Testing Nondeterministic Finite State Machines With Respect to the Separability Relation

Natalia Shabaldina[1], Khaled El-Fakih[2], Nina Yevtushenko[1]

[1]Tomsk State University, 36 Lenin Str.. Tomsk, 634050, Russia
snv@kitidis.tsu.ru, yevtushenko@elefot.tsu.ru
[2]American University of Sharjah, PO Box 26666, UAE
kelfakih@aus.edu

Abstract. In this paper, we propose a fault model and a method for deriving complete test suites for nondeterministic FSMs with respect to the separability relation. Two FSMs are separable if there exists an input sequence such that the sets of output responses of these FSMs to the sequence do not intersect. In contrast to the well-known reduction and equivalence relations, the separability relation can be checked when the «all weather conditions» assumption does not hold for a nondeterministic Implementation Under Test (IUT). A (complete) test suite derived from the given (nondeterministic) FSM specification using the separability relation can detect every IUT that is separable from the given specification after applying each test case only once. Two algorithms are proposed for complete test derivation without the explicit enumeration of all possible implementations. The first algorithm can be applied when the set of possible implementations is the set of all complete nondeterministic submachines of a given mutation machine. The second algorithm is applied when the upper bound on the number of states of an IUT is known.

Keywords: separability relation, testing nondeterministic FSMs.

1. Introduction

A number of conformance testing methods have been developed for deriving tests when the system specification and implementation are represented by nondeterministic FSMs [1–16]. Non-determinism occurs due to various reasons such as performance, flexibility, limited controllability, and abstraction [8] [11] [13] [17].

A number of methods have been proposed for test generation against nondeterministic FSM specifications with the guaranteed fault coverage with respect to appropriate fault models. The methods given in [3] and [4] derive test suites with respect to the equivalence relation for a nondeterministic implementation against a nondeterministic specification while the methods given in [6], [8], [9], and [10] derive test suites for a complete deterministic implementation against a nondeterministic specification with respect to the reduction relation. Two FSMs are *equivalent* if they have the same input/output behavior and an FSM T is a *reduction* of FSM S if the input/output behavior of T is a subset of that of S. Hierons [11] presented a test derivation method with respect to the reduction relation when a system implementation can be nondeterministic. Petrenko and Yevtushenko [15] generalize the work given in [18] and proposed a method that derives a test suite with respect to the reduction and equivalence relations for a nondeterministic implementation against possibly a partial specification.

When deriving test suites with respect to the reduction and equivalence relations with the guaranteed fault coverage the so-called *complete testing* assumption [3–4] (called «*all weather conditions*» by Milner in [19]) is assumed to be satisfied when testing a nondeterministic implementation. According to this assumption, if an input sequence (a test case) is applied a number of times to a nondeterministic implementation, then all possible output sequences of the implementation to this test case can be observed. However, when an Implementation Under Test (IUT) has a limited controllability, as happens, for instance, in remote testing, the complete testing assumption cannot be satisfied. In this case, the only relation that can be used for the preset test derivation with the guaranteed fault coverage [20] [21] is the separability relation defined by Starke in [22]. Two FSMs are *separable* if there is an input sequence, called a *separating sequence*, such that the sets of output responses of these FSMs to the sequence do not intersect, i.e., the sets are disjoint. It is known [15] [21] that test suites derived with respect to the equivalence and reduction relations cannot be used for testing the separability relation. The fact that an FSM with m states is not equivalent (or not a reduction) of an FSM with n states can always be established by an input sequence of length up to mn [15], while there exist two FSMs which can be separated only with an input sequence of exponential length [5].

The separability relation was further studied by Alur et al. in [5] where an algorithm for deriving a separating sequence for two separable states of an FSM with n states is proposed and is shown that the upper bound on the length of a shortest separating sequence is exponential. In [23], it is shown that given FSMs S with n states and T with m states, the length of a shortest separating sequence is at most 2^{mn-1} and this upper bound is reachable. An algorithm is proposed for deriving a shortest separating sequence of the given FSMs. However, experiments with the proposed algorithm show that on average, the length of a shortest separating sequence is less than mn and the existence of a separating sequence significantly depends on the number of nondeterministic transitions in the given FSMs. For all conducted experiments the upper bound 2^{mn-1} on the length of a separating sequence was never reached.

In this paper, we consider the test derivation w.r.t. the reduction relation when «all weather conditions» assumption may not be held for an IUT. A test suite is called complete up to the separability relation if it detects every non-reduction of the specification FSM from a given fault domain that is separable from the given FSM specification. If each test case of the test suite is applied to an IUT of the fault domain once and an IUT is separable from the specification FSM, then the IUT will be detected with such a test suite. However, this test suite can also detect some other implementations which are non-reductions of the specification FSM but are not separable with the specification FSM. Correspondingly, we refine the notions of a fault model and of a complete test suite. We propose a method for deriving a complete test suite without the explicit enumeration of FSMs of the fault domain when the fault domain is the set of all submachines of a given mutation machine (including those which are non-deterministic) and when the fault domain has each implementation FSM up to *m* states. We also demonstrate that not every test suite that is complete w.r.t. the reduction relation under the complete testing assumption can be used for testing up to the separability relation when each test case is applied to an IUT at most once.

This paper is organized as follows. Section 2 includes all necessary definitions. In Section 3 we refine the notion of a fault model and define a complete test suite w.r.t. the refined fault model. Sections 4 and 5 contain algorithms for building a complete test suite w.r.t. the refined model where the fault domain is the set of all complete (not only deterministic) submachines of a given nondeterministic FSM (a mutation machine) and where the fault domain is the set of all nondeterminisitic FSMs with at most *m* states. Section 6 concludes this paper.

2. Preliminaries

A *finite state machine* (*FSM*) S is a 5-tuple $\langle S, I, O, h_S, s_1\rangle$, where S is a finite nonempty set with s_1 as the initial state; I and O are input and output alphabets; and $h_S \subseteq S\times I\times O\times S$ is a behavior relation. The behavior relation defines all possible transitions of the machine. Given a current state s_j and input symbol i, a 4-tuple $(s_j,i,o,s_k)\in h_S$ represents a possible transition from state s_j under the input i to the next state s_k with the output o, usually written as $s_j \xrightarrow{i/o} s_k$. If for each pair $(s,i)\in S\times I$ there exists $(o,s')\in O\times S$ such that $(s,i,o,s')\in h_S$ then FSM S is said to be *complete*; otherwise, FSM S is *partial*. If for each $(s,i,o)\in S\times I\times O$ there is at most one transition $(s,i,o,s')\in h_S$ then FSM S is said to be *observable*. Given FSM $S = \langle S, I, O, h_S, s_1\rangle$, state s and an input i, state s' is a *successor* of state s under the input i or simply an *i-successor* of state s if there exist $o \in O$ such that the 4-tuple $(s,i,o,s')\in h_S$. Given a set of states $b \subseteq S$ and an input i, the set of states b' is a *successor* of the set b under the input i or simply an *i-successor* of b if b' is the set of all i-successors of states of the set b.

In the usual way, the behavior relation h_S is extended to input and output sequences. Given states $s,s'\in S$, input sequence $\alpha=i_1i_2\ldots i_k\in I^*$ and output sequence $\beta=o_1o_2\ldots o_k\in O^*$. Transition $(s, \alpha, \beta, s')\in h_S$ if there exist states $s=s_1, s_2, \ldots, s_k, s_{k+1}=s'$ such that $(s_i,i_i,o_i,s_{i+1})\in h_S$, $i=1, \ldots, k$. As usual, given a defined input sequence α at state s, $h_S^O(s, \alpha)$ denotes the set of all output sequences which FSM S produces at state s under the input sequence α, i.e. $h_s^O(s, \alpha) = \{\beta: \exists\, s'\in S\ [(s,\alpha,\beta,s')\in h_S]\}$.

Given an FSM $M = \langle M,I,O,h_M,m_1\rangle$, an FSM $S = \langle S,I,O,h_S,s_1\rangle$, $S\subseteq M$, $s_1 = m_1$, is a *submachine* of FSM M if $h_S \subseteq h_M$, i.e., if each transition of FSM S is obtained by fixing an appropriate transition of the FSM M. The set of all complete submachines, including those which are non-deterministic, of a complete FSM S is denoted $Sub_{nd}(S)$.

Given FSMs $S = \langle S,I,O,h_S,s_1\rangle$ and $T = \langle T,I,O,h_T,t_1\rangle$, the *intersection* $S \cap T$ is defined as the largest connected submachine of the FSM $\langle S{\times}T,I,O,h,s_1t_1\rangle$ where $(st,i,o,s't')\in h \Leftrightarrow (s,i,o,s')\in h_S$ & $(t,i,o,t')\in h_T$.

Complete FSMs S and T are *equivalent*, written $S \cong T$, if for each sequence $\alpha \in I^*$ $[h_T{}^O(t_1, \alpha) = h_S{}^O(s_1, \alpha)]$, i.e., the sets of output sequences of FSMs S and T under each input sequence coincide. If there exist sequence $\alpha \in I^*$ $[h_T{}^O(t_1, \alpha) \neq h_S{}^O(s_1, \alpha)]$ then FSMs S and T are *distinguishable*, written $S \not\cong T$.

A state t of a complete FSM T is a *reduction* of a state s of a complete FSM S, written $t \leq s$, if for each sequence $\alpha \in I^*$ $[h_T{}^O(t, \alpha) \subseteq h_S{}^O(s, \alpha)]$, i.e., the set of output sequences of FSM S at state s contains the set of output sequences of FSM T at state t under each input sequence. If there exists sequence $\alpha \in I^*$ $[h_T{}^O(t, \alpha) \not\subset h_S{}^O(s, \alpha)]$ then state t is not a reduction of state s, written $t \not\leq s$. FSM T is a *reduction* of a FSM S, written $T \leq S$, if the reduction relation holds between the initial states, i.e., for each sequence $\alpha \in I^*$ $[h_T{}^O(t_1, \alpha) \subseteq h_S{}^O(s_1, \alpha)]$. If there exists sequence $\alpha \in I^*$ $[h_T{}^O(t_1, \alpha) \not\subset h_S{}^O(s_1, \alpha)]$ then FSM T is not a reduction of FSM S, written $T \not\leq S$.

A state t of a complete FSM T is *r-compatible* with a state s of a complete FSM S [21], written $t \simeq s$, if there exists a complete FSM $B = \langle B,I,O,h_B,b_1\rangle$ and state $b \in B$ such that b is a reduction of both states t and s. If states t and s are not r-compatible then they are *r-distinguishable*, written $t \not\simeq s$. In this case, there exists an r-distinguishability finite set W of input sequences such that for each complete FSM $B = \langle B,I,O,h_B,b_1\rangle$ and each state $b \in B$ there exists $\alpha \in W$ such that $[h_B{}^O(b, \alpha) \not\subset h_S{}^O(s, \alpha)]$ or $[h_B{}^O(b, \alpha) \not\subset h_T{}^O(t, \alpha)]$. FSMs T and S are r-compatible, written $T \simeq S$ (or r-distinguishable, written $T \not\simeq S$) if the corresponding relation holds between their initial states.

Complete FSMs T and S are *non-separable*, written $T \sim S$, if for each sequence $\alpha \in I^*$ $[h_T{}^O(t_1, \alpha) \cap h_S{}^O(s_1, \alpha) \neq \varnothing]$, i.e., the sets of output sequences of FSMs T and S under each input sequence intersect. If there exists sequence $\alpha \in I^*$ $[h_T{}^O(t_1, \alpha) \cap h_S{}^O(s_1, \alpha) = \varnothing]$ then FSMs T and S are *separable*, written $T \nsim S$. In the latter case, the sequence α is called a *separating* sequence of FSMs T and S.

3. Fault model and a test suite

When testing w.r.t. the reduction relation the traditional fault model is a triple $<S, \leq, \mathfrak{R}>$, where S is a specification FSM, $\leq$ is the reduction relation, fault domain $\mathfrak{R}$ is the set of all possible (faulty and non-faulty) implementation FSMs with the same input and output alphabets as the specification FSM S. As usual, FSMs of the set $\mathfrak{R}$ represent all possible faults which can happen when implementing the specification. An implementation FSM $T \in \mathfrak{R}$ is called *conforming* if $T \leq S$;

otherwise, T is a *non-conforming* implementation. Given the specification FSM S, a *test case* is a finite input sequence of S. As usual, a *test suite* is a finite set of test cases. Given an implementation FSM $T \in \Re$ and an output response β of T to a test case α, the FSM T *passes* the test case if β is in the set of output responses of the specification FSM S to α. Otherwise, the FSM T *fails* the test case. Given a test suite, an Implementation Under Test (IUT) *passes* the test suite if the IUT passes each test case.

In this paper, we generalize the traditional fault model $<S, \leq, \Re>$ by adding the relation $\nsim$ into the fault model. Formally a fault model becomes a 4-tuple $<S, (\leq, \nsim), \Re>$. This model indicates that a test suite is complete up to the subset of $\Re$ that contains all implementations $T \nsim S$. That is a test suite is *complete* w.r.t. to the fault model $<S, (\leq, \nsim), \Re>$ if each non-reduction T of S such that $T \nsim S$ can be detected with this test suite.

Here we note that in FSM-based testing when using a traditional fault model it is usually assumed that each non-conforming implementation can be detected with a complete test suite. However, if the assumption of «all weather conditions» fails when testing a non-deterministic implementation, as happens, for example, in the remote testing, then a non-deterministic implementation cannot be tested up to the reduction or up to the equivalence relation. In this paper, we show that in this case, an implementation can be tested up to the separability relation without relying on «all weather conditions» assumption. As usual, we assume that both specification and implementation FSMs are complete. However, we do not require either the specification FSM or an implementation FSM to be observable.

The relation «not a reduction» contains the separability relation, i.e. for nondeterministic FSMs $\nsim \subseteq \not\leq$. However, as the following example shows a complete test suite w.r.t. the fault models $<S, \leq, \Re>$ is not always complete w.r.t. the fault model $<S, (\leq, \nsim), \Re>$.

Consider the specification FSM S in Figure 1 with states $\{a,b\}$, inputs $\{x, y\}$ and outputs $\{1,2,3,4\}$. States of S are separated by the input y and both states are deterministically reachable from the initial state a. We use the method for test derivation [21] and obtain a test suite $TS = \{xy, yxy, yyy\}$ w.r.t. the fault model $<S, \leq, \Re_2>$ where $\Re_2$ contains each complete FSM with up to 2 states over the input alphabet $\{x, y\}$. The test suite TS is complete when the assumption of «all weather conditions» holds. However, if the assumption of «all weather conditions» fails then the implementation FSM T in Figure 1 that is separable with S, i.e., is not a reduction of S, can remain undetected with the test suite TS when each test case is applied at most once, i.e., TS is not complete w.r.t. the fault model $<S, (\leq, \nsim), \Re_2>$. By direct inspection, one can assure that a shortest sequence that separates the initial states a and 1 of S and T has length four while to each input sequence of length up to three the sets of output responses of S and T intersect.

S	a	b
x	a/0,1,2,3	a/1,2
y	b/1,2	a/0 b/3

T	1	2
x	1/0 2/1	1/0,1
y	1/1 2/0,2	1/3 2/0

Fig. 1. FSMs S and T

The reason is that the r-distinguishability relation between states of the specification FSM cannot be used when deriving tests w.r.t. the separability relation. Given two r-distinguishable states s_1 and s_2 of the specification FSM with an r-distinguishability set W and a state t of an implementation FSM, there exists a sequence $\alpha \in W$ such that $t \not\leq_\alpha s_1$ or $t \not\leq_\alpha s_2$. Therefore, if a test suite has two sequences α_1 and α_2 that take the specification FSM to states s_1 and s_2 appended with the r-distinguishability set W then each implementation FSM that is taken with sequences α_1 and α_2 to the same state t will be detected with such a test suite. Unfortunately, the above property does not hold for the separability relation. Given two separable states s_1 and s_2 in the specification FSM with a separating sequence α, a state t of an implementation FSM can be non-separable with both states s_1 and s_2 w.r.t. the sequence α.

Given the specification FSM S and the fault domain $\Re$, a complete test suite w.r.t. the fault model $<S, (\leq, \nsim), \Re>$ can be derived by the explicit enumeration of all machines of the set $\Re$. For each $T \in \Re$ that is separable with S, a separating sequence is derived that is used as a test case to detect a wrong implementation FSM T. The set of all test cases is a complete test suite w.r.t. the fault model $<S, (\leq, \nsim), \Re>$). Below we include the algorithm given in [23] for deriving a separating sequence for two complete FSMs.

Algorithm 1. Deriving a separating sequence of two complete FSMs

Input: Complete FSMs $S = \langle S, I, O, h_S, s_1 \rangle$ and $T = \langle T, I, O, h_T, t_1 \rangle$

Output: A shortest separating sequence of FSMs S and T (if it exists)

Step 1. Derive the intersection $S \cap T$. If the intersection is a complete FSM then the FSMs S and T are non-separable. END Algorithm 1.

Step 2. If the intersection $S \cap T$ is a partial FSM, then derive a truncated successor tree of the intersection $S \cap T$. The root of this tree, which is at the 0th level, is the initial state (s_1, t_1) of the intersection; the nodes of the tree are labeled with subsets of states of the intersection. Given already derived j tree levels, $j \geq 0$, a non-leaf (intermediate) node of the jth level labeled with a subset P of states of the intersection, and an input i, there is an outgoing edge from this non-leaf node labeled with i to the node labeled with the subset of the i-successors of states of the subset P. A current node *Current*, at the kth level, $k \geq 0$, labeled with the subset P of states, is claimed as a leaf node if one of the following conditions holds:

Rule 1: There exists an input i such that each state of the set P has no i-successors in the intersection $S \cap T$.

Rule 2: There exists a node at a jth level, $j < k$, labeled with subset R of states with the property: for each state (s', t') of R there exists a state (s, t) of P such that $(s', t') \leq (s, t)$.

Step 3. If none of the paths of the truncated tree derived at Step 2 is terminated using Rule 1 then FSMs S and T are non-separable. END Algorithm 1. Otherwise, if there is a leaf node, *Leaf*, labeled with the subset P of pairs of states such that for some input i, each pair of the set P has no i-successors, then a shortest sequence αi where α labels the path from the root of the tree to *Leaf*, is a shortest separating sequence of FSMs S and T.

Theorem 1. Given FSMs S and T over input alphabet I and output alphabet O, Algorithm 1 returns a shortest separating sequence of FSMs S and T (if a separating sequence exists).

Proof. In order to separate FSMs S and T we need an input sequence α under which the intersection $S \cap T$ enters the set P of states such that there exists some input i that separates each pair of the set P, i.e., in the intersection, each state of the set P has no successors under input i. If none of the paths of the truncated tree derived at Step 2 of Algorithm 1 is terminated according termination Rule 1, then there is no such an input sequence α, and thus, FSMs S and T are non-separable. If there exists a path of the truncated tree derived at Step 2 of Algorithm 1 that is terminated according termination Rule 1 then the sequence α which labels this path takes the intersection $S \cap T$ to the set P such that there exists an input i that separates each pair of the set P, and thus, αi is a separating sequence of FSMs S and T. Given a current node *Current* labeled with a subset P at kth level, let there exist a node at a jth level, $j < k$, labeled with subset R of states with the property: for each state (s', t') of R there exists a state (s, t) of P such that $(s', t') \leq (s, t)$. In this case, each input sequence that separates each pair of states of the set P also separates each pair of states of the set R. Therefore, each path of the truncated tree traversing the node *Current* can be terminated, since a shorter separating sequence can be derived when traversing the node labeled with the set R.

❑

As usual, the explicit enumeration of all machines in the fault domain $\mathfrak{R}$ can be applied only to small fault domains. Accordingly, in the following section, we propose a method for deriving a complete test suite w.r.t. the fault model $<S, (\leq, \nsim), \mathfrak{R}>$ without the explicit enumeration of the machines in $\mathfrak{R}$. This is done by using a nondeterministic FSM called a *mutation machine* MM in [9], to represent, in a compact way, all possible implementations of S. In this case, the fault domain $\mathfrak{R}$ equals the set $Sub_{nd}(MM)$ of all complete submachines of MM. In Section 5, we extend the method to the case when the upper bound on the number of states of an IUT is given.

4. Deriving a complete test suite w.r.t. fault model <S, ($\leq$,$\nsim$), $Sub_{nd}(MM)$>

According to Algorithm 1, in order to derive a complete test suite w.r.t. the fault model <S, ($\leq$,$\nsim$), $Sub_{nd}(MM)$> a truncated successor tree of the intersection $S \cap T$ should be derived for each complete submachine T of the FSM MM. Therefore, given a current node *Current* labeled with a subset P and input i, we should have an edge labeled by i not only to the i-successor of P but to all non-empty subsets of the i-successor. Moreover, we cannot terminate a path comparing the label of its node with labels of another path, since now these paths can belong to different submachines of MM. The above two observations lead us to a method below when deriving a complete test suite w.r.t. the fault model <S, ($\leq$,$\nsim$), $Sub_{nd}(MM)$>.

Algorithm 2. Deriving a complete test suite w.r.t. the fault model <S, ($\leq$,$\nsim$), $Sub_{nd}(MM)$>
Input: Complete FSMs S and MM
Output: A complete test suite TS w.r.t. the fault model <S, ($\leq$,$\nsim$), $Sub_{nd}(MM)$>
Step 1. Derive the intersection $S \cap MM$.
Step 2. Derive a truncated successor tree of the intersection $S \cap MM$. The root of this tree, which is at the 0th level, is the initial state (s_1, m_1) of the intersection; the nodes of the tree are labeled with subsets of states of the intersection. Given already derived j tree levels, $j \geq 0$, a non-leaf (intermediate) node of the jth level labeled with a subset P of states of the intersection, and an input i, there is an outgoing edge from this non-leaf node labeled with i to the node labeled with each subset of the i-successor of the subset P. A current node *Current*, at the kth level, $k \geq 0$, labeled with the subset P of states, is claimed as a leaf node if one of the following conditions holds:

Rule 1: There exists an input i such that each state of the set P has no i-successors in the intersection $S \cap MM$.

Rule 2: There exists a node at the path from the root to this node at jth level, $j < k$, labeled with subset R of states with the property: for each state (s', m') of R there exists a state (s, m) of P such that (s', m') $\leq$ (s, m).

Step 3. For each path of the tree terminated using Rule 1, include into TS an input sequence that labels the path appended with an input i such that each state of the set P corresponded to the final node of the path has no i-successors in the intersection $S \cap MM$.

For each path of the tree terminated using Rule 2, include into TS an input sequence that labels the path.

Theorem 2. Given FSMs S and MM over the input alphabet I and output alphabet O, Algorithm 2 returns a complete test suite w.r.t. the fault model <S, ($\leq$,$\nsim$), $Sub_{nd}(MM)$>.

Proof. According to Algorithm 1, when deriving a separating sequence for two FSMs S and T we use the truncated tree of $S \cap T$. In our case, each FSM T that should be separated with S (if S and T are separable) is a submachine of MM, i.e., $S \cap T$ is a submachine of $S \cap MM$. In order to get an appropriate truncated subtree for each submachine of FSM MM at Step 2 of Algorithm 1, for each non-leaf node *Current* labeled with a subset P and each input i, we add an outgoing edge to each non-empty subset of the i-successor of P. Thus, for each complete submachine T of MM a truncated tree for separating T and S is a subtree of the tree derived by Algorithm 2.

❑

Example. As an application example, consider FSMs S in Figure 1 and MM in Figure 2. We apply Algorithm 2 we obtain the intersection (Figure 3) and the truncated successor tree in Figure 4. Therefore, the set {*xx*, *xyx*, *xyyx*, *xyyy*, *yxx*, *yxyx*, *yxyy*, *yyxx*, *yyxyx*, *yyxyy*, *yyy*} is a complete test suite w.r.t. the fault model $<S, (\leq, \nsim), Sub_{nd}(MM)>$.

MM	1	2
x	1/0,1 1/2,3 2/2,3	1/2,3 2/2,3
y	1/1,2,3 2/1,2	1/0,2 2/2,3

$S \cap MM$	$a1$	$a2$	$b1$	$b2$
x	$a1$/0,1,2,3 $a2$/2,3	$a1$/2,3 $a2$/2,3	$a1$/1,2 $a2$/2	$a1$/2 $a2$/2
y	$b1$/1,2 $b2$/1,2	$b1$/2 $b2$/2	$b1$/3	$a1$/0 $b2$/3

Fig. 2. FSMs MM and $S \cap MM$

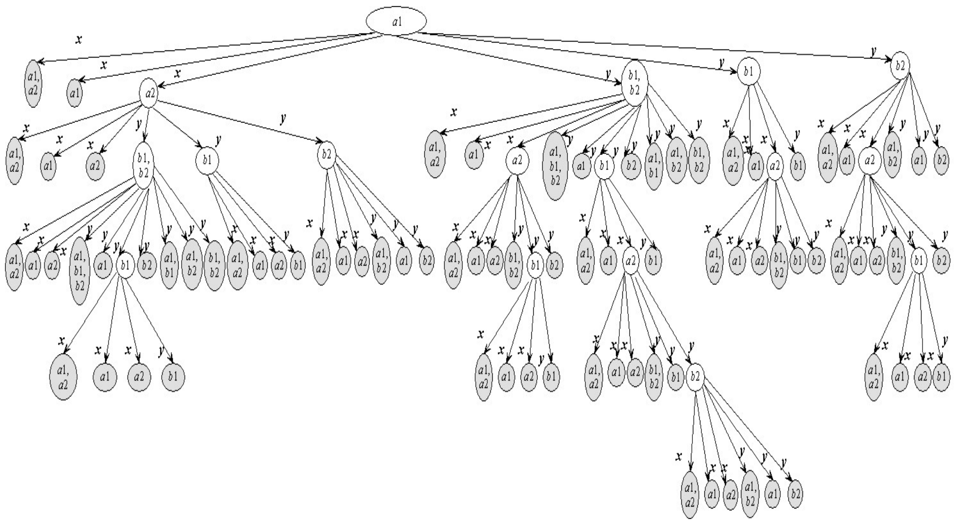

Fig. 3. The truncated successor tree

Sometimes a test suite derived using Algorithm 2 can be shortened by relaxing the conditions of Step 3. For example, given a tail edge $\{sm\}$ $\{s'm'\}$ of some path terminated using Rule 1 and labeled with an input sequence αi, it can happen that the set of output responses of the intersection $S \cap MM$ to i at state sm coincides with that of the MM at state m. In this case, i is unnecessary for separating new submachines of MM from S and it is enough to include into a test suite the sequence α instead of αi. More analysis is needed for reducing a test suite. This is a part of our future work.

We implemented the above algorithm and performed some experiments with FSMs with small number of states. As our experiments show, the total length of a complete test suite w.r.t. the fault model $<S, (\leq, \not\sim), Sub_{nd}(MM)>$ significantly depends on the number of nondeterministic transitions in the specification and mutation machines. Table 1 contains a selected part of conducted experiments for FSMs with 5 states. Each row in the table represents an average test suite length of 100 randomly generated specification FSMs. Each FSM S is a complete nondeterministic FSM with $|S|$ states, $|I|$ inputs, $|O|$ outputs, where for 20 percent of pairs (s, i) there is more than one outgoing transition from state s under input i. Each FSM MM is derived by adding (up to 25 percent) additional transitions to FSM S.

Table 1. A selected part of conducted experiments

$\|I\|$	$\|O\|$	Average test suite length
2	2	2432
3	3	15407
3	4	6826

5. Deriving a complete test suite w.r.t. fault model $<S, (\leq, \not\sim), \mathfrak{R}_m>$

Let $\mathfrak{R}_m$ be the fault domain of a given specification S that contains each complete implementation FSM of S, over the same input and output alphabets of S, with up to a given number m of states. The following theorem can be used for deriving a complete test suite w.r.t. the fault model $<S, (\leq, \not\sim), \mathfrak{R}_m>$. This theorem is a corollary of Theorem 2 given in our previous work [23].

Theorem 3. Given the specification FSM S with n states, a test suite $I^{2^{mn-1}}$ is complete w.r.t. the fault model $<S, (\leq, \not\sim), \mathfrak{R}_m>$.

In the following, based on the idea of counting states of the specification FSM when deriving a complete test suite w.r.t. the fault model $<S, \leq, \mathfrak{R}_m>$) (a SC-method [15]), we propose a test derivation method for reducing the test suite $I^{2^{mn-1}}$. In this case, unlike the above method given in Algorithm 1, we derive a truncated tree using only the specification FSM S. Before terminating a path at a node labeled with a subset K of states of the specification FSM, we make sure that for each complete FSM T with up to m states the path traverses all possible subsets of the Cartesian product $K \times T$ in the intersection $S \cap T$, i.e., the path should traverse not less than $2^{|K|\cdot m}$ subsets of K. If K contains the initial state, then the initial state of the intersection can be excluded from any subset that labels a non-root node of the tree, i.e., the path should traverse not less than $(2^{|K|\cdot m-1} + 1)$ subsets of K.

Algorithm 3. Deriving a complete test suite w.r.t. the fault model $<S, (\leq, \nsim), \Re_m>$
Input: Complete FSM S and an upper bound m on the number of states of any FSM implementation of S

Output: A complete test suite TS w.r.t. the fault model $<S, (\leq, \nsim), \Re_m>$
Step 1. Derive a truncated successor tree of the specification FSM S. The root of this tree, which is at the 0th level, is the initial state s_0 of the FSM S; the nodes of the tree are labeled with subsets of states of the FSM S. Given already derived j levels of the tree, $j \geq 0$, a non-leaf (intermediate) node of the jth level labeled with a subset K of states of the FSM S, and an input i, there is an outgoing edge from this non-leaf node labeled with i to the node labeled with the i-successor of the subset K. A current node *Current*, at the kth level, $k \geq 0$, labeled with the subset K of states of S is claimed as a leaf node if the path from the root to this node has $2^{|K|\cdot m}$ nodes labeled with subsets of K and the initial state s_0 is not in K. If the initial state is in K then the node *Current* is claimed as a leaf node if the path from the root to this node traverses $(2^{|K|m-1} + 1)$ nodes labeled with subsets of K.

Step 2. Include into TS each input sequence which labels the path from the root to a leaf node in the above truncated tree.

Theorem 4. Given the specification FSM S over the input alphabet I and an integer m, Algorithm 3 returns a complete test suite w.r.t. the fault model $<S, (\leq, \nsim), \Re_m>$.

Proof. Given an implementation FSM T, consider the truncated tree $Tree_S$ of the specification FSM S and the truncated tree $Tree_{S \cap T}$ of $S \cap T$. Given a path in the $Tree_S$ to a node labeled with a subset K of states of S, the corresponding path in the tree $Tree_{S \cap T}$ leads to a node that is labeled with a subset P of states of the intersection $S \cap T$ such that the first item of each pair of P is in the set K. The number of such non-empty subsets is $2^{|K|\cdot m} - 1$. Thus, when a path of the $Tree_S$ traverses $2^{|K|\cdot m}$ nodes labeled with subsets of K the corresponding path in the tree $Tree_{S \cap T}$ traverses two nodes labeled with the same subset and can be terminated, according to Algorithm 1. When the initial state of the specification FSM S is in the set K then each subset traversed by the corresponding path in the tree $Tree_{S \cap T}$ does not contain the initial state, i.e., the number of such subsets is $2^{(|K|-1)\cdot m} - 1$. Respectively, when K contains the initial state a path can be terminated if it traverses $(2^{(|K|-1)\cdot m} + 1)$ nodes labeled with subsets of K (counting the initial state of the specification that labels the root of the $Tree_S$), since the corresponding path in the tree $Tree_{S \cap T}$ traverses two nodes labeled with the same subset or with a subset that contains the initial state of the intersection.
❑

As an example, we consider the specification FSM S in Figure 1 (left hand) and derive a complete test suite w.r.t. the fault model $<S, (\leq, \nsim), \mathfrak{R}_2>$. At Step 2 a current node labeled with the state a is claimed as a leaf node if the path from the root to this node traverses $(2^{m-1} + 1) = 3$ nodes labeled with a. A current node labeled with the state b is claimed as a leaf node if the path from the root to this node traverses $2^m = 4$ nodes labeled with b. Finally, a current node labeled with the subset $\{a, b\}$ is claimed as a leaf node if the path from the root to this node traverses $(2^{2m-1} + 1) = 9$ nodes labeled with a, b and $\{a, b\}$. A complete test suite has the total length 277 (Figure 4).

Here we notice that Algorithm 3 does not return a shortest test suite. Consider, for example, a test case *xyyyyyyy* of the above test suite and the corresponding path of the truncated successor tree $Tree_S$: $a_x a_y b_y \{a,b\}_y \{a,b\}_y \{a,b\}_y \{a,b\}_y \{a,b\}_y \{a,b\}$. By direct inspection, one can assure that if an implementation FSM has states 1 and 2 then the corresponding path in the truncated tree $Tree_{S \cap T}$ will be already terminated after $\{a1\}_x \{a2\}_y \{b1,b2\}_y \{b1\}_y \{b2\}_y \{a,b\}_y$. By using such analyzing, a complete test suite with total length 89 can be derived for the fault model $<S, (\leq, \nsim), \mathfrak{R}_2>$. Thus, more analysis of termination rules is needed for reducing the length of obtained test suites. Here we recall (Section 3) that a complete test suite of length 11 is derived using the SC-method w.r.t. the fault model $<S, \leq, \mathfrak{R}_2>$ under the assumption of «all weather conditions». According to this condition, each sequence of the test suite should be applied at least eight times to a given IUT since, on average, there are eight different output responses to a test case. Thus, the total length of a test suite complete w.r.t. the fault model $<S, \leq, \mathfrak{R}_2>$ is around 100 and this test suite still does not guarantee the detection of all implementations with up to 2 states, that are separable from the given specification FSM if we lack the necessary controllability and/or observability over an IUT.

However, more rigorous analysis is necessary in order to refine termination rules, since in general, the exponential bound on the length of a test case cannot be reduced [23].

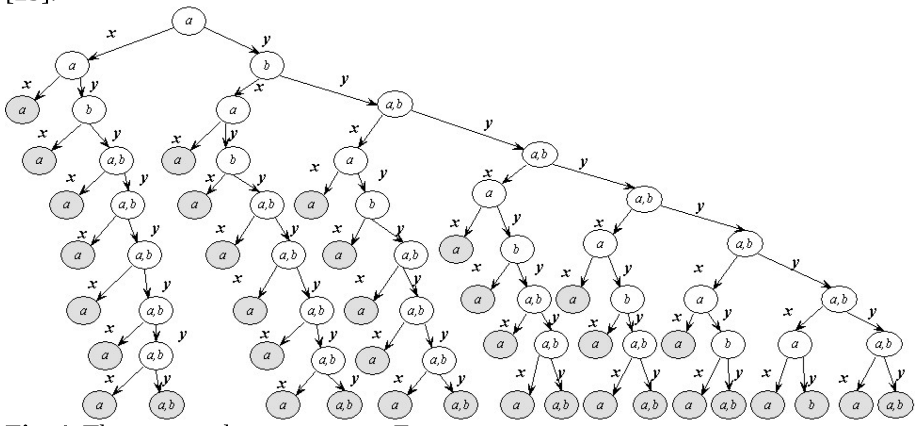
Fig. 4. The truncated successor tree $Tree_S$

6. Conclusion and Future Research Work

In this paper, we have proposed a method for the test derivation against nondeterministic FSMs with respect to the separability relation. This relation can be used without assuming that an implementation under test should satisfy the «all weather conditions» assumption. Refined notions of a fault model and a complete test suite are given. A test suite is called complete up to the separability relation if it detects every implementation that is separable from (i.e., is not a reduction of) the given FSM specification. This complete fault coverage is guaranteed if each test case of the test suite is applied to an IUT only once. The test suite can also detect some implementations that are not reductions of the specification FSM but are non-separable from the specification FSM.

Two algorithms are presented for complete test suite derivation with respect to the separability relation. The first algorithm can be applied when the set of possible implementations is the set of all complete nondeterministic submachines of a given mutation machine. The second algorithm is applied when the upper bound on the number of states of an IUT is known. The proposed algorithms do not return shortest test suites and more work is needed for reducing the length of obtained test suites. Unfortunately, the exponential upper bound on the length of a test case cannot be reduced [23], except of the case when the specification is deterministic or we consider only deterministic implementations. For simplicity of presentation, in this paper, we assume that the specification FSM is complete but the proposed algorithms do not rely on this assumption and thus, can be extended to partial specification FSMs.

References

1. Kloosterman, H.: Test derivation from non-deterministic finite state machines. Proceedings of the IFIP Fifth International Workshop on Protocol Test Systems, Canada, (1992) 297–308.
2. Tripathy, P., Naik, K.: Generation of adaptive test cases from nondeterministic Finite State models. IFIP Trans. C: Commun. System C-11, (1993) 309–320.
3. Luo, G., Petrenko, A., Bochmann, G. v.: Selecting test sequences for partially specified nondeterministic finite state machines. Proc. 7th International Workshop on Protocol Test Systems, 1994.
4. Luo, G., Bochmann, G. v., Petrenko, A.: Test selection based on communicating non-deterministic finite-state machines using a generalized Wp-method. IEEE Transactions on Software Engineering, 20(2), (1994) 149–161.
5. Alur, R. Courcoubetis, C., Yannakakis, M.: Distinguishing tests for nondeterministic and probabilistic machines. Proc. the 27th ACM Symposium on Theory of Computing (1995) 363–372.
6. Petrenko, A., Yevtushenko, N., Bochmann, G. v.: Testing deterministic implementations from their nondeterministic specifications. Proc. 9th International Workshop on Protocol Test Systems, (1996) 125–140.

7. Boroday, S. Yu.: Distinguishing Tests for Non-Deterministic Finite State Machines. Proc. IFIP TC6 11th International Workshop on Testing of Communicating Systems (1998) 101–107.
8. Hierons, R. M.: Adaptive testing of a deterministic implementation against a nondeterministic finite state machine. The Computer Journal, 41(5), (1998) 349–355.
9. Koufareva, I. Evtushenko, N., Petrenko, A.: Design of tests for nondeterministic machines with respect to reduction. Automatic Control and Computer Sciences, USA, 3 (1998).
10. Hierons, R. M.: Using candidates to test a deterministic implementation against a non-deterministic finite state machine, The Computer Journal, 46(3), (2003) 307–318.
11. Hierons, R. M.: Testing from a non-deterministic finite state machine using adaptive state counting. IEEE Transactions on Computers, 53(10), (2004) 1330–1342.
12. Hierons R. M., Ural, H.: Concerning the ordering of adaptive test sequences, Proc. 23rd IFIP International Conference on Formal Techniques for Networked and Distributed Systems (Lecture Notes in Computer Science, vol. 2767), Springer, (2003) 289-302.
13. Hwang, I., Kim, T., Hong, S., Lee, J.: Test selection for a nondeterministic FSM. Computer Communications, 24 (2001) 1213–1223.
14. Zhang, F., Cheung, T.: Optimal transfer trees and distinguishing trees for testing observable nondeterministic finite-state machines. IEEE Transactions on Software Engineering, 29(1), (2003) 1–14.
15. Petrenko, A., Yevtushenko, N.: Conformance tests as checking experiments for partial nondeterministic FSM. Proc. 5th International Workshop on Formal Approaches to Testing of Software (2005).
16. Miller, R., Chen, D., Lee, D., Hao, R.: Coping with nondeterminism in network protocol testing. In Proceedings of the 17th IFIP International Conference on Testing of Communicating Systems, USA (2005).
17. Tanenbaum, A. S. Computer Networks. Prentice-Hall, NJ (1996).
18. Petrenko, A., Yevtushenko, N.: Testing from partial deterministic FSM specifications. IEEE Trans. on Computers, 54(9), (2005) 1154–1165.
19. Milner. R.: A Calculus of Communicating Systems. Lecture Notes in Computer Science, vol 92 (1980).
20. Spitsyna, N., Trenkaev, V., El-Fakih, K., Yevtushenko, N.: FSM interoperability testing, Work In Progress: 23rd International Conference on Formal Techniques for Networked and Distributed Systems (2003).
21. Spitsyna, N.: FSM-based test suite derivation strategies for discrete event systems. Ph.D. Thesis, Tomsk State University, 1–158 (2005).
22. Starke, P.: Abstract automata, American Elsevier, 3–419 (1972).
23. Spitsyna, N., El-Fakih, K., Yevtushenko, N.: Studying the Separability Relation between Finite State Machines. Submitted to Software Testing, Verification and Reliability (2006).