Chiang Mai J. Sci. 2012; 39(3) : 486-497
http://it.science.cmu.ac.th/ejournal/
Contributed Papers

Multi-Faceted Measurement Framework for Test Case Classification and Fitness Evaluation using Fuzzy Logic Based Approach

Manoj Kumar*[a], Arun Sharma[b], Rajesh Kumar[c]

[a] Department of Computer Application, Galgotias University, Gr. Noida- India.

[b] Dept of CSE, Krishna Institute of Engineering & Technology, Ghaziabad- India.

[c] School of Mathematics & Computer Application, Thapar University, Patiala- India.

*Author for correspondence; e-mail: m_pachariya1@yahoo.com

Received: 19 August 2011

Accepted: 9 February 2012

ABSTRACT

The target of software engineering is to produce high quality software product at low cost. Software testing consists of three main activities: selecting tests cases, execution of test cases on the software under test, and evaluating the correctness of the outputs. The first and third of these activities are labour-intensive, ambiguous and error prone. How to provide cost-effective strategies for software test cases classification, minimization and selection has been one of the research focuses in software testing for a long time. Many researchers and academicians have addressed the effectiveness/fitness, selection, classification, minimization of software test cases, and obtained many interesting results. Quality of software testing is low due to uncertainty & inadequate techniques for estimating the fitness, classification, selection of test cases and requires improvement. Test cases fitness depends on several parameters. Vagueness of fitness of test cases and their fitness parameters have created the uncertainty in classification and selection of test cases. However, one issue of paramount importance in software testing i.e. the intrinsic imprecise and uncertainty of test cases fitness, fitness parameters, multi-objective classification and selection, is left unaddressed. However, by applying appropriate test case classification, selection techniques, testing efforts can be reduced considerably. Moreover, by using the multi-faceted classification, selection of test cases with test data adequacy criteria will help in improving the overall quality of the software. Fuzzy logic based multi-faceted measurement framework will be the solution to the problem of test cases fitness evaluation on multiple parameters, and multi-objective test case classification, selection problem. Present paper gives two contributions to software testing research first uncertainty in software testing and secondly insight into multi-faceted measurement framework for test cases classification and fitness evaluation using fuzzy logic based approach.

Keywords: fuzzy synthesis evaluation, multi-faceted classification, test cases, test data adequacy criteria, test case fitness, uncertainty.

1. INTRODUCTION

Software has been a major enabling technology for advancing modern society, and is now an indispensable part of daily life. Complexity, difficulty risks and fuzziness grows day by day in software testing. Because of the increased complexity of these software systems, more effective software testing technologies were needed. Coming in to next generation, mankind society development has already entered into age of intelligent, soft computing and cloud computing techniques. Test cases fitness evaluation, multi-objective test cases classification and selection may be crucial problem for next generation software testing sorority. It requires to device an intelligent and soft computing techniques and methods continuously in long term of research to improve quality of software testing gradually. Software testing plays a vital role in high quality software development. Although software testing is a very time consuming activity and itself an expensive activity, yet launching of software without proper testing may lead to cost potentially much higher than that of testing, specially in systems where human safety is involved [1,2]. The effectiveness of this verification and validation process depends upon the number of errors found and rectified before releasing the system. This, in turn, depends upon the fitness and number of test cases exercised. Test cases are the inputs to the program under test. A test case is a set of conditions or variables under which a tester will determine whether an application or software system is working correctly or not. Test cases pool may contain some redundant, irrelevant and unfit test cases. Since, testing is very expensive process, unnecessary execution of redundant, irrelevant and unfit test cases will increase unnecessary burden of cost. The solution is to choose only the fit test cases and removing the unfit, redundant unnecessary ones, which in turn leads to test cases classification and selection [3-5]. So, test cases classification and selection is required to improve the software testing. Measuring fitness of test cases is always a daunting task. The term “fitness” refers to the appropriateness of test cases to check the quality of software.

An optimization problem is the problem of finding the best solution from all feasible solutions. Multi-Objective optimization (MO) also known as multi-criteria or multi-faceted or multi-attribute optimization is the process of simultaneously optimizing two or more conflicting objectives subject to certain constraints. The objective of MO optimization is to find the set of acceptable solutions and present them to the decision maker to take decision. Test cases classification and selection is the problem of finding the best subset (class) of test cases from a pool of the test case to be audited. It will meet all the objectives of testing concurrently. Most of the researchers evaluated the fitness of test cases only on single parameter fault detecting capability. Though, the fitness of test case depends on several parameters but consideration of only one parameter is not appropriate. The role of the fitness function is to capture coveragebility of multiple test objectives, when achieved, makes a contribution to the desired test adequacy criterion. Using the fitness function as a guide, the multi-objective classification and selection approach seeks test cases that maximize the achievement of all test objectives. Test cases should be classified in such a way that it will achieve maximum of code coverage, maximum of client requirements coverage, high fault detecting capability, maximum mutant killing score. Though, there are several objectives of test case classification and selection, discussed in details in section 3. However most of test cases classification and selection approaches found in the literature are single objective. Some objectives of test cases classification and selection are conflicting in nature, coveragebility of one objective will suffer other objective like cost, fitness of test cases and number of test cases in class while considering all objectives concurrently. It is not appropriate to estimate fitness of test cases just on single parameter and classify and select the test cases on single objective. It is not meeting the objectives of testing. The objective of test case classification and selection is to reduce the number of test cases in classes to be audited and improve the effectiveness of testing process by reducing the efforts, cost, and uncertainty. So, test cases fitness evaluation, classification and selection of test cases should be treated as multi-faceted concept [6,7]. Proposed framework will surely reduce the cost & efforts, uncertainty of software testing and reduce the number of test cases to be exercised also.

Fuzziness of test case fitness, vague nature of fitness parameters, multi-faceted classification, and selection of test cases, error tracing, human involvements and quality of estimation have created uncertainty in software testing, discussed in details in section 2. Fuzzy logic is a powerful problem-solving methodology and provides a remarkably simple way to draw definite conclusions from vague, ambiguous or imprecise information. It helps in taking decision from approximate data and finds precise solutions. It provides a mathematical framework where vague, conceptual phenomena can be rigorously studied. Therefore, fuzziness as a means of modeling linguistic uncertainty can be very well used to model software testing problems. So, we decided to use fuzzy multi-faceted measurement framework for evaluating the fitness, classification and selection of test cases. This paper is organized as follows: In Section 2, we discuss the uncertainty in software testing relevant to the material presented in this paper. Section 3 brings out the fuzzy logic base multi-faceted measurement framework for fitness evaluation, classification and selection of test cases. The final conclusions that we can draw from proposed framework are presented in Section 4.

2. UNCERTAINTY IN SOFTWARE TESTING

Uncertainty is present in our everyday lives. It is also present in software testing. Uncertainty in the problem domain, uncertainty in the solution domain and human participation are three main sources of uncertainty in software testing. Uncertainty in software testing is available due to uncertainty of fitness of test cases, fitness parameters, conflicting multi-objective, test execution, host environment of testing, multi-objective test cases selection, classification, prioritization, test schedules, early test planning, artifacts (SRS, SDD, Source Codes) error checking, quality of estimation and other testing activities. The intrinsic imprecision in fitness of test cases, fitness parameters, quality of estimation, multi-objective classification and selection have been identified as the paramount issues of software testing. Tester has to answer the following questions with economic criteria: What test cases shall tester use to exercise the program?, How to select the test cases with maximum coverage ability?, When and how to determine whether testing has been conducted adequately?, When to stop testing and whether to continue the testing?, When to stop optimization and whether to continue the optimization?, How to determine that whether generate optimized test cases or optimize the randomly generated test cases, which one is the better approach?, How to determine the quality/fitness of test cases?, What will be the probability of test case failure? and others [7-9]. Because of the lack of known strategies and precise information, decisions like these are made on the basis of the experience, intuitive assessments and heuristic rules.

Software testing is also human intensive and thus introduces uncertainty. Human participation includes active role played by humans in every stage of the software lifecycle inevitably introduces uncertainty and unpredictability into software testing. Psychology and mood of human participated in software testing is uncertain, unpredictable. Psychology, knowledge and experience of human have an impact on software testing. Automation of testing process does not require human intervention but it is not necessarily free of uncertainties. Instead, multiple factors or objectives exist; discussed in subsection 3.1, introduce uncertainties to test case fitness values. Fitness parameters, objectives of test cases classification and selection are fuzzy and vague in nature. One objective is more important in one domain or project may not be important for other domain or project e.g. cost is important for web application but it is not so important for embedded software but it requires correctness and precision. Uncertainty is also found in the defect-detection abilities of testing criteria. Since, only exhaustive testing in an ideal environment guarantees absolute confidence in the testing process and its results. This ideal testing scenario is infeasible for all but the most trivial software systems. These uncertainties will certainly affect the testing effort, quality, cost and test cases optimization. So, software testing techniques are outdated and require next generation computing techniques.

3. MULTI-FACETED MEASUREMENT FRAME WORK FOR TEST CASE CLASSIFICATION

Tester will desire to find the class of test cases that accomplish multi-objectives concurrently in order to maximize the value obtained from inherently expansive process of executing several test cases, investigate the output produced by them. Fuzzy multi-faceted approach for test cases classification and selection will be new approach. It will provide optimal or near optimal solution to the test case classification and selection problem. So, multi-faceted measurement framework helps tester to take decision for selecting / filtering/ prioritizing the test cases with highest adequacy. Fuzzy synthesis evaluation algorithm evaluated each test case on multiple criteria at different level. Tester uses multi-faceted measurement framework to get the fitness score of the test cases. Subsequently, assign grades to each test case on the basis of their fitness score using student grading system. Thereafter, test cases are classified into different classes using final grades. Proposed framework helps tester to select and exercise the efficient test cases from various alternate test cases. It will surely reduce the uncertainty, cost & efforts of software testing and improve the quality of software testing. The grades shall be derived from fitness scores of test cases. Test cases are finally awarded by seven grade scales like Excellent (A), Better (B), Good (C), Common/Average(D), Bad (E), Worse (F) and Worst (G)[10]. After grading each test cases the tester might develop a relation between grades and fitness parameters by assigning fitness value using Gaussian membership function.

Proposed framework requires identi fication of parameters, sub-parameters for test case fitness and weight distribution approach. It also requires mathematical model and calculation methods for weight distribution, fitness evaluation. Result of quantitative measurement should be analyzed and design concretely. We used fuzzy feature weighting function to calculate the weight value of each fitness parameter and sub-parameter. In, test cases evaluation systems, first of all determine the evaluation index system according test cases fitness parameters. Subsequently inspection methods are used for measuring the metric elements and getting the value of these metric elements. Weight values of these metrics elements belonging to parameters/sub-parameters are calculated. Same method can be used to evaluate the measured value of every sub-parameters of test cases fitness. The purpose of evaluation is to measure the fitness of test cases on all parameters concurrently, and compare them with predefined evaluation rating/grading system. Final assessment for test cases fitness will be carried out for test cases classification and selection [11]. Proposed framework estimates the fitness of test cases with consideration of several objectives/parameters concurrently; classify the test cases into two broad categories: Eligible for Qualifying Certificate of Fitness/Efficiency (EQCF) and Eligible for Improvement of Fitness (EIOF). Tester will assign the grades to each test case after evaluating the fitness of test cases in numeric score. Test cases are evaluated on several parameters on seven point grading system. Subsequently test cases are filtered in two pools EQCF and EIOF. Tester executes only those test cases having qualifying certificate of fitness and chunk out the test cases, those are eligible for fitness improvement. Test cases belonging to EIOF category are further divided into two sub pools Most Eligible for Improvement of Fitness (MEIOF) and Least Eligible for Improvement of fitness (LEIOF). Some objectives are conflicting in nature and do not have the equal importance. So, grading range for different objective will be different and also have the different meaning. The test cases are eligible for qualifying certificate of fitness, if they obtain minimum common grade (D Grade) on each parameter or objective and final grade also. Though there are several parameters and objectives for estimating the fitness of the test cases but the author evaluated the test cases from such perspectives as number of defect detecting capability, testing cost, testing efforts, control and data flow based adequacy criteria details in Table 1.

3.1 Objectives and Adequacy Criteria for Test Cases Fitness Evaluation

The test problem specification includes three main parts, the purpose of testing, test coverage criteria and the test strategy that will be employed. First step in multi-faceted measurement framework is to identify the fitness parameters and objectives of classification and selection of test cases. Software test adequacy criteria are the rules to determine whether a software system has been adequately tested, which points out the central problem of software testing i.e. “What is a test data adequacy criterion?”. Number of test data adequacy criteria has been proposed and investigated in the literature like control flow based adequacy criteria, data flow based adequacy criteria, fault-based adequacy criteria, and error-based criteria. Control flow based adequacy criteria include statement coverage, branch coverage, path coverage, Length-i path coverage, loop coverage, relational operator coverage, and table coverage. Data flow based adequacy criteria include all definitions criterion, all uses criterion. Fault based adequacy criteria include error seeding and mutant coverage or mutant killing score [13,14]. Execution time is one realistic measure of effort. Selection and execution time are the important factors for test cases classification [15,16]. Physical execution time of test cases is hard to measure accurately. Measurement is confounded by many external factors; different hardware, application software and operating system. The fitness of the test cases is not only concern but cost is also one of the apprehensions of software industry, researcher and academicians. The whole purpose of test case classification and selection is to achieve more efficient testing in terms of the cost. Review of existing literatures has brought out several parameters for assessing the fitness and objectives of test case classification like maximum number of defect detecting capability, minimum test cases design efforts, minimum design cost, minimum execution cost, maximum

Table 1. Software test cases fitness valuation index system.

	Parameters/First Layer Index	Sub-Parameters/Second Layer Index
Software Test Cases Fitness Evaluation Parameters	Fault Detecting Capability X1(0.2)	Error Seeding X11 (0.35)
		Mutant Killing ability(Score) X12(0.35)
		Fault Severity X13 (0.30)
	Control Flow Based Adequacy X1(0.2)	Statement Coverage X21 (0.23)
		Branch Coverage X22 (0.23)
		Path Coverage X23 (0.20)
		Loop Coverage X24 (0.14)
		Relational Operator Coverage X25 (0.11)
		Table Coverage(Array) X26 (.09)
	Data Flow Based Adequacy X3(0.15)	All Definition Criteria X31 (0.50)
		All Uses Criteria X32 (0.50)
	Efforts X4(0.15)	Total Efforts X41 (0.15)
		Design Efforts X42 (0.15)
		Selection Efforts X43(0.3)
		Execution Efforts X44 (0.2)
		Efforts Benefits X45 (0.2)
	Cost X5 (0.15)	Total Cost X51 (0.15)
		Design Cost X52 (0.15)
		Selection Cost X53 (0.3)
		Execution Cost X54 (0.2)
		Cost Benefits/Cost Saving X55 (0.2)
	Requirement Coverage Capability X6(0.15)	Critical Requirement Coverage X61 (0.36)
		Rare Requirement Coverage X62(0.19)
		Least Requirement Coverage X63 (0.45)

coveragebility of client requirements, maximum code coveragebility, minimum setup and data access cost, execution time/ effort, and maximum mutant killing score, etc are objectives and contributing as parameters for assessing the fitness of test cases. These parameters are not contributing at same level for assessing fitness of test case. These parameters values /scores may be maximum /minimum according to the test objectives [7].

3.2 Weight Assigning Method

The weight is a statistical measure used to evaluate how important a parameter or objective is to a test case in a collection or test cases repository. Second step in multi-faceted measurement framework is to identify the importance of fitness parameters and objectives of classification and selection of test cases. Thereafter weight distribution calculation strategy will be employed. Test cases fitness parameters and testing objectives have been identified in previous section. Some objectives are conflicting in nature. Contribution of test case parameters towards their fitness is vague and imprecise. Importance of testing objectives is also fuzzy. So, concurrent consideration of all fuzzy parameters and testing objectives creates uncertainty. So, there is a need to devise fuzzy feature weighting approach. All fitness parameters of test cases are not equally important for each project. They are not contributing equally to fitness of test cases. Weight values for fitness parameters are calculated by using Feature Weighting Functions (FWF). A weighting function is used to associate a weight $w_{i,j}$ to each fitness parameter (factor) P_j of test cases. The classical weighting function is used for relevance computation in testing objectives. FWF, whose details can be found in Ref. 17,18, leads us to the following:

Definition 1. Let $w_{i,k}$ denote a weight of k-th fitness parameter P_k with respect the i-th module/component and $n_{i,k}$ is the frequency of the k-th parameter P_k in i-th component/ module. N is the size of Test Cases Repository(TCR), f is the total number of parameters/ objectives and n_k is the number of TCR items showing P_k, then $w_{i,k}$ is as follows :

$$\mathbf{w_{i,k}} = \frac{n_{i,k} * \left(\log\frac{N}{n_k}\right)}{\sqrt{\left(\sum_{j=1}^{f} (n_{i,j})^2 * \left(\log\frac{N}{n_j}\right)^2\right)}} \tag{1}$$

Refinement of feature weight is the fuzzy weight, whose characteristic function $w_{i,k,j}$ is called the fuzzy weight of k-th fitness parameter P_k with respect the i-th module/components of j-th project. It should be defuzzified by using Ordered Weighting Techniques, which require enhanced version of the fuzzy weight. The values of $w_{i,k,j}$ are ordered in a decreasing manner then component/module weight cw_i, is calculated.

Definition 2. Let cw_i denote the component/ module weight of a fitness parameter P_k which depends on the ratio between the number of steps of test case i where P_k occurs and the total number of occurrences of P_k in the components/module test cases's steps. as follows:

$$\mathbf{cw_i} = \sum_{i=1}^{i=n} \frac{\mathbf{Step_i, P_k}}{\mathbf{Step\ P_k}} \tag{2}$$

Where the $step_i$,P_k is the number of test steps and Pk is relevant in the ith test case. Decision maker will assign importance to each module by supplying auxiliary values $\{a_1, a_2, a_3, a_4, \ldots\ldots a_n\}$, $0 \le a_i \le 1$, default value is 0. Component weight and auxiliary weights are used to calculate the effective fuzzy weight ϵi, as follows:

$$\epsilon_i = \{\max\{a_i, 1 - \alpha\}\} * w_i \qquad (3)$$

Where α is defined as:

$$\alpha = \frac{1}{n-1}\sum_{i=1}^{n}(n-i) * cw_i \qquad (4)$$

Lastly, we will estimate defuzzified weight ∈ as follows:

$$\in = \sum_{i=1}^{n} \tilde{\epsilon}_i * CW_i \qquad (5)$$

Where $\tilde{\epsilon}$ are in decreasing manner values of $\in_i$

3.3 Fuzzy Synthetic Evaluation Approach

The term synthesis is used here to predict the fitness of test cases and then classify the test cases into broad class EQCF and EIOF. Several individual elements are evaluated and components of an evaluation system are synthesized into an aggregate form. The evaluation is usually described in natural language terms, since a numerical evaluation is often too complex, too unacceptable, too ephemeral and too vague. We used the index system of software test cases fitness evaluation built in Table 1 to carry out measurement. Obviously, software test cases fitness evaluation belongs to multilayer fuzzy synthesis evaluation, idiographic steps for carrying out as follows:

3.3.1 Ascertain valuation set:

Y= {Excellent, Better, Good, Common, Bad, Worse, Worst}

3.3.2 Ascertain evaluation factor:

We used the index item of the test cases evaluation index system for fitness evaluation and classification built in Table 1. There are two layers, the names and serial numbers of each layer index item have given out. Fuzzy synthesis evaluation factor set is composed by decompounding and subdividing the subjection relation of each layer index item layer by layer, for example, the evaluation factor set of the first layer index is {X1, X2, X3, X4, X5, X6}. The evaluation factor set of X1 is {X11, X12, X13}, and the evaluation factor set of other layer index can be got from Table 1 by using this method.

3.3.3. Ascertain the weight of each layer evaluation index:

Here weight of each index item is calculated by fuzzy feature weighting function and classical weighting approach, and discussed in subsection 3.2. Weight coefficients are given in Table 1.

3.3.4 Ascertain fuzzy evaluation matrix:

According to multilayer fuzzy synthesis evaluation process, first we need to ascertain the fuzzy evaluation matrix, which is correlative with the two layer index items of software test cases fitness evaluation index system (Table (1)). Fuzzy evaluation matrix values are the membership degree values of each factor (comment) in comment set Y, which is correlative with each index item. Ref. 19 leads us to adopt the factor evaluation algorithm researched by USA RADC (Rome Air Development Center). Factor evaluation algorithm is used to measure the second layer index items separately. These values are just distributed in interval (0,1).Taking 0 means worst and taking 1 means excellent. Limiting to length, we just list out X11= Number of predefined (seeded) errors detected /Total number of errors seeded, X12 = k/ (g-n-q), where k, g, n, q are the numbers of killed, all generated mutants, anomalous and mutants equivalent to original program. X13= Number of severe faults detected /Total number of severe faults. The measurement of other second layer indexes can refer to testing metrics defined, and we doesn't give necessary details here. The second layer index measurement value is only holistic comment of software test cases fitness. It is not acting as membership degree value of different comment y_i in the valuation set Y. How do we take different membership degree values for the different comment y_i in the valuation set Y?. According to experience measured software test cases fitness parameters and second layer indexes for the valuation set is, as follows:

$$\mu(x) = e^{-((x-m)|c))^2} \quad (6)$$

Where, m and c are constant. Note formula (6), As x=m, μ(m)=1 is maximum. So, m must be the middle point of the interval in Table 2. For example, the interval for the index X11 corresponding comment y1 (excellent) is (0.85,1), and then m=(0.85+1)/2=0.925. The all interval middle point's m

Table 2. Second layer indexes taking value intervals.

Second Layer Indexes	Excellent	Better	Good	Common	Bad	Worse	Worst
X11	(0.85,1)	(0.75, 0.85)	(0.60, 0.75)	(0.50, 0.60)	(0.40, 0.50)	(0.30, 0.40)	(0.00, 0.3)
X12	(0.86,1)	(0.76, 0.86)	(0.61, 0.76)	(0.51, 0.61)	(0.41, 0.51)	(0.31, 0.41)	(0.00, 0.31)
X13	(0.84,1)	(0.74, 0.84)	(0.59, 0.74)	(0.49, 0.59)	(0.35, 0.49)	(0.20, 0.35)	(0.00, 0.20)

Table 3. Second layer indexes normal distribution parameters.

Second Layer Indexes	Excellent	Better	Good	Common	Bad	Worse	Worst
	m, c						
X11	(0.93,0.09)	(0.80, 0.06)	(0.68, 0.09)	(0.58, 0.09)	(0.45, 0.06)	(0.35, 0.06)	(0.15, 0.18)
X12	(0.93,0.08)	(0.81, 0.06)	(0.69, 0.09)	(0.56, 0.06)	(0.46, 0.06)	(0.36, 0.06)	(0.16, 0.19)
X13	(0.92,0.10)	(0.79, 0.06)	(0.67, 0.09)	(0.54, 0.06)	(0.42, 0.08)	(0.28, 0.09)	(0.10, 0.12)

sub-parameters, we firstly gave second layer index taking value intervals for different comment y_i, shown as Table 2.

Generally speaking, the membership degree values of the second layer indexes (software metric elements) for different comment y_i in valuation set Y are all normal distribution in different intervals shown in Table 2. So, the membership function of the and c are shown in Table 3.

Furthermore, as x is a boundary point of the tow neighbor intervals for same second layer index taking value, and then membership degree values is same for the two intervals corresponding the two comments, and u(x=boundary point) about equal 0.5. Therefore, the constant c is the root of equation $e^{-\{((x_r-x_l)|2c)\}^2} = 0.5$ in which x_r and x_l are separately the right endpoint and left endpoint of the intervals in Table 2. Thus, all m and c constant values in all intervals are computed and shown in Table3.Subsequently, the measurement values of all second layer indexes are calculated with RADC measurement algorithm. According to these values, the Table 3 and the formula (6), we can compute the fuzzy membership degree values of the second class indexes (in the Table 1) corresponding different comment y_i (in the set Y) and in different intervals (in the Table 2). As an example, we only gave the fuzzy membership degree values for the index X11, X12 and X13 corresponding comment y_i, and these values are shown in Table 4.

Table 4. Second layer indexes membership degree.

Second Layer Indexes$=r_{i,j,k}$	Excellent	Better	Good	Common	Bad	Worse	Worst
	μ(x)						
X11=0.73	0.01	0.26	0.69	0.00	0.00	0.00	0.00
X12=0.89	0.80	0.17	0.01	0.00	0.00	0.00	0.00
X13=0.58	0.00	0.00	0.41	0.64	0.03	0.00	0.00

Thereupon, the fuzzy valuation matrixes of the second layer indexes are built by the membership degree values. For instance, the fuzzy valuation matrix which is related the indexes X11, X12 and X13 corresponding with the first layer index X1 is (Table 4). Of course, we can also get all fuzzy evaluation matrixes for the all second layer indexes in Table 1 with the same method.

$$R11=\begin{bmatrix}0.01 & 0.26 & 0.69 & 0.00 & 0.00 & 0.00 & 0.00\\ 0.80 & 0.17 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00\\ 0.00 & 0.00 & 0.41 & 0.64 & 0.03 & 0.00 & 0.00\end{bmatrix}$$

3.3.5 Synthesis evaluation:

According to the improved model of fuzzy synthesis evaluation algorithm, we begin with the corresponding the fuzzy evaluation matrixes (as R11) of the second layer indexes, and compute the corresponding evaluation values of the first layer indexes in turn. It is the membership degree for evaluation set Y of the first layer index. For example, the fuzzy evaluation matrix for the first layer index x1 can be calculated from the below formula (7):

$$R1 = A11\emptyset R11 \quad (7)$$

$$R1=(0.35, 0.35, 0.30)\begin{bmatrix}0.01 & 0.26 & 0.69 & 0.00 & 0.00 & 0.00 & 0.00\\ 0.80 & 0.17 & 0.01 & 0.00 & 0.00 & 0.00 & 0.00\\ 0.00 & 0.00 & 0.41 & 0.64 & 0.03 & 0.00 & 0.00\end{bmatrix}$$

$$=(0.28,\ 0.15,\ 0.37,\ 0.19,\ 0.01,\ 0.00,\ 0.00)$$

With above computing method, the fuzzy evaluation matrix of the first layer indexes are all ascertained by merging interrelated fuzzy valuation matrixes of the second layer indexes and associated weight metrics via the unitary management. It is the membership degree for evaluation set Y of the first layer indexes, and so the fuzzy evaluation matrix of the first layer indexes is formed. Finally, we get the fuzzy synthesis evaluation value of the evaluated test cases. In order to measure and describe the final evaluation result of the test cases well and truly, we have to adapt the measurement practice of daily evaluation. We adjust the final fuzzy synthesis evaluation value, and distributing them in the interval [0, 100]. The test cases fitness evaluation need every comment in the evaluation set. We divide the interval [0, 100] into seven subintervals which correspond to the seven comments. Finally, we give the corresponding evaluation according to the subinterval which the adjusted value of fuzzy synthesis evaluation value is in.

4. Conclusion and Future Scope

Fuzzy logic based multi-faceted measurement framework is a next generation mathematical model and more suitable for modern software testing requirements. Proposed framework is very useful and effective to software test cases optimization and fitness evaluation problem. It will certainly improve the accuracy, practicability, reliability and flexibility of software test cases fitness evaluation and classification approach. Multi-faceted measurement framework will potentially provide significant benefits to software testing. Proposed framework will also reduce the efforts, cost and uncertainty of software testing. Proposed framework also classifies the test cases and helps testers to select efficient test cases from various alternative classes of fit test cases.

Proposed framework uses an improved fuzzy synthesis evaluation algorithm, which has a good rate, accuracy and credibility for software test cases fitness assessment and classification. It is evident that the proposed framework conducts not only the overall evaluation of the fitness of software test cases, but also can do sub-item or sub-parameters assessment for different levels of evaluation factors. This flexible measurement framework is very useful in actual test cases fitness evaluation, multi-objective classification and selection work. However, implementation of proposed framework is due. Unfit test cases are ignored in proposed framework. These outliers are very important. Fitness of these outliers can be improved and can be converted into fit category. These test cases may outperform in future. Framework for fitness improvement of unfit test cases and conversion of unfit test cases to fit cases is pending. Conversion framework will be proposed in future.

References

[1] Alberts D.S., The economics of software, Proceedings of National Computer Conference on Quality Assurance, held at Montvale, N.J., 1976; **45:** 433-442.

[2] Berndt D.J. and Watkins A., High volume software testing using genetic algorithms, Proceedings of 38th IEEE International Conference on System Sciences, Hawaii, 2005; **8(5):** 1-5.

[3] Yoo S. and Harman M., Pareto efficient multi-objective test case selection, Proceedings of the International Symposium on Software Testing and Analysis (ISSTA), held at London, U.K., 2007: 140-150.

[4] Mala D.J., Mohan V., Quality improvement and optimization of test cases – a hybrid genetic algorithm based approach, *ACM SIGSOFT Software Eng. Notes*, 2010; **35(3):** 1-14.

[5] Shihab A.H., Framework for Intelligent Meaningful Test Data Generation Model-IMTDG, WOO, 2000: 11-16.

[6] Kumar M., Sharma A. and Kumar R., Towards multi-faceted test cases optimization, *J. Software Eng. Appl.*, 2011; **4(9):** 550-557.

[7] Kumar M., Sharma A. and Kumar R., Soft computing-based software test cases optimization: A survey, *Int. Rev. Comp. Software*, 2011; **6(4):** 512-526.

[8] Mohanty R., Ravi V. and Patra M.R., The application of intelligent and soft-computing techniques to software engineering problems: A review, *Int. J. Information and Decision Sci*., 2010; **2(3)**: 233-272.

[9] Mala D.J. and Mohan V., ABC tester - artificial bee colony based software test suite optimization approach, *Int. J. Software Eng.*, 2009; **2(2):** 15-43.

[10] Yogesh S., Kaur A. and Suri B., Test case prioritization using ant colony optimization, *ACM SIGSOFT Software Eng. Notes,* 2010; **35(4):** 1-7.

[11] Ross Timothy J., Fuzzy Logic with Engineering Applications, second edition, John Willey & Sons, 2004.

[12] Law C-K., Using fuzzy numbers in educational grading system, *Fuzzy Sets and Systems*, 1996; **83**: 311-323.

[13] Zhu H., Axiomatic assessment of control flow-based software test adequacy criteria, *Software Eng. J.*, September 1995; **10(5)**: 194-204.

[14] Kim J.M. and Porter A., A history-based test prioritization technique for regression testing in resource constrained environments, Proceedings of the 24th International Conference on Software Engineering, held at Orlando, FL, 2002: 119-129.

[15] Kumar V. and Kumar S.M., Test case prioritization using fault severity, *Int. J. Comp. Sci. Technol*., 2010; **1(1):** 67-71.

[16] Leon D. and Podgurski A., A comparison of coverage-based and distribution-based techniques for filtering and prioritizing test cases, Proceedings of the 14th International IEEE Symposium on Software Reliability Engineering (ISSRE'03), 2003: 442-453.

[17] Salton G. and Buckley C., Term weighting approaches in automatic text retrieval, *Inf. Proc. Manage*., 1998; **24(5)**: 513-523.

[18] Yager R.R., On ordered weighted averaging aggregation operators in multi-criteria decision making, *IEEE Trans. Systems, Man and Cybernetics*, 1998; **18(1)**: 183-190.

[19] Jianli D. and Ningguo S., Research and improvement of the fuzzy synthesis evaluation algorithm based on software quality, *Comp. Eng. Sci*., 2007; **29(1):** 66-69.