DOI: 10.15514/ISPRAS-2019-31(4)-12

FSM abstraction based method for deriving test suites with guaranteed fault coverage against nondeterministic Finite State Machines with timed guards and timeouts

[1] A.S. Tvardovskii, ORCID: 0000-0001-7705-7214 <tvardal@mail.ru>
[2,3] N.V. Yevtushenko, ORCID: 0000-0002-4006-1161 <evtushenko@ispras.ru>
[1] National Research Tomsk State University,
36, Lenin Ave., Tomsk, Russia 634050
[2] Ivannikov Institute for System Programming of the Russian Academy of Sciences,
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia
[3] National Research University Higher School of Economics,
3, Kochnovskiy Proezd, Moscow, 101000, Russia

Abstract. Finite State Machine (FSM) based approaches are widely used for deriving tests with guaranteed fault coverage for discrete event systems and as the behavior of many nowadays information and control systems depends on time, classical FSMs are extended by clock variables. Moreover, optionality in the real system's specifications motivates the studying test derivation against models with the nondeterministic behavior. In this paper, we adapt classical FSM based test derivation methods for nondeterministic FSMs with timed guards and timeouts (TFSMs). We show that unlike classical FSM conformance relation, the check cannot be reduced to checking the correspondence between TFSMs transitions and this violates the main principle of FSM based test derivation methods. Respectively, a proposed approach and the appropriate fault model are based on the FSM abstraction of the given TFSM specification that is used to adequately describe the behavior of a TFSM. The fault domain contains TFSMs with the known upper boundary on the number of FSM abstraction states and allows to avoid explicit enumeration of implementations under test. We study properties of the FSM abstraction for a nondeterministic TFSM and justify that the use of an FSM abstraction allows to adapt classical FSM based test derivation methods when deriving tests with guaranteed fault coverage for TFSMs. A method is proposed for deriving a complete test suite for a complete possibly nondeterministic TFSM when an implementation under test is a deterministic complete TFSM.

Keywords: Finite State Machine; timeout; timed guard; nondeterministic Timed Finite State Machine; test derivation; fault coverage

For citation: Tvardovskii A.S., Yevtushenko N.V. FSM abstraction based method for deriving test suites with guaranteed fault coverage against nondeterministic Finite State Machines with timed guards and timeouts. Trudy ISP RAN/Proc. ISP RAS, vol. 31, issue 4, 2019. pp. 175-188. DOI: 10.15514/ISPRAS-2019-31(4)-12

Acknowledgments. This work is partly supported by RFBR project № 19-07-00327/19.

Синтез тестов с гарантированной полнотой для недетерминированных автоматов с таймаутами и временными ограничениями на основе конечно автоматных абстракций

[1] А.С. Твардовский, ORCID: 0000-0001-7705-7214 <tvardal@mail.ru>
[2,3] Н.В. Евтушенко, ORCID: 0000-0002-4006-1161 <evtushenko@ispras.ru>
[1] НИУ Томский Государственный университет,
634050, Томск, пр. Ленина, 36
[2] Институт системного программирования им. В.П. Иванникова РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25
[3] Национальный исследовательский университет «Высшая школа экономики»
101000, Россия, Москва, Кочновский проезд, 3

Аннотация. Конечно-автоматные методы широко используются при синтезе проверяющих тестов с гарантированной полнотой для дискретных систем. Поскольку поведение современных информационных и управляющих систем часто зависит от времени, классическая модель конечного автомата расширяется введением временных переменных. Более того, опциональность в спецификациях реальных систем побуждает к исследованиям в области синтеза тестов для недетерминированных автоматов. В настоящей работе мы адаптируем классические конечно автоматные методы синтеза тестов к недетерминированным автоматам с временными ограничениями и таймаутами (временным автоматам). Показывается, что в отличие от классических конечных автоматов, проверка отношений конформности между временными автоматами не может быть сведена к проверке соответствия между переходами, что нарушает основной принцип конечно автоматных методов синтеза тестов. Соответственно, предложенный подход и модель неисправности основаны на конечно автоматной абстракции автомата-спецификации, которая используется для описания поведения временного автомата. Область неисправности содержит временные автоматы с известной верхней границей числа состояний конечно автоматных абстракций и позволяет избежать явного перечисления множества тестируемых реализаций. Мы исследуем свойства конечно автоматных абстракций недетерминированных временных автоматов и показываем, что использование такой абстракции позволяет адаптировать классические методы к синтезу тестов с гарантированной полнотой для временных автоматов. Предложенный метод синтеза тестов позволяет строить полные проверяющие тесты для полностью определённых возможно недетерминированных автоматов с таймаутами и временными ограничениями для тестирования реализаций, поведение которых описывается детерминированными временными автоматами.

Ключевые слова: конечный автомат; таймаут; временные ограничения; недетерминированные временные автоматы; синтез тестов с гарантированной полнотой

Для цитирования: Твардовский А.С., Евтушенко Н.В. Синтез тестов с гарантированной полнотой для недетерминированных автоматов с таймаутами и временными ограничениями на основе конечно автоматных абстракций. Труды ИСП РАН, том 31, вып. 4, 2019 г., стр. 175-188 (на английском языке). DOI: 10.15514/ISPRAS-2019-31(4)-12

Благодарности. Работа частично поддержана проектом РФФИ № 19-07-00327/19.

1. Introduction

Finite State Machines (FSMs) are widely used for analysis and synthesis of discrete event systems [1]. In particular, FSM based approaches can be effectively used when deriving test sequences for determining whether a given implementation considered as a «black box» conforms to its specification. A number of methods exist for deriving complete test suites with respect to various fault models [see, for example, 2-5] without the explicit enumeration of possible FSMs under test. Well-known W-method [2] and many its derivatives have been developed including those for FSMs with the nondeterministic behavior [4, 6]. In many papers, researchers consider the case when the specification is a nondeterministic FSM, while an implementation FSM is deterministic and conforms to the specification if the implementation behavior is contained in that of the specification

[6, 7]. In other words, the specification nondeterminism occurs according to the optionality of the informal requirements' description and the behavior of a conforming implementation must not violate the specification.

Nowadays time aspects become very important when describing the behavior of digital and hybrid control systems, and, respectively, similar to automata [8] classical FSMs were extended with clock variables [5, 9-14]. When the behavior of a system under test is described by a Timed Finite State Machine (TFSM), classical FSM-based methods have to be modified and extensions of the W-based methods are considered in the context of systems with timed constraints [9], [14]. In [11], Merayo et al. consider timed possibly nondeterministic FSMs where time elapsed when an output has to be produced after an input has been applied to an FSM under test is limited. The model also takes into account input timeouts at states. However, the authors do not consider test derivation; yet establish a number of conformance relations. El-Fakih et al. [10] consider the test derivation and assessment for FSMs with timed guards; such an FSM has a single clock that is reset at every transition. In the paper by Zhigulin et al. [13], a method is proposed for deriving complete test suites for FSMs with timeouts. The authors consider a traditional fault domain assuming that the number of states of an implementation TFSM (Implementation Under Test) does not exceed that of the state reduced specification TFSM as well as the maximal finite timeout of the IUT does not exceed that of the specification. However, as we further show, two reduced TFSMs with timeouts can be equivalent but not isomorphic and this fact violates the main idea of W-based methods of checking the isomorphism or homomorphism between the specification and implementation under test. In [12], the authors show that the behavior of a deterministic TFSM can be adequately described by its FSM abstraction and this is a hint that a fault model can be derived based on such abstraction for which well elaborated FSM based methods for deriving tests with guaranteed fault coverage can be applied. Such a fault model is considered in [15] for deriving a complete test suite against deterministic TFSMs.

In this paper, we consider FSMs with timed guards, timeouts and output delays (TFSM) which generalize the TFSM model that has only timed guards or only input timeouts [12]. Moreover, in our case, a TFSM can be nondeterministic. Timed guards describe the system behavior depending on a time instance when an input is applied. If no input is applied until an (input) timeout expires then the system can spontaneously move to another state. An output delay describes a time for processing a given transition.

We propose a method for deriving a test suite with guaranteed fault coverage against a complete possibly nondeterministic specification FSM with timed guards, input timeouts and output delays with respect to the reduction relation assuming that an implementation TFSM under test is complete and deterministic. The fault model and a procedure for deriving a complete test suite are based on the FSM abstraction of a given TFSM specification since according to [12], the behavior of a TFSM can be adequately described by its corresponding (untimed) FSM abstraction.

The structure of the paper is as follows. Section 2 contains the preliminaries for classical and timed FSMs. It also contains the explanation how the behavior of a TFSM can be described using an appropriate FSM abstraction. In Section 3, a brief sketch of related work on test derivation methods for nondeterministic FSMs with respect to the reduction relation is presented while Section 4 contains such a review on test derivation against Timed FSMs. In Section 5, a method is proposed for deriving a complete test suite against a nondeterministic FSM with timed guards and timeouts by determining an appropriate fault model based on their FSM abstractions; the section also contains an example for a test derivation procedure. Section 6 concludes the paper.

2. Preliminaries

This section contains basic definitions of classical Finite State Machines as well as of Timed Finite State Machines as their extension. We also show how the behavior of a TFSM can be adequately described by the corresponding FSM and establish some useful properties of such FSM abstractions.

2.1 Finite State Machines

A Finite State Machine (FSM) [1] describes the behavior of a system that moves from state to state under input stimuli and produces predefined output responses. Formally, an initialized FSM is a 5-tuple $\mathsf{S} = (S, I, O, h_\mathsf{S}, s_0)$ where S is a finite non-empty set of states with the designated initial state s_0, I and O are input and output alphabets, and $h_\mathsf{S} \subseteq (S \times I \times O \times S)$ is the transition (behavior) relation. A transition (s, i, o, s') describes the situation when an input i is applied to S at the current state s. In this case, the FSM moves to state s' and produces the output (response) o. FSM S is *nondeterministic* [6] if for some pair $(s, i) \in S \times I$, there exist several pairs $(o, s') \in O \times S$ such that $(s, i, o, s') \in h_\mathsf{S}$; otherwise, the FSM is *deterministic*. FSM S is *complete* [6] if for each pair $(s, i) \in S \times I$ there exists $(o, s') \in O \times S$ such that $(s, i, o, s') \in h_\mathsf{S}$; otherwise, the FSM is *partial*. FSM S is *observable* if for every two transitions (s, i, o, s_1), $(s, i, o, s_2) \in h_\mathsf{S}$ it holds that $s_1 = s_2$. In the following, we consider complete observable possibly nondeterministic FSM specifications, while an implementation is a complete deterministic FSM.

A trace or an *Input/Output sequence* α/γ, written often as an I/O sequence, of the FSM S at state s is a sequence of consecutive input/output pairs starting at the state s. Given a trace α/γ, α is the *input projection* of the trace (input sequence) while γ is the corresponding *output projection* (output sequence), i.e., a possible output response of the FSM when the sequence α is applied at state s. Given a complete nondeterministic FSM, there can exist several output responses for an input sequence at a given state. A complete nondeterministic FSM is reduced if for every two different states, the sets of traces do not coincide. The unique reduced form exists for any complete nondeterministic FSM and can be derived similar to that for complete deterministic FSMs [16].

Given states s and p of complete FSMs S and P, state p is a *reduction* of s (written, $p \leq s$) if the set of I/O sequences of FSM P at state p is contained in the set of I/O sequences of FSM S at state s. FSM P is a *reduction* of FSM S if the reduction relation holds between the initial states of these machines.

2.2 Timed Finite State Machines

A *Timed FSM* (TFSM) is extended with a clock variable, timed guards, timeouts and output delays [12, 13]. The timed guards at a state have less time upper bounds than the timeout at the state and describe the behavior at a given state for inputs which arrive at different time instances. The clock variable accumulates time and is reset to zero when applying an input, producing an output and moving between states by timeout transitions. Correspondingly, an initialized TFSM is a 6-tuple $\mathsf{S} = (I, S, O, h_\mathsf{S}, \Delta_\mathsf{S}, s_0)$ where S is a finite non-empty set of states with the designated initial state s_0, I and O are input and output alphabets, $h_\mathsf{S} \subseteq S \times I \times O \times S \times \Pi \times Z$ is the *transition relation* and Δ_S is the timeout function. The set Π is a set of *input timed guards* and Z is the set of output delays which are non-negative integers. The *timeout function* is the function Δ_S: $S \to S \times (N \cup \{\infty\})$ where N is the set of positive integers: for each state this function specifies the maximum time for waiting for an input. If no input is applied until an (input) timeout expires then the system can spontaneously move to another state. By definition, for each state of TFSM exactly one timeout is specified. An input timed guard $g \in \Pi$ describes the time domain of clock variable when a transition can be executed and is given in the form of interval $<min, max>$ from $[0; T)$, where $< \in \{(, [\}, > \in \{),]\}$ and T is the input timeout at the current state. We also denote the largest finite boundary of timed guards and timeouts as B_S. The transition $(s, i, o, s', g, d) \in S \times I \times O \times S \times \Pi \times Z$ means that TFSM S being at state s accepts an input i applied at time $t \in g$ measured from the initial moment or from the moment when TFSM S has moved to the current state; the clock then is set to zero and S produces output o exactly after d time units and moves to state s'. Given state s of TFSM S such that $\Delta_\mathsf{S}(s) = (s', T)$, if no input is applied before the timeout T expires, the TFSM S moves to state s'. If $\Delta_\mathsf{S}(s) = (s', \infty)$ then $s' = s$, and this means that the TFSM can stay at state s indefinitely long waiting for an input.

Given TFSM S, S is a *complete* TFSM if the union of all input timed guards at any state s under every input i equals $[0; T)$ when $\Delta_S(s) = (s', T)$. In this paper, we consider only complete TFSMs and the question about the interpretation of undefined transitions in partial machines and their augmentation is out of the scope of this paper [17].

TFSM S is a *deterministic* TFSM if for each two transitions $(s, i, o_1, s_1, g_1, d_1)$, $(s, i, o_2, s_2, g_2, d_2) \in h_S$, $s_1 \neq s_2$, $d_1 \neq d_2$ or $o_1 \neq o_2$, it holds that $g_1 \cap g_2 = \varnothing$, otherwise, TFSM S is *nondeterministic*. In this paper, we assume that the system specification is a complete observable, possibly nondeterministic TFSM while the behavior of an implementation under test (IUT) is described by a complete deterministic TFSM. In other words, the specification describes a set of possible permissible behaviors of an IUT and a conforming implementation must be one of them.

Example. Consider a TFSM S in Figure 1 with two states, one input and three outputs, where a is the initial state and $\Delta_S(a) = (b, 2)$, i.e., the timeout at state a is 2. For state b, $\Delta_S(b) = (b, \infty)$, and this loop is not shown in the figure. If input i is applied to the TFSM at state a at time instance 1 measured from the initial moment then S moves to state b producing output o_2 after one time unit. However, if no input is applied to the TFSM until time value reaches 2 then S moves to state b using a timeout transition. At state b, TFSM S can wait for an input infinitely long.

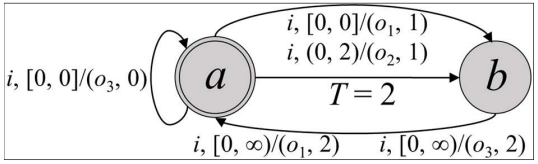

Fig. 1. Timed Finite State Machine S

A *timed input* is a pair (i, t) where $i \in I$ and t is a real; a timed input (i, t) means that input i is applied to the TFSM at time instance t measured from the initial moment or from the moment when TFSM S has produced the last output. A timed output is a pair (o, d) where $o \in O$ and d is the output delay measured from the moment when an input has been applied. In order to determine the output response of the TFSM at state s to a timed input (i, t), state s', which is reached by the TFSM by timeout transitions at time instance t, is calculated first [13]. State s' is a state where TFSM moves from state s via timeout transitions such that the maximum sum Σ of all timeouts starting from state s is less than t. At the second step, a transition (or several transitions for nondeterministic TFSM) (s', i, o, s'', g, d) such that $t - \Sigma \in g$ is considered. According to this transition, the machine produces the timed output (o, d) to a timed input (i, t) applied at state s and moves to the next state s''.

A sequence of timed inputs $\alpha = (i_1, t_1) \ldots (i_n, t_n)$ is a *timed input sequence*, a sequence of timed outputs $\gamma = (o_1, d_1) \ldots (o_n, d_n)$ is a *timed output sequence*. Given the initialized TFSM at state s_1 with the value of the clock variable equal to 0 at the initial moment and a timed input sequence $(i_1, t_1) \ldots (i_n, t_n)$, an input i_1 is applied when the value of the clock variable reaches $t_1' = t_1 - \Sigma_1$ where Σ_1 that is the maximum sum of timeouts for the sequence of timeout transitions starting from state s_1 is less than t_1, but becomes equal or bigger when adding the timeout at the current state s_1'; after applying the input at state s_1' the clock variable is set to 0 and the machine produces an output o_1 and moves to a prescribed state s_2 when clock value is equal to d_1. After producing the output o_1 the clock is reset and the machine is waiting for another input i_2 that is applied when the clock variable value equals $t_2' = t_2 - \Sigma_2$, etc. A sequence $\alpha/\gamma = (i_1, t_1)/(o_1, d_1) \ldots (i_n, t_n)/(o_n, d_n)$ of consecutive pairs of timed inputs and timed outputs starting at the state s is a *timed I/O sequence* or a *timed trace* of TFSM S at state s. Note that time of the first timed input in the sequence is counted from startup of the system at state s while time of all next inputs is counted from the time instance when a previous output has been produced. Similar to FSMs, α is an applied timed input sequence while γ is the

corresponding output response of the TFSM to sequence α of applied inputs. Given a state of a complete nondeterministic TFSM, there can exist several output responses to a timed input sequence.

Similar to FSMs, the set of all timed traces at the initial state specifies the behavior of an initialized TFSM.

Example. Consider TFSM S in fig. 1. If a timed input sequence $(i. 2.5).(i, 0)$ is applied to S at state a then TFSM first moves to state b by the timeout transition when the clock variable value reaches 2. The clock is reset and output $(o_1, 2)$ or $(o_3, 2)$ is produced, the system moves back to state a and the clock is reset. When the next input $(i, 0)$ is immediately applied, the TFSM moves either to state a with timed output $(o_3, 0)$ or to state b with timed output $(o_1, 1)$.

Given states s and p of complete TFSMs S and P, state p is a *reduction* of s (written, $p \leq s$) if the set of timed I/O sequences of TFSM P at state p is contained in the set of timed I/O sequences of TFSM S at state s. TFSM P is a *reduction* of TFSM S if the reduction relation holds between the initial states of the machines. For deterministic TFSMs S and P, the reductions relation is reduced to the equivalence relation.

2.3 FSM abstraction

The behavior of a TFSM can be adequately described using a classical FSM that is called the *FSM abstraction* of the TFSM and is derived similar to [12]; however, in [12], output delays are not considered.

Given a complete observable possibly nondeterministic TFSM $S = (S, I, O, \lambda_S, \Delta_S, s_0)$, the largest finite boundary of timed guards and timeouts B_S and maximum output delay D, we derive the FSM abstraction of TFSM S as the FSM $A_S = (S_A, I \cup \{\mathtt{I}\}, O_A, \lambda_{AS}, s_0)$ where $S_A \subseteq \{(s, 0), (s, (0, 1)), \ldots, (s, (B_S - 1, B_S)), (s, B_S), (s, (B_S, \infty)): s \in S\}$, $O_A = \{(o, 0), (o, 1), \ldots, (o, D): o \in O\} \cup \{\mathtt{I}\}$. The input (output) I is a special input (output) of the FSM abstraction. Given state (s, t_j), $t_j = 0, \ldots, B_S$, of FSM A_S and input i, a transition $((s, t_j), i, (o, d), (s', 0))$ is a transition of the FSM abstraction A_S if and only if there exists a transition $(s, i, o, s', g_i, d) \in \lambda_S$ such that $t_j \in g_i$. Given state (s, g_j), $g_j = (0, 1), \ldots, (B_S - 1, B_S), (B_S, \infty)$, of FSM A_S and input i, a transition $((s, g_i), i, (o, d), (s', 0))$ is a transition of A_S if and only if there exists a transition $(s, i, o, s', g, d) \in \lambda_S$ such that $g_i \subseteq g$. In other words, transitions under input $i \in I$ correspond to timed inputs (i, t) where t is 'hidden' as the second item of states of the FSM abstraction A_S. Transitions under the special input I correspond to the clock change between non-integer and integer values, or to a timeout transition between states. Given state s such that $\Delta_S(s) = (s', T)$, transitions $((s, n), \mathtt{I}, \mathtt{I}, (s, (n, n + 1)))$ and $((s, (n - 1, n)), \mathtt{I}, \mathtt{I}, (s, n))$ are in the transition relation λ_{AS} if and only if $n < T$. Transition $((s, (n - 1, n)), \mathtt{I}, \mathtt{I}, (s', 0)) \in \lambda_{AS}$ if and only if $n = T < \infty$. In [12], it is shown that the FSM abstraction of complete and deterministic TFSM S is also complete and deterministic. In the same way, it can be shown that the FSM abstraction of a complete observable nondeterministic TFSM S is complete observable and nondeterministic.

Example. For a deterministic TFSM S in fig. 1, the corresponding FSM abstraction is shown in fig. 2. FSM abstraction A_S has states $(a, 0)$, $(a, (0, 1))$, $(a, 1)$, $(a, (1, 2))$, $(b, 0)$, $(b, (0, \infty))$. Transitions $((a, 0), i, (o_1, 1), (b, 0))$ and $((a, 0), i, (o_3, 0), (a, 0))$ exist in FSM abstraction A_S since TFSM S has transitions $(a, i, o_1, b, [0, 0], 1)$ and $(a, i, o_3, a, [0, 0], 0)$. FSM abstraction A_S has transition $((a, (0, 1)), i, (o_2, 1), (b, 0))$ since TFSM S has transition $(a, i, o_2, b, (0, 2), 1)$. Transition $((a, 0), \mathtt{I}, \mathtt{I}, (a, (0, 1)))$ of A_S corresponds to clock change at state a from time instance 0 to the interval $(0, 1)$.

A timed input sequence α of TFSM S can be transformed into a corresponding input sequence α_{FSM} of the FSM abstraction A_S. In this case, each timed input (i, t) is replaced by sequence $\mathtt{I}.\mathtt{I} \ldots \mathtt{I}.i$ of inputs of the FSM abstraction where the number of inputs I equals the number of clock transitions between a non-integer and integer values for the time duration t. At the same time the response of the FSM abstraction to sequence $\mathtt{I}.\mathtt{I}.\ldots.\mathtt{I}.i$ equals $\mathtt{I}.\mathtt{I}.\ldots.\mathtt{I}.(o, d)$, where the number of inputs

I is the same as for the timed input (i, t) and (o, d) is the response of the TFSM to timed input (i, t). Thus, the output sequence of the FSM abstraction γ_{FSM} can be transformed into corresponding timed output sequence γ by removing all outputs I. The following statement can be established.

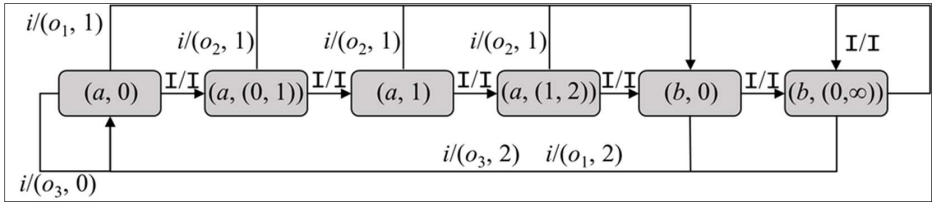

Fig. 2. FSM abstraction A_S of TFSM S in fig. 1

Proposition 1. A timed trace α/γ exists for TFSM S if and only if there exists a trace $\alpha_{FSM}/\gamma_{FSM}$ for the FSM abstraction A_S.

Proposition 2. There exists a timed trace α/γ at state s of a possibly nondeterministic TFSM S if and only if the FSM abstraction A_S has a trace $\alpha_{FSM}/\gamma_{FSM}$ at state (s, 0).

Indeed, all the transitions under input I are deterministic and correspond to the clock change between integer and non-integer value and equal to increasing of clock variable while transitions at state (a, g) of abstraction under another input i corresponds to transitions of TFSM at state a at time (or timed interval) g.

Example. Consider TFSM S in fig. 1 and its FSM abstraction in fig. 2. Timed trace (i, 2.5)/(o_1, 2).(i, 0)/(o_3, 0) of TFSM S corresponds to trace I/I.I/I.I/I.I/I.I/I.i/(o_1, 2).i/(o_3, 0) of FSM abstraction A_S, and vice versa.

According to Proposition 2, all the trace features of a TFSM are preserved for its FSM abstraction and thus, the set of reductions of a TFSM can be analyzed based on a set of reductions of a classical FSM. The following statement establishes necessary and sufficient conditions for two TFSM states to be in the reduction relation.

Proposition 3. State s of TFSM S is a reduction of state p of TFSM P if and only if state (s, 0) of the FSM abstraction A_S is a reduction of state (p, 0) of FSM A_P.

Thus, the conclusion about the reduction relation between two TFSMs can be drawn based on their FSM abstractions and there exist methods for checking the reduction relation between two FSM states or between two FSMs.

3. Fault models and test suites

FSM based testing can be preset and adaptive. We first consider the preset testing where *test cases* which are (timed) input sequences, are derived from the given TFSM specification to determine whether a given IUT, which is assumed to have the FSM behavior, conforms to the given specification.

In this paper, an implementation FSM P conforms to the specification if FSM P is a reduction of the specification FSM. In other words, an implementation FSM P conforms to the specification FSM if for each input sequence the output response of the FSM P is contained in the set of output responses of the specification FSM to this input sequence. In this case, the *fault model* $FM_m^{FSM} = <S, \leq, \Im_m>$ is considered where S is the specification that is a complete observable possibly nondeterministic FSM, ≤ is the reduction relation, $\Im_m$ is the fault domain which contains each deterministic complete FSM with at most m states over the same input alphabet as the specification. Here we notice that differently from the paper [18] where only deterministic FSMs are considered, the specification can be nondeterministic and the conformance relation is not the equivalence but the reduction relation.

Correspondingly, different transfer and separating sequences have to be used when deriving a test suite with guaranteed fault coverage.

A test suite is *complete* with respect to the $FM_m^{FSM} = <S, \leq, \Im_m>$ if for each FSM P ∈ $\Im_m$ that is not a reduction of S, the test suite has a sequence for which an output response of P is not in the set of output responses of S to this sequence.

A complete test suite with respect to FM_m^{FSM} can be derived using an appropriate modification of FSM based methods for nondeterministic FSMs [6] which are based on *deterministically-transfer* (*d-transfer*) and separating sequences. A state s is *deterministically reachable* (*d-reachable*) from the initial state of the FSM S if there exists an input sequence α such that for any output response β to α, the machine S moves from the initial state to state s when α is applied. In this case, α is a *d-transfer* sequence for state s. States s_1 and s_2 of an FSM S are *separable* if there exists an input sequence α such that the sets of output responses of the FSM at states s_1 and s_2 to α do not intersect; in this case, sequence α is called a *separating* sequence for states s_1 and s_2. If a sequence separates each pair of different states of the FSM S then this sequence is a separating sequence for FSM S. Once again we remind that differently from [18], not each input sequence is a *d*-transfer of the nondeterministic specification and separable states and separating sequences for the nondeterministic specification are defined in a different way.

If FSM S has a separating sequence γ and each state is *d*-reachable from the initial state, the procedure for deriving a complete test suite w.r.t. the fault model $FM_n^{FSM} = <S, \leq, \Im_n>$ where n is the number of states of S, has the following steps:

1. A *d-cover* set of the FSM S is derived. This set contains a *d*-transfer sequence for each state of the FSM S.
2. Each sequence of the *d*-cover set is appended with the separating sequence γ of the FSM S and every input that also is appended with the separating sequence γ.

If an adaptive test suite is derived, an adaptive distinguishing sequence can be used instead of a separating sequence while *d*-transfer sequences can be replaced by adaptive transfer sequences (if they exist) [19]. Adaptive distinguishing (separating) and *d*-transfer sequences can be shorter then preset, and moreover, they exist more often.

An input sequence α is *adaptive* if the next input depends on the outputs of the FSM. Such an input sequence can be represented by an FSM called a *test case* [19]. At each state of a test case, either there are transitions for one input with all outputs or there are no transitions and in the latter case, a state is called *terminal*. Given a test case (TC) D for FSM S, an adaptive sequence specified by is applied in the following way. If input i_1 is defined at the initial state d_0 of D then first the input i_1 is applied to FSM S and TC D moves to the i_1o-successor d_1 of state d_0 if o is the output the response of S to the input i_1. The next input to apply is the input defined at state d_1, etc. The procedure terminates when a terminal state is reached.

A test case represents an *adaptive separating* sequence for states s_1 and s_2 of the FSM S if each input-output sequence from the initial to the terminal state of the test case can happen at most at one of states s_1 or s_2. In the former case, the state s_1 is identified, while in the latter case it will be state s_2. States s_1 and s_2 of the FSM S are *adaptively separable* if there is a test case that represents an adaptive separating sequence for states s_1 and s_2. In this case, the corresponding trace from the initial state to a terminal state of an adaptive separating test case allows to determine what was a state of the FSM S before the experiment.

If an adaptive sequence separates each pair of different states of the FSM S, then such a sequence is an *adaptive separating sequence* for the FSM S.

A test case can also represent an adaptive sequence from the initial state of the FSM S to the state s if each input-output sequence of the test case from the initial to a terminal state is ended at state s [19, 20]. In this case, state s is *adaptively reachable* from the initial state. The derivation of a complete adaptive test suite is almost the same as the preset: the only difference is that adaptive

distinguishing sequences are used instead of separating sequences and adaptive transfer sequences are used instead of *d*-transfer sequences.

If FSM S has no (adaptive) separating sequence or S has states which are not *d*-reachable from the initial state then a complete test suite cannot be derived using the above procedure. In this case, the so-called state counting reduction (SCR) method should be applied [6].

Below, we describe the main steps of the general SCR-method when deriving a complete preset test suite with respect to the fault model $FM_m^{FSM} = <\mathbf{S}, \leq, \Im_m>$.

1. Determine subset S_d of all *d*-reachable states and derive *d-cover* of the FSM S which contains a *d*-transfer sequence for each state of S_d.
2. Determine the set $R = \{R_1, R_2, \ldots, R_p\}$ of maximal subsets of pairwise separable states; for each subset $R_j \in R$, denote R_{jd} a subset of all *d*-reachable states of R_j. For each subset $R_j \in R$, derive a distinguishing set W_j that contains a separating sequence for each pair of different states of R_j.
3. For each state s_k of S_d, derive a set of input sequences N_k: an input sequence $\alpha \in N_k$ if for each *I/O* sequence α/β at state s_k, it holds that α/β traverses states of some $R_j \in R$ at least $m - |R_{jd}| + 1$ times and this does not hold for any proper prefix of α. Concatenate each prefix of sequence α with each sequence of the set W_j.
4. Concatenate each *d*-transfer sequence with each sequence of each set W_j that was used at Step 3 when terminating an input sequence of the set N_k, $k = 1, \ldots, p$.

Here we notice that in general case, complete test suites derived by SCR method are much longer than for the case when the specification FSM has a separating sequence and the derivation method is much more complex. To minimize our efforts for deriving a complete test suite with respect to the fault model $FM_m^{FSM} = <\mathbf{S}, \leq, \Im_m>$, the adaptive testing can be used instead of the preset [19].

It is known that a test suite can be usually shorter if the specification FSM has a sequence, which separates every two states [6]. In this case, set W_j contains only one separating sequences α and $R = \{S\}$. However, such a separating sequence does not always exist and thus, we are obliged to use a set of separating sequences for test derivation. Adaptive distinguishing (separating) sequences exist more often than the preset and are usually shorter, thus, adaptive distinguishing sequences can be preferable for test derivation. Anyway, using adaptive distinguishing sequences can increase the size of subsets of pairwise distinguishable states from R, and thus, shorten sets W_j and the sets N_k, and correspondingly, minimize a complete test suite.

In the next section, we consider an existing approach for adaptation classical FSM based test derivation methods for Timed FSM.

4. Related work on TFSM based testing

The problem of deriving a complete test suite against a nondeterministic FSM with timed guards with respect to the reduction relation has been considered in [20]. The proposed approach is based on the FSM abstraction of TFSM but that abstraction is a bit different from that considered in the «Preliminaries» section. In that case, one-to-one mapping between sets of states of TFSM and corresponding FSM abstraction has been established. The latter allows to inherit the above described steps for deriving a complete test with respect to the fault domain which contains each deterministic complete TFSMs with timed guards with at most m states over the same input alphabet as the specification TFSM S and the largest boundary B_S for input timed guards. However, in general case, this approach cannot be applied for FSMs with time guards and timeouts since the one-to-one mapping between transitions of two state reduced equivalent TFSMs with timeouts not always can be established.

In [15], it is shown that initialized reduced deterministic TFSM specification and TFSM implementation with timeouts can be equivalent yet not isomorphic; moreover, they can have

different number of states. The latter violates the main assumption of W-based methods about checking the correspondence between FSM transitions. As an example, consider TFSMs in fig. 3.

Each state in R and Q is reachable from the initial state and each two different states of each machine are not equivalent, i.e., both TFSMs are connected and state reduced. By direct inspection, one can assure that equivalent machines in Figure 3 have different number of states and thus, are not isomorphic.

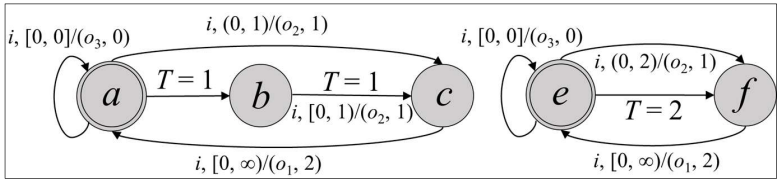

Fig. 3. Two state reduced deterministic complete TFSMs R and Q

On the other hand, according to Propositions 1-3, the necessary relationship holds between transitions of their FSM abstractions. For example, reduced forms of FSM abstractions of TFSMs R and Q (fig. 3) are isomorphic. FSM abstraction A_R and its reduced form is shown in fig. 4. Thus, in order to derive a complete test suite for deterministic TFSMs we considered the fault domain containing every TFSM P over the same input alphabet as S such that the reduced form of the FSM abstraction of P has at most $m > 1$ states [15]. A similar approach can be applied for the test derivation against nondeterministic FSMs with timeouts and timed guards; in the next section, corresponding fault model and test derivation method are proposed.

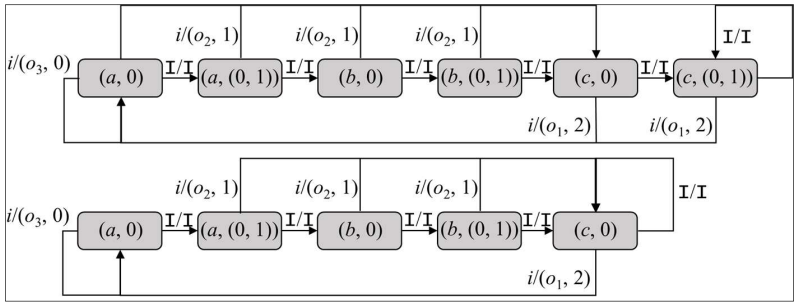

Fig. 4. The FSM abstraction A_R of TFSMs R (Figure 3) and its reduced forms

5. Test derivation method for nondeterministic FSM with timed guards and timeouts

In order to derive a test suite with guaranteed fault coverage against the nondeterministic specification TFSM, we propose a fault model based on the FSM abstraction of the TFSM and algorithm of applying the SCR-method to such abstraction.

Given a nondeterministic TFSM S with n states (fig. 1), two deterministic equivalent TFSM implementations R and Q (fig. 3) which are reductions of S can have different number of states. However, the reduced forms of their FSM abstractions are isomorphic and are reductions of FSM abstraction A_S. Another example in Figure 5 demonstrates that for nondeterministic TFSM Y there can exist a deterministic TFSM Y′ with the same number of states and the boundary B_S, such that the reduced form [16] of FSM abstraction $A_{Y'}$ has more states than that of A_Y.

Given the TFSM specification S, we consider the fault model $FM_m^{TFSM} = <\mathbf{S}, \leq, \aleph_m>$, where S is the complete observable, possibly nondeterministic TFSM specification, $\leq$ is the reduction relation,

$\aleph_m$ is the fault domain that contains each deterministic complete TFSM P over the same input alphabet as the specification such that the reduced form of its FSM abstraction A_P has at most $m > 1$ states.

As it is demonstrated below it can well happen that some timed FSMs with less states than the specification TFSM are not included into the fault domain and vice versa, a number of timed FSMs which have more states than the specification TFSM are included into the fault domain.

Example. Consider TFSM specification S (fig. 1) with two states. Fault domain $\aleph_m$ of the fault model $FM_m^{TFSM} = <S, \leq, \aleph_m>$ contains TFSM R (fig. 3) with three states since the FSM abstraction A_R has not more states than the FSM abstraction A_S. At the same time, in Figure 5 the TFSM specification Y and its non-conforming implementation Y′ are shown such that both TFSMs have three states and the finite timed guards' boundary equal two. However, the fault domain $\aleph_m$ does not contain Y′ since the reduced form of its FSM abstraction has more states than A_Y. Thus, it can happen that nonconforming implementations with the same number of states as the specification TFSM can pass a complete test suite with respect to $<S, \leq, \aleph_m>$.

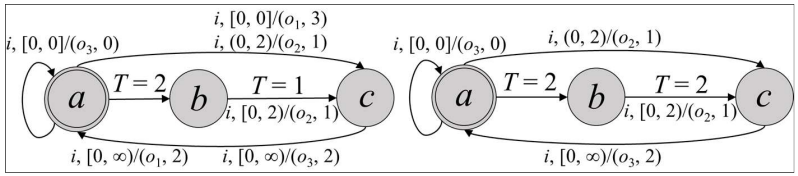

Fig. 5. TFSM Y and its non-conforming implementation Y′

Note that the FSM abstraction of TFSM S can have non-separable states, i.e., the FSM abstraction can have a pair of states for which a separating sequence does not exist when the specification TFSM S has a separating sequence, i.e., all states of the TFSM S are pairwise separable. For example, TFSM S in Figure 1 has a separating sequence (i, 1) while the corresponding FSM abstraction A_S has a pair of non-separable states (b, 0) and (b, (0, ∞)), for which the sets of input/output sequences coincide. In order to derive a complete test suite for such FSM, the SCR method can be used.

As mentioned above, similar to a deterministic FSM abstraction [15], a nondeterministic FSM abstraction can be minimized using the method from [16]. As an example, for FSM abstraction A_S (fig. 2) of TFSM S (Figure 1), equivalent states (b, 0) and (b, (0, ∞)) can be merged into one state. However, unlike deterministic machines, such optimization does not always allow to merge pairs of non-separable states of the FSM abstraction of the specification and thus, the SCR method is still used for test derivation.

Algorithm for deriving a complete test suite with respect to the fault model $FM_m^{TFSM} = <S, \leq, \aleph_m>$ where m is the number of states of the reduced form of the FSM abstraction of S

Input: The complete observable possibly nondeterministic specification TFSM S

Output: A complete test suite *TS* with respect to the fault model $FM_m^{TFSM} = <S, \leq, \aleph_m>$, where $\aleph_m$ contains every TFSM P over the same input alphabet as S such that the reduced form of the FSM abstraction of P has at most $m > 1$ states

Step 1. Derive the reduced form of the FSM abstraction A_S of TFSM S.

Step 2. Derive a test suite TS_A with respect to the fault model $FM_m^{FSM} = <A_S, \leq, \Im_m>$ using the SCR-method described above, where m is number of states of the FSM abstraction A_S.

Step 3. According to Proposition 1, transform sequences of the test suite TS_A into corresponding timed sequences over the TFSM S and obtain the test suite *TS*.

Proposition 4. The test suite *TS* returned by the above Algorithm is complete with respect to the fault model $FM_m^{TFSM} = <S, \leq, \aleph_m>$.

Proof. Let a test suite *TS* be returned by the above algorithm and TFSM P which is not a reduction of specification TFSM S is in the set $\aleph_m$. By definition of the fault domain $\aleph_m$, the reduced form of the FSM abstraction A_P has at most m states. Since P is not a reduction of S, the FSM A_P is not a reduction of A_S (Proposition 3). Thus, a test suite TS_A derived at Step 2 contains an input sequence α_{FSM}, which separates FSMs A_P and A_S. By Proposition 2, for sequence α_{FSM} of the FSM A_P there exists the corresponding timed input sequence α of the TFSM P that will demonstrate that P is not a reduction of the TFSM S. The latter guarantees that each non-conforming implementation P of the set $\aleph_m$ is detected by the test suite *TS*.

The fault domain in the above algorithm can be extended and for TFSM S with the reduced form of its FSM abstraction A_S which has n states, a complete test suite can be derived by SCR-method with respect to $\aleph_m$ when $m > n$. However, in this case length of a complete test suite significantly increases [21].

In the worst case, the length of a test suite derived by SCR-method exponentially depends on the number of states of FSM and this also holds for FSM with timed aspects. As experimental results show, in practice, length of adaptive distinguishing sequences is usually polynomial with respect to the number of FSM states when such a sequence exists [20, 7]. Respectively, similar results can be derived for a TFSM when proposed algorithm is used and the boundary on timed guards is not too big. Note that length of the test suite also significantly depends on timed aspects of the specification TFSM such as the upper bounds of timed guards and value of timeouts [20, 21].

We note again that the FSM abstraction of TFSM S can have non-separable states while all states of the TFSM are pairwise separable. However, we underline that the FSM abstraction inherits the d-reachability of states from the specification TFSM and the following proposition holds.

Proposition 5. States (s, 0), (s, (0, 1)), (s, 1), (s, (1, 2)) … of FSM abstraction A_S are d-reachable if and only if state s is d-reachable in TFSM S.

The statement is implied by Propositions 1-2 due to a deterministic transition under the special input I. Respectively, all states of FSM abstraction A_S are d-reachable if and only if all states of TFSM S is d-reachable.

Example. Consider TFSM S in Figure 1 and its FSM abstraction A_S in Figure 2. We derive a complete test suite with respect to the fault model $FM_6^{TFSM} = <S, \leq, \aleph_6>$. For state ($b$, 0) of A_S there exists a d-transfer sequence I.i and respectively, state b of TFSM S has a timed d-transfer sequence (i, 0,5). Other states of FSM abstraction are d-reachable from states (a, 0) and (b, 0) by a sequence of I inputs. Thus, all states of A_S are d-reachable from the initial state and for the FSM abstraction A_S, $S_d = \{(a, 0), (a, (0, 1)), (a, 1), (a, (1, 2)), (b, 0), (b, (0, \infty))\}$.

Given FSM A_S, we can also determine two maximal subsets of pairwise separable states $R_1 = \{(a, 0), (a, (0, 1)), (a, 1), (a, (1, 2)), (b, 0)\}$, $R_2 = \{(a, 0), (a, (0, 1)), (a, 1), (a, (1, 2)), (b, (0, \infty))\}$ and corresponding distinguishing sets $W_1 = W_2 = \{ i, I.i, I.I.i\}$. Note that $R_1 = R_{1d}$ and $R_2 = R_{2d}$ since all states of A_S are d-reachable.

Consider state (b, 0) and the set $N_{(b, 0)}$ of input sequences derived at Step 3 of the SCR-method when a test suite is derived with respect to the fault model $<S, \leq, \Im_6>$. Input/Output sequences with the input projection of the set $N_{(b, 0)}$ should traverse states of some R_j at least $2 = 6 - 5 + 1$ times while this does not hold for any proper prefix of the input sequence, and respectively, $i.i$ is in the set $N_{(b, 0)}$ which traverses states (a, 0) and (b, 0) of R_1. Other sequences at state (b, 0) are i.I (traverses (a, 0), (a, (0, 1))), I.i (traverses (b, (0, ∞)), (a, 0)), I.I (traverses (b, (0, ∞)), (b, (0, ∞))) and thus, $N_{(b, 0)} = \{i.i, i.I, I.i, I.I, i, I\}$.

A fragment of the tree that is obtained when deriving a test suite, is shown in fig. 6. One of test sequences of TS_A is I.i.i.i.I.I.i and a corresponding timed input sequence of test *TS* is (i, 0,5).(i, 0).(i, 0).(i, 1) = γ where (i, 0,5) is a d-transfer sequence and (i, 1) is a separating sequence from W_1. Each sequence of the test suite is applied to TFSM implementation at the initial state. First input i of γ is applied when clock value is equal to 0,5; after applying the input the clock is set to 0 and the

machine produces corresponding output when clock value is equal to 1 when an implementation is conforming. After producing the output o_2 the clock is reset and the machine is waiting for the next input i from timed input (i, 0) that is immediately applied after resetting the clock. After applying this input the clock is reset again and the machine produces an output o_1 or o_3 when the clock value is equal to 2. After producing any of outputs by the TFSM the clock is reset and the machine is waiting for a next input, etc.

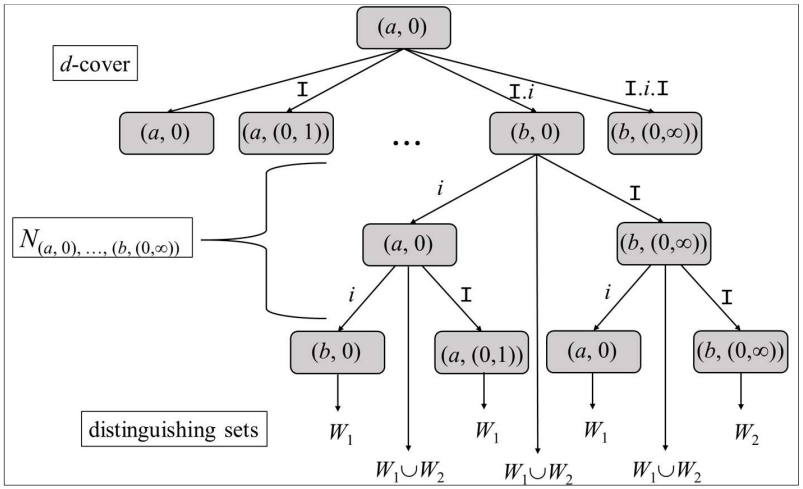

Fig. 6. A fragment of test suite TS_A for the FSM abstraction A_S

6. Conclusion

In this paper, we have proposed an approach for deriving complete test suites with respect to the reduction relation against nondeterministic Finite State Machines with timed guards and timeouts. Both, a proposed approach and a corresponding fault model are based on the FSM abstraction of machines with timed guards and timeouts and this allows inheriting the known FSM based SCR-method when deriving test suites with guaranteed fault coverage for nondeterministic TFSMs.

References / Список литературы

[1]. Gill A. Introduction to the Theory of Finite-State Machines, McGraw-Hill, 1962, 272 p. / Гилл А. Введение в теорию конечных автоматов. Наука, 1966, 272 стр.

[2]. Chow T.S. Test design modeled by finite-state machines. IEEE Transactions on Software Engineering, vol. 4, no. 3, 1978,pp. 178–187.

[3]. Dorofeeva R., El-Fakih K., Maag S., Cavalli A., Yevtushenko N. FSM-based conformance testing methods: A survey annotated with experimental evaluation. Information and Software Technology, vol. 52, issue 12, 2010, pp. 1286–1297.

[4]. Hierons R.M., Merayo M.G., Nunez M. Testing from a Stochastic Timed System with a Fault Model. Journal of Logic and Algebraic Programming, vol. 72, no. 8, 2009, pp. 98–115.

[5]. Krichen M. and Tripakis S. Conformance testing for real-time systems. Formal Methods in System Design, vol. 34, no. 3, 2009, pp. 238–304.

[6]. Petrenko A., Yevtushenko N. Conformance tests as checking experiments for partial nondeterministic FSM. Lecture Notes in Computer Science, vol. 3997, 2006, pp. 118–133.

[7]. Tvardovskii A.S., Yevtushenko N.V. Deriving adaptive distinguishing sequences for Finite State Machines. Trudy ISP RAN/Proc. ISP RAS, vol. 30, issue 4, 2018, pp. 139–154 (in Russian). DOI: 10.15514/ISPRAS-2018-30(4)-9 / Твардовский А.С., Евтушенко Н.В. К синтезу адаптивных различающих последовательностей для конечных автоматов. Труды ИСП РАН, том 30, вып. 4, 2018, стр. 139–154.

[8]. Alur R. and Dill D.L. A theory of timed automata. Theoretical Computer Science, vol. 126, issue 2, 1994, pp. 183–235.

[9]. Springintveld J., Vaandrager F., and D'Argenio P. Testing timed automata. Theoretical Computer Science, vol. 254, no. 1–2, 2001, pp. 225–257.

[10]. El-Fakih K., Yevtushenko N., and Fouchal H., Testing timed finite state machines with guaranteed fault coverage. In Proc. of the 21st IFIP WG 6.1 International Conference on Testing of Software and Communication Systems and 9th International FATES Workshop, 2009, pp. 66–80.

[11]. Merayo M.G., Nunez M., and Rodriguez I. Formal testing from timed finite state machines. Computer Networks, vol. 52, issue 2, 2008, pp. 432–460.

[12]. Bresolin D., El-Fakih K., Villa T., and Yevtushenko N. Deterministic timed finite state machines: Equivalence checking and expressive power. In Proc. of the 5th International Symposium on Games, Automata, Logics and Formal Verification (GandALF 2014), 2014, pp. 203–216.

[13]. Zhigulin M., Yevtushenko N., Maag S., Cavalli A. FSM-Based Test Derivation Strategies for Systems with Time-Outs. In Proc. of the 11th International Conference on Quality Software, 2011, pp. 141–150.

[14]. En-Nouaary A., Dssouli R., Khendek F. Timed Wp-Method: Testing Real-Time Systems. IEEE Transactions on Software Engineering, vol. 28, issue 11, 2002, pp. 1023–1038.

[15]. Tvardovskii A., El-Fakih K, Yevtushenko N. Deriving Tests with Guaranteed Fault Coverage for Finite State Machines with Timeouts. Lecture Notes in Computer Science, vol. 11146, 2018, pp. 149–154.

[16]. Starke P. Abstract Automata. North-Holland Publishing Company, 1972, 419 p.

[17]. Villa T., Kam T., Brayton R.K., Sandgiovanni-Vincentelli A. Synthesis of Finite State machines: Logic Optimization, Springer, 1997, 381 p.

[18]. Lee D., Yannakakis M. Principles and methods of testing finite state machines - a survey. Proceedings of the IEEE, vol. 84, issue 8, 1996, pp. 1090–1123.

[19]. Yevtushenko N., El-Fakih K., and Ermakov A., On-the-fly construction of adaptive checking sequences for testing deterministic implementations of nondeterministic specifications. Lecture Notes in Computer Science, vol. 9976, 2016, pp. 139–152.

[20]. Tvardovskii A.S., Gromov M.L., El-Fakih Khaled, Yevtushenko N.V. Testing Timed Nondeterministic Finite State Machines with the Guaranteed Fault Coverage. Automatic Control and Computer Sciences, vol. 51, № 7, 2017, pp. 724–730.

[21]. Tvardovskii A., Vinarskii E. Yevtushenko N. Experimental Evaluation of Timed Finite State Machine Based Test Derivation. In Proc. of the International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices, 2019, pp. 102–107.

Информация об авторах / Information about authors

Александр Твардовский получил степень магистра радиофизики в Томском государственном университете. С 2017 года обучается в аспирантуре. Исследовательские интересы включают теорию автоматов и тестирование программного обеспечения.

Aleksandr Tvardovskii received his Master's degree from Faculty of Radiophysics of Tomsk State University. He is a Ph.D. student since 2017. His research interests include automata theory and software testing.

Нина Евтушенко, доктор технических наук, профессор, главный научный сотрудник ИСП РАН, профессор НИУ ВШЭ. До 1991 года работала научным сотрудником в Сибирском физико-техническом институте. С 1991 г. работала в ТГУ профессором, зав. кафедрой, зав. лабораторией по компьютерным наукам. Её исследовательские интересы включают формальные методы, теорию автоматов, распределенные системы, протоколы и тестирование программного обеспечения.

Nina Yevtushenko, Doctor of Technical Sciences, professor, a chief researcher of ISP RAS, professor at HSE. She worked at the Siberian Scientific Institute of Physics and Technology as a researcher up to 1991. In 1991, she joined TSU as a professor and then worked as the chairhead and the head of Computer Science laboratory. Her research interests include formal methods, automata theory, distributed systems, protocol and software testing.