IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 11, November 2019

A Software Based Mathematical Approach for the Selection of IT Industries to the Geographical Map

Sahidul Haque[1] and Sahida Sultana[2]

Geographical Scientist Section, University Kolkata, Scientist of Geographical Map and Technology,

Jamia Millia Islamia (A Central University), New Delhi, India[1]

M. Tech Scholar, Department of Computer Science and Engineering, Faculty of Engineering and Technology,

AL-Falah University, Dhuj, Faridabad, Haryana, India[2]

Abstract: Geographical science survey says as lots of countries (Bihar & Patna, Jammu Kashmir etc) unable to bulid software industries to the geographical map due to governmental scarcity of management ability to the technical education, governmental scarcity of technical ability to the technical education . to overcome the above problems , we have selected some talented brain from the listed countries "Kolkata, English Bazar and Delhi" and approved green color mark countries for IT industries to the geographical map due outstanding performance to the technical background. The success rate of software system depends upon the following: requirements elicitation technique, modeling, analysis, verification, validation &testing. In literature, we have identified different types of Software Testing Techniques like, black box techniques, white box techniques, and gray box techniques; and choosing one of them is not an easy task according to need/criteria of the software projects. Therefore, in order to address this issue, we present a fuzzy based approach for the selection of Software Testing Techniques. Finally, the utilization of the proposed approach is demonstrated with the help of an example.

Keywords: Requirement Prioritization, Fuzzy Set theory, Decision Making Process, S/W Testing Technique

I. INTRODUCTION

Software testing identifies defect, flows or errors in the software. In literature, we have identified various definitions of software testing. Few of them are given below: (i) testing is the process of demonstrating that errors are not present (ii) The purpose of testing is to show that a program performs its intended functions correctly. The three most important techniques that are used for finding errors are functional testing, structural testing and gray box testing [6,7]. Functional testing is also referred to as black box testing in which contents of the black box are not known. Functionality of the black box is understood on the basis of the inputs and outputs in software. There are different methods which are used in black box testing methods like boundary value analysis, robustness testing, equivalence class partitioning, and decision table testing. White box testing or structural testing is the complementary approach of functional testing or black box testing. White box testing permits us to examine the internal structure of the program. In functional testing all specifications are checked against the implementation. This type of testing includes path testing, data flow testing, and mutation testing. In white box testing there are various applications of graph theory which is used to identify the independent path in a program or software like decision to decision (DD) flow graph, Cyclomatic complexity [6] etc.

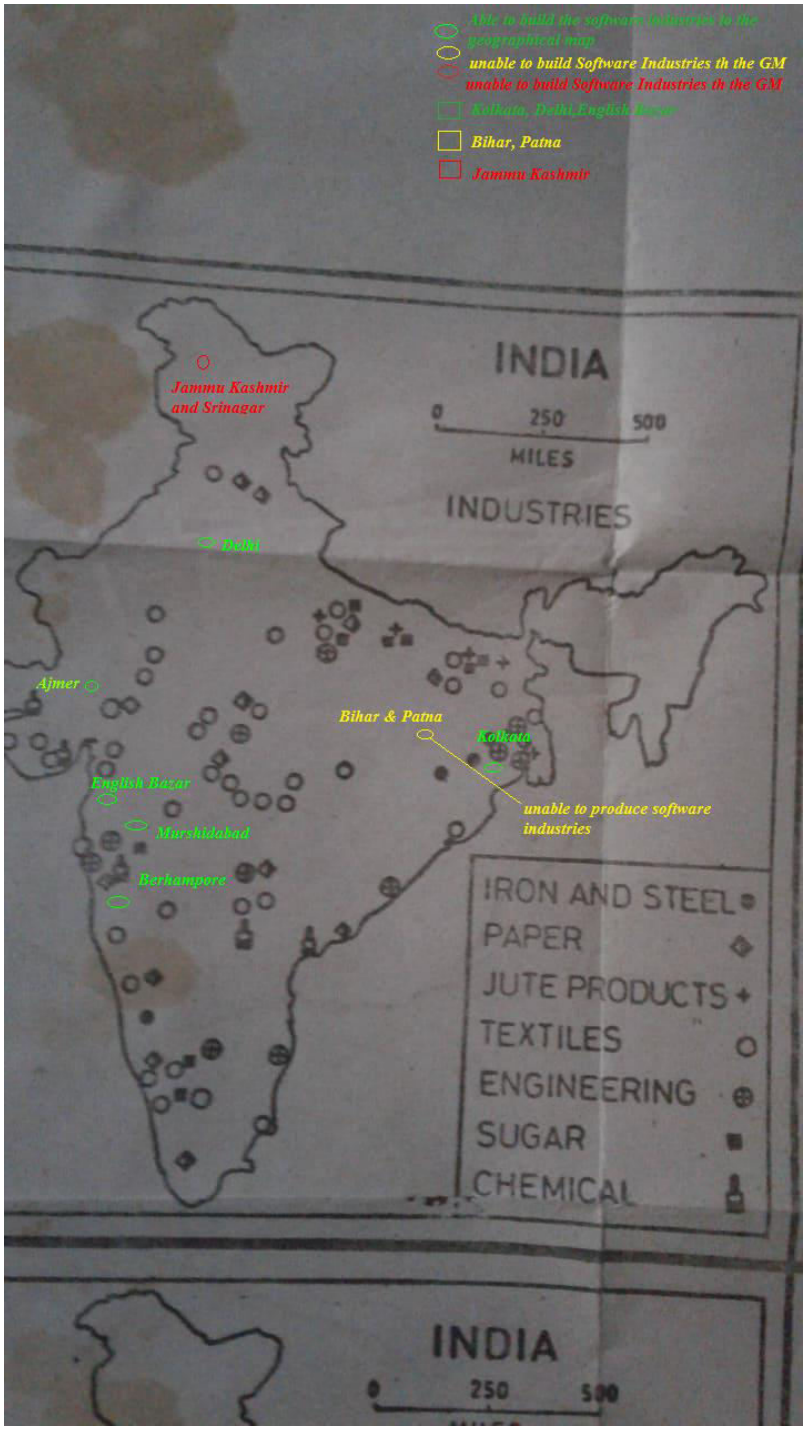
Gray box testing is the testing of software application using effective combination of white box testing, black box testing, mutation, and regression testing [2]. This testing provides a method of testing software that will be both easy to implement and understand using commercial of the shelf (COTS) software [1]. In the Gray box testing, tester is usually has knowledge of limited access of code and based on this knowledge the test cases are designed; and the software application under test treat as a black box & tester test the application from outside. Gray box software testing methodology is a ten steps process for testing computer software. The methodology starts by identifying all the inputs and output requirements to computers systems. This information is captured in the software requirements documentation. The steps are given as follows: (i) Identify inputs (ii) Identify outputs (iii) Identify major paths (iv) Identify sub-function (SF) X (v) Develop inputs for SF X (vi) Develop outputs for SF X (vii) Execute test cases for SF X (viii) Verify correct results for SF X (ix) Repeat steps from 4 to 8 for other SF X and (x) Repeat steps 7 to 8 for regression [1].

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 11, November 2019



II. LITERATURE REVIEW

Most of the work in literature is based on either black box testing or white box testing for example, in 2012; Khan, Bhatia, and Sadiq [8] develop a BBTool to generate the tests cases using black box testing. In a similar study, in 2011, Khan and Sadiq [7] analyze the various black box testing techniques. In literature, authors are trying to integrate the concepts of genetic algorithms with testing, for example, In 2011 Sabharwal et al. [9] proposed a technique for optimizing static testing efficiency by identifying the critical path clusters using genetic algorithm. The testing efficiency is optimized

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 8, Issue 11, November 2019

by applying the genetic algorithm on the test data. The test case scenarios are derived from the source code. The information flow metric is adopted in this work for calculating the information flow complexity associated with each node of the control flow graph generated from the source code. In 2009, Mohapatra et al. [5] used genetic algorithm to optimize the test cases that are generated using the category-partition and test harness patterns. In a similar study, Vieira et al. [11] proposed a GUI Testing Using a Model-driven Approach. The authors demonstrated and evaluated their method based on use cases that was developed for testing a graphical user interface (GUI).

Huang et al. [3] proposed repairing GUI test suites using a genetic algorithm. In this paper they develop a method to automatically repair GUI test suites, generating new test cases that are feasible. They use a genetic algorithm to evolve new test cases that increase our test suite's coverage while avoiding infeasible sequences. In 2007, Memon et al. [4] proposed an event flow model of GUI-based applications for testing. This paper consolidates all of the models into one scalable event-flow model and outlines algorithms to semi-automatically reverse-engineer the model from an implementation. Earlier work on model-based test-case generation, test-oracle creation, coverage evaluation, and regression testing is recast in terms of this model by defining event-space exploration strategies (ESESs) and creating an end-to-end GUI testing process. Three such ESESs are described: for checking the event-flow model, test-case generation, and test- oracle creation.

III. FUZZY SET THEORY

In this section, we briefly review the basic concepts of fuzzy sets, linguistic variable, fuzzy triangular numbers, and fuzzy preference relation. The fuzzy set, originally proposed by Zadeh in 1965 [31], is defined as follows: In a universe of discourse Ux, a fuzzy subset A of Ux is characterized by a membership function f A(x), where fA : Ux → [0, 1] and the membership function associates with each member of x of Ux a number of f A(x) in the interval [0,1], representing the grade of membership of x in A. Linguistic variables are variables whose values are words or sentences in a natural or artificial language [31, 32]. For example, poor is a linguistic variable if its values are assumed to be the fuzzy variables labelled very poor, poor, fair, good, and very good; rather than the numbers 0,1,2,3 etc.

There are several formats of fuzzy numbers, such as Triangular, Trapezoidal, Gaussian, or Sigmoid that can be used in decision making processes. In practical applications, triangular fuzzy numbers (TFNs) are widely used to represent the approximate value range of linguistic variables [26]. In the proposed method we adopt TFNs because of their simplicity in both concepts and computation [19, 26, 27]. TFNs can be defined as follows:

Let R is the real line, which is viewed as a universal set of all fuzzy sub-sets. A triangular fuzzy number A is normal, convex fuzzy subset of R, with a piece wise linear relationship function μA, defined by:

$$\mu_A(x) = \begin{cases} \frac{(x-a)}{(b-a)}, a \le x \le b, \\ \frac{(c-x)}{(c-b)} b \le x \le c, \\ 0, \quad otherwise \end{cases} \qquad (1)$$

The TFNs can be denoted by A = (a, b, c) as depicted in Fig. 1. The parameters a, b, and c respectively, indicate the smallest possible value, the most promising value, and the largest possible value that describe a fuzzy event. There are several operations that can be performed on triangular fuzzy numbers (TFN) like addition, subtraction, inverse etc.

Let $A_1 = (a_1, b_1, c_1)$ and $A_2 = (a_2, b_2, c_2)$ then:

Addition: $A_1 \oplus A_2 = (a_1+a_2, b_1+b_2, c_1+c_2)$ (2)

Subtraction: $A_1 \ominus A_2 = (a_1-c_2, b_1-b_2, c_1-a_2)$ (3)

Multiplication: $A_1 \odot A_2 = (a_1.a_2, b_1.b_2, c_1.c_2)$ (4)

Inverse: $(A_1)^{-1} = (1/c_1, 1/b_1, 1/a_1)$ (5)

Negation of $A_1 = (-c_1, -b_1, -a_1)$ (6)

Division: $A_1 / A_2 = (a_1/c_2, b_1/b_2, c_1/a_2)$ (7)

Preference relation is a useful tool for representation of information used in decision making problems. It is used when we want to aggregate expert's preferences into group preferences. A fuzzy preference relation P on **R** is a fuzzy subset of **R** x **R** with membership function $f_P(A, B), \forall A, B \subseteq R$, where $f_P(A, B)$ represents the degree of preference of A over B [12]:

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 11, November 2019

1. P is reciprocal iff $f_P(A, B) = 1 - f_P(B, A)$, $A, B \subseteq R$.
2. P is transitive iff $f_P(A, B) \geq 1/2$ and $f_P(B, C) \geq 1/2 \rightarrow f_P(A, C) \geq 1/2$, $\forall A, B, C \subseteq R$.
3. P is a fuzzy total ordering iff P is reciprocal, transitive, and comparable.

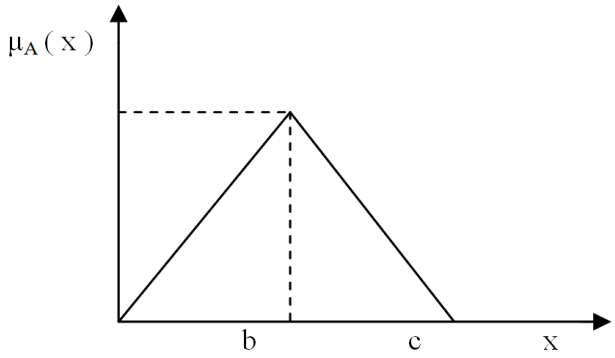


Fig. 1. The membership function of a TFN A = (a, b, c)

IV. PROPOSED METHOD

This Section presents fuzzy based approach for selection of software testing technique using fuzzy set. The proposed method is presented in the following:

1. Identify the criteria
2. Identify functional and non functional requirements.
3. Selection of a Software Testing Techniques
4. Collect Decision making fuzzy assessment to establish the relationship between FR and NFR.
5. Construct comprehensive performance and weight matrix and then apply the following step: (3.1) aggregate fuzzy performance rating with fuzzy weights (3.2) define each sub-goal/requirements as a fuzzy number (3.3) define extended average(EA) (3.4) define the extended difference (3.5) calculate the ranking values (rv) for each requirements
6. Apply Binary tree sort method on rv of the requirements to get the prioritized list of requirements.

Step1 Identify the criteria

Before the selection of any Software Testing Techniques, tester should identify the criteria's the selection of an Software Testing Techniques. On the basis of our literature review, we have identified the following factors which influence the decision of choosing a software testing methodology:

(a) New or existing software
(b) Cost of requirements
(c) Independent path

Step2 Collect decision maker's fuzzy assessment

In this step, expert's opinions regarding the importance of each requirement are obtained in the form of linguistic variable such as, very good, good, medium etc.In this step ,we collect the experts' fuzzy assessments and express their opinions on the importance of each requirement.

Step3 Compute fuzzy group preference from the fuzzy individual preferences

For the prioritization of requirements on the basis of various criteria's, we aggregate fuzzy performance rating through all decision maker by means of extended addition and scalar multiplication to form a comprehensive performance matrix P, in which performance rating:

Pij= (1/n) Θ $(Pij^1 \oplus Pij^2 + \cdots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots, \oplus Pij^n$

Is a triangular fuzzy number of the form:

(P1ij,P2ij,P3ij) = $(1/n \sum_{k=1}^{n} p1ij^k, 1/n \sum_{k=1}^{n} p2ij^k, 1/n \sum_{k=1}^{n} p3ij^k)$

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 8, Issue 11, November 2019

Now Calculate the fuzzy weight through all DM by means of extended addition and scalar multiplication to form a comprehensive WV. Once we have obtained the comprehensive performance and weight matrix the apply the following steps(Li 1999):

Step 3.1 Aggregate fuzzy ratings with fuzzy weights by means of extended multiplication to form a weighted, comprehensive decision matrix D, in which
dij = pij Θ wj
is a fuzzy number with parabolic membership functions in the form of:

(∂1ij, ∂2ij, ∂3ij|dij|Δ1ij, Δ2ij, Δ3ij)
∂1ij=(w2j-w1j)(p2j-p1j)
∂2ij=w1j(p2ij-p1ij)+p1ij(w2ij-w1ij)
∂3ij=WijPij
Δij=(W3j-W2j)(P3j-P2j)
Δ2j=W3j=(P3ij-P2ij)+P3ij(W3j-W2j)
Δ3j=W3jP3j
and
dj=W2jP2ij

Step 3.2 Define each sub-goal/requirement as a fuzzy number Ai, i=1,2,……. M by means of extended addition and scalar multiplication through the following criteria:
Ā=1/m Θ (A1⊕A2⊕A3+……⊕Am)
With probalitic membership function in the form of
(∂1, ∂2, ∂3| Ā|Δ1, Δ2, Δ3) where
$\partial 1=1/m\sum_{i=1}^{m} \partial 1i \quad ,i = 1,2,3$
$\Delta 1=1/m\sum_{i=1}^{m} \Delta 1i \quad ,i = 1,2,3$
$\Delta=1/m \sum_{i=1}^{m} \bar{A}i \quad ,i = 1,2,3$
And
$EAi=1/m(\sum_{j=1}^{m} Aij$

Step 3.3 Define EA means of extended addition and scalar multiplication through all alternatives (sub-goals/requirements).
EA=1/n Θ (g1⊕g2⊕,……⊕gh)
With probalitic membership function in the form of
(∂1, ∂2, ∂3| Sum_EA|Δ1, Δ2, Δ3) where
$\partial 1=1/m\sum_{i=1}^{m} \partial 1i \quad ,i = 1,2,3$
$\Delta 1=1/m\sum_{i=1}^{m} \Delta 1i \quad ,i = 1,2,3$
$\Delta=1/m \sum_{i=1}^{m} \bar{A}i \quad ,i = 1,2,3$
And
$Sum\_EA=1/n(\sum_{j=1}^{n} EAi$

Step 3.4 Define the extended difference, EAi Θ Sum_EA, for each Ai ∈ R, with parabolic membership function in the form of :

$$(\delta_1 i - \Delta_1) \ , (\delta_2 i + \Delta_2) , (\delta_3 i - \Delta_3) | EAi - Sum_{\_EA} | (\Delta_1 i - \delta_1) \ , (-\Delta_2 i - \delta_2) , (\Delta_3 i - \delta_3)$$

Step 3.5 Calculate rv of each requirements
In this step, we calculate the ranking values (rvi) for each requirements Ai by means of F-preference relation R:

if $(\Delta_3 i - \delta_3) < 0, (\Delta_3 i - \delta_3) \geq 0, EAi \geq Sum_{EAi}$; then
Rvi= μR(Ai Θ EA,0)= $\beta^+ / \ \beta^+ + \ \beta^-$
Else if $(\Delta_3 i - \delta_3) \leq 0, (\Delta_3 i - \delta_3) > 0, EAi \leq Sum_{EAi}$ then
Rvi= μR(Ai Θ EA,0)= $\gamma^+ / \gamma^+ + \gamma^+$
Else if $(\Delta_3 i - \delta_3) = 0, (\Delta_3 i - \delta_3) = 0, EAi = Sum_{EAi}$ then

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 11, November 2019

Rvi= μR(Ai Θ EA,0)=0.5;
Else if $(\Delta_3 i - \delta_3) \geq 0, (\Delta_3 i - \delta_3) > 0, EAi \geq Sum_{EAi}$
Rvi= μR(Ai Θ EA,0)=1;
Else if $(\Delta_3 i - \delta_3) < 0, (\Delta_3 i - \delta_3) \leq 0, EAi \leq Sum_{EAi}$ then
Rvi= μR(Ai Θ EA,0)=0.

where

$$\beta^{+} = \begin{bmatrix} \frac{1}{4}(\Delta_1 i - \delta_1) - \frac{1}{3}(-\Delta_2 i - \delta_2) + \frac{1}{2}(\Delta_3 i - \delta_3) \\ + \frac{1}{4}(\delta_1 i - \Delta_1)(1 - {\mu_3}^4) + \frac{1}{3}(\delta_2 i + \Delta_2)(1 - {\mu_3}^3) \\ + \frac{1}{2}(\delta_3 i - \Delta_3)(1 - {\mu_3}^2) \end{bmatrix}$$

$$\beta^{-=} \left[\frac{1}{4}(\delta_1 i - \Delta_1){\mu_3}^4 + \frac{1}{3}(\delta_2 i + \Delta_2){\mu_3}^3 + \frac{1}{2}(\delta_3 i - \Delta_3){\mu_3}^2\right]$$

$$\mu_1 = \frac{-(\delta_2 i + \Delta_2) + \sqrt{(\delta_2 i + \Delta_2)^2 - 4(\delta_1 i - \Delta_1)(\delta_3 i - \Delta_3)}}{2(\delta_1 i - \Delta_1)}$$

$$\gamma^{+} = \left[\frac{1}{4}(\Delta_1 i - \delta_1){\mu_2}^4 + \frac{1}{3}(-\Delta_2 i - \delta_2){\mu_2}^3 + \frac{1}{2}(\Delta_3 i - \delta_3){\mu_2}^2\right]$$

$$\gamma^{-} = -\begin{bmatrix} \frac{1}{4}(\delta_1 i - \Delta_1) + \frac{1}{3}(\delta_2 i + \Delta_2) + \frac{1}{2}(\delta_3 i - \Delta_3) \\ -\frac{1}{4}(\Delta_1 i - \delta_1)(1 - {\mu_2}^4) - \frac{1}{3}(\Delta_2 i + \delta_2)(1 - {\mu_2}^3) \\ + \frac{1}{2}(\Delta_3 i - \delta_3)(1 - {\mu_2}^2) \end{bmatrix}$$

$$\mu_2 = \frac{(\Delta_2 i + \delta_2) - \sqrt{(-\Delta_2 i - \delta_2)^2 - 4(\Delta_1 i - \delta_1)(\Delta_3 i - \delta_3)}}{2(\Delta_1 i - \delta_1)}$$

Step 4: Binary tree sort method
In this step, we create the BST of the rv that we have obtained in previous step and then the tree is traversed in IN-ORDER the IN-ORDER traversal of BST lists the elements in ascending order.the algorithm to traverse a non empty binary tree in IN-ORDER is given below (Aho et al. 1983)

(a) Traverse the left sub-tree in IN-ORDER.
(b) Visit the root node.
(c) Traverse the right sub-tree in IN-ORDER.

V. IMPLEMENTATION

This Section Presents a Case Study for the Selection of the software technique using fuzzy based approach.

Table1 Comprehensive performance matrix

Model	C_1	C_2	C_3
BB	(5.2,7.2,8.8)	(6.8,8.8,10)	(4.8,6.4,8)
GB	(5.2,7.2,8.8)	(4.4,6,7.6)	(4.8,6.4,8)
WB	(4.8,6.4,7.6)	(4.4,6,7.6)	(5.2,6.8,8.4)

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 8, Issue 11, November 2019

Table 2: Triangular Fuzzy numbers of Linguistics values for each goal of Software Testing Techniques.

S. No.	Linguistics value	Triangular fuzzy number
1	VL (Very Low)	(0,0,0.25)
2	L (Low)	(0,0.25,0.5)
3	M (Middle)	(0.25,0.5,0.75)
4	H (High)	(0.5,0.75,1)
5	VH (Very high)	(0.75,1,1)

Table 3: Triangular fuzzy numbers of linguistics values for the relationship between goals and criteria.

S. No.	Linguistics value	Triangular fuzzy number
1	VW (Very Weak)	(2, 2, 4)
2	W(Weak)	(2, 4, 6)
3	M (Medium)	(4, 6, 8)
4	S (Strong)	(6, 8, 10)
5	VS (Very Strong)	(8, 10, 10)

Table 4: Relationship between Software Testing Techniques and Criteria evaluated by five DM.

DM	Models	Criteria's		
		C_1	C_2	C_3
DM_1	BB	W	M	W
DM_2		M	W	W
DM_3		M	W	W
DM_4		W	W	W
DM_5		M	M	W
DM_1	GB	M	M	S
DM_2		M	S	S
DM_3		M	M	M
DM_4		M	M	M
DM_5		M	M	M
DM_1	WB	S	M	S
DM_2		S	S	S
DM_3		S	S	S
DM_4		M	S	S
DM_5		M	S	S

4.176QF=0.46,1.896,1.82|4.176|0.32,4.24,6.864
4.84QF=0.44,2.156,2.244|4.84|0.24,2.9,7.5
3.84QF=0.352,1.661,1.824|3.84|0.288,2.6
4.29QF=0.417,1.905,1.963|4.29|0.283,2.865,6.8
4.176QF=0.46,1.896,1.52|4.176|0.32,3.008,6.864
3.3QF=0.352,1.496,1.452|3.3|0.32,2.92,5.7
3.84QF=0.352,1.664,1.824|3.84|0.288,0.5228,0.69
3.772QF=0.388,1.685,1.699|3.772|0.309,2.806,6.24,6.268
3.712QF=0.368,1.664,1.68|3.712|0.24,2.46,5.93
3.3QF=0.352,1.496,1.452|3.3|0.32,2.72,5.7
4.08QF=0.352,1.752,1.98|4.08|0.288,2.76,6.55
3.697QF=0.357,1.637,1.704|3.697|0.283,2.647, 6.06

After applying the steps 3.3 & 3.4, the EA of all FR by means of extended addition and scalar multiplication in the form of:

$R_1 \ominus \bar{R} = \mathbf{0.126, 4.677, -4.427|1.01| - 0.104, -4.607, 5.072}$
$R_2 \ominus \bar{R} = \mathbf{0.097, 4.457, -4.691| - 0.138| - 0.078, 1.064, 4.48}$
$R_3 \ominus \bar{R} = \mathbf{0.066, 4.409, -4.686| - 0.222| - 0.104, -4.389, 4.272}$

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

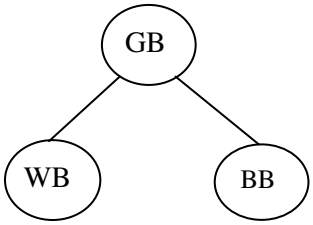
International Journal of Advanced Research in Computer and Communication Engineering

Vol. 8, Issue 11, November 2019

After executing steps 3.5 , we get the following rv of FR: $r1 = 1.22183375,\ r2 = 1.80234639,\ r3 = 1.31728872.$

Now we construct the BST of the given models on the basis of the rv that we have obtained in the previous step (see Fig). Then we traverse the BST in IN-ORDER to get the following order:

$r_2 > r_3 > r_1\ i.e\ GB > WB > BB$



VI. CONCLUSION

This paper presents a method for the selection of Software Testing Techniques using Fuzzy Set Theory. Proposed method is a four step process, namely, (i) identify the criteria, (ii) construct the hierarchical structure of Software Testing Techniques, (iii) construct the decision matrix, and (iv) the selection of a technique. Proposed method selects the agile methods for the testing of the project. On the basis of our analysis, we identify that there is a need to improve the agile methods by intertwining of decision making approaches for the selection and prioritization of requirements. Future research agenda includes the following:

1. To improve the analysis phase of adaptive process model for agile development by applying TOPSIS method.
2. To propose a fuzzy decision making approach or the selection of Software Testing Techniques.
3. To propose a hybrid approach of Software Testing Techniques.
4. To propose a method for the selection of Software Testing Techniques using hybrid techniques like fuzzy Set Theory and fuzzy ANP.

REFERENCES

[1]. Coulter A, “Gray Box Software Testing Methodology”, White paper, Version 0.8.
[2]. Coulter Andre, “Gray box Software testing Methodology-Embedded software testing technique”, 18th IEEE Digital Avionics Systems Conference Proceedings, pp. 10.A.5-2, 1999.
[3]. Huang et al, “Repairing GUI test Suites using Genetic Algorithms”.
[4]. Memon A, “An Event Flow Model of GUI based Applications for Testing”, Software Testing Verification, and Reliability, Wiley Inter Science, pp. 137-157, 2007.
[5]. Mohapatra, Bhuyan, and Mohapatra, “Automated Test Cases Generation and Its Optimization using Genetic Algorithm and Sampling”, IEEE International Conference on Information Engineering, 2009.
[6]. Mohd. Sadiq, “Application of Graph Theory to Software Engineering”, South East Asian Journal of Mathematics and Mathematical Sciences, India, Vol.3, No.3, pp 53-57, 2005.
[7]. Mumtaz Ahmad Khan and Mohd. Sadiq, “Analysis of Black Box Software Testing Techniques: A Case Study”, IEEE International Conference and Workshop on Current Trends in Information Technology, pp.1-5, December, 2011, Dubai, UAE.
[8]. Mumtaz Ahmad Khan, Preeti Bhatia, and Mohd. Sadiq, “BBTool: A Tool to Generate the Test Cases”, International Journal of Recent Technology and Engineering, Vol. 1, Issue 2, pp. 192- 197, June 2012.
[9]. Sabharwal, Sibal, and Sharma, “A Genetic Algorithm based Approach for Prioritization of Test Cases Scenarios in Static Testing”, IEEE International Conference of Communication and Technology, 2011.
[10]. Sharma, Sabharwal, and Sibal, “ A Survey on Software Testing Techniques using Genetic Algorithms”, IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 1, No.1, 2013.
[11]. Viera et al., “Automation of GUI testing Using Model –Driven Approach”, ACM- AST, China, 2006.
[12]. Li RJ(1999) Fuzzy method in group decision making. Comput Math Appl-Elsevier 38:91-101.
[13]. Aho AV, Hopcroft JE,Uliman JD (1983) Data structures and algorithms. Addison-Wesley, Amsterdam.
[14]. American Society of Heating Refrigerating and Air-Conditioning Engineers, 2009 ASHRAE Handbook - Fundamentals (I-P Edition) .: American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc., 2009.
[15]. Fariborz Haghighat, Edward Morofsky, Edward Kutrowski Brian Coffey, "A Software framework for model predictive control with GenOpt," Elsevier , vol. 42, pp. 1084-1092, January 2010.
[16]. Svend Svendsen Steffen Petersen, "Method for simulating predictive control of building systems operation in early stages of building design," ELSEVIER , pp. 4597- 4606, May 2011.
[17]. A. Abdurazik and J. Offutt, “Using uml collaboration diagrams for static checking and test generation,” in Proceedings of the third International Conference on the UML. York, UK: Lecture Notes in Computer Science, Springer-Verlag GmbH, 2000, pp. 383 – 395.
[18]. C. Mingsong, Q. Xiaokang, and L. Xuandong, “Automatic test case generation for uml activity diagrams,” in Proceedings of the 2006 international workshop on Automation of software test, Shanghai, China, 2006, pp. 2 – 8.