FAULT COVERAGE AND DIAGNOSIS OF PROTOCOLS AND SYSTEMS MODELED AS EXTENDED FINITE STATE MACHINES

by

Mark Habib Hassoun

A Thesis Presented to the Faculty of the
American University of Sharjah
College of Engineering
in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science in
Computer Engineering

Sharjah, United Arab Emirates
January 2015

Approval Signatures

We, the undersigned, approve the Master's Thesis of Mark Habib Hassoun

Thesis Title: Fault Coverage and Diagnosis of Protocols and Systems Modeled As Extended Finite State Machines

Signature **Date of Signature**

___________________________ _______________

Dr. Khaled El-Fakih
Associate Professor, Department of Computer Science and Engineering
Thesis Advisor

___________________________ _______________

Dr. Tarik Ozkul
Professor, Department of Computer Science and Engineering
Thesis Committee Member

___________________________ _______________

Dr. Ashraf Elnagar
Professor, Department of Computer Science, University of Sharjah
Thesis Committee Member

___________________________ _______________

Dr. Assim Sagahyroon
Head, Department of Computer Science and Engineering

___________________________ _______________

Dr. Mohamed El-Tarhuni
Associate Dean, College of Engineering

___________________________ _______________

Dr. Leland Blank
Dean, College of Engineering

___________________________ _______________

Dr. Khaled Assaleh
Interim Vice Provost for Research & Graduate Studies

Acknowledgment

I would like to express my sincere gratitude to my advisor Dr. Khaled El-Fakih for his guidance and invaluable suggestions at every step throughout the thesis. Also I would like to thank him for his patience, motivation, enthusiasm, and immense knowledge.

Abstract

Automatic test derivation from formal specifications offers a rigorous discipline to functional conformance testing. In various application domains, such as communication protocols and other reactive systems, the specification can be represented in the form of an Extended Finite State Machine (EFSM). Many methods can be used for deriving test suites from an EFSM specification. In practice, developing and applying these test suites to an Implementation Under Test (IUT) is time consuming and costly. Thus, it is desirable to determine high quality test suites in order to reduce the cost of testing. To this end, in the first part of this thesis, using six realistic application examples, we conduct experiments, assess, determine the fault coverage, and accordingly rank various known types of EFSM-based test suites. While the purpose of conformance testing is to check if an IUT is different from its specification, an interesting, complementary, yet more complex step, is called fault diagnosis or diagnostic testing. The objective of fault diagnosis is to determine the faulty implementation, and thus find the differences between the specification and its implementation. In the second part of this thesis, we present a diagnostic method, conduct experiments, and assess the fault localization capabilities of the EFSM-based test suites considered in the first part of the thesis. The fault localization capability of a test suite is determined for many types of diagnostic candidates, representing possibly faulty EFSM implementations, such as candidates with single or double transfer faults, candidates with single assignment faults, and many other types of candidates. In addition, for each considered test suite, the method determines the diagnostic tests required, in addition to the considered test suite, for locating a faulty EFSM IUT.

Search Terms: Extended Finite State Machine, Fault Diagnosis, Test Derivation, Test Assessment, Software Engineering, Software Testing, Mutation Testing, Conformance Tests.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AOIS	Arithmetic Operator Insertion - Shortcut
AOIU	Arithmetic Operator Insertion - Unary
AORB	Arithmetic Operator Replacement - Binary
AORS	Arithmetic Operator Replacement - Shortcut
COD	Conditional Operator Deletion
COI	Conditional Operator Insertion
COR	Conditional Operator Replacement
DTF	Double Transfer Fault
EFSM	Extended Finite State Machine
EP	Edge Pair Test Suite
FSM	Finite State Machine
IUT	Implementation under Testing
PPST	Prime Path with Side Trip Test Suite
Rand	Random Test Suite
ROR	Relational Operator Replacement
SAD	Single Assignment Deletion Fault
SAI	Single Assignment Insertion Fault
SARHS	Single Assignment Right-hand-side Fault
SITS	State Identifiers Test Suite
SOPF	Single Output Parameter Fault
STF	Single Transfer Fault
TC	Test Case
TS	Test Suite
TT	Transition Tour Test Suite

Chapter 1: Introduction

Automatic test derivation from formal specifications offers a rigorous discipline to functional conformance testing of various reactive systems. In several application domains, such as communication protocols and other reactive systems, the specification can be represented in the form of an Extended Finite State Machine (EFSM). In particular, EFSMs are the underlying models for formal description techniques, such as the Specification and Description Language [26]. EFSMs extend the traditional (Mealy) Finite State Machine (FSM) model with input and output parameters, context variables, update statements and predicates (or guards) defined over context variables and input parameters. The EFSM model is widely acknowledged as a very powerful model for test derivation.

Many EFSM-based test derivation methods are presented considering the coverage of particular types of EFSM faults such as single and double transfer faults, assignment faults and single output parameter faults. Test suites, which are sequences of input/output pairs of (executable, for feasible) traces of the EFSM specification, are usually derived from a given specification considering some fault coverage criteria. Given a deterministic EFSM specification, a set of deterministic EFSM mutants of the specification representing possible faulty implementations, a test suite of one or many test cases is usually derived from the given specification in such a way that these tests can distinguish the given specification from the derived mutants. A mutant is distinguished from another mutant (specification) by a test case if the output responses of the mutant and the other mutant (specification) to the input sequence of the test case are different. Known types of EFSM mutants used in test derivation include mutants with Single Transfer Faults (STFs), Double Transfer Faults (DTFs), and Single Output Parameter Faults (SOPFs). Corresponding test suites are thus called STF, DTF, and SOPF test suites. Further, EFSM-based test derivation can also be done from the Flow-Graph representation of the EFSM specification using the well-known data-flow All-Uses criterion that covers the All-Uses of each context variable and every parameterized input of the specification. Another way for test derivation is to consider the graph representation of the specification, and derive tests using the so-called Edge-Pair (EP), Prime Paths (PP), and Prime Paths with Side Tours (PPST) [8] coverage criteria. An Edge Pair (EP) test suite covers each executable path of length up to 2 of the EFSM graph, and a Prime Path (PP) test suite covers each simple path (a path where no node appears more than once in the path) that does not appear as a proper sub-path of any other simple path, while a PPST test suite covers the same path covered by the PP test suite and every edge in this path. Another EFSM-based test derivation criterion is based on reaching states of the EFSM and then applying special input sequences, called distinguishing sequences, which are capable of distinguishing intended states of the EFSM. Such test suites are called SITS test suites. Another possibility for test derivation is to randomly derive a test suite with one (executable) test case of a particular length from the given EFSM specification or derive a test suite, called a Transition Tour (TT), of one test case that starts at the initial state and traverses all transitions of the EFSM.

In practice, developing test suites and applying these test suites to an Implementation Under Test (IUT) is time consuming and costly. It is well known that deriving a test suite that can detect many types of EFSM faults in an IUT is impractical as the length of such a suite would be huge, even if some assumptions were made regarding the behavior of an IUT. Thus, determining high quality test suites reduces the cost of software testing.

For specifications modeled as EFSMs, a preliminary assessment of many types of EFSM test suites, such as STF, DTF, TT, All-Uses, and some random test suites has been recently presented in [6]. Fault coverage of a test suite is determined in terms of the capability of the test suite in killing all possible STF or DTF mutants that are distinguishable from the given EFSM specification. In addition, a similar study has been presented in [7] [44]. However, unlike the work in [6], (fault) coverage of a test suite was assessed in terms of its capability for detecting code mutants of an implementation of the EFSM specification. However, in [7] [44], only three application examples were considered in the study and, as reported in [7] [44], there is a need to consider more application examples to verify the obtained results and also there is a need to consider more types of EFSM test suites. Accordingly, in the first part of this thesis, we conduct experiments to assess the fault coverage of many EFSM-based test suites as done in [7] [44]; however, our study considers six working examples including the two used in [7] [44]. In addition, we consider more types of EFSM test suites, namely, EP and PPST test suites. Furthermore, in this thesis, a comprehensive assessment of the fault coverage of random test suites, hereafter named Rand, is carried out. In particular, in the first part of this thesis, we evaluate the fault coverage of all the above mentioned test suites using six known EFSM specifications and corresponding Java code implementations. The fault coverage of these test suites is determined using code mutants of the Java implementations where code mutants are derived using the traditional arithmetic, logical, and conditional operators [1]. Ranking (from best to worst) of the test suites is done based on two criteria, the first is based solely on *fault coverage* (or *mutation score*), and the other is based on both the fault coverage and length (called *coverage-length score*) of the test suites. In summary, based on the conducted experiments, the best performing test suites, in terms of fault coverage, are the SITS (61.4 %) followed by the PPST (59.6 %), TT (59.5 %), STF (59.2 %), All-Uses (56.3 %), Rand (55.2 %), and then the EP (50.2 %) test suites. However, when considering the *coverage-length* score, the TT (250.59) and All-Uses (232.15) test suites have comparable scores, and they outperform the other test suites by approximately 73 percent. The STF (82.11), SITS (77.99), PPST (60.99), EP (53.72) and Rand (49.80) test suites have comparable scores, but each of these test suites scores less than the TT and the All-Uses test suites by approximately 73 percent. Test suite fault coverage of Conditional Operator Insertion (COI) and Conditional Operator Deletion (COD) faults is on average 86%, and it is significantly higher than the coverage of mutants with other types of operator faults by approximately 29 percent. Test suite coverage of AORS, AORB, AOIS, AOIU, ROR and COR faults are comparable, but this coverage is less than the coverages of COI and COD by approximately 29 percent. Test suite coverage of conditional faults (73%) is significantly higher than the coverage of mutants with arithmetic and relational faults by approximately 17.5 percent. Test suite coverage of mutants with arithmetic faults is comparable to the coverage of mutants with relational faults, but this coverage is less than the coverage of conditional faults by approximately 17.5 percent. SITS test suites have the best fault coverage of arithmetic faults (65%), conditional faults (81%) and relational faults (69%). The remaining test suites have comparable coverages in terms of arithmetic and conditional and relational faults, but their coverage is less than the coverage of the SITS test suites by approximately 12 percent. When considering the *coverage-length* score, the TT and All-Uses test suites have comparable scores, and they outperform the other test suites in terms of score of arithmetic, conditional and relational faults by approximately 74 percent. The remaining test suites have comparable scores but they are less than the scores of the TT and All-Uses tests by approximately 74 percent.

While the purpose of EFSM-based or FSM-based (conformance) testing is to check whether an implementation is different from its specification, an interesting complementary, yet more complex, step is to locate the differences between a specification and its implementation. The purpose of fault diagnosis (or diagnostic testing) is to locate the differences between a specification and its implementation, when the implementation is found to be faulty. Given an EFSM specification and a test suite derived from the specification, and given an EFSM black-box IUT, in general, fault diagnosis involves the derivation of all possible EFSM mutants of the specification, called *diagnostic candidates*, that respect or have the same input/output behavior with respect to the given test suite, as the given IUT. Thus, each of these candidates is indistinguishable from the given IUT and there is a need to derive addition tests, called *diagnostic tests*, capable of locating the candidate (or a set of indistinguishable candidates) that is (are) indistinguishable from the given black-box IUT.

In the software domain where a system can be represented as an FSM, some work has already been done for the diagnostic and fault localization problems [37] [38] [39]. In [37], [39] and [40] the differences between the system specification and its implementation is located under the assumption of a single fault in the implementation. In [41] the differences can be located for multiple faults under the assumption that each of the faults is reachable through non-faulty transitions. In [42], considering a system consisting of two communicating FSMs, a method is presented to decide if it is possible to locate a faulty component machine, and if this is possible, then diagnostic tests for locating the fault(s) are derived. In [43], a fault localization method for EFSMs is presented based on the derivation of mutants of a particular type, represented in a compact way in a so-called fault function, and the derivation of (diagnostic) tests that distinguish fault functions and thus their constituent mutants.

In the second part of this thesis, we conduct a comprehensive case study for assessing the fault localization capabilities of all the above mentioned EFSM test suites. The study allows us to rank (from best to worst) the test suites with respect to their fault localization (or diagnosis) capability. Two criteria are presented for assessing the fault diagnosis capabilities of test suites. The first, called FD1 score, determines the capability of a test suite in locating the fault considering all diagnostic candidates of a particular type(s) of faults. The second, named FD2 score, determines the capability of locating the fault with respect to a set of distinguishable classes of diagnostic candidates. That is, sets of distinguishable classes of the diagnostic candidates are formed where any two candidates in the same set are indistinguishable from each other where any candidate in a set is distinguishable from all candidates in another set of candidates. In addition, the presented study includes an assessment of the additional efforts, measured in terms of length of the diagnostic tests in addition to the length of a considered EFSM test suite, required to locate the faulty IUT. An algorithm is presented which, given an EFSM test suite and a certain types(s) of EFSM fault(s), the algorithm derives possible mutants of the considered type of fault(s), eliminates those which are indistinguishable from the given specification (as those are non-faulty mutants) and obtains a set of diagnostic candidates. Then, the candidates are distributed into sets of distinguishable classes, and afterwards considering a given faulty IUT, the algorithm determines the fault diagnosis capability of the test suite in locating the considered faulty implementation. Many types of diagnostic candidates are considered in the assessment, namely, candidates with STF, DTF, and SOPF faults. In addition, we consider candidates with many types of single assignment faults. Based on the conducted experiments, the following results are obtained. Using the fault localization score FD1, on average, the SITS (98.4%) test suites have the best FD1 score. The EP (98%) and STF (97.6%) test suites have comparable FD1 scores, but these score are less than those of the SITS by approximately 0.5 percent. All-Uses (95.4 %) and Rand (91%) test suites scores are less than that of the SITS by approximately 3 and 7.5 percent, respectively. TT (84.7 %) and PPST (83.7 %) test suites have comparable FD1 scores, but these scores are less than that of the SITS by approximately 14 percent. Using the fault localization score FD2, on average, the SITS (91.3%) test suites have the best FD2 score. The EP (80.5%) and STF (80.3%) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 10.9 percent. The All-Uses (77.3%) and Rand (75%) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 15.2 percent. The TT (69.7%) and PPST (69.4%) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 21.8 percent. When ranking the test suites based on a score computed using the fault diagnosis score FD1 and the length of the test suite the All-Uses (393.5) and TT (357) test suites have comparable scores, which are greater than the scores of other test suites by approximately 71.5 percent. The STF (135.5), SITS (124.9), EP (104.9), PPST (85.7) and Rand (82.2) test suites have comparable scores, but these scores are less than the scores of All-Uses and TT test suites by approximately 71.5 percent. When ranking the test suites based on a score computed using the fault diagnosis score FD2 and the length of the test suite, the All-Uses (318.7) and TT (293.7) test suites have comparable scores, which are greater than the scores of other test suites by approximately 70.5 percent. The SITS (115.9), STF (111.5), EP (86.1), PPST (71.1) and Rand (67.7) test suites have comparable scores, but these scores are less than the scores of the All-Uses and TT test suites by approximately 70.5 percent. When comparing the test suites in terms of the total length of a test suite in addition to the length of the additional diagnostic tests needed for locating the fault, the best performing test suite is the TT, followed by the STF, All-Uses, SITS, PPST, EP then the Rand test suites.

This thesis is organized as follows. Chapter 2 includes preliminaries about EFSMs and EFSM-based test suites and types EFSM faults. Chapter 3 includes an assessment of random test suites and an assessment and ranking of the considered test suites. Chapter 4 includes an assessment of the fault diagnosis capabilities of these test suites and Chapter 5 concludes this thesis.

Chapter 2: Preliminaries

In this chapter, we introduce the deterministic extended finite state machine EFSM model. Many types of EFSM-based test suites are introduced, namely, Transition Tour (TT), All-Uses of context variables, Single Transfer Fault (STF), State Identifiers (SITS), Edge Pair (EP), Prime Paths with Side Tripe (PPST) and random test suites (Rand). At the end of this chapter, EFSM-based and Code-based mutation testing mechanisms are introduced.

2.1 The EFSM Model

The deterministic EFSM model extends the traditional Mealy FSM model with variables, assignment statements, predicates and parameterized inputs and outputs. Here we illustrate notions related to EFSMs, mostly taken from [9], and describe how an EFSM operates through a working example.

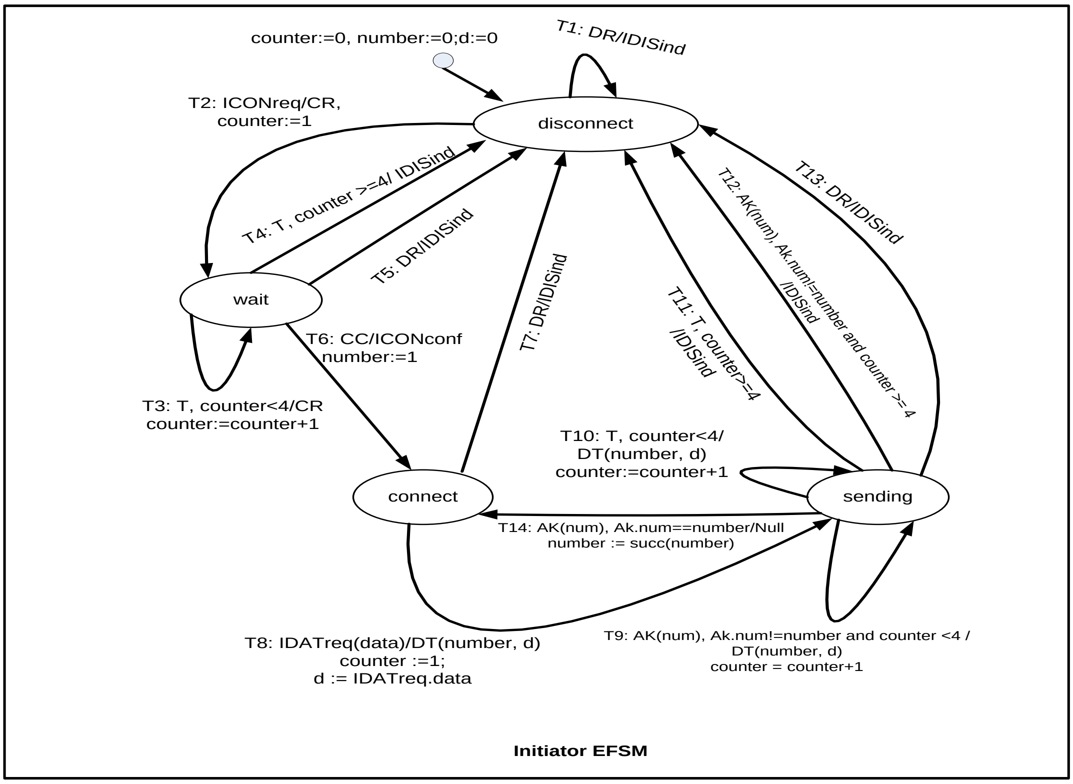

Figure 2.1 The Initiator EFSM [10]

An EFSM is defined over states S, with initial state $s_0 \in S$, inputs X, outputs Y, parameters R, and context variables V. For $x \in X$, $R_x \subseteq R$ denotes the set of input parameters and D_{Rx} denotes the set of valuations of the parameters over the set R_x.

Similarly, for $y \in Y$, $R_y \subseteq R$ denotes the set of output parameters and D_{Ry} denotes the set of valuations of the parameters over the set R_y. The set D_V denotes the set of context variable valuations. A context *variable valuation*, or *valuation vector*, is denoted as **v**. Considering the Initiator EFSM [10] shown in the figure above, it is defined over state set S = {*disconnect*, *wait*, *connect*, *sending*} with *disconnect* as the initial state s_0, inputs X ={*DR*, *ICONreq*, *T*, *CC*, *IDATreq*, *Ak*}, where *IDATreq* and *Ak* are parameterized inputs with integer parameters *IDATreq.data* and *Ak.num* which can have values 0 or 1, thus, the set of parameterized inputs R_x = {*IDATreq.data*, *Ak.num*} with domains $DR_{IDATreq} = DR_{Ak}$ = {0, 1}. The set of outputs of the machine is Y = {*IDISind*, *CR*, *ICONconf*, *DT*, *Null*} where *DT* is a parameterized output with integer output parameter *DT.number* which can have the values 0 or 1. The set of context variables of the machine is V = {*number*, *d*, *counter*} where *number* and *d* are integers with possible values 0 or 1, respectively, and *counter* is an integer over the domain {0, ∞}, thus, the set of the context variables *number*, *d*, and *counter* valuations equals $D_V = \{0, 1\} \times \{0, 1\} \times \{0, \infty\}$.

An EFSM has a set of transitions T between states in S, such that each transition $t \in T$ is a tuple (s, x, P, op, y, up, s') such that s and s' are the *start* and *final* states of t, $x \in X$ is the input and $y \in Y$ is the *output*, P is a predicate (guard) of t defined as $P : D_{Rx} \times D_V \rightarrow$ {*True*, *False*}, *up* is a context update (assignment of context variables) defined as $up : D_{Rx} \times D_V \rightarrow D_V$, and *op* the *output parameter update* of t defined as $op : D_{Rx} \times D_V \rightarrow D_{Ry}$. We note that an input x (or output y) can have no parameters; in this case, $R_x = \emptyset$ ($R_y = \emptyset$), and the input (output) is simply denoted by x (y). For example, the machine in the figure above has transition T_2= (*disconnect*, *ICONreq*, *True*, *CR*, *counter*:= 1, *wait*) with states *disconnect* and *wait* as the starting and final states of the transition, respectively, *ICONreq* as an input; T_2 has no guard (or predicate), i.e. has the trivial guard *True*, and T_2 has *CR* as an output and the context update function *counter* := 1. The machine also has transition T_5= (*sending*, *Ak*, (*Ak.num* != *number* and *counter* < 4), *DT*, *counter*:= *counter* +1, *sending*) with parameterized input *Ak* with input parameter *Ak.num* and guard (*Ak.num* != *number* and *counter* < 4), parameterized output *DT* carrying the values of the context variables *number* and *d*, and context update *counter* := *counter* + 1. A context variable valuation $\mathbf{v} \in D_V$ is called a *context* of M. A *configuration* of M is a tuple (s, **v**) where s is a state and **v** is a context. For example, configuration (*sending*, (**1, 1, 1**)) represents the fact that the machine is in the state of *sending* where the current values of each of the context variables *number, d*, and *counter* is 1, i.e., context valuation vector equals (**1, 1, 1**).

An EFSM operates as follows. Assume that EFSM is at a current configuration (s, $\mathbf{v}$) and the machine receives an input ($\boldsymbol{x}$, $\mathbf{p}_x$) such that ($\mathbf{v}$, $\mathbf{p}_x$) satisfies the guard P of an outgoing transition t = (s, x, P, op, y, up, s'). Then the machine being at (s, $\mathbf{v}$), upon receiving the input ($\boldsymbol{x}$, $\mathbf{p}_x$), executes the update statements of t, produces the (parameterized) output where parameter values are provided by the output parameter function op, and moves to configuration (s', $\mathbf{v}'$), where $\mathbf{v}' = up(\mathbf{p}_x, \mathbf{v})$. Thus, a transition can be represented as $(s, \mathbf{v})$ - $(x, \mathbf{p}_x)/(y, \mathbf{p}_y) \rightarrow (s', \mathbf{v}')$, where $op(\mathbf{p}_x, \mathbf{v}) = (y, \mathbf{p}_y)$. Such a transition can also be written as $((s, \mathbf{v}), (x, \mathbf{p}_x), (y, \mathbf{p}_y), (s', \mathbf{v}'))$. In our working example, assume that (*sending*, (**1, 1, 1**)) is a current configuration of the EFSM and the machine receives the parameterized input *Ak*(0), i.e., *Ak.num* = 0. One of the transitions starting in state *sending* with input *Ak* whose guard is satisfied (considering the context variables and input parameters) can be executed. As only the guard of T_5 holds, transition T_5 is executed; according to the context update function *counter*:=*counter*+ 1 = 1 + 1 = 2, the output *DT*(1, 1) is produced, and the machine remains at the state *sending*. In fact, the machine moves from configuration (*sending*, (**1, 1, 1**)) to configuration (*sending*, (**1, 1, 2**)). An EFSM M is *deterministic* if any two transitions outgoing from the same state with the same input have mutually exclusive predicates. In this thesis, we consider deterministic EFSM specifications where at each state for each (parameterized) input only one transition can be executed under the selected input.

Given input x and the input parameter valuations, a *parameterized input* (or an input) is a tuple (x, $\mathbf{p}_x$), where $\mathbf{p}_x \in D_{Rx}$. A sequence of parameterized and/or non-parameterized inputs is also called *an input sequence*. An *output sequence* can be defined in a similar way. A *path* is a sequence s_1 - $x_1/y_1 \rightarrow s_2$ - $x_2/y_2 \rightarrow \ldots$ - $x_l/y \rightarrow s_l$ of states and input/output pairs of an EFSM starting from the designated state s_1. A path is *feasible* or *executable* if there is a sequence of transitions $(s_1, \mathbf{v}_1)$ - $(x_1, \mathbf{p}_{x1})/(y_1, \mathbf{p}_{y1}) \rightarrow (s_2, \mathbf{v}_2)$ - $(x_2, \mathbf{p}_{x2})/(y_2, \mathbf{p}_{y2}) \rightarrow (s_3, \mathbf{v}_3) \ldots (s_{l-1}, \mathbf{v}_{l-1})$ - $(x_l, \mathbf{p}_{xl})/(y_l, \mathbf{p}_{yl}) \rightarrow (s_l, \mathbf{v}_l)$ in EFSM M starting from configuration $(s_1, \mathbf{v}_1)$. The *input/output projection* of such an executable path is the *sequence of input/output pairs* $(x_1, \mathbf{p}_{x1})/(y_1, \mathbf{p}_{y1})$ $(x_2, \mathbf{p}_{x2})/(y_2, \mathbf{p}_{y2})$ $\ldots(x_l, \mathbf{p}_{xl})/(y_l, \mathbf{p}_{yl})$ and is called a *trace* of M starting from configuration $(s_1, \mathbf{v}_1)$. The input projection of such a trace is an input sequence α=(x_1, $\mathbf{p}_{y1}$) (x_2, $\mathbf{p}_{x2}$) … (x_l, $\mathbf{p}_{x1}$) and the output projection is the corresponding output sequence β = (y_1, $\mathbf{p}_{y1}$) (y_2, $\mathbf{p}_{y2}$) … (y_l, $\mathbf{p}_{y1}$). As an example, consider the feasible path corresponding to the sequence of transitions starting from the initial configurations (**0**, **0**, **0**) of the EFSM in the figure above, (*disconnect*, (**0**, **0**, **0**)) - *DR*/*IDISind* → (*disconnect*, (**0**, **0**, **0**)) - *ICONreq*/*CR* → (*wait*, (**0**, **0**, **1**)) - *T*/*CR*→ (*wait*, (**0**, **0**, **2**)). The corresponding trace is *DR*/*IDISind ICONreq*/*CR T*/*CR* with the input projection *DR ICONreq T* and output projection *IDISind CR CR*.

We use the notation (s_1, $\mathbf{v}_1$) - α → (s_l, $\mathbf{v}_l$) to denote the fact that there exists a trace from (s_1, $\mathbf{v}_1$) to the configuration (s_l, $\mathbf{v}_l$) such that the input sequence of the trace is α. In this case, we say that the input sequence α is *defined* at configuration (s_1, $\mathbf{v}_1$) and we also say that the configuration (s_l, $\mathbf{v}_l$) is reached from (s_1, $\mathbf{v}_1$) by applying α. In this thesis, we consider *executable or feasible* test cases. Thus, hereafter, a *test case* is the sequence of input/output pairs of a trace of the EFSM specification that starts from the initial configuration of the specification machine. A test case is *executable* or *feasible*, as, by definition, it has a corresponding feasible path in *M*. A *Test Suite* (*TS*) is a finite set of test cases. The length of a test case is the number of input/output pairs of the corresponding trace and the length of a test suite *TS* is the total length of its corresponding test cases.

2.2 Types of EFSM Mutants

In this section, we describe the types of EFSM mutants, namely, the transfer fault mutants with single or double transfer faults.

- Single Transfer Fault (STF): Given an EFSM *M*, a transition t = (s, x, P, op, y, up, s') of an EFSM IUT M' has a *transfer fault* if its final state is different from that specified by *M*, i.e., M' has a transition (s, x, P, op, y, up, s''), $s'' \neq s'$, $s'' \in S$. Such M' is a *mutant of M with a single transfer fault*.
- Double Transfer Fault (DTF): Given an EFSM specification *M*, an EFSM mutant M' of *M* has *double transfer fault* if it has two transitions, each with a single transfer fault.

2.3 EFSM-Based Test Suites

In this section, we describe the considered types of EFSM based test suites. Given two EFSMs *M* and M', we say that *M* and M' are *distinguishable* if their initial configurations are distinguishable by an input sequence (or a test case) α. In this case, we say that α *kills M´*.

2.3.1 Single Transfer Faults (STF) Test Suites

An STF test suite is a test suite that covers single transfer faults of *M*, such that for each mutant of *M* with a single transfer fault distinguishable from *M*, the test suite has at least one test case that kills such a mutant.

2.3.2 Transition Tour (TT) Test Suites

A TT test suite of *M* is an input sequence that starts at the initial configuration of *M* and traverses each transition of *M*.

2.4 EFSM Flow-Graph Based Test Suites

Here, we describe the EFSM flow-graph based test suites.

2.4.1 All-Uses Test Suites

An All-Uses test suite is a set of test cases of an EFSM *M* that covers the All-Uses of each context variable and every parameterized input of *M*. Such a test suite can be derived directly from *M* as illustrated in [22] or from a flow-graph representation of *M* as illustrated in [23].

2.4.2 State Identifier (SITS) Test Suites

An input sequence α_{ij} is a *distinguishing sequence* for states s_i and s_j of *M* if α_{ij} distinguishes each pair of configurations (s_i, v) and (s_j, v′), v, v′ $\in D_V$, of *M*. *M* is *state reduced* if each two different states of *M* are distinguishable. Given state $s_j \in S$ of a state reduced EFSM *M* with *n* states, a set W_j of input sequences is called a *distinguishing set* of state s_j, if for any other state s_i there exists a sequence $\alpha \in W_j$ that distinguishes states s_i and s_j. Given distinguishing sets $W = \{W_0, W_1, .. , W_{n-1}\}$ of states of *M*, a State Identifier Test Suite (SITS) is a set of test cases that satisfies the following property. For every transition $t = (s, x, P, op, y, up, s´)$ of *M* and each $\alpha \in W_j$, the *TS* has the input sequence $\gamma.(x, p_x).\alpha$, where γ is the input sequence that takes *M* from the initial configuration to a configuration (*s*, v) such that (v, p_x) satisfies *P* of *t*.

2.5 EFSM Graph-Based Test Suites

Given the EFSM specification *M*, by removing the inputs, outputs, input and output parameters, guards, and update statements of *M*, we obtain a *graph representation* of the EFSM *M*. In the following, we describe two known methods that can be used for deriving test suites from the obtained graph representation of the EFSM.

2.5.1 Edge Pair (EP) Test Suite

An Edge pair test suite is a test suite that covers each executable path of length up to 2 of a given graph. More precisely, Edge-pair coverage requires covering each pair of consecutive edges, or a path of length 2 of the given graph. The phrase "length up to 2" is used to include graphs that have fewer than two edges [8].

2.5.2 Prime Path with Side Trip (PPST) Test Suite

Given a graph representation of an EFSM, an executable path from node n_i to node n_j in the graph is *simple* if no node appears more than once in the path, with the exception that the first and last nodes may be identical. A path from node n_i to node n_j is a *prime path* if it is a simple path and it does not appear as a proper sub-path of any other simple path. A *prime path with side trip* is a path *p* that tours the prime path *q* such that every edge in *q* is also in *p* in the same order [8].

2.6 Random Test Suites

A random test suite is a test suite generated by a random walk through (or from a randomly generated path of) the EFSM specification.

2.7 EFSM-Based and Code-Based Mutation Testing

Mutation testing is a mechanism to evaluate and assess the quality of a test suite and to guarantee its efficiency by checking the coverage of the test suite in terms of number of killed mutants [31]. Mutation Testing is considered an expensive software testing technique. Research and studies have shown that Mutation Testing has a very high and strong rate over other testing techniques in fault and error detection [25].

Code-based mutation testing is a technique for selecting the best test suite depending on fault-based criteria by checking the coverage of each test suite versus the code mutants. Mutants are derived from the code implementation of the specification [26]. The main principle of code-based mutation testing is that every single mutant operator represents a fault that the programmers often make. Therefore, by choosing the appropriate types of mutants carefully, we will be able to eliminate a huge number of programmers' faults. Code-based mutation testing has been widely applied and tested as a white-box technique for many programming languages such as Fortran [27], C [28], and more recently, C# [29], and Java [30].

There are many code mutation operators, but mainly they can be divided into two levels: the traditional (method) level [24] and class level mutants [24]. In this thesis, we derive traditional code mutants using the following well-known types of mutation operators that are of three categories: Arithmetic, Conditional or Relation operator category as described in [24].

Table 2.1 Mutation Operators and Categories

Category	Operator	Description
Arithmetic	AORB	Arithmetic Operator Replacement - Binary
	AORS	Arithmetic Operator Replacement - Shortcut
	AOIS	Arithmetic Operator Insertion - Shortcut
	AOIU	Arithmetic Operator Insertion - Unary
Conditional	COR	Conditional Operator Replacement
	COI	Conditional Operator Insertion
	COD	Conditional Operator Deletion
Relational	ROR	Relational Operator Replacement

Chapter 3: Assessing the Fault Coverage of EFSM Test Suites

Developing test suites and applying these test suites to an implementation under test is time consuming and costly. It is well known that deriving a test suite that can detect many types of EFSM faults in an implementation under test (IUT) is impractical as the length of such a suite would be huge, even if some assumptions were made regarding the behavior of an IUT. Thus, determining high quality test suites reduces the cost of software testing.

In this chapter, we conduct experiments, assess, and compare the fault coverage of EFSM-based test suites in order to determine the quality of these test suites, and thus reduce the cost of testing.

3.1 Considered EFSM Specification Examples

In our experiments, we consider five well-known communication protocols in addition to a CD player specification [35]. Namely, we consider the Trivial File Transfer Protocol (TFTP) [33], the Post Office Protocol V.3 (POP3) [34], The Initiator [44], the Responder [44], the SCP [44], and the CD player [35] specification EFSMs.

3.2 Assessment of Fault Coverage of EFSM Test Suites

Given an EFSM specification *spec* and a Java code implementation of *spec*, and given EFSM test suites derived from *spec*, namely the TT, All-Uses, SITS, STF, EP, PPST and Rand test suites. Considering the code mutants derived using the mutation operators illustrated in Table 2.1, the fault detection capabilities (fault coverage) of each of these test suites is measured as follows:

$$\textit{\textbf{Mutation Score}} = (J_{killed} \,/\, J_{Mutants}) \times 100 \qquad \textbf{(1)}$$

where $J_{Mutants}$ denotes the number of derived mutants of the Java implementation and J_{killed} is the number of these mutants killed by the given test suite.

In addition, we consider an assessment based on both the mutation score (fault coverage) and length of obtained test suites as follows:

$$\textit{\textbf{Coverage-Length Score}} = \frac{0.99 \times Mutation\ Score}{0.01 \times Test\ Suite\ Length} \qquad \textbf{(2)}$$

We show that the ranking of the test suites changes even if we give 99% importance based on the mutation score and only 1% importance based on the length of the test suites as given in (2) (i.e., using the coverage-length score).

The following subsection describes the assessment method in more detail. This method is similar to that reported in [44]; however, in this thesis, we consider six application examples, including the three considered in [44], and more types of test suites as illustrated below.

3.3 Assessment Method in More Detail

The method has three steps. In **Step 1**, for each considered EFSM specification, all EFSM mutants of *M* with STF faults are derived and a corresponding STF test suite (with optimal or near optimal length) is derived as illustrated in [6]. For each considered EFSM specification, the corresponding Transition Tour (TT), SITS, are derived by hand. In addition, the EP and PPST are derived with the help of the *graph coverage* web application tool [8]. Moreover, for every specification, a corresponding flow-graph representation annotated with definitions and uses of variables is constructed and then a corresponding All-Uses test suite is derived from the obtained flow-graph as illustrated in [6] based on related work [23]. The derivation of these test suites is also done by hand with the help of the *graph coverage* web application tool [8]. In **Step 2**, three corresponding Java code implementations are developed by three different software engineers, based on the EFSM specification and its textual description, under the following coding rules. State variables cannot be explicitly or implicitly introduced in an implementation; for example, no state variables and no flags indicating state variables can be used; moreover, no labels and no Go-to statements can be used. In addition, names of context variables, inputs and outputs with their parameters of the EFSM specification should be preserved in a code implementation. Each implementation is coded as one function that inputs a string separated by a delimiter "," representing an input sequence to the function and returns as an output a string representing the output response of the implementation to the input sequence. A Reader/Writer class is used in all implementations that handles reading/writing the input and the output strings in order to separate reading and writing outputs from the function that implements the specification and thus, code mutants are only derived from the function that implements the specification. We note that before deriving mutants, each Java implementation is thoroughly tested using all the considered test suites written in JUnit. In **Step 3**, 1-Order Java code mutants are derived using the Java arithmetic, relational and conditional operators. As usual, 1-Order code mutants are considered to alleviate problems related to the coupling effect of using N-order mutants, when $N > 1$. Afterwards, the fault detection capabilities (*fault coverage*) of each considered test suite of a given EFSM specification is determined using the mutation score of the test suite described above. MuClips [36], MuJava [1], and JUnit are used for the automatic derivation of mutants, execution of test suites, and for determining fault coverage. In addition, the ranking of test suites that we have is done according to multiple criteria as described below.

3.4 Fault Coverage of Random Test Suites

In this section, we study the fault coverage of random test suites and determine the length of the best test suite for each considered EFSM machine. In particular, for each of the considered EFSM examples, we consider varying length test suites, and for each considered length five random test suites are derived and applied to a Java implementation of the EFSM specification. Corresponding fault coverage is determined and we keep increasing the length of random test suites until the following stopping criterion is satisfied and the best random suite length is determined accordingly. The *stopping criterion* states that the length of random test suites keeps increasing until the average mutation score of the five test suites of a considered length does not increase by more than five percent or decrease by more than five percent from the average mutation score of the random test suites with more length. We select the least length that satisfies the above criterion and state that it is the best length of a random test suite for the considered examples. In the following chapter, we assess and compare the fault coverage of (best) random test suites with other EFSM-based test suites.

a) TFTP Example: Determining the Best Random Test Suite Length

Figure 3.1 depicts the length of random test suites studied for one implementation of the TFTP EFSM. Figure 3.1 includes the mutation score of each random test suite with a particular length and the average mutation score of the five considered test suites of the same length.

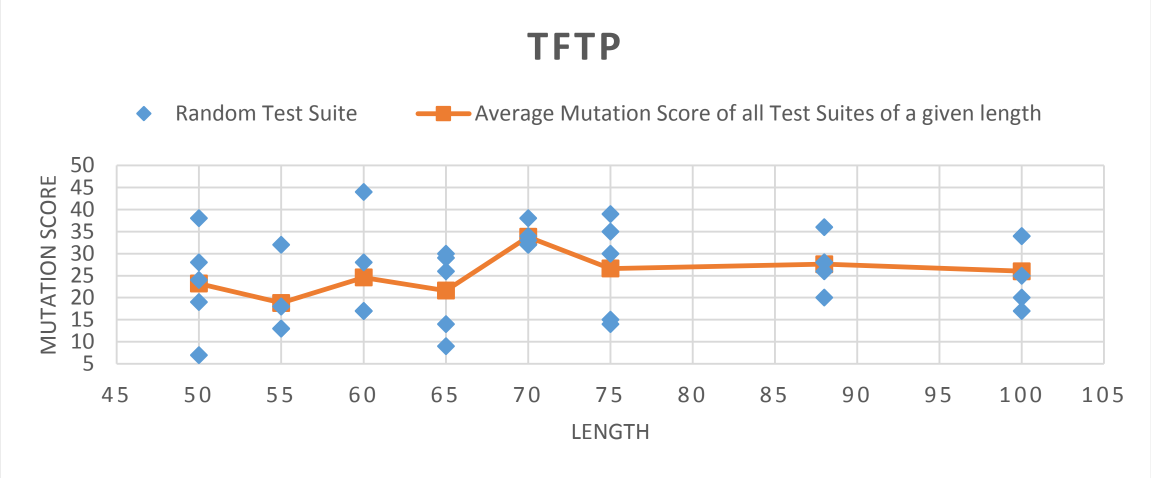

Figure 3.1 Random Test Suites of TFTP

According to the stopping criterion described before, the best random test suite of the TFTP is that with the length of 88.

b) CD Player Example: Determining the Best Random Test Suite Length

Figure 3.2 depicts the length of random test suites studied for one implementation of the CD Player EFSM. Figure 3.2 includes the mutation score of each random test suite with a particular length and the average mutation score of the five considered test suites of the same length.

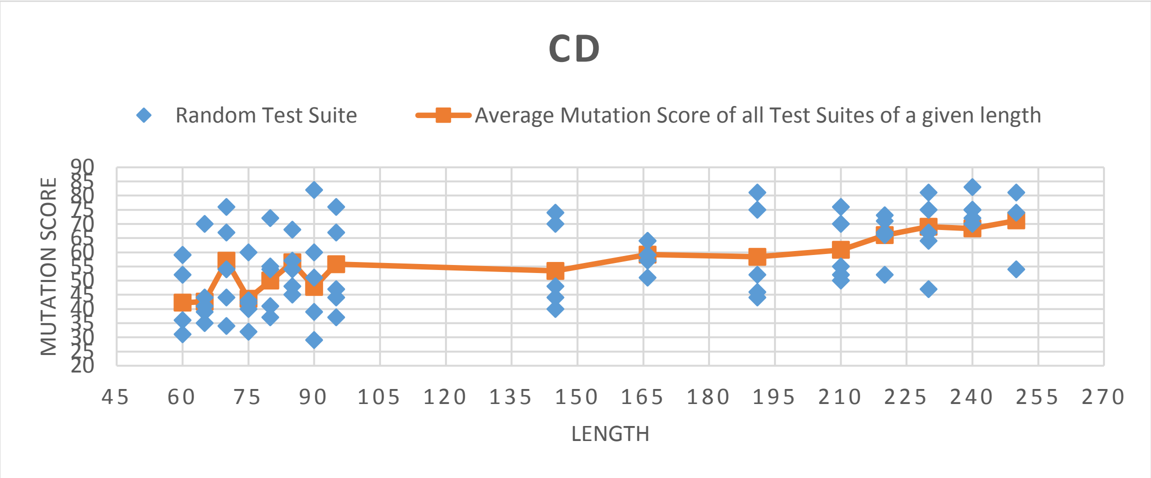

Figure 3.2 Random Test Suites of CD Player

According to the stopping criterion described before, the best random test suite of the CD Player is that with the length of 230.

c) **POP3 Example: Determining the Best Random Test Suite Length**

Figure 3.3 depicts the length of random test suites studied for one implementation of the POP3 EFSM. Figure 3.3 includes the mutation score of each random test suite with a particular length and the average mutation score of the five considered test suites of the same length.

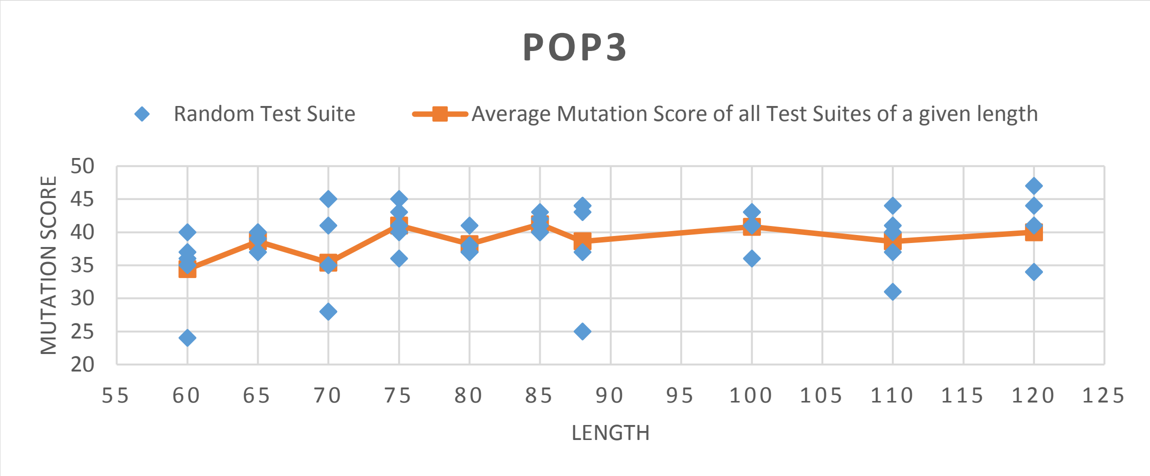

Figure 3.3 Random Test Suites of POP3

According to the stopping criterion described before, the best random test suite of the POP3 is that with the length of 110.

d) **Initiator Example: Determining the Best Random Test Suite Length**

Figure 3.4 depicts the length of random test suites studied for one implementation of the Initiator EFSM. Figure 3.4 includes the mutation score of each random test suite with a particular length and the average mutation score of the five considered test suites of the same length.

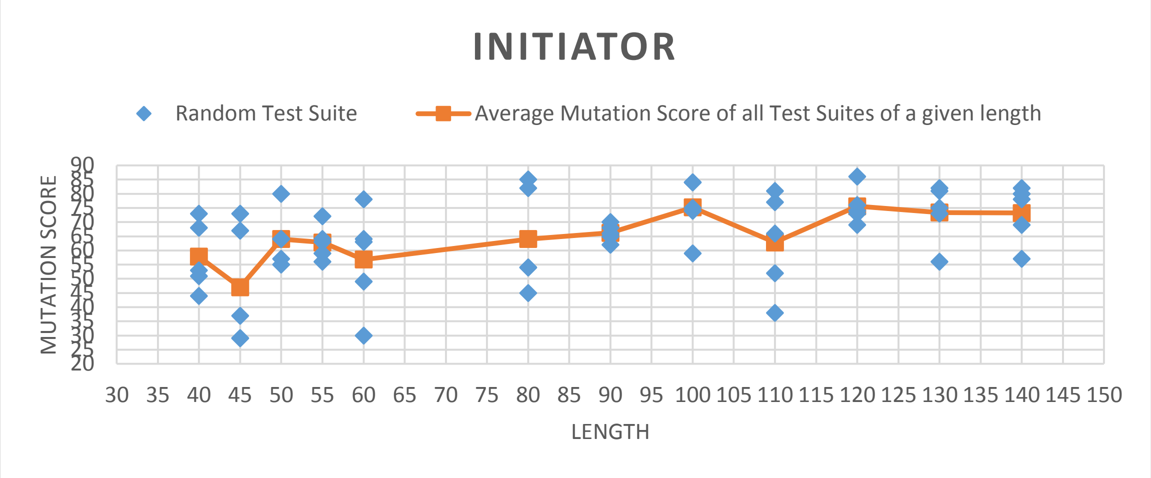

Figure 3.4 Random Test Suites of Initiator

According to the stopping criterion described before, the best random test suite of the Initiator is that with the length of 120.

e) Responder Example: Determining the Best Random Test Suite Length

Figure 3.5 depicts the length of random test suites studied for one implementation of the Responder EFSM. Figure 3.5 includes the mutation score of each random test suite with a particular length and the average mutation score of the five considered test suites of the same length.

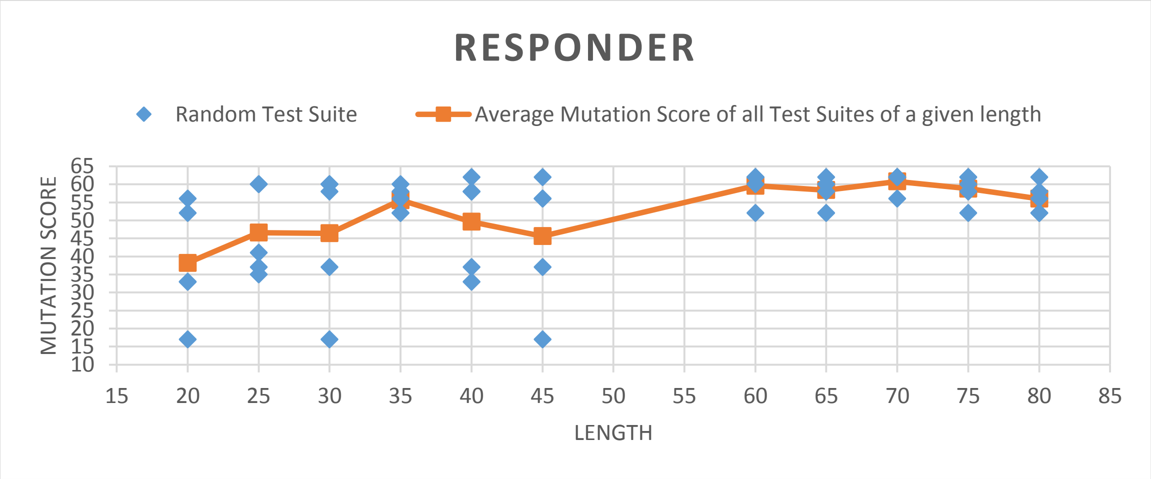

Figure 3.5 Random Test Suites of Responder

According to the stopping criterion described before, the best random test suite of the Responder is that with the length of 65.

f) **SCP Example: Determining the Best Random Test Suite Length**

Figure 3.6 depicts the length of random test suites studied for one implementation of the SCP EFSM. Figure 3.6 includes the mutation score of each random test suite with a particular length and the average mutation score of the five considered test suites of the same length.

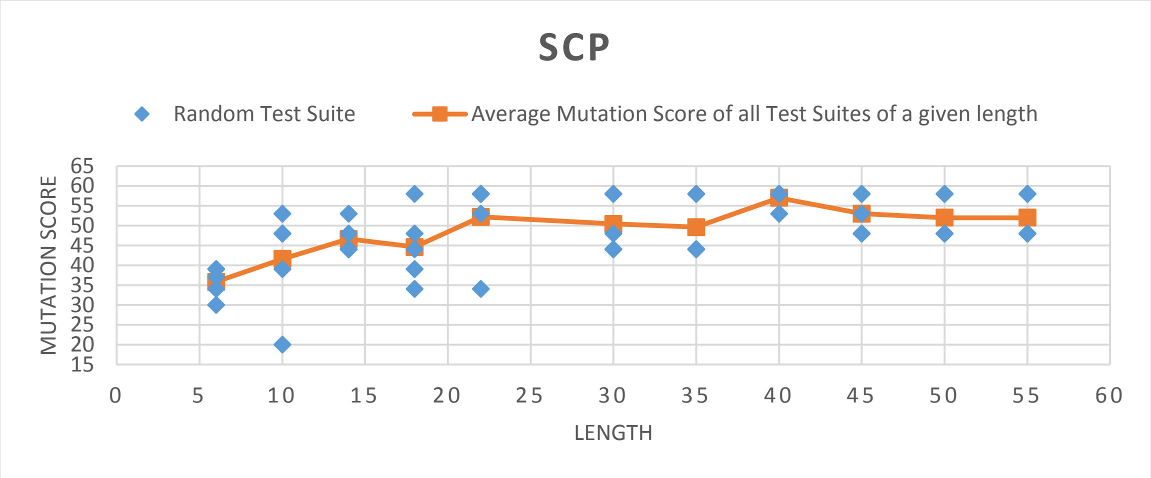

Figure 3.6 Random Test Suites of SCP

According to the stopping criterion described before, the best random test suite of the SCP is that with the length of 45.

3.5 Experimental Evaluation

In the following we present, discuss, rank, and analyze the obtained results of mutation scores (using (1) and (2) described before) as shown in the following tables and figures.

3.5.1 Assessment of the Fault Coverage of Test Suites

This section includes the mutation scores (fault coverage) of the considered test suites for the considered EFSM specifications. For each specification and each considered test suite, average mutation score of the corresponding three java implementations is determined.

a) **TFTP Test Suites Fault Coverage Assessment**

Figure 3.7 includes the mutation scores and length of each considered test suite for the TFTP example.

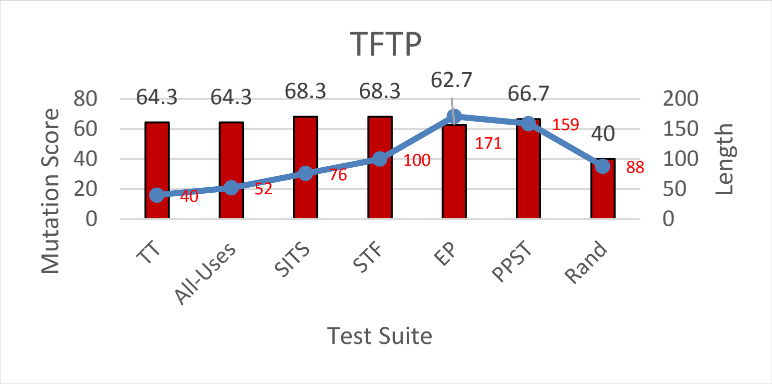

Figure 3.7 TFTP Test Suites Mutation Score and Length

b) CD Player Test Suites Fault Coverage Assessment

Figure 3.8 includes the mutation scores and length of each considered test suite for the CD Player example.

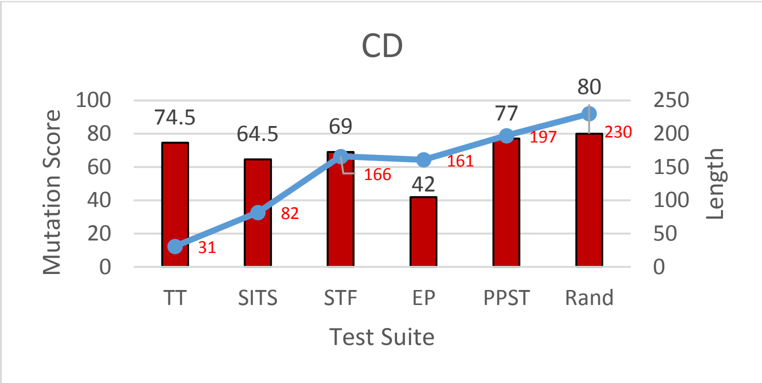

Figure 3.8 CD Player Test Suites Mutation Score and Length

c) POP3 Test Suites Fault Coverage Assessment

Figure 3.9 includes the mutation scores and length of each considered test suite for the POP3 example.

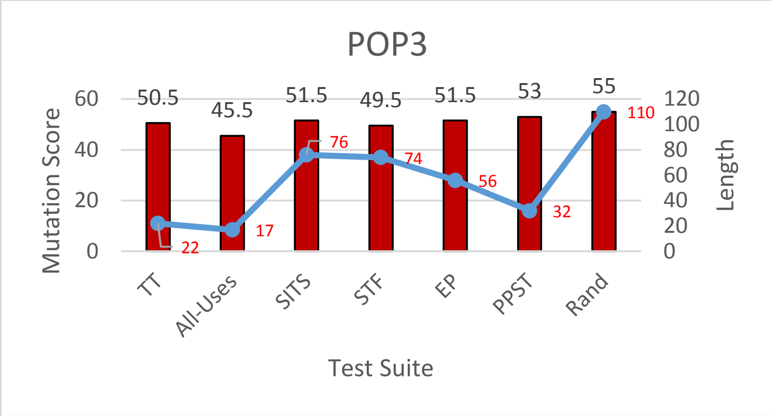

Figure 3.9 POP3 Test Suites Mutation Score and Length

d) Initiator Test Suites Fault Coverage Assessment

Figure 3.10 includes the mutation scores and length of each considered test suite for the Initiator example. We note that there is no SITS test suite for the Initiator as there are no state identifiers for the corresponding states.

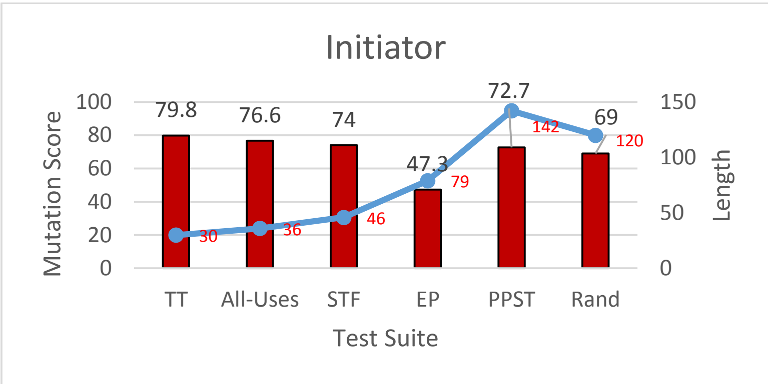

Figure 3.10 Initiator Test Suites Mutation Score and Length

e) Responder Test Suites Fault Coverage Assessment

Figure 3.11 includes the mutation scores and length of each considered test suite for the Responder example. We note that there is no SITS test suite for the Responder as there are no state identifiers for the corresponding states.

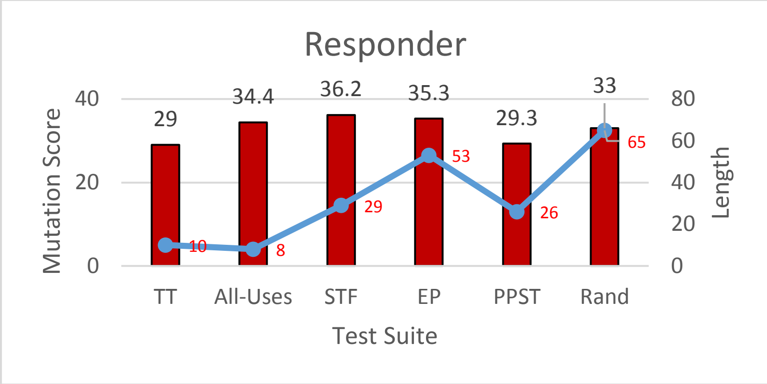

Figure 3.11 Responder Test Suites Mutation Score and Length

f) SCP Test Suites Fault Coverage Assessment

Figure 3.12 includes the mutation scores and length of each considered test suite for the SCP example. We note that there is no SITS test suite for the SCP as there are no state identifiers for the corresponding states.

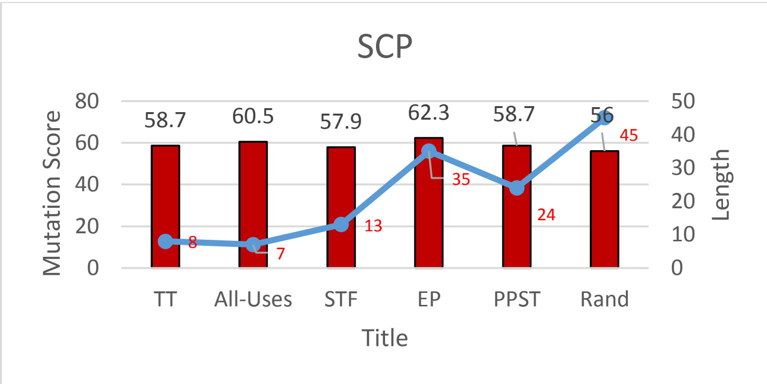

Figure 3.12 SCP Test Suites Mutation Score and Length

3.5.2 Fault Coverage of all Considered Examples

Figure 3.13 includes the average mutation scores and length of each considered test suite for all the above considered examples.

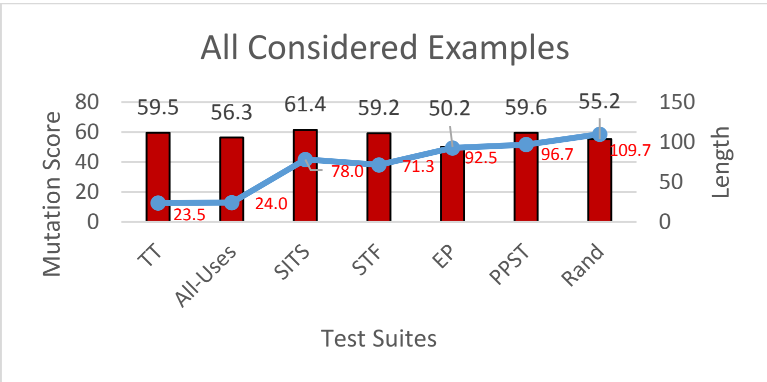

Figure 3.13 Test Suites Mutation Score and Length per all Examples

3.5.3 Ranking of Test Suites

Based on Figure 3.13, Table 3.1 depicts the ranking of test suites (1 – Best, 7 – Worst) using the mutation score of (1).

Table 3.1 Ranking based on Mutation score only

Ranking	Test Suite	Mutation Score
1	SITS	61.4 %
2	PPST	59.6 %
3	TT	59.5 %
4	STF	59.2 %
5	All-Uses	56.3 %
6	Rand	55.2 %
7	EP	50.2 %

Based on Figure 3.13, Table 3.2 depicts the ranking of test suites (1 – Best, 7 – Worst) using the coverage-length score of (2).

Table 3.2 Ranking based on Mutation score and Length

Ranking	Test Suite	Score
1	TT	250.6
2	All-Uses	232.1
3	STF	82.1
4	SITS	78.0
5	PPST	61.0
6	EP	53.7
7	Rand	49.8

The ranking of test suites shows that the best performing test suite in terms of mutation score of (1) is SITS. However, when considering the coverage-length score of (2), the TT and All-Uses test suites outperform the other test suites.

3.5.4 Assessment per Mutation Operators and Operator Categories

In this section, we assess the fault coverage of considered test suites per each mutation operator and per each operator category described in Table 2.1. We also assess the fault coverage per each mutation operator and per each operator category of all considered test suites. The fault coverage of an operator category per a particular test suite is calculated as the average of the fault coverage of corresponding test suites per all operators in the corresponding category. In contrast, the fault coverage of the mutation operator or category per all test suites is calculated as the average of fault coverage of all test suites per the corresponding mutation operator or category. In this section, the assessment is done based on the average results of three implementations of each considered EFSM specification.

a) TFTP Fault Coverage per Arithmetic, Conditional, and Relational Operators

Table 3.3 illustrates the mutation score of each test suite per each mutation operator for the TFTP example.

Table 3.3 TFTP Average Mutation Operator Coverage Table

TFTP	Arithmetic				Conditional			Relational
	AORB	AORS	AOIS	AOIU	COR	COI	COD	ROR
TT	100	100	51	100	39	100	100	74
All-Uses	100	100	58	100	25	90	100	80
SITS	100	100	51	100	50	100	100	82
STF	100	100	51	100	50	100	100	82
EP	75	100	50	67	51	91	100	69
PPST	100	100	51	100	50	100	100	76
Rand	0	0	21	0	40	72	0	46

(a.1) Assessment of Test Suites per each Mutation Operator

The Figure 3.14 depicts the coverage of each test suite in terms of mutation score per each arithmetic operator.

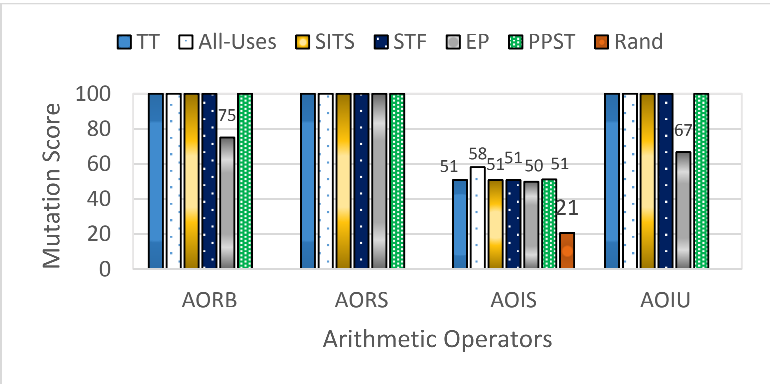

Figure 3.14 Coverage per Arithmetic Operators

The Figure 3.15 depicts the coverage of each test suite in terms of mutation score per each conditional operator.

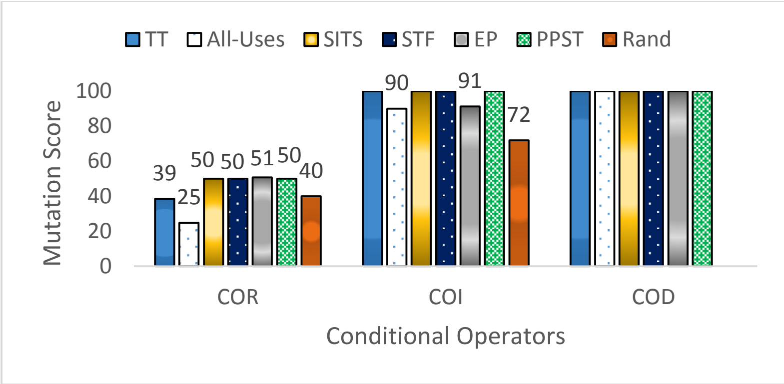

Figure 3.15 Coverage per Conditional Operators

The Figure 3.16 depicts the coverage of each test suite in terms of mutation score per ROR relational operator.

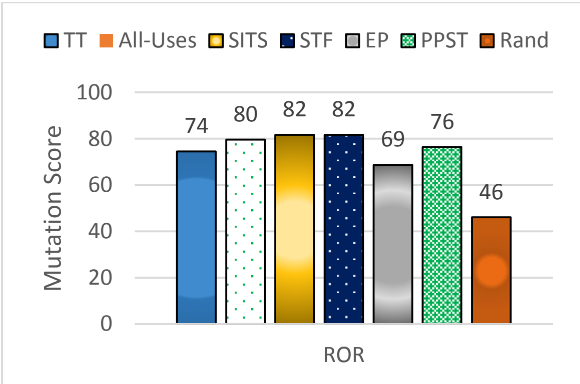

Figure 3.16 Coverage per ROR Relational Operators

(a.2) Average Assessment of each Mutation Operator

The Figure 3.17 depicts the coverage of all test suites in terms of mutation score per each mutation operator.

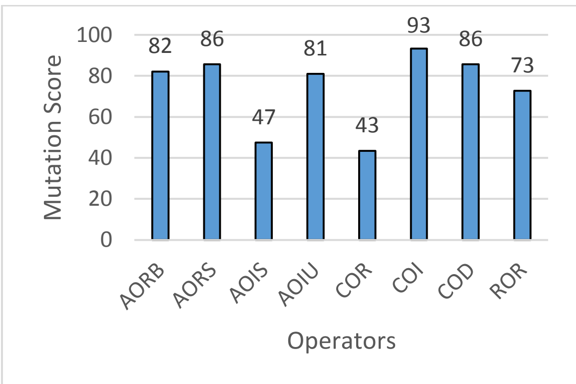

Figure 3.17 Operators Coverage over All Test Suites

(a.3) Assessment of Test Suites per each Mutation Operator Category

The Figure 3.18 depicts the coverage of each test suite in terms of mutation score per arithmetic operator category.

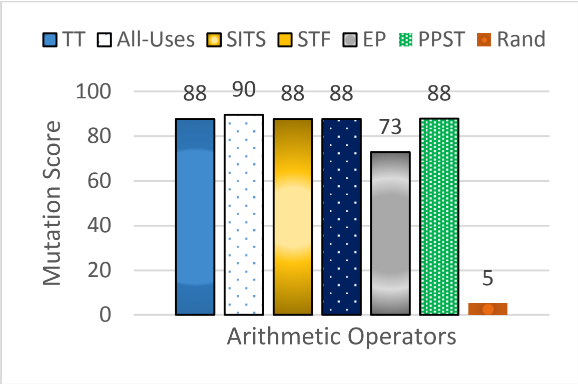

Figure 3.18 Coverage per All Arithmetic Operators

The Figure 3.19 depicts the coverage of each test suite in terms of mutation score per conditional operator category.

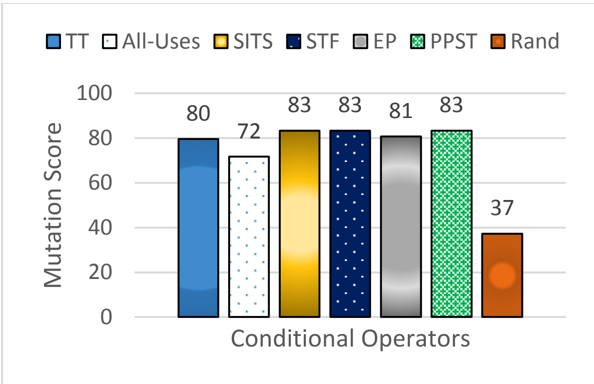

Figure 3.19 Coverage per All Conditional Operators

The Figure 3.20 depicts the coverage of each test suite in terms of mutation score per relational operator category.

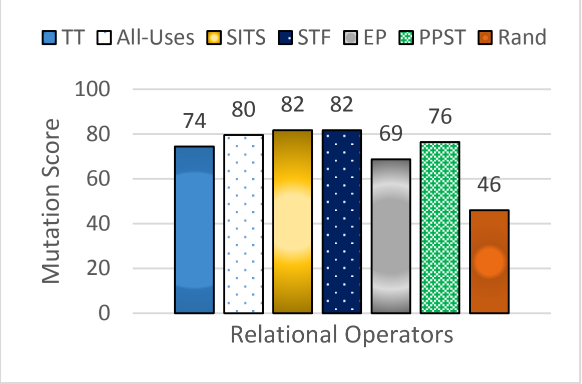

Figure 3.20 Coverage per All Relational Operators

(a.4) Average Assessment of each Mutation Operator Category

The Figure 3.21 depicts the coverage of all test suites in terms of mutation score per each mutation operator category.

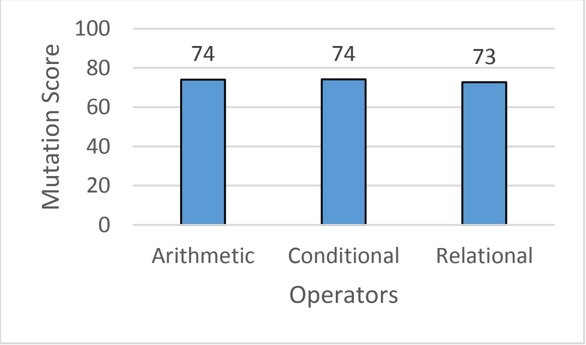

Figure 3.21 Coverage of Mutation Operator Categories of TFTP

b) CD Player Fault Coverage per Arithmetic, Conditional, and Relational Operators

Table 3.4 illustrates the mutation score of each test suite per each mutation operator for the CD Player example.

Table 3.4 CD Player Average Mutation Operator Coverage Table

CD	Arithmetic Operators		Conditional Operators			Relational Operators
	AORS	AOIS	COR	COI	COD	ROR
TT	86	79	58	97	100	64
SITS	57	56	58	100	100	67
STF	57	67	58	100	100	67
EP	29	28	58	81	100	43
PPST	100	81	58	100	100	68
Rand	100	89	58	100	100	67

(b.1) Assessment of Test Suites per each Mutation Operator

The Figure 3.22 depicts the coverage of each test suite in terms of mutation score per each arithmetic operator.

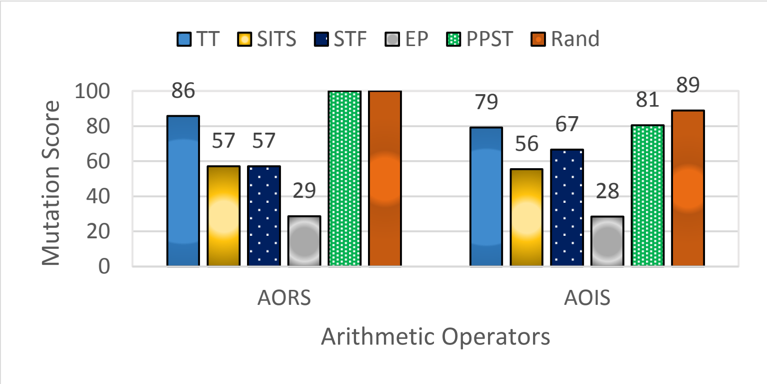

Figure 3.22 Coverage per Arithmetic Operators

The Figure 3.23 depicts the coverage of each test suite in terms of mutation score per each conditional operator.

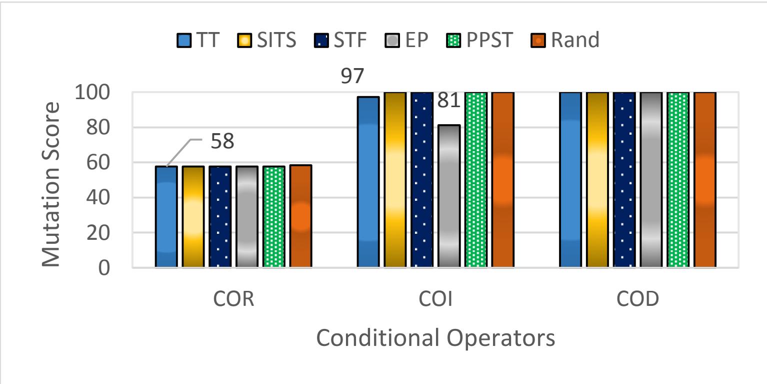

Figure 3.23 Coverage per Conditional Operators

The Figure 3.24 depicts the coverage of each test suite in terms of mutation score per ROR relational operator.

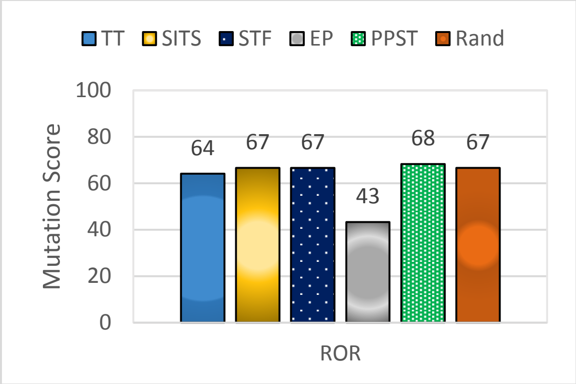

Figure 3.24 Coverage per ROR Relational Operators

(b.2) Average Assessment of each Mutation Operator

The Figure 3.25 depicts the coverage of all test suites in terms of mutation score per each mutation operator.

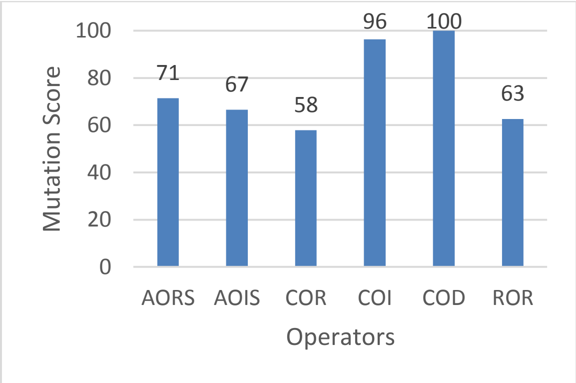

Figure 3.25 Operators Coverage over All Test Suites

(b.3) Assessment of Test Suites per each Mutation Operator Category

The Figure 3.26 depicts the coverage of each test suite in terms of mutation score per arithmetic operator category.

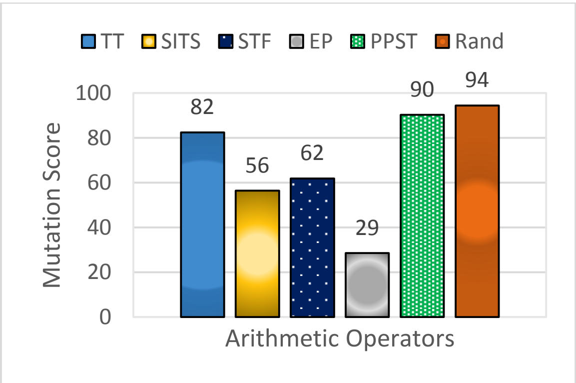

Figure 3.26 Coverage per All Arithmetic Operators

The Figure 3.27 depicts the coverage of each test suite in terms of mutation score per conditional operator category.

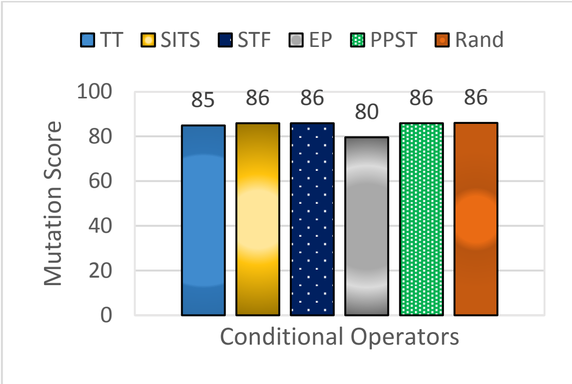

Figure 3.27 Coverage per All Conditional Operators

(b.4) Average Assessment of each Mutation Operator Category

The Figure 3.28 depicts the coverage of all test suites in terms of mutation score per each mutation operator category.

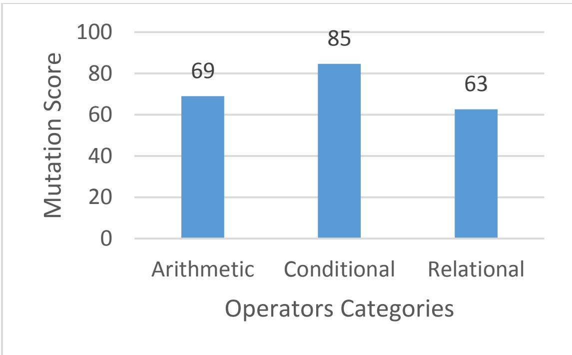

Figure 3.28 Coverage of Mutation Operators Categories of CD

c) POP3 Fault Coverage per Arithmetic, Conditional, and Relational Operators

Table 3.5 illustrates the mutation score of each test suite per each mutation operator for the POP3 example.

Table 3.5 POP3 Average Mutation Operator Coverage Table

POP3	Arithmetic	Conditional			Relational
	AOIS	COR	COI	COD	ROR
TT	42	25	94	100	63
All-Uses	44	7	94	100	58
SITS	50	25	93	100	59
STF	44	25	93	100	59
EP	50	23	94	100	59
PPST	50	25	94	100	63
Rand	56	13	89	100	60

(c.1) Assessment of Test Suites per each Mutation Operator

The Figure 3.29 depicts the coverage of each test suite in terms of mutation score per each arithmetic operator.

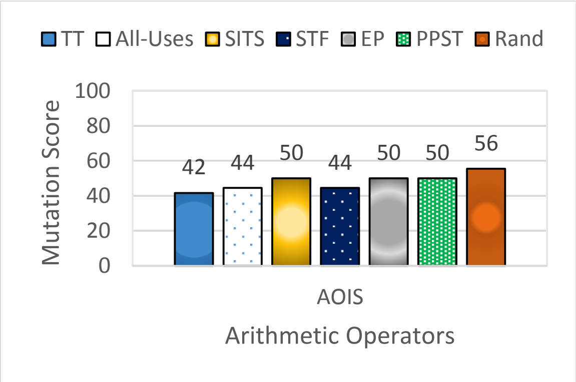

Figure 3.29 Coverage per Arithmetic Operators

The Figure 3.30 depicts the coverage of each test suite in terms of mutation score per each conditional operator.

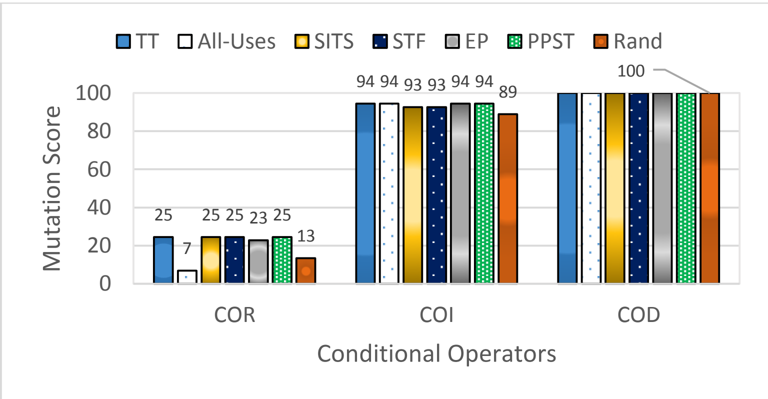

Figure 3.30 Coverage per Conditional Operators

The Figure 3.31 depicts the coverage of each test suite in terms of mutation score per ROR relational operator.

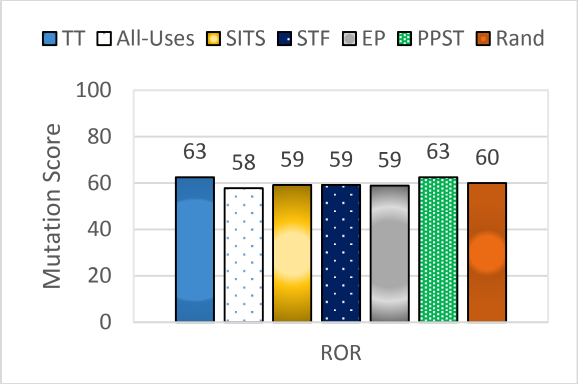

Figure 3.31 Coverage per ROR Relational Operators

(c.2) Average Assessment of each Mutation Operator

The Figure 3.32 depicts the coverage of all test suites in terms of mutation score per each mutation operator.

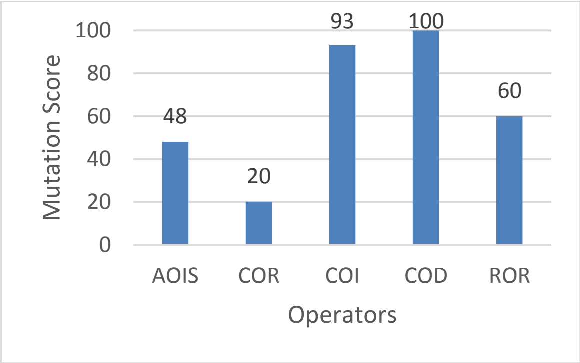

Figure 3.32 Operators Coverage over All Test Suites

(c.3) Assessment of Test Suites per each Mutation Operator Category

The Figure 3.33 depicts the coverage of each test suite in terms of mutation score per arithmetic operator category.

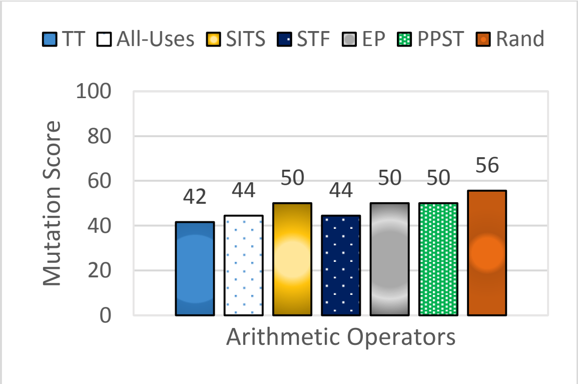

Figure 3.33 Coverage per All Arithmetic Operators

The Figure 3.34 depicts the coverage of each test suite in terms of mutation score per conditional operator category.

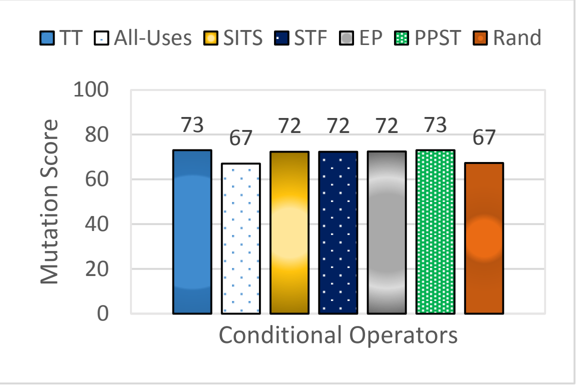

Figure 3.34 Coverage per All Conditional Operators

The Figure 3.35 depicts the coverage of each test suite in terms of mutation score per relational operator category.

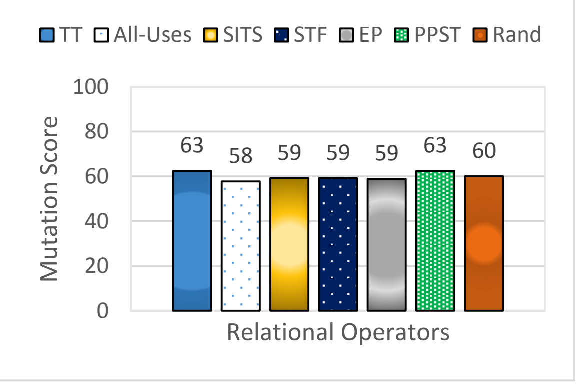

Figure 3.35 Coverage per All Relational Operators

(c.4) Average Assessment of each Mutation Operator Category

The Figure 3.36 depicts the coverage of all test suites in terms of mutation score per each mutation operator category.

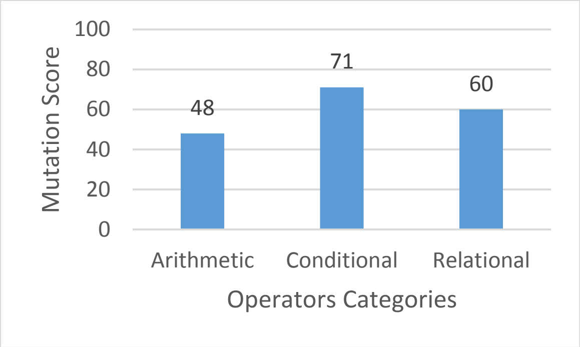

Figure 3.36 Coverage of Mutation Operators Categories of POP3

d) Initiator Fault Coverage per Arithmetic, Conditional, and Relational Operators

Table 3.6 illustrates the mutation score of each test suite per each mutation operator for the Initiator example.

Table 3.6 Initiator Average Mutation Operator Coverage Table

Initiator	Arithmetic				Conditional			Relational
	AORB	AORS	AOIS	AOIU	COR	COI	COD	ROR
TT	88	100	82	52	77	100	100	70
All-Uses	88	100	80	52	71	93	100	69
STF	58	67	73	34	77	100	100	70
EP	0	0	33	0	66	79	67	48
PPST	39	22	47	39	78	100	67	73
Rand	50	0	69	50	75	100	0	73

(d.1) Assessment of Test Suites per each Mutation Operator

The Figure 3.37 depicts the coverage of each test suite in terms of mutation score per each arithmetic operator.

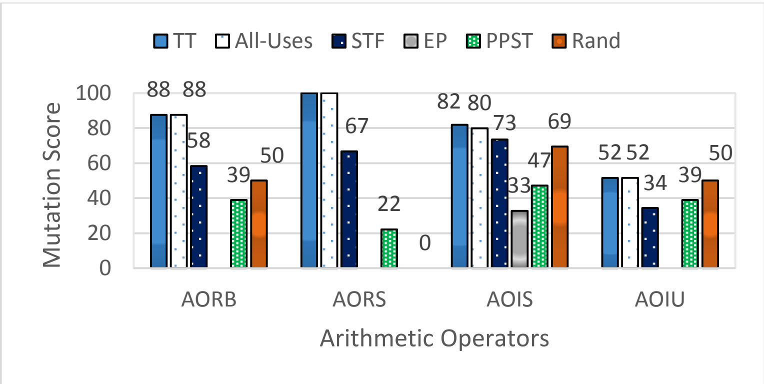

Figure 3.37 Coverage per Arithmetic Operators

The Figure 3.38 depicts the coverage of each test suite in terms of mutation score per each conditional operator.

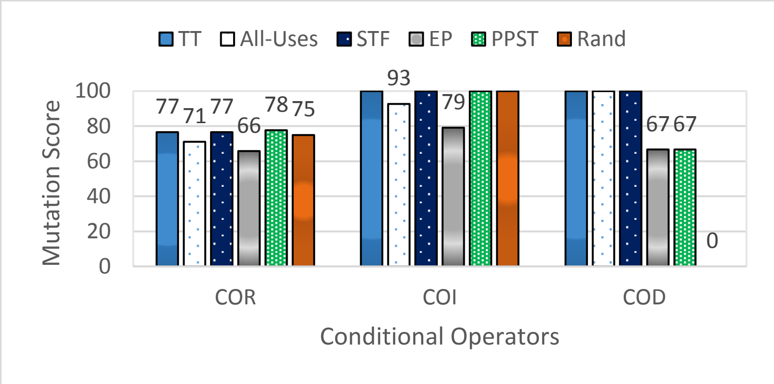

Figure 3.38 Coverage per Conditional Operators

The Figure 3.39 depicts the coverage of each test suite in terms of mutation score per ROR relational operator.

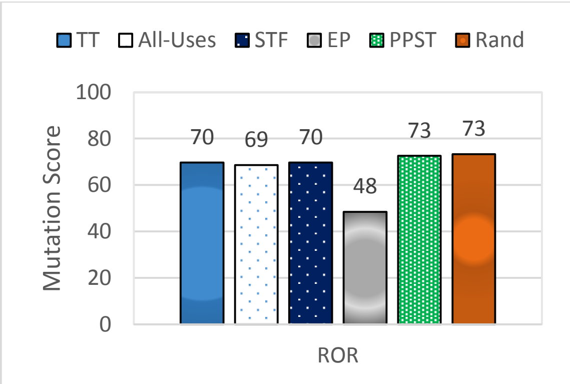

Figure 3.39 Coverage per ROR Relational Operators

(d.2) Average Assessment of each Mutation Operator

The Figure 3.40 depicts the coverage of all test suites in terms of mutation score per each mutation operator.

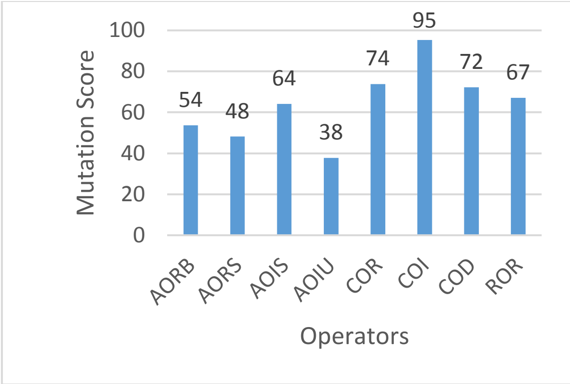

Figure 3.40 Operators Coverage over All Test Suites

(d.3) Assessment of Test Suites per each Mutation Operator Category

The Figure 3.41 depicts the coverage of each test suite in terms of mutation score per arithmetic operator category.

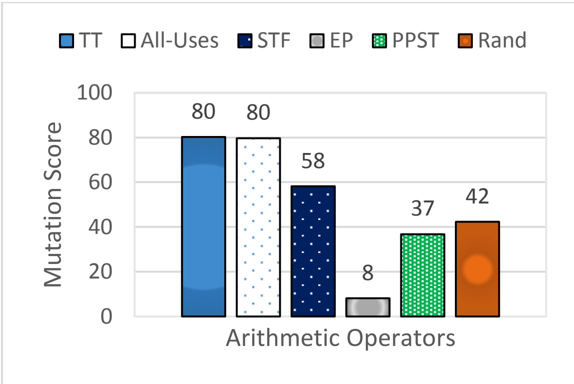

Figure 3.41 Coverage per All Arithmetic Operators

The Figure 3.42 depicts the coverage of each test suite in terms of mutation score per conditional operator category.

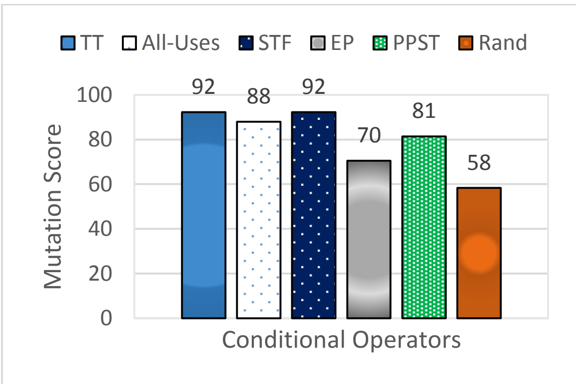

Figure 3.42 Coverage per All Conditional Operators

The Figure 3.43 depicts the coverage of each test suite in terms of mutation score per relational operator category.

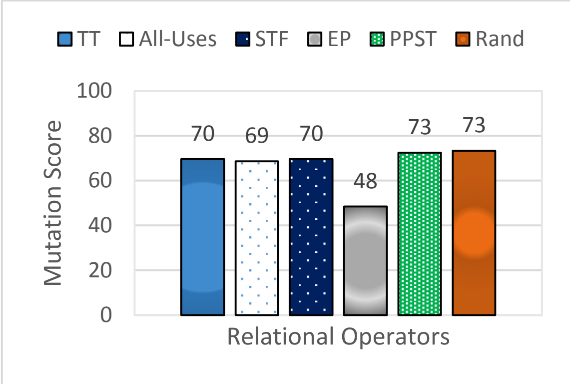

Figure 3.43 Coverage per All Relational Operators

(d.4) Average Assessment of each Mutation Operator Category

The Figure 3.44 depicts the coverage of all test suites in terms of mutation score per each mutation operator category.

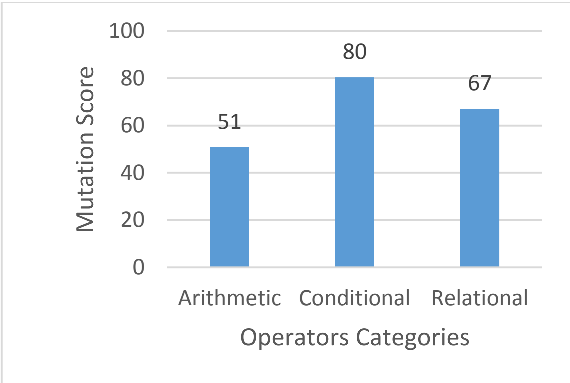

Figure 3.44 Coverage of Mutation Operators Categories of Initiator

e) Responder Fault Coverage per Arithmetic, Conditional, and Relational Operators

Table 3.7 illustrates the mutation score of each test suite per each mutation operator for the Responder example.

Table 3.7 Responder Average Mutation Operator Coverage Table

Responder	Arithmetic	Conditional			Relational
	AOIS	COR	COI	COD	ROR
TT	28	54	43	100	11
All-Uses	36	38	52	100	17
STF	36	54	52	100	17
EP	29	54	52	33	19
PPST	19	54	52	33	16
Rand	38	50	60	0	0

(e.1) Assessment of Test Suites per each Mutation Operator

The Figure 3.45 depicts the coverage of each test suite in terms of mutation score per each arithmetic operator.

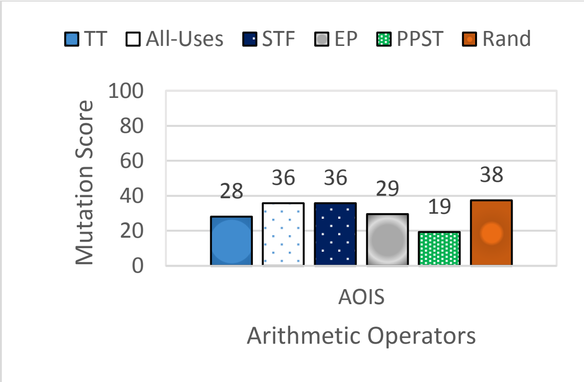

Figure 3.45 Coverage per Arithmetic Operators

The Figure 3.46 depicts the coverage of each test suite in terms of mutation score per each conditional operator.

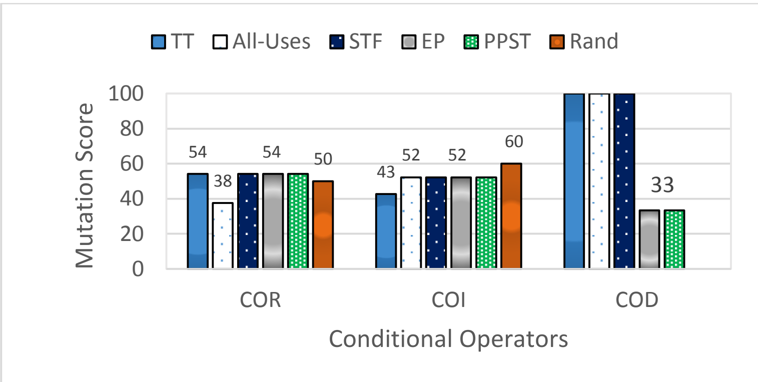

Figure 3.46 Coverage per Conditional Operators

The Figure 3.47 depicts the coverage of each test suite in terms of mutation score per ROR relational operator.

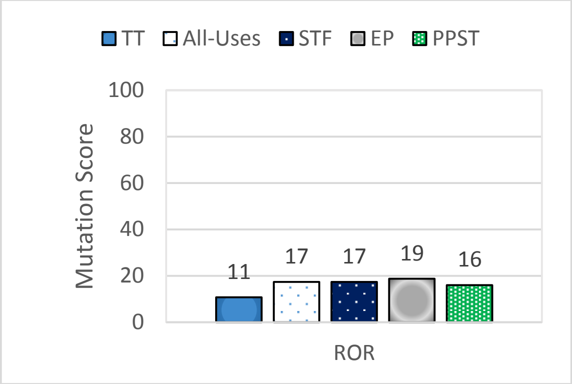

Figure 3.47 Coverage per ROR Relational Operators

(e.2) Average Assessment of each Mutation Operator

The Figure 3.48 depicts the coverage of all test suites in terms of mutation score per each mutation operator.

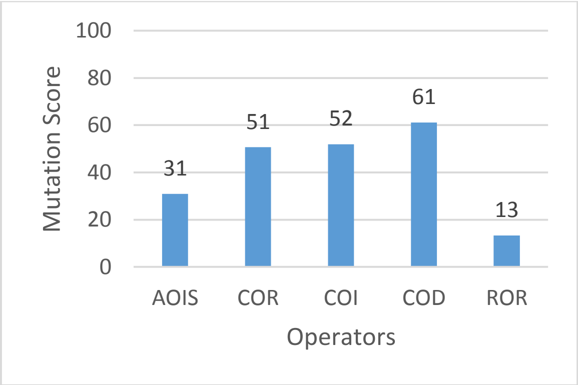

Figure 3.48 Operators Coverage over All Test Suites

(e.3) Assessment of Test Suites per each Mutation Operator Category

The Figure 3.49 depicts the coverage of each test suite in terms of mutation score per arithmetic operator category.

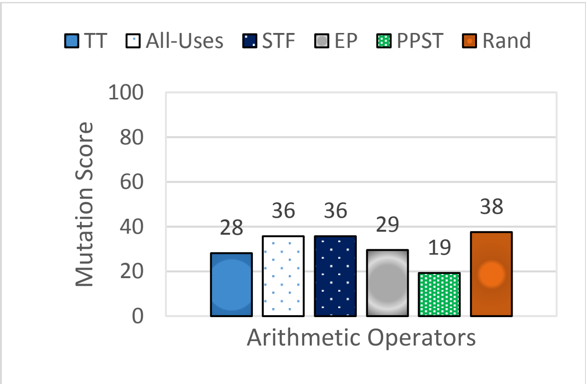

Figure 3.49 Coverage per All Arithmetic Operators

The Figure 3.50 depicts the coverage of each test suite in terms of mutation score per conditional operator category.

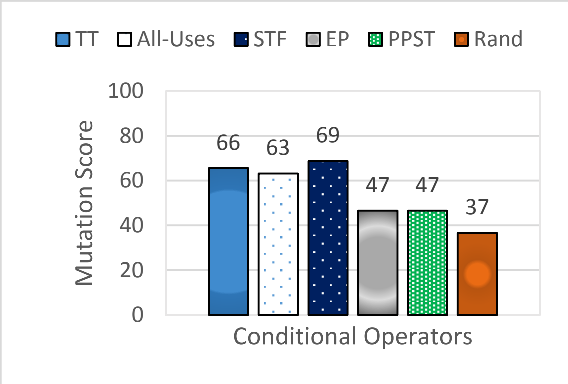

Figure 3.50 Coverage per All Conditional Operators

The Figure 3.51 depicts the coverage of each test suite in terms of mutation score per relational operator category.

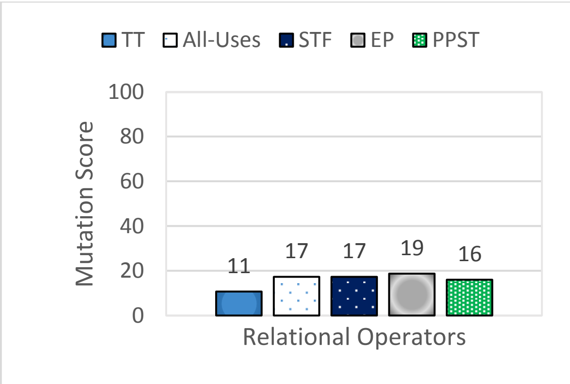

Figure 3.51 Coverage per All Relational Operators

(e.4) Average Assessment of each Mutation Operator Category

The Figure 3.52 depicts the coverage of all test suites in terms of mutation score per each mutation operator category.

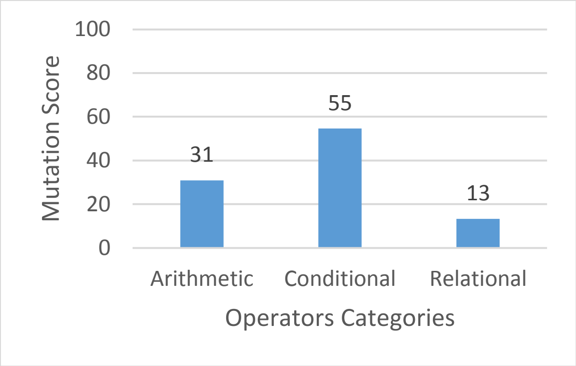

Figure 3.52 Coverage of Mutation Operators Categories of Responder

f) SCP Fault Coverage per Arithmetic, Conditional, and Relational Operators

Table 3.8 illustrates the mutation score of each test suite per each mutation operator for the SCP example.

Table 3.8 SCP Average Mutation Operator Coverage Table

SCP	Arithmetic				Conditional			Relational
	AORB	AORS	AOIS	AOIU	COR	COI	COD	ROR
TT	25	100	61	50	49	75	93	46
All-Uses	63	100	60	33	49	81	93	53
STF	38	100	64	0	49	73	93	43
EP	13	67	67	11	49	86	62	56
PPST	13	67	60	11	49	86	62	56
Rand	13	67	85	0	47	83	7	47

(f.1) Assessment of Test Suites per each Mutation Operator

The Figure 3.53 depicts the coverage of each test suite in terms of mutation score per each arithmetic operator.

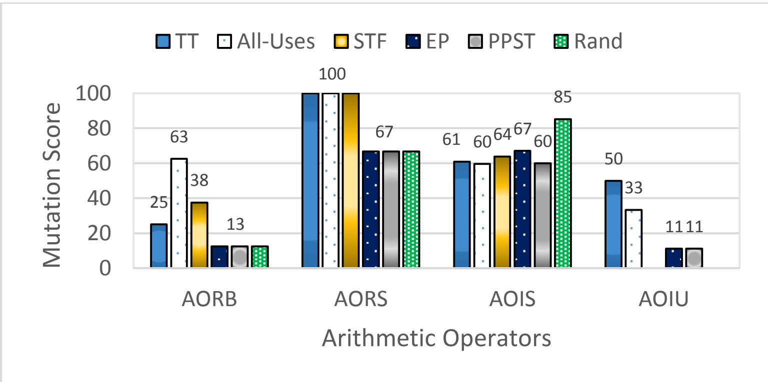

Figure 3.53 Coverage per Arithmetic Operators

The Figure 3.54 depicts the coverage of each test suite in terms of mutation score per each conditional operator.

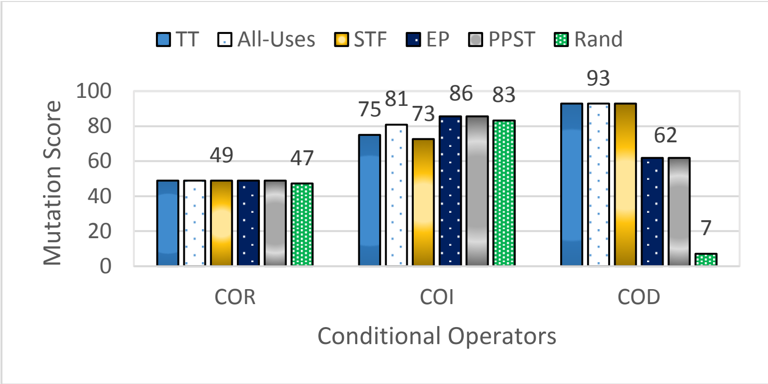

Figure 3.54 Coverage per Conditional Operators

The Figure 3.55 depicts the coverage of each test suite in terms of mutation score per ROR relational operator.

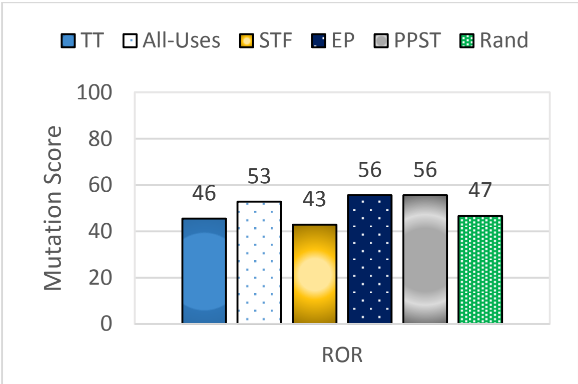

Figure 3.55 Coverage per ROR Relational Operators

(f.2) Average Assessment of each Mutation Operator

The Figure 3.56 depicts the coverage of all test suites in terms of mutation score per each mutation operator.

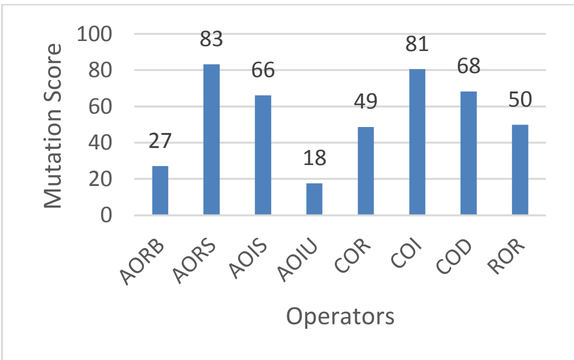

Figure 3.56 Operators Coverage over All Test Suites

(f.3) Assessment of Test Suites per each Mutation Operator Category

The Figure 3.57 depicts the coverage of each test suite in terms of mutation score per arithmetic operator category.

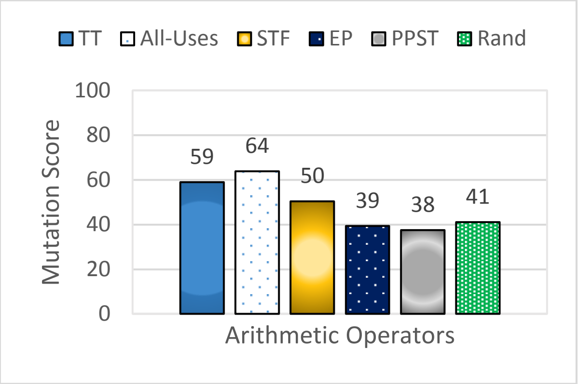

Figure 3.57 Coverage per All Arithmetic Operators

The Figure 3.58 depicts the coverage of each test suite in terms of mutation score per conditional operator category.

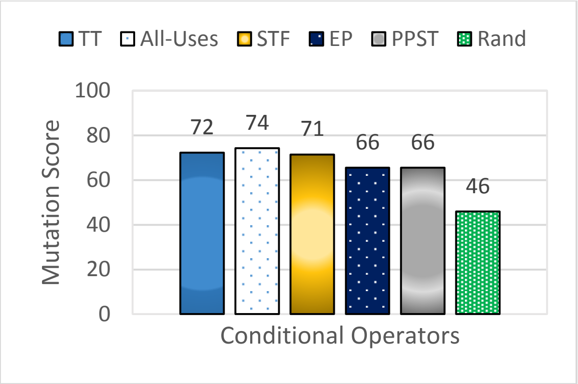

Figure 3.58 Coverage per All Conditional Operators

The Figure 3.59 depicts the coverage of each test suite in terms of mutation score per relational operator category.

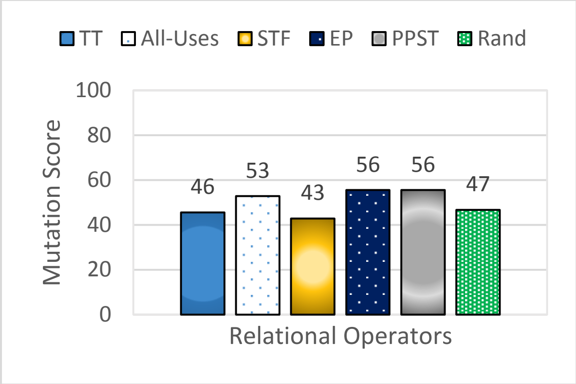

Figure 3.59 Coverage per All Relational Operators

(f.4) Average Assessment of each Mutation Operator Category

The Figure 3.60 depicts the coverage of all test suites in terms of mutation score per each mutation operator category.

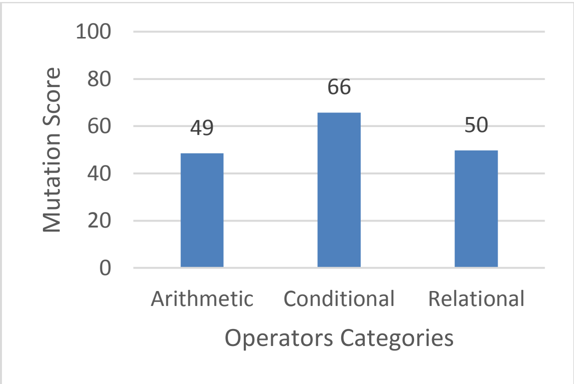

Figure 3.60 Coverage of Mutation Operators Categories of SCP

3.5.5 All Considered Examples Fault Coverage per Arithmetic, Conditional, and Relational Operators

Table 3.9 illustrates the mutation score of each test suite per each mutation operator for all considered examples.

Table 3.9 All Examples Average Mutation Operator Coverage Table

All	Arithmetic				Conditional			Relational
	AORB	AORS	AOIS	AOIU	COR	COI	COD	ROR
TT	71	96	57	67	50	85	99	55
All-Uses	83	100	56	62	38	82	99	55
SITS	100	79	52	100	44	98	100	69
STF	65	81	56	45	52	86	99	56
EP	29	49	43	26	50	81	77	49
PPST	50	72	51	50	52	89	77	59
Rand	18	50	56	13	49	83	48	51

(a) Assessment of Test Suites per each Mutation Operator

The Figure 3.61 depicts the coverage of each test suite in terms of mutation score per each arithmetic operator.

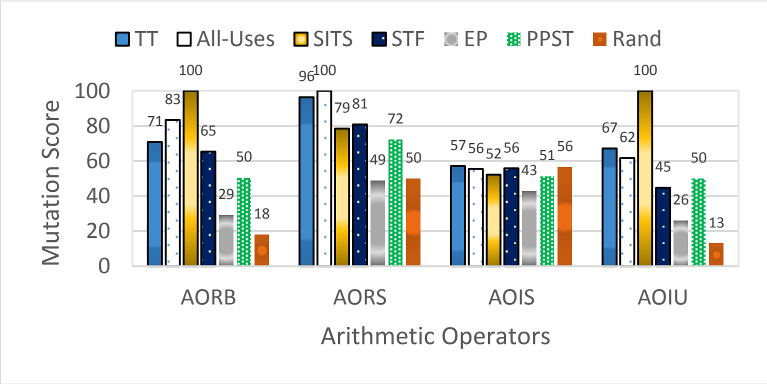

Figure 3.61 Coverage per Arithmetic Operators

The Figure 3.62 depicts the coverage of each test suite in terms of mutation score per each conditional operator.

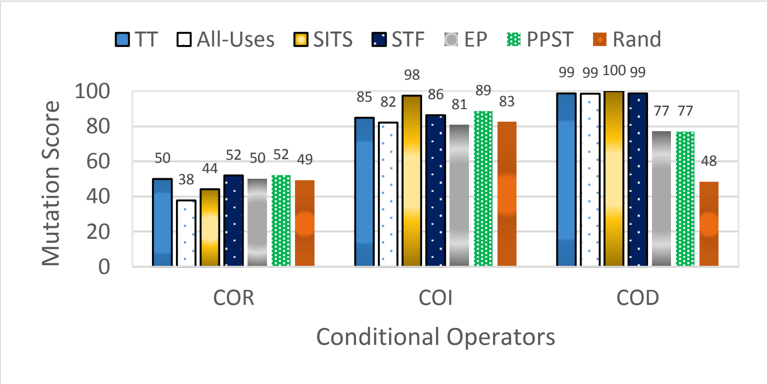

Figure 3.62 Coverage per Conditional Operators

The Figure 3.63 depicts the coverage of each test suite in terms of mutation score per ROR relational operator.

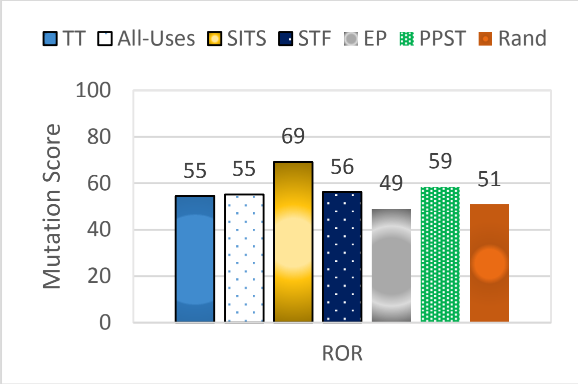

Figure 3.63 Coverage per ROR Relational Operators

(b) Average Assessment of each Mutation Operator

The Figure 3.64 depicts the coverage of all test suites in terms of mutation score per each mutation operator.

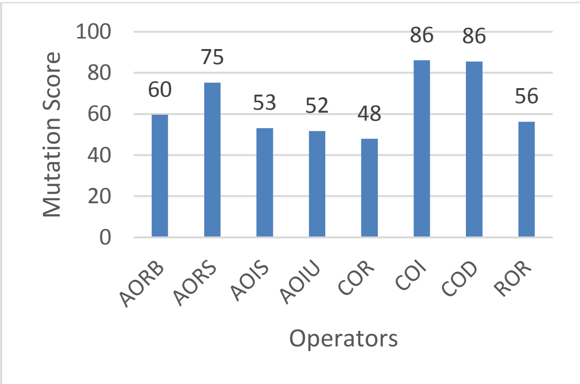

Figure 3.64 Operators Coverage over All Test Suites

According to the results depicted in Figure 3.64, we notice that the best mutation operators over all test suites are COI and COD Conditional Operators (86 %).

(c) Assessment of Test Suites per each Mutation Operator Category

The Figure 3.65 depicts the coverage of each test suite in terms of mutation score per arithmetic operator category.

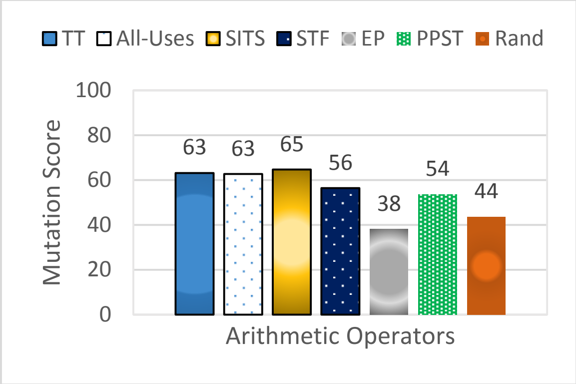

Figure 3.65 Coverage per All Arithmetic Operators

The Figure 3.66 depicts the coverage of each test suite in terms of mutation score per conditional operator category.

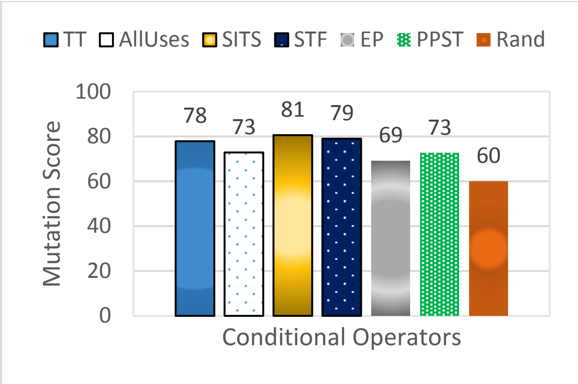

Figure 3.66 Coverage per All Conditional Operators

The Figure 3.67 depicts the coverage of each test suite in terms of mutation score per relational operator category.

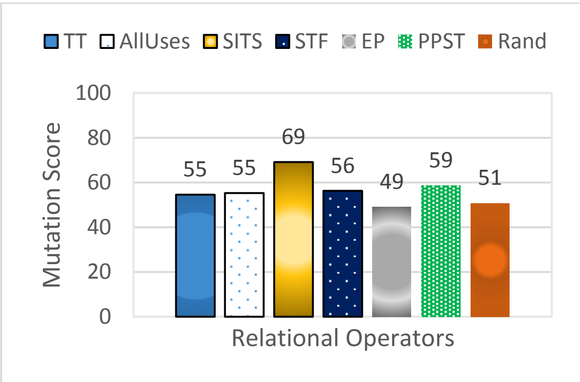

Figure 3.67 Coverage per All Relational Operators

(d) Average Assessment of each Mutation Operator Category

The Figure 3.68 depicts the coverage of all test suites in terms of mutation score per each mutation operator category.

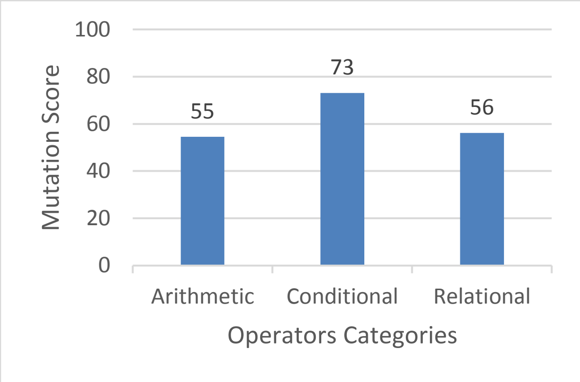

Figure 3.68 Coverage of Mutation Operators categories of All Examples

According to the results depicted in Figure 3.68, we notice that the best mutation operator category is the Conditional Category (73 %).

3.5.6 Ranking of Test Suites per Mutation Operators and per Operator Categories

In this section, we rank the test suites per mutation operators and categories based on the score of (1) and (2) described before.

(a) Ranking of Test Suites per each Mutation Operator Based on (1) Score

Table 3.10 includes the ranking of test suites per each mutation operator based on the average mutation score per operator only.

Table 3.10 Ranking per Operators based on Average Mutation score per Operator only

Rank	AORB	AORS	AOIS	AOIU	COR	COI	COD	ROR
1	SITS	All-Uses	TT	SITS	PPST STF	SITS	SITS	SITS
2	All-Uses	TT	Rand STF All-Uses	TT	TT EP	PPST	TT STF All-Uses	PPST STF
3	TT	STF	SITS	All-Uses	Rand	STF	EP PPST	All-Uses TT
4	STF	SITS	PPST	PPST	SITS	TT	Rand	Rand
5	PPST	PPST	EP	STF	All-Uses	Rand		EP
6	EP	Rand		EP		All-Uses		
7	Rand	EP		Rand		EP		

(b) Ranking of Test Suites per each Operator Category Based on (1) Score

Table 3.11 includes the ranking of test suites per each operator category based on the average mutation score per operator category only.

Table 3.11 Ranking per Category based on Average Mutation Score per Category only

Rank	Arithmetic	Conditional	Relational
1	SITS	SITS	SITS
2	TT All-Uses	STF	PPST
3	STF	TT	STF
4	PPST	All-Uses PPST	All-Uses TT
5	Rand	EP	Rand
6	EP	Rand	EP

(c) Ranking of Test Suites per each Mutation Operator Based on (2) Score

Table 3.12 includes the ranking of test suites per each mutation operator based on the average mutation score per operator and length of the corresponding test suite.

Table 3.12 Ranking per Operator based on Average Mutation Score per Operator and Length

Rank	AORB	AORS	AOIS	AOIU	COR	COI	COD	ROR
1	All-Uses	All-Uses	TT	TT	TT	TT	TT	TT
2	TT	TT	All-Uses	All-Uses	All-Uses	All-Uses	All-Uses	All-Uses
3	SITS	STF	STF	SITS	STF	SITS	STF	SITS
4	STF	SITS	SITS	STF	SITS	STF	SITS	STF
5	PPST	PPST	PPST	PPST	EP	PPST	EP	PPST
6	EP	EP	Rand	EP	PPST	EP	PPST	EP
7	Rand	Rand	EP	Rand	Rand	Rand	Rand	Rand

(d) Ranking of Test Suites per each Operator Category Based on (2) Score

Table 3.13 includes the ranking of test suites per each operator category based on the average mutation score per operator category and length of the corresponding test suite.

Table 3.13 Ranking per Category based on Average Mutation Score per Category and Length

Rank	Arithmetic	Conditional	Relational
1	TT	TT	TT
2	All-Uses	All-Uses	All-Uses
3	SITS	STF	SITS
4	STF	SITS	STF
5	PPST	PPST	PPST
6	EP	EP	EP
7	Rand	Rand	Rand

3.5.7 Summary of All Obtained Results

Below we include a summary of the experimental results in Sections 3.5.1 to 3.5.6

- The best performing test suites in terms of fault coverage are the SITS (61.4 %) followed by the PPST (59.6 %), TT (59.5 %), STF (59.2 %), All-Uses (56.3 %), Rand (55.2 %) then the EP (50.2 %) test suites. However, when considering the *coverage-length* score, the TT

(250.59) and All-Uses (232.15) test suites have comparable scores, and they outperform the other test suites by approximately 73 percent. The STF (82.11), SITS (77.99), PPST (60.99), EP (53.72) and Rand (49.80) test suites have comparable scores, but each of these test suites score less than the TT and the All-Uses test suites by approximately 73 percent.

- Test suite fault coverage of COI and COD faults is on average 86 %, and it is significantly higher than the coverage of mutants with other types of operator faults by approximately 29 percent. Test suite coverage of AORS, AORB, AOIS, AOIU, ROR and COR faults are comparable, but this coverage is less than the coverages of COI and COD by approximately 29 percent. Test suite coverage of conditional faults (73 %) is significantly higher than the coverage of mutants with arithmetic and relational faults by approximately 17.5 percent. Test suite coverage of mutants with arithmetic faults is comparable to the coverage of mutants with relational faults, but this coverage is less than the coverage of conditional faults by approximately 17.5 percent.
- SITS test suites have the best fault coverage of arithmetic faults (65 %), conditional faults (81 %) and relational faults (69 %). The remaining test suites have comparable coverages in terms of arithmetic, conditional and relational faults, but their coverage is less than the coverage of the SITS test suites by approximately 12 percent. When considering the *coverage-length* score, the TT and All-Uses test suites have comparable scores, and they outperform the other test suites in terms of score of arithmetic and conditional and relational faults by approximately 74 percent. The remaining test suites have comparable scores but they are less than the scores of the TT and All-Uses test suites by approximately 74 percent.

3.6 Related Work on Assessment of the Fault Coverage of Test Suites

Empirical assessment studies related to the work presented in this thesis are mostly summarized in [2, 3, 4, 5, 11, 12, 13, 14, 15, 16, 17]. In summary, the studies reported in [18, 19, 16, 20] consider code-based mutation testing and the All-Uses criterion. Li et al. [14] conduct code-based experiments using code-based mutation, EP, All-Uses and the PP coverage criteria. Aynur et al. [12] compare three specification-based criteria, namely, the full predicate, transition-pair and some specification-based mutation criteria. Assessment of tests from different UML diagrams using the full predicate and message sequence path coverage are reported in [21].

For specifications modeled as EFSMs, a preliminary assessment of STF, TT, All-Uses, All-Predicates, double transfer faults and some random test suites has been recently presented in [6]. The study considers three known EFSM specifications and analytically compares the effectiveness of many test selection criteria in covering EFSM mutants of these specifications with single and double transfer faults. The coverage of some randomly generated test suites from the given specifications with the same length of test cases as the All-Uses (All-Transitions) test suite is assessed. The results of the study showed that the best performing test suites in terms of fault coverage of EFSM mutants with transfer faults were the DTF, TT, All-Transitions, All-Uses followed by the All-Predicates test suites. Moreover, the test suites have approximately 14% more coverage of double transfer faults than single transfer faults of the considered EFSM specification. On one hand, random test suites with the same length of test cases as All-Uses test suites have comparable fault coverages; on the other hand, All-Transitions test suites outperform random test suites (with the same length of test cases).

Recently, another study has been presented in [7] [44], but unlike [6], the considered test suites are assessed in terms of their coverage of code mutants of implementations of these specifications, which allows comparing the coverage of considered test suites w.r.t. traditional code-based types of mutants. Additionally, in [7] [44], SITS test suites are also considered in the assessment. The results of the study showed that All-Uses, STF, and TT test suites provide comparable (fault) coverages and SITSs outperform all other considered test suites. An analysis of one type of random test suite is considered, namely, random tests with same length as other EFSM tests. The results of random test suites showed that the Random-All-Uses and All-Uses test suites provide comparable coverage where SITS test suites slightly outperform Random- SITSs. Furthermore, results showed that test suite coverage of conditional faults is significantly higher than their coverage of mutants with arithmetic, logical, or relational faults. Test suite coverage of mutants with relational faults is much less than that of the coverage of mutants with arithmetic, conditional or logical faults. The All-Uses and TT test suites both achieve comparable coverages of mutants with arithmetic faults; however, STF test suites have significantly lower coverage of arithmetic faults than All-Uses and TT test suites. All considered, test suites provide comparable coverages of conditional faults; also, all test suites provide comparable coverages of logical (or relational) faults. However, in [7] [44], only three application examples are considered in the study and as reported in [7] [44] there is a need to consider more application examples to verify the results and also there is a need to consider more types of EFSM test suites. Accordingly, in the first part of this thesis, we conduct experiments to assess the fault coverage of many EFSM-based test suites as done in [7] [44]; however, our study considers six working examples including the three used in [7] [44]. In addition, we consider more types of EFSM test suites, namely, EP and PPST test suites. Furthermore, in this thesis, a comprehensive assessment of the fault coverage of random test suites is carried out. According to [7] [44], the best test suite in terms of mutation score is SITS (67.29%) followed by the TT (62.22%), STF (60.35%) then the All-Uses (60.08 %) test suites, which is same as the pattern of results concluded in this thesis. According to [7] [44], test suite coverage of conditional faults (76.44%) is higher than their coverage of mutants with arithmetic faults (61.1%) and relational faults (43.84%), which is same as the pattern of results concluded in this thesis.

Chapter 4: Fault Diagnosis Capability of Test Suites

In this chapter, we introduce EFSM-based fault diagnosis, and propose a simple algorithm that can be used for locating a faulty EFSM IUT and assessing the fault diagnosis capabilities of the EFSM test suites considered in the previous chapters. In addition, for each considered test suite, the algorithm determines the diagnostic tests (if any) needed for locating the given faulty IUT. Furthermore, an assessment of the test suites based on both the test suite length and the corresponding diagnostic test suite are carried out. Two criteria and introduced for comparing and ranking the test suites fault localization capabilities.

4.1 Introduction to Fault Diagnosis

While the purpose of FSM-based (conformance) testing is to check whether an implementation is different from its specification, an interesting complementary, yet more complex, step is to locate the differences between a specification and its implementation. The purpose of fault diagnosis (or diagnostic testing) is to locate the differences between a specification and its implementation, when the implementation is found to be faulty. Fault diagnosis has various applications, for example, it facilitates the job of correcting a protocol implementation so that it conforms to its specification.

In FSM-based and EFSM-based diagnosis, in general, a diagnostic method (or algorithm) that localizes the faulty mutant EFSM IUT includes the following steps: Given an EFSM specification *spec* and a test suite *TS* derived from *spec*, and a black-box faulty IUT EFSM, locating/determining the faulty IUT is carried out using the following steps:

1. **Generation of diagnostic candidates**: Diagnostic candidates are EFSM mutants of *spec* that are suspected to be faulty. These candidates are distinguishable from the *spec*.
2. **Discrimination between candidates and the Given IUT**: Once the step of candidate generation terminates, we often end up with a huge number of diagnostic candidates. The given test suite *TS* and the observed behavior of applying *TS* on the faulty IUT can be used to reduce the number of candidates,

where each candidate that produces an output to a test case of *TS* is different than that observed by applying the test case to the given IUT is removed.

3. **Generation of diagnostic tests**: If not all candidates are removed by the above step, additional tests, called *diagnostic tests*, are derived to locate the faulty component as follows: For every two candidates, we derive a test that distinguishes these candidates (if possible), run the obtained test on the given IUT, and based on the observed output, either one or the two candidates are eliminated. The process repeats till only one (or many indistinguishable) candidates remain in the set of possible candidates, and thus the fault candidate (or the set of non-distinguishable candidates) is located.

4.2 Assignment Faults and Output Parameter Faults

The derivation of diagnostic candidates is carried out based on the types of EFSM-based faults. In Chapter 3, we described single and double transfer faults and corresponding mutants (diagnostic candidates), and below we describe other types of faults considered in this thesis.

Given an EFSM *M*, a transition $t = (s, x, P, op, y, up, s')$ of an EFSM IUT *M´* has an *assignment fault* if it has an update statement that is different from that specified by *M*, i.e., *M´* has a transition $(s, x, P, op, y, up', s')$, $up' \neq up$. Such *M´* is a *mutant of M with an assignment fault* (SAF). In this thesis, we consider the following traditional types of single assignment faults and mutants with single assignment faults:

- Single Assignment Insertion (SAI): A transition *t* with an update statement *up'* of *M´* has an *assignment insertion fault* if an update statement (defined only over the context variable of *M*) of some transition (other than *t*) in *M*, is added to the update statements *up'*, while the added update statement is not in *up*. Such *M´* is a *mutant of M with a single assignment Insertion fault* (SAI).
- Single Assignment Deletion (SAD): A transition *t* with an update states *up'* of *M´* has an *assignment deletion fault* if one update statement in *up* of *M* is deleted and thus it is no longer in *up'*. Such *M´* is a *mutant of M with a single assignment Deletion fault* (SAD).
- Single Assignment Right-hand-side Fault (SARHS): A transition *t* with an update states *up'* of *M´* has a *right-hand-side assignment fault* if the right-hand-side (RHS) of one of the update statements in *up'* is different than that of

up. That is, if a context variable of *M* in the RHS of *up* is added (or deleted) to (from) *up'*, or if the value of a constant in the RHS of *up* is deleted (or changed to another value) in *up'*. Such *M´* is a *mutant of M with a single assignment Right-hand-side fault* (SARHS).

Another type of fault called a single output parameter fault (SOPF) is considered in this thesis. Given an EFSM *M*, a transition *t* = (*s*, *x*, *P*, *op*, *y*, *up*, *s´*) of an EFSM IUT *M´* has an *output parameter* fault if an output parameter of *op* defined over a context variable (or a constant) is replaced by another context variable or a constant, i.e., *M´* has a transition (*s*, *x*, *P*, *op*, *y*, *up´*, *s´*), $op' \neq op$. Such *M´* is a *mutant of M with a single output parameter fault*.

4.3 EFSM Based Fault Diagnosis Candidates

Given EFSM specification *M* and types of EFSM based faults and mutants described above and in Chapter 3, we define the following types of diagnostic candidates considered in our work.

- STF Diagnostic Candidates: includes as candidates all STF mutants of *M*.
- DTF Diagnostic Candidates: includes as candidates all DTF mutants of *M*.
- STF/DTF Diagnostic Candidates: includes all STF candidates and DTF candidates of *M*.
- SAI Diagnostic Candidates: includes as candidates all SAI mutants of *M*.
- SAD Diagnostic Candidates: includes as candidates all SAD mutants of *M*.
- SARHS Diagnostic Candidates: includes as candidates all SARHS mutants of *M*.
- STF/SAF Diagnostic Candidates: includes all STF candidates and SAF candidates of *M*.
- DTF/SAF Diagnostic Candidates: includes all DTF candidates and SAF candidates of *M*.
- SOPF Diagnostic Candidates: includes as candidates all SOPF mutants of *M*.

4.4 Assessing the Fault Diagnosis Capabilities of EFSM Test Suites

In this thesis, we assess the fault diagnostic capabilities of the TT, All-Uses, STF, SITS, EP, and the PPST test suites based on locating given sets of diagnostic candidates derived as described above. In particular, we determine and compare the fault diagnosis capabilities of these test suites using the two criteria, called FD1 and FD2, described below. In addition, for every considered test suite, we determine the additional tests (i.e. the diagnostic tests) requited for locating the faulty IUT. This allows us to compare the EFSM test suites in terms of how many tests required are required in total, i.e. using the test suites and the diagnostic tests, for locating the faulty IUT (or the class of candidates that are indistinguishable from the give IUT).

Given the EFSM specification *spec* and a set of EFSM and a set of fault candidates $D= D_1, D_2 \dots D_n$, First, we determine the candidates of D that that are distinguishable from *spec*. Then, we distribute these candidates into different classes, such that each class contains the fault candidates that are distinguishable from the given specification, yet all candidates in a class are indistinguishable from each other, whereas, all the fault candidates in a class are distinguishable from the given specification and are also distinguishable from each other fault candidate in a different class.

Given TS= {TT, All-Uses, SITS, STF, EP, PPST}, we study the fault diagnosis capability of each $TS_j \in TS$ over the fault candidates described above. Fault diagnosis capabilities are determined as the following:

- FD1 (score): Fault Diagnosis Capability of TS_j considering one fault candidate per each class. This determines the capability of a test suite to locate/determine which class a faulty machine is in. Thus, this determines the fault localization capability of a test suite up to sets of equivalent classes of indistinguishable candidates.
- FD2 (score): Fault Diagnosis Capability of TS_j considering all fault candidates per each class. This determines the capability of a test suite in locating/determining the faulty machine using all possible diagnostic fault candidates in all classes.
- FD3 (score): Fault Diagnosis Capability of $TS_j \cup TS_{j'}$ considering one fault candidate per each class, where $TS_{j'}$ is a diagnostic test suite required in addition to a given test suite, named TS_j, for complete fault diagnosis per a class of fault candidates, that is for determining in which class the fault machine is in.

- FD4 (score): Fault Diagnosis Capability of $TS_j \cup TS_{j'}$ considering all fault candidates per each class, where $TS_{j'}$ is a diagnostic test suite required in addition to a given test suite, named TS_j, for complete fault diagnosis per a class of fault candidates that is for determining the faulty machine using all possible diagnostic fault candidates in all classes.

The following section includes a detailed description of all steps used in the assessment method.

4.5 Assessment of the Fault Diagnosis of Test Suites

Given EFSM specification *spec* and a set of EFSM fault candidates $D = D_1, D_2 \ldots D_n$, and test suites $TS = TS_1 \cup TS_2 \ldots TS_n$

Step 1 Let $\mathfrak{J}'$ denote the set of fault candidates of D that that are distinguishable from *spec*

Step 2 Derive from $\mathfrak{J}'$ the set of fault candidates $\mathfrak{J}$ such that every fault candidate is distinguishable from all other fault candidates in $\mathfrak{J}$ and for every fault candidate D_i in D let $|\mathfrak{J}'_i|$ denote the number of other fault candidates in $\mathfrak{J}'$ that are indistinguishable from D_i.

Step 3 For every $TS_j \in TS$

Let $\mathfrak{J}_{TSj}^{dis\_from\_Spec}$ = the set of fault candidates in $\mathfrak{J}$ killed by TS_j ;

Step 3.1 For every $D_i \in \mathfrak{J}_{TSj}^{dis\_from\_Spec}$

Assume D_i is the faulty IUT machine

Run inputs of TS_j on D_i and obtain the corresponding observed output

counter1= **0**

Let $\mathfrak{J}_{TSj}^{indis\_from\_Di} = \varnothing$; the set of fault candidates in $\mathfrak{J}_{TSj}^{dis\_from\_Spec}$ that are indistinguishable from D_i using TS_j

Step 3.1.1

For each $D_e \in \mathfrak{J}_{TSj}^{dis\_from\_Spec}$, $(e \neq i)$

If D_e is killed by TS_j then counter1++

else

Add D_e into the set $\mathfrak{J}_{TSj}^{indis\_from\_Di}$

EndFor (Step 3.1.1)

Step 3.1.2

Let $TS_{j'}$ be the test suites that kills every fault candidate in $\mathfrak{I}_{TS,j}^{indis\_from\_Di}$ that is distinguishable from *Di* using test suite TS_j

Let counter2 be the number of fault candidates $\mathfrak{I}_{TSj}^{indis\_from\_Di}$ killed by $TS_{j'}$

$FD_{\mathfrak{I},TSj}^{Di} = \frac{counter1}{|\mathfrak{I}_{TSj}^{dis\_from\_Spec}|-1} * 100$;

$FD_{\mathfrak{I}',TSj}^{Di} = \frac{counter1}{|\mathfrak{I}_{TSj}^{dis\_from\_Spec}|-1+|\mathfrak{I}'_{i}|-1} * 100$

$FD_{\mathfrak{I},TSj\cup TSj'}^{Di} = \frac{counter1+counter2}{|\mathfrak{I}_{TSj}^{dis\_from\_Spec}|-1} * 100$

$FD_{\mathfrak{I}',TSj\cup TSj'}^{Di} = \frac{counter1+counter2}{|\mathfrak{I}_{TSj}^{dis\_from\_Spe}|-1+|\mathfrak{I}'_{i}|-1} * 100$

EndFor (Step 3.1)

$FD_{\mathfrak{I},TSj}^{AVG} = \frac{\sum FD_{\mathfrak{I},TSj}^{Di}}{|\mathfrak{I}_{TSj}^{k}|}$

$FD_{\mathfrak{I}',TSj}^{AVG} = \frac{\sum FD_{\mathfrak{I}',TSj}^{Di}}{|\mathfrak{I}_{TSj}^{k}|}$

$FD_{\mathfrak{I},TS\cup TSj'}^{AVG} = \frac{\sum FD_{\mathfrak{I},TSj\cup TSj'}^{Di}}{|\mathfrak{I}_{TSj}^{k}|}$

$FD_{\mathfrak{I}',TSj\cup TSj'}^{AVG} = \frac{\sum FD_{\mathfrak{I}',TSj\cup TSj'}^{Di}}{|\mathfrak{I}_{TSj}^{k}|}$

EndFor(Step3)

4.6 Application Example of Test Suite Fault Diagnosis Assessment

In this section, we describe the steps of the above mentioned assessment assuming we have the following fault candidates of part of the POP3 EFSM [34] depicted in Figure 4.1.

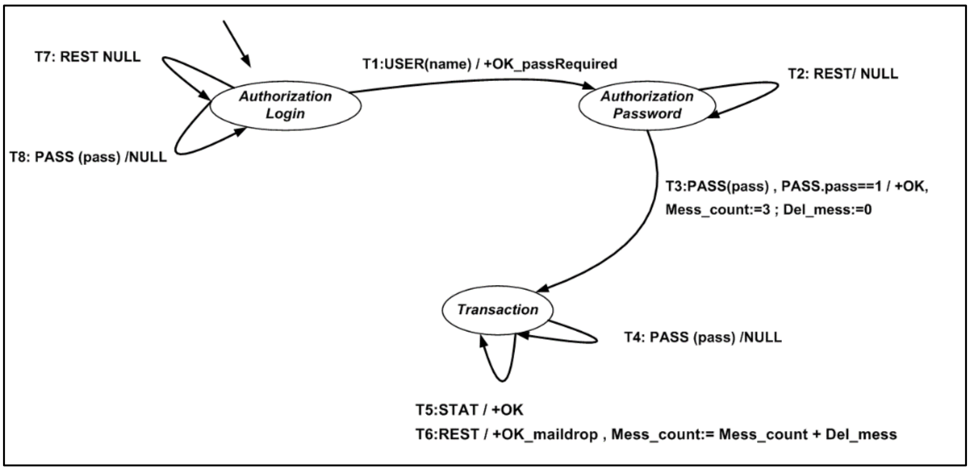

Figure 4.1 Part of POP3 EFSM [34]

The first fault candidate D_1 is a faulty implementation with the single transfer fault where *T5* transfers to state "*Authorization Login*" instead of state "*Transaction*". Thus, D_1 is actually as depicted in Figure 4.2.

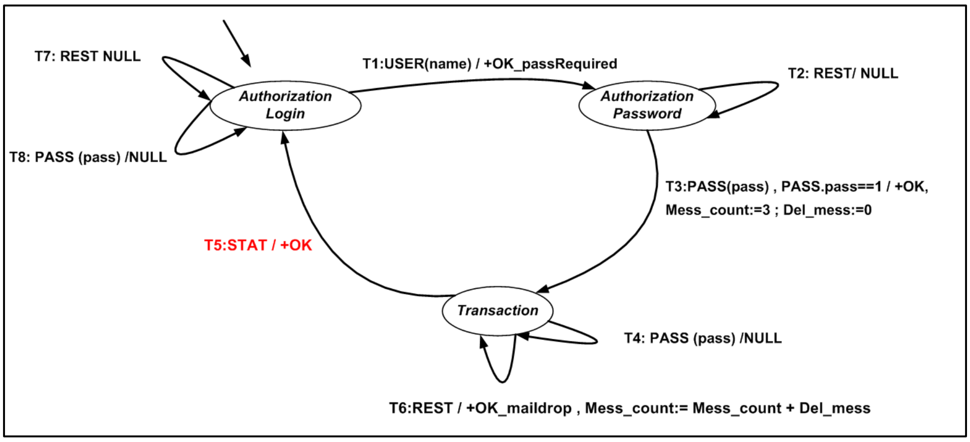

Figure 4.2 Fault Candidate D1

The fault candidate D_2 is due to a transfer fault in *T5* that transfers to state "*Authorization Password*" Instead of state "*Transaction*". Thus, D_2 is actually as depicted in Figure 4.3.

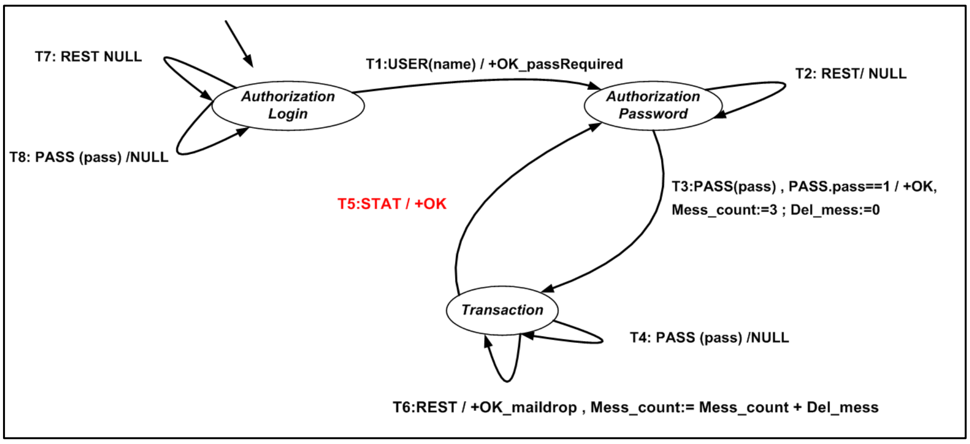

Figure 4.3 Fault Candidate D2

The fault candidate D_3 is a faulty implementation with the single transfer fault where *T6* transfers to state "*Authorization Login*" instead of state "*Transaction*". Thus, D_3 is actually as depicted in Figure 4.4.

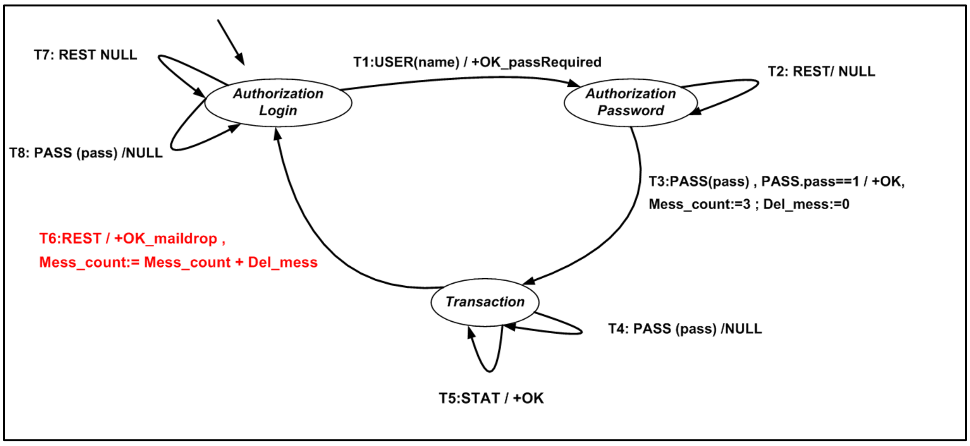

Figure 4.4 Fault Candidate D3

The fault candidate D_4 is due to an assignment fault in *T7,* where a new update statement, "*Mess_count*:=**3**", was inserted to *T7*. Thus, D_4 is actually as depicted in Figure 4.5.

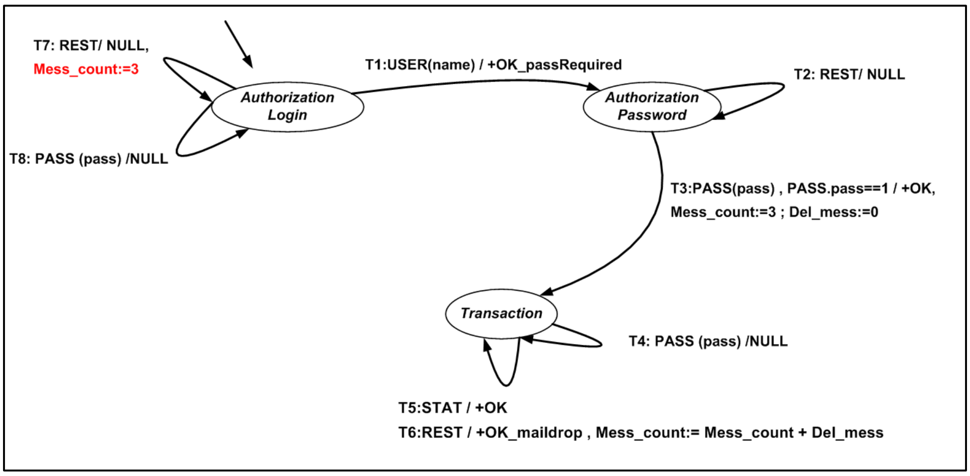

Figure 4.5 Fault Candidate D4

Step 1: The fault candidate D_4 is indistinguished from *spec*, whereas, D_1,D_2 and D_3 are all distinguished from *spec*; therefore, the list of distinguished fault candidates from *spec* is $\mathfrak{I}'=\{D_1,D_2,D_3\}$.

Step 2: The fault candidates D_1,D_2 and D_3 are all distinguished from each other; therefore, we have total number of three classes of fault candidates, and each class has one fault candidate ($|\mathfrak{I}| = \mathbf{3}$, AVG($|\mathfrak{I}_i|$) = **1**).

Step 3: Given test suite TS_j with the "USER(0),PASS(1),STAT,REST,REST"; If IUT=*spec*, applying TS_j on *spec* will lead to the following output:" +OK_passRequired,+OK,+OK,+OK_maildrop,+OK_maildrop".

Line 10: Now we run TS_j on one fault candidate from all classes that we have, and we notice that the three fault candidates D_1, *D2* and D_3 are killed by TS_j so $\mathfrak{I}_{TSj}^{dis\_from\_Spec} = \{D_1, D2, D3\}$.

Line 12 and Line 13: We let the first fault candidate D_1=IUT and compute the capability of locating this candidate using the different proposed scores. The output obtained from running TS_j on D_1 is "+OK_passRequired,+OK,+OK,NULL,NULL" .

Step 3.1.1: Now we run TS_j on D_2 and we obtain the output which is "+OK_passRequired,+OK,+OK,NULL,NULL" which is the same as the above output. This means that TS_j does not distinguish D_2 from D_1 so we add D_2 into the set $\mathfrak{I}_{TSj}^{indis\_from\_Di}$. Now we run TS_j on D_3 and we obtain the output

"+OK_passRequired,+OK,+OK,+OK_maildrop,NULL" which is different from the output obtained when running TS_j on D_1. It means that TS_j distinguishes D_3 from D_1; therefore, we increase the value of counter1 by **1**. Then, in **Line 29:** we compute, FD1= $\frac{counter1}{|\Im_{TSj}^{dis\_from\_Spec}|-1} * 100 = \frac{1}{3-1} * 100 = 50$ %. In **Line 30:** Since we have only one fault candidate in the class which has D_1, it means that FD2 = FD1 = 50 %.

Step 3.1.2: To distinguish D_1 from D_2, we need to add the new test suite $TS_{j'}$ = "USER(0),PASS(1),STAT,pass(1)". Therefore, applying $TS_{j'}$ on D_1 will lead to the following output "+OK_passRequired,+OK,+OK,NULL". Applying $TS_{j'}$ on D_2 will lead to different output which is "+OK_passRequired,+OK,+OK,+OK" . It means that D_2 is distinguished from D_1 by $TS_{j'}$ so we increase the value of counter2 by **1**. Then **Line 31:** we compute, FD3 = $\frac{counter1+counter2}{|\Im_{TSj}^{dis\_from\_Spec}|-1} * 100 = \frac{1+1}{3-1} * 100 = 100\%$ .In **Line 32:** Since we have one fault candidate only in the class which has D_1, it means that FD4 = FD3 = 100 %.

Line 34 and Line 37: Afterwards, we have to let D_2= IUT then D_3=IUT and we repeat the above mentioned steps again to calculate the average fault diagnosis capability of TS_j per all fault candidates.

4.7 Experimental Evaluation

We conduct experiments using the considered EFSM specifications, namely the TFTP, POP3, CD Player, Initiator, Responder, and the SCP examples. In addition, we consider the test suites, derived for these EFSM examples, described in previous chapters, namely, the TT, All-Uses, SITS, STF, EP, PPST and the Rand test suites.

In this chapter, for each specification, we illustrate the fault diagnosis capability FD1, FD2 and FD3 of each test suite over each type of diagnosis candidate and per all diagnosis candidates. At the end of this chapter, we determine the average results for all considered examples, and we illustrate the ranking of all considered test suites.

For evaluating the test suites, we consider the average fault diagnosis score and the average length of each test suite per all examples, then we determine the best test suite in terms of fault diagnosis score only and based on fault diagnosis score and the length of the corresponding test suite. For evaluating the test suites based on the coverage, we compare the test suites depending on their fault diagnosis score only. For evaluating the test suites based on the fault diagnosis and length, we give 99% importance to the fault diagnosis score of the corresponding test suite and 1% importance rate to its length; therefore, the total score of each test suite is determined as the following:

- ***FD1-length score =*** $\frac{0.99 \times FD1\ score}{0.01 \times Test\ Suite\ Length}$
- ***FD2-length score =*** $\frac{0.99 \times FD2\ score}{0.01 \times Test\ Suite\ Length}$
- ***FD3-length score =*** $\frac{0.99 \times FD3\ score}{0.01 \times Test\ Suite\ Length}$
- ***FD4-length score =*** $\frac{0.99 \times FD4\ score}{0.01 \times Test\ Suite\ Length}$

4.7.1 Fault Diagnosis Capabilities (FD1 and FD2) of EFSM Test Suites

In this section, for each specification, we determine the fault diagnosis capabilities (FD1 and FD2) of each test suite over each type of diagnosis candidate and per all diagnosis candidates. At the end of this section, we rank the test suites based on the criteria described above.

a) Test Suites FD1 and FD2 Assessment for the TFTP Example

In this section, we assess the fault diagnosis capabilities (FD1 and FD2) for the TFTP example.

(a.1) FD1 Assessment per Considered Diagnostic Candidates

Table 4.1 shows FD1 for each test suite over each type of fault candidates.

Table 4.1 FD1 for each test suite over each type of fault candidates

FD1	Length	STF	DTF	STF/DTF	STF/SAF	DTF/SAF
TT	10	98	98	98	98	98
All-Uses	8	99	99	99	99	99
SITS	**76**	99	100	100	99	100
STF	29	100	100	100	100	100
EP	53	99	100	100	99	100
PPST	26	99	100	100	99	100
Rand	65	99	98	98	99	98

The Figure 4.6 shows FD1 for each test suite over each type of fault candidates.

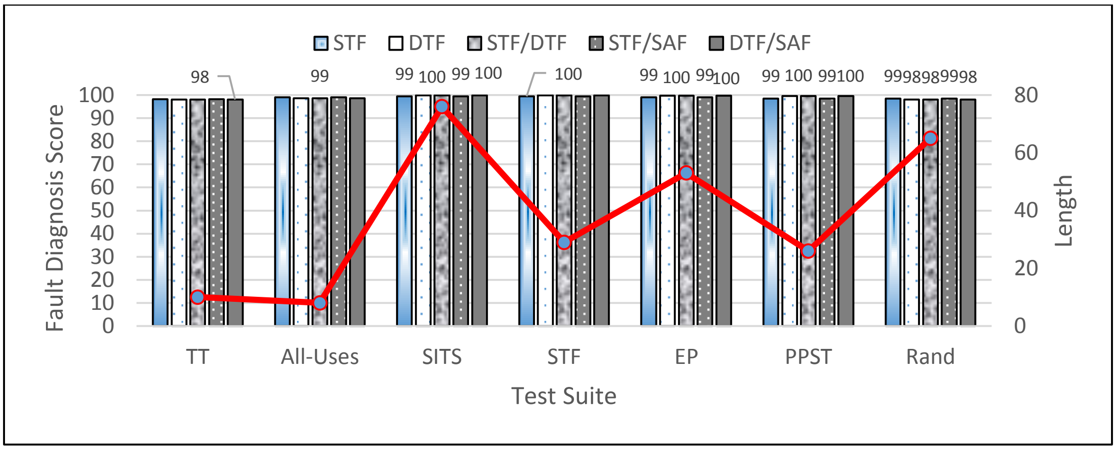

Figure 4.6 FD1 for each test suite over each type of fault candidates

(a.2) FD2 Assessment per Considered Diagnostic Candidates

Table 4.2 shows FD2 for each test suite over each type of fault candidates.

Table 4.2 FD2 for each test suite over each type of fault candidates

FD2	Length	STF	DTF	STF/DTF	STF/SAF	DTF/SAF
TT	10	98	98	98	98	98
All-Uses	8	99	98	98	99	98
SITS	**76**	99	99	99	99	99
STF	29	100	99	99	100	99
EP	53	99	99	99	99	99
PPST	26	99	99	99	99	99
Rand	65	99	97	97	99	97

The Figure 4.7 shows FD2 for each test suite over each type of fault candidates.

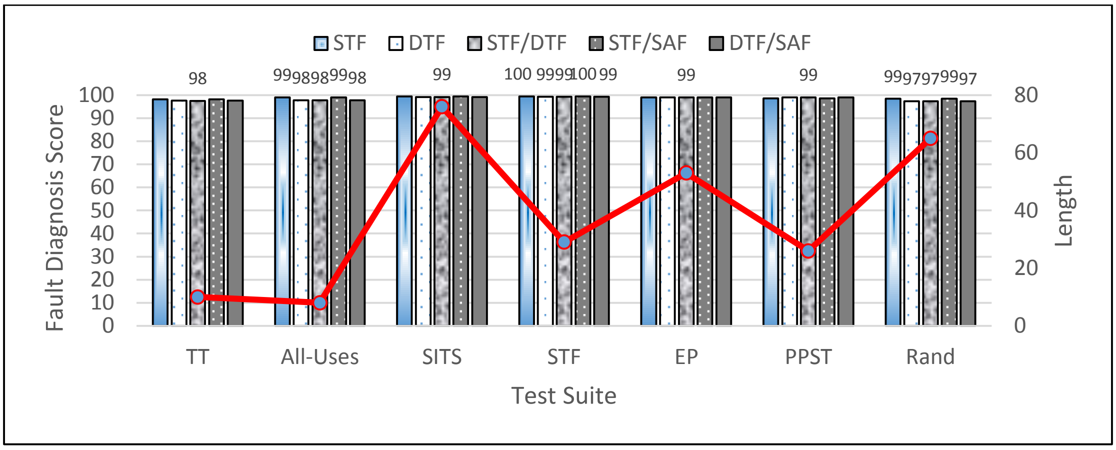

Figure 4.7 FD2 for each test suite over each type of fault candidates

(a.3) FD1 and FD2 Assessment considering all Diagnostic Candidates

Table 4.3 shows the fault diagnosis details for each test suite over all fault candidates.

Table 4.3 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	SITS	STF	EP	PPST	Rand
D	735.9	738	735.9	735.9	735.9	735.9	735.9
$\Im'$	715	715	715	715	715	715	715
$\|\Im\|$	643.6	643.4	643.6	643.6	643.6	643.6	643.6
AVG($\|\Im_i\|$)	1.1	1.1	1.1	1.1	1.1	1.1	1.1
$\Im_{TSj}^{dis\_from\_Spec}$	613.3	548.6	643.6	642.4	627.1	633	622.6
FD1	98.2	98.8	99.7	99.7	99.5	99.2	98.2
FD2	97.8	98.3	99.3	99.4	99.1	98.9	97.8
FD3	100	100	100	100	100	100	100
FD4	99.7	99.5	99.6	99.6	99.6	99.7	99.6
TSj Length	40	52	76	100	171	159	88
$TS_{j'}$ Length	74.8	50.8	11.2	11.2	21.2	24.2	83.2
$TS_{j'}$ Text Cases	22.2	16.8	1.6	1.6	5.2	6	24.6
$TS_j \cup TS_{j'}$ Length	115	103	87	111	192	183	171

The Figure 4.8 shows FD1 and FD2 for each test suite over all fault candidates.

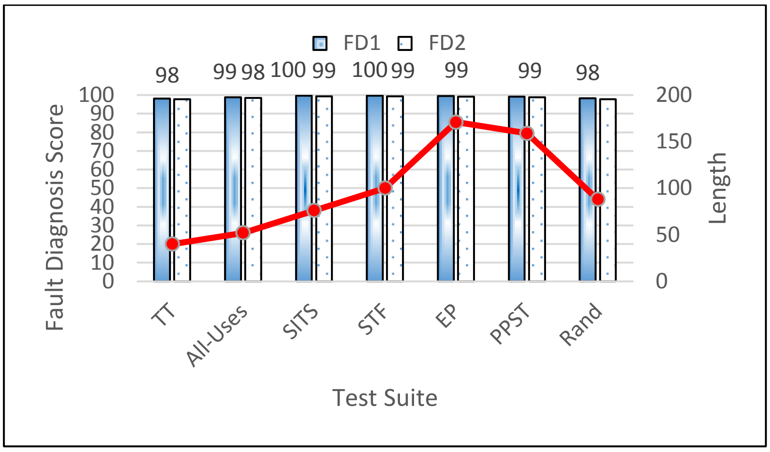

Figure 4.8 FD1 and FD2 for each test suite over all fault candidates

b) Test Suites FD1 and FD2 Assessment for the CD Player Example

In this section, we assess the fault diagnosis capabilities (FD1 and FD2) for the CD Player example.

(b.1) FD1 Assessment per Considered Diagnostic Candidates

Table 4.4 shows FD1 for each test suite over each type of fault candidates.

Table 4.4 FD1 for each test suite over each type of fault candidates

FD1	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF
TT	31	100	99	99	88	98	99
SITS	82	100	100	100	97	100	100
STF	166	99	100	100	95	99	100
EP	161	100	100	100	98	100	100
PPST	197	100	100	100	98	100	100
Rand	230	100	99	99	98	99	99

The Figure 4.9 shows FD1 for each test suite over each type of fault candidates.

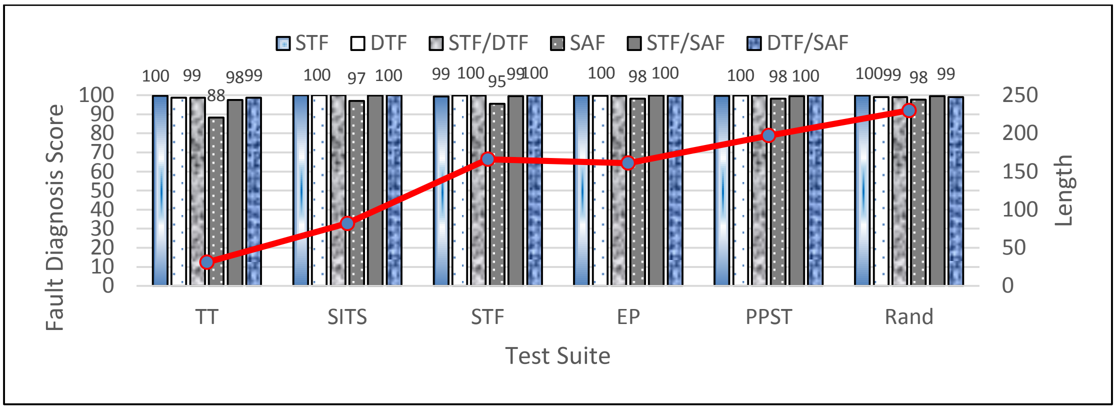

Figure 4.9 FD1 for each test suite over each type of fault candidates

(b.2) FD2 Assessment per Considered Diagnostic Candidates

Table 4.5 shows FD2 for each test suite over each type of fault candidates.

Table 4.5 FD1 for each test suite over each type of fault candidates

FD2	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF
TT	31	100	99	99	85	96	49
SITS	82	100	100	100	94	99	50
STF	166	99	100	100	93	99	50
EP	161	100	100	100	98	100	50
PPST	197	100	100	100	94	98	50
Rand	230	100	99	99	89	96	49

The Figure 4.10 shows FD2 for each test suite over each type of fault candidates.

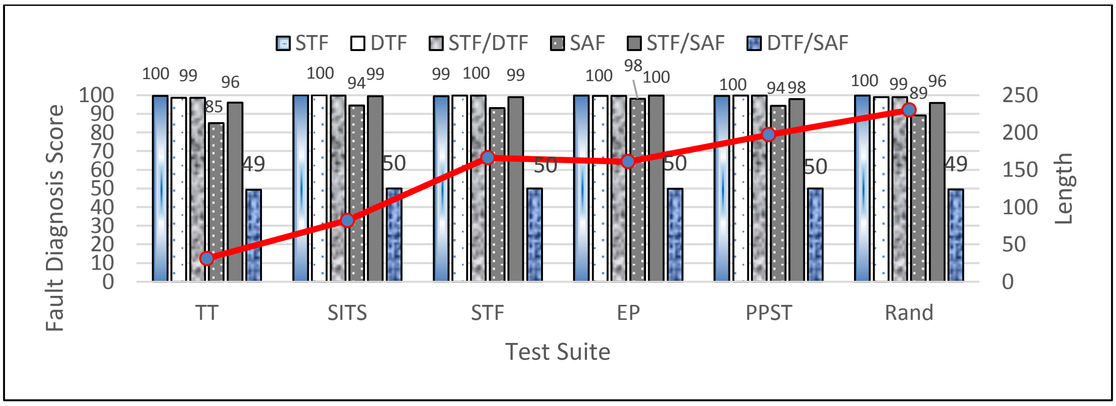

Figure 4.10 FD2 for each test suite over each type of fault candidates

(b.3) FD1 Assessment per Considered Assignment Diagnostic Candidates

Table 4.6 shows FD1 for each test suite over each type of assignment faults candidates.

Table 4.6 FD1 for each test suite over assignment faults candidates

FD1	Length	SAI	SAD	SARHS
TT	31	70	100	92
SITS	82	100	100	97
STF	166	100	100	97
EP	161	100		100
PPST	197	96	100	99
Besr Random	230	71	100	99

The Figure 4.11 shows FD1 for each test suite over each type of assignment fault candidates.

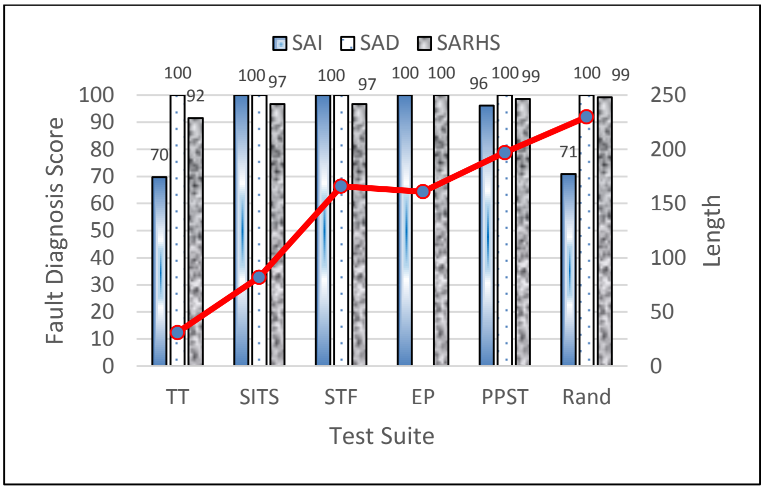

Figure 4.11 FD1 for each test suite over assignment fault candidates

(b.4) FD2 Assessment per Considered Assignment Diagnostic Candidates

Table 4.7 shows FD2 for each test suite over each type of assignment fault candidates.

Table 4.7 FD2 for each test suite over assignment fault candidates

FD2	Length	SAI	SAD	SARHS
TT	31	70	100	92
SITS	82	100	100	97
STF	166	100	100	97
EP	161	100		100
PPST	197	96	100	99
Rand	230	71	100	98

The Figure 4.12 shows FD2 for each test suite over each type of assignment fault candidates.

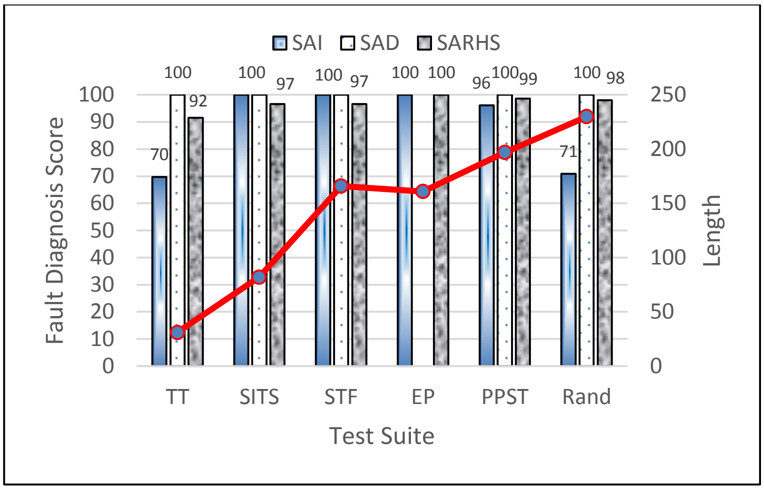

Figure 4.12 FD2 for each test suite over assignment fault candidates

(b.5) FD1 and FD2 Assessment considering all Diagnostic Candidates

Table 4.8 shows the fault diagnosis details for each test suite over all fault candidates.

Table 4.8 Fault diagnosis details for each test suite over all fault candidates

	TT	SITS	STF	EP	PPST	Rand
D	1359.3	1359.3	1359.3	1359.3	1359.3	1359.3
$\mathfrak{I}'$	1350.7	1350.7	1350.7	1350.7	1350.7	1350.7
$\lvert\mathfrak{I}\rvert$	1001.6	1001.6	1001.6	1001.6	1001.6	1001.6
AVG($\lvert\mathfrak{I}_i\rvert$)	1.1	1.1	1.1	1.1	1.1	1.1
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	896.6	985	985.3	943.2	988.7	993.6
FD1	93.6	99.3	99	99.6	99.1	96
FD2	87.6	93.4	93.1	93.4	92.9	89
FD3	100	100	100	100	100	100
FD4	92.2	91.1	92.4	91.7	93	90.9
TSj Length	31	82	166	161	197	230
$TS_{j'}$ Length	65	53.2	15.9	38.3	13.6	64.4
$TS_{j'}$ Text Cases	17.6	10.5	3.6	9.2	2.6	16.4
$TS_j \cup TS_{j'}$ Length	96	135.2	181.9	199.3	210.6	294.4

The Figure 4.13 shows FD1 and FD2 for each test suite over all fault candidates.

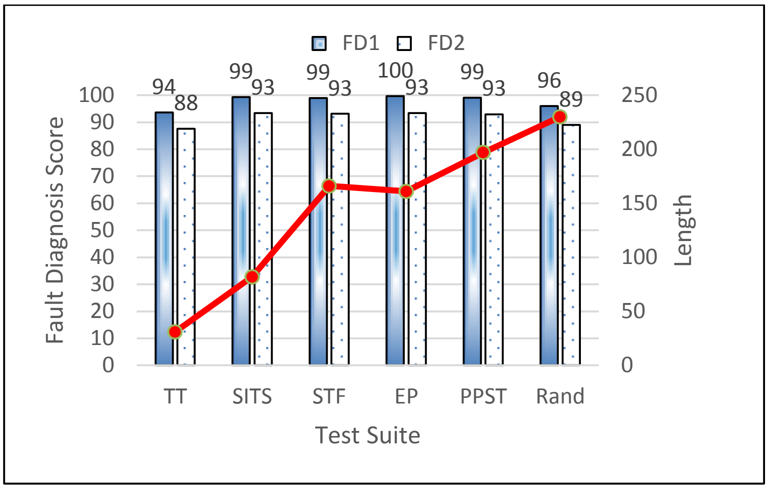

Figure 4.13 FD1 and FD2 for each test suite over all fault candidates

c) Test Suites FD1 and FD2 Assessment for the POP3 Example

In this section, we assess the fault diagnosis capabilities (FD1 and FD2) for the POP3 example.

(c.1) FD1 Assessment per Considered Diagnostic Candidates

Table 4.9 shows FD1 for each test suite over each type of fault candidates.

Table 4.9 FD1 for each test suite over each type of fault candidates

FD1	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF
TT	22	84	90	90	100	83	90
All-Uses	17	63	79	78	100	65	79
SITS	76	89	97	97	100	90	97
STF	74	89	97	97	100	90	97
EP	56	76	91	90	100	74	91
PPST	32	85	95	94	100	83	95
Rand	110	40	50	49	100	39	50

The Figure 4.14 shows FD1 for each test suite over each type of fault candidates.

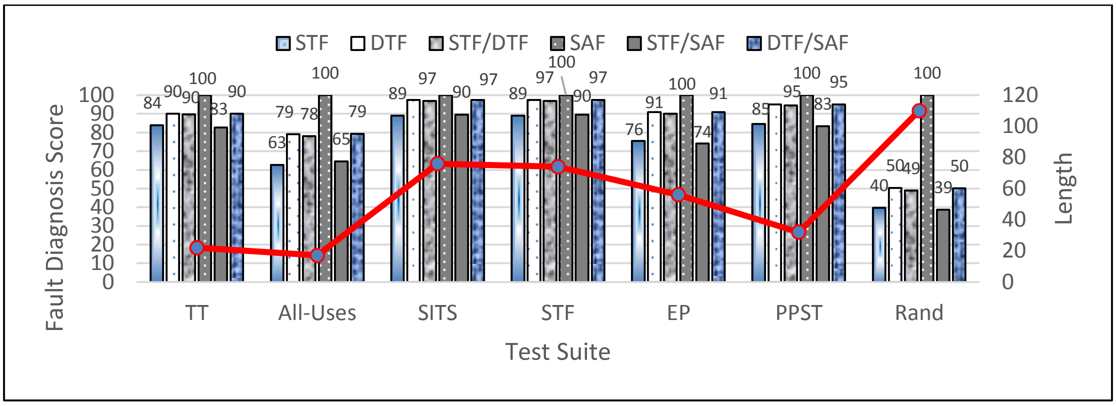

Figure 4.14 FD1 for each test suite over each type of fault candidates

(c.2) FD2 Assessment per Considered Diagnostic Candidates

Table 4.10 shows FD2 for each test suite over each type of fault candidates.

Table 4.10 FD2 for each test suite over each type of fault candidates

FD2	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF
TT	22	84	89	89	33	82	89
All-Uses	17	63	78	77	33	63	78
SITS	76	89	97	96	33	88	97
STF	74	89	97	96	33	88	97
EP	56	76	90	89	33	73	90
PPST	32	85	94	94	33	82	94
Rand	110	40	50	48	33	38	50

The Figure 4.15 shows FD2 for each test suite over each type of fault candidates.

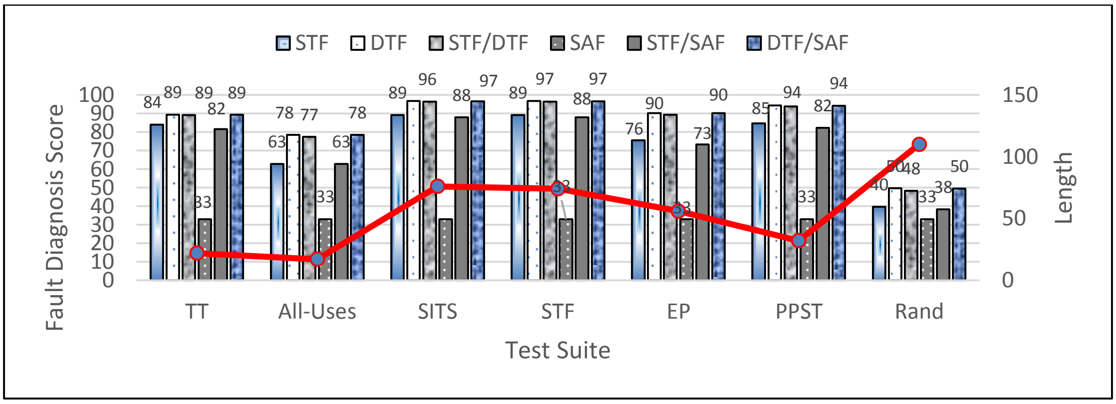

Figure 4.15 FD2 for each test suite over each type of fault candidates

(c.3) FD1 Assessment per Considered Assignment Diagnostic Candidates

Table 4.11 shows FD1 for each test suite over each type of assignment fault candidates.

Table 4.11 FD1 for each test suite over assignment fault candidates

FD1	Length	SAD	SARHS
TT	22	100	100
All-Uses	17	100	100
SITS	76	100	100
STF	74	100	100
EP	56	100	100
PPST	32	100	100
Rand	110	100	100

The Figure 4.16 shows FD1 for each test suite over each type of assignment fault candidates.

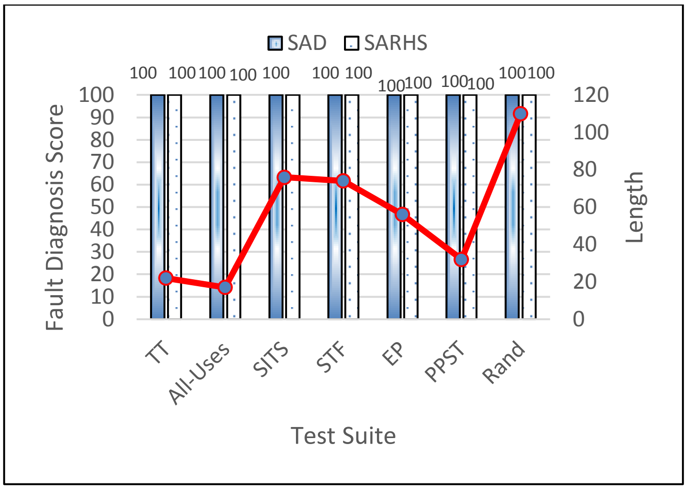

Figure 4.16 FD1 for each test suite over assignment fault candidates

(c.4) FD2 Assessment per Considered Assignment Diagnostic Candidates

Table 4.12 shows FD2 for each test suite over each type of assignment fault candidates.

Table 4.12 FD2 for each test suite over assignment fault candidates

FD2	Length	SAD	SARHS
TT	22	100	50
All-Uses	17	100	50
SITS	76	100	50
STF	74	100	50
EP	56	100	50
PPST	32	100	50
Rand	110	100	50

The Figure 4.17 shows FD2 for each test suite over each type of assignment fault candidates.

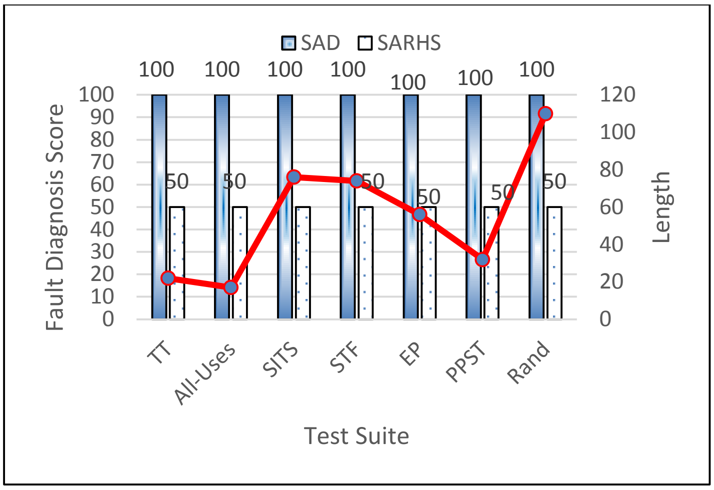

Figure 4.17 FD2 for each test suite over assignment fault candidates

(c.5) FD1 and FD2 Assessment Considering all Diagnostic Candidates

Table 4.13 shows the fault diagnosis details for each test suite over all fault candidates.

Table 4.13 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	SITS	STF	EP	PPST	Rand
D	234.7	234.7	234.7	234.7	234.7	234.7	234.7
$\mathfrak{I}'$	217.3	217.3	217.3	217.3	217.3	217.3	217.3
$\lvert\mathfrak{I}\rvert$	189.4	189.4	189.4	189.4	189.4	189.4	189.4
AVG($\lvert\mathfrak{I}_i\rvert$)	1.4	1.4	1.4	1.4	1.4	1.4	1.4
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	189.4	189.4	189.4	189.4	189.4	189.4	189.4
FD1	92.1	83	96.3	96.3	90.2	94.1	66
FD2	77	67.8	81.2	81.2	75.2	79	51.1
FD3	100	100	100	100	100	100	100
FD4	99.2	99.2	99.2	99.2	99.2	99.2	99.2
TSj Length	22	17	76	74	56	32	110
$TS_{j'}$ Length	117	232.6	55.8	56.4	168.6	81.4	525.4
$TS_{j'}$ Text Cases	35.6	74.6	13.8	13.8	42.4	21	174
$TS_j \cup TS_{j'}$ Length	139	249.6	131.8	130.4	224.6	113.4	635.4

The Figure 4.18 shows FD1 and FD2 for each test suite over all fault candidates.

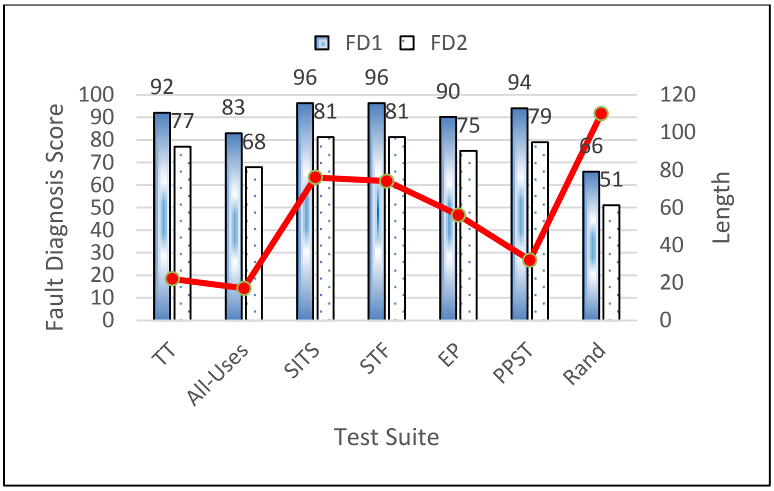

Figure 4.18 FD1 and FD2 for each test suite over all fault candidates

d) Test Suites FD1 and FD2 Assessment for the Initiator Example

In this section, we assess the fault diagnosis capabilities (FD1 and FD2) for the Initiator example.

(d.1) FD1 Assessment per Considered Diagnostic Candidates

Table 4.14 shows FD1 for each test suite over each type of fault candidates.

Table 4.14 FD1 for each test suite over each type of fault candidates

FD1	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
TT	30	100	89	89	82	91	94	100
All-Uses	36	100	97	98	90	94	97	100
STF	46	100	98	99	74	89	97	98
EP	79	100	98	99	100	100	98	98
PPST	142	100	99	99	82	92	98	100
Rand	120	96	92	92	66	82	92	100

The Figure 4.19 shows FD1 for each test suite over each type of fault candidates.

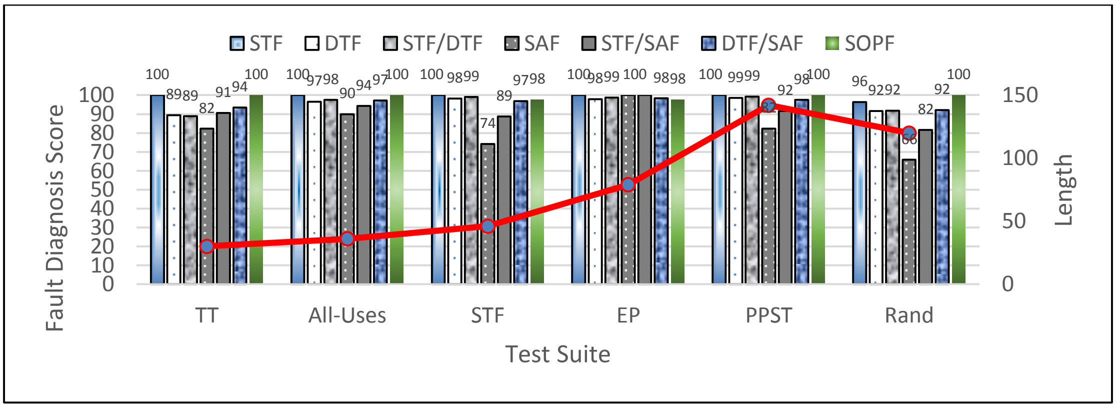

Figure 4.19 FD1 for each test suite over each type of fault candidates

(d.2) FD2 Assessment per Considered Diagnostic Candidates

Table 4.15 shows FD2 for each test suite over each type of fault candidates.

Table 4.15 FD2 for each test suite over each type of fault candidates

FD2	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
TT	30	94	66	63	67	77	72	58
All-Uses	36	94	70	69	76	82	75	58
STF	46	94	71	70	59	76	73	57
EP	79	94	71	69	62	81	70	57
PPST	142	94	72	70	68	80	74	58
Rand	120	90	67	65	55	72	71	58

The Figure 4.20 shows FD2 for each test suite over each type of fault candidates.

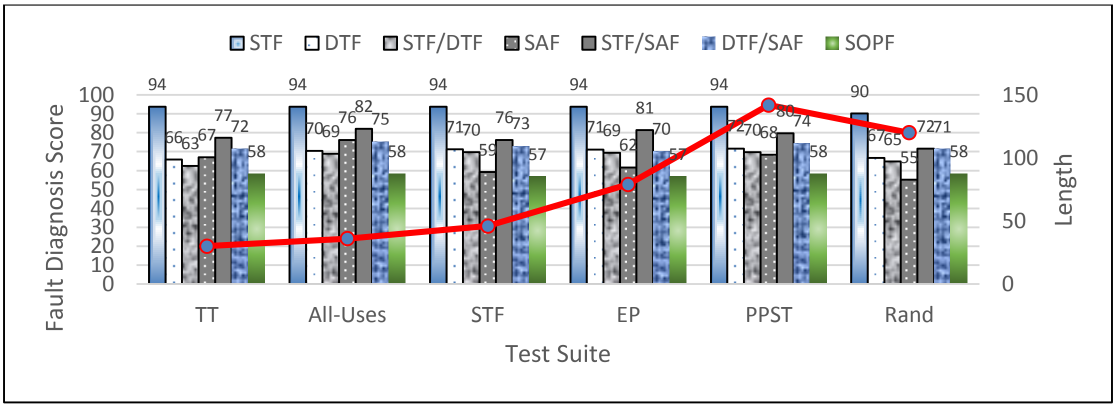

Figure 4.20 FD2 for each test suite over each type of fault candidates

(d.3) FD1 Assessment per Considered Assignment Diagnostic Candidates

Table 4.16 shows FD1 for each test suite over each type of assignment fault candidates.

Table 4.16 FD1 for each test suite over assignment fault candidates

FD1	Length	SAI	SAD	SARHS
TT	30	100	95	82
All-Uses	36	100	95	88
STF	46	100	90	73
EP	79	100	100	100
PPST	142	100	100	73
Rand	120	100	93	53

The Figure 4.21 shows FD1 for each test suite over each type of assignment fault candidates.

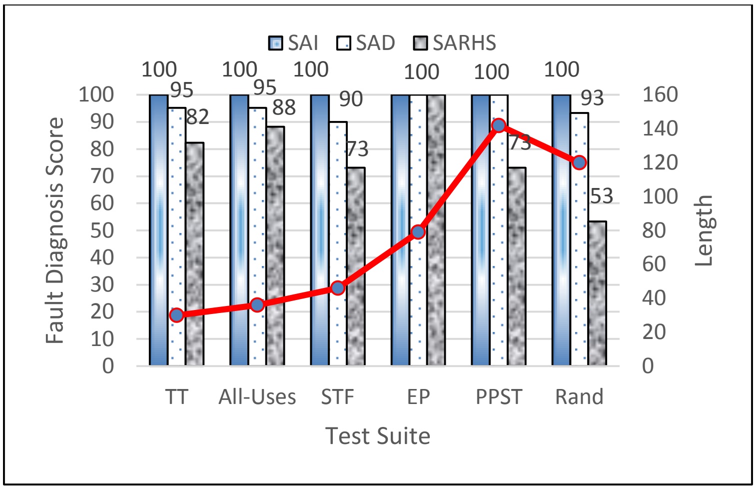

Figure 4.21 FD1 for each test suite over assignment fault candidates

(d.4) FD2 Assessment per Considered Assignment Diagnostic Candidates

Table 4.17 shows FD2 for each test suite over each type of assignment fault candidates.

Table 4.17 FD2 for each test suite over assignment fault candidates

FD2	Length	SAI	SAD	SARHS
TT	30	50	95	71
All-Uses	36	83	95	74
STF	46	50	90	60
EP	79	50	100	83
PPST	142	83	100	60
Rand	120	83	93	47

The Figure 4.22 shows FD2 for each test suite over each type of assignment fault candidates.

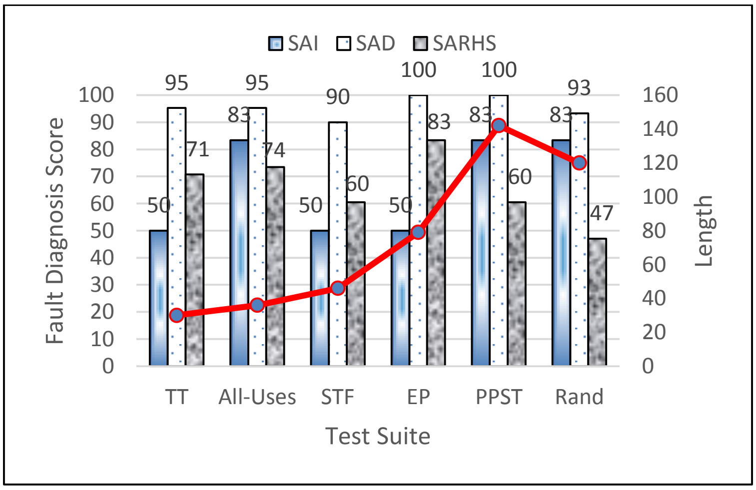

Figure 4.22 FD2 for each test suite over assignment fault candidates

(d.5) FD1 and FD2 Assessment considering all Diagnostic Candidates

Table 4.18 shows the fault diagnosis details for each test suite over all fault candidates.

Table 4.18 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	STF	EP	PPST	Rand
D	289.1	289.1	289.1	289.1	289.1	289.1
$\mathfrak{I}'$	103.7	103.7	103.7	103.7	103.7	103.7
$\vert\mathfrak{I}\vert$	24.7	24.7	24.7	24.7	24.7	24.7
AVG($\vert\mathfrak{I}_i\vert$)	3.1	3.1	3.1	3.1	3.1	3.1
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	22.7	23.8	20.2	15.8	21	22.6
FD1	92.2	95.9	91.8	99.3	94.3	86.6
FD2	71.2	77.7	70.1	73.8	76	70.2
FD3	100	100	100	100	100	99.9
FD4	82.1	82.3	78.9	68.2	79.5	84.9
TSj Length	30	36	46	79	142	120
$TS_{j'}$ Length	23.3	18.4	19	5.5	22.8	35
$TS_{j'}$ Text Cases	3.4	2.7	2.8	1	2.8	5
$TS_j \cup TS_{j'}$ Length	53.3	54.4	65	84.5	164.8	155

The Figure 4.23 shows FD1 and FD2 for each test suite over all fault candidates.

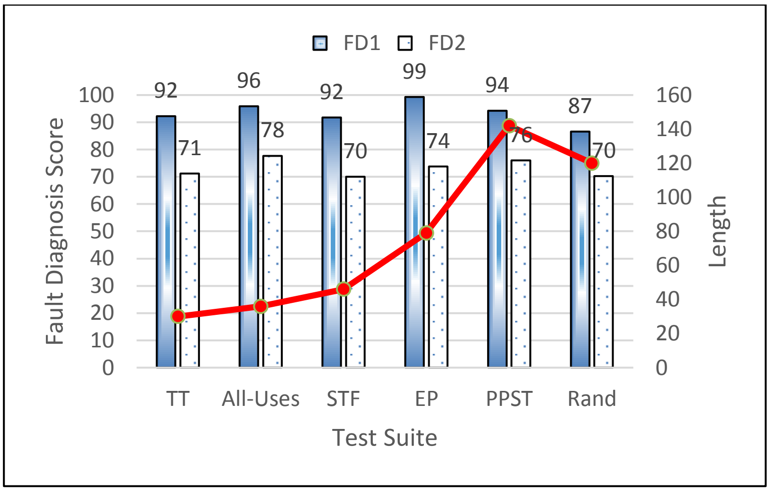

Figure 4.23 FD1 and FD2 for each test suite over all fault candidates

e) Test Suites FD1 and FD2 Assessment for the Responder Example

In this section, we assess the fault diagnosis capabilities (FD1 and FD2) for the Responder example.

(e.1) FD1 Assessment per Considered Diagnostic Candidates

Table 4.19 shows FD1 for each test suite over each type of fault candidates.

Table 4.19 FD1 for each test suite over each type of fault candidates

FD1	Length	SAF	STF/SAF	DTF/SAF	SOPF
TT	10				100
All-Uses	8	100	100	100	100
STF	29	100	100	100	100
EP	53	100	100	100	100
PPST	26				
Rand	65	100	100	100	100

The Figure 4.24 shows FD1 for each test suite over each type of fault candidates.

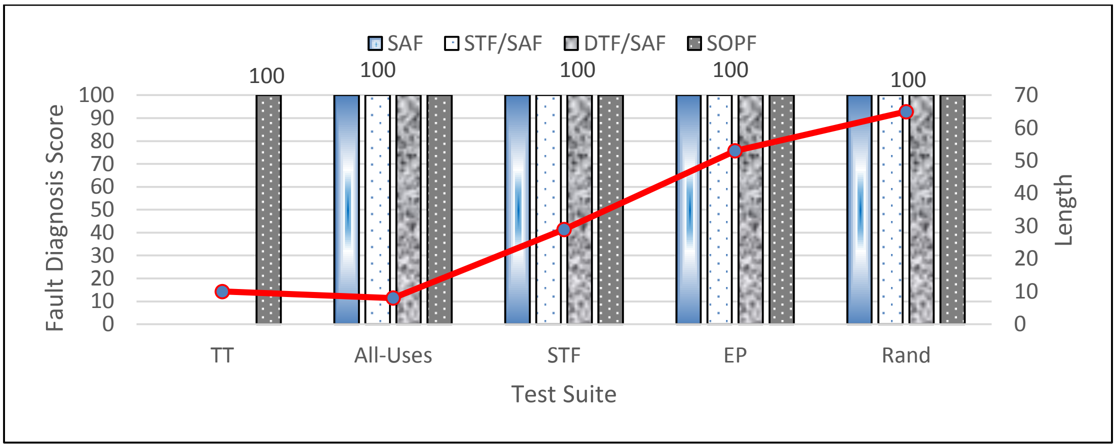

Figure 4.24 FD1 for each test suite over each type of fault candidates

(e.2) FD2 Assessment per Considered Diagnostic Candidates

Table 4.20 shows FD2 for each test suite over each type of fault candidates.

Table 4.20 FD2 for each test suite over each type of fault candidates

FD2	Length	SAF	STF/SAF	DTF/SAF	SOPF
TT	10	0	0	0	88
All-Uses	8	75	75	75	88
STF	29	75	75	75	88
EP	53	75	75	75	88
PPST	26	0	0	0	0
Rand	65	75	75	75	90

The Figure 4.25 shows FD2 for each test suite over each type of fault candidates.

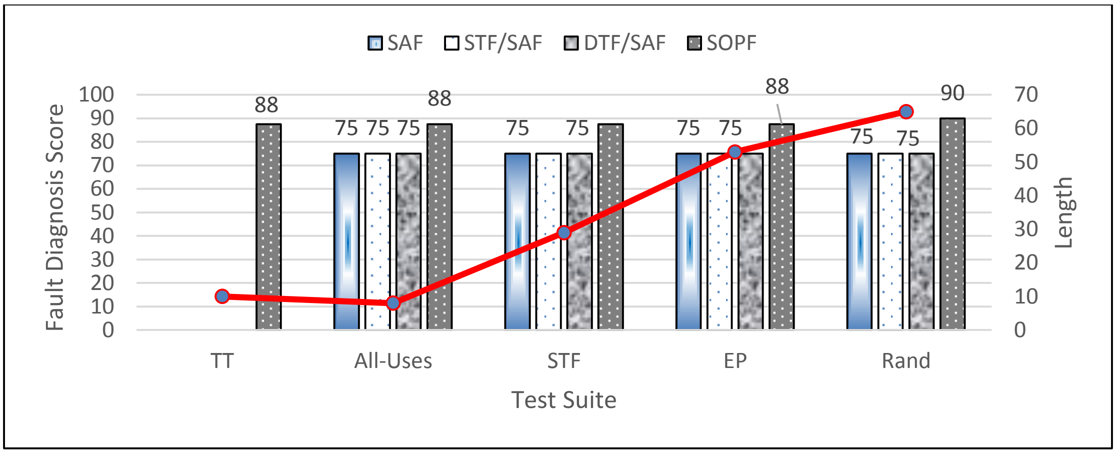

Figure 4.25 FD2 for each test suite over each type of fault candidates

(e.3) FD1 Assessment per Considered Assignment Diagnostic Candidates

Table 4.21 shows FD1 for each test suite over each type of assignment fault candidates.

Table 4.21 FD1 for each test suite over of assignment fault candidates

FD1	Length	SAD	SARHS
TT	10	100	0
All-Uses	8	100	100
STF	29	100	100
EP	53	100	100
PPST	26	100	0
Rand	65	100	100

The Figure 4.26 shows FD1 for each test suite over each type of assignment fault candidates.

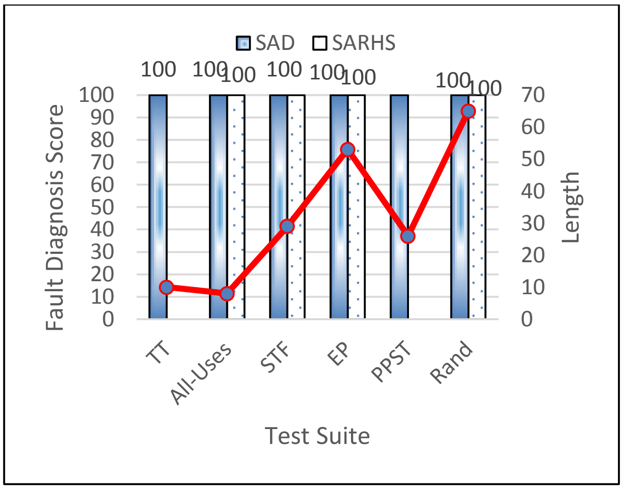

Figure 4.26 FD1 for each test suite over of assignment fault candidates

(e.4) FD2 Assessment per Considered Assignment Diagnostic Candidates

Table 4.22 shows FD2 for each test suite over each type of assignment fault candidates.

Table 4.22 FD2 for each test suite over assignment fault candidates

FD2	Length	SAD	SARHS
TT	10	100	0
All-Uses	8	100	100
STF	29	100	100
EP	53	100	100
PPST	26	100	0
Rand	65	100	100

The Figure 4.27 shows FD2 for each test suite over each type of assignment fault candidates.

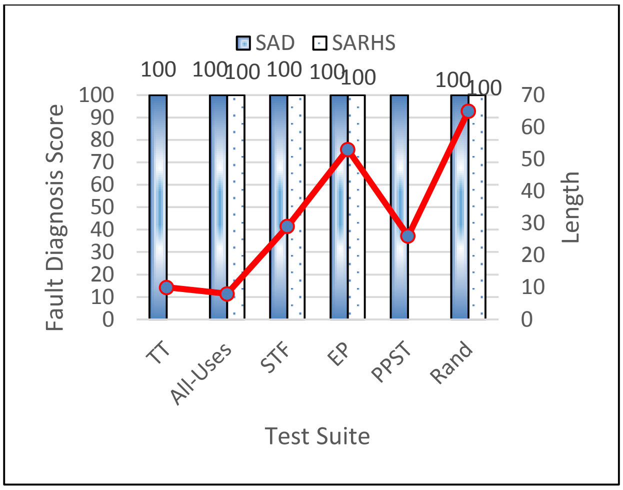

Figure 4.27 FD2 for each test suite over assignment fault candidates

(e.5) FD1 and FD2 Assessment considering all Diagnostic Candidates

Table 4.23 shows the fault diagnosis details for each test suite over all fault candidates.

Table 4.23 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	STF	EP	PPST	Rand
D	45.8	45.8	45.8	45.8	45.8	45.8
$\Im'$	1.8	1.8	1.8	1.8	1.8	1.8
$\|\Im\|$	1.4	1.4	1.4	1.4	1.4	1.4
AVG($\|\Im_i\|$)	1.3	1.3	1.3	1.3	1.3	1.3
$\Im_{TSj}^{dis\_from\_Spec}$	1.3	1.3	1.3	1.3	1.1	1.4
FD1	33.3	100	100	100	16.7	100
FD2	31.3	85.4	85.4	85.4	16.7	85.8
FD3	100				100	
FD4	81				85	
TSj Length	10	8	29	53	26	65
$TS_{j'}$ Length	4				4	
$TS_{j'}$ Text Cases	1				1	
$TS_j \cup TS_{j'}$ Length	14	8	29	53	30	65

The Figure 4.28 shows FD1 and FD2 for each test suite over all fault candidates.

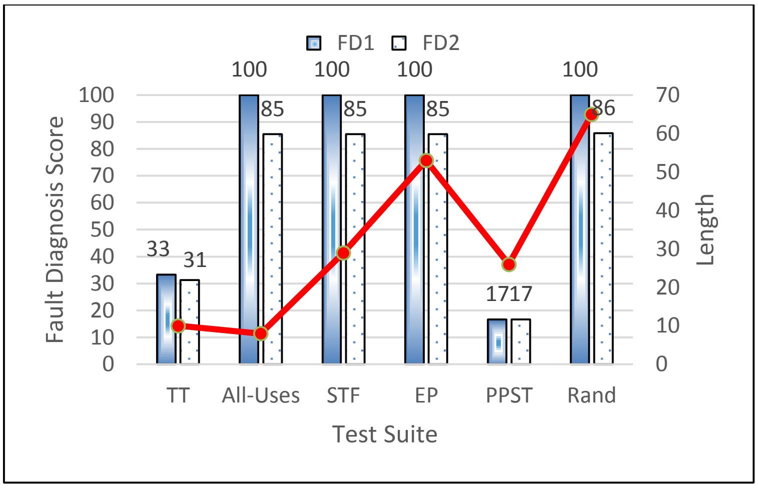

Figure 4.28 FD1 and FD2 for each test suite over all fault candidates

f) Test Suites FD1 and FD2 Assessment for the SCP Example

In this section, we assess the fault diagnosis capabilities (FD1 and FD2) for the SCP example.

(f.1) FD1 Assessment per Considered Diagnostic Candidates

Table 4.24 shows FD1 for each test suite over each type of fault candidates.

Table 4.24 FD1 for each test suite over each type of fault candidates

FD1	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
TT	8	100	100	100	100	100	100	92
All-Uses	7	100	100	100	100	100	100	93
STF	13	100	100	100	100	100	100	89
EP	35	100	100	100	100	100	100	94
PPST	24	100	100	100	100	100	100	89
Rand	45	100	100	100	100	100	100	94

The Figure 4.29 shows FD1 for each test suite over each type of fault candidates.

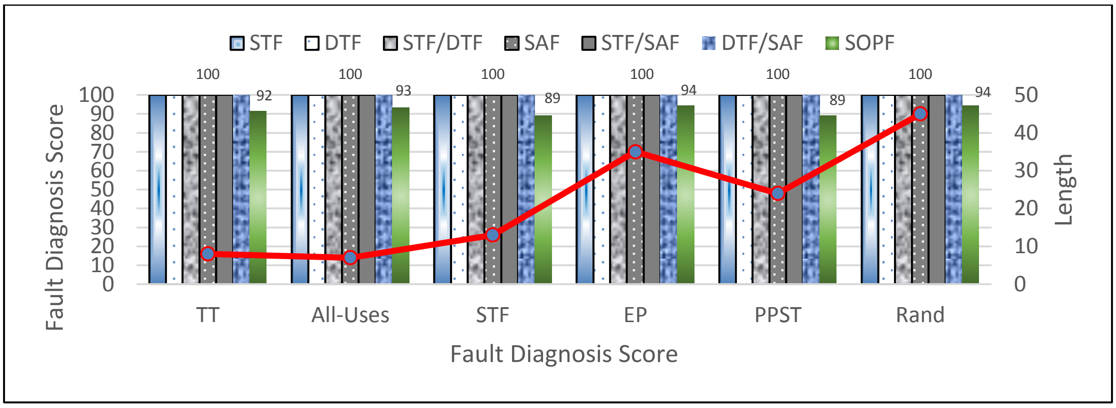

Figure 4.29 FD1 for each test suite over each type of fault candidates

(f.2) FD2 Assessment per Considered Diagnostic Candidates

Table 4.25 shows FD2 for each test suite over each type of fault candidates.

Table 4.25 FD2 for each test suite over each type of fault candidates

FD2	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
TT	8	75	9	8	63	53	39	70
All-Uses	7	75	9	8	69	60	48	74
STF	13	75	9	8	63	53	39	65
EP	35	75	9	8	69	60	48	72
PPST	24	75	9	8	63	53	39	65
Rand	45	75	9	8	69	60	48	72

The Figure 4.30 shows FD2 for each test suite over each type of fault candidates.

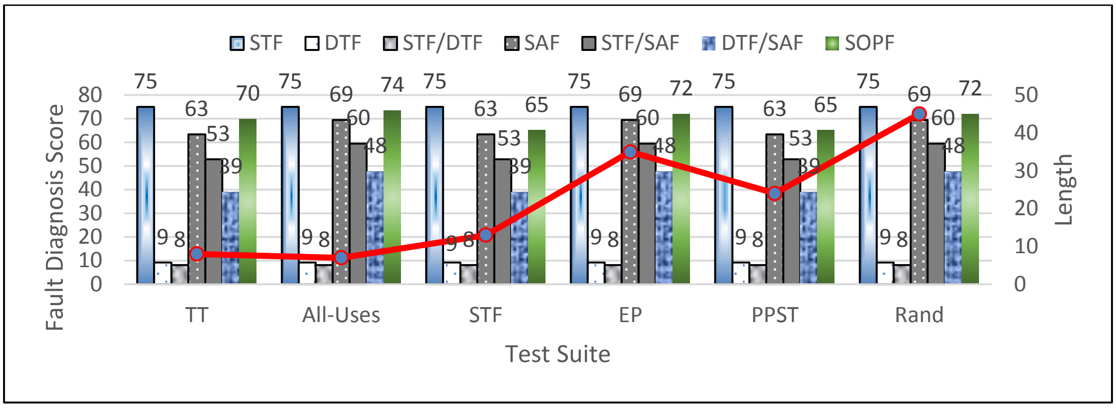

Figure 4.30 FD2 for each test suite over each type of fault candidates

(f.3) FD1 Assessment per Considered Assignment Diagnostic Candidates

Table 4.26 shows FD1 for each test suite over each type of assignment fault candidates.

Table 4.26 FD1 for each test suite over assignment fault candidates

FD1	Length	SAD	SARHS
TT	8	100	100
All-Uses	7	100	100
STF	13	100	100
EP	35	100	100
PPST	24	100	100
Rand	45	100	100

The Figure 4.31 shows FD1 for each test suite over each type of assignment fault candidates.

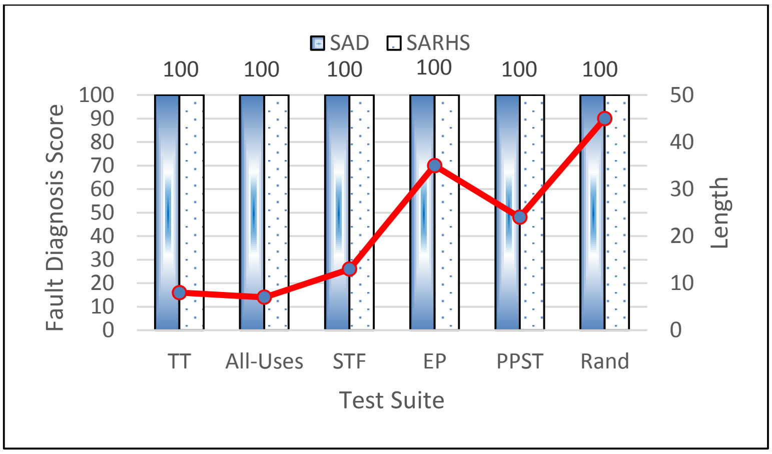

Figure 4.31 FD1 for each test suite over assignment fault candidates

(f.4) FD2 Assessment per Considered Assignment Diagnostic Candidates

Table 4.27 shows FD2 for each test suite over each type of assignment fault candidates.

Table 4.27 FD2 for each test suite over assignment fault candidates

FD2	Length	SAD	SARHS
TT	8	100	63
All-Uses	7	100	70
STF	13	100	63
EP	35	100	63
PPST	24	100	63
Rand	45	100	63

The Figure 4.32 shows FD2 for each test suite over each type of assignment fault candidates.

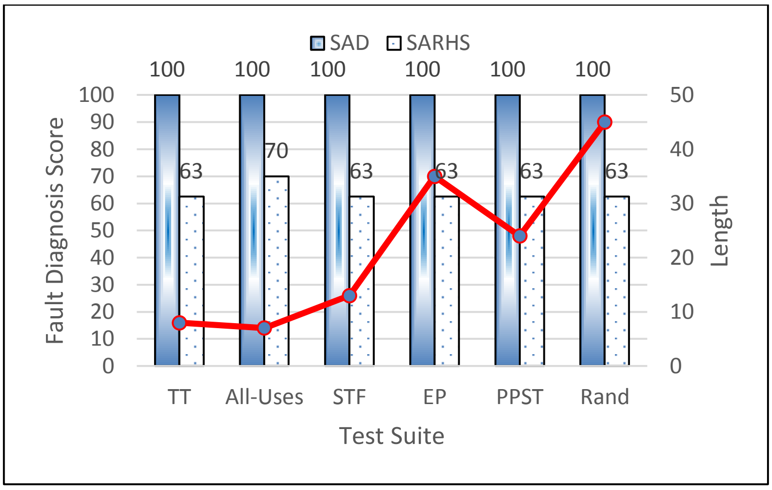

Figure 4.32 FD2 for each test suite over assignment fault candidates

(f.5) FD1 and FD2 Assessment considering all Diagnostic Candidates

Table 4.28 shows the fault diagnosis details for each test suite over all fault candidates.

Table 4.28 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	STF	EP	PPST	Rand
D	31	31	31	31	31	31
$\Im'$	13.9	13.9	13.9	13.9	13.9	13.9
$\lvert\Im\rvert$	4.9	4.9	4.9	4.9	4.9	4.9
AVG($\lvert\Im_i\rvert$)	4.1	4.1	4.1	4.1	4.1	4.4
$\Im_{TSj}^{dis\_from\_Spec}$	3.9	4.4	3.8	4.3	3.8	4.3
FD1	99.1	99.3	98.8	99.4	98.8	99.4
FD2	53.3	56.9	52.8	55.9	52.8	55.9
FD3	100	100	100	100	100	100
FD4	75.9	78.3	72.9	75.9	72.9	75.9
TSj Length	8	7	13	35	24	45
$TS_{j'}$ Length	4	4	4	4	4	4
$TS_{j'}$ Text Cases	1	1	1	1	1	1
$TS_j \cup TS_{j'}$ Length	12	11	17	39	28	49

The Figure 4.33 shows FD1 and FD2 for each test suite over all fault candidates.

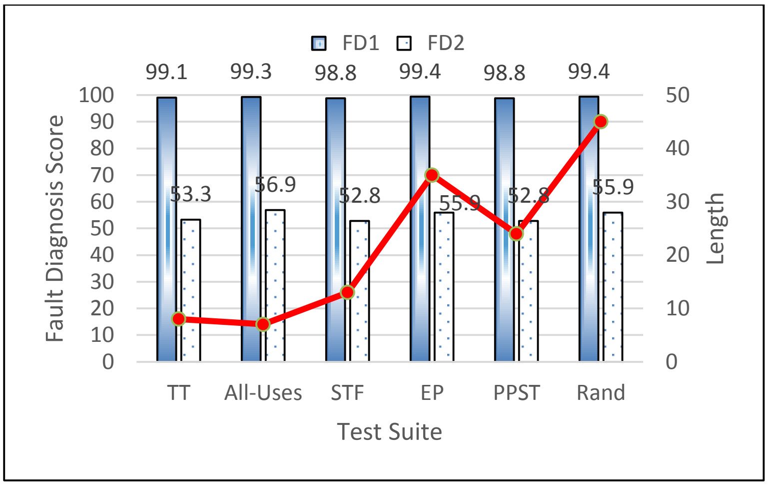

Figure 4.33 FD1 and FD2 for each test suite over all fault candidates

4.7.2 Assessment of FD1 and FD2 per All Considered Examples

In this section, we assess the fault diagnosis capabilities (FD1 and FD2) for all considered examples.

(a) FD1 Assessment per Considered Diagnostic Candidates

Table 4.29 shows the average FD1 for each test suite over each type of fault candidates for the six implementations.

Table 4.29 Average FD1 for each test suite over each type of fault candidates for the six implementations

FD1	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
TT	19	96	95	95	93	94	96	97
All-Uses	15	90	94	94	98	92	95	98
SITS	78	96	99	99	99	96	99	
STF	60	98	99	99	94	96	99	96
EP	73	95	98	98	100	96	98	97
PPST	75	97	99	99	95	95	98	95
Rand	106	87	88	88	93	86	90	98

The Figure 4.34 shows the average FD1 for each test suite over each type of fault candidates for the six implementations.

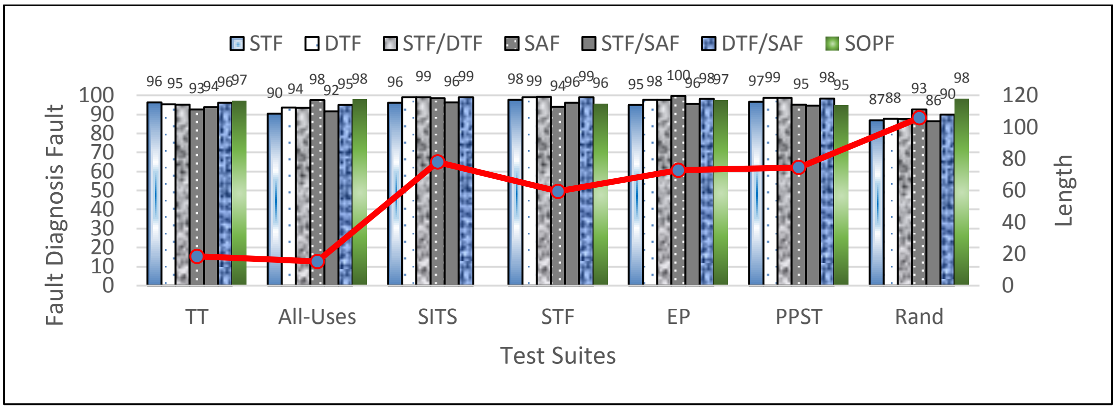

Figure 4.34 Average FD1 for each test suite over each type of fault candidates for the six implementations

(b) FD2 Assessment per Considered Diagnostic Candidates

Table 4.30 shows the average FD2 for each test suite over each type of fault candidates for the six implementations.

Table 4.30 Average FD2 for each test suite over each type of fault candidates for the six implementations

FD2	Length	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
TT	19	90	72	71	50	68	58	72
All-Uses	15	83	64	63	63	76	75	73
SITS	78	96	99	98	64	96	82	
STF	60	91	75	75	65	82	72	70
EP	73	89	74	73	67	81	72	72
PPST	75	90	75	74	52	69	59	41
Rand	106	81	64	64	64	73	65	73

The Figure 4.35 shows the average FD2 for each test suite over each type of fault candidates for the six implementations.

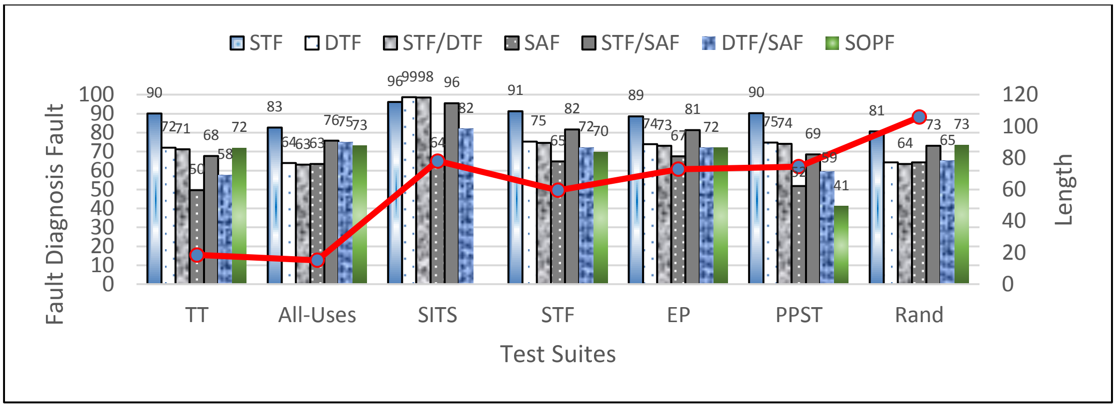

Figure 4.35 Average FD2 for each test suite over each type of fault candidates for the six implementations

(c) FD1 Assessment per Considered Assignment Diagnostic Candidates

Table 4.31 shows the average FD1 for each test suite over each type of assignment fault candidates for the six implementations.

Table 4.31 Average FD1 for each test suite over assignment fault candidates for the six implementations

FD1	Length	SAI	SAD	SARHS
TT	20	85	98	75
All-Uses	17	100	99	97
SITS	79	100	100	99
STF	66	100	98	95
EP	77	100	100	100
PPST	84	98	100	75
Rand	114	85	98	92

The Figure 4.36 shows the average FD1 for each test suite over each type of assignment fault candidates for the six implementations.

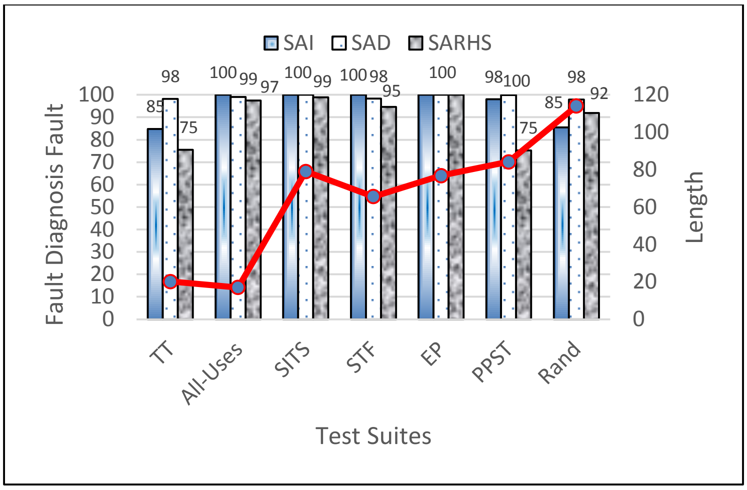

Figure 4.36 Average FD1 for each test suite over assignment fault candidates for the six implementations

(d) FD2 Assessment per Considered Assignment Diagnostic Candidates

Table 4.32 shows the average FD2 for each test suite over each type of assignment fault candidates for the six implementations.

Table 4.32 Average FD2 for each test suite over assignment fault candidates for the six implementations

FD2	Length	SAI	SAD	SARHS
TT	20	60	97	56
All-Uses	17	83	98	74
SITS	79	100	100	74
STF	66	75	97	75
EP	77	75	98	79
PPST	84	90	99	56
Rand	114	77	97	73

The Figure 4.37 shows the average FD2 for each test suite over each type of assignment fault candidates for the six implementations.

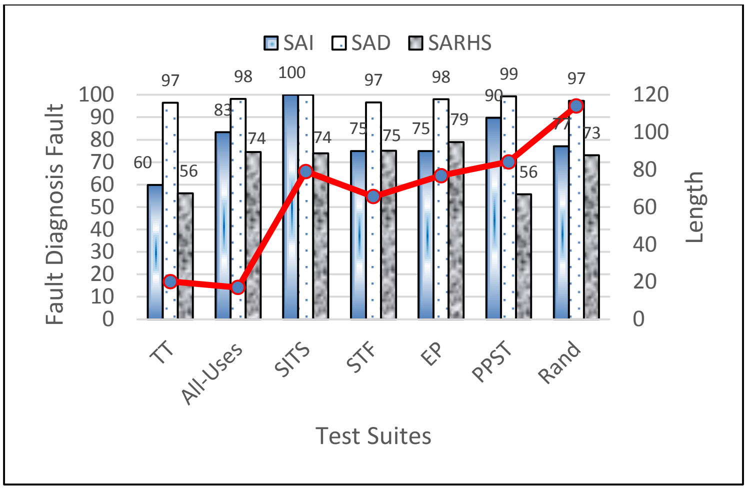

Figure 4.37 Average FD2 for each test suite over assignment fault candidates for the six implementations

(e) FD1 and FD2 Assessment considering all Diagnostic Candidates

Table 4.33 shows the average fault diagnosis details for each test suite over all fault candidates for the six implementations.

Table 4.33 Average fault diagnosis details for each test suite over all fault candidates for the six implementations

	TT	All-Uses	SITS	STF	EP	PPST	Rand
D	449	267.7	776.6	449	449	449	449
$\mathfrak{I}'$	400.4	210.3	761	400.4	400.4	400.4	400.4
$\lvert\mathfrak{I}\rvert$	310.9	172.8	611.5	310.9	310.9	310.9	310.9
AVG($\lvert\mathfrak{I}_i\rvert$)	2.0	2.2	1.2	2	2	2	2.1
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	287.9	153.5	606.0	307.1	296.9	306.2	305.6
FD1	84.7	95.4	98.4	97.6	98	83.7	91
FD2	70	77.3	91.3	80.3	80.5	69.4	75
FD3	100	100	100	100	100	100	100
FD4	88.4	89.8	96.6	88.6	86.9	88.2	90.1
TSj Length	23.5	24.0	78	71	93	96.7	109.7
$TS_{j'}$ Length	48.0	76.5	40.1	21.3	47.5	25.0	142.4
$TS_{j'}$ Text Cases	13.5	23.8	8.6	4.5	11.8	5.7	44
$TS_j \cup TS_{j'}$ Length	71.5	100.5	118.1	92.6	140.0	121.7	252.1

The Figure 4.38 shows the average FD1 and FD2 for each test suite over all fault candidates for the six implementations.

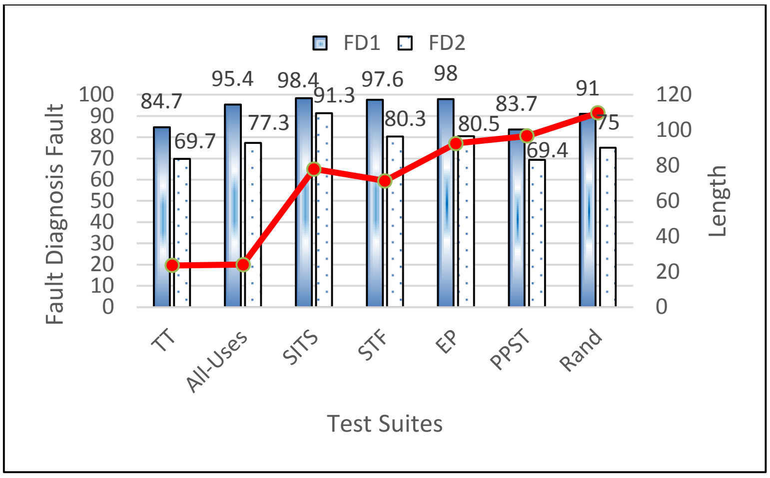

Figure 4.38 Average FD1 and FD2 for each test suite over all fault candidates for the six implementations

4.7.3 Ranking of Test Suites

In this section, we rank the test suites according to the criteria described before.

(a) Ranking Test Suites Based on Fault Diagnosis Capabilities Considering All Types of Diagnostic Candidates

Table 4.34 shows the FD1 ranking of each test suite over each type of fault candidates considering the fault diagnosis score only.

Table 4.34 FD1 ranking of test suites considering the fault diagnosis score only

Rank	**Fault candidates**						
	STF	**DTF**	**STF/DTF**	**SAF**	**STF/SAF**	**DTF/SAF**	**SOPF**
1	STF	SITS STF PPST	SITS STF PPST	EP	SITS STF EP	SITS STF	Rand All-Uses
2	PPST	EP	EP	SITS	PPST	PPST EP	EP TT
3	TT SITS	TT	TT	All-Uses	TT	TT	STF
4	EP	All-Uses	All-Uses	PPST	All-Uses	All-Uses	PPST
5	All-Uses	Rand	Rand	STF	Rand	Rand	SITS
6	Rand			Rand TT			

Table 4.35 shows the FD2 ranking of each test suite over each type of fault candidates considering the fault diagnosis score only.

Table 4.35 FD2 ranking of test suites considering the fault diagnosis score only

Rank	Fault candidates						
	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
1	SITS STF	SITS	SITS	EP	STF	SITS	Rand All-Uses
2	PPST TT	STF PPST	STF	STF	EP	All-Uses	EP TT
3	EP	EP	PPST	Rand SITS	All-Uses	STF EP	STF
4	All-Uses	TT	EP	All-Uses	Rand	Rand	PPST
5	Rand	Rand All-Uses	TT	PPST	PPST	PPST	SITS
6			Rand	TT	TT	TT	
7			All-Uses				

(b) Ranking Test Suites Based on Fault Diagnosis Capabilities and Test Suites Length Considering All Types of Diagnostic Candidates

Table 4.36 shows the FD1 ranking of each test suite over each type of fault candidates considering the FD1-length score.

Table 4.36 FD1 ranking of test suites considering the fault diagnosis score and length

Rank	Fault candidates						
	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
1	All-Uses	All-Uses	All-Uses	All-Uses	All-Uses	All-Uses	All-Uses
2	TT	TT	TT	TT	TT	TT	TT
3	STF	STF	STF	STF	STF	STF	STF
4	EP	EP	EP	EP	EP	EP	EP
5	PPST	PPST	PPST	PPST	PPST	PPST	PPST
6	SITS	SITS	SITS	SITS	SITS	SITS	Rand
7	Rand	Rand	Rand	Rand	Rand	Rand	SITS

Table 4.37 shows the FD2 ranking of each test suite over each type of fault candidates considering the FD2-length score.

Table 4.37 FD2 ranking of test suites considering the fault diagnosis score and length

Rank	Fault candidates						
	STF	DTF	STF/DTF	SAF	STF/SAF	DTF/SAF	SOPF
1	All-Uses	All-Uses	All-Uses	All-Uses	All-Uses	All-Uses	All-Uses
2	TT	TT	TT	TT	TT	TT	TT
3	STF	STF	SITS	STF	STF	STF	STF
4	SITS	SITS	STF	EP	SITS	SITS	EP
5	EP	EP	EP	SITS	EP	EP	Rand
6	PPST	PPST	PPST	PPST	PPST	PPST	PPST
7	Rand	Rand	Rand	Rand	Rand	Rand	SITS

(c) Ranking Test Suites Based on Fault Diagnosis Capabilities Considering Candidates with Assignment Faults

Table 4.38 shows the FD1 ranking of each test suite over each type of assignment fault candidates considering the fault diagnosis score only.

Table 4.38 FD1 ranking of test suites over assignment faults considering fault diagnosis score only

Rank	Assignment Fault candidates		
	SAI	**SAD**	**SARHS**
1	All-Uses SITS STF EP	SITS EP PPST	EP
2	PPST	All-Uses	SITS
3	Rand TT	STF TT Rand	All-Uses
4			STF
5			Rand
6			TT PPST

Table 4.39 shows the FD2 ranking of each test suite over each type of assignment fault candidates considering the fault diagnosis score only.

Table 4.39 FD2 ranking of test suites over assignment faults considering fault diagnosis score only

Rank	Assignment Fault candidates		
	SAI	**SAD**	**SARHS**
1	SITS	SITS	EP
2	PPST	PPST	STF
3	All-Uses	All-Uses EP	All-Uses SITS
4	Rand	Rand STF TT	Rand
5	STF EP		TT PPST
6	TT		

(d) Ranking Test Suites Based on Fault Diagnosis Capabilities and Test Suites Length Considering Candidates with Assignment Faults

Table 4.40 shows the FD1 ranking of each test suite over each type of assignment fault candidates considering the FD1-length score.

Table 4.40 FD1 ranking of test suites over assignment faults considering fault diagnosis score and length

Rank	Assignment Fault candidates		
	SAI	**SAD**	**SARHS**
1	All-Uses	All-Uses	All-Uses
2	TT	TT	TT
3	STF	STF	STF
4	EP	EP	EP
5	SITS	SITS	SITS
6	PPST	PPST	PPST
7	Rand	Rand	Rand

Table 4.41 shows the FD2 ranking of each test suite over each type of assignment fault candidates considering FD2-length score.

Table 4.41 FD2 ranking of test suites over assignment faults considering fault diagnosis score and the length

Rank	Assignment Fault candidates		
	SAI	**SAD**	**SARHS**
1	All-Uses	All-Uses	All-Uses
2	TT	TT	TT
3	SITS	STF	STF
4	STF	EP	EP
5	PPST	SITS	SITS
6	EP	PPST	PPST
7	Rand	Rand	Rand

4.7.4 Fault Diagnosis Capabilities (FD1 and FD3) of Test Suite

In this section, for each specification, we determine the fault diagnosis capabilities (FD1 and FD3) of each test suite per all diagnosis candidates. At the end of this section, we rank the test suites based on the increment in length of the new test suite.

a) Test Suites FD1 and FD3 Assessment for the TFTP Example

Table 4.42 shows the fault diagnosis details for each test suite over all diagnosis candidates.

Table 4.42 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	SITS	STF	EP	PPST	Rand
D	735.9	738	735.9	735.9	735.9	735.9	735.9
$\Im'$	715	715	715	715	715	715	715
$\lvert\Im\rvert$	643.6	643.4	643.6	643.6	643.6	643.6	643.6
AVG($\lvert\Im_i\rvert$)	1.1	1.1	1.1	1.1	1.1	1.1	1.1
$\Im_{TSj}^{dis\_from\_Spec}$	613.3	548.6	643.6	642.4	627.1	633	622.6
FD1	98.2	98.8	99.7	99.7	99.5	99.2	98.2
FD2	97.8	98.3	99.3	99.4	99.1	98.9	97.8
FD3	100	100	100	100	100	100	100
FD4	99.7	99.5	99.6	99.6	99.6	99.7	99.6
TSj Length	40	52	76	100	171	159	88
$TS_{j'}$ Length	74.8	50.8	11.2	11.2	21.2	24.2	83.2
$TS_{j'}$ Text Cases	22.2	16.8	1.6	1.6	5.2	6	24.6
$TS_j \cup TS_{j'}$ Length	115	103	87	111	192	183	171

The Figure 4.39 shows FD1 and FD3 for each test suite over all fault candidates.

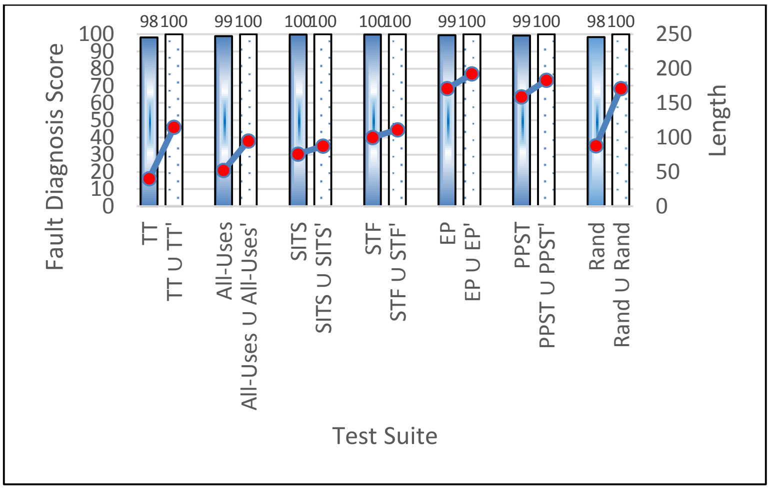

Figure 4.39 FD1 and FD3 for each test suite over all fault candidates

b) Test Suites FD1 and FD3 Assessment for the CD Player Example

Table 4.43 shows the fault diagnosis details for each test suite over all diagnosis candidates.

Table 4.43 Fault diagnosis details for each test suite over all fault candidates

	TT	SITS	STF	EP	PPST	Rand
D	1359.3	1359.3	1359.3	1359.3	1359.3	1359.3
$\mathfrak{I}'$	1350.7	1350.7	1350.7	1350.7	1350.7	1350.7
$\vert\mathfrak{I}\vert$	1001.6	1001.6	1001.6	1001.6	1001.6	1001.6
AVG($\vert\mathfrak{I}_i\vert$)	1.1	1.1	1.1	1.1	1.1	1.1
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	896.6	985	985.3	943.2	988.7	993.6
FD1	93.6	99.3	99	99.6	99.1	96
FD2	87.6	93.4	93.1	93.4	92.9	89
FD3	100	100	100	100	100	100
FD4	92.2	91.1	92.4	91.7	93	90.9
TSj Length	31	82	166	161	197	230
$TS_{j'}$ Length	65	53.2	15.9	38.3	13.6	64.4
$TS_{j'}$ Text Cases	17.6	10.5	3.6	9.2	2.6	16.4
$TS_j \cup TS_{j'}$ Length	96	135.2	181.9	199.3	210.6	294.4

The Figure 4.40 shows FD1 and FD3 for each test suite over all fault candidates.

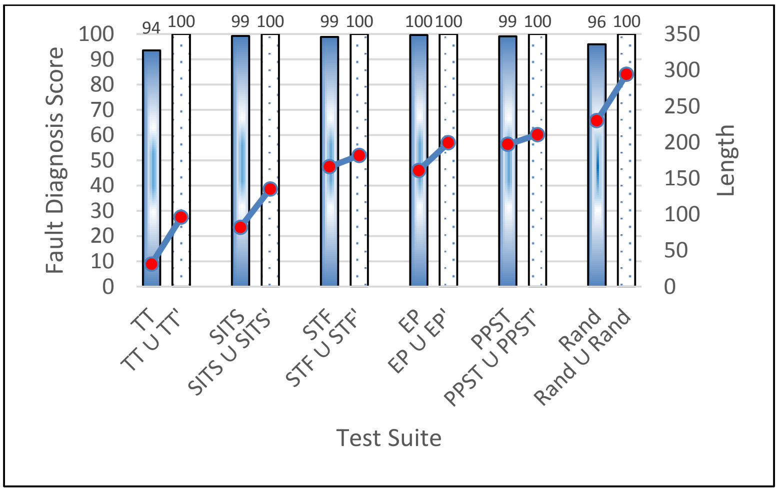

Figure 4.40 FD1 and FD3 for each test suite over all fault candidates

c) Test Suites FD1 and FD3 Assessment for the POP3 Example

Table 4.44 shows the fault diagnosis details for each test suite over all diagnosis candidates.

Table 4.44 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	SITS	STF	EP	PPST	Rand
D	234.7	234.7	234.7	234.7	234.7	234.7	234.7
$\mathfrak{I}'$	217.3	217.3	217.3	217.3	217.3	217.3	217.3
$\lvert\mathfrak{I}\rvert$	189.4	189.4	189.4	189.4	189.4	189.4	189.4
AVG($\lvert\mathfrak{I}_i\rvert$)	1.4	1.4	1.4	1.4	1.4	1.4	1.4
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	189.4	189.4	189.4	189.4	189.4	189.4	189.4
FD1	92.1	83	96.3	96.3	90.2	94.1	66
FD2	77	67.8	81.2	81.2	75.2	79	51.1
FD3	100	100	100	100	100	100	100
FD4	99.2	99.2	99.2	99.2	99.2	99.2	99.2
TSj Length	22	17	76	74	56	32	110
$TS_{j'}$ Length	117	232.6	55.8	56.4	168.6	81.4	525.4
$TS_{j'}$ Text Cases	35.6	74.6	13.8	13.8	42.4	21	174
$TS_j \cup TS_{j'}$ Length	139	249.6	131.8	130.4	224.6	113.4	635.4

The Figure 4.41 shows FD1 and FD3 for each test suite over all fault candidates.

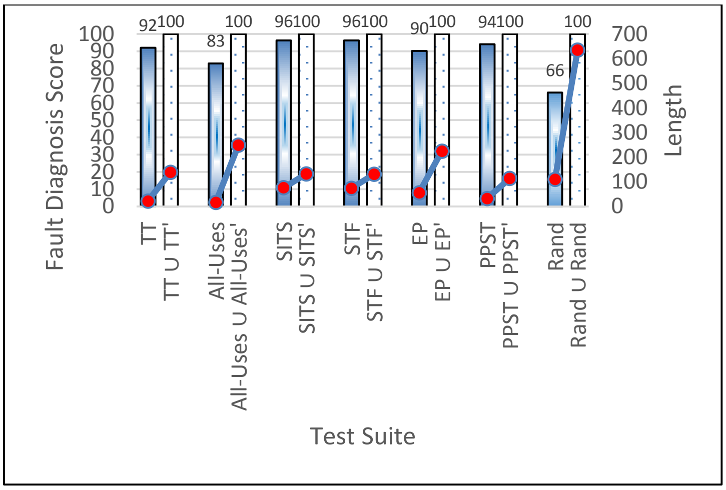

Figure 4.41 FD1 and FD3 for each test suite over all fault candidates

d) Test Suites FD1 and FD3 Assessment for the Initiator Example

Table 4.45 shows the fault diagnosis details for each test suite over all diagnosis candidates.

Table 4.45 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	STF	EP	PPST	Rand
D	289.1	289.1	289.1	289.1	289.1	289.1
$\mathfrak{I}'$	103.7	103.7	103.7	103.7	103.7	103.7
$\lvert\mathfrak{I}\rvert$	24.7	24.7	24.7	24.7	24.7	24.7
AVG($\lvert\mathfrak{I}_i\rvert$)	3.1	3.1	3.1	3.1	3.1	3.1
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	22.7	23.8	20.2	15.8	21	22.6
FD1	92.2	95.9	91.8	99.3	94.3	86.6
FD2	71.2	77.7	70.1	73.8	76	70.2
FD3	100	100	100	100	100	100
FD4	82.1	82.3	78.9	68.2	79.5	84.9
TS_j Length	30	36	46	79	142	120
$TS_{j'}$ Length	23.3	18.4	19	5.5	22.8	35
$TS_{j'}$ Text Cases	3.4	2.7	2.8	1	2.8	5
$TS_j \cup TS_{j'}$ Length	53.3	54.4	65	84.5	164.8	155

The Figure 4.42 shows FD1 and FD3 for each test suite over all fault candidates.

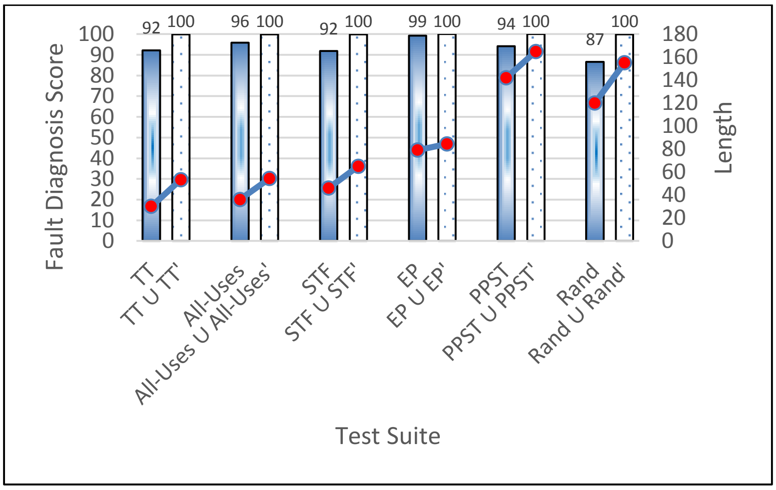

Figure 4.42 FD1 and FD3 for each test suite over all fault candidates

e) Test Suites FD1 and FD3 Assessment for the Responder Example

Table 4.46 shows the fault diagnosis details for each test suite over all diagnosis candidates.

Table 4.46 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	STF	EP	PPST	Rand
D	45.8	45.8	45.8	45.8	45.8	45.8
$\mathfrak{I}'$	1.8	1.8	1.8	1.8	1.8	1.8
$\lvert\mathfrak{I}\rvert$	1.4	1.4	1.4	1.4	1.4	1.4
AVG($\lvert\mathfrak{I}_i\rvert$)	1.3	1.3	1.3	1.3	1.3	1.3
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	1.3	1.3	1.3	1.3	1.1	1.4
FD1	33.3	100	100	100	16.7	100
FD2	31.3	85.4	85.4	85.4	16.7	85.8
FD3	100				100	
FD4	81				85	
TSj Length	10	8	29	53	26	65
$TS_{j'}$ Length	4				4	
$TS_{j'}$ Text Cases	1				1	
$TS_j \cup TS_{j'}$ Length	14	8	29	53	30	65

The Figure 4.43 shows FD1 and FD3 for each test suite over all fault candidates.

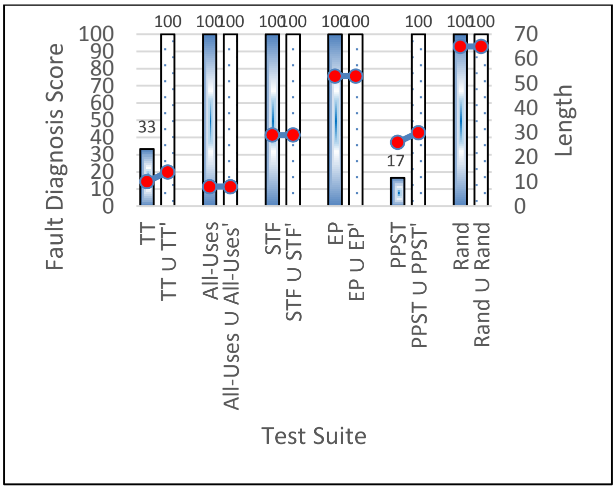

Figure 4.43 FD1 and FD3 for each test suite over all fault candidates

f) Test Suites FD1 and FD3 Assessment for the SCP Example

Table 4.47 shows the fault diagnosis details for each test suite over all diagnosis candidates.

Table 4.47 Fault diagnosis details for each test suite over all fault candidates

	TT	All-Uses	STF	EP	PPST	Rand
D	31	31	31	31	31	31
$\mathfrak{I}'$	13.9	13.9	13.9	13.9	13.9	13.9
$\lvert\mathfrak{I}\rvert$	4.9	4.9	4.9	4.9	4.9	4.9
AVG($\lvert\mathfrak{I}_i\rvert$)	4.1	4.1	4.1	4.1	4.1	4.4
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	3.9	4.4	3.8	4.3	3.8	4.3
FD1	99.1	99.3	98.8	99.4	98.8	99.4
FD2	53.3	56.9	52.8	55.9	52.8	55.9
FD3	100	100	100	100	100	100
FD4	75.9	78.3	72.9	75.9	72.9	75.9
TSj Length	8	7	13	35	24	45
$TS_{j'}$ Length	4	4	4	4	4	4
$TS_{j'}$ Text Cases	1	1	1	1	1	1
$TS_j \cup TS_{j'}$ Length	12	11	17	39	28	49

The Figure 4.44 shows FD1 and FD3 for each test suite over all fault candidates.

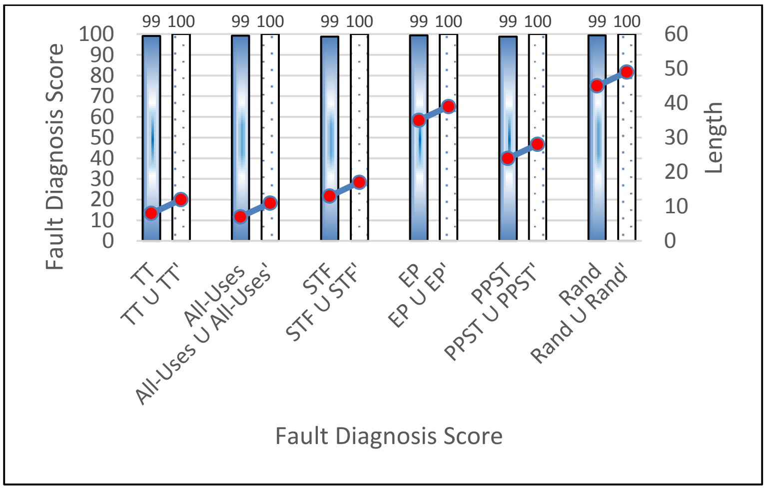

Figure 4.44 FD1 and FD3 for each test suite over all fault candidates

4.7.5 Assessment of FD1 and FD3 per All Considered Examples

Table 4.48 shows the average fault diagnosis details for each test suite over all diagnosis candidates for the six implementations.

Table 4.48 Average fault diagnosis details for each test suite over all fault candidates for six implementations

	TT	All-Uses	SITS	STF	EP	PPST	Rand
D	449	267.7	776.6	449	449	449	449
$\mathfrak{I}'$	400.4	210.3	761	400.4	400.4	400.4	400.4
$\lvert\mathfrak{I}\rvert$	310.9	172.8	611.5	310.9	310.9	310.9	310.9
AVG($\lvert\mathfrak{I}_i\rvert$)	2.0	2.2	1.2	2	2	2	2.1
$\mathfrak{I}_{TSj}^{dis\_from\_Spec}$	287.9	153.5	606.0	307.1	296.9	306.2	305.6
FD1	84.7	95.4	98.4	97.6	98	83.7	91
FD2	70	77.3	91.3	80.3	80.5	69.4	75
FD3	100	100	100	100	100	100	100
FD4	88.4	89.8	96.6	88.6	86.9	88.2	90.1
TSj Length	23.5	24.0	78	71	93	96.7	109.7
$TS_{j'}$ Length	48.0	76.5	40.1	21.3	47.5	25.0	142.4
$TS_{j'}$ Text Cases	13.5	23.8	8.6	4.5	11.8	5.7	44
$TS_j \cup TS_{j'}$ Length	71.5	100.5	118.1	92.6	140.0	121.7	252.1

The Figure 4.45 shows the average FD1 and FD3 for each test suite over all fault candidates for the six implementations.

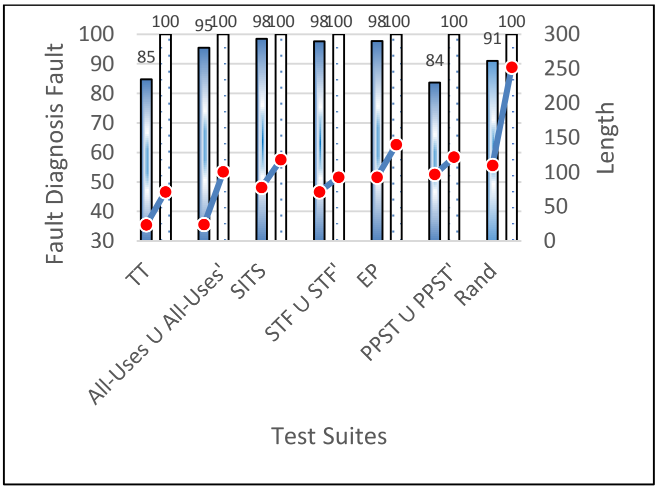

Figure 4.45 FD1 and FD3 for each test suite over all fault candidates for six implementations

According to the results depicted in Figure 4.45, the best test suite in terms of new test suite length is TT (71.5) followed by the STF (92.6), All-Uses (100.5), SITS (118.1), PPST (121.7), EP (140), and then Rand (252.1) test suites.

4.7.6 Summary of All Obtained Results

Below we include a summary of the experimental results in Sections 4.7.1 to 4.7.5:

- Using the fault localization score FD1, on average, the SITS (98.4 %) test suites have the best FD1 score. The EP (98 %) and STF (97.6 %) test suites have comparable FD1 scores, but these score are less than that of the SITS by approximately 0.5 percent. The All-Uses (95.4 %) and Rand (91 %) test suites scores are less than that of the SITS by approximately 3 and 7.5 percent, respectively. The TT (84.7 %) and PPST (83.7 %) test suites have comparable FD1 scores, but these scores are less than that of the SITS by approximately 14 percent.
- Using the fault localization score FD2, on average, the SITS (91.3 %) test suites have the best FD2 score. EP (80.5 %) and STF (80.3 %) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 10.9 percent. The All-Uses (77.3 %) and Rand (75 %) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 15.2 percent. The TT (69.7 %) and PPST (69.4 %) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 21.8 percent.
- When ranking the test suites based on a score computed using the fault diagnosis score FD1 and the length of the test suite, i.e. based on FD1-length score, the All-Uses (393.5) and TT (357) test suites have comparable scores, which are greater than the scores of other test suites by approximately 71.5 percent. The STF (135.5), SITS (124.9), EP (104.9), PPST (85.7) and Rand (82.2) test suites have comparable scores, but these scores are less than the scores of the All-Uses and TT test suites by approximately 71.5 percent.
- When ranking the test suites based on a score computed using the fault diagnosis score FD2 and the length of the test suite, i.e. based on FD2-length score, the All-Uses (318.7) and TT (293.7) test suites have comparable scores, which are greater than the scores of other test suites by approximately 70.5 percent. The SITS (115.9), STF (111.5), EP (86.1), PPST (71.1) and Rand

(67.7) test suites have comparable scores, but these scores are less than the scores of the All-Uses and TT test suites by approximately 70.5 percent.

- When comparing the test suites in terms of the total length of a test suite in addition to the length of the additional diagnostic tests needed for locating the fault, the best performing test suite is the TT, followed by the STF, All-Uses, SITS, PPST, EP, and then the Rand test suites.

4.8 Related work on Fault Diagnosis

In the software domain where a system may be represented as an FSM, some work has already been done for the diagnostic and fault localization problems [37] [38] [39].

In [37], [39] and [40] the differences between a system's specification and its implementation is located under the assumption of a single fault in the implementation.

In [41] the differences can be located for multiple faults under the assumption that each of the faults is reachable through non-faulty transitions.

In [42], considering a system consisting of two communicating FSMs, a method was presented to decide if it is possible to locate a faulty component machine, and if this is possible then tests for locating the fault(s) are derived.

In [43] a fault localization method for EFSMs is presented based on the derivation of mutants of particular type, represented in a compact way in a so-called fault function, and the derivation of (diagnostic) tests that distinguish fault functions, and thus their constituent mutants.

In this thesis, we present a method and conduct comprehensive experiments for assessing the fault localization capabilities of the considered EFSM-based test suites. In addition, for each considered test suite, the method determines the diagnostic tests required, in addition to the considered test suite, for locating a faulty IUT. Accordingly, an assessment of considered test suites is carried out based on the length of a test suite in addition to the length of the corresponding diagnostic tests.

Chapter 5: Conclusion

Testing based on formal models is widely used for deriving test suites for different kinds of reactive systems. In various application domains, such as communication protocols and other reactive systems, the specification can be represented in the form of an Extended Finite State Machine (EFSM), which is widely acknowledged as a very powerful model for test derivation. In practice, developing and applying these test suites to an Implementation Under Test (IUT) is time consuming and costly. Thus, determining high quality test suites reduces the cost of software testing.

In the first part of the thesis, we conduct experiments, assess, and compare the fault coverage of many EFSM-based and random test suites in order to determine the quality of these test suites, and thus reduce the cost of testing. In summary, based on the conducted experiments, the best performing test suites, in terms of fault coverage, are the SITS (61.4 %) followed by the PPST (59.6 %), TT (59.5 %), STF (59.2 %), All-Uses (56.3 %), Rand (55.2 %) and then the EP (50.2 %) test suites. However, when considering the *coverage-length* score, the TT (250.59) and All-Uses (232.15) test suites have comparable scores, and they outperform the other test suites by approximately 73 percent. The STF (82.11), SITS (77.99), PPST (60.99), EP (53.72) and Rand (49.80) test suites have comparable scores, but each of these test suites score less than the TT and the All-Uses test suites by approximately 73 percent. Test suite fault coverage of COI and COD faults is on average 86 %, and it is significantly higher than the coverage of mutants with other types of operator faults by approximately 29 percent. Test suite coverage of AORS, AORB, AOIS, AOIU, ROR and COR faults are comparable, but this coverage is less than the coverages of COI and COD by approximately 29 percent. Test suite coverage of conditional faults (73 %) is significantly higher than the coverage of mutants with arithmetic and relational faults by approximately 17.5 percent. Test suite coverage of mutants with arithmetic faults is comparable to the coverage of mutants with relational faults, but this coverage is less than the coverage of conditional faults by approximately 17.5 percent. SITS test suites have the best fault coverage of arithmetic faults (65 %), conditional faults (81 %) and relational faults (69 %). The remaining test suites have comparable coverages in terms of arithmetic and conditional and relational faults, but their coverage is less than the coverage of the SITS test suites by approximately 12 percent. When considering the *coverage-length* score, the TT and All-Uses test suites have comparable scores, and they outperform the other test suites in terms of score of arithmetic and conditional and relational faults by approximately 74 percent. The remaining test suites have comparable scores but they are less than the scores of the TT and All-Uses by approximately 74 percent.

While the purpose of conformance testing is to check if an IUT is different than its specification, an interesting complementary, yet more complex, step, called fault diagnosis or diagnostic testing, is to determine the faulty implementation, and thus find the differences between the specification and its implementation.

In the second part of the thesis, we present a method and conduct comprehensive experiments for assessing the fault localization capabilities of the EFSM-based test suites considered in the first part of this thesis. In addition, for each considered test suite, the method determines the diagnostic tests required, in addition to the considered test suite, for locating a faulty IUT. Accordingly, an assessment of considered test suites is carried out based on the length of a test suite in addition to the length of the corresponding diagnostic tests. Based on the conducted experiments, the following results are obtained. Using the fault localization score FD1, on average, the SITS (98.4 %) test suites have the best FD1 score. The EP (98 %) and STF (97.6 %) test suites have comparable FD1 scores, but these scores are less than that of the SITS by approximately 0.5 percent. The All-Uses (95.4 %) and Rand (91 %) test suites scores are less than that of the SITS by approximately 3 and 7.5 percent, respectively. The TT (84.7 %) and PPST (83.7 %) test suites have comparable FD1 scores, but these scores are less than that of the SITS by approximately 14 percent. Using the fault localization score FD2, on average, the SITS (91.3 %) test suites have the best FD2 score. The EP (80.5 %) and STF (80.3 %) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 10.9 percent. The All-Uses (77.3 %) and Rand (75 %) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 15.2 percent. The TT (69.7 %) and PPST (69.4 %) test suites have comparable FD2 scores, but these scores are less than that of the SITS by approximately 21.8 percent. When ranking the test suites based on a score computed using the fault diagnosis score FD1 and the length of the test suite, i.e. based on FD1-length score, the All-Uses (393.5) and TT (357) test suites have comparable scores, which are greater than the scores of the other test suites by approximately 71.5 percent. The STF (135.5), SITS (124.9), EP (104.9), PPST (85.7) and Rand (82.2) test suites have comparable scores, but these scores are less than the scores of the All-Uses and TT test suites by approximately 71.5 percent. When ranking the test suites based on a score computed using the fault diagnosis score FD2 and the length of the test suite, i.e. based on FD2-length score, the All-Uses (318.7) and TT (293.7) test suites have comparable scores, which are greater than the scores of other test suites by approximately 70.5 percent. The SITS (115.9), STF (111.5), EP (86.1), PPST (71.1) and Rand (67.7) test suites have comparable scores, but these scores are less than the scores of the All-Uses and TT test suites by approximately 70.5 percent. When comparing the test suites in terms of the total length of a test suite in addition to the length of the additional diagnostic tests needed for locating the fault, the best performing test suite is the TT, followed by the STF, All-Uses, SITS, PPST, EP and then the Rand test suites.

Possible extensions of our work presented in this thesis include assessing the code-based fault localization capabilities of the considered EFSM test suites. Thus, instead of using EFSM diagnostic candidates of a given specification, as done in this thesis, one can consider code mutants of the corresponding implementation of the specification.

References

[1] Y. S. Ma, J. Offutt, and Y. R. Kwon, "MuJava: an automated class mutation system," *Software Testing, Verification and Reliability*, vol. 15, pp. 97-133, 2005.

[2] Y. Jia and M. Harman, "An analysis and survey of the development of mutation testing," *IEEE Transactions on Software Engineering*, vol. 37, pp. 649-678, 2011.

[3] A. P. Mathur, *Foundations of Software Testing*: Addison-Wesley Professional, 2008.

[4] J. H. Andrews, L. C. Briand, Y. Labiche, and A. S. Namin, "Using mutation analysis for assessing and comparing testing coverage criteria," *IEEE Transactions on Software Engineering*, vol. 32, pp. 608-624, 2006.

[5] B. Korel and G. Koutsogiannakis, "Experimental comparison of code-based and model-based test prioritization," *IEEE International Conference on Software Testing, Verification and Validation Workshops*, pp. 77-84, 2009.

[6] K. El-Fakih, A. Simao, N. Jadoon, and J. C. Maldonado, "On Studying the Effectiveness of Extended Finite State Machine Based Test Selection Criteria," in *IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops*, pp. 222-229, 2014.

[7] K. El-Fakih, T. Salameh, and N. Yevtushenko, "On Code Coverage of Extended FSM Based Test Suites: An Initial Assessment," in *Testing Software and Systems*, pp. 198-204, 2014.

[8] P. Ammann and J. Offutt, "Introduction to software testing," *Cambridge University Press*, 2008.

[9] A. Petrenko, S. Boroday, and R. Groz, "Confirming configurations in EFSM testing," *IEEE Transactions on Software Engineering*, vol. 30, pp. 29-42, 2004.

[10] D. Hogrefe, "OSI formal specification case study: the Inres protocol and service," *Technical Report IAM-91-012*, University of Berne, 1992.

[11] Paul Ammann and Jeff Offutt. Graph Coverage Web Application [Online]. http://cs.gmu.edu:8080/offutt/coverage/GraphCoverage. (Accessed: September, 2014).

[12] A. Abdurazik, P. Ammann, W. Ding, and J. Offutt, "Evaluation of three specification-based testing criteria," S*ixth IEEE International Conference on Engineering of Complex Computer Systems*, pp. 179-187, 2000.

[13] M. H. Aouadi, K. Toumi, and A. Cavalli, "On modeling and testing security properties of vehicular networks," in *IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops (ICSTW),* pp. 42-50, 2014.

[14] N. Li, U. Praphamontripong, and J. Offutt, "An experimental comparison of four unit test criteria: Mutation, edge-pair, all-uses and prime path coverage," in *International Conference on Software Testing, Verification and Validation Workshops,* pp. 220-229, 2009.

[15] Wolski Sascha and Hennebrueder Sebastian. (2005, April) Eclipse Junit testing tutorial. [Online]. http://www.laliluna.de/articles/posts/eclipse-junit-testing-tutorial.html. (Accessed: February, 2014).

[16] A. J. Offutt, J. Pan, K. Tewary, and T. Zhang, "An experimental evaluation of data flow and mutation testing," *Softw., Pract. Exper.,* vol. 26, pp. 165-176, 1996.

[17] Vincenzi, M. Delamaro, E. Höhn, and J. C. Maldonado, "Functional, control and data flow, and mutation testing: Theory and practice," in *Testing Techniques in Software Engineering,* ed: Springer, pp. 18-58, 2010.

[18] P. G. Frankl, S. N. Weiss, and C. Hu, "All-uses vs mutation testing: an experimental comparison of effectiveness," *Journal of Systems and Software,* vol. 38, pp. 235-253, 1997.

[19] A. P. Mathur and W. E. Wong, "An empirical comparison of data flow and mutation-based test adequacy criteria," *Software Testing, Verification and Reliability,* vol. 4, pp. 9-31, 1994.

[20] W. E. Wong, A. P. Mathur, and J. C. Maldonado, "Mutation versus all-uses: An empirical evaluation of cost, strength and effectiveness," in *Software Quality and Productivity,* ed: Springer, pp. 258-265, 1995.

[21] S. Kansomkeat, J. Offutt, A. Abdurazik, and A. Baldini, "A comparative evaluation of tests generated from different UML diagrams," in *Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing,* pp. 867-872, 2008.

[22] Bourhfir, E. Aboulhamid, F. Khendek, and R. Dssouli, "Test cases selection from SDL specifications," *Computer Networks,* vol. 35, pp. 693-708, 2001.

[23] H. Ural and A. Williams, "Test generation by exposing control and data dependencies within system specifications in SDL," in *Proceedings of Sixth*

International Conference on Formal Description Techniques, VI, pp. 335-350, 1993.

[24] Y.-S. Ma and J. Offutt, "Description of class mutation operators for java," ed: November, 2005.

[25] R. A. DeMillo and A. J. Offutt, "Experimental results from an automatic test case generator," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 2, pp. 109-127, 1993.

[26] ITU-T, Recommendation Z.100-Specification and Description Language (SDL), Geneva, 1994.

[27] A. Offutt VI and K. N. King, "A Fortran 77 interpreter for mutation analysis," in *ACM SIGPLAN Notices*, pp. 177-188, 1987.

[28] R. H. Untch, A. J. Offutt, and M. J. Harrold, "Mutation analysis using mutant schemata," in *ACM SIGSOFT Software Engineering Notes*, pp. 139-148, 1993.

[29] A. Derezinska, "Quality assessment of mutation operators dedicated for c# programs," in *Sixth International Conference on Quality Software*, pp. 227-234, 2006.

[30] P. Chevalley and P. Thévenod-Fosse, "A mutation analysis tool for Java programs," *International journal on software tools for technology transfer*, vol. 5, pp. 90-103, 2003.

[31] Eclipse Team. Eclipse Classic 3.4.2 | Eclipse Packages. [Online]. http://www.eclipse.org/downloads/packages/eclipse-classic-342/ganymedesr2. (Accessed: February, 2014).

[32] A. Petrenko, G. Bochmann, and M. Yao, "On fault coverage of tests for finite state specifications," *Computer Networks and ISDN Systems*, vol. 29, pp. 81-106, 1996.

[33] M. Zhigulin, S. Prokopenko, and M. Forostyanova, "Detecting faults in TFTP implementations using Finite State Machines with timeouts," in *Proceedings of the Spring/Summer Young Researchers' Colloquium on Software Engineering*, 2012.

[34] A. Nikitin and N. Kushik, "On EFSM-based Test Derivation Strategies," in *Proceedings of the Spring/Summer Young Researchers' Colloquium on Software Engineering*, 2010.

[35] Seifert, "Conformance testing based on uml state machines," in *Formal Methods and Software Engineering*, ed: Springer, 2008, pp. 45-65.

[36] MuClipse Team. (2011, September) MuClipse - Mutation Testing tool for Eclipse. [Online]. http://muclipse.sourceforge.net/. (Accessed: February, 2014).

[37] A. Ghedamsi and G. Bochmann, "Test result analysis and diagnostics for finite state machines," in *Proceedings of the 12th International Conference on Distributed Computing Systems,* pp. 244-251, 1992.

[38] A. Ghedamsi and G. Bochmann, "Diagnostic tests for finite state machines," *Technical Report 807,* Dept. IRO, Univ de Montréal, January 1992.

[39] Lee and K. Sabnani, "Reverse-engineering of communication protocols," in *International Conference on Network Protocols,* pp. 208-216, 1993.

[40] A. Ghedamsi, G. Bochmann, and R. Dssouli, "Diagnostic tests for communicating finite state machines," in *Twelfth Annual International Phoenix Conference on Computers and Communications,* pp. 254-260, 1993.

[41] Ghedamsi, G. Bochmann, and R. Dssouli, "Multiple fault diagnosis for finite state machines," in *Twelfth Annual Joint Conference of the IEEE Computer and Communications Societies,* pp. 782-791, 1993.

[42] K. El-Fakih, N. Yevtushenko, and G. v. Bochmann, "Diagnosing multiple faults in communicating finite state machines," in *Formal Techniques for Networked and Distributed Systems*, pp. 85-100, 2002.

[43] K. El-Fakih, S. Prokopenko, N. Yevtushenko, and G. v. Bochmann, "Fault diagnosis in extended finite state machines," in *Testing of Communicating Systems*, ed: Springer, pp. 197-210, 2003.

[44] T. M. Salameh, "Testing and Assessment of Protocols and Systems Modeled as Extended Finite State Machines," *M.S. thesis,* Dept. Computer Science and Engineering, American University of Sharjah, 2013.

[45] Y.-S. Ma and J. Offutt, "Description of method-level mutation operators for java," *Technical Report,* Electronics and Telecommunications Research Institute, Korea, 2005.

Appendix A

Types of Code Based Mutants

Table A.1 Traditional (Method) Level Mutants [45]

<table>
<tr><th>Mutation Operator</th><th>Mutation Primitive</th><th>Example</th></tr>
<tr><td rowspan="5">AORB: Arithmetic Operator Replacement – Binary</td><td>aorb+</td><td>a / b => a + b</td></tr>
<tr><td>aorb-</td><td>a / b => a – b</td></tr>
<tr><td>aorb*</td><td>a / b => a * b</td></tr>
<tr><td>aorb/</td><td>a – b => a / b</td></tr>
<tr><td>Aorb%</td><td>a / b => a % b</td></tr>
<tr><td rowspan="4">AORS: Arithmetic Operator Replacement – Shortcut</td><td>aors++</td><td>p-- => p++</td></tr>
<tr><td>++aors</td><td>--p => ++p</td></tr>
<tr><td>aors--</td><td>p++ => p--</td></tr>
<tr><td>--aors</td><td>++p => --p</td></tr>
<tr><td rowspan="2">AOIU: Arithmetic Operator Insertion –Unary</td><td>aoiu+</td><td>count => +count</td></tr>
<tr><td>aoiu-</td><td>count => -count</td></tr>
<tr><td rowspan="4">AOIS: Arithmetic Operator Insertion – Shortcut</td><td>aois++</td><td>I => i++</td></tr>
<tr><td>++aois</td><td>I => ++i</td></tr>
<tr><td>aois--</td><td>I => i--</td></tr>
<tr><td>--aois</td><td>I => --i</td></tr>
<tr><td rowspan="6">ROR: Relational Operator Replacement</td><td>ror></td><td>a % 2 == 1 => a % 2 > 1</td></tr>
<tr><td>ror>=</td><td>a % 2 == 1 => a % 2 >= 1</td></tr>
<tr><td>ror<</td><td>a % 2 == 1 => a % 2 < 1</td></tr>
<tr><td>ror<=</td><td>a % 2 == 1 => a % 2 <= 1</td></tr>
<tr><td>ror==</td><td>a > 0 => a == 0</td></tr>
<tr><td>ror!=</td><td>a <= 0 => a != 0</td></tr>
<tr><td rowspan="5">COR: Conditional Operator Replacement</td><td>cor&&</td><td>1||a>0 => 1&&a>0</td></tr>
<tr><td>cor||</td><td>1&&a>0 => 1||a>0</td></tr>
<tr><td>cor&</td><td>a & b => a | b</td></tr>
<tr><td>cor|</td><td>a & b => a | b</td></tr>
<tr><td>cor^</td><td>a <= 0 || b <= 0 || c <= 0 => (a <= 0 || b <= 0) ^ c <= 0</td></tr>
<tr><td>COI: Conditional Operator Insertion</td><td>coi!</td><td>A== 0 => !(a == 0)</td></tr>
<tr><td>COD: Conditional Operator Deletion</td><td>cod!</td><td>!(a == 0) => a== 0</td></tr>
</table>

Vita

Mark Habib Hassoun was born on July 9, 1987, in Lattakia, Syria. He was educated in local public schools in Syria and graduated from National High School in 2005.

He Studied in Tishreen University in Lattakia, Syria, from which he graduated in 2011 and obtained a Bachelor of Science degree in Information Technology.

Mr. Hassoun moved to the United Arab Emirates in 2012 where he joined the Master's program in Computer Engineering at the American University of Sharjah.