Distinguishing Transition Systems with the Nondeterministic Behavior

Igor Burdonov[0000-0001-9539-7853], Nina Yevtushenko[0000-0002-4006-1161] and Alexandre Kossachev[0000-0002-3959-7284]
Ivannikov Institute for System Programming of RAS, 25 Alexander Solzhenitsyn str., 109004, Moscow, Russia
{igor, evtushenko, kos}@ispras.ru

Abstract. Test generation is an important issue when checking functional and nonfunctional requirements for components of distributed systems and formal models are utilized in order to derive test suites with guaranteed fault coverage, i.e., test suites which detect critical component faults. Finite transition systems are often used as such formal models and there are a number of methods for deriving complete test suites for Finite State Machines (FSMs) where each input is followed by an output. However, the FSM model is not always appropriate, as sequences of inputs can be applied before obtaining any output response or a sequence of output responses from a system under test, while this situation can be adequately handled using Input/Output (I/O) automata. When critical faults are enumerated, a test suite can be derived as a set of sequences distinguishing the specification I/O automaton from each considered mutant, and thus, techniques for deriving sequences which distinguish two I/O automata have to be elaborated. There are different notions of distinguishability and in this paper, we consider a so-called (adaptive) separability relation. If two automata which possibly have the nondeterministic behavior are (adaptively) separable then they can be distinguished by applying a corresponding (adaptive) input sequence only once differently from the quasi-equivalence and quasi-reduction relations where each test case has to be applied appropriate number of times under the so-called "all weather conditions" assumption. In this paper, we introduce the notion of a (adaptive) separating sequence for two I/O automata and propose a technique for deriving such a sequence for I/O automata of a special class where at each state, transitions under only inputs or under only outputs are specified. The length of a separating sequence if it exists is also briefly evaluated.

Keywords: Input/Output automaton, (adaptive) separating sequence.

1 Introduction

Deriving test suites with guaranteed fault coverage for various kinds of reactive discrete and hybrid control systems is not possible without the use of formal models [1]. Transition systems with inputs and outputs are widely used for this purpose; such a transition system can be considered as a trace model that maps sequences of inputs

(input sequences) into sequences of outputs (output sequences). However, the requirement to have an output after each input as it happens in Finite State Machines (FSMs) [2, 3] is very strict and in order to weaken the assumption, the researchers consider the model of an Input/Output (I/O) automaton where an output can occur only after a sequence of inputs and there can be a sequence of such outputs. When deriving test suites with guaranteed fault coverage under the 'white box" testing assumption, the distinguishability notion is very important. There has to be a possibility to distinguish fault-free and faulty components, and special distinguishing sequences are used for this purpose when using the active testing. Such distinguishing sequences, sometimes called distinguishing experiments, are well studied for deterministic complete FSMs but components under test are usually only partially specified while having a nondeterministic behavior. In this paper, we consider Input/Output (I/O) automata [4], define the notion of an (adaptive) separating sequence for two automata and propose a technique for deriving such a sequence (if it exists). Differently from other conformance / distinguishability relations, if such a sequence exists then two automata can be distinguished after applying the sequence only once and thus, such sequences can be very useful for mutation testing.

The rest of the paper is structured as follows. Section 2 contains the preliminaries. In Section 3, the features of I/O automata are discussed for which an (adaptive) separating sequence can be constructed using the well known FSM based methods; the length of such sequences is briefly evaluated when they exist. The conclusion presents some avenues for the future work.

2 Preliminaries

The section has the necessary definitions for trace models and the notion of a separating sequence for I/O automata is introduced. In this paper, a finite *I/O automaton*, *automaton* for short, is a 4-tuple $\mathcal{S} = (S, s_0, I, O, h_S)$ where S is a finite nonempty set of states with the designated initial state s_0, I is a finite nonempty set of input actions while O is a finite nonempty set of output actions, $I \cap O = \varnothing$, and $h_S \subseteq S \times (I \cup O) \times S$ is a *transition relation*. There is a transition from state s to state s' under action a if and only if the triple $(s, a, s') \in h_S$. The automaton is *deterministic* if at each state, there is at most one transition under each action. An automaton can be considered as a trace model where a trace is a sequence of actions of the alphabet $I \cup O$ permissible at the initial state. When testing, only finite traces can be observed and correspondingly, we assume that the automaton has no cycles labeled only with output actions. Moreover, in order to avoid races at the automaton states, we consider automata where at each state either only inputs or only outputs are specified. In other words, in this paper, an I/O automaton is a deterministic automaton $\mathcal{S} = (S, s_0, I, O, h_S)$, where S is partitioned into three pairwise disjoint sets S_1, S_2 and S_3: at states of S_1 only transitions under input actions are defined (and there exists at least one such transition), at states of S_2 only transitions under output actions are defined (and there exists at least one such transition). At states of the set S_3 there are no defined outgoing transitions, i.e., these states are *deadlock states*. In general, any of these sets can be empty. A trace at the initial state is *complete* if the trace takes the automaton to a state where no outputs

are defined. In order to be able to observe such traces a proper "silent output" $\delta \notin I \cup O$ (quiescence) is added to the automaton [4], and thus at each state of the sets S_1 and S_3 a loop under δ is added where δ is considered as an output. Correspondingly, the automaton $\boldsymbol{S}^{\delta}$ is obtained and σ is a complete trace in $\boldsymbol{S}$ if and only if $\boldsymbol{S}^{\delta}$ has a trace $\sigma\delta$ sometimes called a δ-trace; the latter corresponds to the fact that after this trace none of outputs of O can appear. According to our assumptions, the automaton has no cycles labeled only with outputs and thus, every automaton trace is a prefix of some complete trace.

If the initial state of the automaton is in the set S_1, the input $i \in I$ is *strictly defined* at the initial state if there is a defined transition at the initial state under this input. If the initial state of the automaton is in the set S_2, the input $i \in I$ is *strictly defined* at the initial state if there is a defined transition under this input at each state reachable from the initial state under a trace where actions are labeled with outputs of O. A sequence αi of inputs is *strictly defined* if α is strictly defined at the initial state and at each state that is reachable from the initial state via a complete trace with the projection α, a transition under input i is defined.

3 Separating Input/Output Automata

Given I/O automata $\boldsymbol{S} = (S, s_0, I, O, h_S)$ and $\boldsymbol{P} = (P, p_0, I, O, h_P)$ of the considered set, $\boldsymbol{S}$ и $\boldsymbol{P}$ are called *nonseparable* if for each input sequence that is strictly defined at the initial states of $\boldsymbol{S}$ и $\boldsymbol{P}$, the sets of output projections of complete traces with the input projection α of $\boldsymbol{S}$ и $\boldsymbol{P}$ are not disjoint. Otherwise, the automata are *separable* and an input sequence that is strictly defined at the initial states of $\boldsymbol{S}$ и $\boldsymbol{P}$ such that the sets of output projections of complete traces with the input projection α of $\boldsymbol{S}$ и $\boldsymbol{P}$ do not intersect is a *separating* sequence for the automata.

When distinguishing a faulty implementation $\boldsymbol{P}$ from the specification $\boldsymbol{S}$, if $\boldsymbol{S}$ и $\boldsymbol{P}$ are separable then after applying a separating input sequence α to an automaton under experiment and observing a corresponding output response we could uniquely conclude which automaton is under experiment when the hypothesis of applying input sequences holds [6]. Before applying the next input the tester waits for an output until an appropriate timeout t is expired. In other words, the distinguishing experiment with a given automaton is performed as follows: the tester waits for an output until the timeout t expires; if a system under test produces an output then the timer is advanced from 0 and the tester waits for an output again until the timeout expires. If there is no output until the timeout t expires then we assume that the system produced the output δ. After this, the tester applies the next input (if any) under the above conditions. For an automaton of the considered class, an appropriate possibly partial and nondeterministic Finite State Machine can be derived and the technique from [7] can be used for checking the separability of derived FSMs.

Finite State machine or simply an FSM is a 5-tuple $S = \langle S, X, Y, h_S, s_0 \rangle$ where S is a finite nonempty set of states with the designated initial state s_0, X and Y are finite nonempty alphabets, $X \cap Y = \varnothing$, and $h_S \subseteq S \times X \times Y \times S$ is *a transition relation*. There is a transition from state $s \in S$ to state $s' \in S$ for an input/output pair x/y (xy) if and

only if $(s, x, y, s') \in h_S$. FSM S is *observable*, if for each two transitions (s, x, y, s'), $(s, x, y, s'') \in h_S$ it holds that $s'' = s'$. If FSM S is observable, $x \in X$ and $y \in Y$, then state s' is called the *xy-successor* of state s, if $(s, x, y, s') \in h_S$. The set of all non-empty *xy*-successors of state s for all outputs y is the *x-successor* of state s. The notions of *xy*- and *x*-successors can be defined for a pair of different states s_1 and s_2, if an input x is a specified input at each of these states. In this case, the *xy*-successor is defined as a pair of *xy*-successors of these states. If *xy*-successors of these states coincide or the *xy*-successor exists only for one state of s_1 и s_2 then the *xy*-successor of the pair $\{s_1, s_2\}$ is a corresponding singleton. The set of all non-empty *xy*-successors of the pair $\{s_1, s_2\}$ is the *x*-successor of this state pair.

In usual way, the transition relation is extended to input and output sequences. By default, for each state $s \in S$ the 4-tuple $(s, \varepsilon, \varepsilon, s)$ is in the transition relation of S where ε is the empty sequence. The extended transition relation is denoted by the same symbol h_S. A sequence of input/output pairs which can be successively traversed starting from the initial state, is called an *input/output sequence* or *a trace* of the FSM (at the initial state). An input x is a *defined* input at state s if $(s, x, y, s') \in h_S$ for some y and s'. Input sequence αx is a *defined* input sequence for the FSM if α is a defined input sequence at the initial state, and input x is defined at each state that is reachable from the initial state by a trace with the input projection α. Two FSMs over the same input and output alphabets are *separable*, if there exists an input sequence α defined for each FSM such that the sets of traces with this input projection are disjoint. Otherwise, the FSMs are *non-separable*. There are techniques how the separability relation can be checked for complete and partial, for observable and non-observable FSMs and these techniques can be used when checking whether two I/O automata are separable. Given an automaton $\boldsymbol{S}$ of the considered class, we construct possibly a nondeterministic FSM using a technique of the paper [6].

Algorithm 1 of deriving an FSM for a given automaton

Input: a deterministic I/O automata $\boldsymbol{S} = (S, s_0, I, O, h_S)$, where S is the union of three pairwise disjoint sets S_1, S_2 и S_3.

Output: FSM $\boldsymbol{M_S}$ that represents the set of traces of $\boldsymbol{S}^\delta$.

Construct FSM $\boldsymbol{M_S} = (S_1 \cup S_3, I \cup \{null\_in\}, O \cup O^2 \cup \ldots \cup O^{ns} \cup \{\delta\}, T_{MS})$, $null\_in \notin I$, with the empty transition set, i.e. $T_{MS} = \varnothing$, where ns is the maximum length of a trace labeled only with outputs in $\boldsymbol{S}$:

- for each state $s \in S_1$ such that $(s, i, s') \in T_S$, $s' \in S_1 \cup S_3$, add to T_{MS} the transition (s, i, δ, s');
- for each state $s \in S_1$, such that $(s, i, s') \in T_S$, $s' \in S_2$, add to T_{MS} the transition $(s, i, o_1 o_2 \ldots o_k, s'')$, $k \le ns$, where $s'' \in S_1 \cup S_3$ is the $o_1 o_2 \ldots o_k$-successor of state s'.

If the initial state of the automaton $\boldsymbol{S}$ is in S_2, then add to T_{MS} the transition $(s_0, null\_in, o_1 o_2 \ldots o_k, s)$, where $s \in S_1 \cup S_3$ $s \in S_1$, and s is the non-empty $o_1 o_2 \ldots o_k$-successor of state s_0. If the initial state of the automaton $\boldsymbol{S}$ is in S_3, then the FSM transition set T_{MS} is empty, and thus, the set of FSM traces has only the empty sequence ε.

□

By constructing the FSM $\boldsymbol{M_S}$, the following statements hold.

Proposition 1. Given a deterministic I/O automaton $\boldsymbol{S}$ of the considered class, if the initial state is in the set S_1, then an input sequence α is strictly defined in $\boldsymbol{S}$, if and only if α is a defined input sequence of the FSM $\boldsymbol{M_S}$. If the initial state of $\boldsymbol{S}$ is in the set S_2, then an input sequence α is strictly defined in $\boldsymbol{S}$ if and only if the sequence *null_in*α is a defined input sequence of the FSM $\boldsymbol{M_S}$.

Proposition 2. Given a deterministic I/O automaton $\boldsymbol{S}$ of the considered class, if at the initial state of the automaton is in the set S_1 then the set of traces of $\boldsymbol{S}^{\delta}$ and FSM $\boldsymbol{M_S}$ coincide. If at the initial state of the automaton is in the set S_2 then for each trace α of $\boldsymbol{S}$ there is the trace *null_in*α in $\boldsymbol{M_S}$, and vice versa.

Given two automata $\boldsymbol{S}$ and $\boldsymbol{P}$ of the considered class, corresponding FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ can be derived. As a corollary to Propositions 1 and 2, the following statement holds.

Theorem 3. Automata $\boldsymbol{S}$ and $\boldsymbol{P}$ are separable if and only if FSMs $\boldsymbol{M_S}$ и $\boldsymbol{M_P}$ are separable. Moreover, if the initial states of $\boldsymbol{S}$ and $\boldsymbol{P}$ are states of S_1 and P_1, then a sequence α is a separating sequence for $\boldsymbol{S}$ and $\boldsymbol{P}$ if and only α is a separating sequence for $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$. If the initial states of $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ are states of S_2 and P_2 then a sequence α is a separating sequence of $\boldsymbol{S}$ and $\boldsymbol{P}$, if and only *null_in*α is a separating sequence for $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$. If the initial states of $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ are states of S_1 and P_2, or S_2 and P_1, then the empty sequence is a separating sequence for automata $\boldsymbol{S}$ и $\boldsymbol{P}$.

When deriving a separating sequence for FSMs $\boldsymbol{M_S}$ и $\boldsymbol{M_P}$ we use a technique of the paper [7].

Algorithm 2 for deriving a separating sequence for two observable possibly partial FSMs

Input: two observable possibly partial FSMs $\boldsymbol{M_S}$ и $\boldsymbol{M_P}$ over input alphabet X

Output: A separating sequence for $\boldsymbol{M_S}$ и $\boldsymbol{M_P}$, if FSMs are separable, or the message «The FSMs are non-separable»

Step 1. Drive the intersection of $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$. If the intersection is complete then **Return** the message «The FSMs are non-separable».

Step 2. If the intersection of $\boldsymbol{M_S}$ и $\boldsymbol{M_P}$ is partial then derive a truncated successor tree for the initial state of the intersection. The root is labeled by the pair of the initial states; other nodes are labeled by subsets of states of the intersection. Let j levels of the tree, $j \geq 0$, are already constructed and an intermediate (non-terminal) node of the j^{th} level is labeled by a subset P of states of the intersection. There exists an outgoing edge from the node labeled with an input x to the node labeled with the set of x-successors of states of P if x is defined at each state of each pair of P. A current node *Current* at the p^{th} level, $p \geq 0$, labeled by the set P is a ***leaf*** if and only if one of the below conditions holds.

Rule 1:

There exists an input x such that for each state (s, p) of P, x is a defined input at both states s and p and the non-empty x-successors are singletons.

Rule 2:

There exists a node at the j^{th} level, $j < p$, labeled with the set R such that P contains each pair of different states of the set R.

Step 3.

If there is no leaf obtained by applying Rule 1 then the FSMs are not separable. **Return** the message «The FSMs are non-separable».

If there exists a leaf obtained by applying Rule 1, i.e. there exists an input x such that for each state (s, p) of P, x is defined at both states s and p and the non-empty successors of (s, p) are singletons, then the input sequence $\alpha\ x$ is a separating sequence for FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ where sequence α labels the path to this leaf. **Return** the sequence αx.

□

Notice that if a truncated successor tree is completely derived or a width tree search is used when using Algorithm 2 then for separable FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ a shortest separating sequence can be derived.

Therefore, the following technique for checking whether two automata are separable can be proposed.

Algorithm 3 for checking whether two automata are separable and deriving a separating sequence when they are separable

Input: Input/ Output automata $\boldsymbol{S}$ and $\boldsymbol{P}$

Output: A separating sequence α or the message «The automata $\boldsymbol{S}$ and $\boldsymbol{P}$ are non-separable»

Step 1. If the initial state of $\boldsymbol{S}(\boldsymbol{P})$ is in the set $S_1 \cup S_3$ ($P_1 \cup P_3$), while the initial state of $\boldsymbol{P}(\boldsymbol{S})$ is in S_2 (P_2), then the empty sequence separates automata $\boldsymbol{S}$ and $\boldsymbol{P}$. If the initial state of $\boldsymbol{S}(\boldsymbol{P})$ is in S_3 (P_3), while the initial state of $\boldsymbol{P}(\boldsymbol{S})$ is in S_1 (P_1), then **Return** he message «The automata are non-separable»

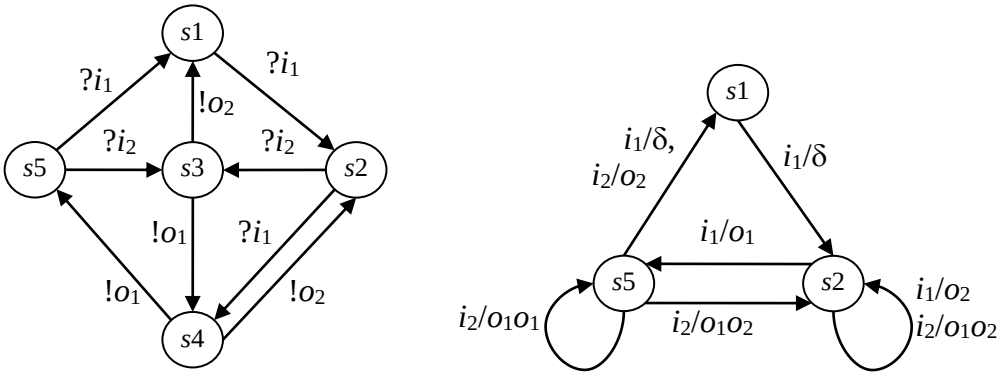
Step 2. Let the initial states of $\boldsymbol{S}$ and $\boldsymbol{P}$ be in the sets $S_1 \cup S_3$ and $P_1 \cup P_3$ or S_2 and P_2. Call Algorithm 1 to derive FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$.

Step 3. Call Algorithm 2 to check if there exists a separating sequence for FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$. If FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ are not separable then **Return** he message «The automata are non-separable».

If there exists a separating sequence for FSMs $\boldsymbol{M_S}$ и $\boldsymbol{M_P}$ then **Return** α, if α is headed by an input of alphabet X. If $\alpha = null\_in\ \beta$, then **Return** β.

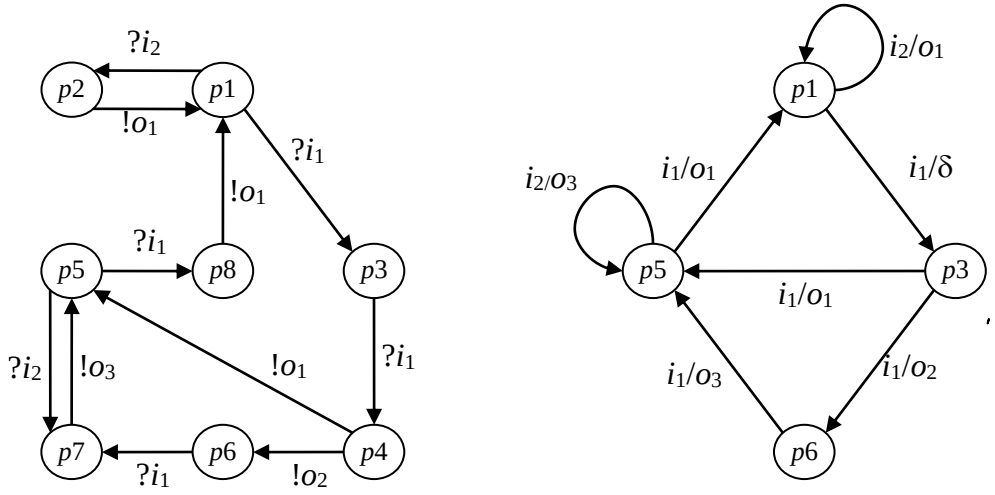
□

Example. Consider automata $\boldsymbol{S}$ and $\boldsymbol{P}$ in Figs. 1a and 2a with the initial states $s1$ and $p1$, and corresponding FSMs $\boldsymbol{M_S}$ и $\boldsymbol{M_P}$ in Figs. 1b and 2b.



(a) (b)

Fig. 1. Automaton $\boldsymbol{S}$ (a) and FSM M_S (b).



(a) (b)

Fig. 2. Automaton $\boldsymbol{P}$ (a) and FSM M_P (b).

Algorithm 2 returns a separating sequence and a frame of a corresponding successor tree is shown in Fig. 3.

For states of the pair ($s2$, $p3$) in Fig. 3, input i_2 is not defined and thus, at this state we have only a transition labeled with input i_1. States $s5$ and $p5$ and states $s2$ and $p6$ can be separated by the input i_1 that is defined at each of these states, and thus, an input sequence $i_1\ i_1\ i_1$ is a separating sequence for automata $\boldsymbol{S}$ and $\boldsymbol{P}$.

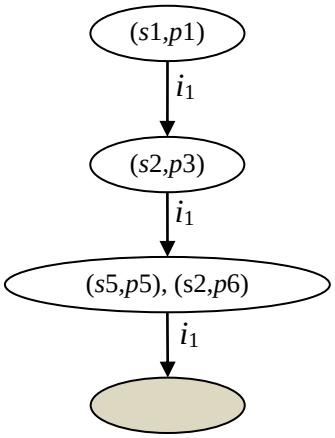


Fig. 3. A frame of a corresponding successor tree for FSMs in Figs. 1b and 2b.

Evaluating the length of a separating sequence. For complete observable FSMs the tight lower bound on the length of a shortest separating sequence with respect to the number of FSMs' states is known [8]. This bound equals 2^{mn-1} when FSMs M_S and M_P have m and n states. Correspondingly the length of a separating sequence for automata $\boldsymbol{S}$ и $\boldsymbol{P}$ which have m и n states in the sets $S_1 \cup S_3$ и $P_1 \cup P_3$ is not bigger than this value. In order to check if this bound is tight additional investigations are needed. However, in the paper [9], we show that for any $k \geq 3$ и $n > 1$, there exist deterministic input complete automata $\boldsymbol{S}_k$ with k states and $\boldsymbol{A}_n$ with *states*, $(2k - 4)$ inputs and k outputs such that the length of a shortest separating sequence equals to $(n–1)2^{k-2} + 1 = \mathbf{O}(n2^k)$.

If automata $\boldsymbol{S}$ и $\boldsymbol{P}$ have the above features but can be nondeterministic then the corresponding FSMs M_S и M_P can be non-observable. In the paper [7], a technique is proposed how to deal with non-observable FSMs when checking their separability: in this case, the notion of an *xy*-successor of a state should be modified since now it is not always a singleton. That technique is also based on using subsets of states and thus, an expected lower bound seems to coincide with that for observable FSMs; however, more research is needed in this direction.

Another interesting question is related to the adaptive separability (distinguishability). An adaptive separating sequence is represented by a so-called *test case* that is an acyclic FSM and there are techniques how such test cases can be derived [8]. Given two FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ over an input alphabet X and an output alphabet Y, a *test case* **TC**(X, Y) that represents an *adaptive* input sequence is an initially connected observable initialized FSM **TC** that has an acyclic transition graph and where at each state, at most one input is defined with all possible outputs. The *length* of the test case **TC** is the length of a longest trace from the initial state to a deadlock state and it is the length of the longest input sequence that can be applied to an FSM under investigation. A test case **TC** *a separating (distinguishing) test case* for observable FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ if (1) the initial state of **TC** is the pair of the initial states of $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$,

(2) for each trace $\gamma = \gamma' x_k y_k$ of **TC** from the initial state to a deadlock state, γ' is a trace at the initial states of $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$, (3) i_k is a defined input at the γ'-successors of the initial states of $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ and (4) every such trace γ is a trace at most at one initial state of $\boldsymbol{M_S}$ or $\boldsymbol{M_P}$. In this case, an *adaptive separating* sequence for $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ is represented by such a test case. If such a test case does not exist for FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$, then machines $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ are (adaptively) non-separable (*indistinguishable*). For two observable possibly partial machines with *n* and *m* states the length of a shortest adaptive separating sequence is at most *nm* [10]. Given automata $\boldsymbol{S}$ and $\boldsymbol{P}$ of the considered class where the sets S_1 and P_1 have *n* and *m* states correspondingly, the FSMs $\boldsymbol{M_S}$ and $\boldsymbol{M_P}$ have the same number of states and thus, the length of an adaptive separating sequence (if it exists) does not exceed *nm*. Therefore, adaptive separating sequences could be more efficient when deriving tests for complex systems under the 'white box' assumption.

4 Conclusions

In this paper, we study the (adaptive) distinguishability relation for Input/Output automata of a special class which often are used as specifications for complex control systems. When deriving tests some mutations are injected into the specification and when testing, such mutations have to be detected. When each pair "specification, mutant" has an (adaptive) separating sequence there is no need for assuming the all weather conditions and thus, each test case is applied only once. On the other hand, this affects the length of a separating sequence and in the future, we plan to study other distinguishability relations for I/O automata of the considered class as well as the extensions of this class.

References

1. Mathur, A.: Foundations of Software Testing. Addison Wesley (2008).
2. Chow, T. S.: Test design modeled by finite-state machines. IEEE Trans. SE, 4(3), 178–187 (1978).
3. Kam, T., Villa, T., Brayton, K. R., Sangiovanni-Vincentelli, A.: Synthesis of FSMs: Functional Optimization. Springer (1997).
4. Tretmans, J.: A formal approach to conformance testing. In: The Intern. Workshop on Protocol Test Systems, 257–276 (1993).
5. Starke, P.: Abstract Automata. American Elsevier (1972).
6. Kushik, N., Yevtushenko, N., Burdonov, I., Kossatchev, A.: Synchronizing and Homing Experiments for Input/output Automata. System Informatics 10, 1–10 (2017).
7. Kushik, N., Yevtushenko, N., Cavalli, A.R.: On Testing against partial nondeterministic machines. In: Intern. Conf. on the Quality of information and Communications Technology, 230–233 (2014).
8. Evtushenko, N.: Kushik. Nekotorye zadachi identifikatsii sostoianii dlia nedeterminirovannykh avtomatov. Tomsk (2018).

9. Burdonov, I., Evtushenko, N., Kosachev, A.: O razdelimosti vkhodo-vykhodnykh polu-avtomatov s nedeterminirovannym povedeniem. Russian Digital Libraries, 23 (2) (2020) (forthcoming).
10. Yenigün, H., Kushik, N., López, J., Yevtushenko, N., Cavalli, A.R.: Decreasing the complexity of deriving tests against nondeterministic finite state machines. In: Proc. of East-West Design & Test Symposium (EWDTS), IEEE Xplore, IEEE (2017), https://doi.org/10.1109/EWDTS.2017.8110091.