Scientific Research Publishing

Journal of Software Engineering and Applications, 2020, 13, 129-142
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

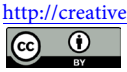
A Fault-Based Testing Approach in Safety Critical Medical Systems

Xaveria Youh Djam[1*], Yisa Henry Kimbi[2]

[1]Department of Computer Science, University of Yaounde I, Yaounde, Cameroon
[2]Urgences, CHE, CNPS, Yaounde, Cameroon
Email: *kdxaveria@gmail.com

How to cite this paper: Djam, X.Y. and Kimbi, Y.H. (2020) A Fault-Based Testing Approach in Safety Critical Medical Systems. *Journal of Software Engineering and Applications*, **13**, 129-142.
https://doi.org/10.4236/jsea.2020.136009

Received: April 18, 2019
Accepted: June 2, 2020
Published: June 23, 2020



Abstract

The advent of technology has opened unprecedented opportunities in health care delivery system as the demand for intelligent and knowledge-based systems has increased as modern medical practices become more knowledge-intensive. As a result of this, there is greater need to investigate the pervasiveness of software faults in Safety critical medical systems for proper diagnosis. The sheer volume of code in these systems creates significant concerns about the quality of the software. The rate of untimely deaths nowadays is alarming partly due to the medical device used to carry out the diagnosis process. A safety-critical medical (SCM) system is a complex system in which the malfunctioning of software could result in death, injury of the patient or damage to the environment. The malfunctioning of the software could be as a result of the inadequacy in software testing due to test suit problem or oracle problem. Testing a SCM system poses great challenges to software testers. One of these challenges is the need to generate a limited number of test cases of a given regression test suite in a manner that does not compromise its defect detection ability. This paper presents a novel five-stage fault-based testing procedure for SCM, a model-based approach to generate test cases for differential diagnosis of Tuberculosis. We used Prime Path Coverage and Edge-Pair Coverage as coverage criteria to ensure maximum coverage to identify feasible paths. We analyzed the proposed testing procedure with the help of three metrics consisting of Fault Detection Density, Fault Detection Effectiveness and Mutation Adequacy Score. We evaluated the effectiveness of our testing procedure by running the suggested test cases on a sample historical data of tuberculosis patients. The experimental results show that our developed testing procedure has some advantages such as creating mutant graphs and Fuzzy Cognitive Map Engine while resolving the problem of eliminating infeasible test cases for effective decision making.

Keywords

Mutation Testing, Software Development, Software Testing, Test Coverage,

Mutation Adequacy Score

1. Introduction

To investigate the pervasiveness of software faults in Safety critical medical system, a fault-based testing approach is desired. The emergence of technology has opened unprecedented opportunities in health care delivery system as the demand for intelligent and knowledge-based systems has increased as modern medical practices become more knowledge-intensive [1]. As a result of this, there is a greater need to investigate the pervasiveness of software faults in safety critical medical system for proper diagnosis. A safety-critical medical system is a complex system in which the malfunctioning of software could result in death, injury of the patient or damage to the environment. The rate of untimely deaths nowadays is alarming partly due to medical device used to carry out the diagnosis process. Not surprisingly, software quality is a serious and growing problem. To scope with the limitation of software testing, a five-stage fault-based testing approach has been proposed in this paper that does not only generates test cases but equally a Fuzzy Cognitive Map (FCM) Engine for the system under test. In view of this, mutation analysis is used as a technique to assess the quality of a test suite.

In recognition of the fact that exhaustive testing is impossible in any software development project, the need to have a methodical means to improve on the available diagnostic and treatment modalities for the system under test in order to reduce the rate of mortality, a diagnostic system is desired. This paper presents an innovative five-stage testing procedure for a fault-based approach in Safety Critical Medical System and the system under test is a diagnostic system for the management of Tuberculosis (TB).

Computer based techniques that employ fuzzy Cognitive maps have been extensively examined for improving diagnosis and treatment of diseases [1] [2] [3] [4] and until today remains an active research area because the merging of mutation analysis and FCM to address the oracle problem to determine successful test cases is still an unresolved issue.

The remainder of this paper is organized as follows. Section 2 provides a background on mutation analysis and fuzzy cognitive map. Section 3 provides an overview of our approach, and Sections 4 and 5 present the experimental setup and evaluation of the research. Finally, the paper concludes with a discussion of future work.

2. Related Work

In this section, we present a categorized survey of the background concepts mutation analysis and Fuzzy Cognitive Maps FCM). These related works are classified according to the difference strategies for their test case generation.

2.1. Mutation Analysis

This paper proposes the use of mutation analysis to design effective test cases for safety critical medical systems for the management of TB. Mutation testing modifies a software artefact such as a program, requirements specification, or a configuration file, to create new versions called mutants [5]. The mutants are usually intended to be faulty versions and are created by applying rules for changing the syntax of the software artefact. These rules are called mutation operators [6]. The tester then creates tests that cause the original and each mutated version to exhibit different behaviours, called killing the mutant [9]. Several researches have been conducted on mutation testing to discover faults in some programming languages [7] [8]. To the best of our knowledge, this is the first attempt to define mutation operators for safety critical medical systems that attempt to merge the concepts of mutation analysis and fuzzy cognitive maps.

2.2. Fuzzy Cognitive Maps

The concept of a fuzzy set was originally proposed by [10] as an extension of the notion of a set by allowing partial membership, and the usage of fuzzy sets theory in medical applications can be traced back early to work by [3] [4] who advocated and put the foundations of the theory to model relationships of symptoms and diseases by using the compositional rule of inference (CRI) as an inference mechanism. Fuzzy Cognitive Maps FCMs, which were introduced by (Kosko, 1986 [10]) as an extension of Cognitive Maps, are powerful tools for modeling dynamic systems. A Fuzzy Cognitive Map (FCM), as a branch of fuzzy logic [11] [12], is a causal knowledge-driven methodology for modeling complex decision systems originally developed by [13].

Few frameworks based on fuzzy cognitive maps for medical reasoning have been proposed [1] [2] [3] [4]. Due to the fact that, the complexities of medical practice make traditional approaches of analysis inappropriate, this multi-expert approach presented by these researchers would have efficiently aid in medical decision but for the fact that they did not base on a model-based approach to generate test suits while maintaining accuracy and interpretability. "Improving accuracy while maintaining interpretability" is the main focus of a Clinician. We proposed that the performance of a FCM Engine can extensively be increased by merging the FCM model with mutation analysis for effective decision making.

In a recent research, a fuzzy expert system for tuberculosis diagnosis was developed [1]. The main focus of the work was accuracy and no doubt, the researchers would have achieved "high peak" accuracy. But a physician involved in sensitive decision making about a patient's treatment, demands more than that. Factors including interpretability, system's ability to adopt human reasoning to deal with uncertainties, identification of successful test cases, mutation operators for safety critical medical systems and performance consistency were ignored. While [14] attempted and suggested a novel fuzzy system for TB management,

no test cases were generated and no metric was used to evaluate the system. According to our approach, the performance of the FCM model was improved by developing new mutation operators to remove redundant test cases and the identification of suitable test cases for effective decision making.

The review of related literature shows that there is a painful lack of diagnostic predictive models for TB management and no explicit attempts are made to merge mutation analysis and Fuzzy Cognitive Maps to address the test suite problem for proper medical decision.

3. Materials and Methods

For data gathering, historical patients' medical records on tuberculosis were sourced from Urgences-CHE CNPS Yaounde, Cameroon from 2018 to 2019. Personal interview was conducted with domain experts (Physicians) in order to get the necessary data for this study. We sampled a total of 200 patients' medical records. The method of sampling strategy adopted was systematic sampling method.

We proposed a five stage fault-based testing procedure and analyzed the proposed testing procedure with the help of three metrics consisting of Fault Detection Density, Fault Detection Effectiveness and Mutation Adequacy Score.

Fault-Based Testing Procedure

Our proposed testing procedure consists of the following five stages:

1) Dividing the application based on its structure;
2) Creating a Mutant graph from application structure;
3) Selection of Coverage Criteria and production of test paths;
4) Create a Fuzzy Cognitive Map Engine;
5) Deriving and running test cases.

These stages are explained in the section below.

Dividing the Application Based on Its Structure

In the first step, for all the chosen Domain Experts, all input factors and the output factors are identified. (Identification of input and output parameters), determination of the numbers of linguistic variables associated with each input/output parameter and select an appropriate membership function. Determine all the fuzzy sets for the system and generate linguistics variables (Table 1, Table 2).

1) Procedure to Generate Linguistic Weights for FCM Engine

The procedure to generate linguistic weights that describe the cause-effect relationships among the concepts of the FCM engine was developed and is presented below:

Step 1: For all N Domain Experts, set credibility weight e_{ij} in the interval [−1, 1].

Step 2: For all ordered pair of concepts (C_i and C_j), N Domain Experts describe the interrelationship between concepts using IT-THEN rules derived from

data pair: IF a {Mild, Moderate, Severe or Very Severe} change occurs in the value of C_i THEN a {Mild, Moderate, Severe or Very Severe} change in the value of concept C_j is caused. Thus the influence of concept C_i on concept C_j is {Negatively Strong, Negatively Moderate, Negatively Weak, Negatively Very Weak, No Relationship, Positively Weak, Positively Moderate, Positively Strong or Positively Very Strong}.

Step 3: If causality occurs, it occurs to Maximum Positive or Maximum Negative.

Step 4: Using Fuzzy Ranking Values, the Fuzzy Maximum is applied and a linguistic weight is assigned between concept C_i and concept C_j.

Step 5: IF for two or more interconnections between concept C_i and concept C_j, more than 3N/4 Experts assigned difference linguistic weights, THEN request the Experts to GOTO step 2 and reassign weights for that particular interconnections.

Step 6: Aggregate all the linguistics weights proposed for every interconnection using the SUM method and construct the Weight Matrix.

Table 1. Fuzzy sets for tuberculosis.

Concepts	Linguistic Variables
C1: Cough > 3 weeks	None (No Cough), Moderate (Productive Cough), Severe (Non-Productive Cough)
C2: Headache	None, Mild, Moderate, Severe and Very Severe
C3: Constipation	Absence, Presence (Absence = 0, Presence = 1)
C4: Fever	None, Mild (Low grade fever: 36˚ - 38.4˚), Severe (high grade fever > 38.5˚)
C5: Weight Loss	None, Mild, Moderate, Severe and Very Severe
C6: Drenching night sweats	Absence, Presence (Absence = 0, Presence = 1)
C7: Loss of appetite	Absence, Presence (Absence = 0, Presence = 1)
C8: Anaemia	None, Mild, Moderate, Severe and Very Severe
C9: Sputum AFB	None, (+)Mild, (++)Moderate, (+++)Severe, (++++)Very Severe,
C10: Mantoux Test	None (0 - 5 mm), Mild (5 - 6 mm), Moderate (7 - 8 mm), Severe (9 - 10 mm) and Very Severe (≥10 mm)
C11: Haemoptysis (Coughing of blood)	Absence, Presence (Absence = 0, Presence = 1)
C12: Tachypnea (Fast Breathing)	Mild, Moderate, Severe and Very Severe
DS: Severity of the Disease	None, Mild, Moderate, Severe and Very Severe

Table 2. Fuzzy Ranking of Input/output variables (concepts) for Tuberculosis.

Linguistic Variables	Fuzzy Values	Classification
Neutral	0	0
Mild	$0 < x < 0.3$	1
Moderate	$0.3 \le x < 0.6$	2
Severe	$0.6 \le x < 0.8$	3
Very Severe	$0.8 \le x \le 1.0$	4

Creating a Mutant Graph from Application Structure

Given: The original program **P**, consisting of input concepts/vectors C_i and a set of Test Cases T. A test suit $T = \{t_1, t_2, \cdots, t_n\}$, with each test case represented as Coverage Vector $V_{ti} = \{v_{t1}, v_{t2}, \cdots, v_{tn}\}$, such that v_{ij} is one (1) if t_i identifies fault and Zero (0) if t_i does not identifies fault.

Different mutants can be created to discover the behavior of each concept in every state of the program. In this approach model-based mutation testing is used to generate mutant graphs by applying mutation operators. In our testing procedure, we need mutation operators to add vectors, update vector, normalized vector to the FCM Engine. Since the operators, such as delete node, add node, delete edge, or add edge, cannot create a new vector/concept/node, add node, update node and normalized node are considered as the mutation operators. As a result, new edges can be produced by changing the input and output parameters of each state vector (C1-C12 represent input parameters) while eliminating the infeasible test cases and detecting all redundant test cases. According to the number of created edges, the output of this third stage can be one or more mutant graphs (**Figure 1**) to be used for the FCM Engine.

Selection of Coverage Criteria and Production of Test Paths

The goal of the third stage is to produce test paths from the Mutants graphs. At this stage a proper coverage criterion (**Table 3**) and an automatic tool are required. As shown in **Figure 2** (Ammann and Offutt, 2017 [9]), Complete Path Coverage (CPC) can produce all test paths from a graph. Nevertheless, the number of paths obtained can be infinite through a loop due to the path explosion problem. Thus, to satisfy all coverage criteria without compromising quality of a safety critical medical system for the management of TB, Prime Path Coverage (PPC) and Edge-Pair Coverage (EPC) which are the two subsets of CPC are selected for the coverage criteria. Web Graph Coverage was used as the tool to produce test paths in our novel approach.

Create a Fuzzy Cognitive Map Engine

At this stage, Fuzzy Cognitive Map Engine can now be developed. At this stage, FCM graphs are created based on the opinion of Domain Experts. The Decision Concept (DS) represents the severity of tuberculosis infection and takes four fuzzy values (Mild, Moderate, Severe, Very Severe) as shown in **Table 1**. The 13 identified concepts with 13 fuzzy sets keep relation with each other, in order to characterize the process of diagnosing tuberculosis. After the determination of fuzzy sets, each expert was asked to define the degree of influence among the concepts and equally describe their interrelationships using IF-THEN rules derived from fuzzy decision tree and the inferred fuzzy weights were combined and normalized using sigmoid function and the result was a crisp value representing the weight of each interconnection. In this way, the weights of interconnections between the concepts were determined.

According to domain experts' judgments, input variables selected for this re-

search were described with five linguistic variables (Neutral, Mild, Moderate, Severe and Very Severe) in the fuzzy interval of [0, 1] as shown in Table 2.

From the FCM Engine (Figure 3), an Overall Weight Diagnostics Matrix ($M_{\text{OverallWeight}}$) with 13 rows and 13 columns was computed as shown in Table 4 below.

Deriving and Running Test Suite

All the test paths that are extracted from mutant graphs and FCM Engine are inputs to the final stage to generate test cases. The test cases are executed, the actual results are compared with the expected results and test oracle is established as shown in Table 5. The output of the final stage is represented as a list of faults arising from running test cases.

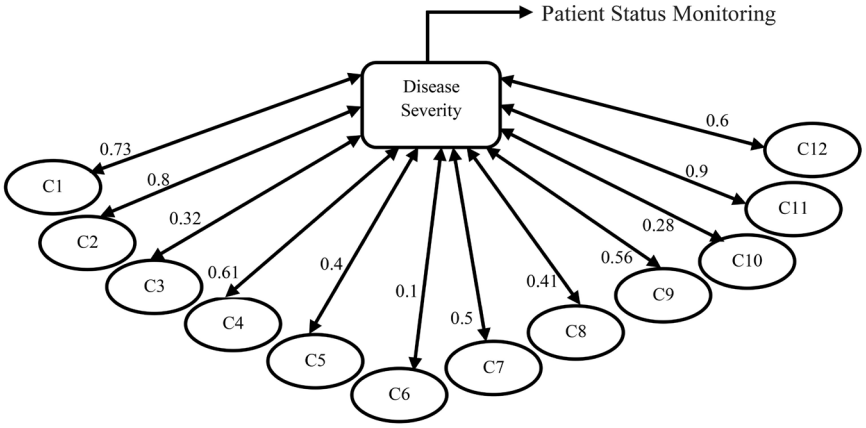


Figure 1. A sample weighted mutant graph for the diagnostics TB.

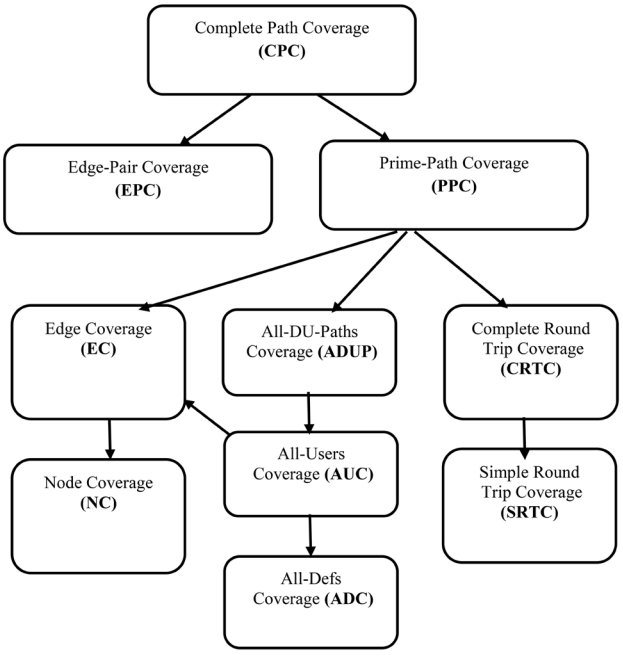


Figure 2. Graph coverage criteria (Adapted from Ammann and Offutt, 2017 [9]).

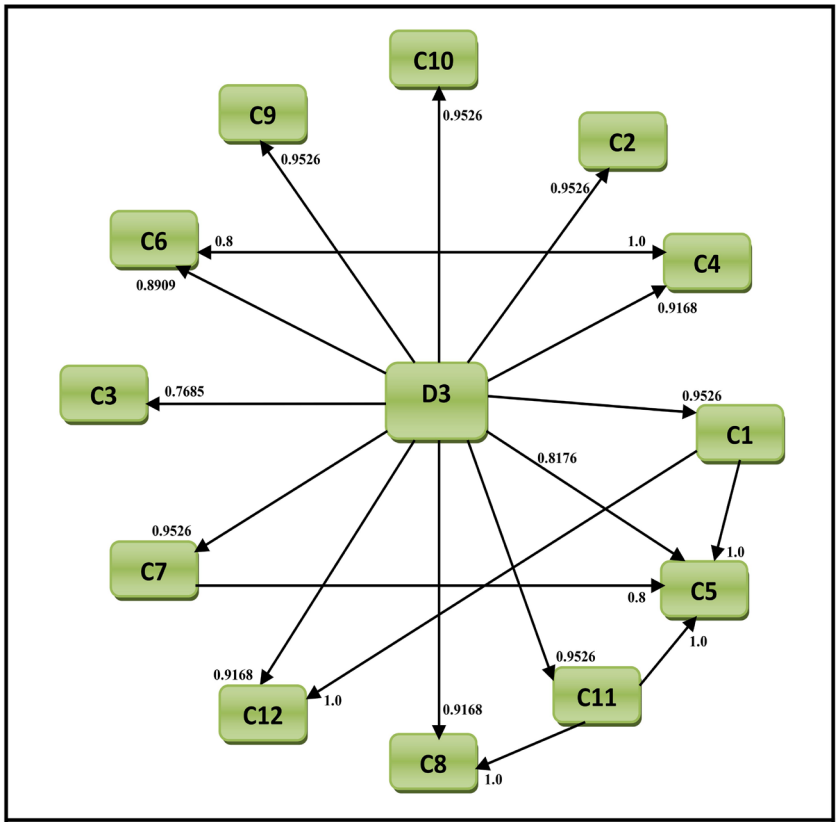


Figure 3. FCM engine for assessing the severity of tuberculosis.

Table 3. Types of graph coverage criteria (adapted from Ammann and Offutt, 2017 [9]).

Name	Description	Terms
Node Coverage (NC)	Test set T Satisfies node coverage on graph G if and only if for every syntactically reachable node n in N, there is some path p in path (T) such that p visits n. Test Requirement (TR) contains each node in G	**Test Requirements (TR):** Describe properties of test paths
Edge Coverage (EC)	TR contains each reachable path of length up to 1, inclusive, in G	**Test Criterion**
Edge Pair Coverage (EPC)	TR contains each reachable path of length up to 2, inclusive, in G	The rules that define test requirements
Complete Path Coverage (CPC)	TR contains all paths in G	**Satisfaction**
Specific Path Coverage (SPC)	TR contains a set of S of test paths, where S is supplied as a parameter	Given a set TR of test requirements for a criterion C, a set tests T satisfies C on graph if and only if for every test requirement in TRN there is a test path in path (T) that meets the test requirement TR
Prime Path Coverage (CPC)	**Simple Path:** A path from node n_i to n_j is simple if no node appears more than once, except possibly the first and last nodes are the same. **Prime Path:** A simple path that does not appear as a proper sub path of any other simple path. TR contains each prime path in G	**Test Criterion**

Table 4. Overall weight diagnostics matrix for tuberculosis.

Concepts	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	DS
C1	0	0	0	0	1.0	0	0	0	0	0	0	1.0	0
C2	0	0	0	0	0	0	0	0	0	0	0	0	0
C3	0	0	0	0	0	0	0	0	0	0	0	0	0
C4	0	0	0	0	0	0.8	0	0	0	0	0	0	0
C5	0	0	0	0	0	0	0	0	0	0	0	0	0
C6	0	0	0	1.0	0	0	0	0	0	0	0	0	0
C7	0	0	0	0	0.8	0	0	0	0	0	0	0	0
C8	0	0	0	0	0	0	0	0	0	0	0	0	0
C9	0	0	0	0	0	0	0	0	0	0	0	0	0
C10	0	0	0	0	0	0	0	0	0	0	0	0	0
C11	0	0	0	0	1.0	0	0	1.0	0	0	0		0
C12	0	0	0	0	0	0	0	0	0	0	0	0	0
DS	**0.9526**	**0.9526**	**0.7685**	**0.9168**	**0.8176**	**0.8909**	**0.9526**	**0.9168**	**0.9526**	**0.9526**	**0.9526**	**0.9168**	**0**

Table 5. An example of test suit document with test paths inclusive.

Test Paths	Test Case	Expected Results	Actual Results	Test Oracle
[1, 3, 2, 6, 7, 9, 8, 12, 4, 11, 5, 10,] {input parameters path}	Diagnosis	Severe Diagnosis	Should report severe diagnosis	Pass
[ds, 10] {Disease severity Path }	Treatment Outcome	Disease Progressive State	Should report disease progressive state	Fail

4. Results

4.1. Dividing the Application Based on Its Structure

For this stage of our testing procedure for the safety critical medical system, by analyzing the concepts (input variables) the main diagnostic structures in the system are presented in Table 1.

4.2. Creating a Mutant Graph from Application Structure

Considering the add node, update node and normalized node as the mutation operators for the safety critical medical system, 76 mutants were created (for both diagnosis and treatment) and 14 mutants were killed as shown in Table 6 and Figure 1 shows a sample of the mutant graphs. The mutants were killed by EPC and PPC test sets. Equivalent mutants were identified by hand analysis. Table 7 shows the results for each mutation operator.

4.3. Selection of Coverage Criteria and Production of Test Paths

A sample path can be seen in Table 5 with the corresponding test case. We used Prime Path Coverage and Edge-Pair Coverage as coverage criteria to ensure maximum coverage to identify feasible paths.

Table 6. Number of created mutants.

Mutant Class	Number of Created Mutants	Mutant Killed	Equivalent Mutants
Diagnosis	41	13	12
Treatment	35	02	04
Total	76	14	16

Table 7. Empirical results for each mutation operator.

Mutation Operator	Mutants Killed	Equivalent Mutants	Live Mutants	Total Mutants	Mutation Scores
Add Node	3	0	1	4	0.750
Update Node	10	4	21	35	0.675
Normalized Node	0	0	2	2	0.693
Subtotal	13	04	24	41	0.351

4.4. Create a Fuzzy Cognitive Map Engine

By simulating a safety critical medical diagnostic system, our FCM Engine performance is illustrated by means of simulating two scenarios of tuberculosis infection as a means to diagnose and manage a TB patient. In each of these scenarios, we have an initial state vector V_{Initial} representing the presented events at a given time of the process and a final state vector V_{Final} representing the last state vector produced in convergence region—the final value of the decision concept.

First Scenario: For this scenario, an immunocompromised patient health status (with respect to tuberculosis) with patient identification number as 015 was considered, with Non-productive Cough > 3 weeks (C1 = 1.0), Severe Headache (C2 = 0.7), Severe Fever (C4 = 0.7), Moderate Weight Loss (C5 = 0.5), Presence of Drenching Night Sweats = (C6 = 1.0), Severe Sputum AFB (C9 = 1.0), Severe Mautoux Test (C10 = 0.7), Presence of Haemoptysis (C11 = 1.0), Tachynea (C12 = 0.7).

Thus the initial state vector (Input Vector) V_{Initial} is:

V_{Initial} =	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13
	[1.0	0.7	0	0.7	0.5	1.0	0	0	1.0	0.7	1.0	0.7	0]

After the fuzzy inference process (simulation process), at the eighth iteration step, the FCM concept values did not changed, which indicates that the equilibrium region is reached at a fixed point with the final state vector (Decision Output Vector) V_{Final} as:

V_{Final} =	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13
	[0.89	0.78	0	0.69	0.75	0.87	0	0	0.77	0.91	0.86	0.67	0.82]

The calculated value of the decision concept (DS = 0.82) indicates that patient number 015 was diagnosed for very severe TB with 82% possibility with gives the result specified by the domain experts.

Second Scenario: For this scenario, a young patient health status (with respect to tuberculosis) with patient identification number as 025 was considered, with productive Cough > 3 weeks (C1 = 0.7), Severe Constipation (C3 = 1.0), Presence of Drenching Night Sweats = (C6 = 1.0), Loss of Appetite (C7 = 1.0), Moderate Sputum AFB (C9 = 0.7), Moderate Mautoux Test (C10 = 0.7), Presence of Haemoptysis (C11 = 1.0).

Thus the initial state vector (Input Vector) V_{Initial} is:

V_{Initial} =	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13
	[0.7	0	1.0	0	0	1.0	1.0	0	0.7	0.7	1.0	0	0]

After the fuzzy inference process (simulation process), at the eleventh iteration step, the FCM concept values did not changed, which indicates that the equilibrium region is reached at a fixed point with the final state vector (Decision Output Vector) V_{Final} as:

V_{Final} =	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13
	[0.44	0	0.55	0	0	0.67	0.62	0	0.53	0.65	0.39	0	0.56]

The calculated value of the decision concept (DS = 0.56) indicates that patient number 025 was diagnosed for moderate TB with 56% possibility with gives the result specified by the domain experts.

Similarly, we computed the results for the other patients and got results that were in the range of predefined limits by the domain experts.

4.5. Deriving and Running Test Cases

By applying our developed testing procedure on safety critical medical system, 1778 test cases were produced. An example of a test case document is shown in Table 5.

5. Evaluation

In this study, we used the following three performance metrics consisting of Fault Detection Density, Fault Detection Effectiveness and Mutation Adequacy Score to show the performance ability of the developed fault-based model. We evaluated the effectiveness of our testing procedure by running the suggested test cases on a sample historical data of tuberculosis patients sourced from Urgences-CHE CNPS Yaounde, Cameroon from 2018 to 2019.

✧ **Fault Detection Density (FDD):** FDD is the ratio of the total number of faults detected by each test case ($\neq tf_i$) and the total number of test cases ($\neq T$) in the total number of unique faults ($\neq F_{\text{unique}}$) and is calculated as:

$$\text{FDD} = \neq \frac{\left(tf_1 + tf_2 + \cdots + tf_n\right)}{\left(\neq T \times \neq F_{\text{unique}}\right)} \tag{5.1}$$

✧ **Fault Detection Effectiveness (FDE):** FDE is the ratio of the total number of faults ($\neq F_{\text{Total}}$) and the total number of unique faults($\neq F_{\text{Unique}}$). FDE is calculated as below:

$$\text{FDE} = \frac{\left(\neq F_{\text{Total}}\right)}{\left(\neq F_{\text{Unique}}\right)} \tag{5.2}$$

✧ **Mutation Adequacy Score (MAS):** MAS, used for evaluating mutated graphs, is the ratio of the total number of mutants killed ($\neq \text{Mutant}_{\text{killed}}$) and the total number of mutants ($\text{Mutant}_{\text{Total}}$) with no equivalent mutants ($\neq \text{Mutant}_{\text{Equivalent}}$). M Mutation Adequacy Score is calculated as below:

$$\text{Mutation Score} \frac{\neq \text{Mutant}_{\text{killed}}}{\left(\neq \text{Mutant}_{\text{Total}} - \neq \text{Mutant}_{\text{Equivalent}}\right)} \tag{5.3}$$

The mutation score for the case study is represented in Table 7 and Table 8 presents the values of FDD, FDE and the unique faults for the stages of the developed testing procedures.

6. Conclusions and Future Work

This paper presents an innovative approach to fault-based testing in Safety Critical Medical Systems. We introduced a new five-stage testing procedure that has some advantages: 1) creating testing cases while resolving the problem of removing infeasible test cases, 2) creating FCM Engine for our case study because without sound diagnosis and accurate treatment, medical practice is as good as guess work.

While the research is promising, several research questions remain unanswered. An important evaluation, currently being planned, is to do a full fault study. We will generate tests to kill all non-equivalent mutants, then evaluate those tests to determine how many faults the tests detect, and compare with tests generated for other criteria (possibly statement or branch coverage).

7. Threads to Validity

Our empirical evaluation has some threats to validity. Firstly, equivalent mutants were identified manually by one person. Secondly, our implementation of the three proposed mutation operators may include faults. To ensure they work as expected, we tested our suggested testing procedure constantly, and checked mutants generated by hand very carefully. Thirdly, like most software engineering experiments, it is not possible to guarantee the representativeness of selected subjects,

Table 8. Evaluating developed testing procedure.

Testing Procedure	FDD (%)	FDE (%)	Unique Fault No.
Mutated Graphs	**3**	**4**	**3**
EPC Criterion	**2**	**3**	**10**
PPC Criterion	**5**	**5**	**2**
FCM Engine	**0.6**	**1**	**2**

we made an honest attempt and choose real life case study from the medical domain. The fact that the case study was used by previous researchers provides consistency across multiple studies.

Acknowledgements

The authors are very grateful for every constructive comment from Anne Marie Sah for her guidance and editing, Ntam Victor and Richard Favour for their advice on the fault detection and coverage criteria used in this research. Furthermore, the authors would like to thank the reviewers for suggesting qualitative changes in the paper.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Djam, X.Y. and Kimbi, Y.H. (2011) A Decision Support System for Tuberculosis Diagnosis. *Pacific Journal of Science and Technology*, **12**, 410-425.

[2] Greenhalgh, J., Flynn. R., Long, A.F. and Tyson, S. (2008) Tacit and Encoded Knowledge in the Use of Standardized Outcome Measures in Multidisciplinary Team Decision Making: A Case Study of In-Patient Neurorehabilitation. *Social Science & Medicine*, **67**, 183-194. https://doi.org/10.1016/j.socscimed.2008.03.006

[3] Papageorgiou, E.I., Stylios, C.D. and Groumpos, P. (2003) An Integrated Two-Level Hierarchical Decision Making System Based on Fuzzy Cognitive Maps (Fcms). *IEEE Transactions on Biomedical Engineering*, **50**, 1326-1339. https://doi.org/10.1109/TBME.2003.819845

[4] Papageorgiou, E.I., Spyridonos, P., Ravazoula, P., Groumpos, P.P. and Nikiforidis, G. (2006) A Soft Computing Method for Tumour Grading Cognitive Maps. *Journal of Artificial Intelligence in Medicine*, **36**, 58-70.

[5] DeMillo, R.A., Lipton, R.J. and Sayward, F.G. (1978) Hints on Test Data Selection: Help for the Practicing Programmer. *Computer*, **11**, 34-41. https://doi.org/10.1109/C-M.1978.218136

[6] Deng, L., Mirzaei, N., Ammann, P. and Offutt, J. (2015) Towards Mutation Analysis of Android Apps. 2015 *IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops* (*ICSTW*), Graz, 13-17 April 2015, 1-10. https://doi.org/10.1109/ICSTW.2015.7107450

[7] Jhamb, M., Singhal, A. and Bansal, A. (2013) A Survey on Different Approaches for Efficient Mutation Testing. *International Journal of Scientific and Research Publications*, **3**, 1-5.

[8] Offutt, A.J. (1994) A Practical System for Mutation Testing: Help for the Common Programmer. *Proceedings of the International Test Conference*, Washington, DC, 2-6 October 1994, 824-830. https://doi.org/10.1109/TEST.1994.528535

[9] Ammann, P. and Offutt, J. (2017) Introduction to Software Testing. Cambridge University Press, Cambridge. https://doi.org/10.1017/9781316771273

[10] Kosko, B. (1986) Fuzzy Cognitive Maps. *International Journal of Man-Machine Studies*, **24**, 65-75. https://doi.org/10.1016/S0020-7373(86)80040-2

[11] Zadeh, L.A. (1965) Fuzzy Sets. *Information and Control*, **8**, 338-353. https://doi.org/10.1016/S0019-9958(65)90241-X

[12] Zadeh, L.A. (1965) Fuzzy Sets and Systems. In: Fox, J., Ed., *Proceedings Symposium on System Theory*, Polytechnic Institute of Brooklyn, New York, 29-37.

[13] Zadeh, L.A. (1969) Biological Applications of the Theory of Fuzzy Set and Systems, In: Proctor, L.D., Ed., *The Proceedings of an International Symposium on Biocybernetics of the Central Nervous System*, Little, Brown and Company, Boston, 199-206.

[14] Mohammad, S.H., Fatema, T.J., Faisal, A. and Karl, A. (2017) A Belief Rule Based Expert System to Assess Tuberculosis under Uncertainty. *Journal of Medical Systems*, **41**, Article number: 43. https://doi.org/10.1007/s10916-017-0685-8