An Evidence-Based Software Engineering Evaluation Approach
Shahab Saleghaffari, Nan Niu & Sandeep Reddivari

Abstract—Evaluation of software project management is a crucial content for software development enterprise. However, the process of software project management is burdened by various sources of uncertainty due to the lack of information for object of evaluation, differences in knowledge and experience of participated experts, fuzzy reviews of experts and other reasons. A step-by-step algorithm for uncertainty reasoning of software project management using mathematical tools of evidence theory is presented. It is shown that the presented evidential uncertainty reasoning is very efficient for evaluating the successful level of software project management and helpful for software development enterprise to improve management.

Keywords: evaluation, decision making, project management, data collection, evidence theory

1. Introduction

A software project should be in accordance with the pre-set costs, schedule and quality in order for its successful completion. Project management which aims to analyze and manage the cost, personnel, schedule, quality and risks helps successful completion of a software project. Through following software project management activities, managers can control software projects so that pre-set costs can be met and high quality products can be delivered to customers on time over the software life cycle which includes analysis, design, coding, testing and maintenance [1].

However, the involved activities in software project management is burdened by huge sources of epistemic uncertainties due to the lack of information for object of evaluation, differences in knowledge and experience of participated experts, fuzzy reviews of experts and other reasons. The probability theory is incapable of quantifying such epistemic uncertainties involved in software project management [2]. Instead, the Dempster-Shafer theory of evidence provides efficient means to deal with epistemic uncertainties coming from lack of knowledge, information and the conflicting expert opinions. This paper aims to investigate an evidential approach which is able to solve qualitative problems in software project management in a quantitative way.

Here, principals of evidence theory are studied. Different aggregation rules of evidence to combine different expert opinions and to address the issue of dealing with conflicting expert experiences is provided. Finally, a step by step decision making approach based on basis of evidence theory is given for a benchmark problem in software project management [1]. It is shown that the presented evidential uncertainty reasoning approach can handle software project management tasks effectively. Also, the presented evidential uncertainty reasoning framework is shown to be a comprehensive assessment method which combines qualitative analysis with quantitative analysis.

2. Principals of Evidence Theory

Evidence theory, also known as Dempster-Shafer theory of evidential reasoning, provides an alternative to the traditional probability theory by allowing less restrictive statements in representing uncertainty. As such, it is suitable for modeling epistemic uncertainty. Much of the fundamental work in this area was done by Dempster [3] and Shafer [4]. A brief overview of evidence theory is provided here, although a more thorough description can be found in the literature [3-5].

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a finite set of elements representing mutually exclusive hypotheses, events, or propositions (frame of discernment) and $P(X)$ the restricted set (power set) of subsets of $X$ including the empty set. The measure of uncertainty known as basic belief assignment (BBA) for each element of $P(X)$ is represented by the belief function $m: P(X) \rightarrow [0,1]$ with two properties: $m(\emptyset) = 0$ and $\sum_{A \subseteq P(X)} m(A) = 1$. Any subset $A$ of $X$ with non-zero BBA is called a focal element and represents the available belief that supports it. Hence, $m(A)$ expresses the proportion of all available evidence supporting the belief in $A$ [3]. The set of all focal elements represents the body of evidence. Unlike the Bayesian belief structure, it is possible to have a BBA for a subset consisting of multiple elements or just one element in $X$.

In contrast to the theory of probability, the following conditions can be encountered in evidence theory [3, 4]:

1. $m(\{x_1\}) + m(\{x_2\}) \neq m(\{x_1, x_2\})$
2. $m(\{x_1\}) \geq m(\{x_2\})$ even though $x_1 \subseteq x_2$
3. $m(X) < 1$.

For a given BBA, the lower and upper bounds of an
imprecise probability interval are defined by belief and plausibility functions expressed as

\[ Bel(B) = \sum_{A \subseteq B} m(A) \quad \text{for all} \quad B \subseteq X \] (Equ. 1)

\[ Pl(B) = \sum_{A \subseteq B \neq \emptyset} m(A) \quad \text{for all} \quad B \subseteq X \] (Equ. 2)

where A and B are subsets of X, Bel(B) represents the level of total confidence that supports trustworthiness of subset B, and Pl(B) represents the total belief that can be placed on subset B due to lack of knowledge or data.

Equations (1) and (2) give the probability bounds on B as \( Bel(B) \leq Pl(B) \leq I(B) \). The gap between plausibility and belief gives a measure of epistemic uncertainty (belief interval) or the imprecision on the “true probability” of hypothesis B [6].

Although in probability theory, the probability of a hypothesis and its complement add up to 1, the sum of the belief of a hypothesis and its complement is generally less than 1 in evidence theory and equal to 1 only when there is sufficient knowledge (no ignorance) about the hypothesis. Unlike probability theory, evidence theory suggests that the belief of a hypothesis and its complement plus the level of ignorance (uncertainty) should add up to one.

When the information or data comes from multiple sources of evidence, it is possible to encounter diverse belief structures with different assessments for the same frame of discernment. However, uncertainty can be represented only by a single belief structure. To reconcile this challenge, several approaches have been proposed for aggregation of evidence from all sources of information. The key to selecting a proper aggregation rule is to recognize how conflict and ignorance should be treated in a particular application. Amongst the number of aggregation rules reported in the literature [7, 8], Dempster’s rule [3] and Yager’s rule [5, 7] have attracted more attention.

Dempster’s rule ignores completely the conflict between information obtained from the different sources and attributes the BBA associated with conflict to the null set. Therefore, when high level of conflict exists, Dempster’s rule is not appropriate for combining evidence. Two pitfalls that are often cited for Dempster’s rule of combination are as follows:

1) Dempster’s rule assigns 100% uncertainty to a minority opinion when conflicting evidence exists [7].

2) The combination of information from an evidence source that assigns a BBA to the base set (means complete ignorance is considered by this source of evidence) with that which does not consider ignorance results in BBA structure that does not assign BBA to the universal set. This issue gives a false impression that precise probabilistic information dominates the belief.

Yager’s rule [5, 7] was developed to address some of the shortcomings of Dempster’s rule. The two major differences between Dempster’s and Yager’s rules of aggregation can be summarized as follows:

1) Yager’s rule does not change the evidence through normalization as opposed to Dempster’s rule.

2) Yager’s rule allocates conflict to the universal set X as opposed to Dempster’s rule that attributes conflict to the null set \( \emptyset \).

Given the Yager’s rule’s ability to more accurately deal with conflicting evidence, we have henceforth adopted this rule of aggregation. For a thorough mathematical description of Yeager’s rule, the reader is referred to [5, 7].

3. Description of the Software Project Management

Investigations on software project management and the associated risks has been reported in [1, 9]. Time, money, personnel and customers which are the key elements of software project can be affected by many risk factors. Interestingly enough, these risks introduce different effects at different stages of software project life cycle (problem definition, feasibility analysis, general description, system design, coding, debugging, testing, check and accept, operational maintenance, upgrade and abandon). A typical description of a software project management is given in figure 1. For more precise description of a software project management problem, other contributing factors should be considered in the process of software management in addition to personal management, project process control and customer satisfaction. Also, all of these main factors should be divided into more sub-sections in a more informative software project management problem. Here, the aim is to evaluate a benchmark problem in software project management. Note that the approach can be adopted for more complicated problems as well. Next sections explains the benchmark problem and application of the presented evidential uncertainty reasoning approach.

4. Description of the Benchmark Problem

For the sake of simplicity, a benchmark problem for the presented software project management is divided only to main influencing parameters of personal management, project process control and customer satisfaction. No sub-sections are considered herein.

A number of three experts in one group and the other two in another group are reviewed for evaluation of the three main influencing parameters given in the presented benchmark problem. They are asked to raise their opinions on different aspects of software project by three choices of poor, good and excellent.

The reviewing results are summarized in table 1 for both groups of experts. In table 1, P, G and E represent poor, good and excellent evaluation. Note that some experts were not sure on their evaluation and gave fuzzy response like “I am not sure if it was poor or good”. These fuzzy responses include multiple elements and are mentioned like PG in table 1. It is worth noting that an evaluation like PGE means that the expert had no idea for evaluation.
5. A Step-by-Step Evidential Uncertainty Reasoning Approach

A. BBA Construction for Expert Opinions

The first step for decision making on software project management evaluation based on fuzzy expert opinions is to represent these opinions in the mathematical form of evidence theory. To do so, the probability of each opinions of expert (see table 1) for all factors of software project should be considered as degree of belief (or BBA structure) supporting that opinion. Table 2 shows mathematical representation of expert opinions to be used in the presented evidential uncertainty reasoning framework. It can be seen that a degree of belief (BBA) is assigned to each expert responses based on the available information provided in table 1.

B. Aggregation of Expert Opinions

As it can be seen from table 2, expert responses in different groups suggest different BBA structures. However, for the purpose of decision making, a consolidated BBA structure which includes key information of expert responses in both groups 1 and 2 is needed. Here, Yager’s aggregation rule of evidence is adopted to provide the mentioned informative BBA structure for expert opinions. Conceptually, Yager’s rule assigns a BBA to the intersection of two responses from different sources of evidence. Responses that are in agreement between different sources are treated as valid evidence. In the case of no intersection (conflict between two responses from different sources), Yager’s rule allocates the associated conflicting BBA to the universal set (the set PGE). In fact, Yager’s rule considers conflict between two expert responses as ignorance or lack of knowledge, and—as a result—assigns a BBA to ignorance response (PGE). Using Yager’s rule, a unique BBA structure representing expert responses in groups 1 and 2 for each influencing factors of software project is provided in table 3.

C. Joint BBA Construction for Expert Opinion

As it can be seen from figure 1, evaluation of software project management urges assessment of personnel management, project control and customer satisfaction. Experts give fuzzy opinions for each of these contributing factors and as a result make the evaluation of project management so uncertain.

Uncertainty quantification of a system with dependence on multiple uncertain factors requires the construction of a joint belief structure, which is equivalent to joint probability distribution in probability theory. A joint BBA is obtained by the Cartesian product of the belief structures of all uncertain factors for a system [10-13]. This involves the multiplication of final BBA found for each expert responses of one contributing factor (see table 3).
with those of the other factors involved in the Cartesian product. In this example, the constructed joint belief structure contains 24 members as obtained by multiplication of 4 BBAs for personnel management, 2 BBAs for project control and 3 BBAs for customer satisfaction (see table 3). The joint BBA structure of expert responses for evaluation of software project management is provided in table 4. For each member of joint BBA in table 4, the first, second and third element relates to expert opinions on personnel management, project control and customer satisfaction, respectively.

In fact, the presented BBA structure in table 4 represents all fuzzy expert opinions on different aspects of software project management in an understandable mathematical form of evidence theory. This joint BBA structure will be adopted for evaluation of software project management in next step.

### Table 4. Final BBA of Expert Responses

<table>
<thead>
<tr>
<th>No.</th>
<th>Response</th>
<th>BBA</th>
<th>N. o.</th>
<th>Response</th>
<th>BBA</th>
</tr>
</thead>
</table>

### D. Software project management evaluation

Prior to evaluate software project management using the constructed joint BBA representing all fuzzy evaluation of experts in the mathematical form of evidence theory, a criteria for consideration of software project management as successful activity should be determined. Again, this criterion should be represented in mathematical form of evidence theory. Here, a software project management is considered as successful activity when experts evaluate all of its contributing factors (personnel management, project control and customer satisfaction) as good or excellent. Mathematically we seek to estimate the belief and plausibility (uncertainty measures of evidence theory) of the set [GE, GE, GE] to evaluate software project management. Considering equations (1) and (2), the belief and plausibility of a successful project management using the constructed joint BBA structure of expert opinions (see table 4) can be estimated as following:

\[ \text{Bel}([\text{GE, GE, GE}]) = \text{BBA}(14)+ \text{BBA}(15)+ \text{BBA}(17)+ \text{BBA}(18) = 1/216 + 1/72 + 5/216 + 5/72 = 0.111. \]

\[ \text{Pl}([\text{GE, GE, GE}]) = \text{BBA}(8)+ \text{BBA}(9)+ \text{BBA}(11)+ \text{BBA}(12)+ \text{BBA}(14)+ \text{BBA}(15)+ \text{BBA}(17)+ \text{BBA}(18)+ \text{BBA}(20)+ \text{BBA}(21)+ \text{BBA}(23)+ \text{BBA}(24) = 1/216 + 5/216 + 5/72 + 1/216 + 1/72 + 5/216 + 5/72 + 1/108 + 1/36 + 5/108 + 5/36 = 0.444. \]

Hence, the probability of successful software management activity is somewhere between 0.111 and 0.444. In fact, fuzzy evaluation of experts causes imprecision on true probability of a successful software project management activity which is equal to 0.111 (Bel). The amount of imprecision on true probability of a successful software project management activity equals to the gap between belief and plausibility which is 0.333 = 0.444 - 0.111 due to the fuzzy expert opinions. The low values of the estimated belief and plausibility indicates that the software project management activity was not successful.

It is worth noticing that for more precise evaluation of software management, a more detailed review is necessary. A detailed review includes evaluation for various branches of main contributing factors of software project management. Also, in such detailed review, expert should have more choices for their evaluation instead of only three choices of poor, good and excellent. In fact, necessary information can be extracted better from a more detailed review and this can help experts to give less fuzzy responses. In case of having fuzzy and conflicting expert opinions, a huge gap between belief and plausibility will be created and this can make the evaluation of software project management more challenging. However, as it is shown in this paper, evidence theory is able to handle evaluation of software project management when expert gives fuzzy and conflicting responses.

### 6. Conclusion

Mathematical tools of evidence theory are used to evaluate software project management when experts give fuzzy evaluations. The process of software project management is divided into three sections to help experts for their evaluations. Aggregation rules of evidence are used to combine evaluation of experts in two different groups. Collected evaluation from experts for each contributing factors of software project management is determined in mathematical form. A joint belief structure that puts together all necessary evaluation on different aspects of software project management is constructed and used for decision making. A criterion for successful completion of software project management is defined and determined in an understandable mathematical form. Using uncertainty measures (belief and plausibility) of evidence theory, satisfying completion level of software project management tasks is determined quantitatively. It is shown that when fuzzy expert opinions exist, evidence theory provide a good means for evaluation of software project management.
References


Shahab Salehghaffari is a Postdoctoral Fellow at Northwestern University. He got his PhD degree in Computational Engineering, 2009 from Mississippi State University. His research interests include: Bayesian and Non-probabilistic Uncertainty Quantification, Design under Uncertainty, Evidence-Based & Reliability-Based Design Optimization, Software Reliability and Software Engineering.

Nan Niu is an Assistant Professor in Computer Science and Engineering at Mississippi State University, USA. His research interests include software engineering, requirements engineering, information informatics, and human and social aspects of software development. He received his Ph.D. from the University of Toronto, Canada, M.Sc. from the University of Alberta, Canada, and B.Eng. from Beijing Institute of Technology, China, all in Computer Science.

Sandeepr Reddivari is a PhD student in the Department of Computer Science and Engineering, Mississippi State University. He received his MS in Computer Science and Engineering from Texas A&M University-Kingsville and B. Tech degree in Electronics and Communications Engineering from JNT University, India. His research interests include software engineering, requirements engineering, software exploration and program comprehension.