ON STUDYING THE EFFECTIVENESS OF EXTENDED FINITE STATE MACHINE BASED TEST SELECTION CRITERIA

by

Noshad Khan Jadoon

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
July 2015

Approval Signatures

We, the undersigned, approve the Master's Thesis of Noshad Khan Jadoon

Thesis Title: On Studying the Effectiveness of Extended Finite State Machine Based Test Selection Criteria

Signature	**Date of Signature**
__________________________	_______________

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

__________________________ _______________

Dr. Amjad Ghawanme
Assistant Professor, Department of Electrical and Computer Engineering,
Khalifa University of Science and Technology
Thesis Committee Member

__________________________ _______________

Dr. Gerassimos Barlas
Professor, Department of Computer Science
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
Director of 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. I would also like to thank him for his patience, motivation, enthusiasm and immense knowledge. We would like to thank Dr. Adenilso Simao from the University of Sao Paulo, Brazil, for providing a software tool that is used along with our tool for the derivation of some types of considered test suites.

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). A number of 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 is time consuming and costly. Thus, it is desirable to determine high quality test suites in order to reduce the cost of testing. This research aims at determining and comparing the quality of various test suites. Using six realistic application examples, various known types of EFSM based test suites are derived and experiments are conducted to assess the fault coverage of these test suites. The assessment is carried out using EFSM mutants of these specifications, namely, EFSM mutants with single and double transfer faults, single assignment faults and single output parameter faults. The various types of considered test suites include single transfer fault, double transfer fault, all uses, single assignment fault, transition tour, state identifier, edge pair, prime path, prime path with side trip, and random test suites Ranking of the test suites, in terms of fault coverage and in terms of both coverage and test suite length, is established for each considered type of faults.

Search Terms: Extended Finite State Machine, Model Based Testing, Mutation Testing, Test Derivation, Test Assessment, Software Engineering, Software Testing.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AD	Single Assignment Deletion Fault
AI	Single Assignment Insertion Fault
CRHS	Single Assignment Right-hand-side Fault
DTF	Double Transfer Fault
EFSM	Extended Finite State Machine
EP	Edge Pair Test Suite
FSM	Finite State Machine
IUT	Implementation Under Testing
PP	Prime Path Test Suite
PPST	Prime Path with Side Trip Test Suite
Rand	Random Test Suite
SAF	Single Assignment Fault
SITS	State Identifiers Test Suite
SOPF	Single Output Parameter Fault
STF	Single Transfer Fault
TC	Test Case
TS	Test Suite
TT	Transition Tour

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 [1]. 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 highly powerful model for test derivation.

Several EFSM-based test derivation methods are presented considering the coverage of particular types of EFSM faults, such as single and double transfer faults, single 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, which is a set of deterministic EFSM mutants of the specification representing possible faulty implementations, a test suite of one or various 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), single output parameter faults (SOPFs) and single assignment faults (SAFs). Corresponding test suites are thus called STF, DTF, SOPF and SAF test suites. 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) [2] coverage criteria. An EP test suite covers each executable path of length up to 2 of the EFSM graph, and a 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 is time consuming and costly. It is well known that deriving a test suite that can detect several 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.

In this thesis, a comprehensive assessment of the fault coverage of the above mentioned test suites in addition to random test suites, hereafter named Rand, is carried out. The fault coverage of considered test suites is evaluated using six known EFSM specifications (or application examples) and using EFSM-based mutants of the specifications, where mutants are derived using a software tool that is implemented for this purpose. We consider single transfer fault mutants, double transfer fault mutants, single output parameter fault mutants and single assignment fault mutants of many types, namely, Assignment Insertion (AI), Assignment Deletion (AD), and Assignment Change Right Hand Side (CRHS) mutants. Ranking the test suites from best to worst is done based on two criteria; the first criterion is based solely on *fault coverage* (or *mutation score*), and the other one is based on the fault coverage and the length (called *coverage-length score*) of the test suites.

In nutshell, based on the conducted experiments, the best performing test suites for STF mutants, in terms of fault coverage, are the SI and STF followed by the PPST , DTF, EP, TT, Rand, PP, All Uses and then the SAF test suites. However, when considering the *coverage-length* score, the SI and STF test suites have comparable scores, and they again outperform the other test suites by approximately 14%. The PPST, EP, Rand, DTF, TT, SAF, PP and all uses test suites have comparable scores, but each of these test suites scores less than the SI and the STF test suites by approximately 14%.

For DTF mutants, the best performing test suites in terms of fault coverage are SI, STF, DTF and PPST followed by EP, TT, Rand, All Uses, PP and SAF. When considering the *coverage-length* score, PPST, EP and SI outperform the other test suites by approximately 7%.

For SOPF mutants, the best performing test suites in terms of coverage are All Uses, EP and TT followed by STF, DTF, SAF, Rand, PP and PPST. However, when considering the *coverage-length* score, the EP outperform the other test suites by approximately 13.7%.

Test suites coverage of All Assignments mutants is lower than all other mutants. The best performing test suite for All Assignments mutants is SAF followed by All Uses, TT, DTF, PPST, STF, Rand, EP, PP and SI. When considering the *coverage-length* score, SAF outperform the other test suites.

Test suites coverage of CRHS mutants is higher than AD mutants. In addition, test suites coverage of AD mutants is higher than AI mutants. Other than SAF test suites, coverage of AD and AI mutants of all other test suites is low. TT is the second best test suite, after All Uses, in covering AD, AI and CRHS mutants. PPST is ranked second in covering AI faults; whereas, it is ranked sixth and fifth in covering AD and CHRS faults. Other than SAF, All Uses, TT and PPST, all other test suites have significantly low coverage of assignment faults. When comparing the test suites based on the average of the obtained scores over all considered examples, STF outperform all the other test suites followed by TT, DTF, EP, All Uses, SAF, PPST, SI, Rand and PP.

A preliminary version of this thesis is published in [3]. This thesis extends that work in various ways, for example, more application examples and more types of EFSM test suites and faults are included in the experiments and assessment presented in this thesis.

This thesis is organized as follows. Chapter 2 includes preliminaries about EFSMs and EFSM-based test suites and types of EFSM faults. Chapter 3 includes an assessment of random test suites and an assessment and ranking of the considered test suites. Chapter 4 concludes the thesis and provides suggestions for further research.

Chapter 2: Preliminaries

In this chapter, the deterministic extended finite state machine EFSM model is introduced. Besides, several types of EFSM-based test suites are introduced, namely, single transfer fault (STF), double transfer fault (DTF), all uses of context variables, single assignment fault (SAF), transition tour (TT), state Identifiers (SITS), edge pair (EP), prime path (PP), prime path with side trip (PPST) and random test suites (Rand). At the end of this chapter, EFSM-based mutation testing mechanism is 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, notions related to EFSMs, mostly taken from [2], are illustrated, and how an EFSM operates through a working example is described.

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 [4] shown in Figure 2.1, 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 $\{0, \infty\}$. Therefore, the set of the context variables *number*, *d* and *counter* valuations equals $D_V = \{0, 1\} \times \{0, 1\} \times \{0, \infty\}$.

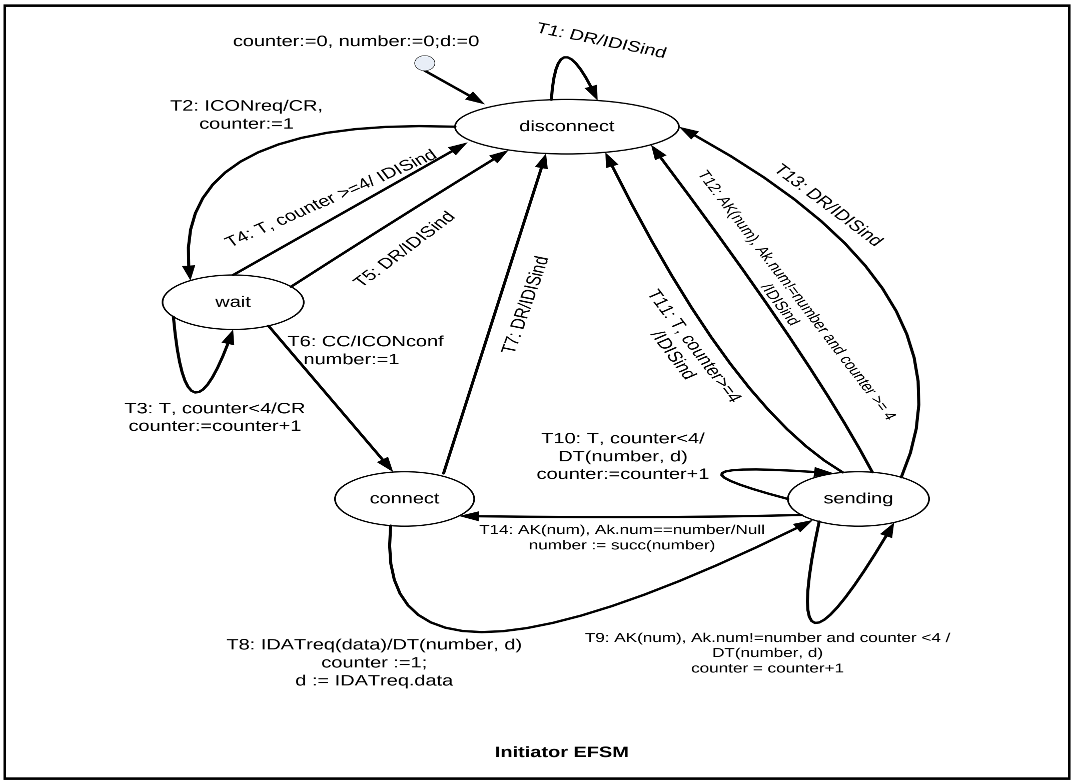

Figure 2.1 The Initiator EFSM [4]

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 is the *output parameter update* of t defined as $op : D_{Rx} \times D_V \rightarrow D_{Ry}$. It is noted 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 Figure 2.1 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. It has *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 $(s, \mathbf{v})$, where s is a state and $\mathbf{v}$ is a context. For example, configuration (*sending*, $(\mathbf{1, 1, 1})$) 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 $(\mathbf{1, 1, 1})$.

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 $\mathbf{v}' = up(\mathbf{p}_x, \mathbf{v})$. Hence, 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 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*, $(\mathbf{1, 1, 1})$ 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*, $(\mathbf{1, 1, 1})$ to configuration *sending*, $(\mathbf{1, 1, 2})$. 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, deterministic EFSM specifications are considered, 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 figure 2.1, (*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*.

The notation (s_1, $\mathbf{v}_1$) - α → (s_l, $\mathbf{v}_l$) is used to denote the fact that a trace from (s_1, $\mathbf{v}_1$) to the configuration (s_l, $\mathbf{v}_l$) exists, so that the input sequence of the trace is α. In this case, it is said that the input sequence α is *defined* at configuration (s_1, $\mathbf{v}_1$) and that the configuration (s_l, $\mathbf{v}_l$) is reached from (s_1, $\mathbf{v}_1$) by applying α. In this thesis, *executable or feasible* test cases are considered. 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, the types of EFSM mutants are described, namely, the transfer fault mutants with single or double transfer faults, single output parameter fault and single assignment fault with insertion, deletion and change right hand side.

- 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.
- Single Output Parameter Fault (SOPF): 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´≠ op*. Such *M´* is a *mutant of M with a single output parameter fault*. 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´≠ op*. Such *M´* is a *mutant of M with a single output parameter fault*.

- Single Assignment Faults (SAF): 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´≠ up*. Such *M´* is a *mutant of M with an assignment fault* (SAF). In this thesis, the following traditional types of single assignment faults and mutants with single assignment faults are considered:
  - Single Assignment Insertion (AI): 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 transitions (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 (AD): 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 Change Right-hand-side Fault (CRHS): 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 from that of *up*; that is, if a context variable of *M* in the RHS of *up* is added/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 Change Right-hand-side fault* (CRHS).

2.3 EFSM-Based Test Suites

In this section, the considered types of EFSM based test suites are described. Given two EFSMs *M* and *M´*, it is said that *M* and *M´* are *distinguishable* if their initial configurations are distinguishable by an input sequence (or a test case)α. In this case, it is said 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*, so 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 Double Transfer Fault (DTF) Test Suites

A DTF test suite is a test suite that covers double transfer faults of *M*, so that for each mutant of *M* with a double transfer faults distinguishable from *M*, the test suite has at least one test case that kills such a mutant.

2.3.3 Single Assignment Fault (SAF) Test Suites

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

2.3.4 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*.

2.3.5 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 a sequence $\alpha \in W_j$ that distinguishes states s_i and s_j exists for any other state s_i. 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*.

2.4 EFSM Flow-Graph Based Test Suites

Here, the EFSM flow-graph based test suites are described.

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 [5] or from a flow-graph representation of *M* as illustrated in [6].

2.5 EFSM Graph-Based Test Suites

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

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 less than two edges [7].

2.5.2 Prime Path (PP) Test Suite and 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 if 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*, so that every edge in *q* is also in *p* in the same order [7].

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 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 [8]. Mutation Testing is considered an expensive software testing technique. Research and studies have shown that Mutation Testing has a considerably high and strong rate in fault and error detection compared to other testing techniques [9].

EFSM-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 EFSM mutants. Test selection for a particular type of faults can be done with traditional EFSM based mutation testing techniques, by enumerating from the given EFSM specification all its EFSM mutants with that particular type of fault, and then derive a test suite for every mutant that is distinguishable from the given specification the test suite has a test case that detects (kills or distinguishes) the mutant. Two EFSMs are distinguishable if there exists an input sequence that when applied to the initial configurations of these machines, the output sequences produced by each machine in response to the input sequence are different. Methods for deriving distinguishing sequences for EFSMs are reported in [2]; analysis and lists of types of EFSM and other behavior models mutants are reported in various publications, such as [10] – [16]. This thesis consider six known EFSM specifications and analytically compare the effectiveness of several test selection criteria in covering EFSM mutants of these specifications with single and double transfer faults, single assignment faults and single output parameter faults.

Chapter 3: Assessing the Fault Coverage of EFSM Test Suites

In this chapter, a method is provided for assessing the fault coverage of the considered EFSM-based test suites. Experiments are conducted and the fault coverage of the test suites is compared in order to determine the best performing test suites in terms of fault coverage of EFSM mutants.

3.1 Considered EFSM Specification Examples

In experiments conducted in this study, five well-known communication protocols in addition to a CD player specification are considered [17]. Namely, the Trivial File Transfer Protocol (TFTP) [18], the Post Office Protocol V.3 (POP3) [19], the Initiator [4], the Responder [4], the SCP [20] and the CD player [17] specification EFSMs are considered.

3.2 Assessment of Fault Coverage of EFSM Test Suites

Given an EFSM specification *spec* and given EFSM test suites derived from *spec*, namely the STF, DTF, All-Uses, SAF, TT, SITS, EP, PP, PPST and Rand test suites. As described in [21], the coverage measure of a given test suite TS for a set of EFSM mutants derived using a test selection criterion is the mutation score computed as follows:

$$\textbf{\textit{Mutation Score}} = M_{killed} / (M_{total} - M_{inds}) \times 100 \qquad \textbf{(1)}$$

where M_{total} is the total number of all derived mutants satisfying the test selection criterion, M_{inds} is the number of generated mutants that are indistinguishable from the given EFSM specification, and M_{killed} is the number of mutants killed, i.e., distinguishable from the EFSM specification by the considered test suite.

In order to calculate an average mutation score of AI, AD and CRHS mutants for the same example, the following formula was used.

$$\textbf{\textit{Average Mutation Score}} = (M^{AI}_{killed} + M^{AD}_{killed} + M^{CHRS}_{killed}) / (M^{AI}_{total} + M^{AD}_{total} + M^{CHRS}_{total}) - (M^{AI}_{inds} + M^{AD}_{inds} + M^{CHRS}_{inds}) \times 100 \qquad \textbf{(2)}$$

where M^{AI}_{total}, M^{AD}_{total} and M^{CHRS}_{total} are the total number of all derived mutants satisfying the test selection criterion for AI, AD and CHRS mutants, respectively. M^{AI}_{inds}, M^{AD}_{inds} and M^{CHRS}_{inds} are the number of generated mutants that are indistinguishable from the given

EFSM specification, and M^{AI}_{killed}, M^{AD}_{killed} and M^{CHRS}_{killed} are the number of mutants killed, i.e., distinguishable from the EFSM specification by the considered test suite.

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

Coverage-Length Score = (Fault-Coverage-of-TS) W_{FC} + (length-of-TS) W_{Length} **(3)**

W_{FC} and W_{Length} are weights where $W_{FC} + W_{Length} = 1$. Note that scores are determined based on the following combinations of W_{FC} and W_{Length}, (.95, 0.05), (0.9, 0.1), (0.85, 0.15). Only the scores obtained using the combination (0.85, 0.15) were used as it was found that this combination produces a ranking that is highly close to the other combinations, and it provides a clear idea of how ranking changes considering these combinations in comparison with the ranking obtained using fault coverage scores. The following subsection describes the assessment method in more detail.

3.3 Assessment Method in More Detail

The method has three steps as shown in Figure 3.1. In **Step 1**, for all considered EFSM specifications, all EFSM mutants of *M* with STF, DTF, AI, AD, CRHS, All Assignments Faults and SOPF are derived using a software tool that is implemented for this purpose. The tool also determines and eliminates all generated mutants that are indistinguishable from the considered EFSM specification. In **Step 2**, STF, DTF and SAF test suites are automatically derived from the mutants as follows. For each considered EFSM mutant, a test that kills the mutant from the specification *M* is derived and added to the test suite if needed , i.e., if the test suite does not already have a test case that kills the mutant. It is noted that the derived STF, DTF and SAF test suites are of optimal or near optimal length as shortest length distinguishing sequences are derived. Transition tour (TT), SITS, is derived manually by hand. In addition, the EP, PP and PPST are derived with the help of the *graph coverage* web application tool [7]. Moreover, for every specification, a corresponding flow-graph representation annotated with definitions and uses of variables is constructed, and then corresponding all uses test suite (set of paths) is derived from the obtained flow-graph exactly as described in previous related research work [6]. **Step 3**, in order to determine the mutation scores of a considered test suite, such as STF, DTF, All Uses, SAF, TT, SITS, EP, PP and PPST against all the considered mutants, the test cases of the test suite against these mutants are run, and the mutants killed by the test cases of the test suite are detected, and then the corresponding mutation score is computed as given in (1).

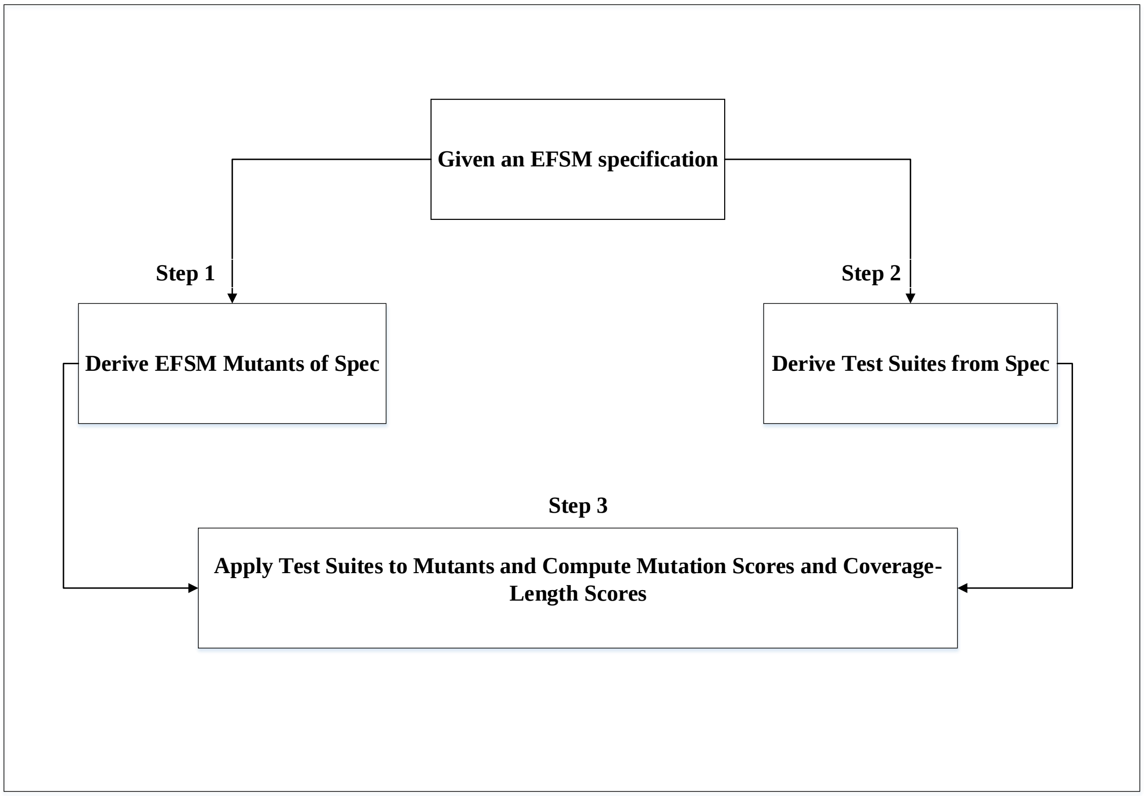

Figure 3.1 Assessment Methodology

3.4 Fault Coverage of Random Test Suites

In this section, the fault coverage of random test suites is studied and the length of the best test suite for each considered EFSM machine is determined. In particular, for each of the considered EFSM examples, varying length test suites are considered, and for each considered length, five random test suites are derived and applied to all considered mutants *M* of the EFSM specification. Corresponding fault coverage is determined and the length of random test suites is increased 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 5% or decrease by less than 5% from the average mutation score of the random test suites with more length. The least length that satisfies the above criterion is selected as the best length of a random test suite for the considered examples. In the following chapter, the fault coverage of (the best) random test suites with other EFSM-based test suites is assessed and compared.

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

Figure 3.2 depicts the length of random test suites studied for one implementation of the TFTP EFSM. It 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.

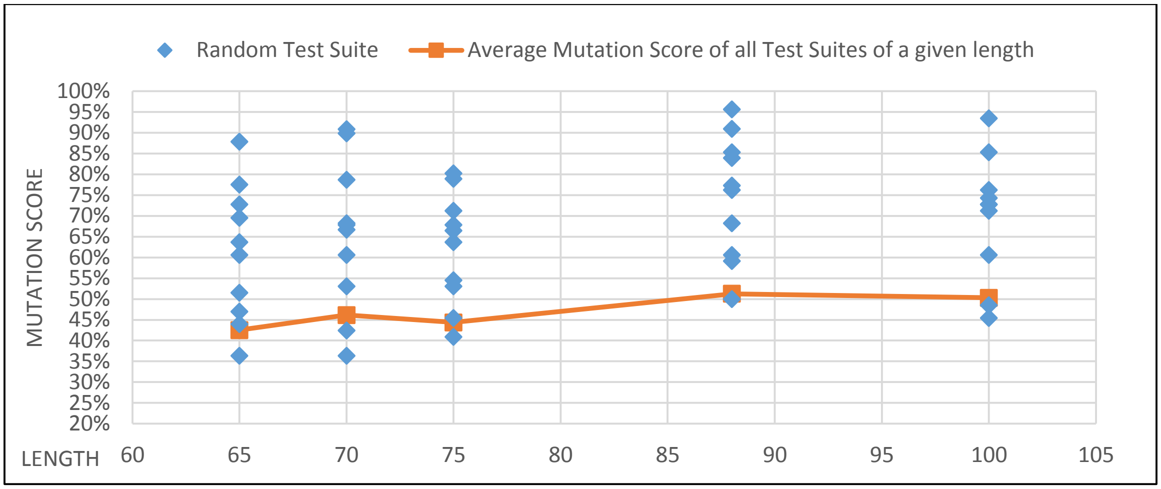

Figure 3.2 Random Test Suites of TFTP

According to the stopping criterion described earlier in this section, the best random test suite of the TFTP is that with the length of 70.

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

Figure 3.3 depicts the length of random test suites studied for one implementation of the CD Player EFSM. It 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.

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

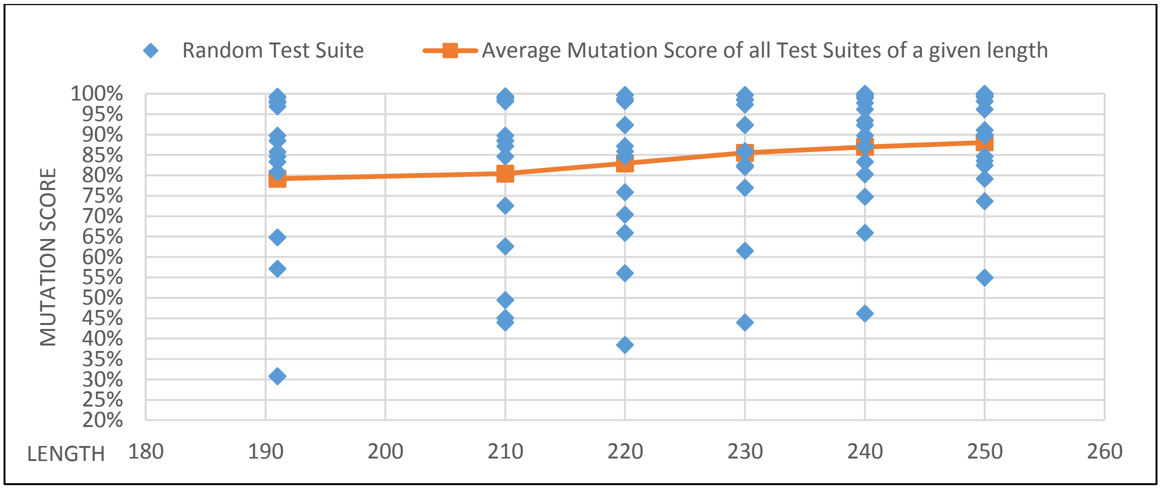

Figure 3.3 Random Test Suites of CD

c) POP 3 Example: Determining the Best Random Test Suite Length

Figure 3.4 depicts the length of random test suites studied for one implementation of the POP3 EFSM. It 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.

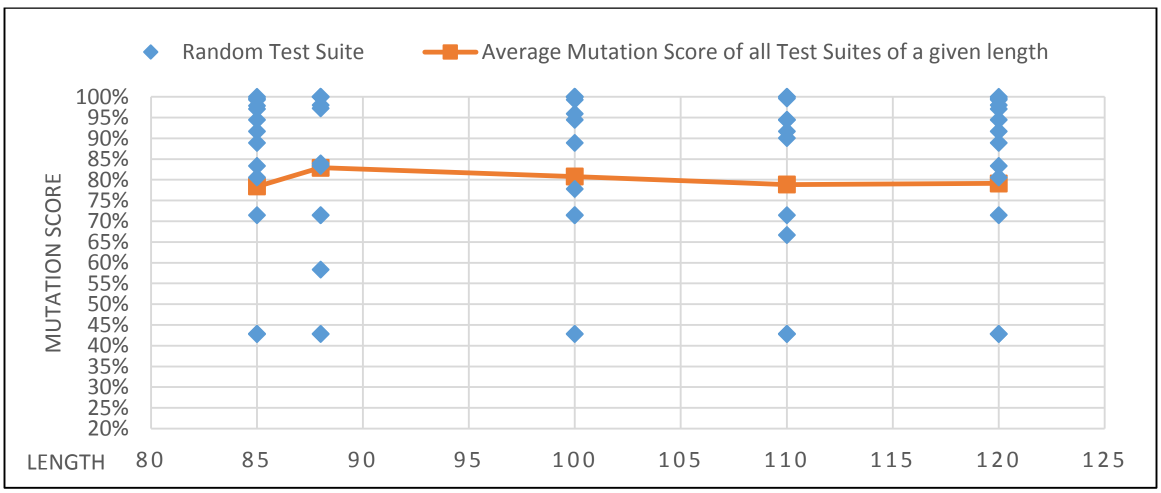

Figure 3.4 Random Test Suites of POP 3

According to the stopping criterion described earlier, the best random test suite of the POP 3 is that with the length of 85.

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

Figure 3.5 depicts the length of random test suites studied for one implementation of the Initiator EFSM. It 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.

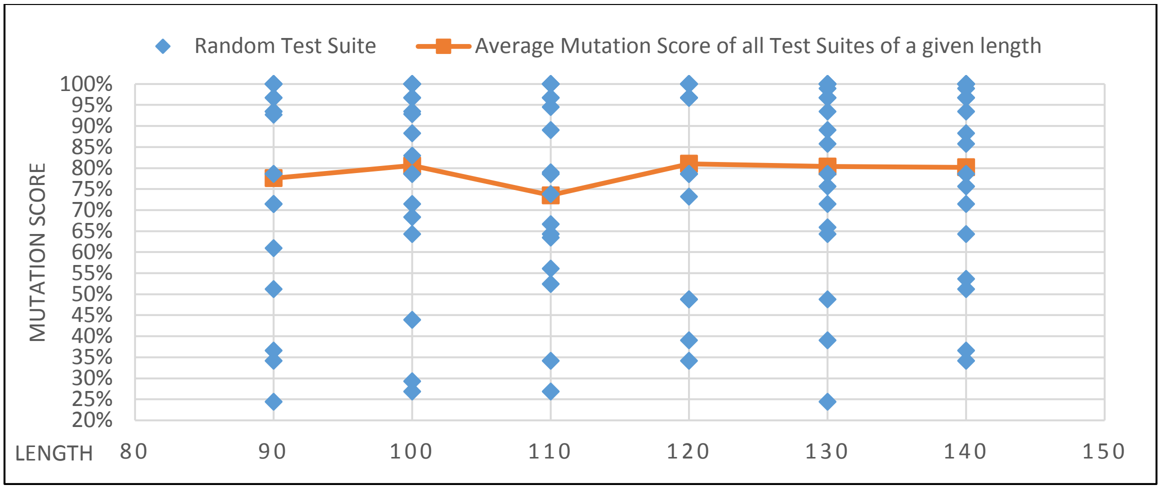

Figure 3.5 Random Test Suites of Initiator

According to the stopping criterion described earlier, the best random test suite of the Initiator is that with the length of 90.

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

Figure 3.6 depicts the length of random test suites studied for one implementation of the Responder EFSM. It 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.

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

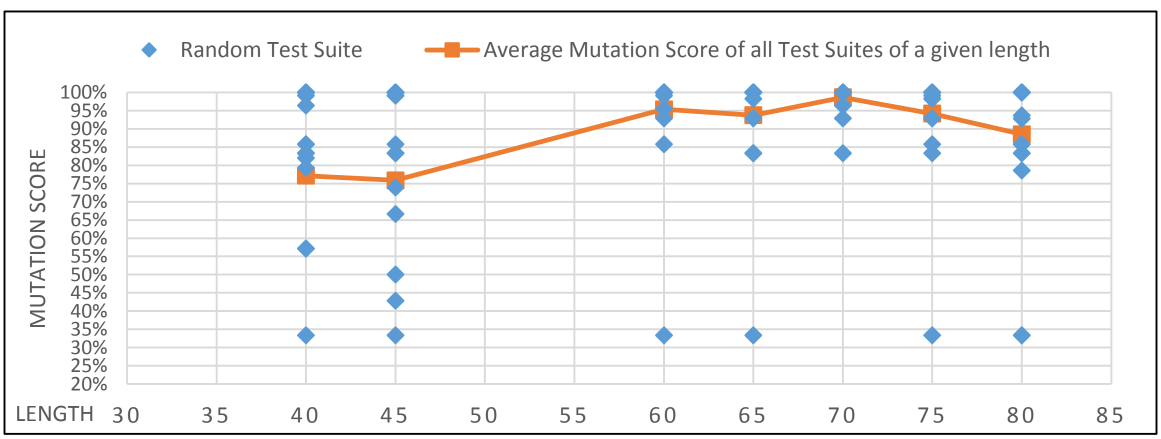

Figure 3.6 Random Test Suites of Responder

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

Figure 3.7 depicts the length of random test suites studied for one implementation of the SCP EFSM. It 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.

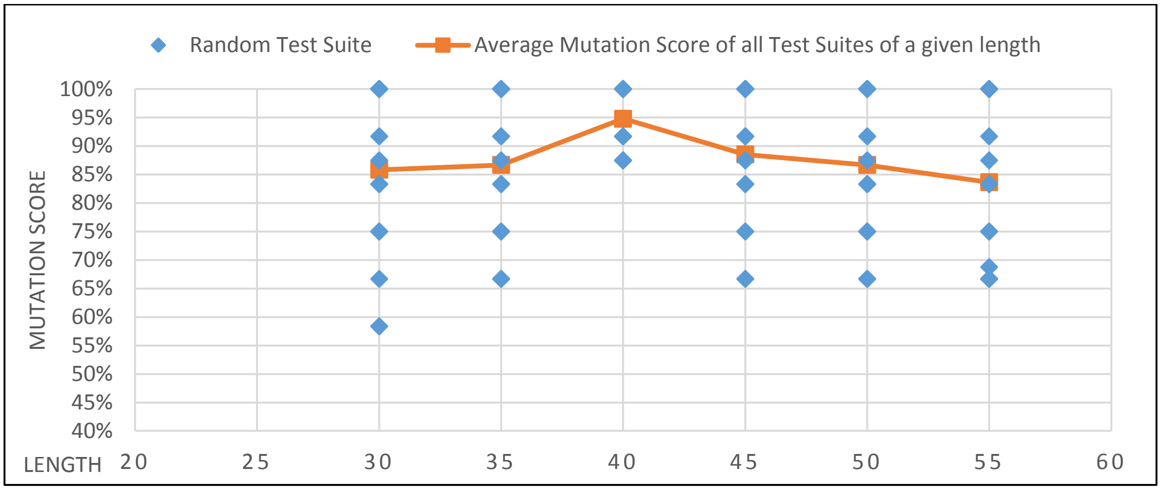

Figure 3.7 Random Test Suites of SCP

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

3.5 Experiment Evaluation

In the following sections, the obtained results of mutation scores (using (1), (2) and (3) described earlier in section 3.2) are presented, discussed, ranked and analyzed as shown in the following tables and figures.

3.5.1 Assessment of Fault Coverage of Single Transfer Fault Mutants

This section includes the mutation scores (fault coverage) of the considered test suites with respect to STF mutants for each considered EFSM specifications. In addition, it includes the average mutation score of the considered test suites with respect to STF mutants. It is noted that there is no SITS test suite for the Initiator, Responder and SCP examples as there are no state identifiers for the corresponding states.

a) TFTP Test Suites Fault Coverage Assessment

Figure 3.8 includes the mutation score and length of each considered test suite for the TFTP example.

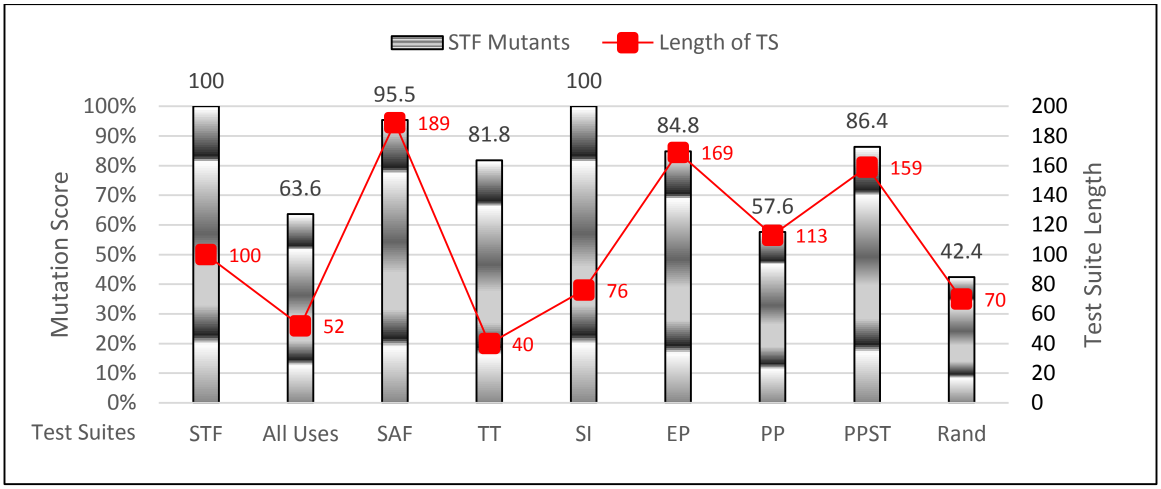

Figure 3.8 TS Coverage of STF Mutants in TFTP Example

b) CD Test Suites Fault Coverage Assessment

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

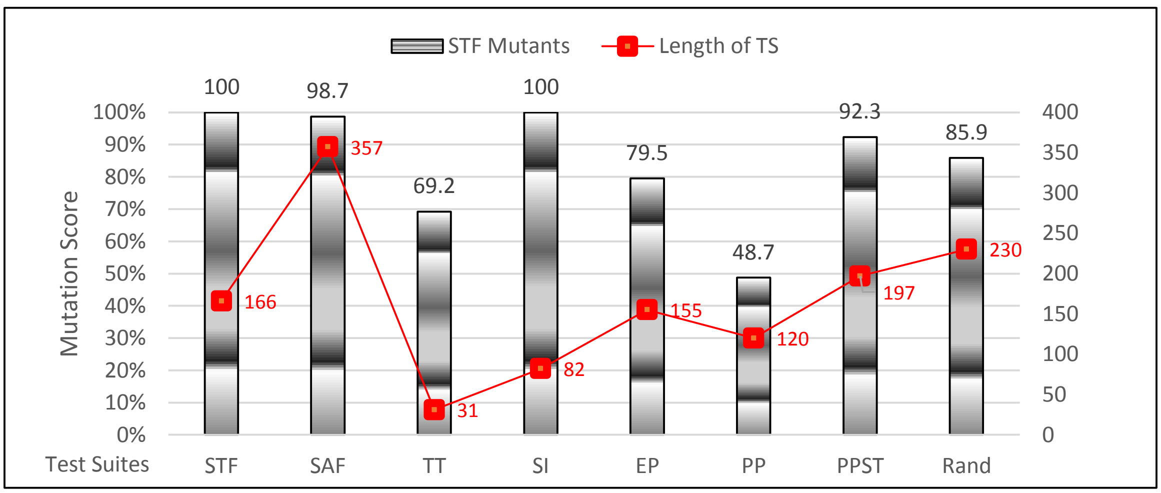

Figure 3.9 TS Coverage of STF Mutants in CD Example

c) POP 3 Test Suites Fault Coverage Assessment

Figure 3.10 includes the mutation scores and length of each considered test suite for the POP 3 example.

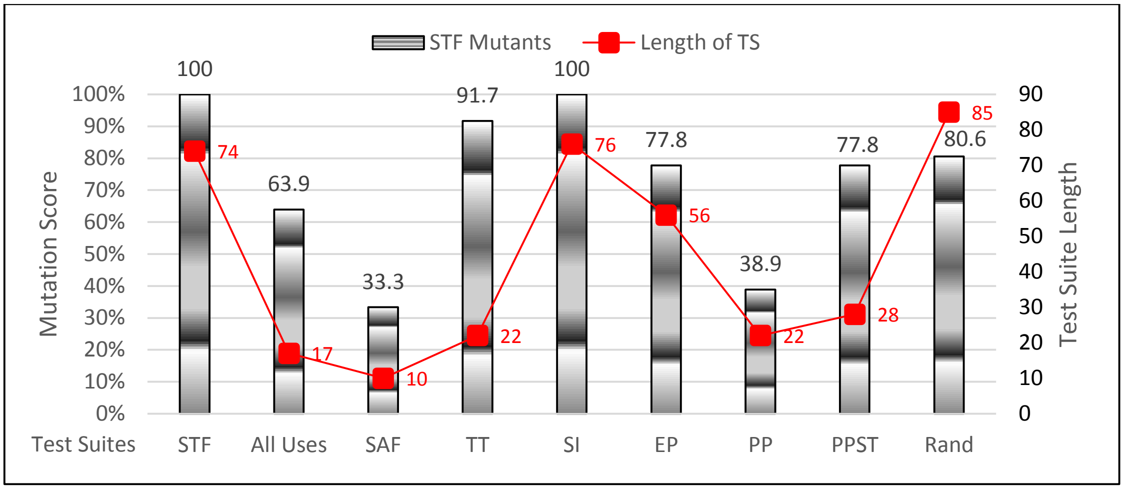

Figure 3.10 TS Coverage of STF Mutants in POP 3 Example

d) Initiator Test Suites Fault Coverage Assessment

Figure 3.11 includes the mutation scores and length of each considered test suite for the Initiator example.

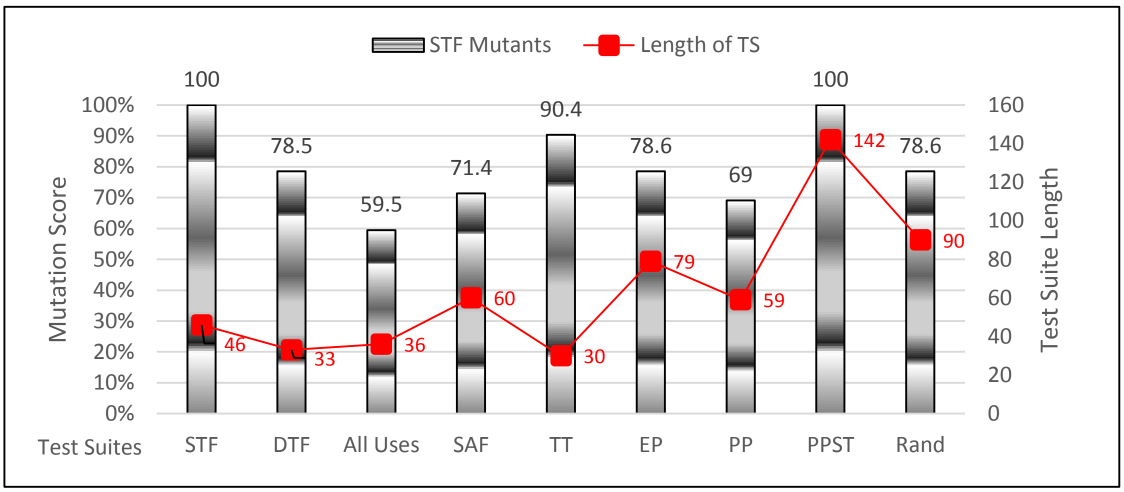

Figure 3.11 TS Coverage STF Mutants in Initiator Example

e) Responder Test Suites Fault Coverage Assessment

Figure 3.12 includes the mutation scores and length of each considered test suite for the Responder example.

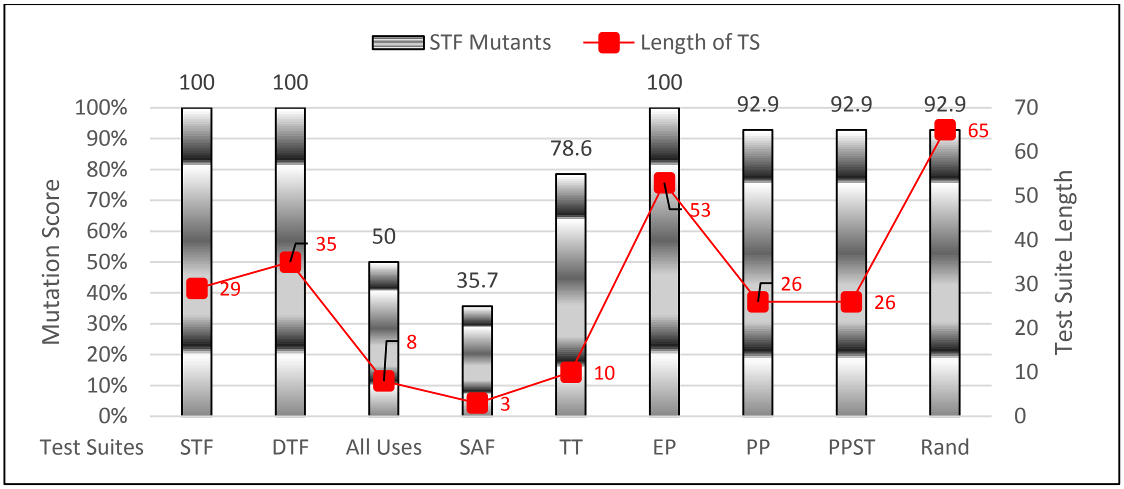

Figure 3.12 TS Coverage of STF Mutants in Responder Example

f) SCP Test Suite Fault Coverage Assessment

Figure 3.13 includes the mutation scores and length of each considered test suite for the SCP example.

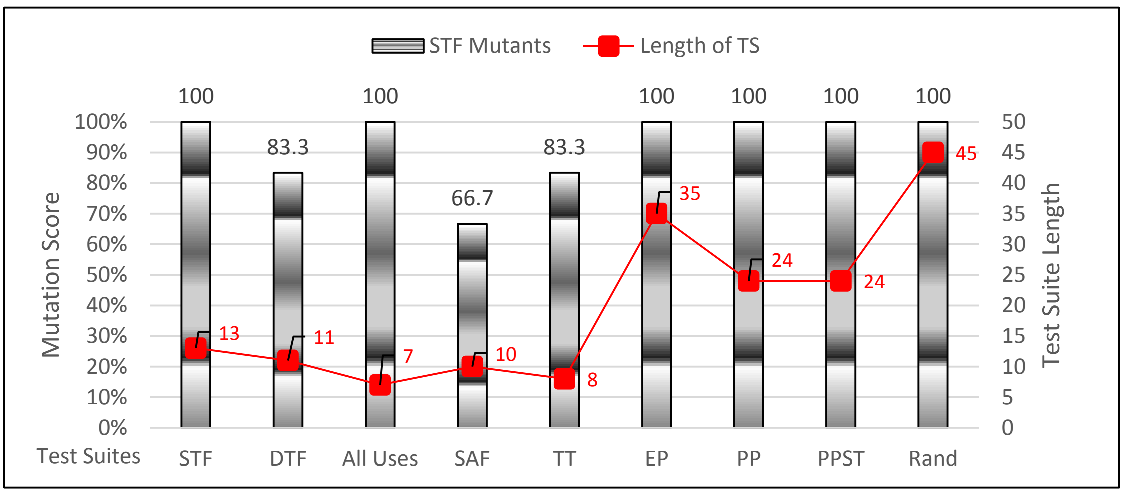

Figure 3.13 TS Coverage of STF Mutants in SCP Example

g) Summary of Single Transfer Fault Mutants for all Considered Examples

Figure 3.14 includes the average mutation scores of STF mutants and average length of each considered test suite for all the above considered examples.

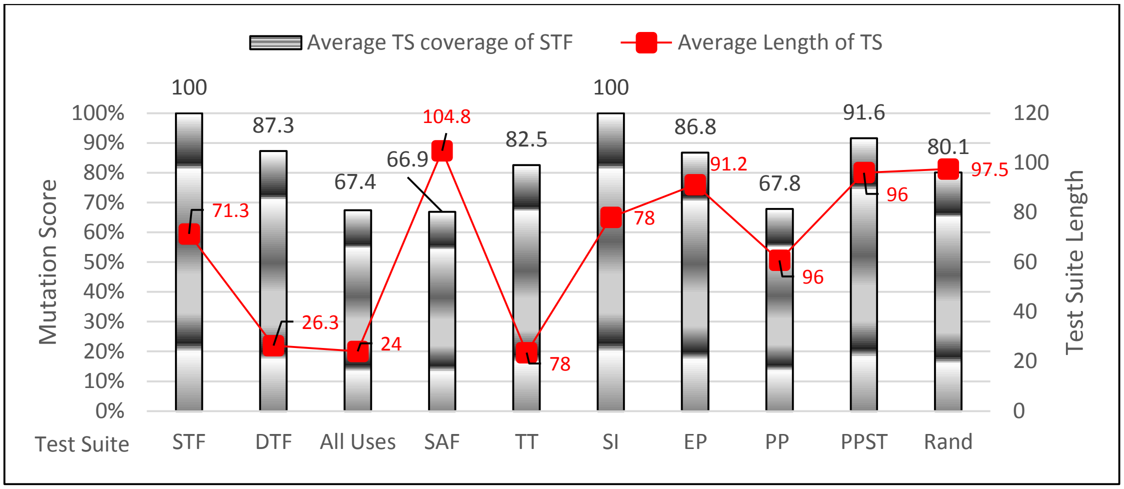

Figure 3.14 Average TS Coverage of STF Mutants for all Examples

3.5.2 Assessment of Fault Coverage of Double Transfer Fault Mutants

This section includes the mutation scores (fault coverage) of the considered test suites with respect to DTF mutants for each considered EFSM specifications. It also includes the average mutation score of the considered test suites with respect to DTF mutants. It is noted that there is no SITS test suite for the Initiator, Responder and SCP examples as there are no state identifiers for the corresponding states.

a) TFTP Test Suites Fault Coverage Assessment

Figure 3.15 includes the mutation scores of DTF mutants and length of each considered test suite for the TFTP example.

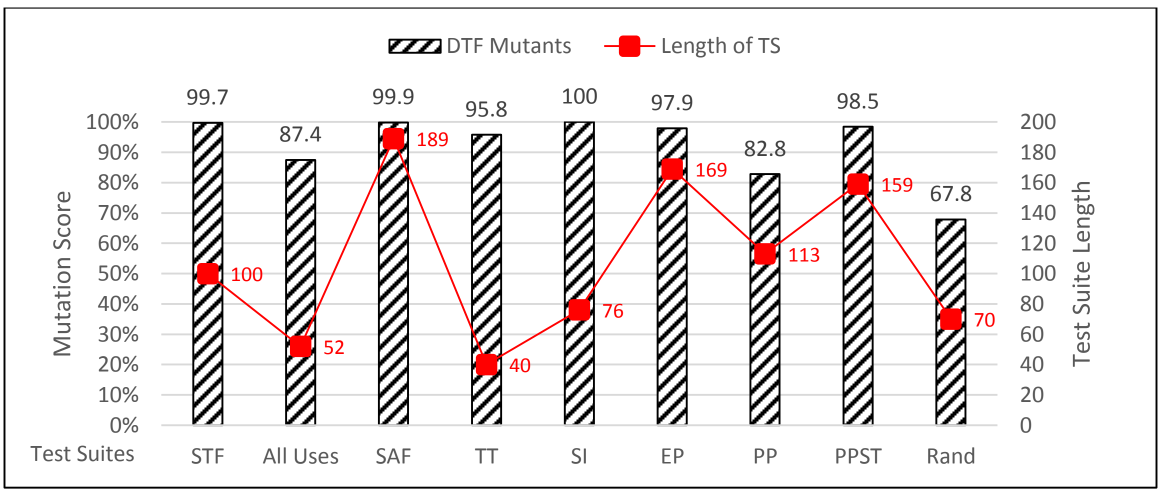

Figure 3.15 TS Coverage of DTF Mutants in TFTP Example

b) CD Player Test Suites Fault Coverage Assessment

Figure 3.16 includes the mutation scores of DTF mutants and length of each considered test suite for the CD Player example.

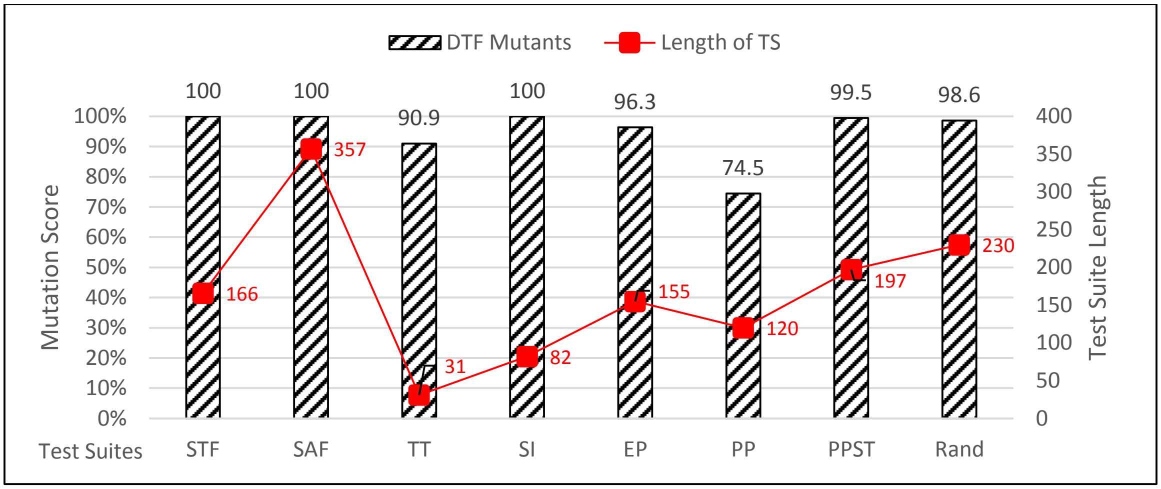

Figure 3.16 TS Coverage of DTF Mutants in CD Example

c) POP 3 Test Suites Fault Coverage Assessment

Figure 3.17 includes the mutation scores of DTF mutants and length of each considered test suite for the POP 3 example.

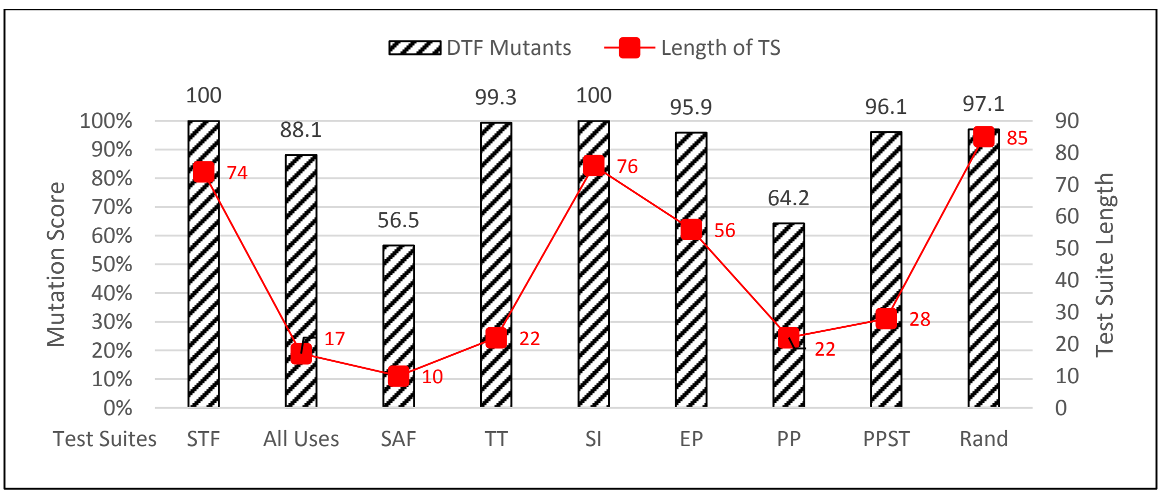

Figure 3.17 TS Coverage of DTF Mutants in POP 3 Example

d) Initiator Test Suites Fault Coverage Assessment

Figure 3.18 includes the mutation scores of DTF mutants and length of each considered test suite for the Initiator example.

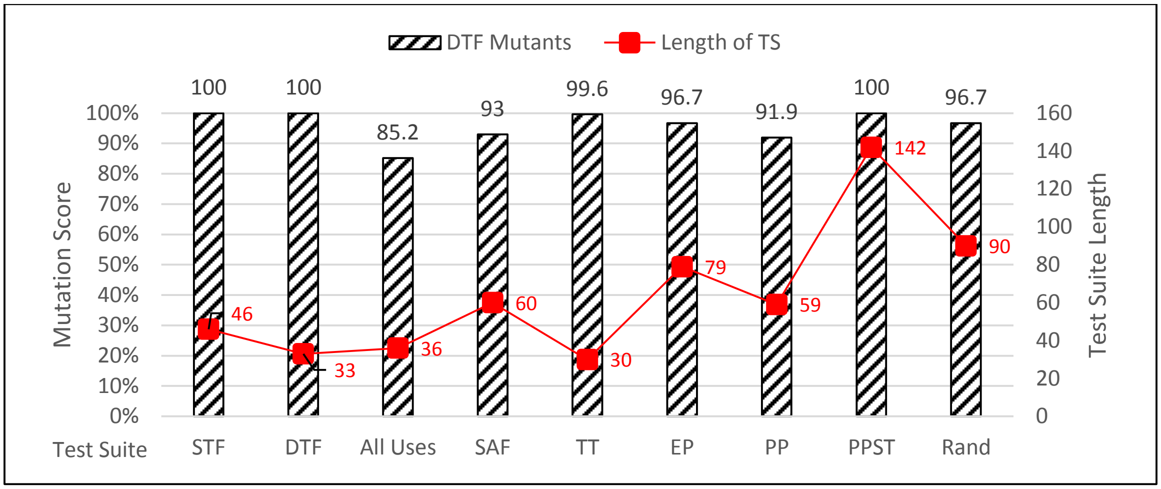

Figure 3.18 TS Coverage of DTF Mutants in Initiator Example

e) Responder Test Suites Fault Coverage Assessment

Figure 3.19 includes the mutation scores of DTF mutants and length of each considered test suite for the Responder example.

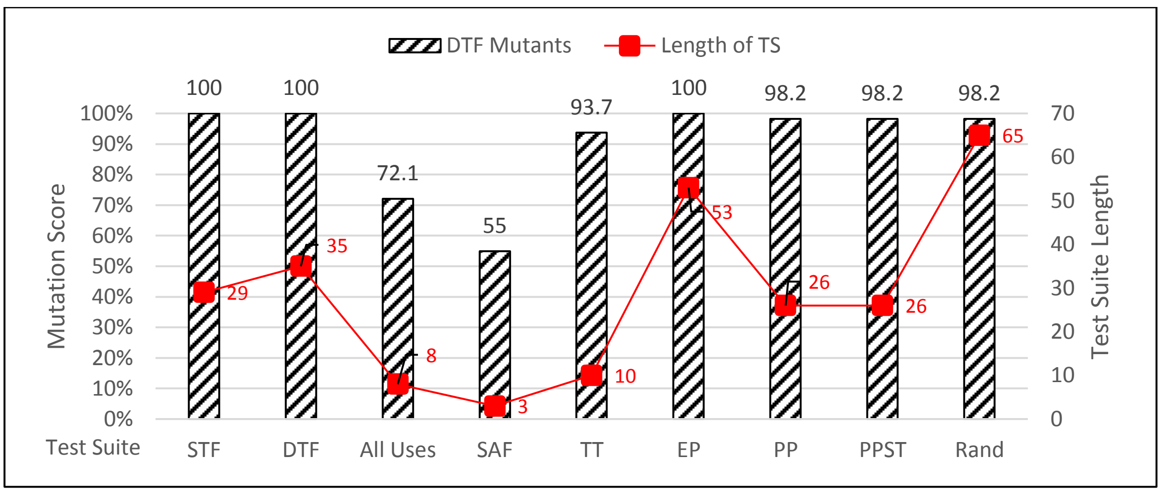

Figure 3.19 TS Coverage of DTF Mutants in Responder Example

f) SCP Test Suites Fault Coverage Assessment

Figure 3.20 includes the mutation scores of DTF mutants and length of each considered test suite for the SCP example.

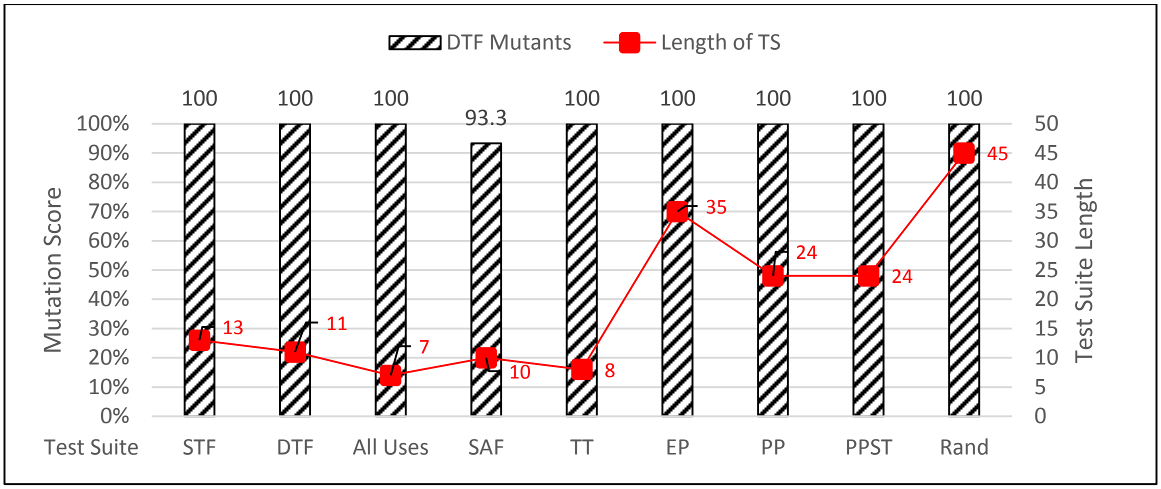

Figure 3.20 TS Coverage of DTF Mutants in SCP Example

g) Summary of Double Transfer Fault Mutants for All Considered Examples

Figure 3.21 includes the average mutation scores of DTF mutants and average length of each considered test suite for all the above considered examples.

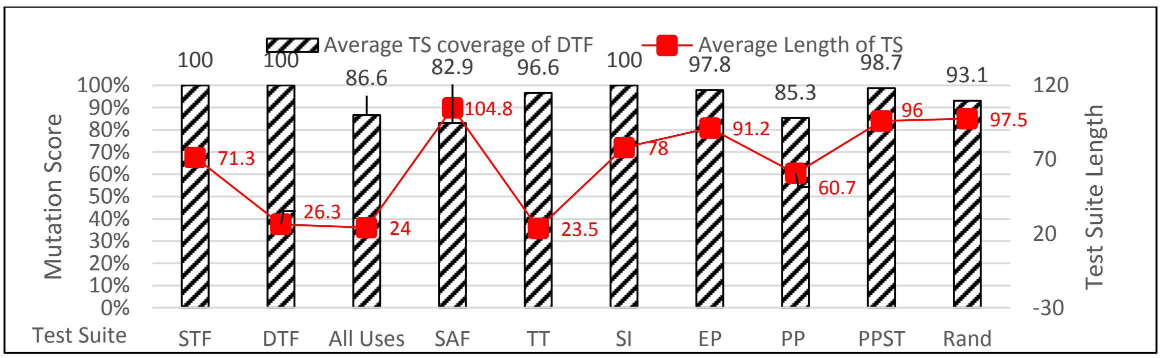

Figure 3.21 Average TS Coverage of DTF Mutants for All Examples

3.5.3 Assessment of Fault Coverage of Single Output Parameter Faults Mutants

This section includes the mutation scores (fault coverage) of the considered test suites of SOPF mutants for each considered EFSM specifications. It also includes the average mutation score of the considered test suites with respect to SOPF mutants. It is noted that there are no Output Parameters in TFTP, CD Player and POP 3 examples. Furthermore, it is noted that there is no SITS test suite for the Initiator, Responder and SCP examples as there are no state identifiers for the corresponding states.

a) Initiator Test Suites Fault Coverage Assessment

Figure 3.22 includes the mutation scores of SOPF mutants and length of each considered test suite for the Initiator example.

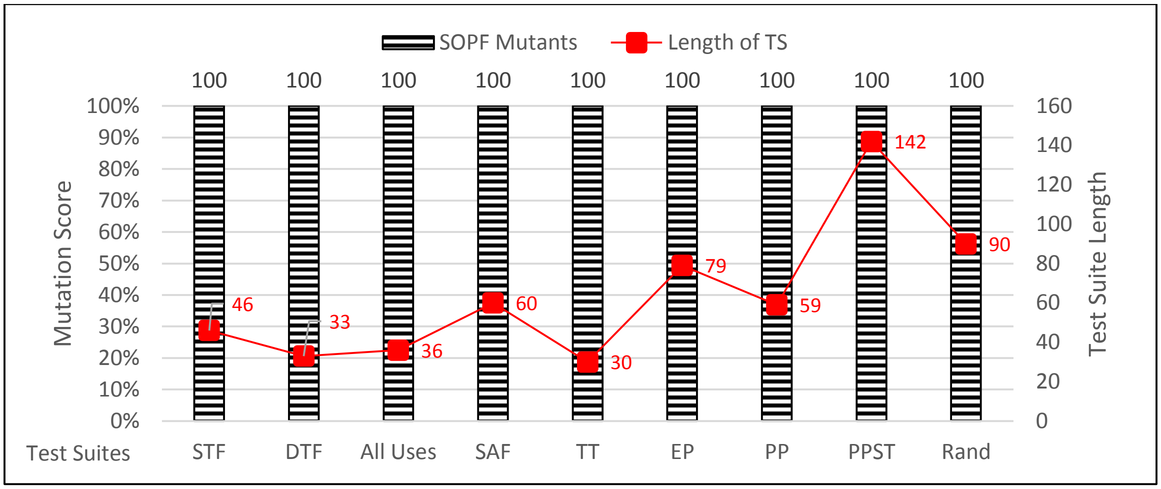

Figure 3.22 TS Coverage of SOPF Mutants in Initiator Example

b) Responder Test Suites Fault Coverage Assessment

Figure 3.23 includes the mutation scores of SOPF mutants and length of each considered test suite for the Responder example.

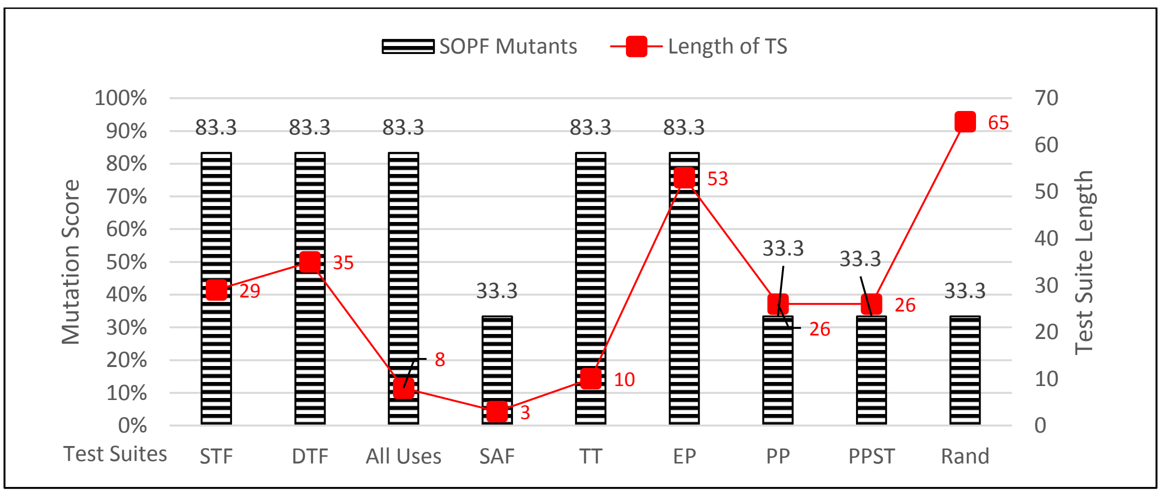

Figure 3.23 TS Coverage of SOPF Mutants in Responder Example

c) SCP Test Suites Fault Coverage Assessment

Figure 3.24 below includes the mutation scores of SOPF mutants and length of each considered test suite for the SCP example.

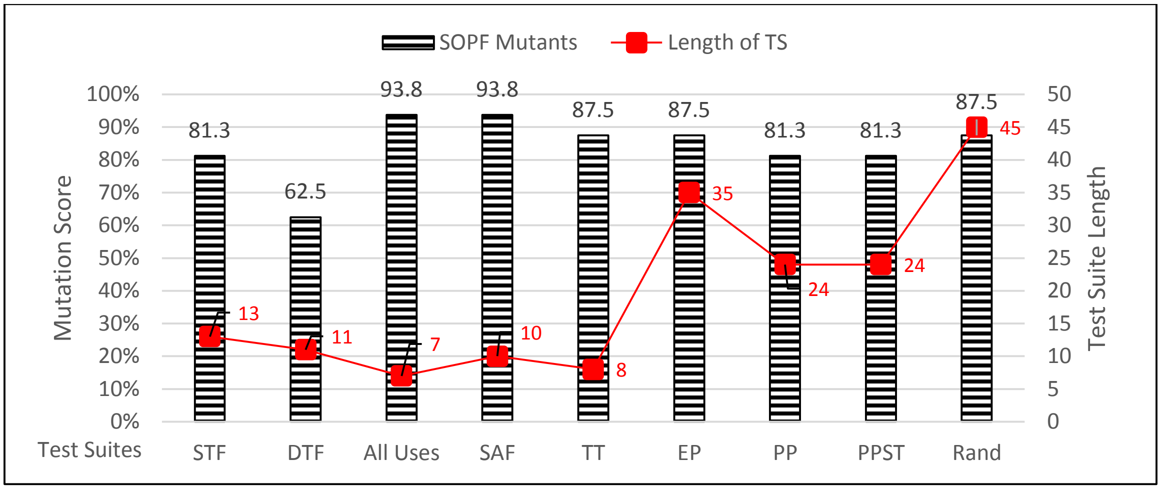

Figure 3.24 TS Coverage of SOPF Mutants in SCP Example

d) Summary of Single Output Parameter Faults Mutants for All Considered Examples

Figure 3.25 includes the average mutation scores of SOPF mutants and average length of each considered test suite for all the above considered examples.

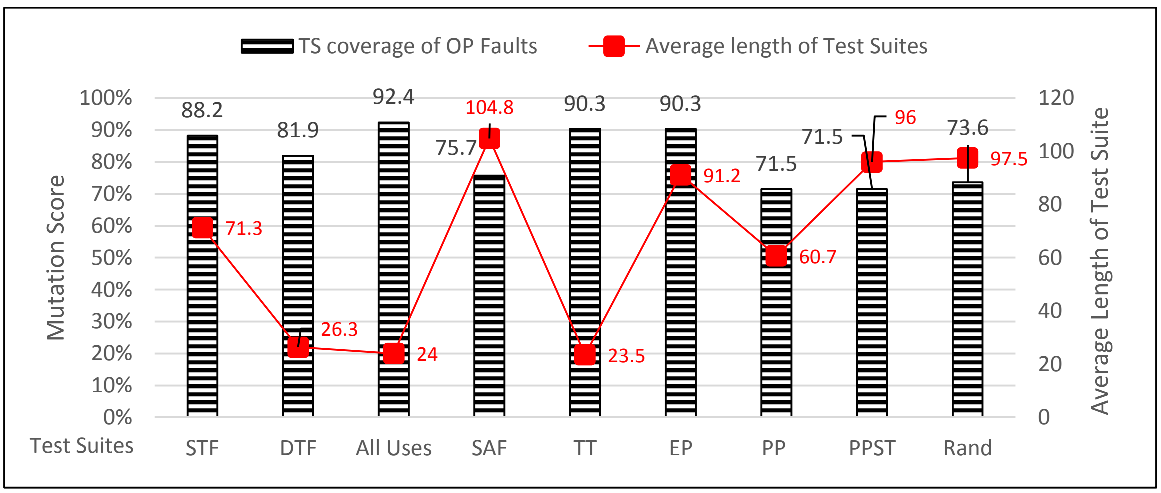

Figure 3.25 Average TS Coverage of SOPF Mutants for All Examples

3.5.4 Assessment of Fault Coverage of AI, AD, CRHS and All Assignments Faults Mutants

This section includes the mutation scores (fault coverage) of the considered test suites with respect to AI, AD, CRHS and All Assignments mutants for each considered EFSM specifications. Mutation scores of All Assignments mutants are the average mutation scores of AI, AD and CRHS mutants, which were calculated using (2) as described earlier, for all considered test suites. It is noted that there is no SITS test suite for the Initiator, Responder and SCP examples as there are no state identifiers for the corresponding states.

a) TFTP Test Suites Fault Coverage Assessment

Figure 3.26 includes the mutation scores of AI, AD, CHRS and All Assignments mutants and length of each considered test suite for the TFTP example.

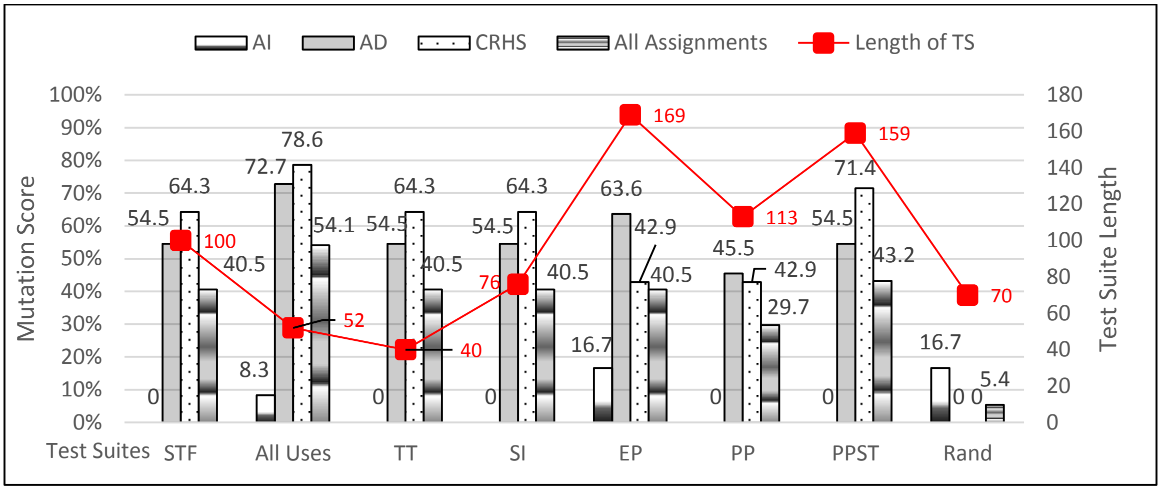

Figure 3.26 TS Coverage of AI, AD, CHRS and All Assignments Mutants in TFTP Example

b) CD Player Test Suites Fault Coverage Assessment

Figure 3.27 includes the mutation scores of AI, AD, CHRS and All Assignments mutants and length of each considered test suite for the CD Player example.

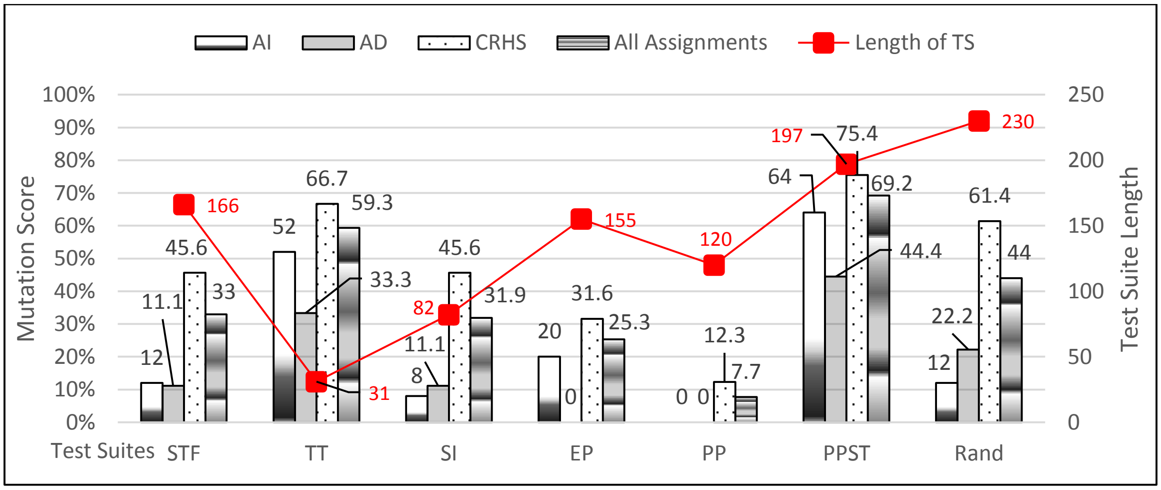

Figure 3.27 TS Coverage of AI, AD, CHRS and All Assignments Mutants in CD Example

c) POP 3 Test Suites Fault Coverage Assessment

Figure 3.28 includes the mutation scores of AI, AD, CHRS and All Assignments mutants and length of each considered test suite for the POP 3 example.

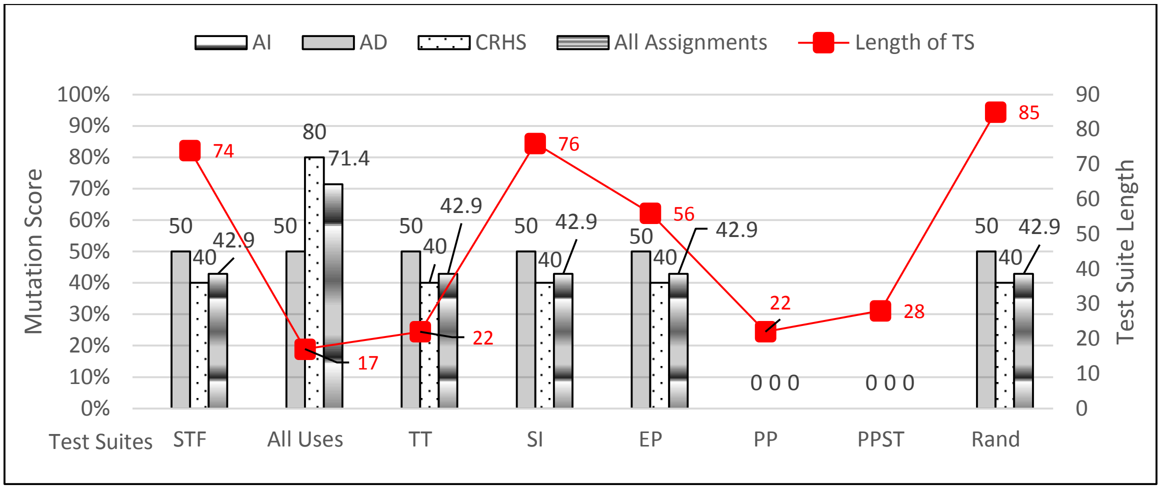

Figure 3.28 TS Coverage of AI, AD, CHRS and All Assignments Mutants in POP 3 Example

d) Initiator Test Suites Fault Coverage Assessment

Figure 3.29 includes the mutation scores of AI, AD, CHRS and All Assignments mutants and length of each considered test suite for the Initiator example.

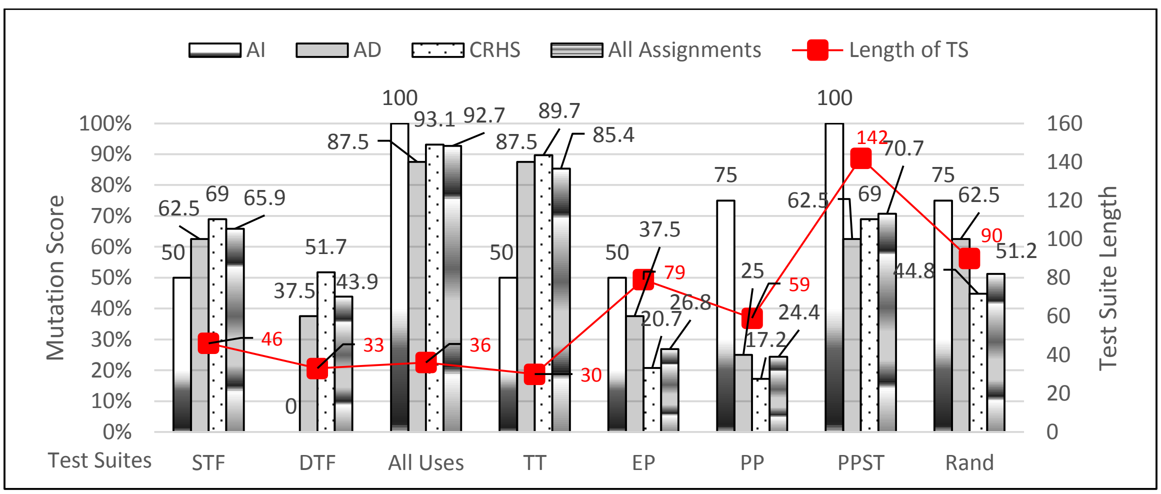

Figure 3.29 TS Coverage of AI, AD, CHRS and All Assignments Mutants in Initiator Example

e) Responder Test Suites Fault Coverage Assessment

Figure 3.30 includes the mutation scores of AI, AD, CHRS and All Assignments mutants and length of each considered test suite for the Responder example.

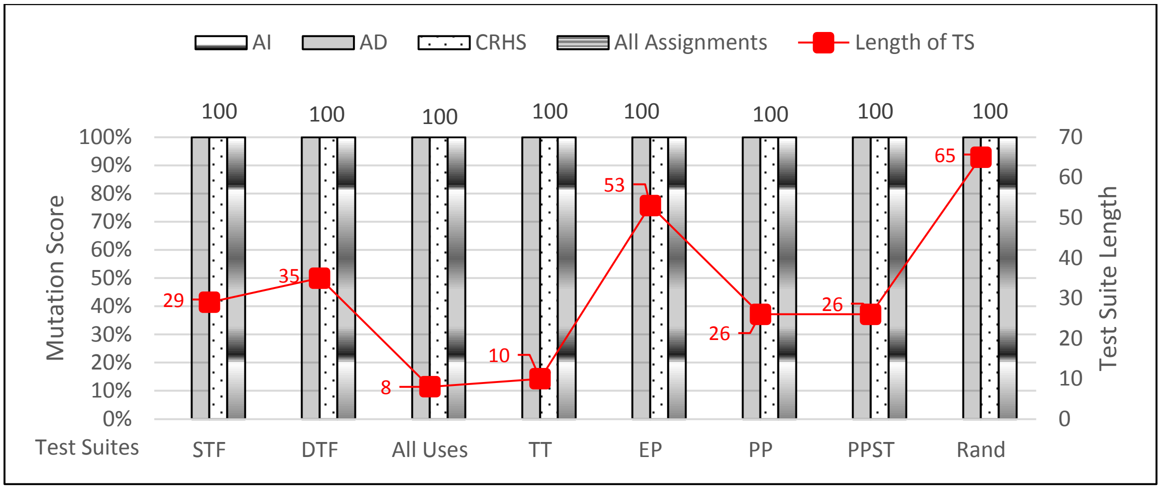

Figure 3.30 TS Coverage of AI, AD, CHRS and All Assignments Mutants in Responder Example

f) SCP Test Suites Fault Coverage Assessment

Figure 3.31 includes the mutation scores of AI, AD, CHRS and All Assignments mutants and length of each considered test suite for the Responder example.

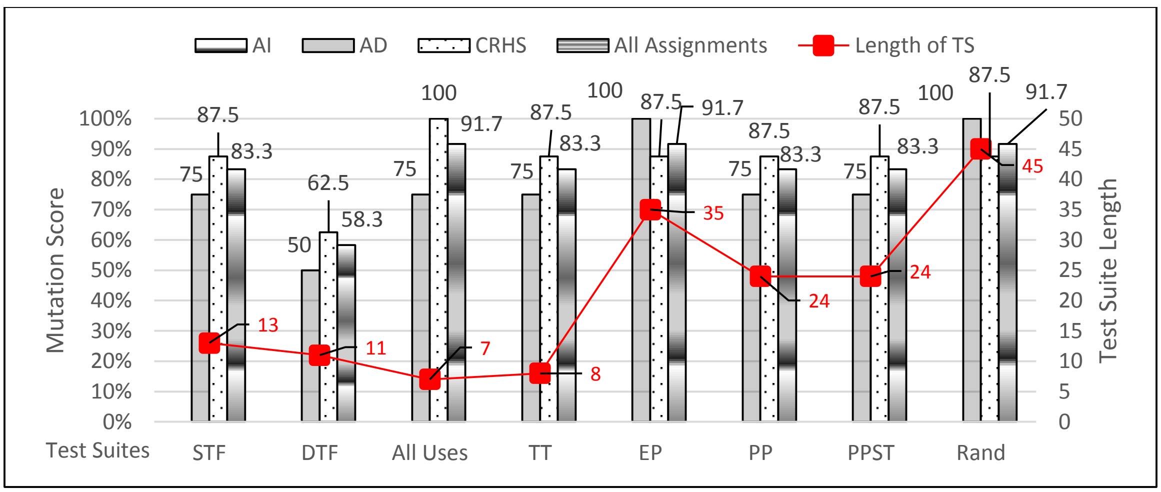

Figure 3.31 TS Coverage of AI, AD, CHRS and All Assignments Mutants in SCP Example

g) Summary of AI, AD, CRHS and All Assignments mutants for All Considered Examples

Figure 3.32 includes the average mutation scores of AI, AD, CRHS and All Assignments mutants and average length of each considered test suite for all the above considered examples.

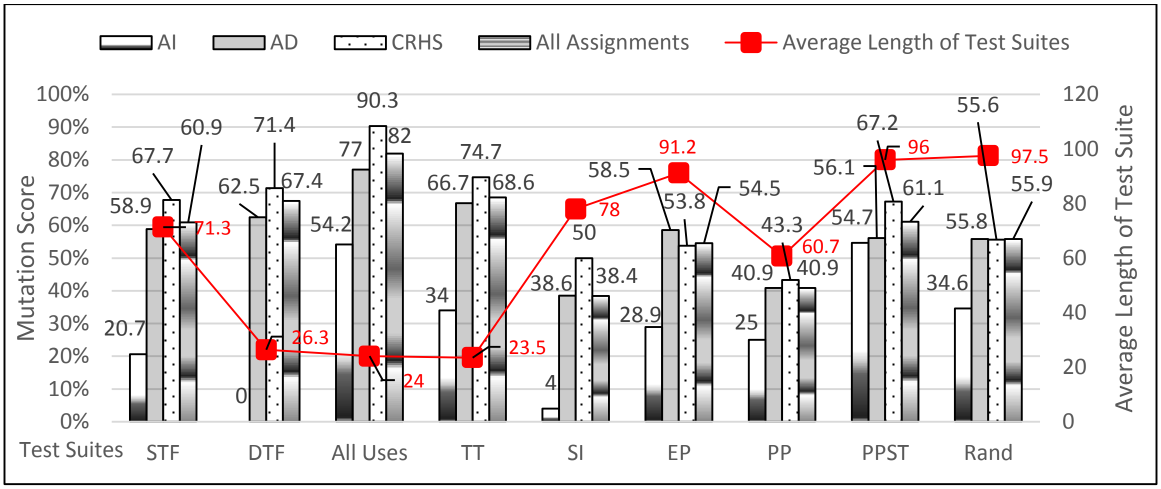

Figure 3.32 Average TS Coverage of AI, AD, CRHS and All Assignments mutants for all Considered Examples

3.5.5 Assessment of Fault Coverage of All Assignments mutants

This section includes the mutation scores (fault coverage) of the considered test suites with respect to All Assignments mutants for each considered EFSM specifications. It also includes the average mutation score of the considered test suites with respect to All Assignments mutants. It is noted that there is no SITS test suite for the Initiator, Responder and SCP examples as there are no state identifiers for the corresponding states.

a) TFTP Test Suites Fault Coverage Assessment

Figure 3.33 includes the mutation scores of All Assignments mutants and length of each considered test suite for the TFTP example.

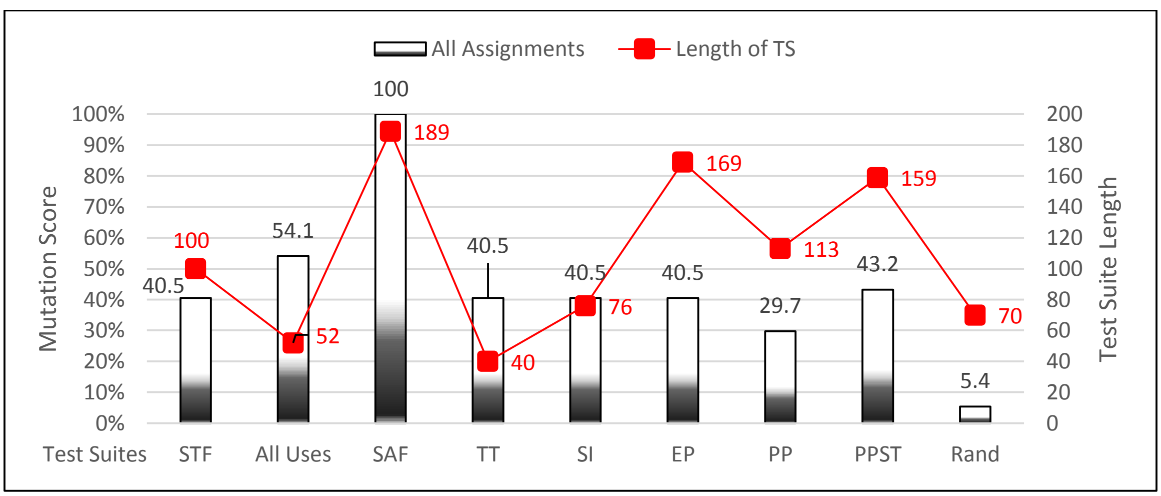

Figure 3.33 TS Coverage of All Assignments Mutants in TFTP Example

b) CD Player Test Suites Fault Coverage Assessment

Figure 3.34 includes the mutation scores of All Assignments mutants and length of each considered test suite for the CD Player example.

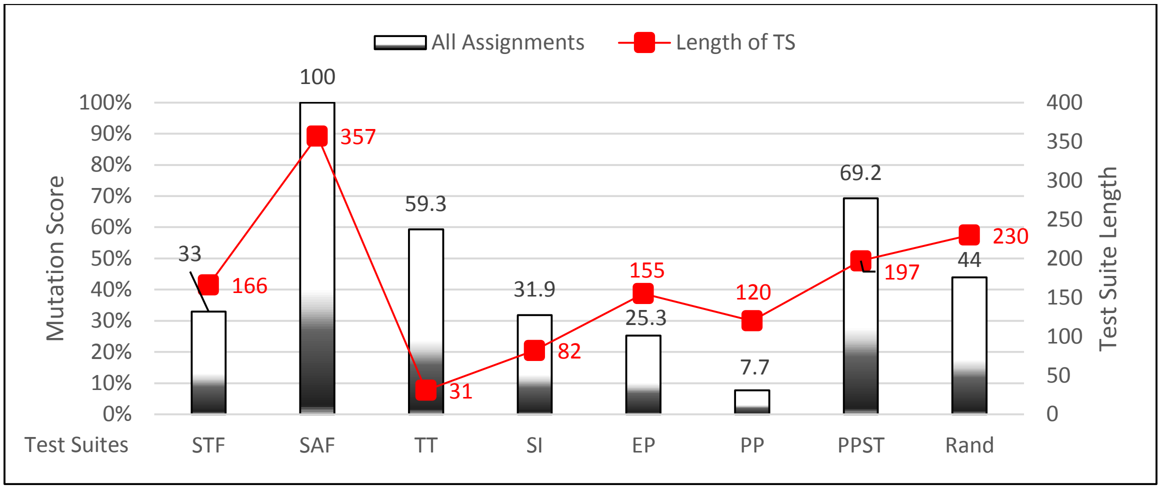

Figure 3.34 TS Coverage of All Assignments Mutants in CD Example

c) POP 3 Test Suites Fault Coverage Assessment

Figure 3.35 includes the mutation scores of All Assignments mutants and length of each considered test suite for the POP 3 example.

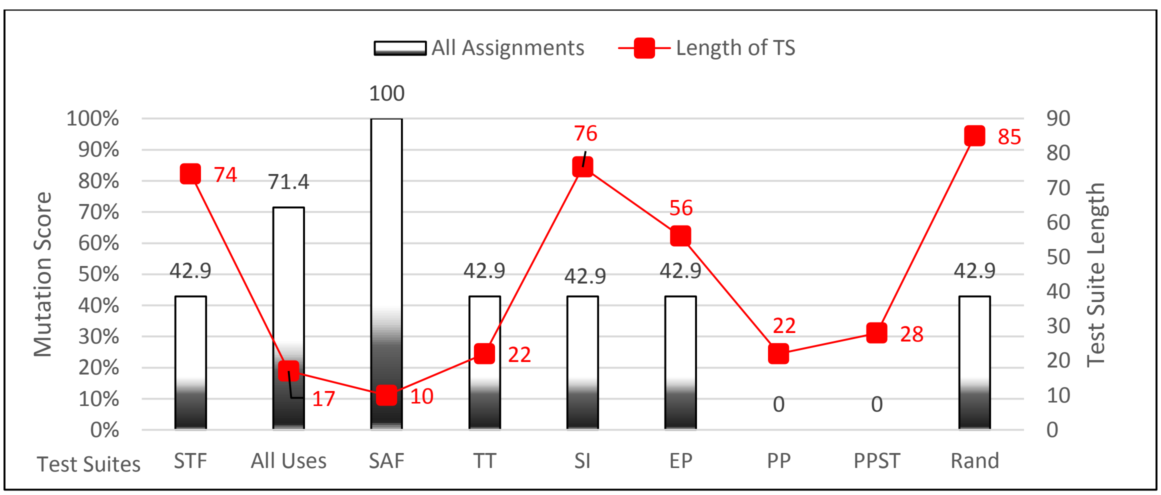

Figure 3.35 TS Coverage of All Assignments Mutants in POP 3 Example

d) Initiator Test Suites Fault Coverage Assessment

Figure 3.36 includes the mutation scores of All Assignments mutants and length of each considered test suite for the Initiator example.

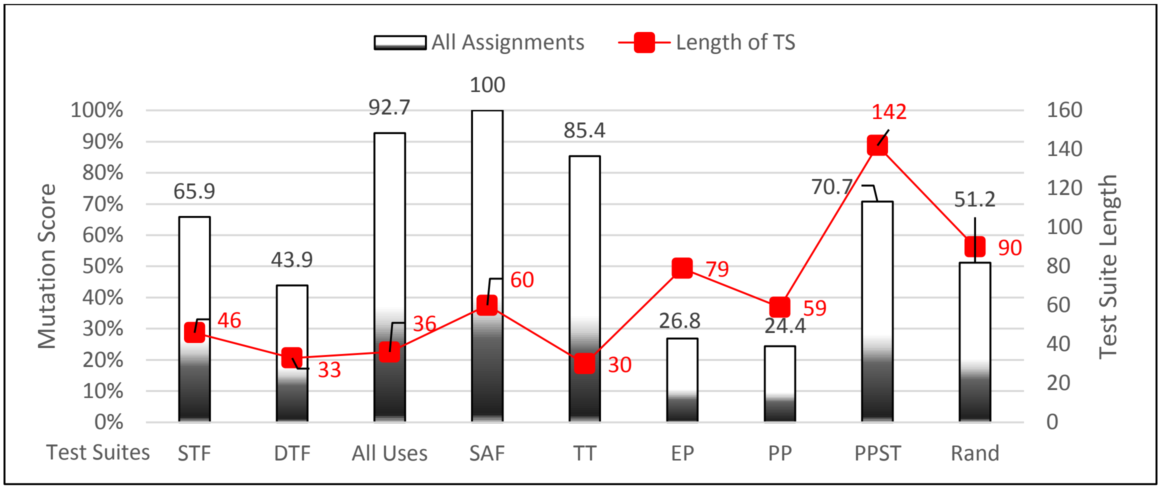

Figure 3.36 TS Coverage of All Assignments Mutants in Initiator Example

e) Responder Test Suites Fault Coverage Assessment

Figure 3.37 includes the mutation scores of All Assignments mutants and length of each considered test suite for the Responder example.

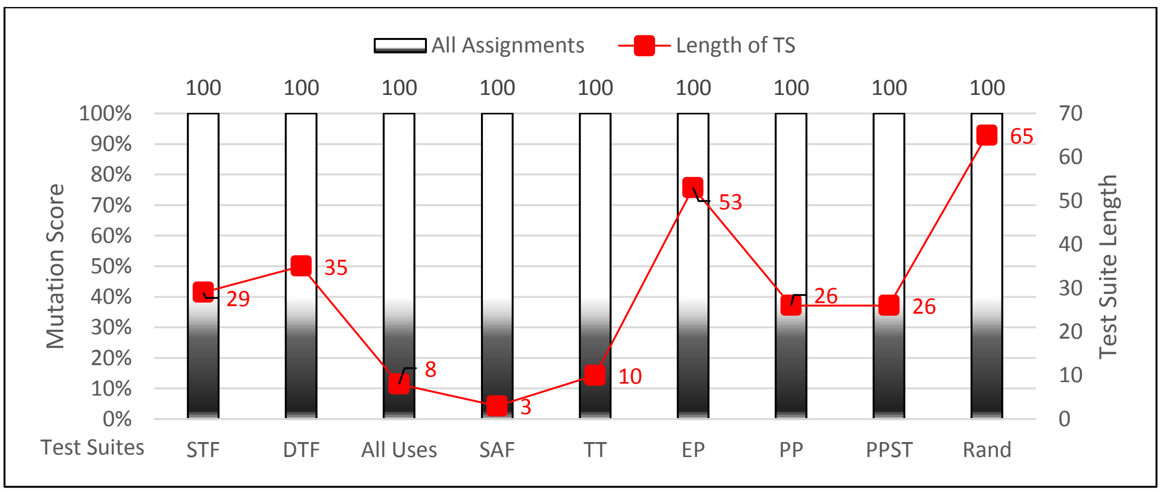

Figure 3.37 TS Coverage of All Assignments Mutants in Responder Example

f) SCP Test Suites Fault Coverage Assessment

Figure 3.38 includes the mutation scores of All Assignments mutants and length of each considered test suite for the SCP example.

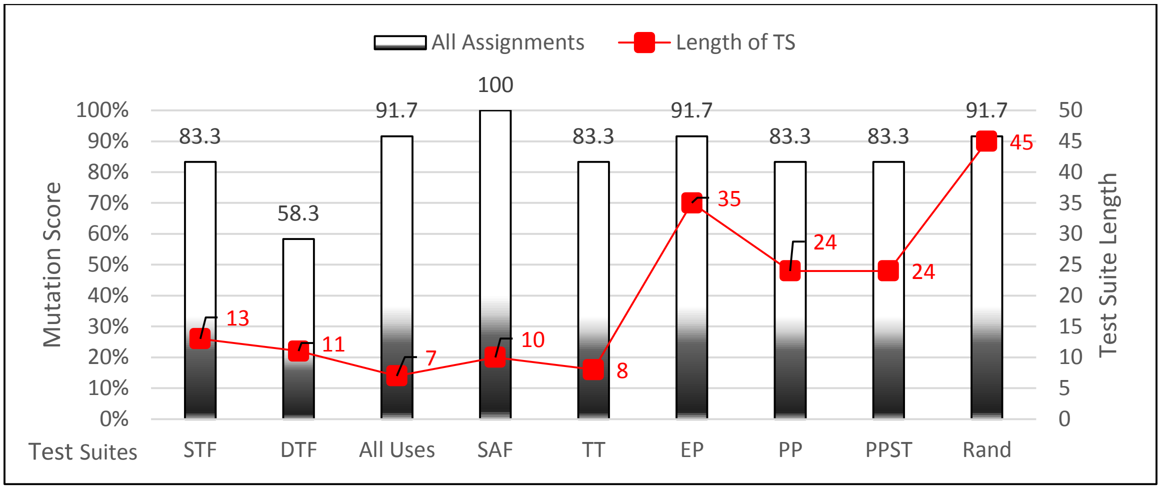

Figure 3.38 TS Coverage of All Assignments Fault (SAF) Mutants in SCP Example

g) Summary of All Assignments Fault (SAF) Mutants for all Considered Examples

Figure 3.39 includes the average mutation scores of All Assignments mutants and average length of each considered test suite for all the above considered examples.

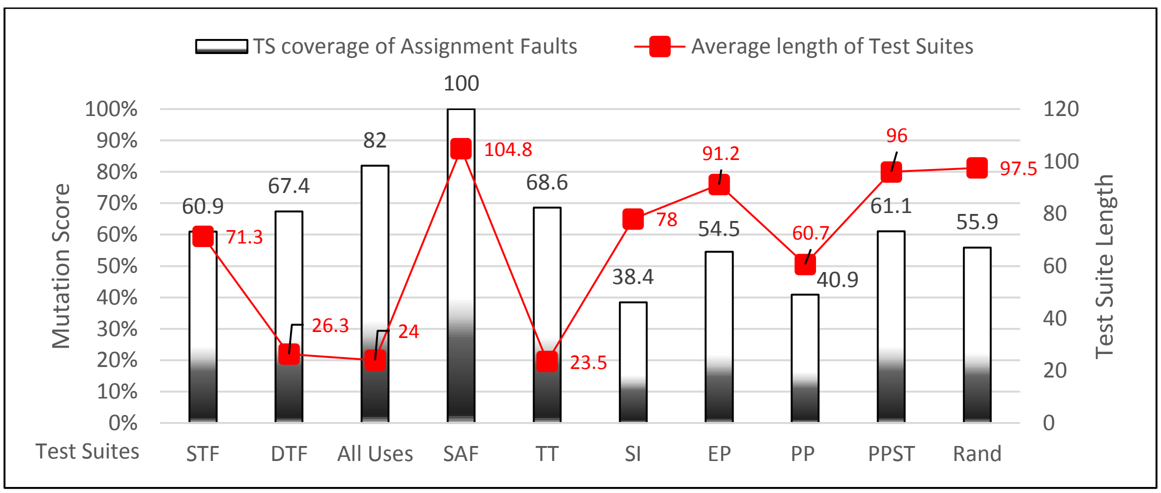

Figure 3.39 Average TS Coverage of All Assignments Mutants for All Examples

3.5.6 Ranking of Test Suites Coverage of Single and Double Transfer Fault Mutants

Based on Figures 3.14 and 3.21, Table 3.1 depicts the ranking of test suites (1 – Best, 6 – Worst) using the mutation score of (1).

Table 3.1 Ranking of Test Suites over STF and DTF Mutants Considering Mutation Score only

Ranking	STF Mutants	DTF Mutant
1	SI (100%), STF (100%)	SI (100%), STF (100%), DTF (100), PPST (98.7)
2	PPST (91.6%)	EP (97.8%), TT (96.6%)
3	DTF (87.3%), EP (86.8%)	Rand (93.1%)
4	TT (82.5%)	All Uses (86.6%), PP (85.3%)
5	Rand (80.1%)	SAF (82.9%)
6	PP (67.8%), All Uses (67.4%), SAF (66.9%)	

Based on Figures 3.14 and 3.21, Table 3.2 depicts the ranking of test suites (1 – Best, 8 – Worst) using the coverage-length score of (3).

Table 3.2 Ranking of Test Suites over STF and DTF Mutants Considering Mutation Score and Length

Ranking	STF Mutants	DTF Mutant
1	SI(96.7),STF(95.7)	PPST(98.3),EP(96.8),SI(96.7)
2	PPST(92.2)	STF(95.7),Rand(93.7)
3	EP(87.4)	DTF(89)
4	Rand(82.7)	SAF(86.2)
5	DTF(78.1)	TT(85.6)
6	TT(73.7),SAF(72.6)	PP(81.6)
7	PP(66.8)	All Uses(77.2)
8	All Uses(60.9)	

3.5.7 Ranking of Test Suites Coverage of Single Output Parameter Fault Mutants

Based on Figure 3.25, Table 3.3 depicts the ranking of test suites (1 – Best, 6 – Worst) using the mutation score of (1).

Table 3.3 Ranking of Test Suites over SOPF Mutants Considering Mutation Score only

Ranking	SOP Mutants
1	All Uses(92.4%), EP(90.3%), TT(90.3%)
2	STF(88.2%)
3	DTF(81.9%)
4	SAF(75.7%)
5	Rand(73.6%)
6	PP(71.5%), PPST(71.5%)

Based on Figure 3.25, Table 3.4 depicts the ranking of test suites (1 – Best, 6 – Worst) using the coverage-length score of (3).

Table 3.4 Ranking of Test Suites over SOPF Mutants Considering Mutation Score and Length

Ranking	SOP Mutants
1	EP(90.4)
2	STF(85.7)
3	All Uses(82.1),TT(80.3),SAF(80.1)
4	Rand(77.2),PPST(75.2)
5	DTF(73.6)
6	PP(69.9)

3.5.8 Ranking of Test Suites Coverage of All Assignments Fault Mutants

Based on Figure 3.39, Table 3.5 depicts the ranking of test suites (1 – Best, 6 – Worst) using the mutation score of (1).

Table 3.5 Ranking of Test Suites over All Assignments Mutants Considering Mutation Score only

Ranking	SAF Mutants
1	SAF(100%)
2	All Uses(82%)
3	TT(68.6%), DTF(67.4%)
4	PPST(61.1%), STF(60.9%)
5	Rand(55.9%), EP(54.5%)
6	PP(40.9%), SI(38.4%)

Based on Figure 3.39, Table 3.6 depicts the ranking of test suites (1 – Best, 7 – Worst) using the coverage-length score of (3).

Table 3.6 Ranking of Test Suites over All Assignments Mutants Considering Mutation Score and Length

Ranking	SAF Mutants
1	SAF(100.7)
2	All Uses(73.3)
3	PPST(66.3)
4	STF(62.5)
5	Rand(62.1),TT(61.8),DTF(61.3)
6	EP(60)
7	SI(44.4),PP(43.8)

3.5.9 Ranking of Test Suites Coverage of AD, AI and CRHS Mutants

Based on Figure 3.32, Table 3.7 depicts the ranking of test suites (1 – Best, 8 – Worst) using the mutation score of (1).

Table 3.7 Ranking of Test Suites over AI, AD and CRHS Mutants Considering Mutation Score only

Ranking	AD	AI	CRHS
1	SAF(100%)	SAF(100%)	SAF(100%)
2	All Uses(77%)	PPST(54.7%), All Uses(54.2%)	All Uses(90.3%)
3	TT(66.7%)	Rand(34.6%), TT(34%)	TT(74.7%)
4	DTF(62.5%)	EP(28.9%)	DTF(71.4%)
5	STF(58.9%), EP(58.5%)	PP(25%)	STF(67.7%), PPST(67.2%)
6	PPST(56.1%), Rand(55.8%)	STF(20.7%)	Rand(55.6%), EP(53.8%)
7	PP(40.9%)	SI(4%)	SI(50%)
8	SI(38.6%)	DTF(0%)	PP(43.3%)

Based on Figure 3.32, Table 3.8 depicts the ranking of test suites (1 – Best, 7 – Worst) using the coverage-length score of (3).

Table 3.8 Ranking of Test Suites over AI, AD and CRHS Mutants Considering Mutation Score and Length

Ranking	AD	AI	CRHS
1	SAF(100.7)	SAF(100.7)	SAF(100.7)
2	All Uses(69.1)	PPST(60.9)	All Uses(80.4)
3	EP(63.4), PPST(62.1), Rand(62)	All Uses(49.6)	PPST(71.5)
4	STF(60.7), TT(60.2)	Rand(44)	STF(68.3), TT(67)
5	DTF(57.1)	EP(38.2)	DTF(64.6)
6	SI(44.5), PP(43.9)	TT(32.4), PP(30.4)	Rand(61.9)
7		STF(28.3)	EP(59.4)
8		SI(15.1)	SI(54.2)
9		DTF(4)	PP(45.9)

3.5.10 Ranking of Test Suite Coverage of All Types of Considered Faults

For every test suite, all mutants with single transfer, double transfer, single Output Parameter, and all assignment faults are considered, and the corresponding mutation score is computed. The averages of the obtained scores over all considered examples are depicted in Table 3.9.

Table 3.9 Ranking Based on Fault Coverage Considering All Types of Faults

Ranking	All Types of Considered Faults
1	STF (87.3%)
2	TT (84.5%), DTF (84.2%)
3	EP (82.3%), All Uses (82.1%), SAF (81.4%)
4	PPST (80.7%), SI (79.5%)
5	Rand (75.6%)
6	PP (66.4%)

3.5.11 Summary of All the Obtained Results

Below is a summary of the experimental results presented in the above sections.

- For STF mutants, the best performing test suites in terms of fault coverage are the SI (100 %) and STF (100 %) followed by the PPST (91.6 %), DTF (87.3 %), EP (86.8 %), TT (82.5 %), Rand (80.1 %), PP (67.8 %), All Uses (67.4 %) and then the SAF (66.9 %) test suites. However, when considering the *coverage-length* score, the SI (96.7) and STF (95.7) test suites have comparable scores, and again they outperform the other test suites by approximately 14%. The PPST (92.2), EP (87.4), Rand (82.7), DTF (78.1), TT (73.7), SAF (72.6), PP (66.8) and (60.9) test suites have comparable scores, but each of these test suites scores less than the SI and the STF test suites by approximately 14%.
- For DTF mutants, the best performing test suites in terms of fault coverage are the SI (100 %), STF (100 %), DTF (100 %) and PPST (98.7) followed by EP (97.8 %), TT (96.6 %), Rand (93.1 %), All Uses (86.6 %), PP (85.3 %) and SAF (82.9 %). When considering the *coverage-length* score, the PPST (98.3), EP (96.8) and SI (96.7) outperform the other test suites by approximately 7%. It is noticed that test suites coverage of DTF mutants is higher than all the other mutants. On average, test suites coverage of DTF mutants is higher than STF mutants, SOPF mutants and All Assignment mutants by approximately 11%, 12% and 31%, respectively.

- For SOPF mutants, the best performing test suites in terms of coverage are All Uses (92.4 %), EP (90.3 %) and TT (90.3 %) followed by STF (88.2 %), DTF (81.9 %), SAF (75.7 %), Rand (73.6 %), PP (71.5 %) and PPST (71.5 %). However, when considering the *coverage-length* score, the EP (90.4) outperform the other test suites by approximately 13.7%. It is noticed that Rand and PPST test suites did not perform well in covering SOP mutants in contrast to their high coverage of STF mutants and DTF mutants.
- Test suites coverage of All Assignment mutants is lower than all other mutants. The best performing test suite for All Assignment mutants is SAF (100 %) followed by All Uses (82 %), TT (68.6%), DTF (67.4%), PPST (61.1%), STF (60.9%), Rand (55.9%), EP (54.5%), PP (40.9%) and SI (38.4%). When considering the *coverage-length* score, SAF (100.7) outperform the other test suites. All Uses (73.3) is ranked the second best after SAF test suites followed by PPST (66.3), STF (62.5), Rand (62.1), TT (61.8), DTF (61.3), EP (60), SI (44.4) and PP (43.8).
- Test suites coverage of CRHS mutants is higher than AD mutants; moreover, test suites coverage of AD mutants is higher than AI mutants. Other than SAF test suites, all test suites coverage of AD and AI mutants is low. After All Uses, TT is the second best test suite in covering AD, AI and CRHS mutants. PPST is ranked second in covering AI faults; whereas, it is ranked sixth and fifth in covering AD and CHRS faults. Other than SAF, All Uses, TT and PPST, all test suites have significantly low coverage of assignment faults.
- Based on the obtained results presented in Table 3.9, STF (87.3 %) outperform all the other test suites when considering all types of mutants, followed by TT (84.5 %), DTF (84.2 %), EP (82.3 %), All Uses (82.1 %), SAF (81.4 %), PPST (80.7 %), SI (79.5 %), Rand (75. 6%) and PP (66.4 %). Random test suites outperform All Uses, PP and SAF test suites in covering STF and DTF mutants. However, their coverage of SAF mutants is low as compared to these other test suites, and accordingly their overall coverage considering all types of faults is ranked 5/6. PP test suites are the worst in covering faults. In addition, they do not have high coverage of any considered type of faults. A considerable variation is observed in the SI test

suites coverage of STF (100 %) and DTF (100 %) faults as compared to their coverage of All Assignment (38.4 %) faults coverage.

3.5.12 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 [15], [21] – [30]. In summary, the studies reported in [21], [31] – [33] consider code-based mutation testing and the all uses criterion. Li et al. [28] conduct code-based experiments using code-based mutation, EP, all uses and the PP coverage criteria. Aynur et al. [26] 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 is reported in [34].

Recently, a study has been presented in [35] and [36]. 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 [35] and [36], SITS test suites are considered in the assessment. The results of the study show 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 a random test suite is considered, namely, random tests with same length as other EFSM tests. The results of random test suites show that the Random-All-Uses and All-Uses test suites provide comparable coverage where SITS test suites slightly outperform Random- SITSs. However, in [35] and [36], only three application examples are considered in the study, and as reported in [35] and [36], there is a need to consider more application examples to verify the results, and there is also a need to consider more types of EFSM test suites. In [37], study considers the same study as [35] and [36], but six working examples, including the three used in [35] and [36], are considered. In addition, [37] considers more types of EFSM test suites, namely, EP and PPST test suites. Furthermore, in [37], a comprehensive assessment of the fault coverage of random test suites is carried out. In this thesis, unlike [37], EFSM-based mutation testing on the same examples is considered as presented in [37]. That is, fault coverage of test suites is assessed with respect to EFSM mutants and not code based mutants as in [35] and [36]. Besides, the same types of test suites as presented in [37] are considered. According to [35] [36] and [37], the best test suite in terms of mutation score is SITS followed by the TT, STF and then the all uses test suites. Whereas, in this thesis, the pattern of results concluded is STF followed by TT, All Uses and then SITS, where the coverage of SITS is ranked 1st in the transfer faults but lowest when it comes to the coverage of Assignment faults.

3.6 Software Tool

In order to conduct experiments, a software tool has been implemented. This tool provides the following capabilities.

- Input an EFSM specification and test suite
- Derive the following types of EFSM mutants from an EFSM specification.
    1. All mutants with single transfer faults
    2. All mutants with double transfer faults
    3. Mutants with single assignment insertion faults
    4. Mutants with single assignment deletion faults
    5. Mutants single assignment CRHS faults
    6. Mutants with single output parameter faults
- Convert an EFSM specification into the corresponding FSM specification and also convert an EFSM test suite into the corresponding FSM test suite.
- Derive STF and DTF test suites as follows:
    1. Derive from the given EFSM specification the corresponding FSM specification
    2. Derive from the EFSM STF (DTF) mutants their corresponding FSM mutants.
    3. Use the tool [38] to generate using the FSM specification and considered mutants the corresponding STF and DTF FSM test suites.
    4. Use the tool to convert the STF (and DTF) FSM test suites into corresponding EFSM test suites
- Compute the mutation score and the coverage-length score of a given EFSM test suite and a particular type of considered faults
    1. Convert the EFSM test suite into the corresponding FSM test suite.
    2. Determine using the FSM specification and the considered FSM mutants, of a particular type, which of these mutants that are indistinguishable from the specification using the tool given in [38]

and the mutants killed by the test suite. Then compute the corresponding mutation score and coverage-length score.

We note that our software tool is used in many other studies, for example, it is used in the work reported in [35] [36] and [37] to derive various types of EFSM test suites.

Chapter 4: 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 highly powerful model for test derivation. In practice, developing and applying these test suites to an implementation under test is time consuming and costly. Thus, determining high quality test suites reduces the cost of software testing.

In this thesis, experiments and assessments are conducted, and the fault coverage of several EFSM-based test suites is compared in order to determine the quality of these test suites and thus reduce the cost of testing. Considered test suites include single transfer fault, double transfer fault, all uses, single assignment fault, transition tour, state identifier, edge pair, prime path and prime path with side trip test suites, and random test suites. The assessment is conducted using EFSM mutants of the considered specifications, namely, EFSM mutants with single and double transfer faults, single output parameter faults, and single assignment faults of many types such as assignment deletion, assignment insertion, and assignment change right hand side faults. Two criteria are used in assessing (or ranking) the test suites, the first criterion is based solely on *fault coverage* (or *mutation score*), and the other one is based on the fault coverage and the length (called *coverage-length score*) of the test suites.

In summary, the best performing test suites for STF mutants, in terms of fault coverage, are the SI (100 %) and STF (100 %) followed by the PPST (91.6 %), DTF (87.3 %), EP (86.8 %), TT (82.5 %), Rand (80.1 %), PP (67.8 %), All Uses (67.4 %) and then the SAF (66.9 %) test suites. However, when considering the *coverage-length* score, the SI (96.7) and STF (95.7) test suites have comparable scores, and again they outperform the other test suites by approximately 14%. The PPST (92.2), EP (87.4), Rand (82.7), DTF (78.1), TT (73.7), SAF (72.6), PP (66.8) and All Uses (60.9) test suites have comparable scores, but each of these test suites scores less than the SI and the STF test suites by approximately 14%.

For DTF mutants the best performing test suites in terms of fault coverage are the SI (100 %), STF (100 %), DTF (100 %) and PPST (98.7) followed by EP (97.8 %), TT

(96.6 %), Rand (93.1 %), All Uses (86.6 %), PP (85.3 %) and SAF (82.9 %). When considering the *coverage-length* score, the PPST (98.3), EP (96.8) and SI (96.7) outperform the other test suites by approximately 7%. For SOPF mutants, the best performing test suites in terms of coverage are All Uses (92.4 %), EP (90.3 %) and TT (90.3 %) followed by STF (88.2 %), DTF (81.9 %), SAF (75.7 %), Rand (73.6 %), PP (71.5 %) and PPST (71.5 %). However, when considering the *coverage-length* score, the EP (90.4) outperform the other test suites by approximately 13.7%.

Test suites coverage of All Assignment mutants is lower than all other mutants. The best performing test suite for All Assignment mutants is SAF (100 %) followed by All Uses (82 %), TT (68.6%), DTF (67.4%), PPST (61.1%), STF (60.9%), Rand (55.9%), EP (54.5%), PP (40.9%) and SI (38.4%). When considering the *coverage-length* score, SAF (100.7) outperform the other test suites.

Test suites coverage of CRHS mutants is higher than AD mutants. Similarly, test suites coverage of AD mutants is higher than AI mutants. Other than SAF test suites, all test suites coverage of AD and AI mutants is low. After All Uses, TT is the second best test suite in covering AD, AI and CRHS mutants. PPST is ranked second in covering AI faults; whereas, it is ranked sixth and fifth in covering AD and CHRS faults. Other than SAF, All Uses, TT and PPST, all test suites have significantly low coverage of assignment faults. When comparing the test suites based on the average of the obtained scores over all considered examples, STF outperform all the other test suites followed by TT, DTF, EP, All Uses, SAF, PPST, SI, Rand and PP. A considerable variation is observed in the SI test suites coverage of STF (100 %) and DTF (100 %) faults as compared to their coverage of All Assignment (38.4 %) faults coverage.

Finally, we note that though a clear pattern of results is shown using the conducted experiments for specifications modeled and extended finite state machines and for extended finite state machine types of faults. The results cannot be generalized for systems modeled using other formalisms and for other types of faults. Thus, there is a need to conduct assessments similar to that presented in this thesis to systems modeled using other formalisms, for example for UML state charts and other modeling.

References

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

[2] A. Petrenko, S. Boroday, and R. Groz, "Confirming configurations in EFSM testing," *IEEE Transactions on Software Engineering*, vol. 30, no. 1, pp. 29-42, 2004.

[3] 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*, 2014, pp. 222-229.

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

[5] Bourhfir, E. Aboulhamid, F. Khendek, and R. Dssouli, "Test cases selection from SDL specifications," *Computer Networks*, vol. 35, no. 6, pp. 693-708, 2001.

[6] 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.

[7] P. Ammann and J. Offutt, *Introduction to Software Testing*, *Cambridge University Press*, 2008.

[8] Eclipse Team. Eclipse Classic 3.4.2 | Eclipse Packages. [Online]. http://www.eclipse.org/downloads/packages/eclipse-classic-342/ganymedesr2 [Accessed on: 2015, July].

[9] 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, no. 2, pp. 109-127, 1993.

[10] N. Bombieri, F. Fummi, G. Pravadelli, "A mutation model for the SystemC TLM 2.0 communication interfaces," in *Proc. of the Conference on Design, Automation and Test in Europe* (*DATE'08*), 2008, pp. 396-401.

[11] 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.

[12] S. P. F. Fabri, M. E. Delamaro, J.C. Maldonado, P. Masiero, "Mutation analysis testing for finite state machines," in *Proc. of the International Symposium Reliability Engineering*, 1994, pp. 220-229.

[13] G. Fraser, F. Wotawa, P. E. Ammann, "Testing with model checker: a survey," *Journal of Software Testing, Verification, and Reliability*, vol. 19, no. 3, pp. 215-261, 2009.

[14] R. M. Hierons, M. G. Merayo, "Mutation testing from probabilistic and stochastic finite state machines," *Journal of Systems and Software*, vol. 82, no. 11, pp. 1804-1818, 2009.

[15] Y. Jia and M. Harman, "An analysis and survey of the development of mutation testing," *IEEE Transactions on Software Engineering*, vol. 37, no. 5, pp. 649-678, 2011.

[16] T. Sugeta, J. Maldonado and W. Wong, "Mutation testing applied to validate SDL specifications," Springer, pp. 193-208, 2004.

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

[18] 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, pp. 115-118.

[19] A. Nikitin and N. Kushik, "On EFSM-based Test Derivation Strategies," in *Proceedings of the Spring/Summer Young Researchers' Colloquium on Software Engineering*, 2010, pp. 116-119.

[20] W. Chen, "Executable test sequences for the protocol data flow property," *Int. Conference on Formal Techniques for Networked and Distributed Systems*, ed: Springer, pp. 285-299, 2002.

[21] A. J. Offutt, J. Pan, K. Tewary, and T. Zhang, "An experimental evaluation of data flow and mutation testing," *Software Practice and Experience*, vol. 26, no. 2, pp. 165-176, 1996.

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

[23] 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, no. 8, pp. 608-624, 2006.

[24] 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*, 2009, pp. 77-84.

[25] Paul Ammann and Jeff Offutt. Graph Coverage Web Application [Online]. http://cs.gmu.edu:8443/offutt/coverage/GraphCoverage [Accessed on: 2015, July]

[26] A. Abdurazik, Aynur, 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*, 2000, pp. 179-187.

[27] 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),* 2014, pp. 42-50.

[28] 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,* 2009, pp. 220-229.

[29] Wolski Sascha and Hennebrueder Sebastian. (2005, April) Eclipse Junit testing tutorial. [Online]. http://www.laliluna.de/articles/posts/eclipse-junit-testing-tutorial.html [Accessed on: 2015, July].

[30] 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.

[31] 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, no. 3, pp. 235-253, 1997

[32] 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, no. 1, pp. 9-31, 1994.

[33] 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.

[34] 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*, 2008, pp. 867-872.

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

[36] K. El-Fakih, T. Salameh and N. Yevtushenko, "On Code Coverage of Extended FSM Based Test Suites: An Initial Assessment," in *Proc. of the 26th International Conference on Testing Software and Systems, Madrid, Spain, Nov. 23-25, Lecture Notes in Computer Science 8763*, 2014, pp. 198-204.

[37] M. Hassoun, "Fault Coverage and Diagnosis of Protocols and Systems Modeled As Extended Finite State Machines," *M.S. Thesis, Dept. Computer Science and Eng., American University of Sharjah,* 2015.

[38] A. da Silva Simao, A. Ambrósio, S. Fabbri, A. do Amaral, E. Martins and J. Maldonado, "Plavis/FSM: an Environment to Integrate FSM-based Testing Tools", *Tool Session of XIX Brazilian Symposium on Software Engineering*, 2008, pp. 1-6.

Vita

Noshad Khan Jadoon was born on June 1, 1984, in Dubai, UAE. He was educated in private schools in Pakistan and graduated from Pakistan International Public School and Boys College in 2003.

He studied at COMSATS Institute of Information Technology in Abbottabad, Pakistan, from which he graduated in 2007 and obtained a Bachelor's Degree in Computer Engineering.

Mr. Noshad moved to the United Arab Emirates in 2008. In 2010 he joined the Master's program in Computer Engineering at the American University of Sharjah.