Evaluating the length of distinguishing sequences for nondeterministic Input/Output automata

Igor Burdonov
Software Engineering department
Ivannikov Institute for System Programming of RAS
Moscow, Russia
igor@ispras.ru

Alexandr Kossachev
Software Engineering department
Ivannikov Institute for System Programming of RAS
Moscow, Russia
igor@ispras.ru

Nina Yevtushenko
Software engineering deparment
Ivannikov Institute for System Programming
Moscow, Russia
evtushenkor@ispras.ru

Alexey Demakov
Software engineering deparment
Ivannikov Institute for System Programming
Moscow, Russia
demakov@ispras.ru

***Abstract*——Distinguishing sequences are used in model based mutation testing in order to distinguish the specification from its mutants that usually represent critical implementation faults. In this paper, we consider distinguishing sequences for Input/Output automata when a sequence of inputs can be applied before getting any response or a sequence of output responses from an implementation under test. We propose a technique for deriving an *r*-distinguishing trace, i.e. a distinguishing trace with respect to the trace inclusion (quasi-reduction) relation, and obtain the least and upper bounds on the length of a shortest *r*-distinguishing trace showing that the exponential upper bound with respect to the number of states of the specification automaton is reachable; the results are then adapted for a proper case of Input/Output automata when each input is followed by an output, i.e., for Finite State Machines.**

Keywords—Input/Output automata, quasi-reduction relation, r-distinguishing trace

I. INTRODUCTION

Test generation with guaranteed fault coverage is an important issue in developing complex critical systems (see, for example, [1]) and guaranteed fault coverage immediately asks for involving formal models. Finite transition systems are widely used for deriving tests and there are a number of methods [2] for deriving test suites with guaranteed fault coverage for Finite State Machines (FSMs) when each input is followed by an output. However, the above FSM model is not always appropriate and sequences of inputs can be applied before getting any response or a sequence of output responses; this situation can be adequately handled by the use of so-called Input/Output(I/O) automata [3]. Nevertheless, all the methods developed for such automata usually return infinite tests when talking about the 'black-box' testing model [4]. In a number of cases when critical faults could be enumerated, a test suite can be derived as a set of distinguishing sequences for specification and mutant I/O automata. In this case, a technique for deriving an appropriate preset or an adaptive distinguishing sequence for two I/O automata has to be elaborated and the complexity of a corresponding test suite has to be evaluated. For FSMs there are many publications how to derive such sequences but we are not aware of these results for I/O automata.

In this paper, we consider a variation of the well known *ioco* relation [4] but as we consider automata that not necessary are input complete we modify *ioco* as a quasi-reduction relation (similar to FSMs [5]). Automaton A is not a quasi-reduction of automaton B if there exists a trace defined at both automata such that the set of outputs after this trace of automaton A is not a subset of that of automaton B and propose a technique how to check whether this relation holds. If the automaton B is deterministic then the obtained criterion describes necessary and sufficient conditions for checking the quasi-reduction relation. However, if the automaton B is nondeterministic then the conditions become only sufficient. Moreover, as we consider automata which not necessary are input complete and we do not observe states when testing, only traces for which an input is defined at any state after a corresponding prefix are considered as distinguishing test cases and such traces are called permissible. In order to completely check the quasi-reduction relation when B has a trace that takes the automaton from the initial state to two different states, the B has to be determinized; however, the deterministic equivalent of B is a bit different from the ordinary [6] as it contains only permissible traces and in this paper, we also evaluate the length of such trace obtaining lower and upper bounds for *r*-distinguishing traces depending on the number of states of both automata.

The rest of the paper is structured as follows. As usual, Section 2 contains preliminaries while a technique for deriving a distinguishing trace together with the least bound of such trace is presented in Section 3. Section 4 shows that the exponential upper bound on length of a shortest *r*-distinguishing trace with respect to the number of states of the specification automaton is reachable if the latter can be nondeterministic; the results are adapted for FSMs in Section 5. Section 6 concludes the paper.

.

II. Preliminaries

An *I/O automaton*, simply an *automaton* throughout this paper, is a 5-tuple $\boldsymbol{S} = (V(\boldsymbol{S}), X, Y, E(\boldsymbol{S}), s_0)$ where $V(\boldsymbol{S})$ is a finite nonempty set of states with the initial state s_0, X is a finite nonempty set of inputs, Y is a finite nonempty set of outputs, $X \cap Y = \varnothing$, $E(\boldsymbol{S}) \subseteq V(\boldsymbol{S}) \times (X \cup Y) \times V(\boldsymbol{S})$ is a set of transitions. Sometimes, we refer to inputs and outputs as to *actions*. According to the above definition, we consider automata without the nonobservable action.

For $s, s' \in V(\boldsymbol{S})$ and $z \in (X \cup Y)$, we use the following notations:

$s \text{—}z\text{→} s' \overset{\text{def}}{=} (s, z, s') \in E(\boldsymbol{S})$,

$s \text{—}z\text{→} \overset{\text{def}}{=} \exists\, s' \in V(\boldsymbol{S})\ (s, z, s') \in E(\boldsymbol{S})$.

If there are no transitions at a state under outputs then we add a loop labeled with 'output' δ [4] that is not in the set Y:

$E_\delta(\boldsymbol{S}) = E(\boldsymbol{S}) \cup \{a \text{—}\delta\text{→} a \mid a \in V(\boldsymbol{S})\ \&\ \forall\, y \in Y\ \nexists\, b\ a \text{—}y\text{→} b\}$.

Such an augmented automaton $\boldsymbol{S}$ is denoted as $\boldsymbol{S}_\delta = (V(\boldsymbol{S}), X, Y, E_\delta(\boldsymbol{S}), s_0)$, and a trace in $\boldsymbol{S}_\delta$ is a *S-trace*[1] of $\boldsymbol{S}$. If the contrary not explicitly stated then a trace denotes an S-trace.

Input $x \in X$ *is a defined input* at state $s \in V(\boldsymbol{S})$ if there is a transition $s \text{—}x\text{→}$, i.e., $\exists\, s' \in V(\boldsymbol{S})\ (s, x, s') \in E(\boldsymbol{S})$. Input $x \in X$ is *defined after a trace* if this input is defined at each state reached after this trace. An automaton is *input-complete* if every input is defined at every state.

Given a trace μ and state s, μ is a *permissible* trace at state s if each input in μ is defined after the prefix that directly precedes this input. Given a permissible trace μ at state s, as usual, s-***after***-μ is the set of states where μ can take the automaton from state s. If μ is a *permissible* trace at the initial state of $\boldsymbol{S}$ then instead of s_0-***after***-μ we sometimes write $\boldsymbol{S}$-***after***-μ. In this paper, we assume that two automata can be distinguished only by a trace that is permissible at the initial states of both automata; moreover, we also assume that if after an input no outputs are expected then δ is a corresponding output.

An automaton is *observable* if at each state at most one transition is defined for each action.[2] Given an observable automaton, the set of states s-***after***-μ is either empty or is a singleton. Given a nonobservable automaton, the set s-***after***-μ can have several states.

We also use the following notation: the set of outputs at a state s is the set of defined outputs at this state: $out_{\boldsymbol{S}}(s) \overset{\text{def}}{=} \{y \in Y \mid s \text{—}y\text{→}\}$. If no outputs are defined at state s then $out_{\boldsymbol{S}}(s) \overset{\text{def}}{=} \{\delta\}$. For a subset $B \subseteq V(\boldsymbol{S})$ we have $out_{\boldsymbol{S}}(B) \overset{\text{def}}{=} \cup\{out_{\boldsymbol{S}}(s) \mid s \in B\}$.

An automaton $\boldsymbol{A} = (V(\boldsymbol{A}), X, Y, E(\boldsymbol{A}), a_0)$ is a *quasi-reduction* of the automaton $\boldsymbol{S} = (V(\boldsymbol{S}), X, Y, E(\boldsymbol{S}), s_0)$ if for each trace σ that is permissible in both automata it holds that $out_{\boldsymbol{A}}(\boldsymbol{A}\text{-}\boldsymbol{after}\text{-}\sigma) \subseteq out_{\boldsymbol{S}}(\boldsymbol{S}\text{-}\boldsymbol{after}\text{-}\sigma)$.

If the automaton $\boldsymbol{A} = (V(\boldsymbol{A}), X, Y, E(\boldsymbol{A}), a_0)$ is not a quasi-reduction of the automaton $\boldsymbol{S} = (V(\boldsymbol{S}), X, Y, E(\boldsymbol{S}), s_0)$, then there exists a trace σ that is permissible in both automata such that $out_{\boldsymbol{A}}(\boldsymbol{A}\text{-}\boldsymbol{after}\text{-}\sigma) \nsubseteq out_{\boldsymbol{S}}(\boldsymbol{S}\text{-}\boldsymbol{after}\text{-}\sigma)$. This trace σ is called an *r-distinguishing trace*.

III. The upper bound of an r-distinguishing trace case derivation

Let an automaton $\boldsymbol{A}$ be not a quasi-reduction of $\boldsymbol{S}$, and σ is an r-distinguishing trace. For a trace σ, consider sequences of pairs $(a_j, \boldsymbol{S}\text{-}\boldsymbol{after}\text{-}\sigma_j)$ where σ_j is a prefix of σ of length j, $j = 0, .., |\sigma|$, and $a_j \in (\boldsymbol{A}\text{-}\boldsymbol{after}\text{-}\sigma_j)$, i.e., a_j is a state of $\boldsymbol{A}$ reachable after trace σ_j. If σ is a shortest r-distinguishing trace with this property then there exists at least one sequence of pairs where all the pairs are pairwise different. Since the number of such pairs does not exceed the product of $n = |V(\boldsymbol{A})|$ and $2^k - 1$ where $k = |V(\boldsymbol{S})|$, the length of such sequence does not exceed $n(2^k - 1)$, and thus, the length of a shortest r-distinguishing trace is not bigger than $\boldsymbol{O}(n2^k)$.

Given an automaton $\boldsymbol{A}$ and an observable automaton $\boldsymbol{S} = (V(\boldsymbol{S}), X, Y, E(\boldsymbol{S}), s_0)$ over the same alphabets, in order to derive a set of permissible traces of both automata, the product $\boldsymbol{A} \cap \boldsymbol{S}$ of automata can be constructed. States of the product are pairs of states of the automata, a transition is defined at a state if it is defined at both states.

Proposition 1. Given an automaton $\boldsymbol{A}$ and an observable automaton $\boldsymbol{S}$ over the same alphabets, $\boldsymbol{A}$ is not a quasi-reduction of $\boldsymbol{S}$ if and only if the product $\boldsymbol{A} \cap \boldsymbol{S}$ has a state (a, s), $a \in V(\boldsymbol{A})$, $s \in V(\boldsymbol{S})$, such that the state is reachable from the initial state via a permissible trace at the initial states of both automata and some output is defined at state a while not being defined at state s.

Indeed, let (a, s) be a state with the above features reachable from the initial state via a trace μ. Since the automaton $\boldsymbol{S}$ is observable, s is the only state of the automaton $\boldsymbol{S}$ reachable by μ and the latter immediately implies $out_{\boldsymbol{A}}(\boldsymbol{A}\text{-}\boldsymbol{after}\text{-}\sigma) \nsubseteq out_{\boldsymbol{S}}(\boldsymbol{S}\text{-}\boldsymbol{after}\text{-}\sigma)$. On the other hand, if each permissible trace at the initial states of both automata takes the product $\boldsymbol{A} \cap \boldsymbol{S}$ to a state (a, s) such the set of outputs at state a is a subset of that at state s then by definition, $\boldsymbol{A}$ is a quasi-reduction of $\boldsymbol{S}$.

It is well known how to construct the product of two automata over the same alphabet of actions and thus, Proposition 1 provides necessary and sufficient conditions for checking whether one automaton is a quasi-reduction of another observable automaton. If the product has a state (a, s) with the above features then a permissible trace μ that takes the product to this state is an r-distinguishing trace. Given an automaton $\boldsymbol{A}$ with n states and an observable automaton $\boldsymbol{S}$ with k states, the product $\boldsymbol{A} \cap \boldsymbol{S}$ has at most nk states, and thus, length of a shortest r-distinguishing trace is not bigger than $\mathbf{O}(nk)$.

[1]*Suspension trace*

[2]Sometimes such an automaton is called deterministic [6]. However, we save this notion for an observable automaton where at each state at most one output is defined.

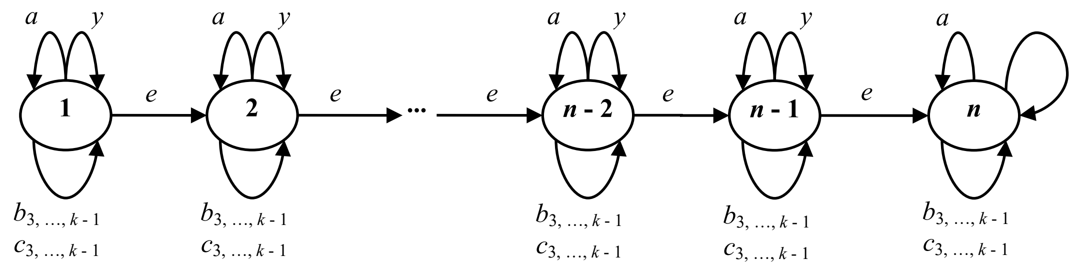

Fig. 1. Automaton $\boldsymbol{A}_n$, $n \geq 2$

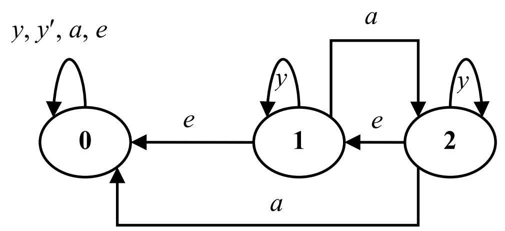

Fig. 2. Automaton $\boldsymbol{S}_3$

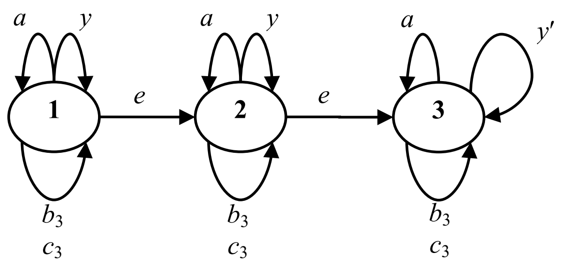

Fig. 3. Automaton $\boldsymbol{A}_3$

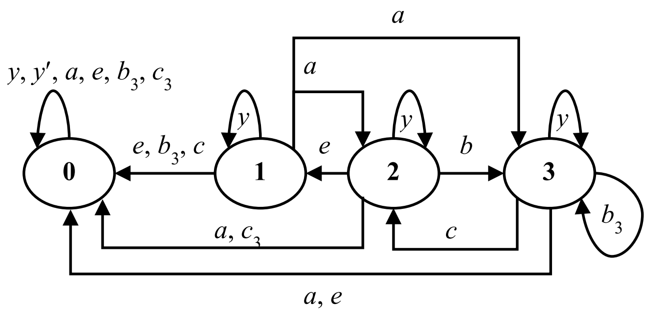

Fig. 4. Automaton $\boldsymbol{S}_4$

If automaton $\boldsymbol{S} = (V(\boldsymbol{S}), X, Y, E(\boldsymbol{S}), s_0)$ is not observable, we define a power-automaton $\mathcal{P}(\boldsymbol{S})$ over the same alphabets X and Y. We use non-empty subsets of the set $V(\boldsymbol{S})$ as states of $\mathcal{P}(\boldsymbol{S})$, i.e., $V(\mathcal{P}(\boldsymbol{S})) = 2^{V(\boldsymbol{S})} \setminus \{\varnothing\}$, and the initial state of $\mathcal{P}(\boldsymbol{S})$ is $\{s_0\}$. In the automaton $\mathcal{P}(\boldsymbol{S})$, there is a transition $A \text{—}x\text{→} B$ under input $x \in X$ if and only if x is a defined input at each state $a \in A$ of $\boldsymbol{S}$, and $B = \{b \mid \exists\, a \in A\ a \text{—}x\text{→} b\}$. In $\mathcal{P}(\boldsymbol{S})$, a transition $A \text{—}y\text{→} C$ is defined under output $y \in Y$ if and only if in $\boldsymbol{S}$, a transition under this output is defined at least at one state $a \in A$, and $C = \{c \mid \exists\, a \in A\ a \text{—}y\text{→} c\}$.

Proposition 1′. Given automata $\boldsymbol{A}$ and $\boldsymbol{S} = (V(\boldsymbol{S}), X, Y, E(\boldsymbol{S}), s_0)$ over the same alphabets, $\boldsymbol{A}$ is not a quasi-reduction of $\boldsymbol{S}$ if and only if the product $\boldsymbol{A} \cap \mathcal{P}(\boldsymbol{S})$ has a state $(a, \check{s})$, $a \in V(\boldsymbol{A})$, $\check{s} \in V(\mathcal{P}(\boldsymbol{S}))$, such that the state is reachable from the initial state via a permissible trace at the initial states of both automata and some output is defined at state a while not being defined at the power-state $\check{s}$.

Propositions 1 and 1′ show the way how an r-distinguishing trace can be constructed if automaton $\boldsymbol{A}$ is not quasi-reduction of $\boldsymbol{S}$.

In the next section, we show that for every $k \geq 2$ and $n \geq 1$, there exist an automaton $\boldsymbol{S}_k$ with k states, $(2k - 2)$ inputs and two outputs and an input-complete automaton $\boldsymbol{A}_n$ with n states over the same input and output alphabets that is not a quasi-reduction of $\boldsymbol{S}_k$ such that a shortest r-distinguishing trace has the length $(n - 1)2^{k - 2} = \boldsymbol{\Omega}(n2^k)$.

IV. The Lower Bound of an r-Distinguishing Trace

Theorem 2. For every $k \geq 3$ and $n \geq 1$, there exist an input-complete automaton $\boldsymbol{S}_k$ with k states, $(2k - 2)$ inputs and two outputs and an input-complete automaton $\boldsymbol{A}_n$ with n states over the same input and output alphabets that is not a reduction of $\boldsymbol{S}_k$, such that a shortest r-distinguishing trace has the length $(n - 1)2^{k - 2} = \boldsymbol{\Omega}(n2^k)$.

Sketch of the proof. In order to prove the statement, we derive automata $\boldsymbol{S}_k$ and $\boldsymbol{A}_n$ for which the bound is reachable. The input alphabet $X = \{a, e, b_3, …, b_{k-1}, c_3, …, c_{k-1}\}$ has $2(k – 2)$ inputs while the output alphabet has two outputs, $Y = \{y, y'\}$. The set $V(\boldsymbol{S}_k)$ of states of $\boldsymbol{S}_k$ is the set $\{0, 1, 2,…, k - 1\}$ and state 1 is the initial state.

Automaton $\boldsymbol{S}_k$ has the following transitions:

State 0: There are transitions under all inputs and all outputs to state 0;

State 1: There is a transition under input a to each state of the set $\{2, …, k - 1\}$; for each input e, b_3, …, b_{k-1}, c_3, …, c_{k-1}, there is a transition to state 0 while there is a transition to state 1 under output y;

State 2: There is a transition under input e to state 1; for each input b_j, $j = 3, …, k - 1$, there is a transition from state 2 to state j under input b_j; for each action a, c_j, $j = 3, …, k - 1$, there is a transition from state 2 to state 0 while there is a transition to state 2 under output y;

State j, j = 3, …, k- 1: there are transitions to state j under inputs $b_3,\ldots, b_j, c_3,\ldots, c_{j-1}$ and transitions to states 2, …., (j – 1) under input c_j; there are transitions to state 0 under inputs $b_{j+1}, \ldots, b_{k-1}, c_{j+1},\ldots, c_{k-1}$ to state 0; there is a transition to state 0 under inputs a and e and a transition to state j under output y.

Automaton A_n is shown in Fig. 1. Automata $\boldsymbol{S}_3$, $\boldsymbol{A}_3$, $\boldsymbol{S}_4$ are shown in Figs. 2, 3, 4.

By direct inspection, one can assure that the automaton $\boldsymbol{A}_1$ is distinguishable from $\boldsymbol{S}_k$. $k \geq 3$, with the empty trace of length 0 and a shortest r-distinguishing trace for automata $\boldsymbol{A}_3$ and $\boldsymbol{S}_3$ is a trace $aeae$ of length 4 = $(3 - 1)2^{3-2}$.

We first establish several statements about properties of automata $\boldsymbol{A}_n$ and $\boldsymbol{S}_k$.

Proposition 3. A trace σ is an r-distinguishing trace of $\boldsymbol{A}_n$ with respect to $\boldsymbol{S}_k$ if and only if this trace takes an power-automaton $\mathcal{P}(\boldsymbol{S}_k)$ from the initial state to any power-state without state 0 while taking the automaton $\boldsymbol{A}_n$ to state n.

Indeed, according to automata definitions, output y' can be produced at any power-state with state 0 and only at such power-state.

Proposition 4. Given a state j of $\boldsymbol{A}_n$, $j < n$, a trace $\gamma \in \{a, b_3, \ldots, b_{k-1}, c_3, \ldots, c_{k-1}\}^*e$ takes the automaton $\boldsymbol{A}_n$ from state j to state $j + 1$, $j = 1, \ldots, n - 1$, while taking the automaton from state n to state n.

The proof is a corollary to the fact that by definition, given a state j of $\boldsymbol{A}_n$, $j \leq n$, a trace $\gamma \in \{a, b_3, \ldots, b_{k-1}, c_3, \ldots, c_{k-1}\}^*$ leaves the automaton at state j.

Due to the definition of the automaton $\boldsymbol{S}_k$, the following statement holds.

Proposition 5. Given automaton $\boldsymbol{S}_k$ and a trace $\gamma \in \{b_3, \ldots, b_{k-2}, c_3, \ldots, c_{k-2}\}^*$ that takes the power-automaton $\mathcal{P}(\boldsymbol{S}_k)$ from a power-state $\{2, \ldots, k - 2\}$, $k \geq 4$, to the power-state $\{2\}$ traversing power-states $D_1, \ldots, D_{|\gamma|}$ without state 0, the trace γ takes the power-automaton $\mathcal{P}(\boldsymbol{S}_k)$ from the power-state $\{2, \ldots, k - 2, k - 1\}$ through the power-states $D_1 \cup \{k - 1\}, \ldots, D_{|\gamma|} \cup \{k - 1\}$ each of which has state $(k - 1)$.

By definition, a trace γ is empty in Proposition 5 when $k = 4$.

We first consider the case of $n \geq 2$ and $k = 3$. In this case, the automaton $\boldsymbol{S}_3$ has only inputs a and e and by direct inspection, one can assure that a trace ae takes the automaton $\boldsymbol{S}_3$ from state 1 to state 1 while taking the automaton $\boldsymbol{A}_n$ from every state $j \neq n$ to state $j + 1$, i.e., the trace $(ae)^{n-1}$ takes the automaton $\boldsymbol{S}_3$ from state 1 to state 1 while taking the automaton $\boldsymbol{A}_n$ from 1 to state n, and thus, is a r-distinguishing trace for $\boldsymbol{A}_n$ and $\boldsymbol{S}_3$. An input e after trace of the set $(ae)^*$ takes the automaton $\boldsymbol{S}_3$ from state 1 to state 0 and input a after any trace of the set $(ae)^*e$ takes the automaton $\boldsymbol{S}_3$ from state 2 to state 0; therefore, $(ae)^{n-1}$ is a shortest r-distinguishing trace for $\boldsymbol{A}_n$ and $\boldsymbol{S}_3$. This trace has length $(n - 1)2^{k-2}$.

Let $n \geq 2$ and $k \geq 4$. We now use the induction on k in order to show that there is a trace of length 2^{k-2} that takes $\boldsymbol{S}_k$ from state 1 to state 1.

Induction base. If $k = 4$ then the trace a, b_3, c_3, e possesses the feature while traversing the following power-states: $\{1\} - a - \{2, 3\} - b_3 - \{3\} - c_3 - \{2\} - e - \{1\}$. At any power-state of the trace, any other input takes the automaton to state 0 or to a power-state already traversed by the trace.

Induction assumption. Let for some $k < m$ hold that a trace $a\gamma$, $\gamma \in \{b_3, \ldots, b_{k-2}, c_3, \ldots, c_{k-2}\}$, takes the power-automaton $\mathcal{P}(\boldsymbol{S}_k)$ from power-state $\{1\}$ to $\{2, \ldots, k - 2\}$ and from power-state $\{2, \ldots, k - 2\}$ to $\{2\}$ traversing power-states $D_1, \ldots, D_{|\gamma|}$ without state 0 and length of this trace is $2^{k-2} - 1$, i.e., the trace $a\gamma e$ takes the power-automaton $\mathcal{P}(\boldsymbol{S}_k)$ from power-state $\{1\}$ to $\{1\}$. We append the trace γ with $b_{k-1}c_{k-1}$, i.e., the trace $a\gamma b_{k-1}c_{k-1}$ of the power-automaton $\mathcal{P}(\boldsymbol{S}_k)$ traverses power-states $\{2, \ldots, k - 1\}$, $D_1 \cup \{k - 1\}, \ldots, D_{|\gamma|} \cup \{k - 1\}$, $\{k - 1\}$, $\{2, \ldots, k - 2\}$ from state $\{1\}$ (Proposition 5). Correspondingly, the trace $a\gamma b_{k-1}c_{k-1}\gamma e$ takes the power-automaton $\mathcal{P}(\boldsymbol{S}_k)$ from power-state $\{1\}$ to $\{1\}$ while taking automaton $\boldsymbol{A}_n$ to state 2 (Proposition 4).

Therefore, the trace $(a\gamma b_{k-1}c_{k-1}\gamma e)^{n-1}$ takes the power-automaton $\mathcal{P}(\boldsymbol{S}_k)$ from power-state $\{1\}$ to $\{1\}$ while taking automaton $\boldsymbol{A}_n$ to state n, and due to Proposition 3, this proves the theorem statement.

V. Evaluating length of an r-distinguishing trace for Finite State Machines

The notion of a Finite State Machine (FSM) is very close to the notion of an I/O automaton. In fact, an FSM correspond to an I/O automaton where only inputs or outputs are defined at each state and each input is followed exactly by a sequence of outputs of length 1. Therefore, there are no races between inputs and outputs in FSMs and this fact makes this model very attractive for deriving test suites.

Formally, an initialized FSM is a 5-tuple $\mathsf{S} = (S, X, Y, h_{\mathsf{S}}, s_0)$ [7] where S is a finite non-empty set of states with the designated initial state s_0, X and Y are input and output alphabets, and $h_{\mathsf{S}} \subseteq S \times X \times Y \times S$ is the transition (behavior) relation. A transition (s, x, y, s') describes the situation when an input x is applied to S at the current state s. In this case, the FSM moves to state s′ and produces the output (response) y. FSM S is nondeterministic [8] if for some pair $(s, x) \in S \times X$, there can exist several pairs $(y, s') \in Y \times S$ such that $(s, x, y, s') \in h_{\mathsf{S}}$; otherwise, the FSM is deterministic. FSM S is observable if for every two transitions (s, x, y, s_1), (s, x, y, s') $\in h_{\mathsf{S}}$ it holds that $s_1 = s_2$; otherwise, the FSM is nonobservable.

FSM S is *complete* if for each pair $(s, x) \in S \times X$ there exists $(y, s') \in Y \times S$ such that $(s, x, y, s') \in h_{\mathsf{S}}$; otherwise, the FSM is *partial*. Given state $s \in S$ and an input $x \in X$, an input x is a *defined* input at state s if there exists $(y, s') \in Y \times S$ such that $(s, x, y, s') \in h_{\mathsf{S}}$. Given an input sequence $\alpha = x_1x_2\ldots x_k \in X^*$, α is a *defined input sequence* at state s if x_1 is a defined input at state s and for each $j = 2, \ldots, k$, x_j is a defined input at any state where input sequence $x_1x_2\ldots x_{j-1}$ can take FSM S from state s.

In usual way, the behavior relation is extended to input and output sequences. Given states s, $s' \in S$, a defined input sequence $\alpha = x_1x_2 \ldots x_k \in X^*$ at state s and an output sequence $\beta = y_1y_2 \ldots y_k \in Y^*$, there is a transition $(s, \alpha, \beta, s') \in h_\mathsf{S}$ if α is a defined input sequence at state s and there exist states $s_1 = s, s_2, \ldots, s_k, s_{k+1} = s'$ such that $(s_{j\text{-}1}, x_j, y_j, s_j) \in h_\mathsf{S}$, $j = 1, \ldots, k$. In this case, the input sequence α can take (or simply takes) the FSM S from state s to state s'. The set $out_\mathsf{S}(s, \alpha)$ denotes the set of all output sequences (responses) that the FSM S can produce at state s in response to a defined input sequence α, i.e. $out_\mathsf{S}(s, \alpha) = \{\beta: \exists\, s' \in S\ [(s, \alpha, \beta, s') \in h_\mathsf{S}]\}$. The pair $\alpha \circ \beta$, $\beta \in out_\mathsf{S}(s, \alpha)$, is an *Input/Output (I/O) sequence* at state s; if s is the initial state s_0 then the pair α/β is an Input/Output (I/O) sequence (or a *trace*) of the FSM S. Given states s and s', the I/O sequence α/β can take (or simply takes) the FSM S from state s to state s' if $(s, \alpha, \beta, s') \in h_\mathsf{S}$. Given FSMs $\mathsf{S} = (S, X, Y, h_\mathsf{S}, s_0)$ and $\mathsf{P} = (P, X, Y, h_\mathsf{P}, p_0)$, the *intersection* (or the *product*) $\mathsf{S} \cap \mathsf{P}$ is the largest connected submachine of FSM = $(S \times P, X, Y, f, s_0p_0)$ where $(sp, x, y, s'p') \in f \Leftrightarrow (s, x, y, p') \in h_\mathsf{S}$ & $(p, x, y, p') \in h_\mathsf{P}$. The set *successor*$(s, \alpha \circ \beta)$ denotes the set of all states reachable from state s after applying the defined input sequence α when getting the output response β, i.e., given a defined input sequence α at state s, *successor*$(s, \alpha \circ \beta) = \{s' : (s, \alpha, \beta, s') \in h_\mathsf{S}\}$.

Given FSMs S and P, FSM P is a *quasi-reduction* of S if for each input sequence α defined at the initial states of FSMs S and P, it holds that $out_\mathsf{P}(p_0, \alpha) \subseteq out_\mathsf{S}(s_0, \alpha)$; otherwise, if there exists input sequence α defined at the initial states of FSMs S and P such that $out_\mathsf{P}(p_0, \alpha) \nsubseteq out_\mathsf{S}(s_0, \alpha)$, then P is not a quasi-reduction of S and α is a *r-distinguishing* (input) sequence. If both machines S and P are complete then the quasi-reduction relation reduces to the reduction relation: FSM P is a *reduction* of S if and only if for each input sequence α, it holds that $out_\mathsf{P}(p_0, \alpha) \subseteq out_\mathsf{S}(s_0, \alpha)$. In [8], it is shown that for two complete observable FSMs P with $n \geq 1$ states and S with $k \geq 1$ states, length of a shortest r-distinguishing sequence does not exceed nk and this bound is reachable for machines with a single input when n and k are relatively prime integers. If FSM S is not observable then an r-distinguishing sequence has length at most $n2^k$ but the reachability for this upper bound was not proven. Converting machines $\boldsymbol{A}_n$ and $\boldsymbol{S}_k$ from Section 4 into FSMs by replacing at each state every input by the pair input/output for the output defined at the state, the following statement can be established.

Theorem 6. For every $k \geq 3$ and $n \geq 1$, there exist complete FSMs S_k with k states, $2(k - 2)$ inputs and two outputs, and a complete deterministic FSM A_n with n states over the same input and output alphabets that is not a reduction of S_k, such that a shortest r-distinguishing sequence has the length $(n - 1)2^{k-2} = \boldsymbol{\Omega}(n2^k)$.

VI. Conclusion

In this paper, we are concerned about the complexity of test suites with guaranteed fault coverage when critical faults are enumerated and a test suite is derived as a set of distinguishing sequences of the specification and mutant I/O automata when a sequence of inputs can be applied before getting a response or a sequence of output responses from an implementation under test. We propose a technique for deriving an r-distinguishing trace of the specification and a mutant I/O automata, i.e., a distinguishing trace with respect to the trace inclusion (quasi-reduction) relation, and obtain the least and upper bounds on the length of a shortest r-distinguishing trace showing that the exponential upper bound with respect to the number of states of the specification automaton is reachable. The results are then adapted for a proper case of I/O automata when each input is followed by an output, i.e., for Finite State Machines. As further directions of our work, we are going to study other distinguishability relations especially those when adaptive input sequences can be used.

Acknowledgment

This work is partly supported by RFBR project N 19-07-00327/19.

References

[1] Mathur, A:. Foundations of Software Testing. Addison Wesley (2008)

[2] Dorofeeva, R., El-Fakih, K., Maag, S., Cavalli, A., and Yevtushenko, N.: FSM-based conformance testing methods: A survey annotated with experimental evaluation. Inf. Software Technol., 52: 1286-1297 (2010)

[3] N. Lynch and M. Tuttle. An introduction to Input/Output automata. CWI-Quarterly, 2(3): 219-246 (1989)

[4] J. Tretmans.: A formal approach to conformance testing. The Intern. Workshop on Protocol Test Systems, 257-276 (1993)

[5] Petrenko, A., Yevtushenko, N., Lebedev, A., Das, A. Nondeterministic State Machines in Protocol Conformance Testing. The Intern. Workshop on Protocol Test Systems: 363-378 (1993)

[6] Hopcroft J.E., Motwani, R., and Ullman J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, second edition (2001)

[7] Kam, T., Villa, T., Brayton, K. R., Sangiovanni-Vincentelli, A.: Synthesis of FSMs: Functional Optimization. Springer (1997)

[8] Yevtushenko N., Petrenko A., Vetrova M.: Nondeterministic Finite State Machines: analysis and synthesis. Part 1: Relations and operations (in Russian). Publishers TSU, Tomsk (2006)