0.0042
1000	3487	5.33E-310	0.7276	0.0050	3355	5.33E-310	0.5769	0.0045
2000	2155	2.25E-310	0.7276	0.0082	2080	2.25E-310	0.5769	0.0073
3000	1263	1.08E-307	0.7276	0.0140	1219	1.08E-307	0.5769	0.0125
4000	977	2.65E-309	0.7276	0.0181	930	2.65E-309	0.5769	0.0164
5000	433	3.96E-266	0.7276	0.0408	415	3.96E-266	0.5769	0.0367
8000	401	1.66E-299	0.7276	0.0740	229	1.66E-299	0.5769	0.0666
10000	239	3.73E-310	0.7276	0.1022	166	3.73E-310	0.5769	0.0918
20000	77	0.0121	0.7276	0.2297	72	0.0121	0.5769	0.2118
Min	77	0.0121	0.7276	0.2297	72	0.0121	0.5769	0.2118

for the projects and the costs. So, if we consider the number NB_CBR_ND as a function of α, we can notice that, for both cost measures C_A and C_R, it is a monotonously increasing function according to α. This is due to the fact that our project similarity measures are monotonously decreasing according to α ($d_\alpha(P_i, P_j) \geq d_{\alpha'}(P_i, P_j)$ $\alpha \leq \alpha'$) Indeed, when α tends towards zero, this implies that the overall similarity between two projects will take into account fewer attributes among those describing software projects. The minimum number of attributes to consider is one. This is the case when using the max operator where the selected attribute is the one for which the associated individual distance is the maximum of all individual distances. As a consequence, the overall similarity between two projects will be higher because we are more likely to find in the COCOMO'81 data set, at least one attribute for which the associated linguistic values are the same for the two projects. Thus, most likely, $d(P_i, P_j)$ will be higher than $C(c_i, c_j)$. By contrast, when α tends towards infinity it implies that the overall similarity between two projects will take into account many attributes among all the available ones describing the software projects. As a maximum we may consider all attributes. This is the case when combining using the min operator. Consequently, the overall similarity will be minor because we are more likely to find in the COCOMO'81 data set one attribute for which the associated linguistic values are different for the two projects. Thus, most likely $d(P_i, P_j)$ will be lower than $C(c_i, c_j)$.

Figure 4 shows the relation between α and the number NB_CBR_ND for the two cost similarity measures C_A and C_R. The min (max) aggregation gives lower (higher) NB_CBR_ND because it considers all (one) attributes in the evaluation of the similarity between projects. For the other α-RIM linguistic quantifiers ($0 < \alpha < \infty$), the number NB_CBR_ND decreases with α because additional attributes will be considered in the evaluation of the overall similarity between projects. For example, a software project P_i, which has an overall similarity with P_j higher than the similarity between their costs when α is equal to 100, may have the opposite relation when α is equal to 300. Indeed, we have $d_{10}(P_i, P_j) \geq d_{30}(P_i, P_j)$. Also, we can notice that the number NB_CBR_ND of the measure C_R is lower or equal to that of the measure C_A for all α-RIM quantifiers. Indeed, in our experiment, we have for each cost c_0 higher or equal to \$24K, $C\_R(c_0,c) \geq C\_A(c_0,c)$. Consequently, all the non-deterministic cases for C_R are also the same for C_A. If c_0 is lower than 24, we have the opposite situation. However, the number of projects that have the costs higher or equal to 24 represents 81% of the COCOMO'81 data set. As a consequence, most likely, the NB_CBR_ND of C_R will be lower to that of C_A.

If we consider the case of using the *all* linguistic quantifier for projects similarity ('Min' row in Table 1), we have only 77 cases for C_A, which represents 1.94% (72 for C_R, 1.81%) where the CBR affirmation is not deterministic. Consequently, one can consider that the CBR affirmation is deterministic for the COCOMO'81 data set. However, when we have used the deterministic approach

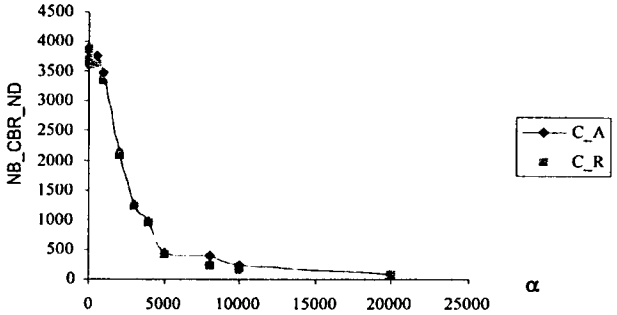

Figure 4 Relationship between α and NB_CBR_ND

in the case adaptation step of Fuzzy Analogy, we have found that it always leads to an empty set of solutions. Indeed, a new project can be similar to many projects in the data set but the costs of these projects are different. For example by using the jack-knife procedure, if we remove the project P_9 from the COCOMO'81 data set, we find that it is similar to the project P_{27} and P_{56} with degrees equal to 0.30 and 0.19, respectively. But, there is no numerical value that can be similar, according to C_R and C_A, to the cost of P_{27} ($88K) and the cost of P_{56} ($958K). To avoid this problem, Dubois *et al.* have suggested that the two similarity measures must be chosen in order to satisfy two properties. These properties guarantee that the set of solutions is not empty [8]. We are not convinced by their proposition because, in software cost estimation, it can lead to some contradictions between the obtained measures and our understanding of the similarity attribute. Consequently, we have chosen to use the non-deterministic approach in order to handle the uncertainty in the Fuzzy Analogy approach.

5. UNCERTAINTY IN ESTIMATES OF FUZZY ANALOGY

Basically, the non-deterministic CBR approach is based on the following affirmation: '*similar projects have possibly similar costs*'. Consequently, if we have a new project P that is similar to one project P_i of the historical data set, it is possible that their costs may be similar. Dubois *et al.* have proposed to represent this affirmation by a possibility rule of the form: 'the more similar P is to P_i, the more likely that the cost of P is similar to the cost of P_i' [8]. This rule expresses that it is possible, at least at the degree $d(P, P_j)$, that the cost of P is similar to the cost of P_i. Also, each value c_0 that is similar to the cost of P_i with degree different from 0 is possibly, at least at the degree $\min(d(P, P_j), C(c_0, \text{cost}(P_i))$, similar to the cost of P. In the following, we use only the measure C_R for the similarity of costs. The fuzzy set of possible values, c, for the cost of P with respect to project P_i is obtained by modeling the implication in the possibility rule of P and P_i with a conjunction (T-norm):

$$\pi_{cost(P_i)}(c) = \min(d(P, P_i), C\_R_{cost(P_i)}(c)) \quad (4)$$

If we have N projects in the historical data set, each project P_i will generate a new possible value for the cost of P according to Equation 4. These various contributions of projects P_i are combined by a max-based aggregation in order to obtain the fuzzy set, Cp of all possible values for the cost of P:

$$C_P(c) = \max_i \pi_{cost(P_i)}(c) \quad (5)$$

The function Cp represents the possibility distribution of the cost of P by considering all the projects in the historical data set. It will be used to evaluate the degree of uncertainty when estimating the cost of a new project. For example, we have removed the project P_{45} from the COCOMO'81 data set and have generated its cost possibility distribution by using Equations 4 and 5 (Figure 5). We have found that the project P_{45} is similar to the projects P_{42}, P_{43}, P_{44} and P_{46} with a degree equal to 0.38, 0.40, 0.38 and 0.36, respectively. The possibility distribution is defined only for the values that are similar to at least one of the four values representing the costs of P_{42}, P_{43}, P_{44} and P_{46}. For the other values, the possibility distribution is unknown. For practical purposes, we need a point estimate of the cost of P_{45}. There are many strategies that may be used to generate this estimate. These strategies are based on defuzzification methods such as the center-of-gravity, the mean-of-maxima, and the center-of-area. These strategies may not be necessarily applied to the whole fuzzy set representing the possibility distribution. It can also be applied only to some of its parts.

As discussed in Section 3.3, our 'case adaptation' step provides a single cost estimate. This approach will be equivalent to the fuzzy-mean defuzzification method if in Equations 4 and 5, we use the product and the summation instead of the min and the max operators. As we can notice, the cost possibility distribution of P_{45} suggests that most likely the actual cost of P_{45} is in the interval [66.4, 151.2]. Indeed, this interval defines the large part of the possibility distribution area. Consequently, if we apply the center-of-area or the center-of-gravity methods to the possibility distribution in this interval, we obtain an estimate equal to $100K. As per our single cost estimate approach in Section 3.3, an estimate equal to $68K is generated. The actual cost of P_{45} is $106K. Here we are not concerned by the accuracy of the estimates generated by defuzzification methods but rather by the degree of information provided by the cost possibility distribution for risk assessment. For the project P_{45}, if we adopt an optimistic estimate, which is

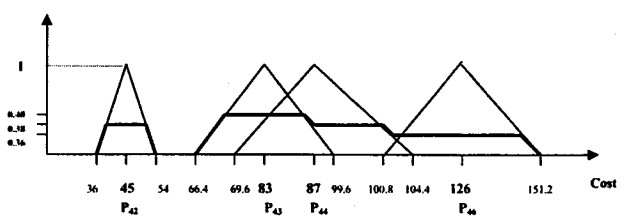

Figure 5 Example of possibility distribution for the project P_{45} of the COCOMO'81 data set

equal to \$45K, and P_{45} has required \$126K, we will need \$81K to achieve the project. Kitchenham and Linkman propose to use the principle adopted by insurance companies in such situation. In our case, we can assign to each project a contingency that depends on the risk and the possibility of this risk. For the project P_{45}, the contingency is equal to \$29.06K against the risk that P_{45} can cost \$126K.

As opposed to the case of project P_{45}, the obtained distribution possibility may be un-informative where one project is similar to many projects, which have different costs. This type of situation was not present in the COCOMO'81 data set. In order to reduce the possibility of its occurrence, an organization may evaluate the cost of its projects by fewer linguistic values such as *low, average* and *high*. Consequently, the fuzzy set of possible values, c, for the cost of P with respect to the project P_i will be:

$$\pi_{P_i}(c) = \max_k \min(d(P, P_i), \mu_{A_k}(c)) \quad (6)$$

where A_k are the fuzzy sets defined for the cost. Considering all projects P_i in the historical data set, the cost possibility distribution C_p of P, is obtained by a max-based aggregation of Equation 6. For example, let us consider a new project that is similar to P_1 and P_2 with a degree equal to 0.5 and 0.7, respectively. Although, the costs of P_1 and P_2 do not belong to the same qualification, there is no interval in which the cost possibility distribution for P is unknown (Figure 6).

6. DISCUSSION AND FUTURE IMPROVEMENTS

In this paper, we have improved the Fuzzy Analogy approach in order to handle the uncertainty in its development cost/effort estimates. We have discussed two sources of uncertainty in our approach. The first source is from measurement errors when evaluating the input variables for the Fuzzy Analogy approach. We have shown that the use of linguistic values rather than numerical data when describing software projects may reduce or avoid the effects of measurement errors in our approach. Indeed, in Fuzzy Analogy, we use the membership degrees in the evaluation of the similarity between projects. In addition, the use of linguistic quantifiers to evaluate the overall similarity between projects may reduce or avoid the effects of measurement errors. Consequently, we have ignored this source of uncertainty in our approach. The second source of uncertainty is related to the accuracy of the Fuzzy Analogy approach.

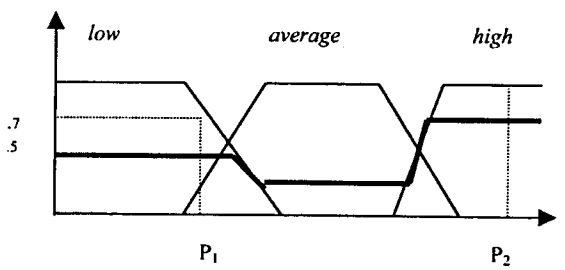

Figure 6 Use of linguistic values for cost can avoid intervals in which possibility distribution is unknown

We adopted the idea discussed by Kitchenham and Linkman, in that cost estimation models must be able to generate a set of values with a probability distribution for the estimated cost. To build the possibility distribution, we conducted an experiment using the COCOMO'81 data set in order to evaluate whether the CBR affirmation is deterministic or non-deterministic in the context of software cost estimation. Based on the experiment, we conclude that the deterministic approach is not useful in software cost estimation for three main reasons:

a) One cannot always accurately describe the software projects,
b) It is not easy to prove that the similarity measures for the projects and the cost are valid, and
c) The set of estimates generated by the deterministic approach may be empty although the new project is similar to many projects in the historical data set.

Consequently, we have chosen the non-deterministic approach to handle the uncertainty of cost estimates issue in Fuzzy Analogy. The non-deterministic approach uses possibility rules and max-based aggregation to generate the cost possibility distribution for a new project. This possibility distribution may be used to generate a point estimate and the risk of this estimate. Further research has been initiated to develop a new strategy based on our approach for risks assessment in software cost estimation.

By using fuzzy logic and the possibility theory in its estimation process, our approach satisfies two (out of three) criteria of the Soft Computing concept, i.e. the tolerance of imprecision when describing software project, and the uncertainty when estimating the development cost. The third criterion of soft computing that Fuzzy Analogy has not yet incorporated into its process is to learn from previous experiences. The learning criterion is required for any cost estimation model. Indeed, the software industry is continuously evolving: engineers use more and more high level programming languages, application generators, web-based technology, etc.

We have initiated the inclusion of some learning functionality in our approach. In the identification step, we can update all information concerning the linguistic variables describing software projects, specifically, their linguistic values that depend on human judgement. For example, the linguistic value *high* for software reliability may mean that the failure intensity is lower than 6 per month, but in the future, we may require less than 3 software failures per month to evaluate it as *high*. In the case retrieval step, we can update the definition of the linguistic quantifier used in the environment. Here also, the meaning of a linguistic quantifier depends on human judgement. However, other learning characteristics that are not included in our approach remain to be examined. For example, Fuzzy Analogy must be able to provide its user with a subset of linguistic variables that have always led to accurate estimates in the past. We may then use this subset in the 'identification of cases' step of Fuzzy Analogy. Thus, the attribute selection problem

can also be addressed. Moreover, Fuzzy Analogy must be able to identify the appropriate linguistic quantifier to be used in 'retrieval of similar cases' step by detecting those that have often yielded accurate cost estimates.

ACKNOWLEDGEMENTS

We express our special thanks to Naeem Seliya for his patient editorial suggestions, reviews, and changes. We also thank Jayanth Rajeevalochanam for his reviews. Dr. Taghi M. Khoshgoftaar was supported in part by the National Science Foundation Grant CCR 9970893.

REFERENCES

1. A. Aamodt and E. Plaza. "Case-Based Reasoning: Foundational Issues. Methodological Variations and System Approaches", *AI Communications*, IOS Press, Vol. 7:1, 1994, pp. 39–59.
2. L. Angelis and I. Stamelos. "A Simulation Tool For Efficient Analogy Based Cost Estimation", *Empirical Software Engineering*, Vol. 5, no. 1, 2000, pp. 35–68.
3. B. W. Boehm. *Software Engineering Economics*, Prentice-Hall, 1981.
4. B. W. Boehm and *et. al.*, "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0", *Annals of Software Engineering on Software Process and Product Measurement*, Amsterdam, 1995.
5. L. Briand, T. Langley, and I. Wieczorek. "Using the European Space Agency data set: A replicated assessment and comparison of common software cost modeling", In *Proc 22nd IEEE International Conference on Software engineering*, Limerick, Ireland, 2000, pp. 377–386.
6. D. S. Chulani. "Incorporating Bayesian Analysis to Improve the Accuracy of COCOMO II and Its Quality Model Extension", *Ph.D. Qualifying Exam Report*, University of Southern California, February 1998.
7. J. C. W. Debuse and V. J. Rayward-Smith. "Feature Subset Selection within a Simulated Annealing Data Mining Algorithm", *Journal of Intelligent Information Systems*, vol. 9, pp. 57–81, 1997.
8. D. Dubois, F. Esteva, P. Garcia, L. Godo, R. I. deMantaras, and H. Prade. "Case-Based Reasoning: A Fuzzy Approach", *IJCAI'97, Workshop on Fuzzy Logic in Artificial Intelligence, Lecture Notes in Artificial Intelligence*, Vol. 1566, Springer, Berlin, 1999, pp. 79–90.
9. A. Idri, L. Kjiri, and A. Abran. "COCOMO Cost Model Using Fuzzy Logic", *7th International Conference on Fuzzy Theory & Technology*, Atlantic City, NJ, USA, February, 2000, pp. 219–223.
10. A. Idri and A. Abran. "Towards A Fuzzy Logic Based Measures For Software Project Similarity", *Sixth Maghrebian Conference on Computer Science*, Fes, Morocco, November 2000, pp. 9–18.
11. 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*, April 2001, London, England, pp. 85–96.
12. A. Idri and A. Abran. "Evaluating Software Project Similarity by using Linguistic Quantifier Guided Aggregations", *9th IFSA World Congress and 20th NAFIPS International Conference*, July 2001, Vancouver, pp. 470–475
13. A. Idri, A. Abran, and T. M. Khoshgoftaar. "Fuzzy Analogy: A New Approach for Software Cost Estimation", *11th International Workshop on Software Measurements*, August 2001, Montreal, Canada, pp. 93–101.
14. A. Idri, A. Abran, and T. M. Khoshgoftaar. "Estimating Software Project Effort by Analogy Based on Linguistic values", *8th International Software Metrics Symposium*, June 2002, Ottawa, Canada, pp.21–30. IEEE Computer Society.
15. R. Jager. "Fuzzy Logic in control", *Ph.D. thesis*, Technic University Delft, Holland, 1995.
16. B. Kitchenham and S. Linkman. "Estimates, Uncertainty, and Risks", *IEEE Software*, vol. 14, no. 3, pp. 69–74, 1997.
17. G. Kadoda, M. Cartwright, L. Chen, and M. Shepperd. "Experiences Using Case-Based Reasoning to Predict Software Project Effort", EASE, Keele, UK, 2000, pp.23.
18. C. Kirsopp, M. Shepperd, and J. Hart. "Search Heuristics, Case-Based Reasoning and Software Project Effort Prediction", *GECCO, Genetic and Evolutionary Computation Conference*, AAAI, July 2002, New York, NY, USA, 2002.
19. R. Kohavi and G. H. John. "Wrappers for Feature Selection for Machine Learning", *Artificial Intelligence* 97. 1997, pp. 273–324.
20. E. R. Mac Cormac, *A Cognitive Theory of Metaphor*, MIT Press, 1985.
21. I. Myrtveit and E. Stensrud. "A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models", *IEEE Transactions on Software Engineering*, 25(4):510–525, July/August, 1999.
22. F. Niessink and H. Van Vliet. "Predicting Maintenance Effort with Function Points", in *Proceedings of the International Conference on Software Maintenance*, Bari, Italy, 1997, IEEE Computer Society.
23. M. Shepperd C. Schofield, and B. Kitchenham, "Effort Estimation using Analogy", *ICSE-18*, Berlin, 1996, pp. 170–178.
24. M. Shepperd and C. Schofield, "Estimating Software Project Effort Using Analogies", *IEEE Transactions on Software Engineering*, 23(12): 736–743, November 1997.
25. M. Shepperd and G. Kadoda. "Using Simulation to Evaluate Prediction Systems", *7th International Symposium on Software Metrics*, April 2001, London, England, pp. 349–358, IEEE Computer Society.
26. D. B. Skalak. "Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms", *11th International Machine Learning Conference*, Morgan Kauffmann, 1994.
27. A. Tversky. "Features of Similarity", *Psychological Review*, 1984, pp.327–52.
28. S. Vicinanza and M.J. Prietulla, "Case-Based Reasoning in Software Effort Estimation", *Proceedings of the 11th International Conference on Information Systems*, 1990.
29. R. R. Yager and J. Kacprzyk. *The Ordered Weighted Averaging Operators: Theory and Applications*, Kluwer Publishing, Norwell, MA, USA, 1997.
30. R. R. Yager. "Quantifier Guided Aggregation using OWA Operators", *International Journal of Intelligent Systems*, 11:49–73, 1996.
31. L. A. Zadeh. "Fuzzy Sets", *Information and Control*, vol. 8, pp. 338–353, 1965.
32. L. A. Zadeh. "Similarity Relations and Fuzzy Ordering", *Information Sciences*, 1971, pp.177–200.
33. L. A. Zadeh. "Fuzzy Sets as a Basis for a Theory of Possibility", *Fuzzy Sets and Systems*, vol. 1, 1979, pp. 3–28.
34. L. A. Zadeh. "Fuzzy Logic, Neural Networks, and Soft Computing", *Communications*, 37(3):77–84, March 1994, Association of Computing and Machinery.
35. L. A. Zadeh. "Computing with Words", *IEEE Transactions on Fuzzy Systems*, 1996, pp. 103–111.
36. L. A. Zadeh. "From Computing with Numbers to Computing with Words: From Manipulation of Measurements to Manipulation of Perceptions", *IEEE Transactions on Circuits and Systems*, vol. 45, pp. 105–119, 1999.