This is the published version (version of record) of:


Available from Deakin Research Online:

http://hdl.handle.net/10536/DRO/DU:30003581

Reproduced with the specific permission of the copyright owner.

Copyright : 2006, The authors
Fuzzy Logic Based Metric in Software Testing ¹

Yanhui Zhou a,b, Zili Zhang a,b

a Key Lab of Intelligence Software and Software Engineering
Southwest University, Chongqing 400715, China
Email: [xiaohui_zhangzlj]@swu.edu.cn

b School of Information Technology, Deakin University, Victoria, 3217, Australia
Email: [yzhou_zhang]@deakin.edu.au

Abstract. How to provide cost-effective strategies for Software Testing has been one of the research focuses in Software Engineering for a long time. Many researchers in Software Engineering have addressed the effectiveness and quality metric of Software Testing, and many interesting results have been obtained. However, one issue of paramount importance in software testing— the intrinsic imprecise and uncertain relationships within testing metrics—is left unaddressed. To this end, a new quality and effectiveness measurement based on fuzzy logic is proposed. The software quality features and analogy-based reasoning are discussed, which can deal with quality and effectiveness consistency between different test projects. Experimental results are also provided to verify the proposed measurement.

Keywords. Fuzzy Logic, Software Testing, ISO/IEC 9126, Testing Metric

1. Introduction

Software industries face a continuous increasing cost of Software Testing. Now that, improving efficiency and increasing effectiveness and quality in Software Testing are two main strategies for reducing testing cost.

1980s saw many studies on improving testing efficiency. Several White-Box methods for automated generation of test cases are described in [1,2,3]. The Effectiveness and quality metric of testing, which is always treated as management issues in Software Engineering (SE), is often solved by an experiential judgement and qualitative analysis. For the sake of quantitative management, many good Software Testing metrics have been defined in [4,5,6]. These traditional approaches, which test programs by injecting simulated faults into the codes, became more mature after a further extension [7]. The ratio of injecting faults to inherent faults found by testers is a basic metric on which project managers rely to make a decision about the effectiveness and quality of testing. Since the source code is always unavailable in Black-Box testing, these metrics are restricted in White-Box testing. On the other hand, the faults injecting process also has some negative effects for the testing cost.

¹Works in this paper are partly supported by projects: swnuq2005022, CSTC2004BB0146 and 200401021.
To our thinking, the primary barrier of reaching more inherent metrics is some imprecise and uncertain relations. These relations, such as test engineer’s experience, software domain knowledge, and quality consciousness, are intrinsic features but are barely represented and measured with their inbeing. Until now, the SE community has often used numbers or classical intervals to represent these vague values. Such transformation and representation does not mimic the way in which the human-mind interprets such relations, and consequently, it cannot deal with imprecision and uncertainty accurately. Therefore, it is necessary to obtain a new metric which can deal with this vague relationships and adapts well to Black-Box testing too.

Some Artificial Intelligence (AI) techniques like fuzzy logic, which is effective to deal with imprecision and uncertainty, have been widely used in SE[8,9], it was even adopted in test case generation method [10] to improve the efficiency of White-Box testing. It is a new attempt to apply fuzzy logic in Software Testing metric. This rigorous and effective metric will highly improve the ability and efficiency to the testing management. Moreover, the cost and workload of the testing will drop down. As a goal of our works, our metric can induce an effective testing whose cost is reasonable, and whose result is reliable and accurate.

2. Fuzzy Logic Based Metric

2.1. Test Quality Characteristics Applied in Test Cases

To address the issues of software product quality, the Joint Technical Committee 1 of the International Organization for Standardization (ISO) and International Electro-technical Commission (IEC) published a set of software product quality standards known as ISO/IEC 9126 [11,12]. ISO/IEC 9126-1 defines a quality model that comprises six characteristics and 27 subcharacteristics of software products quality (see Table 1).

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Subcharacteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>Suitability, accuracy, interoperability, security, functionality compliance</td>
</tr>
<tr>
<td>Reliability</td>
<td>Maturity, fault tolerance, recoverability, reliability compliance</td>
</tr>
<tr>
<td>Usability</td>
<td>Understandability, learnability, operability, attractiveness, usability compliance</td>
</tr>
<tr>
<td>Efficiency</td>
<td>Time behavior, resource utilization, efficiency compliance</td>
</tr>
<tr>
<td>Maintainability</td>
<td>Analyzability, changability, stability, testability, maintainability compliance</td>
</tr>
<tr>
<td>Portability</td>
<td>Adaptability, installability, replaceability, coexistence, portability compliance</td>
</tr>
</tbody>
</table>

In a typical validation testing project, overall, a series of test cases which address different functions of a software product constitute the test project. To fulfill adequate test strength, test cases should overlap in different functions or quality characteristics of software, i.e., each test case has different correlations with the quality characteristics. The correlations are vague and uncertain, yet it can be measured by linguistic values and fuzzy sets. Apparently, test projects have further similarity of correlations between test cases and quality characteristics, they have more consistency of quality and effectiveness.
2.2. Fuzzy Sets of The Correlations

The goal of this step is the characterization of all test projects by a set of quality characteristics of software products. In order to deal with the correlations between the test cases and the characteristics, the Fuzzy Weight assignment algorithm is adopted.

A Weighting function is used to associate a weight $w_{i,j}$ to each Quality characteristics $C_j$ in ISO/IEC 9126. This function is called Feature Weighting Function (FWF) and is the classical weighting function [13] used for relevance computation in document-based domain. FWF, whose details can be found in [14], leads us to the following:

**Definition 1** Let $w_{i,j}$ denote a weight of a quality characteristic $C_j$ which depends on the ratio between the number of steps of test case $i$ where $C_j$ occurs and the total number of occurrences of $C_j$ in the whole test cases’s steps. The total weight $w_j$ is sum of $w_{i,j}$ in a test project which has $n$ test cases, as follow:

$$w_j = \sum_{i=1}^{n} w_{i,j} = \sum_{i=1}^{n} \left( \frac{\text{step}_{i,C_j}}{\text{step}_{C_j}} \times 100 \right)$$

Where, the $\text{step}_{i,C_j}$ is the number of test steps where $C_j$ is relevant in the $i^{th}$ test case.

Consider the propositions: correlation to a characteristic is low, nominal, average, and high, the linguistic variable correlation to a characteristic is constrained by linguistic value Low, Nominal, Average and High, for convenient expression, we can treat characteristic as linguistic variable. We have proposed to use the representation given in Figure 1. The weight associates to the Eq.(1).

![Figure 1. Membership Function of Fuzzy Sets](image)

For instance, Table 2 shows a part of calculation results of weight, linguistic value, and a membership value according to FWF function.

<table>
<thead>
<tr>
<th>Quality Characteristic</th>
<th>Weight</th>
<th>Linguistic Value</th>
<th>$\mu$</th>
</tr>
</thead>
<tbody>
<tr>
<td>accuracy</td>
<td>91.3</td>
<td>High</td>
<td>1</td>
</tr>
<tr>
<td>accuracy</td>
<td>91.3</td>
<td>Average</td>
<td>0</td>
</tr>
<tr>
<td>security</td>
<td>34.3</td>
<td>Average</td>
<td>0.93</td>
</tr>
<tr>
<td>security</td>
<td>34.3</td>
<td>Nominal</td>
<td>0.07</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
2.3. Fuzzy Analogy of the Quality Similarity

Correlation of test cases and characteristics can be represented by fuzzy sets, which are divided by the linguistic values. To the characteristics \( Q \) (whole or a subset of characteristics), \( C_j \) is the \( j^{th} \) characteristic in \( Q \) and represents one linguistic variable. \( A^k_k \) is a fuzzy set associated with \( k^{th} \) linguistic value. \( \mu_{A^k_k} \) represents the membership function of \( A^k_k \).

To compare the similarity of fuzzy sets associated with linguistic variable(quality characteristics) \( C_j \) of two test projects \( P_m \) and \( P_n \), in fuzzy set theory, the associated fuzzy sets must have membership functions with two variables \( C_j(P_m) \) and \( C_j(P_n) \). These fuzzy sets refer to a fuzzy relation. We define the if \(-\) then rule in fuzzy relation as follows:

**Definition 2** Let \( R^C_j(P_m, P_n) \) be a fuzzy relation associated with fuzzy variable \( C_j \) and its specific value \( A^k_k \), this relation implies the following if-then rule:

\[
\text{if } C_j(P_m) \text{ is } A^k_k \text{ then } C_j(P_n) \text{ is } A^k_k
\]  

(2)

Hence, for each linguistic variable \( C_j \), the associated fuzzy relation will imply the if \(-\) then rule. Based on this rule, the similarity can be calculated by each linguistic value in the fuzzy relation. The definition is given below:

**Definition 3** Let \( S^C_j(P_m, P_n) \) denote the similarity of two projects \( P_m \) and \( P_n \) associated with the \( k^{th} \) linguistic variable \( C_j \), given as:

\[
S^C_j(P_m, P_n) = \text{simp}(\mu_{A^k_k}(P_m), \mu_{A^k_k}(P_n))
\]

(3)

Where, \( \text{simp}(x, y) \) is an unspecific function to calculate the similarity of two parameters.

The fuzzy relation \( R^C_j(P_m, P_n) \) is associated with each linguistic value of fuzzy variable \( C_j \), the combination of this relation is a fuzzy relation associated with all linguistic values of variable \( C_j \), we denote this as \( R^C(P_m, P_n) \). Then applying the combination of the fuzzy if \(-\) then rules into \( R^C(P_m, P_n) \), is called as a aggregation. Based on the aggregation, we define the similarity of two projects associated with all linguistic values of \( C_j \) as follow:

**Definition 4** Let \( S^C(P_m, P_n) \) denote the similarity of two projects \( P_m \) and \( P_n \) associated with the all linguistic values of variables \( C_j \), given as:

\[
S^C(P_m, P_n) = \text{aggr}(S^C_j(P_m, P_n))
\]

(4)

Where, \( \text{aggr}(x) \) is an unspecific function which calculates the aggregation of parameter \( x \).

In Eqs. (3) and (4), \( \text{simp}(\cdot) \) and \( \text{aggr}(\cdot) \) are unspecific, we can use possible functions and combinations of these functions based on the environment and context of the usage. We prefer these simple functions:

\[
\text{simp}(x, y) = \begin{cases} \min(x, y) & x \times y \\ \max(x) & \sum x \end{cases}
\]

Then, the possible combinations of these functions described in Eqs. (3) and (4) are as follow:
\[ S_{C_i}(P_m, P_n) = \begin{cases} 
\max(\min(\mu_{A_i^k}(P_m), \mu_{A_i^k}(P_n))) & \text{max - min aggregation} \\
\max(\mu_{A_i^k}(P_m) \times \mu_{A_i^k}(P_n)) & \text{max - product aggregation} \\
\sum_k \mu_{A_i^k}(P_m) \times \mu_{A_i^k}(P_n) & \text{sum - product aggregation} \\
\sum_k \min(\mu_{A_i^k}(P_m), \mu_{A_i^k}(P_n)) & \text{sum - min aggregation} 
\end{cases} \]

These four aggregation functions are not all necessary when calculating the similarity of test projects, and they should satisfy some intuitive and obvious rules in the representation context. The function should be omitted if it contradicts any rule. Some axiom was proposed in [8] to validate the aggregation functions. In our work, three axioms are selected for validation as follows:

\[ S_{C_i}(P_m, P_n) \neq 0 \iff \exists A_k \text{ such that } \mu_{A_k^n}(P_m) \neq 0 \text{ and } \mu_{A_k^m}(P_n) \neq 0 \] (5)

\[ S(P, P_l) \leq S(P, P) \] (6)

\[ S_{C_i}(P_m, P_n) \leq 1 \] (7)

<table>
<thead>
<tr>
<th>Functions</th>
<th>Eq.(5)</th>
<th>Eq.(6)</th>
<th>Eq.(7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>max-min</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>max-product</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>sum-min</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>sum-product</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

The validation results of function combinations are shown in Table 3. By observing the result, only max-min aggregation obeys all axioms, we prefer this aggregation function in our metric.

3. Empirical Results

To ensure the experiential result reflecting the testing practice in reality, we collect the test data from actual Software Testing projects from a third-party software testing organization in China. The experiments are designed in three categories which are three kinds of sample data and verify different aspects of our metric.

Characteristics in ISO/IEC 9126 can be tailored for specific requirement. We select 12 characteristics which can be tested by executable steps and familiar to most software (Table 4).

<table>
<thead>
<tr>
<th>1</th>
<th>accuracy</th>
<th>2</th>
<th>interoperability</th>
<th>3</th>
<th>security</th>
<th>4</th>
<th>fault tolerance</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>recoverability</td>
<td>6</td>
<td>operability</td>
<td>7</td>
<td>Time behavior</td>
<td>8</td>
<td>resource utilization</td>
</tr>
<tr>
<td>9</td>
<td>stability</td>
<td>10</td>
<td>installability</td>
<td>11</td>
<td>replaceability</td>
<td>12</td>
<td>coexistence</td>
</tr>
</tbody>
</table>

In first category, the results of similarity between two projects shown in Figure 2(a), which denote each similarity value of quality characteristics. Some lower similarity val-
ues, such as replaceability (11) and time behavior (7), don't indicate a great difference of similarity. The max-min aggregation function will produce a similarity value less than 1 when both membership function values are less than 1 but centralizing in a small interval. In fuzzy theory, it is called as vicinity of 1. Therefore, as we predicted, measurements of each characteristic in these two projects owned by the same tester are in common, and two projects have a high overall similarity value too.

Figure 2. (a) Two test projects owned by one tester, whose software types are same, are randomly selected. (b) Two test projects owned by one tester, whose software types are different, are randomly selected. (c) Three test projects owned by different testers with different software type, are randomly selected.

In the category 2, two test projects target two software products whose types are different (Figure 2(b)). One software is a Web based ERP application, another is an intellectual desktop game. Similarity value of security (3) is zero, we found that this interesting value is due to existing 0 test case of security characteristics in desktop game project. Probably, the tester thought it is unnecessary to test security in a desktop game. When we checked another bulge point which is at time behavior (7), as a result, there is only one test case relevant to time behavior in the game test, which is game loading time test, but 12 test cases, such as open file time, database update time, and query time etc, exist in ERP application. Similarly, the third bulge point is at installability (10), which is cause by a quite different complexity of installation for this two software.

To achieve the third experiment, we randomly select three GIS-based software products. We create a repository from some projects which have a high evaluation after testing. We calculate quality measurement value of three projects by comparing each project with the repository (Figure 2(c)). It is obvious that the quality undulations are always centralized in some non-functional characteristic such as Interoperability, security, and coexistence etc.
4. Conclusion

This paper focused on how to improve testing cost and quality, which are two important issues in software testing. A new metric was proposed for this purpose. The characteristics in ISO/IEC 9126 were employed as unified quality features. Fuzzy sets were introduced to represent vague and uncertain correlations between test cases and quality features, which led to more appropriate representations of imprecise and uncertain correlations between constituents in a test project. With the proposed new metric, the quality and effectiveness of test projects can be measured easily without increasing testing cost. Especially it can improve the ability of quality evaluation and measurement in an early stage of testing. It can further be used to tackle many quality relative issues in testing management such as quality comparison and quality compliance analysis.

Extensive experiments were conducted. The experimental results show that the proposed metric is effective to measure and evaluate quality and effectiveness of test projects. It is a powerful tool for test management usage. It was also noticed in the experiments that the software type has a notable interference in the metric, which implies that this metric is more suitable for applying in testing software products with common types.

References