Deriving complete finite tests based on state machines

Igor Burdonov\textsuperscript{1}, Alexander Kossatchev\textsuperscript{1}, Nina Yevtushenko\textsuperscript{2}
\textsuperscript{1}Institute for System Programming of the RAS, Moscow, Russia, igor@ispras.ru, kos@ispras.ru.
\textsuperscript{2}Tomsk State University, Tomsk, Russia yevtushenko@sibmail.com

Abstract

Many state machine based strategies return complete but infinite test suites. A usual approach to guarantee the fault coverage with respect to some kind of faults is to limit the number of faults, i.e., to consider a finite fault domain. In this paper, we summarize some results on deriving complete test suites w.r.t. infinite faults domains but w.r.t. special types of the specification machine.

Many state machine based strategies return complete but infinite test suites. One solution for getting complete finite tests is to limit the number of faults, i.e., to consider a finite fault domain. In this paper, we summarize some results on deriving complete test suites w.r.t. infinite fault domains against proper types of the specification machines.

1. Introduction

State Models (State Machines) are widely used for deriving tests with the guaranteed fault coverage in various application domains [1-4]. Input sequences are applied to an Implementation Under Test (IUT) during testing and based on observed outputs the verdict ‘fail’ is drawn when there is some discrepancy between an expected output and the output produced by the IUT. Correspondingly, when talking about tests with the guaranteed fault coverage, the notion of a fault model is introduced that is a triple $<$Spec, $\sim$, FD$>$ [5] where Spec describes the reference behavior, FD is the fault domain that contains each machine that describes the behavior of a possibly faulty implementation system and $\sim$ is a conformance relation. The behavior of an Implementation Under Test (IUT) is usually described by the same state model as the specification. A test suite is complete (w.r.t. to the given fault model) when for each conforming IUT there is the verdict ‘pass’ while for each non-conforming IUT there is the verdict ‘fail’.

In order to derive a complete finite test suite w.r.t. a given set of faults, i.e., w.r.t. a fault domain, test engineers/researchers limit the fault domain to be finite [see, for example, 6]. In this case, a distinguishing test case can be derived for each pair ‘specification_IUT’ and a test suite contains all necessary test cases. In this paper, we show that there exist special classes of state machines for which a complete finite test suite can be derived without limiting the infiniteness of the fault domain, i.e., an IUT can be any machine that has no cycles labeled by actions which cannot be externally observed. We discuss two such specifications and corresponding conformance relations for state models; namely, for a finite automaton where each transition is labeled by an input, an output or by a unobservable action and for a Finite State Machine where each transition is labeled with the input-output pair.

The rest of the paper is organized as follows. In Section 2, it is shown that for special kinds of Input/Output automata, there exist finite complete test suites under minimal limitations for possible implementations. Section 3 is devoted for deriving finite complete test suites for Finite State Machines, and Section 4 concludes the paper.

2. Deriving finite test suites for Input/Output automata

An Input/Output Automation (IOA) $S$ is a 5-tuple $(S, s_0, I, O, h)$, where $S$ is a finite set of states with the initial state $s_0$; $I$ and $O$ are finite non-empty disjoint sets of inputs and outputs, respectively; $h$ is a transition relation $h \subseteq S \times (I \cup O \cup \{\tau\}) \times S$, where a 3-tuple $(s, z, s') \in h$ is a transition. If $z \in I$, a transition $(s, z, s')$ is an input-transition, if $z \in O$, a
transition \((s, z, s')\) is an output-transition; these transitions are observable. A transition \((s, τ, s')\) is unobservable.

IOA \(S\) is strongly convergent if it does not have an infinite sequence of \(τ\)-transitions, completely specified (a complete IOA) if for each pair \((s, i)\in S \times I\) there exists \(s' \in S\) such that \((s, i, s') \in h\).

A trace of \(S\) at state \(s\) is a string of inputs and outputs, which label observable transitions in a finite sequence of transitions starting at state \(s\); the empty trace is denoted by \(ε\). We use the notation \(\mu ≤ σ\) if the trace \(μ\) is a prefix of the trace \(σ\), and \(μ < σ\) when \(μ\) is a proper prefix of \(σ\). The set \(s_{-after-σ}\) is the set of all final states for the trace \(σ\) started at state \(s\). The set \(Tr(S)\) is the set of traces of \(S\) at the initial state \(s₀\). The IOA \(S\) is deterministic if there are no unobservable transitions and \(∀ σ \in Tr(S)\) it holds that \(|s₀_{-after-σ}| = 1\) [6].

A test case is a trace over alphabets \(I\) and \(O\) where the tail symbol is an output. Given a test case \(σ\) and an IUT over the same alphabets, the testing is performed as follows. Let \(μ ≤ σ\), and at some step, the tester observes the trace \(μ\) in the IUT (at the initial step \(μ = ε\)). If \(z \in I\) then this input is applied to the IUT; if \(z \in O\) then the tester gets an output for checking. The verdict after observing a produced IUT output can be drawn in an arbitrary way. When we are concerned about deriving complete test suites, the verdict should correspond to a considered conformance relation. For example, when detecting a wrong output \(z\) after a trace \(μ\) there is the verdict fail if the produced output is \(z\). If the produced output is different from \(z\) then the testing is continued.

Given an IUT \(B\), a test case is safe (w.r.t. the IUT \(B\)) if when executing the test case against \(B\), each time when the tester is waiting for an output, some output will be produced after finite number of time ticks. The latter is not always possible; for example, it can be impossible due to the IUT divergence or to the absence of output- and \(τ\)-transitions at the current IUT state.

A test suite is a set of test cases. A test suite \(T\) is safe w.r.t. the IUT \(B\) if each test case \(t \in T\) is safe.

The interaction between a tester and an IUT is specified by the following rules. 1) If the tester does not submit an input then the IUT can execute only output- and \(τ\)-transitions and the tester checks produced outputs. 2) If the tester submits an input \(i \in I\) to the IUT then the IUT can execute only one transition labeled by \(i\); however, there can exist \(τ\)-transitions before and after this input but the tester observes only a trace ‘‘\(T\)’’.

Given a strongly convergent IOA \(S = (S, s₀, I, O, h)\) and a strongly convergent complete IOA \(B = (B, h₀, I, O, h₀)\), IOA \(B\) is safely testable against \(S\), if for each \(σ \in Tr(S) \cap Tr(B)\), \(o \in O\), it holds that \(σ_o \in Tr(S)\) implies \(∀ b \in h₀_{-after-σ} \exists o' \in O\) such that \(b_{-after-o'} \neq ∅\). In the following, we consider a set of implementation IOAs such that each IUT is safely testable against \(S\) (a safety assumption). Correspondingly, IOA \(B\) is a safely conforming implementation of to \(S\), if \(B\) safely testable against \(S\) and for each \(σ \in Tr(S) \cap Tr(B)\), \(o \in O\), it holds that \(σ_o \in Tr(B)\) implies \(σ_o \in Tr(S)\).

Given the IOA specification \(S\), a trace \(σ_o \notin Tr(S)\), s.t. \(σ \in Tr(S)\) and \(o \in O\), is a test trace; \(Tr(S)\) is the set of all test traces of \(S\). By definition, if IOA \(B\) is safely testable against \(S\) then each test trace is safe for \(B\).

Theorem 1. Given IOA specification \(S\), the set \(Tr(S)\) is a complete safe test suite.

Proof. The completeness of the test suite \(Tr(S)\) is implied by the definitions of the safe conformance, test trace and test case execution. The safety of \(Tr(S)\) is implied by the safety of each test trace. \(□\)

Determinize the given specification \(S\) by the use of subset construction [6] and denote \(S_{det} = (S_{det}, \{s₀\}, I, O, h_{det})\) the obtained deterministic IOA. The largest subset \(A \subseteq S_{det}\) such that for each \(s \in A\) and for each \((s, z, s') \in h_{det}\) it holds that \(s' \in A\) and \((z \in O \Rightarrow ∀ o \in O \exists (s, o, s') \in h_{det})\) is the set of chaotic states. Delete from \(S_{det}\) all the chaotic states and transitions taking \(S_{det}\) to chaotic states and obtain the IOA \(S_{det}^*\).

Theorem 2. If the set \(Tr(S_{det}^*)\) is finite then the set \(Tr(S_{det}^*)\) is a finite complete safe test suite.

Proof. The test suite is finite since the sets \(Tr(S_{det}^*)\) and \(O\) are finite. By definition of chaotic states, each test trace does not traverse a chaotic state. Thus, the test suite has necessary features according to Theorem 1. \(□\)

In other words, Theorem 2 shows a way how to derive finite complete test suites w.r.t. the fault model where specification and implementation systems are IOAs, the conformance relation is the safe conformance and an implementation IOA is any strongly convergent complete IOA over the input and output alphabets of the specification. Therefore, the fault domain is infinite and contains each IOA that is safely testable w.r.t. the specification IOA. Similar to this, another fault models can be considered which use...
Igor Burdonov, Alexander Kossatchev, Nina Yevtushenko.
Deriving complete finite tests based on state machines.
5 стр.

the same interaction rules between the tester and IUT,
safety assumption and safe conformance. Trace
semantics, completed traces, failure semantics, failure
trace semantics, ready trace semantics, readiness
semantics, ioco-semantics can be considered [4]. For
deriving complete test suites, inputs and outputs of the
specification and implementation IOAs should be
correspondingly defined and the conformance relation
has to be correspondingly modified [4]. In particular,
a refusal set for the failure trace semantics and the
quiescence for the ioco-semantics become outputs and
label corresponding transitions. The fault domain, i.e.,
the set of all IUTs, becomes a subset of such set in the
above IOA-semantics after appropriate
transformations. Since each finite test suite
above IOA-semantics after appropriate
quiescence for the ioco-semantics become outputs and
refusal set for the failure trace semantics and the
semantics, ioco-semantics can be considered [4]. For
trace semantics, ready trace semantics, readiness
IOA where each input is followed by an output and
deriving complete test suites, inputs and outputs of the
when considering the quasi-reduction relation as the
initial state
the previous input. Moreover, an FSM has no
next input can be only applied after getting an output to
FSM is labeled with an input-output pair and thus, the
between FSM and IOA is that each transition of an
transition (S, s, I, O, h, b) where S is a finite set of states with the
initial state s0, I and O are finite non-empty disjoint
sets of inputs and outputs, respectively; h is a
transition relation h ⊆ S × I × O × S, where a 4-tuple
(s, i, o, s′) ∈ h is a transition. The main difference
between FSM and IOA is that each transition of an
FSM is labeled with an input-output pair and thus, the
next input can be only applied after getting an output to
the previous input. Moreover, an FSM has no
unobservable transitions. i.e., an FSM corresponds to a
strongly convergent IOA after unfolding transitions into
input- and output-transitions. A state s of the FSM
is output-complete if for each input i ∈ I such that
there is a transition (s, i, o, s′) ∈ h, there exists a
transition (s, i, o′, s′′) ∈ h for each o′ ∈ O. Input
sequence α ∈ I* is a defined input sequence at state s
of S if it labels a sequence of transitions starting at
state s. A trace of S at state s is a string of inputs and
outputs which label a sequence of transitions starting at
state s. Let Tr(S) or Trs denote the set of traces of S
at the initial state. Given a sequence β ∈ (IO)*, the input
(output) projection of β, denoted β↓I (β↓O), is a
sequence obtained from β by erasing symbols in O (I).

FSM S is completely specified (a complete FSM) if for
each pair (s, i) ∈ S × I there exists (o, s′) ∈ O × S
such that (s, i, o, s′) ∈ h, otherwise, the FSM is
partial. The FSM is deterministic if for each pair
(s, i) ∈ S × I there exists at most one transition
(s, i, o, s′) ∈ h; otherwise, the FSM is
nondeterministic. Nondeterministic FSM is observable
if for each two transitions (s, i, o, s1), (s, i, o, s2) ∈ h it
holds that s1 = s2. An FSM is single-input if at each
state, there is at most one defined input; output-
complete if for each pair (s, i) ∈ S × I such that the
input i is defined at the state s, there exists a transition
from s with i for every output; acyclic if Tr is finite.
Given an input sequence α defined at state s, we use
the notation out(s, α) in order to denote the set of
output sequences which can be produced by S in
response to α when applied at state s, that is
out(s, α) = {β↓O | β ∈ Tr(S α)} and β↓I = α. If an input
sequence is not defined at state s then out(s, α) is
empty. Similar to IOA, the largest subset A ⊆ S of
output-complete states such that for each s ∈ A and for
each (s, i, o, s′) ∈ h it holds that s′ ∈ A, is the set of
chaotic states.

In this paper, we consider possibly partial initially
connected observable specification machines; one
could use a standard procedure for automata
determination to convert a given FSM into
observable one. Since an FSM has no unobservable
transitions, each FSM implementation is safely testable
again the FSM specification, i.e., the safety assumption
automatically holds for FSM based fault models.
Define in terms of traces the quasi-reduction relation
between FSMs. Given an FSM S and a complete FSM
B, FSM B is a quasi-reduction of S, if
{β ∈ Tr(B) | β↓I = α} ⊆ {β ∈ Tr(S) | β↓I = α}
for each input sequence α that is defined at the initial state
of the FSM S. In fact, the quasi-reduction relation
between FSMs is very close to the ioco-relation
between IOAs when the quiescence is observable. If
the specification FSM is complete then the quasi-
reduction is reduced to the reduction relation (trace
inclusion relation), i.e., for each input sequence α it
holds that out(b0, α) ⊆ out(s0, α). In the following,
FSM S represents the specification that can be partial
while FSM B describes the behavior of an IUT that is
assumed to be complete, as usual.

Testing deterministic FSMs it is sufficient to use
input sequences as test cases [5]. Dealing with
nondeterministic machines, we may need to consider
adaptive testing, when the next input depends on a produced output or there are no more inputs to execute. An unexpected output triggers in the test case a transition to the state fail and the test execution is over. A test case over input alphabet I and output alphabet O is an acyclic single-input output-complete FSM that can have a designated deadlock state fail [7]. A test suite is a finite set of test cases. A test case is preset if all input projections of its pass traces coincide. A test suite is preset if it has only preset test cases.

In this paper, we assume that the fault domain contains each complete FSM over the input and output alphabets of the specification FSM S. Correspondingly, a test suite is complete for the specification FSM S if for each implementation FSM B that is not a quasi-reduction of S, there exists a test case that is taken to the fail state by some trace of the FSM B. Given a defined input sequence α of the FSM S, let TC(α) denote a test case where traces to the pass state are all possible traces of S with the input projection that is a prefix of α. All other outputs at each state take the test case TC(α) to the fail state [7].

Given the specification FSM S, delete all the chaotic states and denote S the obtained FSM. Let the set Tr(S) of traces be finite and TS be the test suite that contains a test case TC(α) for each longest defined input sequence α of S, i.e., each such sequence α is not a proper prefix of another defined input sequence.

**Theorem 3.** The test suite TS is complete for the specification FSM S.

In fact, let an IUT have the expected behavior for each test case of the set TS. According to the definition of S, the behavior of the IUT to any prolongation of each defined input sequence can have arbitrary outputs. On the other hand, since the set Tr(S) is finite there always exists a finite set of test cases which cover each trace of FSM S.

In some cases, when traces of the FSM are harmonized [3], i.e., given a defined input sequence α of S and two traces β₁, β₂ ∈ Tr(S), β₁↓ = β₂↓ = α, the sets of defined inputs at states of S after traces β₁ and β₂ coincide. In this case, the above theorem can be reformulated for preset test cases.

**Corollary.** If the set Tr(S) is finite and S has only harmonized traces than TS = {β↓ | β ∈ Tr(S)} is a complete preset test suite.

According to Theorem 3, for a special class of specification FSMs a complete test suite can be finite despite of the fact that the set of all possibly faulty implementations is infinite.

If an IUT can be non-deterministic then as usual, we assume that an expected behavior (if such a behavior exists) can be observed after applying each test case at most k times (‘all-weather-conditions’ assumption). In other words, in order to conclude that a given implementation FSM B passes a test case, the test case has to be applied to B a sufficient number of times ensuring that no further application will produce an unexpected output.

4. Concluding remarks

In this paper we consider limitations over specification machines such that a complete finite test suite can be derived when an Implementation Under Test can be a machine over corresponding input and output alphabets with no restrictions on the set of its observable transitions, i.e., the fault domain in infinite. Moreover, the IOA-semantics described in the paper requires the priority of a submitted signal (an input) over an observed symbol (an output), i.e., the above theory can be applied for testing systems with priorities. Additional investigation is needed how the specification can be restricted for deriving shorter test suites which still are complete w.r.t. infinite fault domains.

5. References