К синтезу адаптивных проверяющих последовательностей для недетерминированных автоматов

А.Д. Ермаков <antonermak@inbox.ru>
Н.В. Евтушенко <yevtushenko@sibmail.com>
Национальный исследовательский Томский государственный университет, 634050, Россия, г. Томск, пр. Ленина, д. 36.

Аннотация. Существует достаточно много публикаций, посвященных построению проверяющей последовательности для полностью определенного детерминированного конечного автомата. Тем не менее, для недетерминированных автоматов таких публикаций достаточно мало; исследователи начали с того, что предложили алгоритм построения проверяющей последовательности для инициального недетерминированного автомата относительно эквивалентности. В данной работе рассматривается построение адаптивной проверяющей последовательности относительно редукции. Проверяемый автомат есть редукция полностью определенного автомата-спецификации, если для каждой входной последовательности выходная реакция проверяемого автомата содержится в множестве выходных реакций спецификации на эту входную последовательность. В первой части данной статьи мы предполагаем, что полностью определенный возможно недетерминированный автомат-спецификация имеет разделяющую последовательность разумной длины, каждое состояние детерминировано достижимо из любого другого состояния, и проверяемый автомат (автомат-реализация) является полностью определенным и детерминированным. Поведение проверяемого автомата неизвестно; мы знаем только, что его число состояний не больше числа состояний автомата-спецификации. При описанных выше условиях проверяемый автомат является редукцией спецификации, если и только если проверяемый автомат изоморфен подавтомату автомата-спецификации. Таким образом, необходимо адаптивно построить проверяющую последовательность, проходящую по каждому переходу проверяемого автомата, и проверить конечное состояние перехода посредством разделяющей последовательности. Во второй части статьи мы предлагаем использовать вместо разделяющей последовательности (адаптивный) различающий тестовый пример и на простом примере иллюстрируем, как такая замена может сократить длину адаптивной проверяющей последовательности. Длина тестового примера обычно короче разделяющей последовательности, и вообще говоря, различающий тестовый пример может существовать для автомата-спецификации, не имеющего разделяющей последовательности. В третьей части статьи мы обсуждаем возможность применения

предлагаемой методики построения проверяющей последовательности для частичных возможно недетерминированных автоматов, выделяя наибольший полностью определенный подавтомат. Если такой подавтомат существует, обладает разделяющей последовательностью или различающим тестовым примером, то его можно использовать для построения адаптивной различающей последовательности для исходного частичного, возможно недетерминированного автомата. Тем не менее, следует отметить, что в последнем случае проверяющая последовательность строится относительно не относительно редукции, а относительно отношения квази редукции.

Ключевые слова: недетерминированный конечный автомат; отношение редукции; модель неисправности; адаптивная проверяющая последовательность**.**

DOI: 10.15514/ISPRAS-2016-28(3)-8

Для цитирования: Ермаков А.Д., Евтушенко Н.В. К синтезу адаптивных проверяющих последовательностей для недетерминированных автоматов. Труды ИСП РАН, том 28, вып. 3, 2016 г. стр. 123-144. DOI: 10.15514/ISPRAS-2016-28(3)-8.

1. Введение

Тестирование на основе конечных автоматов широко используется при построении различных тестовых последовательностей для интерактивных дискретных систем [1]; хорошим примером может служить синтез проверяющих тестов для телекоммуникационных протоколов на основе конечных автоматов [2].

В большинстве публикаций предполагается, что спецификация представлена в виде инициального автомата, а проверяющий тест есть множество входных последовательностей, объединяемых посредством надежного сигнала сброса [3]. Если такой сигнал сброса является слишком дорогим, то вместо множества тестовых последовательностей используются так называемые проверяющие последовательности [4, 5]. Для детерминированных автоматов такие последовательности можно построить, когда автоматная спецификация имеет синхронизирующую последовательность, переводящую автомат из любого состояния в одно и то же состояние, и диагностическую последовательность, способную различить любые два различных состояния автомата [4]. Для недетерминированных автоматов количество публикаций на тему синтеза проверяющих последовательностей значительно меньше, тогда как недетерминированные спецификации возникают в различных компьютерных приложениях (программах, системах) [5]. Одной из причин появления недетерминированных спецификаций является опциональность, сопутствующая, например, RFC описаниям многих телекоммуникационных протоколов [6].

В работе [5] авторы предлагают метод построения проверяющей последовательности для полностью определенного недетерминированного автомата относительно эквивалентности при наличии соответствующих ограничений на конечно автоматную спецификацию и область неисправности. В работе [7] авторы обобщают результаты для построения проверяющей последовательности относительно редукции. Как обычно, проверяющая последовательность называется адаптивной, если следующий входной символ зависит от выходных символов, полученных от проверяемой системы в ответ на приложенные ранее входные символы. Еще один метод для построения адаптивной проверяющей последовательности относительно редукции предлагается в [8].

В данной работе мы ослабляем некоторые ограничения работы [8] и предлагаем использовать при построении адаптивной проверяющей последовательности вместо разделяющей последовательности (адаптивный) тестовый пример (адаптивную различающую последовательность). Длина адаптивного различающего тестового примера может быть меньше, чем длина разделяющей последовательности [9,10]; более того, различающий тестовый пример может существовать и при отсутствии разделяющей последовательности.

В первой части данной статьи мы предполагаем, что полностью определенный возможно недетерминированный автомат-спецификация имеет разделяющую последовательность разумной длины, каждое состояние детерминировано достижимо из любого другого состояния, и проверяемый автомат (автомат-реализация) является полностью определенным и детерминированным. Более того, поведение проверяемого автомата неизвестно; мы знаем только, что его число состояний не больше числа состояний автомата-спецификации. Проверяемый автомат есть редукция спецификации, если для каждой входной последовательности выходная реакция проверяемого автомата содержится в множестве выходных реакций автомата-спецификации на эту входную последовательность. При описанных выше условиях проверяемый автомат является редукцией спецификации, если и только если проверяемый автомат изоморфен подавтомату автомата-спецификации [8]. Таким образом, вместо того, чтобы проверять все входные последовательности, достаточно установить взаимно однозначное соответствие между состояниями и переходами автомата-спецификации и проверяемого автомата [4]. Другими словами, при построении проверяющей последовательности нужно «пройти» по каждому переходу проверяемого автомата и проверить конечное состояние перехода посредством разделяющей последовательности. Такой подход позволяет построить проверяющую последовательность разумной длины, если разделяющая и передаточные последовательности имеют полиномиальную длину относительно числа состояний автомата-спецификации. Мы кратко описываем, как построить адаптивную проверяющую последовательность при таком подходе.

Во второй части статьи мы предлагаем использовать вместо разделяющей последовательности (адаптивный) различающий тестовый пример [11,12] и на простом примере иллюстрируем, как такая замена может сократить длину адаптивной проверяющей последовательности. Мы также отмечаем, что различающий тестовый пример может существовать для автомата-спецификации, не имеющего разделяющей последовательности.

В третьей части статьи мы обсуждаем возможность применения методики построения проверяющей последовательности относительно редукции для частичных возможно недетерминированных автоматов, выделяя наибольший полностью определенный подавтомат. Если данный подавтомат существует, обладает разделяющей последовательностью или различающим тестовым примером и является детерминировано связным, то установление взаимно однозначного соответствия между состояниями проверяемого автомата и автомата-спецификации возможно точно так же как для полностью определенного автомата.

Структура статьи следующая. Раздел 2 содержит основные определения и обозначения. Раздел 3 описывает адаптивный подход к построению проверяющей последовательности при использовании разделяющей последовательности. Раздел 4 описывает изменения в адаптивном подходе при использовании различающего тестового примера вместо разделяющей последовательности. В разделе 5 мы обсуждаем возможности использования предложенного подхода к построению адаптивной проверяющей последовательности для частичных, возможно недетерминированных автоматов.

2. Определения

Конечным автоматом, или просто *автоматом* называется четверка $S = \langle S, I, O, h_S \rangle$, где S - конечное непустое множество состояний, I и O - конечные входной и выходной алфавиты, и $h_S \subseteq S \times I \times O \times S$ есть отношение переходов. Автомат S *недетерминированный*, если существует хотя бы одна пара $(s, i) \in S \times I$, для которой существуют несколько пар $(o, s') \in O \times S$ таких, что $(s, i, o, s') \in h_S$. Автомат S называется *полностью определенным*, если для каждой пары $(s, i) \in S \times I$ существует $(o, s') \in O \times S$ такое, что $(s, i, o, s') \in h_S$; в противном случае автомат S называется *частично определенным* или *частичным.* Автомат S *наблюдаемый*, если для каждой пары переходов (s, i, o, s_1), $(s, i, o, s_2) \in h_S$ имеет место $s_1 = s_2$. Автомат S *инициальный*, если существует выделенное начальное состояние s_1 (обозначение: S/s_1). Таким образом, инициальный автомат есть пятерка $\langle S, I, O, h, s_1 \rangle$. Для автоматов $S = \langle S, I, O, h_S, s_1 \rangle$ и $T = \langle T, I, O, h_T, t_1 \rangle$ автомат T есть подавтомат S, если $T \subseteq S$, $t_1 = s_1$ и $h_T \subseteq h_S$. В дальнейшем, мы рассматриваем наблюдаемые и полностью определенные автоматы, если явно не указано иное.

В качестве примера рассмотрим автомат на рис. 1 с множеством состояний $S = \{1, 2, 3\}$ и начальным состоянием 1. Множество $I = \{i_1, i_2\}$ есть множество входных символов автомата, $O = \{0, 1, 2\}$ есть множество входных символов. Автомат является недетерминированным. Например, из состояния 2 под действием входного символа i_2 есть переходы в состояния 2 и 3.

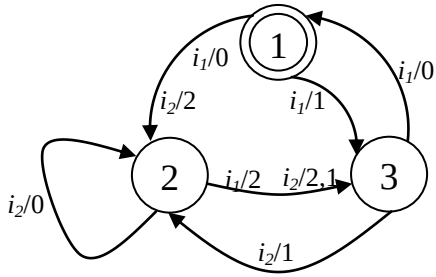

Рис. 1. Автомат S.

Fig. 1. FSM S.

Последовательность входо-выходных пар переходов, начинающаяся в состоянии s автомата $S = \langle S, I, O, h \rangle$, называется *входо-выходной последовательностью* или *трассой* автомата S в состоянии s. Множество всех трасс автомата S в состоянии s, включая пустую последовательность, обозначается $Tr(S/s)$. Обычным образом, отношение переходов автомата распространяется на входные и выходные последовательности. Четверка (s, α, β, s'), в которой s и s' состояния автомата, α и β - входная и выходная последовательности, принадлежит отношению переходов h_S автомата S, если пара последовательностей α и β (обозначение α/β) является трассой в состоянии s, переводящей автомат из состояния s в состояние s'.

Обозначение $successor(s, \alpha)$ используется для обозначения всех состояний, достижимых из состояния s после подачи входной последовательности α, т.е. $successor(s, \alpha) = \{s' : \exists \beta \in O^* [(s, \alpha, \beta, s') \in h_S]\}$.

Множество $out(s, \alpha)$ обозначает множество всех выходных последовательностей (реакций), которые автомат S может выдать в состоянии s после подачи входной последовательности α, т.е., $out(s, \alpha) = \{\beta : \exists s' \in S [(s, \alpha, \beta, s') \in h]\}$. Для автомата на рис. 1 $successor(2, i_2\, i_1) = \{1, 3\}$, а $out(2, i_2\, i_1) = \{02, 20, 10\}$.

Автомат S *детерминировано связный* (*д-связный*), если для каждой пары состояний $s, s' \in S$ существует входная последовательность α такая, что $successor(s, \alpha) = \{s'\}$. В этом случае мы говорим, что s' *детерминировано* (*д-*)*достижимо* из состояния s. Такая *д*-передаточная входная

последовательность обозначается $\alpha_{ss'}$. Инициальный автомат *связный*, если для каждого $s \in S$ существует входная последовательность, которая переводит автомат S из начального состояния в состояние s. Любой ∂-связный автомат является связным по определению.

Пусть $S = \langle S, I, O, h_S, s_1 \rangle$ и $P = \langle P, I, O, h_P, p_1 \rangle$ инициальные автоматы; *пересечением* автоматов $S \cap P$ называется наибольший связный подавтомат автомата $\langle S \times P, I, O, f, s_1 p_1 \rangle$, где $(sp, i, o, s'p') \in f \Leftrightarrow (s, i, o, s') \in h_S \,\&\, (p, i, o, p') \in h_P$. В полностью определенном автомате $S = \langle S, I, O, h_S \rangle$ состояния s_1 и s_2 называются *неразделимыми*, если для каждой входной последовательности $\alpha \in I^*$ имеет место $out(s_1, \alpha) \cap out(s_2, \alpha) \neq \varnothing$, т.е. множества выходных реакций автомата в состояниях s_1 и s_2 на каждую входную последовательность пересекаются; в противном случае, состояния s_1 и s_2 *разделимы*. Для разделимых состояний s_1 и s_2 существует входная последовательность $\alpha \in I^*$ такая, что $out(s_1, \alpha) \cap out(s_2, \alpha) = \varnothing$, т.е. множества выходных реакций в состояниях s_1 и s_2 на входную последовательность α не пересекаются. Такая входная последовательность α называется *разделяющей* последовательностью для состояний s_1 и s_2 (обозначение: $s_1 \not\sim_\alpha s_2$).

В качестве примера рассмотрим состояния 1 и 2 автомата на рис. 1. Множества выходных последовательностей автомата S в состояниях 1 и 2 на входную последовательность $i_2\, i_1$ суть непересекающиеся множества {22} и {02, 20, 10}; таким образом, последовательность $i_2\, i_1$ является разделяющей для этих двух состояний.

Если последовательность α разделяет каждую пару состояний автомата S, то она является *разделяющей* последовательностью для автомата S. Для автомата S на рис. 1 последовательность $i_2\, i_1$ разделяет каждую пару состояний автомата, поскольку множество выходных реакций автомата S на разделяющую последовательность $i_2\, i_1$ в состоянии 3 есть множество {12}. Методы проверки существования разделяющей последовательности и методы её построения можно найти в [13]. К сожалению, разделяющая последовательность существует не для всякого полностью определенного автомата. Кроме того, известно, что длина разделяющей последовательности может быть экспоненциальной относительно числа состояний автомата S. Тем не менее, в [14] отмечается, что согласно экспериментам со случайно сгенерированными автоматами, если разделяющая последовательность для автомата существует, то она довольно короткая.

Пусть S и P полностью определенные автоматы. Состояние p автомата P называется *редукцией* состояния s автомата S (обозначение $p \leq s$), если множество трасс автомата P в состоянии p является подмножеством трасс автомата S в состоянии s; в противном случае, p не является редукцией s. Инициальный автомат P/p_1 является редукцией инициального автомата S/s_1, если $p_1 \leq s_1$, т.е. если множество трасс P/p_1 является подмножеством трасс

S/s_1. Если множества трасс S/s_1 и P/p_1 совпадают, тогда эти автоматы *эквивалентны* [11].

Модель неисправности. При синтезе проверяющих тестов на основе автоматной модели предполагается, что автомат-спецификация описывает эталонное поведение, тогда как область неисправности содержит автоматное описание для каждой возможной реализации спецификации. В данной работе мы предполагаем, что все автоматы инициальные, полностью определенные и наблюдаемые; более того, предполагается, что автомат-реализация (проверяемый автомат) детерминированный и имеет не больше состояний, чем автомат-спецификация. В качестве отношения соответствия (конформности) мы рассматриваем отношение редукции. Другими словами, мы неявно предполагаем, что недетерминизм спецификации вытекает из ее опциональности, где разработчик выбирает лучший вариант в соответствии с некоторыми критериями. В такой системе по-прежнему имеется надежный сигнал сброса, который является довольно дорогим и может быть использован только один раз перед подачей проверяющей последовательности.

Таким образом, мы рассматриваем модель неисправностей $FM = <S/s_1, \leq, \Omega>$, где S/s_1 полностью определенный возможно недетерминированный наблюдаемый инициальный автомат с n состояниями, $n > 1$, Ω – множество всех полностью определенных детерминированных автоматов, определенных на том же входном алфавите, не более чем с n состояниями.

Под *адаптивной стратегией* при тестировании предъявленного (проверяемого) автомата из области неисправности понимается построение входной последовательности, в которой следующий входной символ, подаваемый на проверяемый автомат, вычисляется на основе выходных реакций этого автомата на предыдущие входные символы. Адаптивная стратегия называется *исчерпывающей* относительно модели неисправности FM, если для каждого $P/p_1 \in \Omega$ выходная последовательность на построенную входную последовательность содержится в множестве выходных реакций автомата-спецификации, если и только если P/p_1 есть редукция S/s_1. Соответственно построенная входная последовательность часто называется адаптивной *проверяющей* последовательностью, поскольку гарантировано проверяет только предъявленный к тестированию автомат. Следующие утверждения могут быть полезны при доказательстве того, что адаптивная стратегия является исчерпывающей относительно данной модели неисправностей.

Утверждение 1 [11]. Для полностью определенных наблюдаемых связных автоматов S/s_1 и P/p_1, автомат P/p_1 есть редукция автомата S/s_1, если и только если пересечение $P/p_1 \cap S/s_1$ есть полностью определенный автомат.

Утверждение 2. Пусть полностью определенный наблюдаемый связный автомат S/s_1 обладает разделяющей последовательностью. Если автомат P/p_1 есть редукция S/s_1, то разделяющая последовательность различает каждую пару состояний автомата P/p_1.

Утверждение 3 [15]. Пусть P/p_1 и S/s_1- полностью определенные наблюдаемые связные автоматы, причем автомат S/s_1 обладает разделяющей последовательностью и является ∂-связным. Автомат P/p_1 есть редукция S/s_1, если и только если P/p_1 изоморфен подавтомату S/s_1.

Утверждения 2 и 3 показывают, какой должна быть адаптивная стратегия. Проверяющая последовательность должна пройти по каждому переходу проверяемого автомата, и финальное состояние перехода необходимо проверить посредством разделяющей последовательности. Соответственно, процедура построения проверяющей последовательности разделяется на два этапа. На первом этапе проверяется, что предъявленный к проверке автомат имеет n состояний, и в каждом состоянии проверяемого автомата фиксируются реакция автомата на разделяющую последовательность и соответствующее состояние-преемник. На втором этапе проверяется каждый переход проверяемого автомата.

Пример 1. Рассмотрим автомат-спецификацию на рис. 1. Автомат обладает разделяющей последовательностью $i_2\, i_1$, и в табл. 1 представлены выходные реакции на эту последовательность в каждом состоянии.

Табл. 1. Выходные реакции на последовательность I_2I_1.

Table 1. Output responses to I_2I_1.

Состояния	Выходные реакции для разделяющей последовательности: i_2i_1
1	22
2	02, 20, 10
3	12

Отметим, что автомат на рис. 1 является ∂-связным, т.е. для любой пары различных состояний j и k существует ∂-передаточная последовательность α_{jk}: $\alpha_{12} = i_2$, $\alpha_{23} = i_1$, и $\alpha_{31} = i_1$.

3. Адаптивная стратегия построения проверяющей последовательности

В этом разделе мы кратко повторяем некоторые шаги из [8], которые служат основой при построении адаптивной проверяющей последовательности.

Вход. Автомат $S = (S, I, O, h_S, s_1)$ с n состояниями, разделяющая последовательность δ для автомата S, ∂-передаточные последовательности $\alpha_{ss'}$ для каждой пары различных состояний s и s', полностью определенный детерминированный проверяемый автомат P/p_1 не более чем с n состояниями, структура переходов которого не известна.

Выход. Сообщение 'P/p_1 является редукцией S/s_1' или 'P/p_1 не является редукцией S/s_1' и входная последовательность σ, которая отличает автомат P/p_1 от S/s_1 в последнем случае.

Как было сказано выше, процедура состоит из двух этапов. На первом шаге проверяется реакция на разделяющую последовательность δ в каждом

состоянии проверяемого автомата, т.е. устанавливается взаимно однозначное соответствие между состояниями автоматов P/p_1 и S/s_1, если P/p_1 есть редукция S/s_1 (утверждение 2). Выходные реакции на разделяющую последовательность и соответствующие δ-преемники сохраняются в множестве *Separable*.

Таким образом, множество *Separate* содержит тройки (s, ρ, s'), где s есть текущее состояние автомата S, $\rho = out_P(s, \delta)$ есть выходная реакция автомата P на разделяющую последовательность δ в состоянии, которое соответствует состоянию s и s' есть δ/ρ-преемник состояния s. Процедура построения множества *Separate* достаточно простая: разделяющая последовательность подается на проверяемый автомат, начиная с начального состояния, до тех пор, пока некоторая выходная реакция автомата не будет получена дважды. Если хотя бы на одном входном символе выходная реакция проверяемого автомата не содержится в ожидаемом множестве реакций, то выдается сообщение 'P/p_1 не является редукцией S/s_1' и входная последовательность, которая отличает автомат P/p_1 от S/s_1. Если на все входные символы получена ожидаемая выходная реакция, то по построенному отрезку проверяющей последовательности заполняется множество *Separate*. Пусть финальное состояние рассмотренного отрезка проверяющей последовательности соответствует состоянию s автомата S/s_1. Если в множестве *Separate* отсутствует реакция проверяемого автомата на разделяющую последовательность в некотором состоянии s', то на проверяемый автомат подается передаточная последовательность $\alpha_{ss'}$, и процесс построения передаточной последовательности продолжается с состояния s'. Выполнение первого шага заканчивается, когда выдано сообщение 'P/p_1 не является редукцией S/s_1', или множество *Separate* содержит реакцию проверяемого автомата на разделяющую последовательность в каждом состоянии. В последнем случае нами установлено взаимно однозначное соответствие между состояниями автоматов P/p_1 и S/s_1.

Пример 2. Пусть автомат S на рис. 1 есть автомат-спецификация; в качестве проверяемого автомата рассмотрим автомат P на рис. 2 с начальным состоянием a.

Последовательность $\delta = i_2\, i_1$ является разделяющей последовательностью для автомата S. В начальном состоянии проверяемого автомата мы получаем выходную реакцию 22 на поданную последовательность δ, и таким образом, начальное состояние a проверяемого автомата соответствует состоянию s_1 спецификации S. После повторной подачи последовательности δ будет получена выходная реакция 12, т.е. состояние 3 автомата S, достигнутое после трассы δ/12, соответствует состоянию c проверяемого автомата. Подаем δ еще раз и получаем уже полученную выходную реакцию 12, т.е. мы снова достигаем состояния, которое соответствует состоянию 3. Таким образом, мы можем внести два элемента (1, 22, 3), (3, 12, 3) в множество *Separate* и

отметить, что текущим состояние проверяемого автомата является состояние, соответствующее состоянию 3. После подачи $\alpha_{32} = i_2$ должно быть достигнуто состояние, соответствующее состоянию 2, и после подачи $\delta = i_2\, i_1$ дважды мы заключаем, что проверяемый автомат под действием последовательности δ переходит из состояния *b*, соответствующего состоянию 2, в состояние *a*, соответствующее состоянию 1. Поскольку *P* имеет не более трех состояний, то все его состояния идентифицированы посредством последовательности δ, и δ различает два любые различные состояния проверяемого автомата. Таким образом, *Separate* = {(1, 22, 3), (2, 10, 1), (3, 12, 3)}; в этом множестве в каждой тройке первый элемент обозначает текущее состояние, второй элемент есть выходная реакция в данном состоянии на последовательность δ, и последний элемент показывает δ-преемник текущего состояния. Иными словами, нами установлено взаимно однозначное соответствие между состояниями автоматов *S* и *P*: 1 и *a*, 2 и *b*, 3 и *c*.

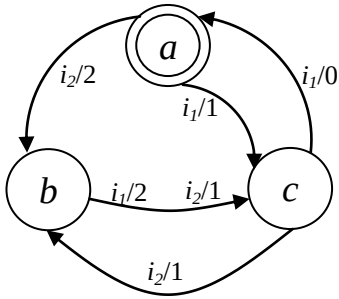

Рис. 2. Проверяемый автомат-реализация P.

Fig. 2. FSM P/p_1 under test.

На следующем этапе строится множество *Transition*. Это множество содержит четверки (*s, i, o, s'*), и его построение заканчивается, как только для каждой пары $(s, i) \in S \times I$ множество *Transition* содержит четверку с такими первыми элементами. Для достижения этой цели в каждом состоянии проверяемого автомата подается каждый входной символ, после которого подается разделяющая последовательность. Если в текущем состоянии проверены переходы по всем входным символам, то используется последовательность уже проверенных переходов из множества *Transition*, чтобы достичь состояния, в котором есть непроверенный переход. Как показано в [8], такая последовательность всегда существует в силу свойств автомата-спецификации. Если хотя бы на одном входном символе выходная реакция проверяемого автомата не содержится в ожидаемом множестве реакций, то выдается сообщение 'P/p_1 не является редукцией S/s_1' и входная последовательность, которая отличает автомат P/p_1 от S/s_1. Если на все входные символы получена ожидаемая выходная реакция, то вторая часть

проверяющей последовательности, построенной для проверки переходов проверяемого автомата, устанавливает взаимно однозначное соответствие между переходами проверяемого автомата и некоторым подавтоматом автомата-спецификации (утверждение 3), т.е. предъявленный автомат является редукцией спецификации.

Пример 3. В табл. 2 представлено множество, полученное на первом этапе построения адаптивной проверяющей последовательности.

Табл. 2. Множество Separate, построенное для автомата P.

Table 2. The set Separate for FSM P.

s	ρ	s'
a(1)	22	c(3)
b(2)	10	a(1)
c(3)	12	c(3)

Перед вторым этапом (проверки переходов) у нас уже есть проверяющая последовательность $\sigma = i_2i_1\ i_2i_1i_2i_1\ i_2\ i_2i_1i_2i_1$, которая заканчивается в состоянии c, соответствующем состоянию 3 автомата S.

На начальном шаге второго этапа множество *Transition* пусто, и существует непроверенный переход из состояния 3 под действием входного символа i_1. После подачи i_1 проверяемый автомат выдает ожидаемый входной символ 1, и после подачи δ мы получаем 12, что соответствует третьей строке табл. 2. Соответственно мы заключаем, что проверяемый автомат перешел в состояние c, соответствующее состоянию 3 автомата S, и заносим четверку (3, i_1, 1, 3) в множество *Transition*. В состоянии 3 есть еще один непроверенный переход под действием входного символа i_2. После подачи i_2 и разделяющей последовательности δ проверяемый автомат выдает ожидаемые выходные реакции 1 и 22, что соответствует третьей строке множества *Separate*, и мы добавляем четверку (3, i_2, 1, 1) в множество *Transition*. Аналогичным образом проверяются остальные переходы автомата P. В результате получается проверяющая последовательность $\sigma = i_2i_1i_2i_1i_2i_1i_2i_2i_1i_2i_1 + i_1i_2i_1\ i_2i_2i_1i_1i_2i_1i_1i_2i_2i_1i_2i_1i_2i_1i_2i_2i_1$, для которой выходная реакция проверяемого автомата содержится в множестве реакций автомата-спецификации на эту последовательность. Соответственно, мы можем заключить, что предъявленный для проверки автомат на рис. 2 есть редукция автомата-спецификации на рис. 1.

4. Использование (адаптивных) различающих тестовых примеров вместо разделяющей последовательности

Для адаптивной стратегии, описанной в разделе 3, существуют достаточно жесткие ограничения на автомат-спецификацию. Такой автомат должен обладать различающей последовательностью и быть d-связным. Известно [11], что не каждый автомат обладает этими свойствами; более того, длина таких

последовательностей (если существуют) может оказаться экспоненциальной относительно числа состояний автомата.

Поскольку мы используем адаптивную стратегию для построения проверяющей последовательности, то имеет смысл заменить разделяющую последовательность так называемым тестовым примером, который представляет адаптивный эксперимент с проверяемым автоматом [13]. Во-первых, известно, что различающий тестовый пример может существовать и при отсутствии разделяющей последовательности, и, во-вторых, длина такого тестового примера во многих случаях существенно меньше [9].

Для входного и выходного алфавитов I и O *тестовый пример* есть связный наблюдаемый инициальный автомат $P = (P, I, O, h_P, p_0)$, граф переходов которого ациклический и в каждом не тупиковым состоянии определены переходы только по одному входному символу со всеми возможными выходными символами. По определению, если $|I| > 1$, то тестовый пример является частичным автоматом. Длина тестового примера определяется как длина самого длинного пути из начального в тупиковое состояние. Вообще говоря, длиной тестового примера является длина самой длинной входной последовательности, которая подается на автомат в процессе адаптивного эксперимента (иногда ее называют высотой адаптивного эксперимента). Как обычно, при тестировании мы бы хотели использовать тестовые примеры минимальной длины. Если известно, что автомат-спецификация является полностью определенным наблюдаемым автоматом без слияний, т.е. для каждых входного символа i и выходного символа o непустые преемники двух различных состояний по входо-выходной паре io не совпадают, то длина различающего тестового примера (если таковой существует) является полиномиальной относительно числа состояний автомата, более точно имеет порядок $O(n^3)$ [16]. Класс автоматов без слияний достаточно большой, по крайней мере, он содержит достаточно много детерминированных автоматов, которые используются в различных приложениях [17].

Пусть S есть полностью определенный и наблюдаемый автомат с входным и выходным алфавитами I и O. Тестовый пример, определенный относительно этих алфавитов, называется *различающим*, если для каждой трассы, переводящей тестовый пример из начального в тупиковое состояние, справедливо, что данная трасса является трассой автомата S только в одном состоянии. Трасса, переводящая тестовый пример в тупиковое состояние, называется *полной* трассой тестового примера. Множество полных трасс тестового примера TC обозначается как $Complete(TC)$.

Пример 4. В качестве примера рассмотрим автомат-спецификацию на рис. 3 из [12] с начальным состоянием 1. Непосредственной проверкой можно убедиться, что автомат не имеет разделяющей последовательности; однако состояния автомата попарно $д$-достижимы и существует различающий тестовый пример (рис. 4). Тупиковые состояния тестового примера помечены соответствующими начальными состояниями. Поскольку проверяющая

последовательность строится адаптивно, то можно использовать различающий пример вместо разделяющей последовательности. Единственное различие с множеством *Separate* из табл. 2 будет в том, что вместо единственной различающей последовательности в проверяемом автомате мы будем фиксировать соответствующие идентификаторы состояний.

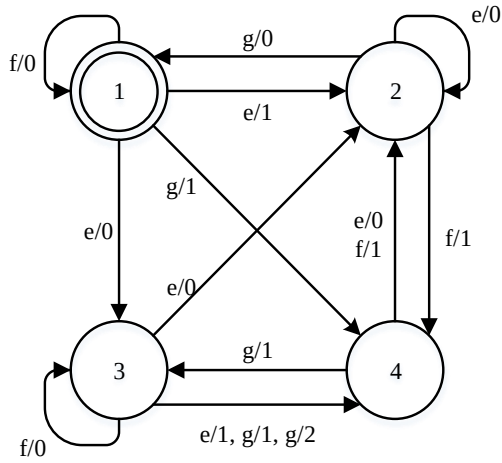

Рис. 3. Автомат-спецификация S.

Fig. 3. Specification FSM S.

Отметим также, что автомат на рис. 3 является *д*-связным; для каждой пары *j* и *k* различных состояний существует *д*-передаточная последовательность α_{jk}: $\alpha_{12} = ge$, $\alpha_{13} = gg$, $\alpha_{14} = g$, $\alpha_{21} = g$, $\alpha_{23} = fg$, $\alpha_{24} = f$, $\alpha_{31} = gfg$, $\alpha_{32} = gf$, $\alpha_{34} = g$, $\alpha_{41} = eg$, $\alpha_{42} = e$, $\alpha_{43} = g$.

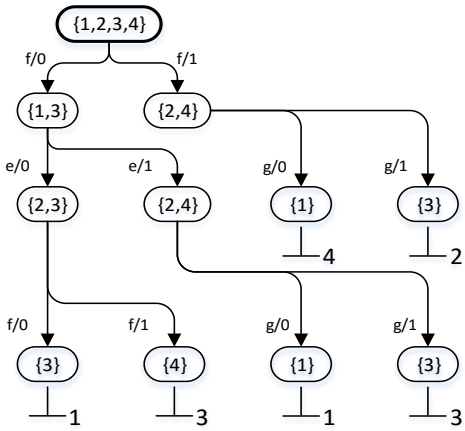

Рис. 4. Различающий тестовый пример T для автомата-спецификации S на рис. 3.

Fig. 4. A distinguishing test case T for the FSM S in Fig. 3.

Для полностью определенного наблюдаемого автомата S и состояния s входная последовательность α есть *идентификатор состояния* s, если α есть разделяющая последовательность для любой пары состояний (s, s'), $s' \neq s$. Последовательности *fef* и *feg* суть идентификаторы состояний 1 and 3 автомата S на рис. 3; для состояний 2 и 4 последовательность *fg* является идентификатором. Пусть автомат-спецификация S/s_1 имеет различающий тестовый пример TC, и P/p_1 есть детерминированный полностью определенный автомат с тем же числом состояний. Автомат P/p_1 называется TC-совместимым с S/s_1, если существует взаимно однозначное соответствие F: $S \rightarrow P$, такое что для каждого состояния $s \in S$ пересечение $Tr(S/s_1) \cap Tr(P/p_1) \cap Complete(TC)$ не пусто, если и только если $p = F(s)$.

Теорема 4. Пусть S/s_1 – полностью определенный наблюдаемый автомат-спецификация, для которого существует различающий тестовый пример TC, и P/p_1 полностью определенный детерминированный автомат, который является TC-совместимым с S/s_1. Для каждого состояния p автомата P/p_1 различающий тестовый пример TC содержит полную трассу α/β, которая является трассой в состоянии p; более того, α есть идентификатор состояния p в P/p_1.

Действительно, если существует взаимно однозначное соответствие между состояниями S/s_1 и P/p_1 согласно различающему тестовому примеру TC, то для любых двух состояний s и s', $s' \neq s$, существует начальный отрезок некоторой полной трассы в тестовом примере TC такой что выходные реакции на этот отрезок в состояниях $p = F(s)$ и $p' = F(s')$ различны. Поскольку TC является различающим тестовым примером для S/s_1, и P/p_1 есть полностью

определенный детерминированный автомата, это означает что входная проекция трассы α/β является идентификатором состояния *p*.

Согласно теореме 4, адаптивную проверяющую последовательность можно построить с использованием различающего тестового примера вместо разделяющей последовательности. Единственное различие состоит в том, что множество *Separate* вместо реакции на различающую последовательность содержит соответствующий идентификатор состояния, выходную реакцию и следующее состояние.

Пример 5. Пусть проверяемым автоматом является автомат P/p_1 на рис. 5 с начальным состоянием *A*, Непосредственной проверкой можно убедиться, что P/p_1 изоморфен подавтомату автомата S/s_1 (рис. 3), т.е. P/p_1 есть редукция автомата S/s_1. Поскольку при тестировании автомат P/p_1 является неизвестным, мы используем адаптивную стратегию для проверки, является ли P/p_1 редукцией S/s_1 (рис. 3). Начиная с начального состояния *A*, мы подаем различающий тестовый пример, подача которого в нашем случае заканчивается подачей входной последовательности *feg*, и эта последовательность является идентификатором состояния 1 автомата *S* (согласно различающему тестовому примеру на рис. 4). Проверяемый автомат выдает реакцию 010, и мы заключаем, что начальное состояние *A* автомата *P* соответствует состоянию 1 автомата *S*.

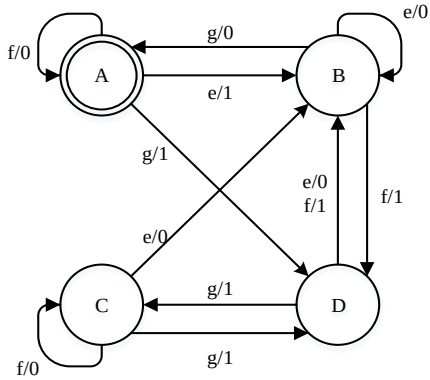

Рис. 5. Автомат P/p_1, для которого строится адаптивная проверяющая последовательность

Fig. 5. FSM P/p_1 under test

Для того чтобы определить, в какое состояние переходит проверяемый автомат под действием входной последовательности *feg*, мы подаем различающий тестовый пример еще раз. В нашем случае это означает, что на проверяемый автомат еще раз подается последовательность *feg*; в качестве выходной реакции мы получаем 010 и заключаем, что *feg* переводит

проверяемый автомат в состояние *A*, которое соответствует состоянию 1 автомата *S*.

Далее мы подаем *д*-передаточную последовательность $\alpha_{12} = ge$, проверяемый автомат выдает 10 и переходит в новое состояние *B*, которое соответствует состоянию 2 автомата *S*. Для того чтобы убедиться в этом, мы подаем идентификатор *fg* состояния 2 из тестового примера на рис. 4 и получаем выходную реакцию 11, т.е. состояние *B* автомата *P* соответствует состоянию 2 автомата *S*. Проверяемый автомат переходит в состояние *C* , которое должно соответствовать состоянию 3 автомата *S*. Мы подаем входную последовательность *fef*, получаем выходную реакцию 001, и проверяемый автомат переходит в новое состояние *D*, которое должно соответствовать состоянию 4 автомата *S*. Чтобы убедиться в этом, мы подаем идентификатор состояния *fg*, получаем выходную реакцию 10, и проверяемый автомат переходит в начальное состояние *A*. Подача входной последовательности *feg*, которая является идентификатором состояния *A*, завершает процедуру, результат которой приведен в табл. 3. Поскольку проверяемый автомат имеет не более четырех состояний, эта таблица содержит идентификатор для каждого состояния, ожидаемую выходную реакцию и следующее ожидаемое состояние.

Табл. 3. Множество Separate для автомата на рис. 3, соответствующее тестовому примеру на рис. 4.

Table. 3. The set Separate for the FSM in Fig. 3 according to the test case in Fig. 4

Текущее состояние	Идентификатор состояния	Выходная реакция	Следующее состояние
A (1)	*feg*	010	A (1)
B (2)	*fg*	11	C (3)
C (3)	*fef*	001	D (4)
D (4)	*fg*	10	A (1)

Множество *Transition* строится точно так же как при использовании разделяющей последовательности, т.е. после подачи каждого входного символа в каждом состоянии следующее состояние верифицируется посредством подачи соответствующего идентификатора. Если в текущем состоянии все входные символы проверены, то, используя уже построенную часть множества *Transition*, автомат переводится в состояние, в котором есть непроверенные входные символы. Для автомата-спецификации на рис. 3 и проверяемого автомата на рис. 5 мы получили адаптивную проверяющую последовательность длины 68, и поскольку были получены только ожидаемые выходные реакции, мы сделали вывод, что проверяемый автомат является редукцией спецификации.

Таким образом, справедливо следующее утверждение.

Теорема 5. Пусть автомат-спецификация S/s_1 является полностью определенным наблюдаемым автоматом, который обладает различающим тестовым примером и является *д*-связным. Тогда адаптивная стратегия,

направленная на построение множеств *Separate* и *Transition*, является исчерпывающей относительно модели неисправности $<S/s_1, \leq, \Omega>$, т.е. предъявленный автомат P/p_1 из области неисправности выдает только ожидаемые выходные реакции на построенную адаптивную проверяющую последовательность, если и только если P/p_1 есть редукция S/s_1.

5. Построение адаптивной проверяющей последовательности для частичных наблюдаемых автоматов

Предположим, что автомат-спецификация является частичным, возможно недетерминированным автоматом. В данной работе мы не обсуждаем построение адаптивной проверяющей последовательности в общем случае, тем не менее, для достаточно широкого класса автоматов такую последовательность можно построить на основе описанных выше результатов.

Пусть в автомате-спецификации S/s_1 существует полностью определенный наблюдаемый *д*-связный подавтомат S'/s_1, который обладает различающим тестовым примером. Согласно утверждению 2, полностью определенный автомат P'/p_1 будет редукцией автомата S'/s_1, если и только если P'/p_1 изоморфен подавтомату S'/s_1. Поскольку подавтомат S' обладает всеми необходимыми свойствами, то согласно теореме 5, для него можно построить исчерпывающую адаптивную стратегию.

Таким образом, множество *Separate* строится на основе подавтомата S'/s_1. После того как установлено взаимно однозначное соответствие между состояниями автомата-спецификации и проверяемого автомата, множество *Transition* строится так же как в разделе 3. Единственное отличие состоит в том, что в каждом состоянии проверяются только входные символы, по которым определены переходы в соответствующем состоянии.

Следует отметить, что если автомат-спецификация является частичным, то отношение редукции в модели неисправности заменяется на отношение квази-редукции [11]. В этом случае условие принадлежности выходной реакции проверяемого автомата множеству реакций спецификации должно выполняться только для входных последовательностей, на которых определено поведение спецификации. С другой стороны, такой подход не требует, чтобы спецификация была, вообще говоря, наблюдаемым автоматом; однако для разработки исчерпывающей адаптивной стратегии для частичных возможно ненаблюдаемых автоматов необходимы дальнейшие исследования.

6. Заключение

В данной статье предлагается адаптивная стратегия построения проверяющей последовательности для случая, когда автомат-спецификация и проверяемый автомат являются полностью определенными инициальными автоматами. При этом автомат-реализация является детерминированным автоматом, число

состояний которого не превышает число состояний автомата-спецификации, и отношением конформности является отношение редукции. Подобно детерминированным автоматам проверяющая последовательность строится при наличии определенных свойств у автомата-спецификации. Автомат-спецификация должен обладать различающим тестовым примером (должен существовать адаптивный различающий эксперимент) и быть детерминировано связным, т.е. каждое состояние должно быть детерминировано достижимо из любого другого состояния. Различающий тестовый пример может существовать и в том случае, когда в автомате-спецификации отсутствует разделяющая последовательность, и длина такого примера обычно значительно меньше длины разделяющей последовательности (если таковая существует). Более того, имеет смысл вместо *д*-передаточных последовательностей использовать адаптивные передаточные последовательности [18], что также расширит возможности использования адаптивной стратегии при построении проверяющей последовательности.

Acknowledgement. Данная работа выполнена при частичной поддержке РФФИ грантом No. 15-58-46013 СТ_а.

Acknowledgement. This work is partly supported by RFBR grant No. 15-58-46013 CT_a.

Список литературы

[1]. Kohavi Z. Switching and Finite Automata Theory, McGraw-Hill, New York, 1978.

[2]. Chow T.S.. “Testing software Design Modelled by Finite State Machines”, In IEEE Trans. Software Eng. Vol. 4 (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”, In Information & Software Technology, Vol. 52 (12), 2010, pp. 1286-1297.

[4]. Hennie F.C. “Fault-Detecting Experiments for Sequential Circuits”, In Proc. Fifth Ann. Symp. Switching Circuit Theory and Logical Design, 1964, pp. 95-110.

[5]. Petrenko A., Simão A., Yevtushenko N. “Generating Checking Sequences for Nondeterministic Finite State Machines”, In Proceedings of the ICST, 2012, pp. 310-319.

[6]. Жигулин М.В., Коломеец А.В., Кушик Н.Г., Шабалдин А.В. “Тестирование программной реализации протокола IRC на основе модели расширенного”, Известия Томского политехнического университета, Т. 318 (5), 2011, сс. 81-84.

[7]. Petrenko A., Simão A. “Generalizing the DS-Methods for Testing Non-Deterministic FSMs”, In Comput. J. 58(7), 2015, pp. 1656-1672.

[8]. Ермаков А.Д.“Синтез проверяющих последовательностей для недетерминированных автоматов относительно редукции”, Труды ИСП РАН, Т. 26(6), 2014, сс. 111-124.

[9]. Alur R., Courcoubetis C., Yannakakis M.. “Distinguishing tests for nondeterministic and probabilistic machines”, In Proceedings of the twenty-seventh annual ACM symposium on Theory of computing, 1995, pp. 363-372.

[10]. Kushik N., El-Fakih K., Yevtushenko N. ”Adaptive Homing and Distinguishing Experiments for Nondeterministic Finite State Machines”, In Lecture Notes in Computer Science, Vol. 8254, 2013, pp. 33-48.
[11]. Petrenko A., Yevtushenko N.. “Conformance Tests as Checking Experiments for Partial Nondeterministic FSM”, In Lecture Notes in Computer Science, Vol. 3997, 2005, pp. 118-133.
[12]. Petrenko A., Yevtushenko N.. “Adaptive Testing of Deterministic Implementations Specified by Nondeterministic FSMs”, In Lecture Notes in Computer Science, Vol. 7019, 2011, pp. 162-178.
[13]. Кушик Н.Г. Методы синтеза установочных и различающих экспериментов с недетерминированными автоматами. Диссертация на соискание ученой степени кандидата физико-математических наук, Томский Государственный университет, 2013.
[14]. Shabaldina N., El-Fakih K., Yevtushenko N. “Testing Nondeterministic Finite State Machines with Respect to the Separability Relation”, In Proceedings of Intern. Conf. on Testing Systems and Software (ICTSS/FATYES), 2007, pp. 305-318.
[15]. Ветрова М.В. Разработка алгоритмов синтеза и тестирования конечно-автоматных компенсаторов. Диссертация на соискание ученой степени технических наук, Томский Государственный университет, 2004.
[16]. Yevtushenko N., Kushik N. “Decreasing the length of Adaptive Distinguishing Experiments for Nondeterministic Merging-free Finite State Machines”, In Proceedings of IEEE East-West Design & Test Symposium, pp.338 – 341.
[17]. Güniçen C., Inan K., Türker U.C., Yenigün H. “The relation between preset distinguishing sequences and synchronizing sequences”, In Formal Aspects of Computing, Vol. 26 (6), 2014, pp. 1153–1167.
[18]. Kushik N., Yevtushenko N., Yenigun H. “Reducing the complexity of checking the existence and derivation of adaptive synchronizing experiments for nondeterministic FSMs”, In Proceedings of International Workshop on Domain Specific Model-based Approaches to Verification and Validation (AMARETTO’2016), 2016, pp. 83-90.

Deriving adaptive checking sequence for nondeterministic Finite State Machines

A.D. Ermakov <antonermak@inbox.ru>
N.V. Yevtushenko <yevtushenko@sibmail.com>
National Research Tomsk State University,
634050, Russia, Tomsk, Lenin Ave., 36.

Abstract. The derivation of checking sequences for Finite State Machines (FSMs) has a long history. There are many papers devoted to deriving a checking sequence that can distinguish a complete deterministic specification FSM from any non-equivalent FSM with the same number of states. To the best of our knowledge, for nondeterministic FSMs, the topic appeared only recently; the authors started with preset checking sequences for FSMs where the initial state is still known but the reset is very expensive. In this paper, a technique is proposed for deriving an adaptive checking sequence for a complete nondeterministic finite state machine with respect to the reduction relation. The main contribution of the paper is the use of a (adaptive) distinguishing test case instead of a separating sequence. Such a test case

is usually shorter that a separating sequence (when it exists) and can exist when there is no separating sequence. We also discuss the possibilities of using adaptive transfer sequences instead of deterministic transfer sequences that also allows to extend the set of FSMs for which the strategy can be used and reduce the length of a checking sequence. The application of a proposed strategy to partial possibly nondeterministic FSMs is briefly discussed.

Keywords: nondeterministic Finite State Machines (FSM); reduction relation; fault model; test derivation; adaptive checking sequences.

DOI: 10.15514/ISPRAS-2016-28(3)-8

For citation: A.D. Ermakov, N.V. Yevtushenko. Deriving adaptive checking sequence for nondeterministic Finite State Machines. *Trudy ISP RAN /Proc. ISP RAS*, 2016, vol. 28, issue 3, pp. 123-144 (in Russian). DOI: 10.15514/ISPRAS-2016-28(3)-8

References

[1]. Kohavi Z. Switching and Finite Automata Theory, McGraw-Hill, New York, 1978.
[2]. Chow T.S.. "Testing software Design Modelled by Finite State Machines", In IEEE Trans. Software Eng. Vol. 4 (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", In Information & Software Technology, Vol. 52 (12), 2010, pp. 1286-1297.
[4]. Hennie F.C. "Fault-Detecting Experiments for Sequential Circuits", In Proc. Fifth Ann. Symp. Switching Circuit Theory and Logical Design, 1964, pp. 95-110.
[5]. Petrenko A., Simão A., Yevtushenko N. "Generating Checking Sequences for Nondeterministic Finite State Machines", In Proceedings of the ICST, 2012, pp. 310-319.
[6]. Zhigulin M., Kolomeez A., Kushik N., Shabaldin A... "EFSM based testing a software implementation of IRC protocol", In Izvestia Tomskogo polytechnicheskogo instituta [Bulletin of the Tomsk Polytechnic University], 318 (5), 2011, pp. 81-84 (in Russian).
[7]. Petrenko A., Simão A.. "Generalizing the DS-Methods for Testing Non-Deterministic FSMs", In Comput. J. 58(7), 2015, pp. 1656-1672.
[8]. Ermakov A.. "Deriving checking sequences for nondeterministic FSMs", In Proceedings of the Institute for System Programming of RAS, Vol. 26, 2014, pp. 111-124 (in Russian).
[9]. Alur R., Courcoubetis C., Yannakakis M.. "Distinguishing tests for nondeterministic and probabilistic machines", In Proceedings of the twenty-seventh annual ACM symposium on Theory of computing, 1995, pp. 363-372.
[10]. Kushik N., El-Fakih K., Yevtushenko N. "Adaptive Homing and Distinguishing Experiments for Nondeterministic Finite State Machines", In Lecture Notes in Computer Science, Vol. 8254, 2013, pp. 33-48.
[11]. Petrenko A., Yevtushenko N.. "Conformance Tests as Checking Experiments for Partial Nondeterministic FSM", In Lecture Notes in Computer Science, Vol. 3997, 2005, pp. 118-133.
[12]. Petrenko A., Yevtushenko N.. "Adaptive Testing of Deterministic Implementations Specified by Nondeterministic FSMs", In Lecture Notes in Computer Science, Vol. 7019, 2011, pp. 162-178.

[13]. Kushik N.. Methods for deriving homing and distinguishing experiments for nondeterministic FSMs. PhD thesis, Tomsk State University, 2013 (in Russian).
[14]. Shabaldina N., El-Fakih K., Yevtushenko N. “Testing Nondeterministic Finite State Machines with Respect to the Separability Relation”, In Proceedings of Intern. Conf. on Testing Systems and Software (ICTSS/FATYES), 2007, pp. 305-318.
[15]. Vetrova M. FSM based methods for compensator design and testing. PhD thesis, Tomsk State University, 2004 (in Russian).
[16]. Yevtushenko N., Kushik N. Decreasing the length of Adaptive Distinguishing Experiments for Nondeterministic Merging-free Finite State Machines / // Proceedings of IEEE East-West Design & Test Symposium, pp.338 – 341.
[17]. Güniçen C., Inan K., Türker U.C., Yenigün H. “The relation between preset distinguishing sequences and synchronizing sequences”, In Formal Aspects of Computing, Vol. 26 (6), 2014, pp. 1153–1167.
[18]. Kushik N., Yevtushenko N., Yenigun H. “Reducing the complexity of checking the existence and derivation of adaptive synchronizing experiments for nondeterministic FSMs”, In Proceedings of International Workshop on Domain Specific Model-based Approaches to Verification and Validation (AMARETTO’2016), 2016, pp. 83-90.